About the ‘Bass Problem’

* There is no subsequent processing that generates ‘tells’ because of processing errors.

* My bass hearing changes by about 10-12dB unreliably.

* Bass isn’t just about 50Hz, and freq like that.   1st order EQ is needed, and the effects of a 50Hz 1st order eq are profound up to 200Hz or more.

* Too much 1st order bass can be totally repressive sounding.   It doesn’t sound like 2nd order ‘bass boost’.

* The first-cut test demo is done with my first EQ estimate, which also produces the most bass that can accommodate the most bass heavy recordings that I can find without obscuring other details.   This means that the least-bass-heavy recordings might have insufficient bass. *Do we need to add a ‘bass boost’ mode? (we won’t call it that .)*   Is my ‘heavy bass’ mode actually too light?

Immediately after decoding, before the LF specific EQ, the signal has (not exactly) 20-24dB too much bass.   The bass I the signal increases from 1kHz down to 20-50Hz.   (Think about it – 1st order EQ is linearly related to frequency, when going from 1kHz to 50Hz, that is 20X or about 16dB…)

ALL WORK IS BASED ON SOURCE CODE CONTROLLED SW.   WE CAN TRACK ANY CHANGE FOR TESTING.   (I have somewhere between 1000 to 2000 versions that I can access if needed – they go all the way back to approx 2014.)   Many versions didn’t even have FA capabilities, and JUST BARELY even DA.

EQ for the 23Mar morning test recordings.

My first Eqs always start out simple, then I keep on tweaking because of my misleading hearing. My first cut of EQ looks similar to what the demos are as of today: (translation from source code follows after the source snippet): THIS EQ IS POTENTIALLY GOOD (or more likely, fairly close to good). *Also, this design is minimized so that it reflects the realities of the cost/physical issues of HW design – esp the 1960s.* The anti-distortion modifications do make the SW implementation more complex, but the design is based upon simpler HW concepts.

(lfeq1diffb == 25Hz, lfeq1diff0 == 20Hz)

```
lfeq1[nlfeq1++].designfilt(1, (20.0 * lfeq1diffb) + lfeq1diff0, da::vm12dB);
lfeq1[nlfeq1++].designfilt(1, (20.0 * lfeq1diffb), da::v12dB);
lfeq1[nlfeq1++].designfilt(1, (20.0 * lfeq1diffb) - lfeq1diff0, da::vm12dB);

lfeq1[nlfeq1++].designfilt(1, (10.0 * lfeq1diffb) + lfeq1diff0, da::vm9dB);
lfeq1[nlfeq1++].designfilt(1, (10.0 * lfeq1diffb), da::v9dB);
lfeq1[nlfeq1++].designfilt(1, (10.0 * lfeq1diffb) - lfeq1diff0, da::vm9dB);

lfeq1[nlfeq1++].designfilt(1, (8.0 * lfeq1diffb) + lfeq1diff0, da::vm6dB);
lfeq1[nlfeq1++].designfilt(1, (8.0 * lfeq1diffb), da::v6dB);
lfeq1[nlfeq1++].designfilt(1, (8.0 * lfeq1diffb) - lfeq1diff0, da::vm6dB);

lfeq1[nlfeq1++].designfilt(1, (8.0 * lfeq1diffb) + lfeq1diff0, da::vm3dB);
lfeq1[nlfeq1++].designfilt(1, (8.0 * lfeq1diffb), da::v3dB);
lfeq1[nlfeq1++].designfilt(1, (8.0 * lfeq1diffb) - lfeq1diff0, da::vm3dB);

lfeq1[nlfeq1++].designfilt(1, (1.0 * lfeq1diffb) + lfeq1diff0, da::vm3dB);
lfeq1[nlfeq1++].designfilt(1, (1.0 * lfeq1diffb), da::v3dB);
lfeq1[nlfeq1++].designfilt(1, (1.0 * lfeq1diffb) - lfeq1diff0, da::vm3dB);
lfeq1[nlfeq1++].designfilt(1, 20, da::vm1p5dB);
lfeq1[nlfeq1++].designfilt(1, 20, da::vm1p5dB);
lfeq1[nlfeq1++].designfilt(1, 0.5 * lfeq1diffb, da::vm3dB);
lfeq1[nlfeq1++].designfilt(1, 10, da::vm1p5dB);
lfeq1[nlfeq1++].designfilt(1, 10, da::vm1p5dB);
lfeq1[nlfeq1++].designfilt(1, 0.25 * lfeq1diffb, da::vm3dB);
```

```
Translated:

500Hz, -12dB
250Hz, -9dB
200Hz, -6dB
200Hz, -3dB
25Hz, -3dB
2ea 20Hz, -1.5dB
12.5Hz, -3dB
2eq 10Hz, -1.5dB
6.25Hz, -3dB
```

