ANALOG DEVICES

TNJ-080

アナログ電子回路技術ノート

ADC の FFT 結果から 1Hz ノイズ電力密度を得る計算方法

FFT 結果から 1Hz ノイズ絶対値を求めるには手順や係数がいろいろある

著者: 石井 聡

はじめに

「AD 変換データから 1Hz ノイズ電力密度の絶対値をどうやって得るのか」という話題に前回の技術ノートから突入しています。

前回の技術ノートでは、1Hz ノイズ電力密度を測定するための絶対基準器となるノイズ発生回路を作り、そのレベルを測定してみました。また単なる測定のみに依らずに、その 1Hz 電力密度レベルを「パーセバルの定理（Parseval's theorem）」という理論ネタを用いて、理論的に導出してみました。そして測定結果と理論値が 0.1dB の誤差で一致したことを示しました。

それでも前回の技術ノートはいわゆる「実験的」な 1Hz ノイズ電力密度を求めるアプローチでした。今回の技術ノートでは AD 変換データを FFT（Fast Fourier Transform; 高速フーリエ変換）した結果を用いて、1Hz ノイズ電力密度絶対値レベルを直接計算する方法を示してみたいと思います。

1Hz 電力密度を測定できるベクトル・シグナル・アナライザ（FFT アナライザ）89410A が元気になった！

さて、私は 89410A というベクトル・シグナル・アナライザを前職で無線機器関連の設計開発用に使っており、アナログ・デバイセズ入社後すぐに「89410A は 1Hz ノイズ電力密度も測定できる名機なのだ」という噂をききつけ、どうしても欲しくなり、ヤフオクでけっこうな金額（笑）で、落札しました。

89410A はいわゆる FFT アナライザで、このところの技術ノートで説明している「1Hz ノイズ電力密度」を測定できる機能を持っています。この技術ノートで紹介するような計算手順を内部で行っているものです。

落札した 89410A は運よくきちんと動作しましたが、PHS 製造試験設備だったようで、CRT がだいぶ焼き付いていました。

① ある日一念発起し、CRT（SONY OEM 品が使用されていたので、オリジナル SONY ブランドのものを）をアメリカから取り寄せ、臓物交換手術を行いました。Service Manual が入手（ダウンロード）できていたので、交換自体に不安はありませんでした。

② 以降は元気に動作していましたが、2013 年のとある日、「Calibration Failure – Channel 1 Trigger」なるエラーが出て、自動校正ができない状態になりました。Service Manual からエラー状態を調べ、怪しそうなモジュールを eBay で入手し交換しても直らず…。

③ 結局ギブアップで、TNJ-074 でもご紹介した修理業者さんに依頼しました。「A35 モジュールが NG だったので交換しました」とのことでしたが、14 万円もかかってしまいました（技術料を買うわけですから仕方ないですね）。A35 モジュールがおかしいとは、②で修理チャレンジしたときには気が付かず…。なお当時エラー状態を調べたときに撮影した Diagnostics（自己診断機能）モードの画面に「A35 がおかしいようだ」と思わせるメッセージが出ているのを、今回の修理時に発見しました…（涙）。

