スーパーサンプリング DAC

無限スプライン関数の変域と精度

小林　芳直*　（東京大学）
肥後　信嗣　　（SLDJ 合同会社）

The domain and the precision of the Infinite-Spline Function

Yoshinao Kobayashi* (Tokyo University)
Nobutsugu Higo (SLDJ Limited Liability Company)

The domain and the precision of the infinite-spline function is valuated in order to fix the hardware size of each process that is enough and necessary. The infinite-spline function sometimes shows off-peak values that the peak value appears not on the sampling-point, but between two sampling-point. Then the maximum value of the infinite-spline function may exceeds the maximum value of sampling-point, then the domain of the infinite-spline function must accurately be evaluated in order to make sure the stable operation.

キーワード：無限スプライン関数，フロントインパルス関数，**バックインパルス**関数，係数の変域，関数の変域
(Keywords, infinite-spline function, front-impulse function, back-impulse function, domain of each coefficient, domain of infinite-spline function)

1.　概要

スーパーサンプリング DAC は、デジタルオーディオ信号のデータ列から無限スプライン関数の係数を計算して、サンプル点とサンプル点の間をスプライン関数で補間をするという DAC である。スプライン関数の性質としてサンプル点以外のところにも極値を持つことができる。そしてその極値はサンプル点の最大値よりも大きくなることがある。無限スプライン関数の変域を正確に見積もることにより、無限スプライン関数を過不足ない精度で実装することができる。

2.　DAC の歴史

〈2･1〉　NOSDAC　　CD の 16 ビット 44.1KHz のデジタルデータ列を元のアナログ波形にするためには DA 変換器が必要です。最初に現れたのは NOSDAC：Non-oversampling DAC、と現在では呼ばれるものです。これは 16 ビットのデジタルデータを 44.1KHz の変換速度でアナログ値に変換して LPF を通してアナログ波形を得るという、DAC の原点といえる構成です。

現在の DAC はサンプル点とサンプル点の間を中間値で補間するオーバーサンプリング：oversampling、が主流なので NOSDAC と称される。

NOSDAC の出力は 44.1KHz 毎にアナログ値が変化する階段波になる。アナログ値が次のサンプル点まで変化せず一定値が保たれることを 0 次ホールドいう。

元のアナログ波形と再生されたアナログ波形を比較すると DAC の出力には多量ののこぎり波の形をした歪が認められる。これはサンプル点の度に元のアナログ波形と DA 変換した値は一致するが、元のアナログ波形は連続して変化するのに対して、0 次ホールドされたアナログ値は一定の値に保たれるので、両者は時間とともに徐々に離れて、次のサンプル点で一致する。結果として歪成分はサンプル点で０、そこから徐々に離れて次のサンプル点で再び 0 になる鋸波になる。鋸波の振幅は元のアナログ波形が速く変化するほど大きくなるから、NOSDAC の周波数特性は高域で減衰し歪は増大する。

これはアパーチャ効果と呼ばれていて NOSDAC の欠点である。NOSDAC は原始的な DAC であり改良の余地は多々あるが、素朴で切れの良い再生音は一定の評価をされている。

NOSDAC には別の欠点もある。初期の DAC チップは精度が悪くデジタルデータとアナログ値の直線性が保証されていなかった。そのためにデジタルデータが直線的に変化しても DA 変換されたアナログ値は正確な階段波にはならず段差のばらつきや時には逆段差も発生していた。これは

DACチップのバラツキで発生するものであり大量生産される製品の個体差の音質の違いとなって現れる。結果として製品の品質と精度が保てない、という致命的な欠陥となって世から駆逐されていった。

〈2･2〉　サインマグニチュード　デジタルデータの多くのビットが同時に変化するときに DAC の直線性の悪さが露呈する。デジタルデータの値が一つの値から次の値に移るときに、二つのデータを比較して反転するビット数をハミング距離という。

ハミング距離が大きいほど DAC の直線性の悪さが出やすくなする。ハミング距離が最大になるのは 16 ビットデジタルデータの場合

-1=”1111111111111111” から　0=”0000000000000000” に変化した時である。アナログデータは交流信号なので０近辺で変化することが多く、しかも DAC チップには０点をクロスするときに最も歪が出やすいという弱点がるとすれば何らかの対策が必要になる。この弱点を隠すために導入されたのがサインマグニチュードと呼ばれる技術である。これはデジタルデータの絶対値を取り、DAC は常に正側で動作させて、正負の切り替えはアナログ回路で行うというものである。0 点近辺の直線性という DAC の最大の弱点は隠せるがそれ以外のハミング距離の大きい変換は苦手であり、DAC の直線性が改善されたわけではない。サインマグニチュードは弥縫策と言える技術である。

