

## サンプリングレート変換のための拡張重複加算法と拡張重複保持法

准 員 村松 正吾<sup>†</sup> 正 員 貴家 仁志<sup>†</sup>

## An Extended Overlap-Add Method and Save Method for Sampling Rate Conversion

Shogo MURAMATSU<sup>†</sup>, Associate Member and Hitoshi KIYA<sup>†</sup>, Member

あらまし 高次の FIR フィルタによる直線畳込みを効果的に実行する一手法として、重複加算法 (OLA) や重複保持法 (OLS) がある。これらの方法は、無限長データに対する直線畳込みを、有限なブロックごとの循環畳込みに帰着させ、その循環畳込みの実行に高速フーリエ変換 (FFT) を利用できる。本論文ではレート変換においても冗長性をなしに直接適用可能な新しい OLA と OLS を提案する。論文では、まず有限長データのもとで、従来の理論を拡張する。次に、無限長データに対してそれを一般化し、本論文の目的である拡張重複加算法と拡張重複保持法の二つの方法を提案する。最後に、レート変換の効果的な実現技術として知られるポリフェーズ構成と本方法を比較し、提案法の有効性を示す。

キーワード サンプリングレート変換, 重複加算法, 重複保持法

## 1. まえがき

アナログ信号を介さずに、直接デジタル信号のレートを変換する技術がデジタル信号処理の分野において不可欠なものとなっている<sup>(1),(2)</sup>。この技術は単なるレート変換技術にとどまらず、画像や音声のサブバンド符号化<sup>(3),(4)</sup>や適応信号処理<sup>(5),(6)</sup>、A-D/D-A 変換技術<sup>(7)</sup>など幅広い分野において基礎技術となっている。

このレート変換の実現において、重要な処理がフィルタリングである。フィルタリングは、レート変換に伴う演算の大部分を占め、また、使用されるフィルタの特性が、レート変換器の特性を決定してしまう。従って、特性の良い高次のフィルタリングをいかに少ない演算量で実行するかが重要な課題となっている。

この問題を解決する際の重要な点に、演算の冗長性の回避がある。一般にレート変換を実現した場合に、フィルタへの入力の多くはゼロ値をもち、しかもフィルタ出力の多くは捨てられてしまう。

この冗長演算を排除するものとして、ポリフェーズ構成や時変 FIR 構成が最も典型的な技術として知られている<sup>(1),(2)</sup>。しかし、これらの技術は、時間領域で畳

込み演算を行っているため、演算量がフィルタのタップ長に比例して増加してしまう。そのほかにも、FFT プルーニング技術の補間処理への応用がある<sup>(10)~(13)</sup>。この技術は、演算量の低減に関しては非常に効果的であるが、FFT 点数以内のデータ長しか扱えない。そのため、一般的なレート変換法としては使用できない。

そこで本論文では、フィルタリングの効果的な実現法である重複加算法 (OverLap-Add method: OLA)<sup>(8),(9)</sup>や重複保持法 (OverLap-Save method: OLS)<sup>(8),(9)</sup>をレート変換処理のために一般的に拡張する。まず 2. では、本論文で必要となる基本技術の概要を述べる。3. では、データを有限と仮定し、レート変換の冗長性を排除する方法を提案する。4. では、この方法に基づいて、OLA と OLS を拡張した新しいレート変換法を提案する。5. では、提案法の演算量について検討し、その有効性を示す。

## 2. 基本技術の概要

準備として、離散フーリエ変換 (Discrete Fourier Transform: DFT) の定義を示し、従来の重複加算法 (OLA) と重複保持法 (OLS) について簡単に説明する。そしてサンプリングレート変換の基本について述べ、OLA や OLS のレート変換への拡張という本論文の目的を明確にする。

<sup>†</sup> 東京都立大学工学部電子情報工学科, 八王子市 Faculty of Technology, Tokyo Metropolitan University, Hachioji-shi, 192-03 Japan

## 2.1 離散フーリエ変換 (DFT)

後の議論のために, DFT と IDFT (Inverse DFT) の定義を示しておく.

今, 長さ  $N$  のデータ列  $f(n)$ ,  $n=0, 1, 2, \dots, N-1$  を用いて, 次式の 1 次変換が実行されるとき,  $F(k)$  は  $f(n)$  の DFT であると言う.

$$F(k) = \sum_{n=0}^{N-1} f(n) W_N^{kn} \quad k=0, 1, 2, \dots, N-1 \quad (1)$$

ここで,

$$W_N = e^{-j\frac{2\pi}{N}} \quad (2)$$

である. また,  $f(n)$  と  $F(k)$  は,

$$f(n) = \frac{1}{N} \sum_{k=0}^{N-1} F(k) W_N^{-nk} \quad n=0, 1, 2, \dots, N-1 \quad (3)$$

のような 1 次変換の関係にあり,  $f(n)$  は  $F(k)$  の IDFT であると言う.

## 2.2 重複加算/保持法<sup>(8),(9)</sup>

ここでは, OLA と OLS の概要について述べる.

OLA と OLS は, それぞれ FIR フィルタによる直線畳込みを実行する一手法としてよく知られている. それらは, 無限長データに対する直線畳込みを, 有限なブロックごとの循環畳込みを介して実行するものである. ここで, 無限長データとは, 非常に長く続くデータのことであり, 便宜上, 本論文ではそれを無限長データと呼ぶ.

循環畳込みは, DFT によって実現でき, 更に DFT は, 高速フーリエ変換 (Fast Fourier Transform: FFT) を用いて実行できる. 従って, OLA や OLS は, 高次の FIR フィルタリングに対して効果的な実現を与える.

以下では, 無限長の入力データ  $v(r)$  と  $L_H$  個のインパルス応答をもつフィルタ  $h(r)$  に対し, その処理手順を示そう. 但し,  $v(r)$  および  $h(r)$  は時間ゼロから始まる数列と仮定する.

### 2.2.1 重複加算法 (OLA)

図 1 に, OLA の処理手順を図的にまとめている.