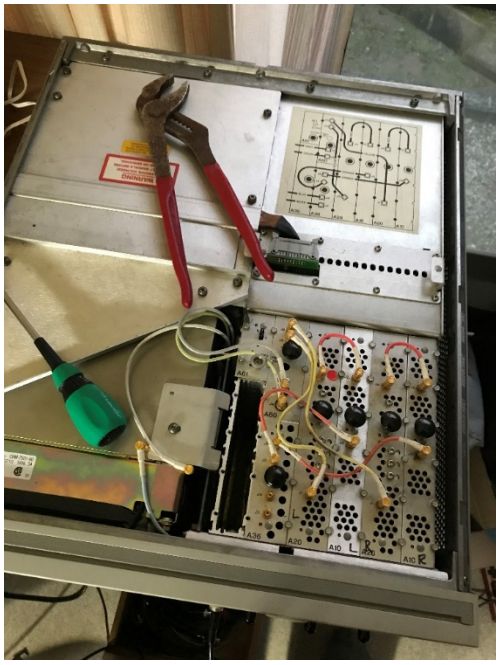

図 1. 89410A のケースを開けてみた。真ん中にみえる抜けた部分が A35 モジュールのスロット

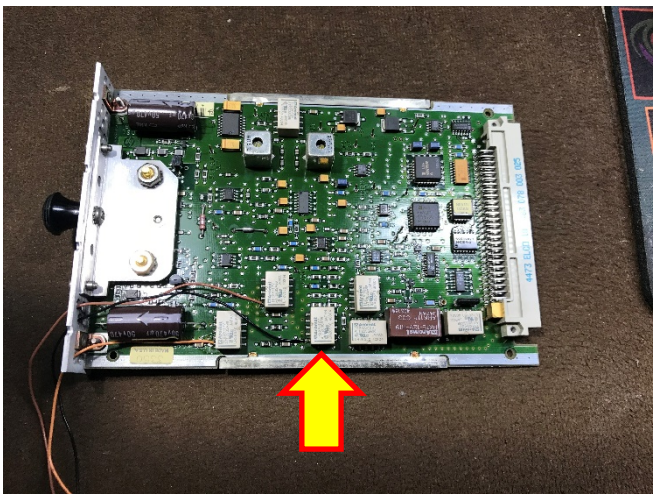

図 2. A35 モジュールのようす。中央下のリレーが故障（固着）していた

Rev. 0

アナログ・デバイセズ株式会社

本　　社／〒105-6891　東京都港区海岸 1-16-1　ニューピア竹芝サウスタワービル
電話 03（5402）8200

大阪営業所／〒532-0003　大阪府大阪市淀川区宮原 3-5-36　新大阪トラストタワー
電話 06（6350）6868

④ アナログ技術セミナー2020 の資料の用意をしているなか、またもや同じエラー「Calibration Failure – Channel 1 Trigger」が出てしまいました！

⑤ お金も時間のないなか、「修理はアナログ技術セミナー終了後か、リタイヤ後（笑）か！」とか思いつつ（ダメなら業者さんにまたお願いですが…）、89410A を Diagnostics モードにして、不具合症状と Service Manual の記載を延々と見比べて、問題点の「特定までだけ」をしていました。

⑥ Diagnostics で、どうやら今回も A35 モジュールがおかしそうだと特定できました。前回撮影してあったエラー画面の写真にも A35 モジュールが正しく動作していないと表示されていたことに、このとき気づきました（涙）。前回の故障のときは気がつかなかったのです。

⑦ しかし Service Manual には回路図は当然なく、信号の流れと回路ブロック（だいぶ端折りあり）だけという簡潔な記載があるのみでした。とはいえ問題は A35 ブロックの 80dB アッテネータ切り替え回路と特定できていました（10dB ステップなので 3 ビットに相当）。そうするとリレー3 個とその駆動回路、もしくは周辺のアッテネータ抵抗が怪しいと目星をつけることができていました。

⑧ アナログ技術セミナー2020 も終わった週末、ケースをあけてみました。このようすを図 1 に示します。この写真では A35 モジュールが取り外されています。図 2 は A35 モジュールです。この技術ノートに掲載するつもりもなかったので、適当かつ曲がった写真撮影です…（汗）。

⑨ 図 2 の A35 モジュールからはリード線が 3 本引き出されています。これは 3 個のリレーそれぞれの駆動ラインで、故障の原因が、リレー駆動回路（MC58-0134 というトランジスタ・アレイらしい IC。Web でサーチしてもデータシートを発見できず詳細不明）か、リレー自体なのかを切り分ける試験用の引き出しです。

⑩ この状態で A35 モジュールを本体に差し込み、89410A を Diagnostic モードにして、80dB アッテネータの切り替えを確認しました。これら駆動ラインはすべてきちんと H/L で動作しています。これでリレー自体が問題と特定できました。また H/L のシーケンスと減衰レベル変化から 3 個のリレーのうち、どのリレーが NG かも判定できました（図 2 に矢印で示してあります。40dB 切り替えリレーでした）。

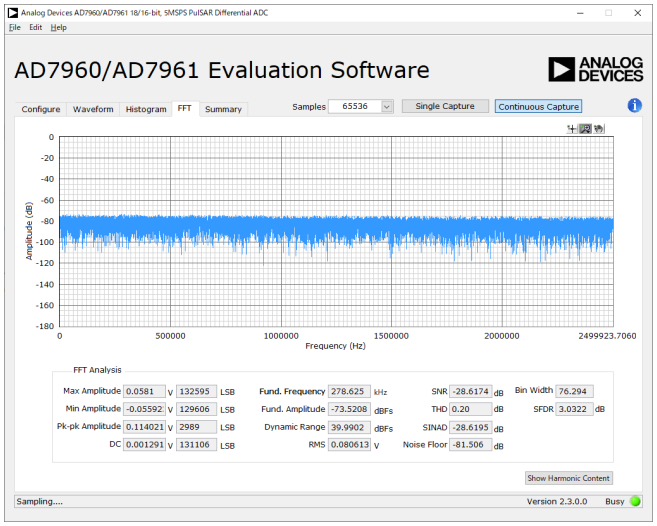

図 3. −78.4dBm/Hz の PRBS-16 を加えたときの FFT スペクトル（これでは 1Hz 電力密度は分からない）

⑪ 89410A の電源をオフにして A35 モジュールを本体から取り出し、リレーをテスタで当たってみると、問題のリレーのみ接点の短絡状態が異なっています。つまりリレーのカンチ・レバーの固着ということです。