〈2･3〉　DAC の並列使用　DAC を複数並べて動作させるとDAC のばらつきが平均化されて統計的に N 個使いなら$\sqrt{N}$分の1に直線性が改善できる。また複数の DAC のタイミングをずらして動作させるとサンプル点とサンプル点の間を直線補間することになり0次ホールドに比べて歪が減少する。

みかけの歪は減少するが聴感上は0次ホールドよりよい評価を得ていない。DAC の精度が出ないときに回路技術でカバーしようとしてもうまくいかないことが多い。DAC の暗黒時代かもしれません。

〈2･4〉 δΣの時代　少ビットのＤＡＣを高速動作させて高精度のＤＡＣを作る技術が開発されました。ＤＡＣを意図的に高速で目的値周辺で往復させることによりＤＡＣのノイズ成分を高域に追いやることができてオーディオ帯域のノイズを少なくすることができる。これをノイズシェーピングという。

ＤＡＣの精度はＤＡＣに与えるクロックの精度だけで決まり、ＤＡＣのバラツキというものが現れません。DAC の直線性を画期的に改善するとともに製品の固体バラツキをなくす救世主として現れた技術です。

〈2･５〉オーバーサンプリングと SINC 関数　サンプル点とサンプル点の間を補間することにより滑らかな再生が可能になる。この補間値を作るために使われるのがディジタル・フィルタ（以下 DF）である。DF にＳＩＮＣ関数を使うとＮＯＳＤＡＣで問題になったアパーチャ効果による高域の減衰を保障することができる。ＳＩＮＣ関数は連続サイン波の周波数特性を改善するために決定的な効果がある。サンプリング定理のほぼ限界まで周波数特性を延ばすことができる。歪も少なくすることができる。

ＳＩＮＣ関数は$\sin x/x$ という関数であり、サンプル点の値だけでなくその周辺のデータも使って連続サイン波を再現する。ＳＩＮＣ関数は$\sin x/x$という関数の形通り収束は非常に悪く、数学的に収束するということがない。そのために DF にウィンドウ関数を被せて、周波数特性はほんの少し犠牲にして、ハードウェアサイズを無制限に大きくしないという回路テクニックが併用される。ウィンドウ関数を被せることにより、遠くのサンプル点の値がＤＡ変換に影響しなくなり、過渡応答を良くするという効果もある。

〈2･６〉プリエコー　DF に方形波が入ると、立ち上がりの前にプリエコーというナイキスト周波数成分の歪が出る。ナイキスト周波数はサンプリング周波数のちょうど半分の周波数であり、サンプリング定理では再生限界とされている周波数である。元のアナログ波形とＤＡＣの再生波形が上下に交互にずれるとナイキスト周波数になる。

これに対して元のアナログ波形とＤＡＣの再生波形が連続して片側にずれていると歪の主成分はサンプリング周波数になる。NOSDAC の歪の主成分はサンプリング周波数である。ナイキスト周波数成分の歪は補間がうまくいっているという証であり高性能のＤＡＣから最も出やすい周波数成分である。しかし出すぎるのは良くない。ＳＩＮＣ関数を DF を使うと方形波の立ち上がりの前にナイキスト周波数の 10～20 波程度のプリエコーが出る。方形波の平坦部ではやはりナイキスト周波数のリンギング、そして方形波の後にポストエコーと呼ばれるナイキスト周波数の歪が続く。

SINC 関数のこのような欠点はよく知られているが、それでも製品の連続サイン波の周波数特性というスペックを整えるために必要なので広く使われている。

〈2･７〉ソフトカットオフ　SINC 関数で周波数特性を高域まで伸ばすと過渡応答が悪くなする。そもそも音質を良くするためには切れの良い過渡応答が必要であり、過渡応答の評価基準として周波数特性がある。連続サイン波の周波数特性を延ばすために過渡応答を犠牲にするというのでは本末転倒である。DF の効きを弱くして、周波数特性をあまり伸ばさない代わりに過渡応答は良くするソフトカットオフの技術が出てきた。複数の DF を用意して音質を選べるようにするのが流行です。