(STEP 1) 無限長入力データ  $v(r)$  を  $L_s$  点ごとに区分し, 式(4)のようにゼロ値を挿入し, 長さ  $L \geq L_s$  のブロック入力データ  $v_i(r')$  をつくる.

$$v_i(r') = \begin{cases} v(iL_s + r') & 0 \leq r' \leq L_s - 1 \\ 0 & L_s \leq r' \leq L - 1 \end{cases} \quad i=0, 1, 2, 3, \dots \quad (4)$$

ここで,  $i$  はブロック番号を示す. 但し, ブロック点数  $L$  とデータ区分点数  $L_s$  は, 次の関係を満たす必要がある.

$$L \geq L_s + L_H - 1 \quad (5)$$

同様に, FIR フィルタ  $h(r)$  のインパルス応答の後に  $L - L_H$  個のゼロ値を挿入し, FIR フィルタもブロック点数  $L$  と同じ個数のインパルス応答をもつと考える.

(STEP 2) ブロック入力データ  $v_i(r')$  とゼロ値を付加されたフィルタ  $h(r')$  との  $L$  点循環畳込みを式(6)のように実行し, ブロック出力データ  $u_i(r')$  を得る.

$$u_i(r') = \sum_{n=0}^{L-1} h((n)_L) v_i((r'-n)_L) \\ = h(r') \textcircled{L} v_i(r') \quad 0 \leq r' \leq L-1 \quad (6)$$

但し, 関数  $(\cdot)_L$  は,  $L$  を法とする引数の剰余を示す. また,  $\textcircled{L}$  は周期  $L$  の循環畳込みを行う演算子を意味する. 今,  $H(l)$ ,  $V_i(l)$ ,  $U_i(l)$  がそれぞれ,  $h(r')$ ,  $v_i(r')$ ,  $u_i(r')$  の  $L$  点 DFT であるとする, 各 DFT 係数は, 次のような関係をもつ.

$$U_i(l) = H(l) V_i(l) \quad (7)$$

(STEP 3) ブロック出力データ  $u_i(r')$  を次式のように, 他のブロック出力データと重複加算しながら連結する.

$$u(r) = \sum_{i=0}^{\infty} u_i(r) \quad r=0, 1, 2, 3, \dots \quad (8) \\ u_i(r) = \begin{cases} u_i(r - iL_s) & iL_s \leq r \leq (i+1)L_s + L_H - 2 \\ 0 & \text{otherwise} \end{cases} \quad (9)$$

最終出力  $u(r)$  は,  $v(r)$  と  $h(r)$  の直線畳込みの結果と一致する<sup>(8),(9)</sup>. すなわち,

$$u(r) = \sum_{n=-\infty}^{\infty} h(n) v(r-n) \\ = h(r) * v(r) \quad -\infty < r < \infty \quad (10)$$

となっている. 但し,  $*$  は, 直線畳込みを行う演算子とする.

ブロック出力データ  $u_i(r')$  の重複加算の点数は, 本来  $L - L_s$  点であるが,  $u_i(r')$  は,  $L_s + L_H - 1 \leq r' \leq L - 1$  の範囲で必ずゼロ値となるため, 実際には,  $L > L_s + L_H - 1$  の場合においても, 重複加算の点数は, 常に  $L_H - 1$  点となる. 従って, 図 1 では重複加算の点数を  $L_H - 1$  としている.

### 2.2.2 重複保持法 (OLS)

無限長データに対する FIR フィルタリングをブロック処理する方法として OLA をまとめた. 同じ目的の他の方法に OLS がある. ここではこれを簡単に定式化する. 但し, 後の議論のために, 循環畳込みの周期性を利用して, 通常的方式<sup>(8),(9)</sup> となった表現をとる

が、基本的な処理は異ならないことに注意して欲しい。図2に、OLSの処理手順を図的にまとめている。

(STEP 1) 無限長入力データ  $v(r)$  から、式(11)のように、点数  $L$  のブロック入力データ  $v_i(r')$  を生成する。

$$v_i(r') = \begin{cases} v(iL_S + r') & 0 \leq r' \leq L_S - 1 \\ v(iL_S - L + r') & L_S \leq r' \leq L - 1 \end{cases} \quad i=0, 1, 2, 3, \dots \quad (11)$$

ここで、隣接するブロックデータ  $v_{i-1}(r')$  と  $v_i(r')$  の間で、 $L - L_S$  点の重なりがある。この重なり量は、式(5)の条件から決定される。

一方、FIR フィルタ  $h(r)$  は、やはり、 $L - L_H$  点だ

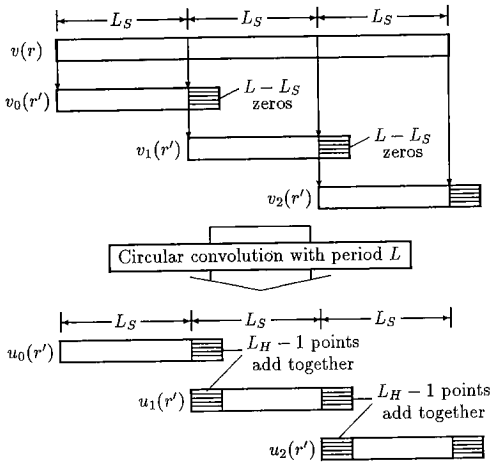


図1 重複加算法  
Fig. 1 The overlap-add method. (OLA)

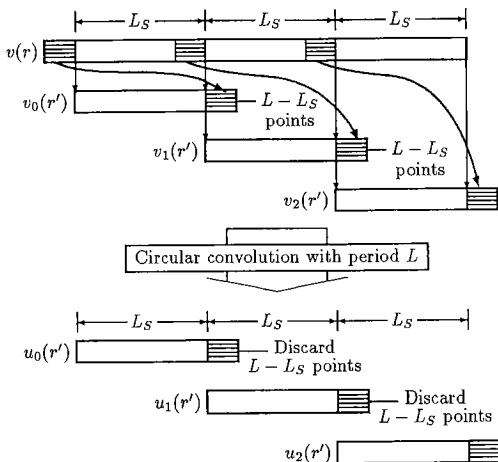


図2 重複保持法  
Fig. 2 The overlap-save method. (OLS)