⑫ 当該リレーの型番でサーチしてみると、生産中止ではありますが海外にいくつか在庫がありました。でもなんと！津市にあるパーツ販売店にも在庫があり、早速注文！

⑬ リレーが到着した日、A35 モジュールをホット・プレートで加熱し、リード端子を太い銅線でブリッジし 2 本のはんだごてで取り外し、基板パッドの残留はんだをソルダ・ウイックで清掃し、新しいリレーに交換しました。89410A に戻して電源を入れてみると…。おお！(^o^)

ということで、前回は 140,000 円程度かかった修理費用が、送料込み 1,000 円程度で済みました（バンザイ！）。

ADI の ADC 評価用ソフトウェア（AD7960）の表示では

評価ソフトウェアの FFT 機能だけでは 1Hz 電力密度は分からない

実験してみる ADC は、今回も AD7960 を使用してみます（実際は評価ボード EVAL-AD7960FMCZ を用います）。図 3 は、AD7960 に前回示した−78.4dBm/Hz の PRBS-16 疑似ノイズを加え、評価用ソフトウェアをもちいて 65536 サンプルだけキャプチャし、ADC 評価用ソフトウェアの FFT 機能を用いてスペクトル表示させたものです。

なお前回の TNJ-079 [1]で示したように、サンプリング動作により生じる折り返しの問題は生じない条件になっています（とくに同技術ノートの「AD 変換での折り返しの影響は？」の節と図 4 をご参照ください）。この詳しい話題は TNJ-077 [2]をご覧ください。

さて、この縦軸は Amplitude (dB)とはなっていますが、これだけでは 1Hz 電力密度がいくつかは分りません。ADC 評価ソフトウェアの FFT 機能だけでは 1Hz 電力密度は分からないのです。

しかし 89410A ではきちんと 1Hz 電力密度レベルが表示されました。数学的には当然なんらかの計算（変換係数）を施していけば、1Hz 電力密度レベルが得られるだろうことも予想できるところでしょう。

この技術ノートはこれを探究してみるものです。

まず最初に FFT で抑えておくべき基本ポイント

FFT は時間軸の信号を周波数軸のスペクトルに変換する場合によく使われます。FFT を用いるうえで一番基本的な抑えておくべきポイントをご紹介しておきます。

図 4 はピーク電圧 2V の 2 周期ぶんの波形を 16 サンプル AD 変換したものです。サンプリング・レートを 1ksps（sample per sec）とすると、サンプル全長の時間は 16msec（1 サンプルが 1ms で、図 4 のように 1ms の中央でサンプリング動作が行われているとして）、波形自体の周期は 8msec になりますので、周波数は 125Hz です。

これを FFT したスペクトルが図 5 です。時間波形 16 サンプル（16 ポイント）を FFT すると、16 ポイントの周波数ポイント（ひとつの周波数ポイントにおける帯域を bin といいます）、つまり同じ数だけの情報量が結果として得られることになります。

