(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号

特許第4324200号
(P4324200)

(45)発行日　平成21年9月2日(2009.9.2)　(24)登録日　平成21年6月12日(2009.6.12)

(51)Int.Cl.		F I		
G06F 5/00	(2006.01)	G06F	5/00	H
H03M 7/30	(2006.01)	H03M	7/30	Z
H03M 7/24	(2006.01)	H03M	7/24	
G10L 19/00	(2006.01)	G10L	19/00	220G
H03M 7/36	(2006.01)	H03M	7/36	

請求項の数 30　(全 41 頁)

(21)出願番号　特願2006-540987 (P2006-540987)
(86)(22)出願日　平成17年10月14日 (2005.10.14)
(86)国際出願番号　PCT/JP2005/018978
(87)国際公開番号　WO2006/041165
(87)国際公開日　平成18年4月20日 (2006.4.20)
審査請求日　平成19年2月2日 (2007.2.2)
(31)優先権主張番号　特願2004-329735 (P2004-329735)
(32)優先日　平成16年10月15日 (2004.10.15)
(33)優先権主張国　日本国(JP)
(31)優先権主張番号　特願2004-329736 (P2004-329736)
(32)優先日　平成16年10月15日 (2004.10.15)
(33)優先権主張国　日本国(JP)

(73)特許権者　000004226
日本電信電話株式会社
東京都千代田区大手町二丁目３番１号
(74)代理人　100121706
弁理士　中尾　直樹
(74)代理人　100066153
弁理士　草野　卓
(74)代理人　100128705
弁理士　中村　幸雄
(72)発明者　原田　登
東京都千代田区大手町二丁目３番１号　日本電信電話株式会社内
(72)発明者　関川　浩
東京都千代田区大手町二丁目３番１号　日本電信電話株式会社内