けゼロ値を付加し、 $L$  点のインパルス応答をもつものとする。

(STEP 2) ブロック入力データ  $v_i(r')$  とゼロ値を付加された FIR フィルタ  $h(r')$  との循環畳込みを式(6)のように実行し、ブロック出力データ  $u_i(r')$  を得る。

(STEP 3) 最終出力  $u(r)$  を得るために、ブロック出力データ  $u_i(r')$  を次式のように合成する。

$$u(r) = \sum_{i=0}^{\infty} u_i(r) \quad r=0, 1, 2, 3, \dots \quad (12)$$

$$u_i(r) = \begin{cases} u_i(r - iL_S) & iL_S \leq r \leq (i+1)L_S - 1 \\ 0 & \text{otherwise} \end{cases} \quad (13)$$

OLS と先の OLA は、ほぼ同じ演算量をもつが、OLS は、最終出力を得るために、各ブロック出力データを加算する必要がないという点で、OLA と異なる<sup>(8),(9)</sup>。本論文の目的は、これら二つの方法をレート変換のために拡張することにある。

### 2.3 サンプリングレート変換

本論文では、図3に示されるような、有理数倍率  $U/D$  のレート変換処理を一般的に考える。但し、 $U$  および  $D$  は、互いに素の関係にある任意の正整数値である。当然、特殊な場合として整数倍 ( $U$  倍) や整数分の1倍 ( $1/D$  倍) のレート変換を含む。また、 $U < D$  の場合、クロックレートを下げる処理となり、 $U > D$  の場合には、クロックレートを上げる処理となる。

図3において、上向き矢印と  $U$  を含む円印はアップサンプル比  $U$  のアップサンプラを、下向き矢印と  $D$  を含む円印はダウンサンプル比  $D$  のダウンサンプラを意味している。各部の信号は、以下の式によって関係づけられる。

$$v(r) = \begin{cases} x\left(\frac{r}{U}\right) & r=nU \\ 0 & \text{otherwise} \end{cases} \quad (14)$$

$$u(r) = h(r) * v(r) \quad (15)$$

$$y(m) = u(mD) \quad (16)$$

### 2.4 レート変換のフィルタリング

図3のフィルタ  $h(r)$  の実行を考える。フィルタ  $h(r)$  の特性は、レート変換の精度を決定する。更に、

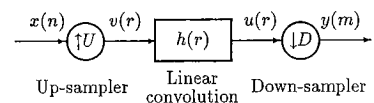


図3  $U/D$  倍サンプリングレート変換  
Fig. 3 Sampling rate conversion by a rational factor  $U/D$ .

その実行は、レート変換における演算の大部分を占める。ここで、レート変換のフィルタリングに関して、以下の二つの特徴を挙げることができる。

- ・高精度なレート変換を実行するために、通常、高次のフィルタが使用される。

- ・アップサンプラにより、フィルタ入力の多くはゼロ値である。また、ダウンサンプラによってフィルタ出力の多くは捨てられる。

後者に関しては、その冗長演算を排除するため、ポリフェーズ構成や時変 FIR 構成といった技術が最も典型的なものとしてよく知られている<sup>(1),(2)</sup>。しかし、これらの技術は、時間領域での畳込み演算を行っているため、演算量がフィルタの次数に比例して増加してしまう。従って、高次のフィルタ処理に対しては、十分な技術とは言えない。

一方、高次のフィルタリングに関しては、先に述べた OLA や OLS が有効であることが知られている。しかし、レート変換がもつ冗長性に関しては、何ら考慮されておらず、それらをレート変換に適用した場合、多くの無駄を含んでしまう。そこで本論文では、図 3 のレート変換において、冗長性を完全に排除した拡張された OLA、拡張された OLS を提案する。

### 3. 有限長データに対するレート変換

図 3 の一般的な実現法を述べる前に、入力信号  $x(n)$  が有限長であると仮定する。今、一般的な入力  $x(n)$  と区別するためにこれを  $\hat{x}(n)$  とおき、長さ  $N_x$  とする。

以下では、以上の条件のもとで、レート変換を DFT 領域で実行する方法を提案する（但し、本論文では、周波数領域の周波数サンプル点に対応する領域を DFT 領域と呼ぶ）。提案する方法は、先に述べたレート変換に伴う演算の冗長性を完全に排除したものである。

#### 3.1 DFT 領域でのレート変換原理

有限長データ  $\hat{x}(n)$  に対するレート変換の原理を図 4(a) に、またそのフィルタリングの DFT 領域での実行原理を図 4(b) に与える。但し、演算の冗長性はまだ排除されていない。

図 4(b) の  $H(l)$ 、 $V(l)$ 、 $U(l)$  は、それぞれ  $h(r)$ 、 $\hat{v}(r)$ 、 $\hat{u}(r)$  の  $L$  点 DFT であり、その DFT 点数  $L$  は、

$$L \geq \max(UN_x, L_H) \tag{17}$$

と選ばれる。

図 4(b) の構成では、DFT 領域において  $L$  点循環

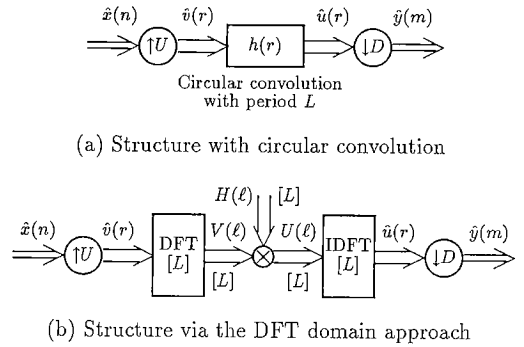


図 4 有限長データ列に対するレート変換  
Fig. 4 Sampling rate conversion for a finite-duration data sequence.

