Diffmeasure

for EEMBC Telecom Benchmarks

Signal to Noise Definition and Discussion White Paper

Rick Foos, Sergei Larin, Alan R. Weiss
EEMBC Certification Laboratories, LLC (ECL)
inquiries@ebenchmarks.com

August 15, 2002

Summary

`Diffmeasure` is a program, written in the C programming language, used to check the quality of output in the EEMBC (Embedded Microprocessor Benchmark Consortium) Telecommunications (Telecom) benchmark suite against a set of canonical verification files. The verification files were originally generated for EEMBC 1.0 on a Cadence simulator with floating point.

ECL has been updating EEMBC 1.0 to produce EEMBC 1.1, a much improved version of the same benchmark kernels, and as part of the EEMBC Version 2 overall effort we took a look at `Diffmeasure`. We did not like what we learned:

There are three fundamental problems with the old `Dffmeasure`:

1) The output was Noise to Signal, not Signal to Noise
2) The math equation was incorrect. It used the equation for Power with Amplitudes. As a result, the output units are not decibels, but *-Decibels/2*.
3) The scaling calculations were based on Peak Amplitude (Impulse Noise), rather than average amplitude (Random Noise).

4)

Introduction

What we are trying to do is to determine how precise were our integer calculations (possibly using only 16-bit representation) compared to a double precision floating point computations of the same algorithm. Let us begin by defining Signal to Noise ratio. First a formal definition of the Signal-to-Noise ratio (S/N) as presented at (http://searchnetworking.techtarget.com/sDefinition/0,,sid7_gci213018,00.html**):**

"In analog and digital communications, signal-to-noise ratio, often written S/N or SNR, is a measure of signal strength relative to background noise. The ratio is usually measured in decibels (dB).

If the incoming signal strength in microvolts is V_s, and the noise level, also in microvolts, is V_n, then the signal-to-noise ratio, S/N, in decibels is given by the formula

$$S/N = 20 \log_{10}(V_s/V_n)$$

If $V_s = V_n$, then S/N = 0. In this situation, the signal borders on unreadable, because the noise level severely competes with it. In digital communications, this will probably cause a reduction in data speed because of frequent errors that require the source (transmitting) computer or terminal to resend some packets of data.

Ideally, V_s is greater than V_n, so S/N is positive. As an example, suppose that V_s = 10.0 microvolts and V_n = 1.00 microvolt. Then

$$S/N = 20 \log_{10}(10.0) = 20.0 \text{ dB}$$

which results in the signal being clearly readable. If the signal is much weaker but still above the noise -- say 1.30 microvolts -- then

$$S/N = 20 \log_{10}(1.30) = 2.28 \text{ dB}$$

which is a marginal situation. There might be some reduction in data speed under these conditions.

If V_s is less than V_n, then S/N is negative. In this type of situation, reliable communication is generally not possible unless steps are taken to increase the signal level and/or decrease the noise level at the destination (receiving) computer or terminal."

Or from another source:

(http://www.its.bldrdoc.gov/fs-1037/dir-033/_4849.htm):

“Signal-to-noise ratio (SNR): The ratio of the amplitude of the desired **signal** to the amplitude of **noise** signals at a given point in **time**. [**JP1**]

Note 1: SNR is expressed as 20 times the logarithm of the amplitude ratio, or 10 times the logarithm of the **power** ratio.

Note 2: SNR is usually expressed in **dB** and in terms of peak values for **impulse noise** and root-mean-square values for **random noise**. In defining or specifying the SNR, both the signal and noise should be characterized, *e.g.,* peak-signal-to-peak-noise ratio, in order to avoid ambiguity.”