最終頁に続く

(54)【発明の名称】情報符号化方法、復号化方法、共通乗数推定方法、これらの方法を利用した装置、プログラム及び記録媒体

(57)【特許請求の範囲】

【請求項１】

所定区間ごとに複数の入力信号をまとめた列を、共通の乗数とその乗数で各入力信号を除算処理した商信号の列とその残りの誤差信号の列とに分解する分解ステップと、

前記商信号の列を圧縮符号化して商符号を求める商符号化ステップと、

前記商符号と前記乗数と前記誤差信号の列とを出力する出力ステップと

を有する情報符号化方法。

【請求項２】

請求項１記載の情報符号化方法であって、

前記分解ステップは、整数形式の前記商信号の列と浮動小数点形式の前記誤差信号の列とに分解するステップであり、

前記誤差信号の列を圧縮符号化して誤差符号を求める誤差符号化ステップも有し、

前記出力ステップが、前記商符号と前記乗数と前記誤差符号とを出力するステップである

ことを特徴とする情報符号化方法。

【請求項３】

請求項１記載の情報符号化方法であって、

前記分解ステップは、浮動小数点形式の前記商信号の列と浮動小数点形式の前記誤差信号の列とに分解するステップであり、

前記浮動小数点形式の各商信号を、整数形式の商信号と浮動小数点形式の差分信号とに

分離し、整数形式の商信号の列と浮動小数点形式の差分信号の列とを求める商信号分離ステップも有し、
前記商符号化ステップは、前記整数形式の商信号の列を圧縮符号化するステップであり、
前記差分信号の列を圧縮符号化して差分符号を求める差分符号化ステップも有し、
前記誤差信号の列を圧縮符号化して誤差符号を求める誤差符号化ステップも有し、
前記出力ステップが、前記商符号と前記差分符号と前記乗数と前記誤差符号とを出力するステップである
ことを特徴とする情報符号化方法。
【請求項４】
所定の区間ごとに複数の浮動小数点形式の入力信号をまとめた列を符号化する情報符号化方法であって、
共通乗数推定部で、前記入力信号列を構成する入力信号に共通する乗数（共通乗数）を推定する共通乗数推定ステップと、
除算処理部で、当該乗数で前記入力信号列を構成する各入力信号を除算処理した商信号の列を求める除算処理ステップと、
変形処理部で、前記商信号の列を構成する各商信号に丸め処理を行った整数形式の列（整数形式商信号列）を求める変形処理ステップと、
商信号符号化部で、前記整数形式商信号列を符号化する商符号化ステップと、
乗算部で、前記整数形式商信号列の各信号に前記共通乗数を乗算した浮動小数点形式の信号の列（浮動小数点形式信号列）を求める乗算ステップと、
誤差符号化部で、前記浮動小数点形式信号列の各信号と入力信号列の各信号との差である誤差信号の列を符号化する誤差符号化ステップと
を有する情報符号化方法。
【請求項５】
所定の区間ごとに複数の浮動小数点形式の入力信号をまとめた列（入力信号列）を符号化する情報符号化方法であって、
共通乗数推定部で、前記入力信号列を構成する入力信号に共通する乗数（共通乗数）を推定する共通乗数推定ステップと、
除算処理部で、当該乗数で前記入力信号列を構成する各入力信号を除算処理した商信号の列を求める除算処理ステップと、
整数化部で、前記商信号の列を構成する各商信号を整数形式に変換した商信号の列（整数形式商信号列）に変換する整数化ステップと、
商信号符号化部で、前記整数形式商信号列を符号化する商符号化ステップと、
浮動小数点化部で、前記整数形式商信号列を構成する各商信号に基づく浮動小数点形式の商信号の列（浮動小数点形式商信号列）を求める浮動小数点化ステップと、
乗算部で、前記浮動小数点形式商信号列の各信号に前記共通乗数を乗算した浮動小数点形式の信号の列（浮動小数点形式信号列）を求める乗算ステップと、
誤差符号化部で、前記浮動小数点形式信号列の各信号と入力信号列の各信号との差である誤差信号の列を符号化する誤差符号化ステップと
を有する情報符号化方法。
【請求項６】
請求項４記載の情報符号化方法であって、
前記共通乗数推定ステップが、
前記入力信号列を構成する入力信号の値から代表値を選定し、
その代表値を分母とし、前記入力信号列を構成する各入力信号の値を分子とし、これら分数の中の、前記各数値の誤差に基づく正側誤差と負側誤差間に存在する分母最小の既約分数を有理近似計算により求め、
これら求めた既約分数中の共通分母を求め、
その共通分母と前記代表値とから前記入力信号列を構成する各入力信号の値に対する共

通の乗数を求める
ことを特徴とする情報符号化方法。
【請求項７】
請求項４または６記載の情報符号化方法であって、
前記変形処理ステップが、
前記各商信号に丸め処理を行った後の各商信号に、
あらかじめ定めた範囲で
最下位ビットに１ずつの加算または減算を行い、
入力信号から、前記の加算または減算された商信号と前記共通乗数の積を減算した結果が０となる商信号を探索する探索サブステップと
前記探索サブステップの結果、全ての商信号に対して前記探索が成功した場合には、当該商信号の集合を前記整数形式商信号列とし、
前記探索の一部が失敗した場合には、共通乗数を１として前記入力信号そのものを商信号とする商信号変更サブステップと
を有する
こと特徴とする情報符号化方法。
【請求項８】
入力符号から商符号と乗数と誤差信号列とを分離する分離ステップと、
前記商符号を復号して商信号列を生成する商信号列復号ステップと、
前記商信号列を構成する各商信号に前記乗数を乗算する乗算ステップと、
前記乗算ステップで生成した信号列の各信号を、前記誤差信号列を構成する各誤差信号で補正して、復号化信号の列を得る補正ステップと
を有する情報復号化方法。
【請求項９】
請求項８記載の方法において、
前記分離ステップは、入力符号から商符号と乗数と誤差符号を分離するステップであり、
前記商信号列復号ステップは、整数形式の商信号列を生成するステップであり、
前記誤差符号を復号して浮動小数点形式の誤差信号列を生成する誤差信号列復号ステップも有し、
前記乗算ステップは、前記整数形式の商信号列の各商信号に前記乗数を乗算して浮動小数点形式の信号列を生成するステップである、
ことを特徴とする情報復号化方法。
【請求項１０】
前記請求項８の方法において、
前記分離ステップは、入力符号から商符号と差分符号と乗数と誤差符号を分離するステップであり、
前記商信号列復号ステップは、整数形式の商信号列を生成するステップであり、
前記差分符号を復号して浮動小数点形式の差分信号列を生成する差分信号列復号ステップも有し、
前記整数形式の商信号列の各商信号と前記差分信号列の対応する差分信号とを加算して浮動小数点形式の商信号の列を求める加算ステップも有し、
前記誤差符号を復号して浮動小数点形式の誤差信号列を生成する誤差信号列復号ステップも有し、
前記乗算ステップは、前記浮動小数点形式の商信号の列の各商信号に前記乗数を乗算して浮動小数点形式の信号列を生成するステップである、
ことを特徴とする情報復号化方法。
【請求項１１】
商復号化部で、商符号を整数形式の商信号列（整数形式商信号列）に復号する商復号化ステップと、