畳込みが実行され、その結果、

$$\hat{u}(r) = h(r) \otimes \hat{v}(r) \tag{18}$$

を得る。そして、この  $\hat{u}(r)$  が  $D$  点ごとに間引かれ、最終出力  $\hat{y}(m)$  を得る。

もし、DFT 点数  $L$  が、

$$L \geq UN_x + L_H - 1 \tag{19}$$

の条件を満たせば、DFT を用いた循環畳込みの結果が直線畳込みの結果と一致するので、最終出力  $\hat{y}(m)$  は、図 3 の構成で DFT を用いずに直接処理したものと一致する。

しかし、図 4 の構成では、フィルタへの入力  $\hat{v}(r)$  の多くはゼロ値であり、また、循環畳込みの結果  $\hat{u}(r)$  の多くは捨てられてしまう。次に、ある条件のもとで、ゼロ値データに対しては、実質的に DFT 処理を行う必要はなく、しかも、ダウンサンプラにより捨てられるデータに対してあらかじめその計算を省略できることを示す。

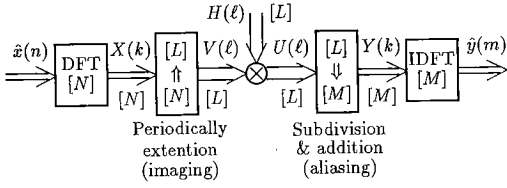
#### 3.2 冗長演算の排除

ここでは、図 4(b) の構成で生じる冗長演算を排除する方法を提案する。結論を先に言えば、循環畳込みの周期  $L$  が式 (17) を満たし、かつ、アップサンプル比  $U$  とダウンサンプル比  $D$  の公倍数に選ばれたとき、図 4(b) の構成を図 5(a) の構成に置き換えることができる。図 5(a) の構成は、異なるサイズの DFT を用いて、冗長演算を排除している。図 5(b) に、この DFT 領域における処理手順を図的に与える。

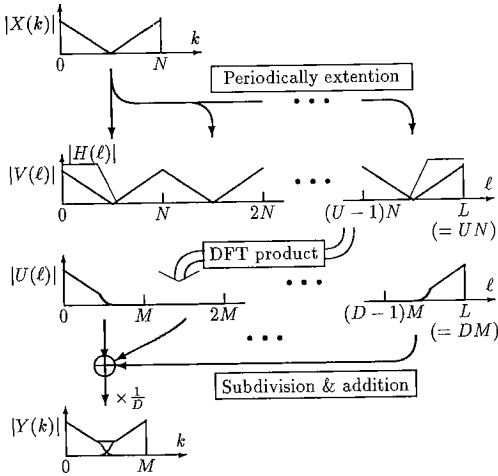
以下に、循環周期  $L$  が  $U$  と  $D$  の公倍数であるとして、図 5(a) の構成によるレート変換の処理手順を示そう。

(STEP 1) 入力  $\hat{x}(n)$  の  $L/U$  点 DFT を求める。ここで、 $L/U$  は整数となるので、これを  $N$  とおく。

STEP1 STEP2 STEP3 STEP4 STEP5



(a) Structure ( $N = L/U, M = L/D$ )



(b) Illustration of the procedure

図 5 DFT 領域でのレート変換

Fig. 5 Sampling rate conversion in DFT domain.

$$X(k) = \sum_{n=0}^{N-1} \hat{x}(n) W_N^{kn} \quad 0 \leq k \leq N-1 \quad (20)$$

(STEP 2)  $X(k)$  を周期的に拡張し、 $V(l)$  を得る。

$$V(l) = X((l)_N) \quad 0 \leq l \leq L-1 \quad (21)$$

但し、関数  $((\cdot))_N$  は、 $N$  を法とする引数の剰余を示す。

(STEP 3)  $H(l)$  と  $V(l)$  の DFT 積をとる。

$$U(l) = H(l) V(l) \quad 0 \leq l \leq L-1 \quad (22)$$

ここで、 $H(l)$  は、 $h(r)$  の  $L$  点 DFT である。

(STEP 4)  $U(l)$  を  $L/D$  点ごとに折り返し加算する。

ここで、 $L/D$  は整数となるので、これを  $M$  とおく。

$$Y(k) = \frac{1}{D} \sum_{d=0}^{D-1} U(dM+k) \quad 0 \leq k \leq M-1 \quad (23)$$

(STEP 5)  $Y(k)$  の  $M(L/D)$  点 IDFT を求める。

$$\hat{y}(m) = \frac{1}{M} \sum_{k=0}^{M-1} Y(k) W_M^{-mk} \quad 0 \leq m \leq M-1 \quad (24)$$

以上の手順によって図 4(b) の構成によるレート変換処理の冗長な演算を排除できる。図 4(b) と図 5(a) の構成を比較すると、DFT 点数は  $1/U$  倍、IDFT 点数は  $1/D$  倍と減っていることがわかる。STEP 2 と

STEP 4 の処理は、図 4(b) の構成では存在しなかったが、スケーリングを除いて乗算なく実現でき、そのスケーリングも式 (22) のフィルタ  $H(l)$  に含めることができることに注意して欲しい。

以下に、この証明を示す。

(証明 1)  $\hat{v}(r)$  の  $L$  点 DFT  $V(l)$  が式 (20)、(21) によって、冗長演算なく求められることを示す。

式 (1) より、アップサンプル後のデータ  $\hat{v}(r)$  の  $L$  点 DFT  $V(l)$  は、

$$V(l) = \sum_{r=0}^{L-1} \hat{v}(r) W_L^{lr} \quad 0 \leq l \leq L-1 \quad (25)$$

と表現できる。今、 $L$  がアップサンプル比  $U$  の倍数であれば、 $N=L/U$  なる整数が存在し、上式は次式のように分解できる。

$$V(l) = \sum_{n=0}^{N-1} \sum_{u=0}^{U-1} \hat{v}(nU+u) W_N^{l(nU+u)} \quad 0 \leq l \leq L-1 \quad (26)$$

更に、アップサンプラの入出力関係の式 (14) より、ゼロ値に対する演算部分を消去すると、

$$V(l) = \sum_{n=0}^{N-1} \hat{x}(n) W_N^{ln} \quad 0 \leq l \leq L-1 \quad (27)$$

という関係が導かれる。但し、 $W_N^{Ul} = W_N^{nl}$  である。ここで  $l$  の範囲以外、右辺の形は  $\hat{x}(n)$  の  $N$  点 DFT となっている。従って、回転子  $W_N$  の周期性により、式 (20)、(21) によって、 $V(l)$  を冗長演算なく得られる。