As one can see in this definition we are dealing with Voltage (i.e. amplitude) definition of the S/N. But if you look at definition of dB itself, you would see the following: (http://www.its.bldrdoc.gov/fs-1037/dir-010/_1468.htm)

“**dB:** *Abbreviation for* **decibel(s).** One tenth of the common logarithm of the ratio of relative powers, equal to 0.1 B (**bel**). *Note 1:* The decibel is the conventional relative **power** ratio, rather than the bel, for expressing relative powers because the decibel is smaller and therefore more convenient than the bel. The ratio in dB is given by

$$dB = 10\log_{10}\left(\frac{P_1}{P_2}\right),$$

where P_1 and P_2 are the actual powers. Power ratios may be expressed in terms of voltage and **impedance**, E and Z, or current and impedance, I and Z, since

$$P = I^2 Z = \frac{E^2}{Z}.$$

Thus dB is also given by

$$dB = 10\log_{10}\left(\frac{E_1^2/Z_1}{E_2^2/Z_2}\right) = 10\log_{10}\left(\frac{I_1^2 Z_1}{I_2^2 Z_2}\right).$$

If $Z_1 = Z_2$, these become

$$dB = 20\log_{10}\left(\frac{E_1}{E_2}\right) = 20\log_{10}\left(\frac{I_1}{I_2}\right).$$

Note 2: The dB is used rather than arithmetic ratios or percentages because when circuits are connected in **tandem**, expressions of power level, in dB, may be arithmetically added and subtracted. For example, in an **optical link**, if a known amount of optical power, in **dBm**, is launched into a fiber, and the losses, in dB, of each **component** (*e.g.* , connectors, splices, and lengths of fiber) are known, the overall link **loss** may be quickly calculated with simple addition and subtraction."

Implementation

As you have noticed in this case the definition is dealing with *Powers*, not Amplitudes. So, simply speaking, when dealing with signal amplitude, we should use

S/N=20*log_10(signal_amplitude/noise_amplitude) but….

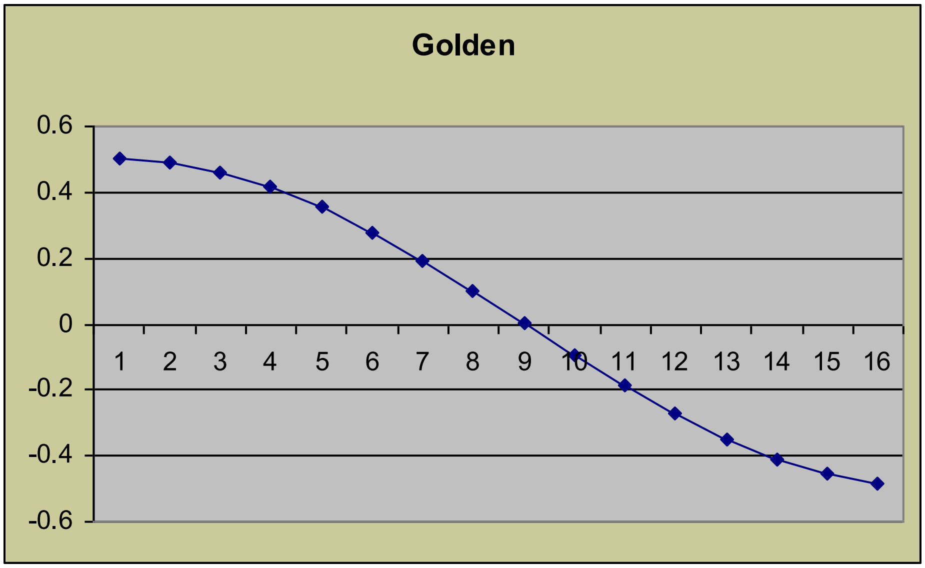
S/N=10*log_10(signal_power/noise_power);

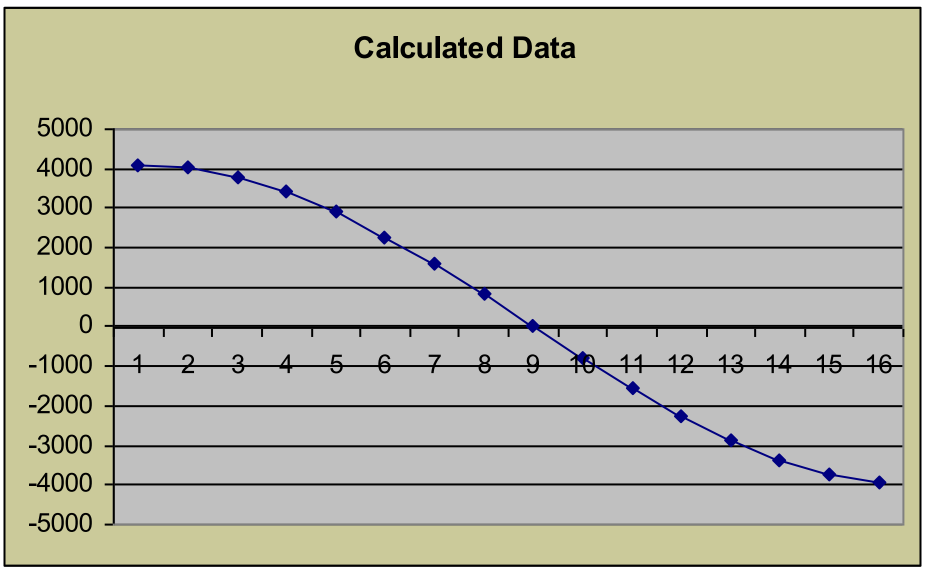
Once applied to EEMBC this definition should yield the following sequence of steps to determine S/N ratio in algorithm computation:

1) Given two sets of numbers: double golden[size_of_golden]; and short data[size_of_data]; Here is an example of one data set from Autocorr/Telecomm:

Golden	Data
0.499994	4095
0.490386	4017
0.461972	3784
0.415875	3406
0.353903	2899
0.278466	2281
0.192486	1576
0.099285	813
0.002455	20
-0.09428	-773
-0.1872	-1534
-0.27276	-2235
-0.34767	-2849
-0.40909	-3352
-0.45468	-3725
-0.48272	-3955

Note that in general case these numbers could be either COMPLEX or REAL. In case of complex numbers we would just use pairs on digits of the same type. In this example we are dealing with REAL numbers. The first check would be to ensure that we are dealing with the same type of data and the same number of elements. This condition definitely holds for our example, and if presented in graphical for it would look like this:


Golden
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16


Calculated Data
5000
4000
3000
2000
1000
0
-1000
-2000
-3000
-4000
-5000
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

2) To begin comparison we need to scale the calculated data so they are presented in comparable *amplitude*. To do that we calculate average of both data sets and use their ratio for scaling (note that for actual algorithm usage of SUM is the same for the equal number of points):

	Golden	Data	
	0.499994	4095	
	0.490386	4017	
	0.461972	3784	
	0.415875	3406	
	0.353903	2899	
	0.278466	2281	
	0.192486	1576	
	0.099285	813	
	0.002455	20	
	-0.09428	-773	
	-0.1872	-1534	
	-0.27276	-2235	
	-0.34767	-2849	
	-0.40909	-3352	
	-0.45468	-3725	
	-0.48272	-3955	Ratio = Golden/Data
Sum	0.546424	4468	0.000122297
Ave	0.034152	279.25	0.000122297

3) Now the next step is to determine Average Signal Power and Average Noise Power. Yes, we are using ***powers***, not amplitude, so beware! Average signal power is determined by accumulating the following:

Ave_Signal_Power =
(SUM_j_0_to_size_of_golden(Real_golden[j]^2 + Imaginary_golden[j] ^2))/size_of_golden;

and Average Noise Power is:
Ave_Noise_Power =
(SUM_j_0_to_size_of_golden(Real_error[j]^2 + Imaginary_error[j] ^2))/size_of_golden;

Where:

Real_error[j] = Real_golden[j] – (Real_data[j] * Scaling_ratio);
Imaginary _error[j] = Imaginary _golden[j] – (Imaginary _data[j] * Scaling_ratio);

So in our example it would look something like this:

Golden	Golden^2	Data	Scaled Data	Error(Golden-Scaled)	Error^2
0.499994	0.249994	4095	0.500807135	-0.000813135	6.6119E-07
0.490386	0.240478429	4017	0.491267952	-0.000881952	7.7784E-07
0.461972	0.213418129	3784	0.462772698	-0.000800698	6.4112E-07
0.415875	0.172952016	3406	0.416544347	-0.000669347	4.4803E-07
0.353903	0.125247333	2899	0.354539654	-0.000636654	4.0533E-07
0.278466	0.077543313	2281	0.27895997	-0.00049397	2.4401E-07
0.192486	0.03705086	1576	0.192740426	-0.000254426	6.4733E-08
0.099285	0.009857511	813	0.099427644	-0.000142644	2.0347E-08
0.002455	6.02703E-06	20	0.002445944	9.05551E-06	8.2002E-11
-0.09428	0.00888853	-773	-0.094535755	0.000256755	6.5923E-08
-0.1872	0.035044589	-1534	-0.187603943	0.000401943	1.6156E-07
-0.27276	0.074395836	-2235	-0.273334297	0.000578297	3.3443E-07
-0.34767	0.120874429	-2849	-0.348424793	0.000754793	5.6971E-07
-0.40909	0.167352992	-3352	-0.409940297	0.000852297	7.2641E-07
-0.45468	0.206733902	-3725	-0.455557162	0.000877162	7.6941E-07
-0.48272	0.233021495	-3955	-0.483685524	0.000962524	9.2645E-07
Signal Power	1.97285939			Noise Power	6.8166E-06