〈2･８〉スーパーサンプリング DAC

SINC 関数の代わりにスプライン関数で補間するのがスーパーサンプリング DAC である。スプライン関数は EXCEL の表を作るときに、複数の点の包絡線を引くといった用途に使われる。一般的なスプライン関数は自然スプライン関数といい領域内のすべての点の値が確定した後でないとスプライン関数の形が決まりません。これに対してオーディオで使う無限スプライン関数は延々と続くデータ列の一部があればスプライン関数の形を決めることができる。スプライン関数を補間に使うとプリエコーやポストエコーといったナイキスト周波数成分の歪は激減するが、周

波数特性はあまりよくない。

３．　無限スプライン関数

〈３･1〉背景　　スプライン関数は自然スプライン関数が主流であり、無限データ列には適応できないとされていた。自然スプライン関数は領域内のすべてのデータが確定しないと係数計算を開始できないからである。正直にCD のデジタルデータに自然スプライン関数を適応しようとすると一曲の読み込みが終わってから係数計算が始まりそれから再生開始となるので遅れ時間の長短ではなく全く使い物になならない。ましてや TV の映画再生になると画像と音声のずれは許し難きものになる。データが流れに沿った再生が必要であり、それを可能にしたのが無限スプライン関数である。

〈３･２〉スプライン関数の基礎　　スプライン関数はサンプル点とサンプル点の間を３次関数で滑らかにつなぐものである。一つのスプライン関数はサンプル点とサンプル点の一つの区間でのみ使われる。区間が変わるとスプライン関数も新たに定義される。

区間の切れ目、つまりサンプル点で前後二つのスプライン関数が接触するが、ここで両者を滑らかにつなぐために二つのスプライン関数の、値が一致、1 次微分が一致、２次微分が一致という制約がある。

$S_j(x) = a_j x^3 + b_j x^2 + c_j x + d_j$　　…①

$a_j + b_j + c_j + d_j = d_{j+1}$　　…②　　∵値が連続

$3a_j + 2b_j + c_j = c_{j+1}$　　…③　　∵１次微分が連続

$6a_j + 2b_j = 2b_{j+1}$　　…④　　∵２次微分が連続

これらの式から

$$a_j = \frac{1}{3}(b_{j+1} - b_j) \quad \cdots⑤$$

⑤式を②式に代入して

$$c_j = d_{j+1} - d_j - \frac{1}{3}b_{j+1} - \frac{2}{3}b_j \quad \cdots⑥$$

$$b_{j-1} + 4b_j + b_{j+1} = 3(d_{j-1} - 2d_j + d_{j+1}) \cdots⑦$$

⑦式はスプライン関数の公式であり、この式を解くことによってスプライン関数の係数を決定することができる。

b_jが独立変数でありb_jを求めればa_jとc_jは⑤⑥式から求めることができる。

さて⑦式は左辺が変数３個、右辺は given なのでこの式を N 式集めると変数の数は（N+2）個、式の数は N 式となって解くことができない。

しかし領域内のすべてのデータが揃うと、両端のb_jは領域からはみ出すので 0 にして構わない。すると変数の数と式の数が一致して解くことができる。このように両端のb_j　を０にして解く方法を自然スプライン関数という。

自然スプライン関数は最上部から下へなめれば各式の変数の数が２になり、そのあと最下部から上へなめれば各式の変数を１にできるので比較的簡単にすべての変数を決定することができる。

〈３･２〉無限スプライン関数　　連続するデータ列からスプライン関数を決定するためには自然スプライン関数のように両端が 0 という境界条件は使えない。

⑦式を時系列で並べたものは次のような行列式になる。これは行列式で書くと

$B\boldsymbol{b} = 3D\boldsymbol{d}$

但し $\boldsymbol{b}$ はb_j　の無限ベクトル、$\boldsymbol{d}$ はd_j　の無限ベクトル、B,D は次のような無限三重対角行列である。

$$B = (B_{ij}) \quad B_{ij} = \begin{cases} 4 & if \ \ i = j \\ 1 & if \ \ |i-j| = 1 \\ 0 & otherwise \end{cases}$$

$$D = (D_{ij}) \quad D_{ij} = \begin{cases} -2 & if \ \ i = j \\ 1 & if \ \ |i-j| = 1 \\ 0 & otherwise \end{cases}$$

以下の式は無限次元の行列式の j 近傍を抜き出したものである。

$b_{j-6} + 4b_{j-5} + b_{j-4} = 3(d_{j-6} - 2d_{j-5} + d_{j-4})$　　α^5 (7-5)