(証明 2)  $\hat{y}(m)$  が、DFT 係数  $U(l)$  から、式 (23)、(24) によって、冗長演算なく求められることを示す。

式 (3) と、ダウンサンプラの入出力関係の式 (16) より、 $U(l)$  の  $L$  点 IDFT をダウンサンプル比  $D$  でダウンサンプリングした結果  $\hat{y}(m)$  は、

$$\hat{y}(m) = \frac{1}{L} \sum_{l=0}^{L-1} U(l) W_L^{-Dml} \quad 0 \leq m \leq \left\lfloor \frac{L}{D} \right\rfloor - 1 \quad (28)$$

と表現できる。ここで、関数  $\lfloor \cdot \rfloor$  はその引数を超える最小の整数値を示す。今、 $L$  がダウンサンプル比  $D$  の倍数であれば、 $M=L/D$  となる整数が存在し、上式は次式のように分解できる。

$$\hat{y}(m) = \frac{1}{DM} \sum_{d=0}^{D-1} \sum_{k=0}^{M-1} U(dM+k) W_{DM}^{-Dm(dM+k)} \quad 0 \leq m \leq M-1 \quad (29)$$

ここで、式 (23) のように、 $M$  点の DFT 係数  $Y(k)$  を定義すると、上式は、式 (24) のように表現できる。すなわち、最終出力データ  $\hat{y}(m)$  は、今定義した  $Y(k)$  の  $M$  点 IDFT として得られる。但し、 $W_{DM}^{-Dm(dM+k)} = W_M^{-mk}$  である。ここで、ダウンサンプラによって捨てられるデータを求めるのに要する冗長な演算は存在しない

い。

OLA や OLS が、DFT 領域でのフィルタリングに基づいているのと同様、以下で提案する拡張された OLA と拡張された OLS は、ここで提案した DFT 領域でのレート変換を基礎とする。

#### 4. 拡張重複加算/保持法の提案

3.では、有限長データに対する DFT 領域でのレート変換法を提案した。本章では、無限長データに対する一般的なレート変換法に対してその結論を拡張する。

ここで示す二つの提案法を、それぞれ、拡張重複加算法 (Extended OLA: EOLA), 拡張重複保持法 (Extended OLS: EOLS) と呼ぶ。

##### 4.1 OLA/OLS のレート変換への拡張

OLA や OLS をレート変換のフィルタリングへ応用した場合、これらは共通に図 6 のように表現することができる。ここで、図 1 および図 2 で示したように、区分 (Segmentation) とは、無限長データを長さ  $L_s$  のブロックに分割することを意味し、連結 (Linkage) は、各循環畳込みの結果を長さ  $L_s$  ごとに結合する操作を表す。

2.4 で述べたように、図 6 の構成のままでは冗長演算が含まれる。しかし、もし図 7 のように、データの区分をアップサンプラの前に、データの連結をダウンサンプラの後に、それぞれ順序を変えることができれば、この処理を図 4 の有限長データに対するレート変換問題に帰着できる。更に、循環畳込み周期  $L$  を  $U$  と

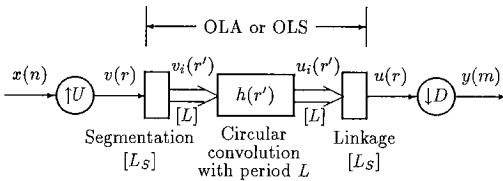


図 6 OLA/OLS によるレート変換

Fig. 6 Structure of sampling rate conversion with OLA or OLS.

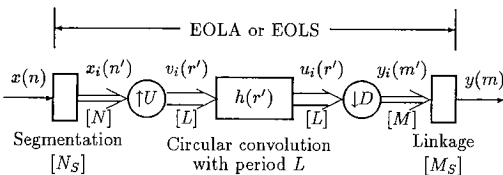


図 7 EOLA/EOLS の基本構成

Fig. 7 Basic structure of EOLA or EOLS.

$D$  の公倍数になるよう選べば、3.2 で提案した図 5 の DFT 領域でのレート変換が利用できるので、冗長演算を排除することができる。従って、無限長データに対する拡張問題は、図 7 の構成を得ることに帰着する。

図 6 の構成を図 7 の構成に置き換えるためには、次の 2 点を考慮しなければならない。

- ・無限長入力データ  $x(n)$  から、ブロック入力データ  $x_i(n')$  をどのように生成するか。

- ・ブロック出力データ  $y_i(m')$  から最終出力データ  $y(m)$  をどのように合成するか。

以上の 2 点を実現する方法を次に提案する。OLA に基づく提案法と OLS に基づく提案法では、その方法が多少異なる。

##### 4.2 拡張重複加算法 (EOLA)

ここでは、新たなサンプリングレート変換法、EOLA を提案する。

まず、図 6 の構成を図 7 の構成に置き換え、かつ演算の冗長性を排除するために、図 6 におけるブロック点数  $L$ 、データ区分点数  $L_s$  の条件を示そう。これらの値は、アップサンプル比  $U$  とダウンサンプル比  $D$  との間に、

$$L = UDK \tag{30}$$

$$L_s = UDK_s \tag{31}$$

が成立しなければならない。但し、 $K$  と  $K_s$  はある正整数であるので、上式は、 $L_s$  および  $L$  を  $U$  と  $D$  の公倍数に選ぶことを意味する。

式 (30) は、3.2 で述べた冗長演算の排除のための条件である。また、式 (31) は、後述するように図 6 を図 7 の処理に置き換えるための必要条件の一つである。

ブロック点数  $L$  は、OLA の使用を想定しているため、更に式 (5) の条件を満たさなければならない。しかし、式 (31) を考慮すれば、より緩和された次式の条件を得ることができる。

$$\begin{aligned} L &\geq (L_s - U + 1) + L_H - 1 \\ &= L_s + L_H - U \end{aligned} \tag{32}$$

なぜならば、アップサンプラのゼロ値挿入により、 $v_i(r')$  の範囲  $0 \leq r' \leq L_s - 1$  の中で、必ず  $v_i(r') = 0$ 、 $L_s - U + 1 \leq r' \leq L_s - 1$  となるからである。すなわち、 $v_i(r')$  の実質的な長さは  $L_s - U + 1$  点とみなすことができる。