図 5 において

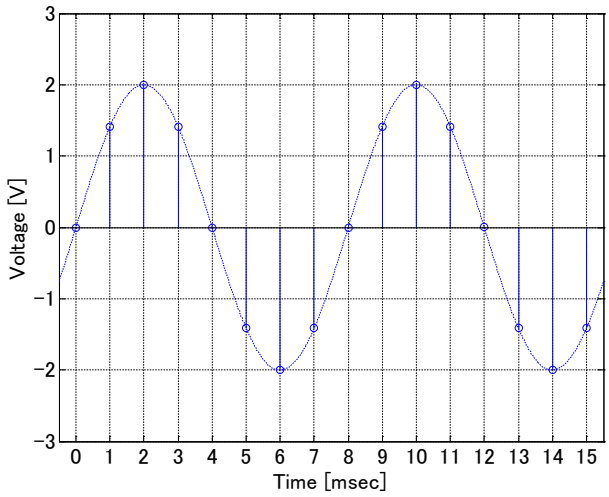

図 4. 2 周期の波形を 16 サンプル AD 変換した時間軸サンプル

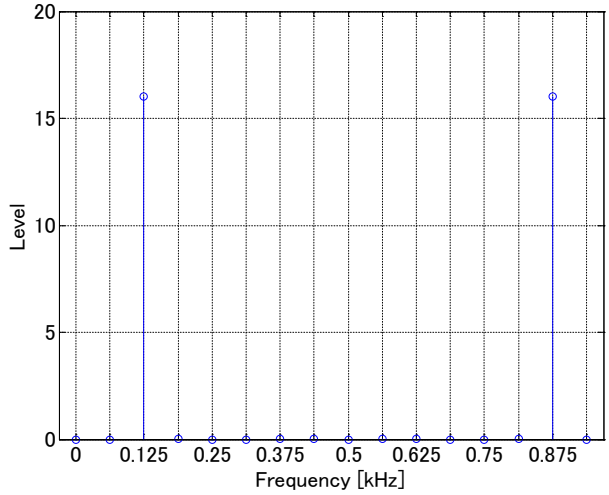

図 5. 図 4 を FFT した周波数スペクトル
（最大周波数は 937.5Hz、周波数ステップ／間隔は 62.5Hz）

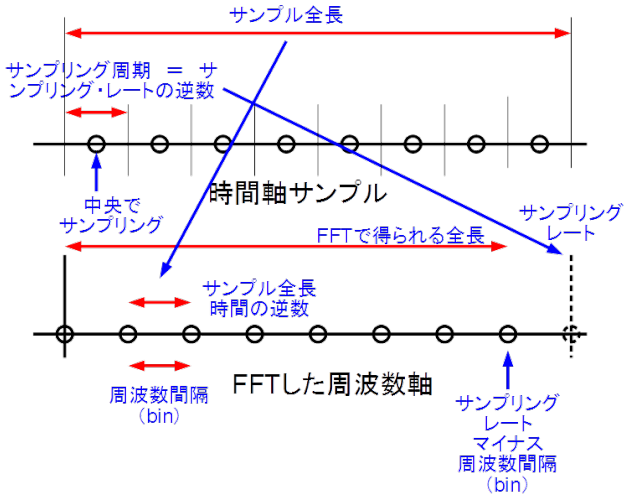

図 6. FFT する前後の時間軸サンプルと FFT スペクトルとの周波数・時間関係

● FFT した離散周波数の最大周波数 bin はサンプリング・レート － 周波数ステップ（1 bin の帯域幅）になる

この例での最大周波数は 1kHz － 1kHz/16 = 937.5Hz です。FFT した bin の周波数ステップは 1kHz/16 = 62.5Hz です。

またサンプル全長の時間は 1ms × 16 = 16ms となり、1 bin は（同じ話しですが）1/16ms = 62.5Hz となります。これから

● FFT した離散周波数の周波数ステップはサンプリング周期×サンプル・ポイント数に相当する時間の逆数になる

となります。上記に●で箇条書きとして示した 2 点が重要です。これを図としてまとめたものを図 6 に示します。

また、ピーク 2V の正弦波を 16 ポイントで FFT すると、折り返しの 2 本のスペクトルが 125Hz に 16, 875Hz（= 1kHz － 125Hz）に 16 で得られており、16/16 ポイント = 1V となっていることも分かります。本来の周波数成分 1V と折り返しの周波数成分 1V の合計が信号のピーク値、2V を形成します。

【ステップ 1】AD 変換データの取得とデータ自体の変換と補正

上記で FFT の基本的なところを示しました。いよいよ 1Hz 電力密度を求める「本題」に進んでまいりましょう。この技術ノートの最後に MATLAB/Octave の m コードを示します。

【ステップ 1.1】AD7960 でのデータ取得

EVAL-AD7960FMCZ を使用し、5Msps で 65,536 ポイントのデータを取得します。これを

$$f_S = 5\text{Msps}$$

$$n = 0, 1, \cdots, 65535$$

とします。当然ですが、以降の説明は別の ADC、別のサンプリング・レートでも成立します。数式で表すとサンプリング動作（サンプル値）は以下で表されます。

$$x_S(n) = a(t) \cdot \delta(nT - t) \qquad (n = 0, 1, \cdots, 65535) \quad (1)$$

δはデルタ関数です。まあ、サンプリングされたサンプル値が$x_S(n)$だと考えてください。しかしこの数値はまだ、AD 変換の結果として得られた ADC の読み値のままです。

【ステップ 1.2】測定値の平均値を計算

測定値の平均値x_{AVR}を計算し

$$x_{S_{AVR}} = \frac{1}{65536} \sum_{n=0}^{65535} x_S(n) \qquad (2)$$

オフセット（DC 成分）を

$$x_{S\_AC}(n) = x_S(n) - x_{AVR} \qquad (3)$$

として除去します。実際は FFT したとき直流成分は DC 部分にしか現れませんから、この計算は不要ともいえるでしょう。以降の計算で桁あふれが生じそうな場合は、このオフセット除去をしておいたほうがよいでしょう。作った MATLAB/Octave のコードではx_{AVR}を求めるのに、65536 要素のオール 1 のベクトルとの内積計算をしています。