誤差復号化部で、誤差符号に基づいて誤差信号列を得る誤差復号化ステップと、
乗算部で、前記整数形式商信号列を構成する各信号に共通乗数を乗算して、浮動小数点形式の信号列を得る乗算ステップと、
前記乗算ステップで生成した信号列の各信号に、前記誤差信号列を構成する各信号を加算する加算ステップと
を有する情報復号化方法。
【請求項１２】
商復号化部で、商符号を整数形式の商信号列（整数形式商信号列）に復号する商復号化ステップと、
誤差復号化部で、誤差符号に基づいて誤差信号列を得る誤差復号化ステップと、
浮動小数点化部で、前記整数形式商信号列を構成する各信号に基づく浮動小数点形式の商信号の列（浮動小数点形式商信号列）を生成する浮動小数点化ステップと、
乗算部で、前記浮動小数点形式商信号列を構成する各信号に共通乗数を乗算して、浮動小数点形式の信号列を得る乗算ステップと、
前記乗算ステップで生成した信号列の各信号に、前記誤差信号列を構成する各信号を加算する加算ステップと
を有する情報復号化方法。
【請求項１３】
請求項１１または１２記載の情報復号化方法であって、
前記誤差復号化ステップが、
共通乗数が１の場合には、各誤差信号の仮数部として、前記整数形式商信号列の対応する商信号の桁数を仮数部の総桁数から減算した数分の下位側のビットに、誤差符号を復号して得られた信号を設定し、残りの上位側のビットに“０”を設定して、浮動小数点表現の誤差信号列を求め、
共通乗数が１以外の場合には、誤差符号から求めた各信号を、浮動小数点表現の各誤差信号の仮数部として、浮動小数点表現の誤差信号列を求める
処理を含む
ことを特徴とする情報復号化方法。
【請求項１４】
所定区間ごとに複数の入力信号をまとめた入力信号列を生成する分割部と、
前記入力信号列が入力され、その入力信号列の各入力信号に共通の乗数と、その乗数で各入力信号を除算処理した商信号の列とその残りの誤差信号の列とに分解する剰余分離処理部と、
前記商信号の列を圧縮符号化して商符号を求める商符号化部と、
前記商符号と前記乗数と前記誤差信号の列とを出力する出力部と
を具備する情報符号化装置。
【請求項１５】
請求項１４に記載の装置において、
前記剰余分離処理部は、整数形式の前記商信号の列と浮動小数点形式の前記誤差信号の列とに分解する構成部であり、
前記誤差信号の列を圧縮符号化して誤差符号を求める誤差符号化部も有し、
前記出力部は、前記商符号と前記乗数と前記誤差符号とを出力する構成部である
ことを特徴とする情報符号化装置。
【請求項１６】
請求項１４に記載の装置において、
前記剰余分離処理部は、浮動小数点形式の前記商信号の列と浮動小数点形式の前記誤差信号の列とに分解する構成部であり、
前記浮動小数点形式の各商信号を、整数形式の商信号と浮動小数点形式の差分信号とに分離し、整数形式の商信号の列と浮動小数点形式の差分信号の列とを求める整数化部も有し、