以上の条件式 (30) ~ (32) を満たせば、図 6 の構成は図 7 のように置き換えられ、しかも各データブロック  $x_i(n')$  に対し、DFT 領域において図 5 の冗長性のないレート変換を実行できる。図 8 は、この一連の処理を

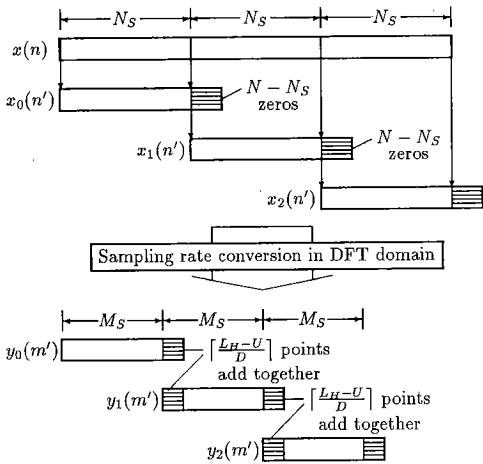


図 8 拡張重複加算法

Fig. 8 The extended overlap-add method. (EOLA)  
( $N=DK, M=UK, N_s=DK_s, M_s=UK_s$ )

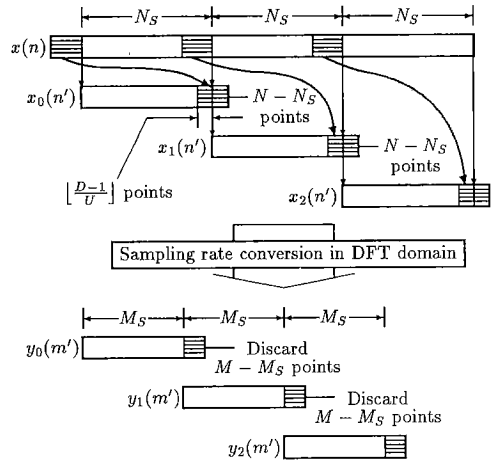


図 9 拡張重複保持法

Fig. 9 The extended overlap-save method. (EOLS)  
( $N=DK, M=UK, N_s=DK_s, M_s=UK_s$ )

概説している。この図に基づき、以下にブロック入力データ  $x_i(n')$  の生成法と、ブロック出力データ  $y_i(m')$  の合成法を具体的に示す。

今、 $N=L/U=DK, N_s=L_s/U=DK_s$  となる正整数が存在し、この  $N$  をブロック点数、 $N_s$  をデータ区分点数とする。

(STEP 1) 無限長入力データ  $x(n)$  から次式によりブロック入力データ  $x_i(n')$  を生成する。

$$x_i(n') = \begin{cases} x(in'_s + n') & 0 \leq n' \leq N_s - 1 \\ 0 & N_s \leq n' \leq N - 1 \end{cases} \quad i=0, 1, 2, 3, \dots \quad (33)$$

(STEP 2) ブロック入力データ  $x_i(n')$  を有限長入力データとして、図 5 の DFT 領域でのレート変換を実行し、ブロック出力データ  $y_i(m')$  を得る。

(STEP 3) ブロック出力データ  $y_i(m')$  から、最終出力  $y(m)$  を次式のように合成する。

$$y(m) = \sum_{i=0}^{\infty} y_i(m) \quad m=0, 1, 2, 3, \dots \quad (34)$$

$$y_i(m) = \begin{cases} y_i(m - iM_s) & \\ iM_s \leq m \leq (i+1)M_s + \left\lceil \frac{L_H - U}{D} \right\rceil - 1 & \\ 0 & \text{otherwise} \end{cases} \quad (35)$$

但し、 $M$  および  $M_s$  は、 $M=L/D=UK, M_s=L_s/D=UK_s$  として与えられる正整数である。

以上のような処理方法を拡張重複加算法 (EOLA) と呼ぶ。式 (30)~(32) は、この EOLA の条件となる。

### 4.3 拡張重複保持法 (EOLS)

ここで、OLS を拡張したサンプリングレート変換法 EOLS を提案する。

図 6 の構成を図 7 のように置き換え、かつ冗長演算を排除するためには、OLA の場合と同様に、図 6 におけるブロック点数  $L$  とデータ区分点数  $L_s$  が、それぞれ式 (30), (31) のように、 $U$  と  $D$  の公倍数となるように選ばばよい。

ブロック点数  $L$  は、OLS の使用を想定しているため、更に式 (5) の条件を満たさなければならない。しかし、式 (31) を考慮すれば、より緩和された次式の条件を得ることができる。

$$L \geq (L_s - D + 1) + L_H - 1 = L_s + L_H - D \quad (36)$$

なぜならば、ダウンサンプラのサンプル棄却により、 $u_i(r')$  の範囲  $0 \leq r' \leq L_s - 1$  のうち、必ず  $L_s - D + 1 \leq r' \leq L_s - 1$  の範囲にある  $D - 1$  点のサンプルは捨てられるからである。すなわち、 $v_i(r')$  のそれに相当するサンプルをあらかじめ捨てることができる。

以上の条件式 (30), (31), (36) を満たせば、図 6 の構成は、図 7 のように置き換えられ、しかも各データブロック  $x_i(n')$  に対し、DFT 領域において図 5 の冗長性のないレート変換を実行できる。図 9 は、この一連の処理を概説している。この図に基づき以下に、ブロック入力データ  $x_i(n')$  の生成法と、ブロック出力データ  $y_i(m')$  の合成法を具体的に示す。

今、 $N=L/U=DK, N_s=L_s/U=DK_s$  となる正整数

が存在し、この  $N$  をブロック点数、 $N_s$  をブロック区分点数とする。

(STEP 1) 無限長入力データ  $x(n)$  から次式によりブロック入力データ  $x_i(n')$  を生成する (付録参照)。

$$x_i(n') = \begin{cases} x(in_s + n') & 0 \leq n' \leq N_s - \lfloor \frac{D-1}{U} \rfloor - 1 \\ x(in_s - N + n') & N_s - \lfloor \frac{D-1}{U} \rfloor \leq n' \leq N-1 \end{cases} \quad i=0, 1, 2, 3, \dots \quad (37)$$