【ステップ 1.3】実際の電圧値に変換

AD7960 のデータシート[3]の Table 7 によると、リファレンス電圧 REF = +5V 時では、入力電圧±5V において 18 ビット・フルスケールになることが分かります。すなわち電圧スパン 10V で、18 ビット分解能（2^{18} = 262,144 電圧ステップ）が形成されることになります。これにより 1 LSB に相当する電圧は

$$v_{LSB} = \frac{FS}{2^N} = \frac{10\text{V}}{262144} \quad (4)$$

となります。ここでFSはフルスケール電圧、Nは ADC のビット数です。取得したデータ$x_S(n)$の真値$x(n)$を得るため、

$$x(n) = x_{S\_AC}(n)v_{LSB} = x_{S\_AC}(n)\frac{10\text{V}}{262144} \quad (5)$$

と掛け算します。

【ステップ 1.4】窓関数と乗算

つづいてスペクトル・リーク（この話題は、回路設計 WEB ラボでも [4]で説明しています）を改善するために、窓関数を掛けます。ここでは Hann 窓を使います。

$$w_{Hann}(n) = 0.5\left[1 - \cos\left(2\pi\frac{n}{65536-1}\right)\right] \quad (6)$$

nは 0 ～ 65536−1 です（この話題はふたつ後の TNJ-082 もご覧ください）。窓関数$w_{Hann}(n)$を掛け算することで、その減衰補正係数をさらに乗算する必要があります（2 か所です）。それぞれの部分で説明していきます。

なお Hann 窓関数は MATLAB では Signal Processing Toolbox に入っており、Octave では signal package に入っているので pkg load signal というコマンドでロードします。

FFT 処理し周波数スペクトルに変換

上記のように加工した時間軸での65,536個のサンプル値$x(n)$を FFT し、サンプル数で割ります。

$$\Omega(m) = \frac{\text{FFT}[x(n)]}{65536} \quad (7)$$

得られたmも、$m = 0, 1, \cdots, 65535$で65,536個になります。ここでΩという記号を用いましたが、これは抵抗の単位「オーム」という意味ではなく、周波数軸（オメガ）のデータ列を表すもの（FFT 結果である離散値としての周波数スペクトル）だとお考えください。そういう私も信号処理理論だかの教科書で、記号「Ω」というものを初めて見たとき、「？」と思ったもので、他人事ではありません（笑）。

この FFT 結果の大きさを考えてみましょう。図 4 では振幅ピーク値が 2V です。図 5 では 3 番目の bin と 14 番目の bin の大きさが 16 になっています。先にも示しましたがこの 16 というのは、FFT 結果として FFT ポイント数（この場合は 16 サンプル／16bin）が係数としてかかっているため、実際の「求めたいスペクトルの大きさ」という点からは、この FFT 結果をサンプル数（bin の本数）で割ります。そうすると 1V になります。

もともとの（図 4 の）時間軸振幅が 2V でしたから、以降に示す式(8)と同じように、FFT 結果は実信号の大きさの 1/2 になっていることが分かります。

FFT 結果との周波数関係

5Msps, 65,536 サンプルの時間軸データを FFT して得られた結果は、最初に説明したように、

- ナイキスト周波数はサンプリング周波数の 1/2 の$f_{NYQ} = 2.5\text{MHz}$になる
- FFT した周波数スペクトルの周波数ステップ（1 bin の帯域幅）はサンプル時間全長の逆数になるため、$f_{BIN} = 1/(0.2\mu\text{s} \times 65536) = 76.29\text{Hz}$になる
- FFT した最大周波数は、サンプリング・レートf_Sマイナスf_{BIN}で、$f_{FFT_max} = 5\text{MHz} - f_{BIN}$になる

FFT 処理結果を補正し電力値に変換

【ステップ 2.1】FFT の折り返しの考慮

実信号を複素信号で表してみると、高校のときに習った式のように

$$V\cos(\omega t) = \frac{V}{2}e^{j\omega i} + \frac{V}{2}e^{-j\omega i} \quad (8)$$

として各信号（電圧だと考えるとよいです）の成分は、正の複素周波数成分と負の複素周波数成分となり、それぞれ元の大きさの 1/2 になります。

これと同じように、図 4 と図 5 においても、FFT 結果の正の周波数成分と、負の周波数成分に相当する折り返し周波数それぞれにおいて、実信号の大きさの 1/2 がスペクトルに現れています。つまり実信号スペクトルのレベルを得たいのであれば、FFT 結果を［まずサンプル数（bin の本数）で割ったうえで］2 倍する必要があるということです。そこで、

$$\Omega(m) = 2 \times \Omega(m) \quad (9)$$

また、ナイキスト周波数範囲内で考えるため、nの範囲を全体の半分に限定しておきます。といっても計算上では、この「限定」は何も対処することはありません。「そう考えてください」ということです。