$b_{j-5} + 4b_{j-4} + b_{j-3} = 3(d_{j-5} - 2d_{j-4} + d_{j-3})$　　α^4 (7-4)

$b_{j-4} + 4b_{j-3} + b_{j-2} = 3(d_{j-4} - 2d_{j-3} + d_{j-2})$　　α^3 (7-3)

$b_{j-3} + 4b_{j-2} + b_{j-1} = 3(d_{j-3} - 2d_{j-2} + d_{j-1})$　　α^2 (7-2)

$b_{j-2} + 4b_{j-1} + b_j = 3(d_{j-2} - 2d_{j-1} + d_j)$　　α (7-1)

$b_{j-1} + 4b_j + b_{j+1} = 3(d_{j-1} - 2d_j + d_{j+1})$　　1 ⑦

$b_j + 4b_{j+1} + b_{j+2} = 3(d_j - 2d_{j+1} + d_{j+2})$　　α (7+1)

$b_{j+1} + 4b_{j+2} + b_{j+3} = 3(d_{j+1} - 2d_{j+2} + d_{j+3})$　　α^2 (7+2)

$b_{j+2} + 4b_{j+3} + b_{j+4} = 3(d_{j+2} - 2d_{j+3} + d_{j+4})$　　α^3 (7+3)

$b_{j+3} + 4b_{j+4} + b_{j+5} = 3(d_{j+3} - 2d_{j+4} + d_{j+5})$　　α^4 (7+4)

$b_{j+4} + 4b_{j+5} + b_{j+6} = 3(d_{j+4} - 2d_{j+5} + d_{j+6})$　　α^5 (7+5)

ここで⑦式から遠ざかるにつれて式全体をα倍するという演算をする。つまり⑦式は１倍でそのまま。　過去側はαの等比で式に積算する。同様に未来側についてもαの等比で式に積算する。

するとb_jについて

α　(7-1)式

4　⑦式

α　(7+1)式

という係数が残る。その他のb_{j-k}については

b_{j-k}の係数　$\alpha^k(1 + 4\alpha + \alpha^2)$

b_{j+k}については

b_{j+k}の係数　$\alpha^k(\alpha^2 + 4\alpha + 1)$

になする。ここで

$\alpha^2 + 4\alpha + 1 = 0$

ならば、すべての式を、無限の過去から、無限の未来まで加算すると左辺のb_j以外の変数はすべて消える。よって次の式が成立する。

$$(4 + 2\alpha)b_j = 3\Big((-2 + 2\alpha)d_j - 6\Big(\sum_{k=1}^{+\infty}\alpha^k d_{j-k} + \sum_{l=1}^{+\infty}\alpha^l d_{j+l}\Big)\Big)$$

$$2\sqrt{3}b_j = 3\left(\left(-6+2\sqrt{3}\right)d_j - 6\left(\sum_{k=1}^{+\infty}\alpha^k d_{j-k} + \sum_{l=1}^{+\infty}\alpha^l d_{j+l}\right)\right)$$

$$b_j/_3 = -\left(\sqrt{3}-1\right)d_j - \sum_{k=1}^{+\infty}\sqrt{3}\alpha^k d_{j-k} - \sum_{l=1}^{+\infty}\sqrt{3}\alpha^l d_{j+l}$$

つまりb_jが無限級数の形で解けたことになる。

ここで
$\alpha^2 + 4\alpha + 1 = 0$
の解は
$\alpha = -2 \pm \sqrt{3}$
と二つあるが、

$$\alpha = -2 + \sqrt{3} \cong -0.267949192 \cong -\frac{1}{4}$$

という値を選ぶとαが小さいのでこの式の右辺は速やかに収束する。一方
$\alpha = -2 - \sqrt{3} \cong -3.732050808$
を選ぶと右辺は発散し値が求まらない。よって以下
$\alpha = -2 + \sqrt{3}$
を使うことにする。

さてこの無限級数の収束は早く、一つ隣に行くたびに２ビット分ほど影響が少なくなることがわかる。級数の打ち切り誤差を計算してみると
16 ビットデータの場合は９近傍まで、24 ビットデータでも 13 近傍まで計算すればそれより外のデータはすべて加算しても LSB 以下となり計算を打ち切って構わない。
この 13 近傍という値は通常の DF が 25 近傍まで計算するのに比べても小さな値であり、IC 化する場合には通常の SINC 関数を使った DF より小さなハードウェアで計算を実行できる。

b_jを計算するための計算量は自然スプライン関数に比べて大きくなる。

自然スプライン関数の場合は N 個のデータ列に対して演算の回数は 2N 回なので一つのデータあたりの演算回数は２回で済む。

これに対して無限スプライン関数では注目するサンプル点の前後 13 サンプル点の計算をして初めて一つb_jが求まる。計算量は単純計算で 28 倍程度になる。これは回路の工夫により少なくすることができるが、いずれにしても DSP では不可能な計算量であり、なんらかのハードウェアの演算装置が必要になる。