前記商符号化部は、前記整数形式の商信号の列を圧縮符号化する構成部であり、
前記差分信号の列を圧縮符号化して差分符号を求める差分符号化部も有し、
前記誤差信号の列を圧縮符号化して誤差符号を求める誤差符号化部も有し、
前記出力部が、前記商符号と前記差分符号と前記乗数と前記誤差符号とを出力する構成部である
ことを特徴とする情報符号化装置。
【請求項１７】
所定の区間ごとに複数の浮動小数点形式の入力信号をまとめた列を符号化する情報符号化装置であって、
前記入力信号列を構成する入力信号に共通する乗数（共通乗数）を推定する共通乗数推定部と、
当該乗数で前記入力信号列を構成する各入力信号を除算処理した商信号の列を求める除算処理部と、
前記商信号の列を構成する各商信号に丸め処理を行った整数形式の列（整数形式商信号列）を求める変形処理部と、
前記整数形式商信号列を符号化する商符号化部と、
前記整数形式商信号列の各信号に前記共通乗数を乗算した浮動小数点形式の信号の列（浮動小数点形式信号列）を求める乗算部と、
前記浮動小数点形式信号列の各信号と入力信号列の各信号との差である誤差信号の列を符号化する誤差符号化部と
を備える情報符号化装置。
【請求項１８】
所定の区間ごとに複数の浮動小数点形式の入力信号をまとめた列（入力信号列）を符号化する情報符号化装置であって、
前記入力信号列を構成する入力信号に共通する乗数（共通乗数）を推定する共通乗数推定部と、
当該乗数で前記入力信号列を構成する各入力信号を除算処理した商信号の列を求める除算処理部と、
前記商信号の列を構成する各商信号を整数形式に変換した商信号の列（整数形式商信号列）に変換する整数化部と、
前記整数形式商信号列を符号化する商符号化部と、
前記整数形式商信号列を構成する各商信号に基づく浮動小数点形式の商信号の列（浮動小数点形式商信号列）を求める浮動小数点化部と、
前記浮動小数点形式商信号列の各信号に前記共通乗数を乗算した浮動小数点形式の信号の列（浮動小数点形式信号列）を求める乗算部と、
前記浮動小数点形式信号列の各信号と入力信号列の各信号との差である誤差信号の列を符号化する誤差符号化部と
を備える情報符号化装置。
【請求項１９】
請求項１７記載の情報符号化装置であって、
前記共通乗数推定部が、
前記入力信号列を構成する入力信号の値から代表値を選定し、
その代表値を分母とし、前記入力信号列を構成する各入力信号の値を分子とし、これら分数の中の、前記各数値の誤差に基づく正側誤差と負側誤差間に存在する分母最小の既約分数を有理近似計算により求め、
これら求めた既約分数中の共通分母を求め、
その共通分母と前記代表値とから前記入力信号列を構成する各入力信号の値に対する共通の乗数を求める
ことを特徴とする情報符号化装置。
【請求項２０】