There is a rationale to the 20Hz region EQ – it has strong effects into the 100Hz+ range, and the careful crafting in the 20Hz range is the only 'tweaking' that was done on this.
Note also the use of fewer dBs individual EQ as the freq decreases. Starting with -12dB at 500Hz, but then ending up at the 3dB rate.

With 1st order EQ, the small-dB filters work differently than one might think. Note that there was 9dB total EQ at 200Hz, and the same total dB at 250Hz. However, using smaller dBs more numbers of times creates a sharper filter. The goal below 250Hz is to accelerate the EQ so that vocals aren't buried into the mid-range.

The above is my first-cut intuitive version that I tend to start with. However, my changing hearing causes confusion. THE ABOVE IS EXACTLY THE LF EQ USED ON the 23 Mar morning test demos. (the intuition about understanding 1st order EQ behavior, and applying some engineering sense)

```
                    This kind of mess is what I usually end up with
       (source included just to show my confusion – not bothering with the translation)

This is the BAD STUFF:  Only here to show my frustration – and the extreme variability of my
hearing!!!   Dealing with EQ is easy for me, but my hearing misguides!!!
```

```
~~  lfeq1[nlfeq1++].designfilt(1, (40.0 * lfeq1diffb) + lfeq1diff0, da::vm1p5dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (40.0 * lfeq1diffb), da::v1p5dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (40.0 * lfeq1diffb) - lfeq1diff0, da::vm1p5dB);~~

~~  lfeq1[nlfeq1++].designfilt(1, (30.0 * lfeq1diffb) + lfeq1diff0, da::vm1p5dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (30.0 * lfeq1diffb), da::v1p5dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (30.0 * lfeq1diffb) - lfeq1diff0, da::vm1p5dB);~~

~~  lfeq1[nlfeq1++].designfilt(1, (20.0 * lfeq1diffb) + lfeq1diff0, da::vm6dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (20.0 * lfeq1diffb), da::v6dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (20.0 * lfeq1diffb) - lfeq1diff0, da::vm6dB);~~

~~  lfeq1[nlfeq1++].designfilt(1, (10.0 * lfeq1diffb) + lfeq1diff0, da::vm3dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (10.0 * lfeq1diffb), da::v3dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (10.0 * lfeq1diffb) - lfeq1diff0, da::vm3dB);~~

~~  lfeq1[nlfeq1++].designfilt(1, (10.0 * lfeq1diffb) + lfeq1diff0, da::vm3dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (10.0 * lfeq1diffb), da::v3dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (10.0 * lfeq1diffb) - lfeq1diff0, da::vm3dB);~~

~~  lfeq1[nlfeq1++].designfilt(1, (8.0 * lfeq1diffb) + lfeq1diff0, da::vm3dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (8.0 * lfeq1diffb), da::v3dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (8.0 * lfeq1diffb) - lfeq1diff0, da::vm3dB);~~

~~  lfeq1[nlfeq1++].designfilt(1, (8.0 * lfeq1diffb) + lfeq1diff0, da::vm3dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (8.0 * lfeq1diffb), da::v3dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (8.0 * lfeq1diffb) - lfeq1diff0, da::vm3dB);~~

~~  lfeq1[nlfeq1++].designfilt(1, (8.0 * lfeq1diffb) + lfeq1diff0, da::vm3dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (8.0 * lfeq1diffb), da::v3dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (8.0 * lfeq1diffb) - lfeq1diff0, da::vm3dB);~~

~~  lfeq1[nlfeq1++].designfilt(1, (4.0 * lfeq1diffb) + lfeq1diff0, da::vm3dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (4.0 * lfeq1diffb), da::v3dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (4.0 * lfeq1diffb) - lfeq1diff0, da::vm3dB);~~

~~  lfeq1[nlfeq1++].designfilt(1, (2.0 * lfeq1diffb) + lfeq1diff0, da::vm3dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (2.0 * lfeq1diffb), da::v3dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (2.0 * lfeq1diffb) - lfeq1diff0, da::vm3dB);~~

~~  lfeq1[nlfeq1++].designfilt(1, (1.0 * lfeq1diffb) + lfeq1diff0, da::vm1p5dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (1.0 * lfeq1diffb), da::v1p5dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (1.0 * lfeq1diffb) - lfeq1diff0, da::vm1p5dB);~~

~~  lfeq1[nlfeq1++].designfilt(1, (1.0 * lfeq1diffb) + lfeq1diff0, da::vm1p5dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (1.0 * lfeq1diffb), da::v1p5dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (1.0 * lfeq1diffb) - lfeq1diff0, da::vm1p5dB);~~

~~  lfeq1[nlfeq1++].designfilt(1, (0.5 * lfeq1diffb), da::vm1p5dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (0.5 * lfeq1diffb), da::vm1p5dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (0.5 * lfeq1diffb), da::vm1p5dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (0.25 * lfeq1diffb), da::vm1p5dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (0.25 * lfeq1diffb), da::vm1p5dB);~~
~~  lfeq1[nlfeq1++].designfilt(1, (0.25 * lfeq1diffb), da::vm1p5dB);~~
```