〈３･３〉インパルス関数　無限級数の計算には次のような漸化式を実行するハードウェアを使う。
$\mathrm{Impulse(j)} = \mathrm{d}_j + \alpha\,\mathrm{Impulse()}$
この関数は一つのデータと、現在の自分自身の値を値をα倍して加算するものである。

この漸化式を繰り返し実行することで無限級数の計算ができる。例えば未来方向の級数計算は次のようになる。
$\mathrm{Impulse(j+13)} = \mathrm{d}_{j+13}$
$\mathrm{Impulse(j+12)} = \mathrm{d}_{j+12} + \alpha\,\mathrm{Impulse(j+13)}$
$\mathrm{Impulse(j+11)} = \mathrm{d}_{j+11} + \alpha\,\mathrm{Impulse(j+12)}$
$\mathrm{Impulse(j+10)} = \mathrm{d}_{j+10} + \alpha\,\mathrm{Impulse(j+11)}$
$\mathrm{Impulse(j+9)} = \mathrm{d}_{j+9} + \alpha\,\mathrm{Impulse(j+10)}$
$\mathrm{Impulse(j+8)} = \mathrm{d}_{j+8} + \alpha\,\mathrm{Impulse(j+9)}$
$\mathrm{Impulse(j+7)} = \mathrm{d}_{j+7} + \alpha\,\mathrm{Impulse(j+8)}$
$\mathrm{Impulse(j+6)} = \mathrm{d}_{j+6} + \alpha\,\mathrm{Impulse(j+7)}$
$\mathrm{Impulse(j+5)} = \mathrm{d}_{j+5} + \alpha\,\mathrm{Impulse(j+6)}$
$\mathrm{Impulse(j+4)} = \mathrm{d}_{j+4} + \alpha\,\mathrm{Impulse(j+5)}$
$\mathrm{Impulse(j+3)} = \mathrm{d}_{j+3} + \alpha\,\mathrm{Impulse(j+4)}$
$\mathrm{Impulse(j+2)} = \mathrm{d}_{j+2} + \alpha\,\mathrm{Impulse(j+3)}$
$\mathrm{Impulse(j+1)} = \mathrm{d}_{j+1} + \alpha\,\mathrm{Impulse(j+2)}$
というように 13 回の計算を実行すれば
$\mathrm{Impulse}(j+1) = \sum_{k=0}^{12}\alpha^k d_{j+1+k}$
というように無限級数の未来 13 近傍までを計算したことになる。計算結果をα倍に縮小してから新しいデジタルデータを加算するので誤差の蓄積も起こらない。
Impulse()は過去側と未来側で同じハードウェアを使うのでこれらを分けるために BackImpluse()と FrontImpulse()を次のように定義する。
$\mathrm{BackImpluse(j)} = \sum_{k=0}^{\delta}\alpha^k d_{j-k} = e_j$
$\mathrm{FrontImpluse(j)} = \sum_{k=0}^{\delta}\alpha^k d_{j+k} = f_j$
ここでδは無限級数の収束誤差を無視できる大きな値とする。

〈３･4〉インパルス関数の変域　この関数の最大値を I とおくと
$\mathrm{I} = 1 + \alpha\,\mathrm{I}$

$$\mathrm{I} = \frac{1}{1-|\alpha|} = \frac{1}{1-2+\sqrt{3}} = \frac{1}{\sqrt{3}-1} = \frac{\sqrt{3}+1}{2} = 1.366025404$$

$$\therefore -\frac{\sqrt{3}+1}{2} \le \mathrm{Impulse(x)} \le \frac{\sqrt{3}+1}{2}$$

Impulse 関数の変域は２以下であり、この計算には MSB 側に＋１ビットの拡張が必要である。Impulse 関数は$\sqrt{3}\alpha$倍された後$b_j/_3$の算出に使われるので都合、スプライン関数のb_jの計算には$3\sqrt{3}\alpha$倍されることになる。つまり

$$b_j \propto 3\sqrt{3}\left(-2+\sqrt{3}\right)Impulse() = \left(-6\sqrt{3}+9\right)Impulse()$$
$$\cong -1.39230484\,Impulse()$$