請求項１７または１９記載の情報符号化装置であって、
前記変形処理部が、
前記各商信号に丸め処理を行った後の各商信号に、
あらかじめ定めた範囲で
最下位ビットに１ずつの加算または減算を行い、
入力信号から、前記の加算または減算された商信号と前記共通乗数の積を減算した結果が０となる商信号を探索する探索手段と
前記探索サブステップの結果、全ての商信号に対して前記探索が成功した場合には、当該商信号の集合を前記整数形式商信号列とし、
前記探索の一部が失敗した場合には、共通乗数を１として前記入力信号そのものを商信号とする商信号変更手段と
を有する
こと特徴とする情報符号化装置。
【請求項２１】
入力符号から商符号と乗数と誤差信号列とを分離する分離部と、
前記商符号を復号して商信号列を生成する商復号化部と、
前記商信号列を構成する各商信号に前記乗数を乗算する乗算部と、
前記乗算部で生成した信号列の各信号を、前記誤差信号列を構成する各誤差信号で補正して、復号化信号の列を得る補正処理部と
を備えることを特徴とする情報復号化装置。
【請求項２２】
請求項２１記載の装置において、
前記分離部は、入力符号から商符号と乗数と誤差符号を分離する構成部であり、
前記商復号化部は、整数形式の商信号列を生成する構成部であり、
前記誤差符号を復号して浮動小数点形式の誤差信号列を生成する誤差復号化部も有し、
前記乗算部は、前記整数形式の商信号列の各商信号に前記乗数を乗算して浮動小数点形式の信号列を生成する構成部である
ことを特徴とする情報復号化装置。
【請求項２３】
前記請求項２１の装置において、
前記分離部は、入力符号から商符号と差分符号と乗数と誤差符号を分離する構成部であり、
前記商復号化部は、整数形式の商信号列を生成する構成部であり、
前記差分符号を復号して浮動小数点形式の差分信号列を生成する差分復号化部も有し、
前記整数形式の商信号列の各商信号と前記差分信号列の対応する差分信号とを加算して浮動小数点形式の商信号の列を求める加算部も有し、
前記誤差符号を復号して浮動小数点形式の誤差信号列を生成する誤差復号化部も有し、
前記乗算部は、前記浮動小数点形式の商信号の列の各商信号に前記乗数を乗算して浮動小数点形式の信号列を生成する構成部である
ことを特徴とする情報復号化装置。
【請求項２４】
商符号を整数形式の商信号列（整数形式商信号列）に復号する商復号化部と、
誤差符号に基づいて誤差信号列を得る誤差復号化部と、
前記整数形式商信号列を構成する各信号に共通乗数を乗算して、浮動小数点形式の信号列を得る乗算部と、
前記乗算部で生成した信号列の各信号に、前記誤差信号列を構成する各信号を加算する加算部と
を備える情報復号化装置。
【請求項２５】
商符号を整数形式の商信号列（整数形式商信号列）に復号する商復号化部と、

誤差符号に基づいて誤差信号列を得る誤差復号化部と、
前記整数形式商信号列を構成する各信号に基づく浮動小数点形式の商信号の列（浮動小数点形式商信号列）を生成する浮動小数点化部と、
前記浮動小数点形式商信号列を構成する各信号に共通乗数を乗算して、浮動小数点形式の信号列を得る乗算部と、
前記乗算部で生成した信号列の各信号に、前記誤差信号列を構成する各信号を加算する加算部と
を備える情報復号化装置。
【請求項２６】
請求項２４または２５記載の情報復号化装置であって、
前記誤差復号化部が、
共通乗数が１の場合には、各誤差信号の仮数部として、前記整数形式商信号列の対応する商信号の桁数を仮数部の総桁数から減算した数分の下位側のビットに、誤差符号を復号して得られた信号を設定し、残りの上位側のビットに“０”を設定して、浮動小数点表現の誤差信号列を求める手段、
共通乗数が１以外の場合には、誤差符号から求めた各信号を、浮動小数点表現の各誤差信号の仮数部として、浮動小数点表現の誤差信号列を求める手段
を有することを特徴とする情報復号化装置。
【請求項２７】
複数の数値の集団から代表値を選定し、
その代表値を分母とし、前記集団の各数値を分子とし、これら分数の中の、前記各数値の誤差に基づく正側誤差と負側誤差間に存在する分母最小の既約分数を有理近似計算により求め、
これら求めた既約分数中の共通分母を求め、
その共通分母と前記代表値とから前記複数の数値に対し共通の乗数を求めることを特徴とする共通乗数推定方法。
【請求項２８】
複数の数値の集団から代表値を選定する分母選定部と、
前記代表値、前記数値の集団、前記数値の誤差が入力され、前記代表値を分母とし、前記各数値を分子とし、これら分数中の、前記誤差に基づく正側誤差と負側誤差との間に存在する分母最小の既約分数を有理近似計算により求める有理近似計算部と、
前記各既約分数が入力され、共通の分母を求める共通分母確認部と、
前記共通の分母と前記代表値が入力され、共通の分母に対し補正を行って前記数値の集団に対する共通乗数を出力する共通因数計算部と
を具備する共通乗数推定装置。
【請求項２９】
請求項１～１３、２７のいずれかに記載した方法の各過程をコンピュータに実行させるプログラム。
【請求項３０】
請求項２９記載のプログラムを記録したコンピュータ読み取り可能な記録媒体。
【発明の詳細な説明】
【技術分野】
【０００１】
この発明は、音声、音楽などの音響信号や画像信号（以下これらを含めて情報信号という）の複数のサンプル値を通信路により伝送したり、情報記録媒体に記録する場合に伝送効率を高めたり、記録容量を少なくするためなどに用いる情報圧縮符号化方法、その伸張復号化方法、前記情報圧縮符号化に用いる複数の情報集合に対する共通の乗数を推定する方法、これらの方法を利用した装置、プログラム及び記録媒体に関する。
【背景技術】
【０００２】