4) Finally we can use the original formula dB = 10*log_10(Signal_Power/Noise_Power) to get the desired measurement:

S/N = 10*log_10(1.97285939/6.8166E-06) = 54.6153 dB

Since it is S/N the large positive number means low noise, high signal == good result. In case when Noise_Power == 0, we can say that signals are identical (+Inf dB).

Now what happens if there was a small error in the result data set:

Golden	Golden^2	Data	Scaled Data	Error(Golden-Scaled)	Error^2	
0.499994	0.249994	4000	0.489188899	0.010805101	0.00011675	
0.490386	0.240478429	4000	0.489188899	0.001197101	1.4331E-06	
0.461972	0.213418129	3770	0.461060537	0.000911463	8.3076E-07	
0.415875	0.172952016	3406	0.416544347	-0.000669347	4.4803E-07	
0.353903	0.125247333	2899	0.354539654	-0.000636654	4.0533E-07	
0.278466	0.077543313	2281	0.27895997	-0.00049397	2.4401E-07	
0.192486	0.03705086	1576	0.192740426	-0.000254426	6.4733E-08	
0.099285	0.009857511	813	0.099427644	-0.000142644	2.0347E-08	
0.002455	6.02703E-06	20	0.002445944	9.05551E-06	8.2002E-11	
-0.09428	0.00888853	-773	-0.094535755	0.000256755	6.5923E-08	
-0.1872	0.035044589	-1534	-0.187603943	0.000401943	1.6156E-07	
-0.27276	0.074395836	-2235	-0.273334297	0.000578297	3.3443E-07	
-0.34767	0.120874429	-2849	-0.348424793	0.000754793	5.6971E-07	
-0.40909	0.167352992	-3300	-0.403580842	-0.005507158	3.0329E-05	
-0.45468	0.206733902	-3700	-0.452499731	-0.002180269	4.7536E-06	
-0.48272	0.233021495	-3900	-0.476959176	-0.005763824	3.3222E-05	**S/N dB**
Signal Power	1.97285939			Noise Power	0.00018963	**40.17184**

And now what about a large error (data loss of a kind):

Golden	Golden^2	Data	Scaled Data	Error(Golden-Scaled)	Error^2	
0.499994	0.249994	0	0	0.499994	0.249994	
0.490386	0.240478429	0	0	0.490386	0.24047843	
0.461972	0.213418129	0	0	0.461972	0.21341813	
0.415875	0.172952016	3406	0.416544347	-0.000669347	4.4803E-07	
0.353903	0.125247333	2899	0.354539654	-0.000636654	4.0533E-07	
0.278466	0.077543313	2281	0.27895997	-0.00049397	2.4401E-07	
0.192486	0.03705086	1576	0.192740426	-0.000254426	6.4733E-08	
0.099285	0.009857511	813	0.099427644	-0.000142644	2.0347E-08	
0.002455	6.02703E-06	20	0.002445944	9.05551E-06	8.2002E-11	
-0.09428	0.00888853	-773	-0.094535755	0.000256755	6.5923E-08	
-0.1872	0.035044589	-1534	-0.187603943	0.000401943	1.6156E-07	
-0.27276	0.074395836	-2235	-0.273334297	0.000578297	3.3443E-07	
-0.34767	0.120874429	-2849	-0.348424793	0.000754793	5.6971E-07	
-0.40909	0.167352992	0	0	-0.409088	0.16735299	
-0.45468	0.206733902	0	0	-0.45468	0.2067339	
-0.48272	0.233021495	0	0	-0.482723	0.23302149	**S/N dB**
Signal Power	1.97285939			Noise Power	1.31100126	**1.77493**

Described above is the algorithm used for both verify (built-in) and diffmeasure (stand-alone) utilities used in EEMBC benchmarks. The actual dB ratings are compared to a similar results obtained on x86 platform using out-of-the-box benchmarking method.