つまり Impulse()の LSB 側は＋１ビットの拡張が望ましい。

〈３･5〉$b_j/_3$　b_jは Impluse()を使えば次のように書ける。

$$b_j/_3 = -\left(\sqrt{3}-1\right)d_j - \sum_{k=1}^{+\infty}\sqrt{3}\alpha^k d_{j-k} - \sum_{l=1}^{+\infty}\sqrt{3}\alpha^l d_{j+l}$$

$$\therefore b_j/_3 = -\left(\sqrt{3}-1\right)d_j - \sqrt{3}\alpha\, e(j-1) - \sqrt{3}\alpha\, f(j+1)$$

ここにImpulse(x)を代入すれば$^{b_j}/_3$の最大値が求まる。

$$\max {}^{b_j}/_3 = (\sqrt{3}-1) + \sqrt{3}(2-\sqrt{3})\left(\frac{\sqrt{3}+1}{2} + \frac{\sqrt{3}+1}{2}\right)$$
$$= \sqrt{3} - 1 + \sqrt{3}(2-\sqrt{3})(\sqrt{3}+1)$$
$$= \sqrt{3} - 1 + \sqrt{3}(\sqrt{3}-1)$$
$$= (\sqrt{3}-1)(\sqrt{3}+1) = 2$$
$$\therefore \quad -2 \le {}^{b_j}/_3 \le +2$$

$^{b_j}/_3$は正確にd_jの２倍の変域を持つので、$^{b_j}/_3$のレジスタはMSB 側に＋１ビット、LSB 側に＋２ビット必要である。

〈３･６〉b_jの変域　b_jの変域は$^{b_j}/_3$の３倍なので６である。

〈３･７〉a_jの変域　　a_jの変域を調べる。

$$a_j = \frac{1}{3}(b_{j+1} - b_j)$$

より見かけの変域は４
これを詳しく調べると

$$^{b_{j+1}}/_3 = -(\sqrt{3}-1)d_{j+1} - \sqrt{3}\alpha(f_{j+2} + e_j)$$

$$^{b_j}/_3 = -(\sqrt{3}-1)d_j - \sqrt{3}\alpha(f_{j+1} + e_{j-1})$$

$$^{b_{j+1}}/_3 - {}^{b_j}/_3 = -(\sqrt{3}-1)(d_{j+1} - d_j) - \sqrt{3}\alpha(f_{j+2} - f_{j+1} + e_j - e_{j-1})$$

ここで
$f_{j+1} = d_{j+1} + \alpha f_{j+2}$
$e_j = d_j + \alpha e_{j-1}$
を代入すると
$= -(\sqrt{3}-1)(d_{j+1} - d_j) - \sqrt{3}\alpha(f_{j+2} - d_{j+1} - \alpha f_{j+2} + d_j + \alpha e_{j-1} - e_{j-1})$
$= -(\sqrt{3}-1)(d_{j+1} - d_j) + \sqrt{3}\alpha(d_{j+1} - d_j) - \sqrt{3}\alpha(f_{j+2} - \alpha f_{j+2} + \alpha e_{j-1} - e_{j-1})$
$= -(\sqrt{3} - 1 - \sqrt{3}\alpha)(d_{j+1} - d_j) - \sqrt{3}\alpha(1-\alpha)(f_{j+2} - e_{j-1})$
$= -\left(\sqrt{3} - 1 - \sqrt{3}(-2+\sqrt{3})\right)(d_{j+1} - d_j) - \sqrt{3}(-2+\sqrt{3})\left(1 - (-2+\sqrt{3})\right)(f_{j+2} - e_{j-1})$
$= -(3\sqrt{3}-4)(d_{j+1} - d_j) + (9\sqrt{3}-15)(f_{j+2} - e_{j-1})$

f_{j+2}、e_{j-1}の最大値は$^{(1+\sqrt{3})}/_2$なのでこの式の最大値は
$= (3\sqrt{3}-4)2 + (9\sqrt{3}-15)(1+\sqrt{3})$
$= 6\sqrt{3} - 8 + 12 - 6\sqrt{3} = 4$
$-4 \le a_j \le +4$
となり詳細に調べても変域は変わらない。
つまりb_{j+1}とb_jは完全に独立変数である。