音響信号データ（サンプル値）に対し歪を許す圧縮符号化方式としてＭＰ３、ＡＡＣ、ＴwinＶＱ等がある。また、画像情報データ（サンプル値）の圧縮符号化方式としてＪＰＥＧ等がある。また歪を許さない可逆な符号化（ロスレス符号化）技術がある（例えば非特許文献１参照）。また編集加工が容易な浮動小数点形式のデータの可逆圧縮も重要である（例えば非特許文献２参照）。また、従来、複数の数値に対し、共通の最大公約数、つまり共通の乗数を求める方法としてユークリッド互除法が知られている。

音響信号の場合には、マイクロホン等で収音した音響信号をサンプリング周波数ｆ、量子化ビット数ｑでサンプリングし、各サンプル値をディジタル値に変換されたディジタルサンプルとして入力される。この場合、各アナログのサンプルに共通の定数を掛け合わせてゲイン調整を行う場合が多い。アナログ－ディジタル変換が行われる前にアナログ領域で定数が掛け合わされ、その後にアナログ－ディジタル変換が行われているような場合には、アナログ信号領域で誤差を含む場合もある。

【０００３】

同様に、画像情報信号を圧縮符号化する際に、２次元の画像信号をラスタ走査して１次元のサンプル列として得る場合にも、元の画像信号を取得する際に、各サンプルに共通の定数を掛け合わせてゲイン調整がなされている場合がある。

元のサンプル列の各サンプルｓ０（ｉ）に、ゲインとして実数Ｇを共通に掛け合わせて得られたサンプルｓ１（ｉ）は、ｓ１（ｉ）＝ｓ０（ｉ）×Ｇとなる。

この実数Ｇを掛け合わせて得られたサンプルｓ１（ｉ）を、２進表現やＩＥＥＥ７５４形式の２進浮動小数点表現で表現し、これらディジタルサンプル列を符号化する場合が多い。ＩＥＥＥ－７５４として標準化されている浮動小数点形式は図１に示すように３２ビットである。上位ビットから極性１ビット、指数部８ビット、仮数部２３ビットで構成される。極性をＳ、指数部８ビットで表す値を１０進数でＥ、仮数部の２進数をＭとすると、この浮動小数点形式の数値は絶対値表現２進数で表わすと式（１）となる。

