A Faster Deterministic Approach To Magic Sinewave Zero Solutions

Don Lancaster
Synergetics, Box 809, Thatcher, AZ 85552

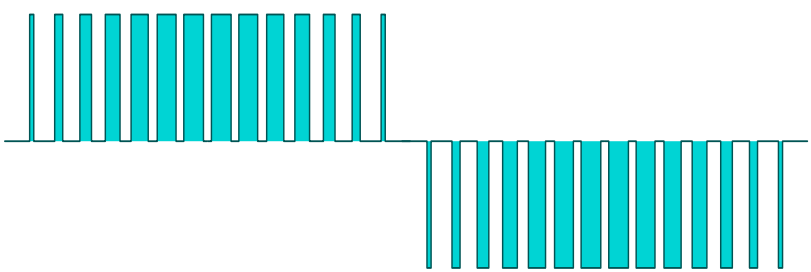
http://www.tinaja.com
don@tinaja.com
(928) 428-4073

Magic Sinewaves are a newly discovered class of mathematical functions that hold significant potential to dramatically improve the efficiency and power quality of solar energy synchronous inverters, electric hybrid automobiles, and industrial motor controls, among many others. An executive summary can be found **here**, a slideshow type intro presentation **here**, a development proposal **here** and detailed additional tutorials and design calculators **here**.

Major goals of such digital sinewave generation including offering the **maximum possible efficiency** by using the fewest of simplest possible switching transitions; offering the **lowest possible distortion** by zeroing out a maximum number of low harmonics that impact power quality, whine, vibration, and circulating currents; and by using **all digital techniques** that are extremely low end microprocessor and/or microcontroller friendly.

Magic sinewaves have two remarkable properties: **Any number of desired low harmonics can be forced exactly to zero** in theory, and to astonishingly low levels when quantized to 8-bit compatible levels. And magic sinewaves use the **absolute minimum possible and simplest energy-robbing transitions** to achieve such harmonic suppression.

A typical **magic sinewave** might look something like this…



Magic sinewaves are extremely exacting in their solutions. A typical equation set for a seven pulse per quadrant best efficiency **magic sinewave** might be…

```
cos( 1*p1s )-cos( 1*p1e )+...+cos( 1*p7s )-cos( 1*p7e ) = ampl*pi/4
cos( 3*p1s )-cos( 3*p1e )+...+cos( 3*p7s )-cos( 3*p7e ) = 0
cos( 5*p1s )-cos( 5*p1e )+...+cos( 5*p7s )-cos( 5*p7e ) = 0
cos( 7*p1s )-cos( 7*p1e )+...+cos( 7*p7s )-cos( 7*p7e ) = 0
cos( 9*p1s )-cos( 9*p1e )+...+cos( 9*p7s )-cos( 9*p7e ) = 0
cos(11*p1s)-cos(11*p1e)+...+cos(11*p7s)-cos(11*p7e) = 0
cos(13*p1s)-cos(13*p1e)+...+cos(13*p7s)-cos(13*p7e) = 0
cos(15*p1s)-cos(15*p1e)+...+cos(15*p7s)-cos(15*p7e) = 0
cos(17*p1s)-cos(17*p1e)+...+cos(17*p7s)-cos(17*p7e) = 0
cos(19*p1s)-cos(19*p1e)+...+cos(19*p7s)-cos(19*p7e) = 0
cos(21*p1s)-cos(21*p1e)+...+cos(21*p7s)-cos(21*p7e) = 0
cos(23*p1s)-cos(23*p1e)+...+cos(23*p7s)-cos(23*p7e) = 0
cos(25*p1s)-cos(25*p1e)+...+cos(25*p7s)-cos(25*p7e) = 0
cos(27*p1s)-cos(27*p1e)+...+cos(27*p7s)-cos(27*p7e) = 0
```

Power polynomials of this complexity are exceptionally unlikely to have a direct solution. Instead, **Newton's Method**, otherwise known as "**shake the box**" has proven to be an effective solution route.

In which a **good guess** is made based on a previously useful result or a nearby amplitude. One pulse edge, such as **p1s** is changed slightly to see if the distortion (the rms sum of the present nonzero harmonic values) increases or decreases. Increments in **p1s** are repeated till a minimum is found. The increment size is decreased and the process repeated until a desired precision level is reached. The process is then repeated for the other pulse edges. Continuing as needed.

An extensive set of JavaScript based interactive calculators is found **here**. At present, these calculators use a brute force iterative method that now requires repeated trig calculations to seek the harmonic distortion minimums. While quite effective and useful, the computing time becomes excessively long when many dozens or hundreds of harmonics are to be zeroed.

An alternate and fully deterministic method is proposed in this **GuruGram**. In which far fewer and much simpler calculations done in many fewer iterations can apparently be used to get the same results. At significantly faster speeds. **Only the method will be explored here**. Actual implementation and verification await additional funding and a more specific need.

The Approach

Assume we are near a true solution to the above equations. Assume further that we wish to optimize edge **p1s**. Assume further that any changes we make in **p1s** will be so small that a **linear slope approximation** can be made to any small change of any harmonic cosine.