〈３･７〉c_jの変域　　c_jの変域の変域を調べる
c_j の最大値は

$$c_j = d_{j+1} - d_j - \frac{1}{3}b_{j+1} - \frac{2}{3}b_j$$

$$^{b_{j+1}}/_3 = -(\sqrt{3}-1)d_{j+1} - \sqrt{3}\alpha(f_{j+2} + e_j)$$

$$^{b_j}/_3 = -(\sqrt{3}-1)d_j - \sqrt{3}\alpha(f_{j+1} + e_{j-1})$$

を代入して
$c_j = d_{j+1} - d_j + (\sqrt{3}-1)d_{j+1} + \sqrt{3}\alpha(f_{j+2} + e_j) + 2(\sqrt{3}-1)d_j + 2\sqrt{3}\alpha(f_{j+1} + e_{j-1})$
$f_{j+1} = d_{j+1} + \alpha f_{j+2}$
$e_j = d_j + \alpha e_{j-1}$
を代入すると
$c_j = d_{j+1} - d_j + (\sqrt{3}-1)d_{j+1} + \sqrt{3}\alpha(f_{j+2} + d_j + \alpha e_{j-1}) + 2(\sqrt{3}-1)d_j + 2\sqrt{3}\alpha(d_{j+1} + \alpha f_{j+2} + e_{j-1})$
$c_j = (1 + \sqrt{3} - 1 + 2\sqrt{3}\alpha)d_{j+1} + (-1 + \sqrt{3}\alpha + 2\sqrt{3} - 2)d_j + (\sqrt{3}\alpha + 2\sqrt{3}\alpha^2)f_{j+2} + (\sqrt{3}\alpha^2 + 2\sqrt{3}\alpha)e_{j-1}$
$= -(6\sqrt{3}-9)d_{j+1} + 0d_j + (21 - 12\sqrt{3})f_{j+2} - (6 - 3\sqrt{3})e_{j-1}$
最大値は

$$= 6\sqrt{3} - 9 + {}^{(27-15\sqrt{3})(\sqrt{3}+1)}/_2$$
$$= 12\sqrt{3} - 18 = 2.784609691$$

つまりc_jの見かけの最大値は８で＋3bit 必要だが変域としては＋２bit でよい。

$$-(12\sqrt{3} - 18) \le c_j \le (12\sqrt{3} - 18) \cong 2.784609691$$

〈３･８〉スプライン関数の最大値
$S_j(x) = a_j x^3 + b_j x^2 + c_j x + d_j$
$x = 0.5$ で最大値を取るとして
$S_j(0.5) = a_j 0.125 + b_j 0.25 + c_j 0.5 + d_j$
の最大値を求める。

$$S_jMax = \frac{1}{3}(b_{j+1} - b_j) \times 0.125 + b_j \times 0.25 + \left(d_{j+1} - d_j - \frac{1}{3}b_{j+1} - \frac{2}{3}b_j\right) \times 0.5 + d_j$$

$$S_jMax = {}^{b_{j+1}}/_{24} - {}^{b_j}/_{24} + {}^{b_j}/_4 + {}^{d_{j+1}}/_2 - {}^{d_j}/_2 - {}^{b_{j+1}}/_6 - {}^{b_j}/_3 + d_j$$

$$S_jMax = {}^{b_{j+1}}/_{24} - {}^{b_{j+1}}/_6 - {}^{b_j}/_{24} + {}^{b_j}/_4 - {}^{b_j}/_3 + {}^{d_{j+1}}/_2 + {}^{d_j}/_2$$

$$S_jMax = -{}^{b_{j+1}}/_8 - {}^{b_j}/_8 + {}^{d_{j+1}}/_2 + {}^{d_j}/_2$$

この時点で最大値は 2.5　さて

$$^{b_{j+1}}/_3 = -(\sqrt{3}-1)d_{j+1} - \sqrt{3}\alpha(f_{j+2} + e_j)$$

$$b_j/_3 = -(\sqrt{3}-1)d_j - \sqrt{3}\alpha(f_{j+1} + e_{j-1})$$

であるから

$$b_{j+1}/_3 + b_j/_3 = -(\sqrt{3}-1)d_{j+1} - \sqrt{3}\alpha(f_{j+2} + e_j) - (\sqrt{3}-1)d_j - \sqrt{3}\alpha(f_{j+1} + e_{j-1})$$
$$= -(\sqrt{3}-1)(d_{j+1} + d_j) - \sqrt{3}\alpha(f_{j+2} + e_j + f_{j+1} + e_{j-1})$$