Appendix

Up till now a different methodology was used to measure difference between golden and real data sets. The algorithm was as following:

1) Find the maximum of absolute values of amplitude of golden and data, then use it as a scale factor:

Golden	Data	ABS(Golden)	ABS(Data)	
0.499994	4095	0.499994	4095	
0.490386	4017	0.490386	4017	
0.461972	3784	0.461972	3784	
0.415875	3406	0.415875	3406	
0.353903	2899	0.353903	2899	
0.278466	2281	0.278466	2281	
0.192486	1576	0.192486	1576	
0.099285	813	0.099285	813	
0.002455	20	0.002455	20	
-0.094279	-773	0.094279	773	
-0.187202	-1534	0.187202	1534	
-0.272756	-2235	0.272756	2235	
-0.34767	-2849	0.34767	2849	
-0.409088	-3352	0.409088	3352	
-0.45468	-3725	0.45468	3725	
-0.482723	-3955	0.482723	3955	Ratio = MA Data/MA Golden
	MaxAmplitude	0.499994	4095	8190.098281

2) Scale Golden data set, and use it to find error:

Golden	Scaled Golden	ABS(Scaled Golden)	Data	ABS(Error(Scaled-Data))	
0.499994	4095	4095	4095	0	
0.490386	4016.309536	4016.309536	4017	0.690464	
0.461972	3783.596083	3783.596083	3784	0.403917	
0.415875	3406.057123	3406.057123	3406	0.057123	
0.353903	2898.500352	2898.500352	2899	0.499648	
0.278466	2280.663908	2280.663908	2281	0.336092	
0.192486	1576.479258	1576.479258	1576	0.479258	
0.099285	813.1539078	813.1539078	813	0.153908	
0.002455	20.10669128	20.10669128	20	0.106691	
-0.094279	-772.1542759	772.1542759	-773	0.845724	
-0.187202	-1533.202778	1533.202778	-1534	0.797222	
-0.272756	-2233.898447	2233.898447	-2235	1.101553	
-0.34767	-2847.451469	2847.451469	-2849	1.548531	
-0.409088	-3350.470926	3350.470926	-3352	1.529074	
-0.45468	-3723.873886	3723.873886	-3725	1.126114	
-0.482723	-3953.548813	3953.548813	-3955	1.451187	N/S
	Signal Sum	41304.46745	Error Sum	11.12651	-35.6964

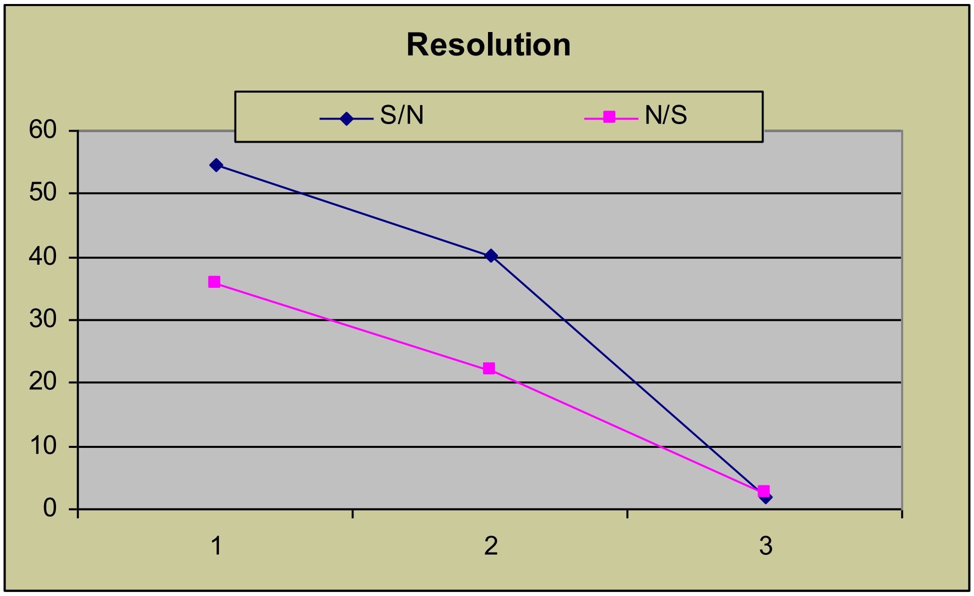
3) Use the formula dB = 10*log_10(Noise_Sum/Signal_Sum) to get the desired measurement. So in this case we are actually calculating Noise to Signal, so smaller (large negative) numbers are better.