但し、関数  $\lfloor \cdot \rfloor$  はその引数を超えない最大の整数値を示す。

(STEP 2) ブロック入力データ  $x_i(n')$  を有限長入力データとして、図 5 の DFT 領域でのレート変換を実行し、ブロック出力データ  $y_i(m')$  を得る。

(STEP 3) ブロック出力データ  $y_i(m')$  から、最終出力  $y(m)$  を次式のように合成する。

$$y(m) = \sum_{i=0}^{\infty} y_i(m) \quad m=0, 1, 2, 3, \dots \quad (38)$$

$$y_i(m) = \begin{cases} y_i(m - iM_s) & iM_s \leq m \leq (i+1)M_s - 1 \\ 0 & \text{otherwise} \end{cases} \quad (39)$$

但し、 $M$  および  $M_s$  は、 $M=L/D=UK$ 、 $M_s=L_s/D=UK_s$  として与えられる正整数である。

以上のような処理方法を拡張重複保持法 (EOLS) と呼ぶ。式 (30)、(31)、(36) は、この EOLS の条件となる。

## 5. 演算量の検討

本章では、前章で提案した EOLA と EOLS の出力点当りの演算量を示し、ポリフェーズ構成との比較によって、これらの有効性を示す。

### 5.1 演算回数

EOLA と EOLS の出力点当りの実乗算回数 # MUL と実加算回数 # ADD はそれぞれ、式 (40)、(41) のようになる。

$$\# \text{MUL} = \frac{\mu(N) + \mu(M) + 3L}{M_s} \quad (40)$$

$$\# \text{ADD} = \frac{\alpha(N) + \alpha(M) + 3L + 2(D-1)M + \beta}{M_s} \quad (41)$$

但し、

$$\beta = \begin{cases} 2 \lfloor \frac{L_H - U}{D} \rfloor & : \text{EOLA} \\ 0 & : \text{EOLS} \end{cases} \quad (42)$$

とする。また、 $\mu(x)$  と  $\alpha(x)$  はそれぞれ、 $x$  点 DFT

(IDFT) の実乗算の総数と実加算の総数を示しており、使用するアルゴリズムに依存する。ここでは、1 回の複素乗算が、3 回の実乗算と 3 回の実加算で達成される 3/3 アルゴリズムの使用を想定した<sup>(14)</sup>。

式 (40)、(41) において、分子は 1 ブロックを処理するのに要する総演算量となっており、分母は 1 ブロックの実質的な出力点数となっている。当然、分母の値  $M_s (=L_s/D=UK_s)$  はできる限り大きな値に選ぶことが望ましい。また、DFT 点数  $N$  と IDFT 点数  $M$  が、2 のべき乗、あるいは素数の積となっているときは、split-radix<sup>(15)</sup> や prime-factor<sup>(16)</sup> 等の FFT アルゴリズムを使用することが可能となる。このとき、 $\mu(x)$  および  $\alpha(x)$ 、すなわち式 (40)、(41) の演算量を低減できる。 $\mu(x)$  や  $\alpha(x)$  の具体的な算出方法は、各文献を参照されたい<sup>(9)~(17)</sup>。

### 5.2 他方法との比較

ここでは、EOLA と EOLS の有効性を示すため、本提案法とポリフェーズ構成<sup>(1),(2)</sup> の出力点当りの実乗算回数の比較を行う。但し、実信号、実フィルタを仮定する。このとき、DFT 係数の対称性により、式 (40) の分子にある  $3L$  を

$$3L \rightarrow 3 \left( \lfloor \frac{L}{2} \rfloor + 1 \right) \quad (43)$$

と減らせることに注意する。また、変換倍率が  $U/D=1/2$  の場合と  $U/D=2$  の場合についての比較を行う。このとき、循環周期  $L$  を 2 のべき乗値に選択可能であり、DFT と IDFT の計算に、2 のべき乗の FFT アルゴリズムを利用することができる。今回は特に、split-radix RFFT の利用を仮定して、比較を行う。このとき、式 (40) において、

$$\mu(x) = \frac{x}{2} \log_2 x - \frac{3x}{2} + 2 \quad (44)$$

とおける<sup>(15),(17)</sup>。一般に、変換倍率  $U/D$  が  $2^i$  ( $i$ : 整数) のとき、2 のべき乗の FFT アルゴリズムを使用することができる。

図 10(a)(b) に、EOLA のフィルタのタップ長  $L_H$  に対する出力点当りの実乗算回数を示す。但し、ポリフェーズ構成の出力点当りの乗算回数は、 $L_H/U$  とした。また、循環周期  $L$  は、選択の自由度が大きいため、便宜上、式 (30) を満たし、かつ  $L > 4L_H$  を満たす最小の 2 のべき乗値に選んだ。また、 $L_s$  は、式 (31) と式 (32) を満たす最大値に選んだ。ここで、EOLS は、EOLA とほぼ同じ演算量なので割愛せよう。

図 10 の結果より、本提案法は高次のフィルタを使用



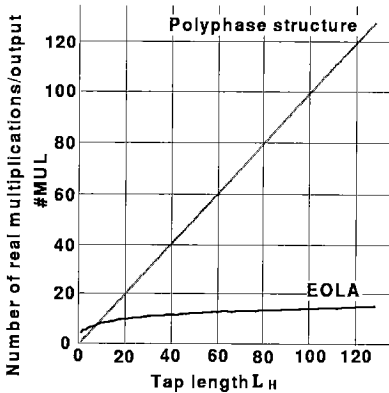
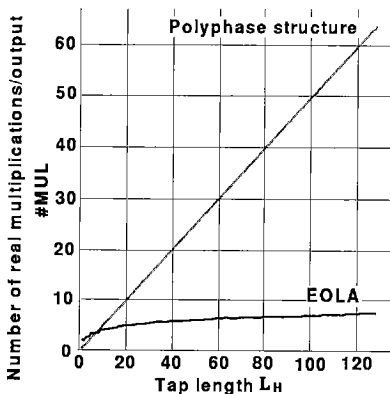
(a) Decimation by an integer factor 2 ( $U = 1, D = 2$ )

図 10 拡張重複加算法の実乗算回数

Fig. 10 Number of real multiplications of EOLA.