ここに

$f_{j+1} = d_{j+1} + \alpha f_{j+2}$

$e_j = d_j + \alpha e_{j-1}$

を代入して

$$b_{j+1}/_3 + b_j/_3 = -(\sqrt{3}-1)(d_{j+1} + d_j) - \sqrt{3}\alpha(f_{j+2} + d_j + \alpha e_{j-1} + d_{j+1} + \alpha f_{j+2} + e_{j-1})$$
$$= -(\sqrt{3} - 1 + \sqrt{3}\alpha)(d_{j+1} + d_j) - \sqrt{3}\alpha(1+\alpha)(f_{j+2} + e_{j-1})$$
$$= -(2-\sqrt{3})(d_{j+1} + d_j) + (9 - 5\sqrt{3})(f_{j+2} + e_{j-1})$$

$$S_jMax = \frac{3}{8}\left((2-\sqrt{3})(d_{j+1} + d_j) - (9-5\sqrt{3})(f_{j+2} + e_{j-1})\right) + d_{j+1}/_2 + d_j/_2$$

$$S_jMax = \frac{10-3\sqrt{3}}{8}(d_{j+1} + d_j) + \frac{3(9-5\sqrt{3})}{8}(f_{j+2} + e_{j-1})$$

ここで

$d_{j+1} = d_j = 1$

$f_{j+2} = e_{j-1} = \frac{\sqrt{3}+1}{2}$

を代入して

$$S_jMax = \frac{11}{8} = 1.375$$

を得る。CD のデジタルデータが自然音を録音したものならばスプライン関数の値も 1 を超えることはほぼないといえるが、これが編集されたものなら 1 を超える可能性がある。よってスプライン関数を計算した後計算データを$8/_{11}$倍すれば DAC が飽和することはなくなる。簡単な飽和対策としてはデータをすべて$1/_2$倍か$3/_4$倍すればよい。

３． 考察

スーパーサンプリング DAC はサンプル点以外のところで極値を持つことがあり、この極値は最大でサンプル点の最大値の 1.375 倍になる。この特徴は通常の DAC がサンプル点以外では極値を持たないのに対して大きな違いである。DAC を波形の伝送再生という観点で見ると元のアナログデータを AD 変換した時にアナログデータのピーク値とサンプル点が一致している保証はないので、サンプル点以外のタイミングでもピーク値がとれるスプライン関数の方が波形伝送に適しているともいえる。

スプライン関数が AD 変換の最大値を超すデータを吐き出す現象には注意が必要である。最近の流行として録音レベルを上げてデジタルデータを意図的にクリップさせた録音は多く、このようなデータが入力されると通常の DAC では DC が出てしまう。この場合には録音が DC なので再生される波形に DC が含まれるのはやむを得ない。

しかし波形の一部がピーク値を超えた場合にはそのピーク値は再生した方がいいかもしれない。このあたりは聴き手の好みになるので選択できるようにするのが好ましい。ピーク値を再生しようとすると DAC のダイナミックレンジは広がるが元のデータが 16 ビットで、DAC も 16 ビットの場合は最下位が桁落ちして 15 ビットしか再生できないことになる。ピーク値を捨ててクリップさせるとビット落ちは起きないが、ごくまれにクリップする音を聞かないといけないことになる。それが CD の編集者の意図と考えて不愉快なクリップ音を甘受するのも一法である。

今回の変域の解析でスプライン関数の変域が明快になり、演算回路の MSB 側、LSB 側に必要なビットマージンが明確になった。

これは将来のハードウェア設計に反映させたい。

３． 謝辞

無限スプライン関数の動作原理について監修を頂いた小出昭夫博士に感謝します。

文　　献

（1） トランジスタ技術誌　2018 年 10 月号　45-62P　ディジタル・フィルタレス FPGA　D-A コンバータの製作
（2） トランジスタ技術誌　2018 年 11 月号 付録　DVD　FPGA ロジック・チップ設計データ・サンプル集
（3） エレクトロアートのデジタルオーディオ実験室
http://fpga.cool.coocan.jp/wordpress/?p=1603

（4） 電子情報通信学会 スーパーサンプリング DAC　α = −2 +√3 無限スプライン関数による補間法 小林 芳直†　　肥後 信嗣(2017)
（5） CT-17-058 スーパーサンプリング DAC α = −2+√3 小林 芳直＊ 肥後 信嗣 (2017)