【０００４】

【数１】

$$(-1)^S \times 1.M \times 2^{E-E_0} \qquad (1)$$

ＩＥＥＥ－７５４によれば、$E_0 = 2^7 - 1 = 127$と決められており、これにより、式（１）中の$E - E_0$は

$$-127 \leqq E - E_0 \leqq 128$$

の範囲の任意の値を取ることができる。ただし、$E - E_0 = 127$の場合はａｌｌ“０”、$E - E_0 = 128$の場合はａｌｌ“１”と定義されている。$E - E_0 = n$は式（１）で表される値の整数部分の桁数（ビット数）から１を減算した値、即ち、最上位の“１”より下位ビット数を表わしている。

【０００５】

従来の圧縮符号化方法、例えば文献１に示す方法は、入力サンプル列の原入力波形（例えば音楽信号波形）の冗長性をなるべく除去して、情報量を減少させるものが主であった。また符号化装置で符号化可能な入力波形の振幅（ビット長）は、例えば１６ビットと決められている。したがって、例えばビット長が２４ビットの場合は上位１６ビットを符号化し、下位の８ビットはそのまま出力するか、別個に圧縮符号化して出力している。

【非特許文献１】Hans M. and Schafer R.W.：Lossless Compression of Digital Audio, IEEE Signal Processing Magazine, Vol.18, No.4, pp.21-32(2001).

【非特許文献２】Dai Yang and Takehiro Moriya：Lossless Compression for Audio Data in the IEEE Floating-Point Format, AES Convention Paper 5987,AES 115th Convention, New York, NY, USA, 2003 OCTOBER 10-13.

【発明の開示】

【発明が解決しようとする課題】

【０００６】

従来の符号化方法（装置）では、予め決められた振幅（ビット長）より大きな入力サンプル列は、符号化できない。符号化可能な振幅（ビット長）に変換して符号化する場合は、符号化品質が劣化する。または符号化できない部分を更に別の情報（符号）とする場合は、情報量が増加するという課題があった。

例えば、このサンプル列を歪を許さないユニバーサル符号化方法で符号化するとし、サンプル列の各サンプルをビット列として見る。ここでサンプル列ｘ（１）～ｘ（３）が、ｘ（１）＝１０１０００００（１０進数１６０），ｘ（２）＝１０００００（１０進数３２），ｘ（３）＝１１１０００００（１０進数２２４）のように０が連続して存在するような場合は、高い圧縮率で符号化することができる。これに対し、ｘ（１）＝１１１１１０１０（１０進数２５０），ｘ（２）＝１１００１０（１０進数５０），ｘ（３）＝１０１０１１１１０（１０進数３５０）のように０と１が混在した複雑な場合は、高い圧縮率で符号化することは困難である。

【０００７】

さらに、従来のユークリッドの互除法は比較的処理が簡単であるが、数値が四捨五入される場合、あるいは誤差を含むような場合は、その誤差を考慮して最大公約数を求めることはできなかった。

この発明の目的は、入力波形の振幅（ビット長）が、圧縮符号化法が可能とする振幅（ビット長）より大きい場合でも、情報量を増加させることなく符号化することができる符号化技術を提供することである。また、入力サンプルを構成するビットの０と１の混在度合いに関わらず効率良くサンプル列を圧縮することができる符号化技術、復号化技術、前記符号化技術に利用する共通乗数を求める技術を提供することである。

【課題を解決するための手段】

【０００８】

この発明では、サンプル列から各サンプルに共通する数（共通の乗数Ａ）を推定する。各サンプルをそれぞれ共通乗数Ａで割算し、その割算結果を圧縮符号化し、その圧縮符号化符号C_Yと、上記推定乗数Ａとを少なくとも出力する。乗数Ａは圧縮符号化してもしなくてもよい。

復号化は、符号C_Yと乗数Ａとを分離し、符号C_Yをその符号化と対応する復号化方法により復号化し、その復号化結果に対し共通乗数Ａを掛け算して、原サンプルを求める。