【ステップ 2.2】FFT 結果を実効値として得るための補正

FFT した結果は、さきの説明のように、その bin 中央周波数にある信号での「振幅ピーク値」を示します。繰り返しますが図 4 で振幅ピーク値が 2V で、図 5 に FFT 折り返しを考慮したときの結果も 2V になっていますから、この話しも腑に落ちるものと思います。

そこで振幅ピーク値である FFT 結果を実効値にする必要があります。

$$\Omega(m) = \frac{\Omega(m)}{\sqrt{2}} \quad (10)$$

【ステップ 2.3】窓関数による減衰補正（スケーリング係数補正）

つづいて窓関数減衰補正係数を掛けます。図 7 は[5]の Table 3 から転記した、いろいろな窓関数の減衰補正係数［Scaling Factor (Coherent Gain)］です。Hann 窓だと係数は 1/0.5 = 2 倍になります。これは元々の信号に対して窓関数をかけることで、信号の大きさが低減して（Hann 窓だと 0.5 倍になって）しまうからです。それを FFT すれば、低減した信号の大きさが結果として得られるからです。なお図 7 の右側の Noise Power Bandwidth は、もうひとつの補正係数です。これは以降で示します。

なお以降の 2 冊の技術ノート、TNJ-081, TNJ-082 でこれらの窓関数補正係数について LTspice でのシミュレーションを含め、詳しくみていきますので「乞うご期待」ください（笑）。

【ステップ 2.4】電力で移動平均してレベルを平準化

ここまでで FFT により得られた電力スペクトルを図 8 に示します。ただし帯域を 400kHz から 600kHz で表記しています。しかしこれではレベル変動が大きすぎて、本来の値がいくつかを特定することができません。

そこで周波数軸でアベレージングを施します。とはいえ単純な電圧レベルでのアベレージでは正しい答えが出ません。Root Sum Square = RSS の考え方で電力での平均化として計算します。

Window	Scaling Factor (Coherent Gain)	Noise Power Bandwidth
Uniform (none)	1.00	1.00
Hann	0.50	1.50
Hamming	0.54	1.36
Blackman-Harris	0.42	1.71
Exact Blackman	0.43	1.69
Blackman	0.42	1.73
Flat Top	0.22	3.77

図 7. いろいろな窓関数の補正係数（[5]の Table 3 より転記）

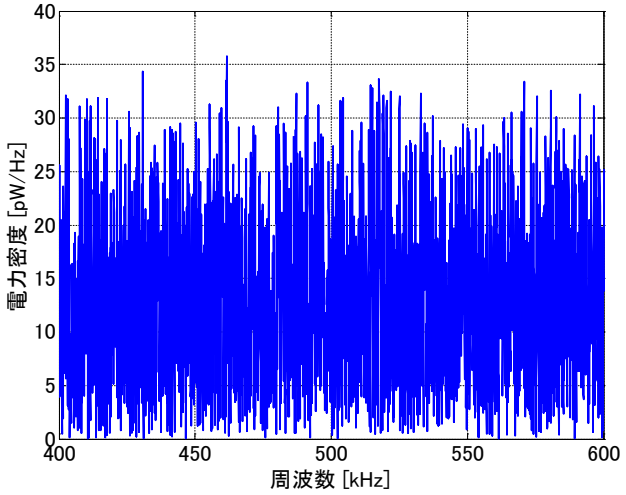

図 8. ここまでで得られた FFT による電力スペクトル（400kHz から 600kHz の帯域で表記）

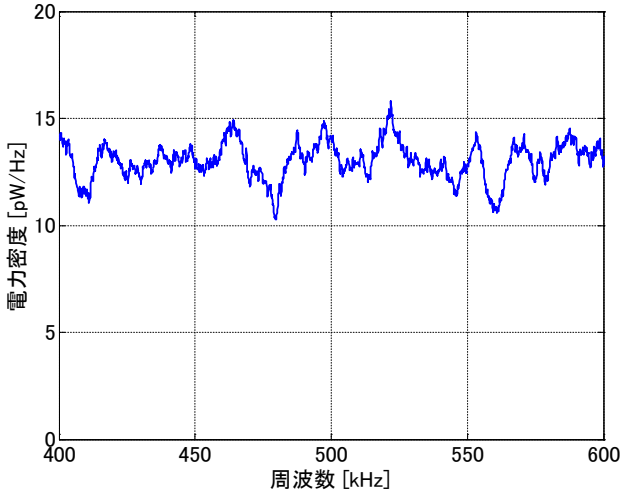

図 9. 図 8 の電力スペクトルを移動平均化して平準化した（400kHz から 600kHz の帯域で表記）

まずここまで得られた周波数スペクトル$\Omega(m)$を、電力に変換します。一般的にスペクトラム・アナライザなどは 50Ω系ですから、50Ωを基準とした電力値Ω_Pにします。