したレート変換に対して有効であることがわかる。ここで、出力点当りの演算量# MULと# ADDは、循環周期 $L$ の値の選択に依存する。今回の $L$ の選択( $L > 4L_H$ )は、演算量に関して最適な選択法ではないが、本提案法の有効性を示すために適当な条件のもとで行われた。

## 6. むすび

本論文では、まず、有限長入力データに対するレート変換を冗長演算なくDFT領域において実行する方法を提案した。次に、このレート変換に基づいて、OLAとOLSをレート変換に拡張し、EOLAとEOLSを提案した。最後に、EOLAとEOLSの実乗算回数をポリフェーズ構成のそれと比較し、高次のFIRフィルタを使用したレート変換に対して、本提案法が効果的であることを確認した。EOLAとEOLSは、OLAとOLSの利点を継承し、更に、レート変換に伴う冗長演算を

排除した非常に有効なレート変換法である。

今後は、本提案法のフィルタバンクへの応用、多次元信号処理への拡張等の検討を行う予定である。

## 文 献

- (1) Crochiere R. E. and Rabiner L. R.: "Multirate Digital Signal Processing", Prentice-Hall, Englewood Cliffs NJ (1983).
- (2) Vaidyanathan P. P.: "Multirate Systems and Filter Banks", Prentice-Hall, Englewood Cliffs NJ (1993).
- (3) Woods J. W.: "Subband Coding of Images", Kluwer Academic Publishers (1981).
- (4) Akansu A. N. and Haddad R. A.: "Multiresolution Signal Decomposition", Academic Press, Inc. (1992).
- (5) Gilloire A. and Vetterli M.: "Adaptive Filtering in Subbands", Proc. IEEE ICASSP'88, pp.1572-1575 (1988).
- (6) Gilloire A. and Vetterli M.: "Adaptive Filtering in Subbands with Critical Sampling: Analysis, Experiments, and Application to Acoustic Echo Cancellation", IEEE Trans. Signal Processing, **40**, pp.1862-1875 (Aug. 1992).
- (7) Candy J. C. and Temes G. C.: "Oversampling Delta-Sigma Data Converters", IEEE Press (1992).
- (8) Proakis J. G. and Manolakis D. G.: "Digital Signal Processing", Macmillan, pp.684-750 (1988).
- (9) 佐川雅彦, 貴家仁志: "高速フーリエ変換とその応用", pp.94-116, 昭晃堂 (1992).
- (10) Holm S.: "FFT Pruning Applied to Time Domain Interpolation and Peak Localization", IEEE Trans. ASSP, **ASSP-35**, 12, pp.1776-1777 (Dec. 1987).
- (11) Adams J. W.: "A Subsequence Approach to Interpolation Using the FFT", IEEE Trans. Circuits Syst., **37**, pp.623-625 (May 1990).
- (12) Fraster D.: "Interpolation by the FFT Revisited—An Experimental Investigation", IEEE Trans. ASSP, **ASSP-37**, 5, pp.665-675 (May 1993).
- (13) クックレフトドル, 西原明法: "効率の良い補間と線形畳み込みのための実数値FFTアルゴリズムの枝打ち", 信学技報, **CAS93-14** (1993-05).
- (14) "デジタル信号処理ハンドブック", 電子情報通信学会, p.71 (1993).
- (15) Sorensen H. V., Heideman M. T. and Burrus C. S.: "On computing the split-radix FFT", IEEE Trans. ASSP, **ASSP-34**, 11, pp.152-156 (Feb. 1986).
- (16) Kolba D. P. and Parks D. W.: "A Prime Factor FFT Algorithm Using High-Speed Convolution", IEEE Trans. ASSP, **ASSP-25**, 4, pp.281-294 (Aug. 1977).
- (17) Sorensen H. V., Jones D. L., Heideman M. T. and Burrus C. S.: "Real-Valued Fast Fourier Transform Algorithms", IEEE Trans. ASSP, **ASSP-35**, 6, pp.849-863 (June 1987).

## 付 録

式(37)の導出手順を示す。まず、無限長入力データ  $x(n)$  から、式(A・1)のようにブロック入力データ  $x_i(n')$  を生成する。

$$x_i(n') = \begin{cases} x(iN_s + n') & 0 \leq n' \leq N_s - 1 \\ x(iN_s - N + n') & N_s \leq n' \leq N - 1 \end{cases} \quad (\text{A} \cdot 1)$$

次にこの  $x_i(n')$  をアップサンプル比  $U$  でアップサンプルすると、式(11)で  $L = UN$ ,  $L_s = UN_s$  とおいた結果  $v_i(r')$  と一致する。

$$L_s - D + 1 \leq r' \leq L_s - 1 \quad (\text{A} \cdot 2)$$

の範囲にあるサンプルをあらかじめ捨てることができる。故に、 $x(n)$  を直接ブロック化するには、ある範囲の  $x_i(n')$  のサンプルをあらかじめ捨てることができる。このあらかじめ棄却可能な  $n'$  の範囲は、 $x_i(n') = v_i(n'U)$  という関係と式(A・2)より、 $n'$  が整数であることを考慮して、

$$N_s - \left\lfloor \frac{D-1}{U} \right\rfloor \leq n' \leq N_s - 1 \quad (\text{A} \cdot 3)$$

と与えられる。

式(37)は、式(A・1)で生成したブロック入力データのうち、式(A・3)の範囲のサンプルをあらかじめ棄却した形として与えられている。

(平成5年12月28日受付, 6年3月2日再受付)



村松 正吾

平5都立大・工・電気卒。現在、同大大学院修士課程在学中。マルチレート信号処理、画像解像度変換処理の研究に従事。IEEE 会員。



貴家 仁志

昭55長岡技科大・工・電気電子システム卒。昭57同大大学院修士課程了。同年東京都立大・工・電気工学科助手。現在、同大電子情報工学科助教授。工博。デジタル信号処理、特にマルチレート信号処理とその応用に関する研究に従事。著書「高速フーリエ変換とその応用」、「デジタル信号処理技術入門」。電子画像学会、テレビジョン学会、IEEE 各会員。