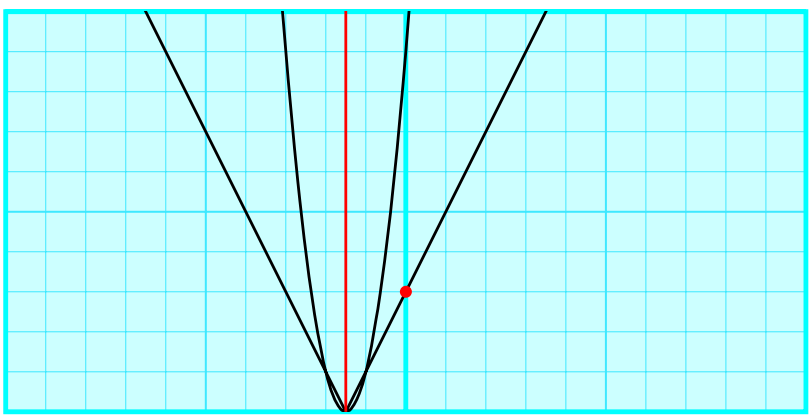
What we really have is a partial differential equation in **p1s**. And if we are only changing p1s, **everything else can be temporarily treated as a constant**. The above zeroed harmonic equations can be temporarily simplified to…

```
cos(3*p1s) + k3 = e3
cos(5*p1s) + k5 = e5
     . . . . .
cos(n*p1s) + kn = en
```

Where **kx** is the temporarily constant contribution of every other pulse edge **not** changing, and **ex** is the present nonzero error residue for each harmonic. What we have is a simple pile of linear equations of plain old form **y = mx + b**. Where **b** is our error and **m** is our slope. **The slope of a cosine is minus its sine, scaled by its harmonic number**.

If we were only interested in zeroing out **e3**, we could simply adjust the cosine to take the entire error out. Solving **y = mx + b** for **y = 0** gives us **x = -b/m**. Once again, m is our slope which here will be **-3sin(3*p1s)** and **b** is our error **e3**.

It is interesting to plot this relationship…



The straight lines represent an arbitrary absolute plot of **y = 2x + 3**. The red dot shows our **x = 0** intercept. The true parabola (in this case only) shows us the square of the error. Our red line predicts a zero error at **-b/m** or **-1.5**. Which the parabola verifies. An input correction of **x = -1.5** would thus zero out our error.

A similar straight line correction could usually zero out our **Magic Sinewave e3** error. Sadly, if we completely correct our **e3** error, some of the other harmonic errors will get better and some will get worse.

We thus want to seek out an adjustment to **p1s** that **minimizes the total error terms for all harmonics**.

Our harmonic distortion for this near solution will be the square root of…

$$e3^2 + e5^2 + ... + e27^2$$

We can treat these as a "pile of linear equations" at **x = 0** and ask what their value would be for a common shifted **x** value…

$$f(x) = (m3x + b3)^2 + (m5x + b5)^2 + ...$$

Expand the squares…

$$f(x) = m3^2x^2 + 2m3b3x + b3^2 + m5^2x^2 + 2m5b3x + b5^2 + ...$$

We seek a **minimum** of this function, which is the equivalent of making the **best** possible adjustment of **p1s**. A minimum (or a maximum) can be found by taking the first derivative and setting it to zero. The derivative is …

$$f(x)' = 2m3^2x + 2m3b3 + 2m5^2x + 2m5b5 + ... = 0$$

Rearrange and divide by two…

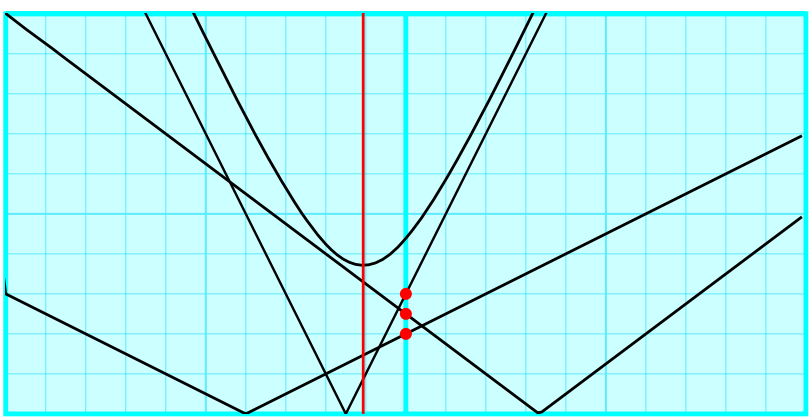
$$(m3^2 + m5^2 + ...)x = (m3b3 + m5b5 + ...)$$

And solve for x

Best pulse edge adjustment =

$$-(m3b3 + m5b5 + ...)/(m3^2 + m5^2 + ...)$$

Now, this seems waaaay too easy and waaay too good to be true. But a check plot seems to verify its validity…



You can get the checkplot routine as part of the **sourcecode** for this document. Here the parabola looking thingy is the actual rms distortion. And we see that the red predicted minimum (based on the above equation) does indeed point to our true minimum. While not a proof, the verification seems to work for random choices of slopes and errors.

There are two minor gotchas. **The routine blows up for zero degrees where all slopes are zero**. But a **magic sinewave** transition should never occur here. And other pulse edges should have at least one non-zero harmonic slope. And the assumption that the edge shift cosines can be linearly approximated may have slight errors for large harmonic values. But this should be easily cured by two or more passes through the code. If convergence is needed at all, it likely will be extremely fast.

Wait! It gets Even Better

Once a good **p1s** correction is found, the **e3** through **en** errors can be adjusted for residues. And you should be able to proceed **directly** to a **p1e** and higher adjustments. With an absolute minimum of trig or other complex calculations. And a bare minimum of iterations.

Potential calculation speedup is by one or two orders of magnitude.

Once again, this **GuruGram** is **only** a preliminary discovery. Considerable work remains to implement the actual fast zero **Magic Sinewave** calculation curve. Your **participation** is invited.

For Additional Assistance

Visit the many **Magic Sinewave** files at **http://www.tinaja.com/magsn01.asp**. Or else email **don@tinaja.com**. Or call **(928) 428-4073**.