$$\Omega_P(m) = \frac{\Omega^2(m)}{50} \ [\mathrm{W}] \tag{11}$$

つづいて 100 ポイントの移動平均（電力値での）として

$$\Omega_P(m) = \frac{1}{100}\sum_{k=-50}^{49}\Omega_P(m+k) \tag{12}$$

なお稿末に掲載する私の MATLAB/Octave コードでは、100 要素の 1 のベクトルと、畳み込みを用いて平均化しています（コードでは周波数位置が少しずれますが、誤差の範囲ですので…）。

図 8 を電力値で平均化したスペクトルを図 9 に示します。だいぶ平準化できていることが分かります。

【ステップ 2.5】窓関数によるノイズ等価帯域幅の補正

つづいて窓関数によるノイズ等価帯域幅の補正を行います。図 7 にも Noise Power Bandwidth として表記されているものです。Hann 窓ではこのノイズ等価帯域幅補正係数は 1.5 になっています（窓関数なしでこの補正係数は 1）。

たとえば時間軸サンプル全長 0.5sec で考えてみましょう。このとき周波数 bin はその逆数の 2Hz になります。窓関数が Hann 窓を使用しているとすれば、このとき等価ノイズ帯域幅 Equivalent Noise Bandwidth = 1.5/0.5 = 3Hz となります。つまり 1 bin の帯域幅f_{BIN}に対して

$$f_{BIN\_ADJ} = 1.5 \times f_{BIN} \tag{13}$$

と補正します。なおこの補正は電圧値に対しての補正ではありません。電力値に対してのものとなります（「全電力 = 1Hz 電力密度 × 帯域幅」の計算と同じ考え方です）

繰り返しますが、このノイズ等価帯域幅の補正、そして前に説明した窓関数減衰補正係数については、以降の技術ノート TNJ-081, TNJ-082 で詳しくみていきます。これがまた奥深く、脳ミソを麻痺させるものでした…（笑）。

補正されたノイズ等価帯域幅で割り 1Hz ノイズ電力密度を得る

ここまで得られた周波数スペクトル電力値$\Omega_P(m)$（mは 0 ～ 65536－1）は、ノイズ等価帯域幅f_{BIN_ADJ} [Hz]という矩形フィルタ帯域内でのノイズ電力値に相当します。1Hz ノイズ電力密度$\Omega_{PSD}(m)$を得るため

$$\Omega_{PSD}(m) = \frac{\Omega_P}{f_{BIN\_ADJ}} \ [\mathrm{W/Hz}] \tag{14}$$

実際には$f_{BIN} = 76.29\mathrm{Hz}$でしたから、$f_{BIN_ADJ} = 114.44\mathrm{Hz}$になります。これで$\Omega_P$を割り、$\Omega_{PSD}$を得ます。

dB に変換し前回の検討結果と比較してみる

最後に、

$$\Omega_{PSD\_dB}(m) = 10 \times \log_{10}\left(1000 \times \Omega_{PSD}(m)\right) \ [\mathrm{dBm/Hz}] \tag{15}$$

で dBm/Hz に変換して出来上がりです。これを図 10 に示します。

前回の技術ノート TNJ-079 の実測結果では、同技術ノートの図 7 の 89410A で－78.3dBm/Hz、また図 8 の 8560B で－78.5dBm/Hz となっていました。同じくパーセバルの定理を利用した理論計算では－78.4dBm/Hz でした。今回ここで得られた答えが、前回技術ノートで検討した結果に沿ったものであること（考え方が正しいこと）が分かります。

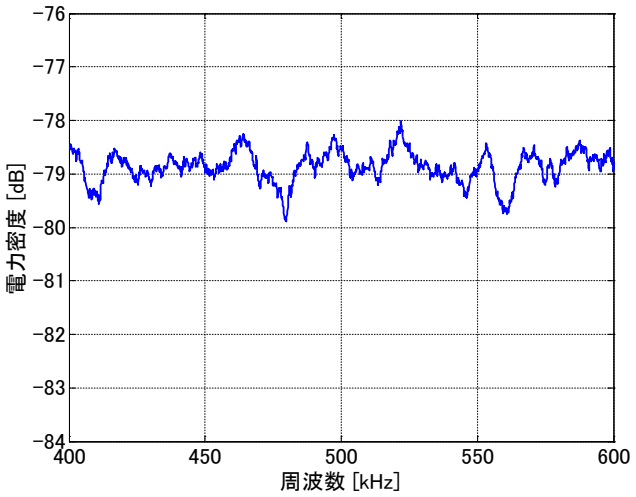

図 10. 図 9 の電力スペクトルを dB に変換した

まとめ

今回の技術ノートでは、前回説明したようなレベルが既知のノイズ源を測定した実測結果を基準にするという現場的アプローチではなく、FFT した結果からそのまま 1Hz 電力密度を求めるというアプローチを考えてみました。私も計算手順を編み出すのにだいぶ手間取ってしまいましたが、1dB 以下のオーダまできちんと答えが合ったことに、またまた今回も書いている本人も驚きを隠しえませんでした。そしてここまでの答えが出たことは、とてもうれしく感じるものでした。