Let us now consider the same error example used earlier:

Golden	Scaled Golden	ABS(Scaled Golden)	Data	ABS(Error(Scaled-Data))	
0.499994	4095	4095	4000	95	
0.490386	4016.309536	4016.309536	4000	16.30954	
0.461972	3783.596083	3783.596083	3770	13.59608	
0.415875	3406.057123	3406.057123	3406	0.057123	
0.353903	2898.500352	2898.500352	2899	0.499648	
0.278466	2280.663908	2280.663908	2281	0.336092	
0.192486	1576.479258	1576.479258	1576	0.479258	
0.099285	813.1539078	813.1539078	813	0.153908	
0.002455	20.10669128	20.10669128	20	0.106691	
-0.094279	-772.1542759	772.1542759	-773	0.845724	
-0.187202	-1533.202778	1533.202778	-1534	0.797222	
-0.272756	-2233.898447	2233.898447	-2235	1.101553	
-0.34767	-2847.451469	2847.451469	-2849	1.548531	
-0.409088	-3350.470926	3350.470926	-3300	50.47093	
-0.45468	-3723.873886	3723.873886	-3700	23.87389	
-0.482723	-3953.548813	3953.548813	-3900	53.54881	N/S
	Signal Sum	41304.46745	Error Sum	258.725	-22.0316

Golden	Scaled Golden	ABS(Scaled Golden)	Data	ABS(Error(Scaled-Data))	
0.499994	4095	4095	0	4095	
0.490386	4016.309536	4016.309536	0	4016.31	
0.461972	3783.596083	3783.596083	0	3783.596	
0.415875	3406.057123	3406.057123	3406	0.057123	
0.353903	2898.500352	2898.500352	2899	0.499648	
0.278466	2280.663908	2280.663908	2281	0.336092	
0.192486	1576.479258	1576.479258	1576	0.479258	
0.099285	813.1539078	813.1539078	813	0.153908	
0.002455	20.10669128	20.10669128	20	0.106691	
-0.094279	-772.1542759	772.1542759	-773	0.845724	
-0.187202	-1533.202778	1533.202778	-1534	0.797222	
-0.272756	-2233.898447	2233.898447	-2235	1.101553	
-0.34767	-2847.451469	2847.451469	-2849	1.548531	
-0.409088	-3350.470926	3350.470926	0	3350.471	
-0.45468	-3723.873886	3723.873886	0	3723.874	
-0.482723	-3953.548813	3953.548813	0	3953.549	N/S
	Signal Sum	41304.46745	Error Sum	22928.72	-2.55617

To finish up the discussion let us consider the “resolution” of both methods – difference in response given the same stimulus (absolute values are used for comparison):



The X axis is the three cases considered above – 1 is "correct" output, 2 – "minor" error, 3 – "serious" error.

Let us also briefly address the matter of measurement units *previously* used for diffmeasure computation. For this example assume S and N are calculated using *amplitude* of the signal. By definition:

x = 20 log_10(S/N) - x is CORRECT and it is measured in dB, then
y = 10 log_10(N/S) - y is what we have been using in "undefined" measurement units;

Now:

x/2 = 10 (log_10(S) - log_10(N));
y = 10 (log_10(N) - log_10(S)) = - 10(log_10(S) - log_10(N));

so:

x/2 = -y; As we have stated above x is measured in dB. Now:
y = -x/2 => -dB/2. ...and this is what we have been using as measurement units.

NOW BEWARE! This is NOT conversion formula from old scores to new ones BECAUSE we have stopped using S and N computation through amplitude and are using power instead. This derivation is presented for measurement units ONLY.