Here is my proposal

*** initial recordings made based on my original/initial intuition.**
*** get specific LF feedback from reviewers – too much, too little? Where, how?**
*** I'll deal with the phasing/ band joining issues – once I know the balance, I can deal with the phase issues. (*I can VERY VERY clearly hear 'tells' about such matters.)***

I promise to be patient, and I need good quality, responsible feedback. FA actually existing is a matter of existence proof – the goal is that the decoder ADEQUATELY recover the damage from the FA compression. Those helping me must at least have open minds that FA might exist.

Normally, I have kept peoples names away from the documents because I am not confident about the quality. Once we know what is going on, I intend to give credit attribution in any major document that describes the program. I have great respect for people, and feel great responsibility to people.

Once we determine the correct balance – and I have code that implements it, then that code will be 'golden' and used as a reference for/if when changes need to be made.

Important issues

We might find recordings that need more or less LF. I believe that I have some worst case recordings that are sensitive to too much bass. Some older recordings (e.g. 1970 Carpenters) are very sensitive to too much bass. *With the initial, intuitive EQ that we are starting with, the 1970 Carpenters (04 Reason To Believe – worst case) has tolerable amounts of bass, and the vocal is not terribly overwhelmed.*

What I hear in the current, first pass, test examples

Some recordings appear to be 'thin' sounding. Here is where our judgment comes in – **can the 'original EQ' be slightly adjusted to better accommodate the apparently 'thin' recordings, or perhaps does there need to be an 'EQ" submode for those 'thin' recordings?** It seems to me like +1.5dB at 200Hz level of EQ does save most recordings, but my hearing is so unreliable that I cannot be a good judge.

I am truly hoping that the 'thin' sound in some recordings is normal and acceptable, or maybe a simple and VERY EASY to understand EQ scheme can be added.

What I am trying to avoid

I am trying to avoid the mess that is shown in the 2nd, hacked EQ sequence. I want to keep the EQ as close as possible to the (IMO close-to-correct) 1st sequence.