また今回は「窓関数」とその「補正係数」というものが出てきました。今回の検討ではそれら補正係数の数値ありきで説明してきましたが、「この補正係数はどうやって決まるのだろうか」と執筆しながら思っていました。次回以降の 2 冊の技術ノートでは、この「窓関数の補正係数」について、その重箱の隅をつつくような、個人的興味による探究に突き進んでみたいと思います（笑）。

参考文献

[1] 石井 聡; ADC の *NF* とフロントエンド込みの最適化（前編）, 回路設計 WEB ラボ, TNJ-079, アナログ・デバイセズ

[2] 石井 聡; ADC で 1Hz ノイズ電力密度を得るためのランダム・ビット発生回路と密度計算式, 回路設計 WEB ラボ, TNJ-077, アナログ・デバイセズ

[3] AD7960 Datasheet, Analog Devices

[4] 石井 聡; 正弦波を A/D 変換し窓関数なしに打ち切って FFT してみると, 回路設計 WEB ラボ, TNJ-008, アナログ・デバイセズ

[5] Michael Cerna, Audrey F. Harvey; The Fundamentals of FFT-Based Signal Analysis and Measurement, Application Note 041, National Instruments

[6] Keysight Technologies, Inc., Window Shapefactor and Equivalent Noise Bandwidth, http://rfmw.em.keysight.com/wireless/helpfiles/ 89600B/WebHelp/Subsystems/gui/content/windows_shapefactor_and_ equiv_noisebw.htm

[7] Mathworks, enbw 等価ノイズ帯域幅, https://jp.mathworks.com/help/signal/ref/enbw.html

参考：MATLAB/Octave コード

```
% AD 変換結果の電力密度の計算 -- psdcalc.m

%信号の読み込みと電圧レベルへの変換
%信号ファイルは AD7960 評価ボードから吐き出された CSV のヘッダを除去しておく
sig = csvread('sampledata.csv');
refvolt = 5;
sig = refvolt * sig / (2^(17));
% 上記は ref_voltage x 2 / (2^18)

%差分用平均値の算出
len = size(sig);
len = len(1, 1);
calavr = sig' * ones(len,1) / len;

%Hann 窓の設定
hannwindow = hann(len);
windowedsig = hannwindow .* (sig - calavr);

%周波数設定
samplefreq = 5e6; % サンプリング周波数に相当
freq = linspace(0, samplefreq, len + 1);
freq = freq(1, 1:len)';

%ノイズの周波数スペクトルを FFT で得て電力に変換
noisefft = 2 * fft(windowedsig) ./ len; % 折り返しの分で 2 倍にしている
noisefft = noisefft / sqrt(2);  % rms レベルに変換
noisefft = 2 * noisefft; %Hann 窓のスケーリング係数 0.5 を補正
noisepowerbin = (abs(noisefft)) .^2 / 50;  % [W]

%周波数軸で電力レベルでアベレージング
%単純な電圧レベルアベレージでは正しい答えが出ない（rss の考え方）
%  （100 回の移動平均で畳み込み平均化することで、周波数位置は少しずれるが…）
noisepowerbin = conv(noisepowerbin, ones(100, 1)) / 100;
noisepowerbin = noisepowerbin(1:len, 1);

%single bin のノイズ帯域幅
binwidth = 1/(len / samplefreq); % サンプル全長の逆数
binwidth = binwidth * 1.5;    % 窓関数ノイズ等価帯域幅補正係数 Hann = 1.5
noisepowerpsd = noisepowerbin/binwidth;

%dB に変換
noisepowerdB = 10 * log10(1e3*noisepowerpsd);
plot(freq/1e3, noisepowerdB, 'LineWidth', 2)
```

```
set(gca, 'xtick', [400:50:600])    %400kHz から
600kHz を表示
set(gca, 'ytick', [-84:1:-76])
axis([400 600 -84 -76])
xlabel('周波数 [kHz]', 'FontSize', 16)
ylabel('電力密度[dB]', 'FontSize', 16)
set(gca, 'FontSize', 16)
grid on
```