また、前記圧縮符号化で共通乗数を求める処理では、まず数値集合中から代表数値を決定する。その代表数値で各数値割算した値で正側誤差の最大と負側誤差の最大との間に、正しい値の代表値（誤差を含まない）で各数値の正しい値を割った既約分数を有理近似で求める。そして、この求めた既約分数値中の分母が各数値に対して共通となる最小のものを求め、その共通の分母を前記代表値で修正して共通乗数とする。

【発明の効果】

【０００９】

例えば、アナログレベルで共通のゲインが乗算された圧縮符号化処理対象のサンプル列の場合は、その共通のゲインが共通の乗数と推定され、この乗算により各サンプルが割算される。したがって、振幅が小さくなり、従来の符号化法（例えば非特許文献１）に示す方法をそのまま適用できる場合が多く、かつ品質劣化が少なく、情報量の増加もほとんどない。

また前記割算の結果、元のサンプル列の共通の冗長性を取り除くことができるため、商の集合のエントロピーを大きく減少させることができる。したがって、結果として全体の情報圧縮率を向上することができる。

【００１０】

例えば、浮動小数点表現された音響信号にこの発明を適用した場合、従来方式と比較して、情報量を平均で１／２～１／３に圧縮することが出来る。

さらに、共通乗数を求める処理では、誤差を含む数値の集合に対し、誤差を考慮した共通乗数を求めることができる。

【図面の簡単な説明】

【００１１】
【図１】ＩＥＥＥ－７５４規格の浮動小数点表現説明図。
【図２】符号化装置の実施例１の機能構成例を示す図。
【図３】その処理手順の例を示す流れ図。
【図４】復号化装置の実施例１の機能構成例を示す図。
【図５】その処理手順の例を示す流れ図。
【図６】図６Ａは、この発明の効果を説明するための具体的数値例を示す図であって、サンプル列の例を示す図。図６Ｂは、この発明の効果を説明するための具体的数値例を示す図であって、サンプル列を共通乗数で除算した例を示す図。図６Ｃは、この発明の効果を説明するための具体的数値例を示す図であって、除算されたサンプル列の例を示す図。
【図７】図７Ａは、この発明の効果を説明するための他の具体的数値例を示す図であって、サンプル列の例を示す図。図７Ｂは、この発明の効果を説明するための他の具体的数値例を示す図であって、除算されたサンプル列の例を示す図。
【図８】符号化装置の実施例２の要部機能構成例を示す図。
【図９】その処理手順の例を示す流れ図。
【図１０】復号化装置の実施例２の要部機能構成例を示す図。
【図１１】符号化装置の実施例３の要部機能構成例を示す図。
【図１２】その処理手順の例を示す流れ図。
【図１３】符号化装置の実施例４の要部機能構成例を示す図。
【図１４】その処理手順の例を示す流れ図。
【図１５】符号化装置の実施例５の要部機能構成例を示す図。
【図１６】符号化装置の実施例６の要部機能構成例を示す図。
【図１７】符号化装置の実施例７の要部機能構成例を示す図。
【図１８】復号化装置の実施例７の要部機能構成例を示す図。
【図１９】符号化装置の実施例７の変形例要部機能構成例を示す図。
【図２０】図１９と対応した復号化装置の実施例の機能構成例を示す図。
【図２１】符号化装置の実施例８の機能構成例を示す図。
【図２２】符号化装置の実施例９の機能構成例を示す図。
【図２３】符号化装置の実施例１０の処理手順の例を示す流れ図。
【図２４】復号化装置の実施例１１の機能構成例を示す図。
【図２５】復号化装置の実施例１２の処理手順の例を示す流れ図。
【図２６】図２６Ａは、符号化装置の実施例１３の機能構成例を示す図。図２６Ｂは、その復号化装置の実施例の機能構成例を示す図である。
【図２７】符号化装置の実施例１４の要部機能構成例を示す図。
【図２８】符号化装置の実施例１５の機能構成例を示す図。
【図２９】復号化装置の実施例１６の機能構成例を示す図。
【図３０】符号化装置の実施例１７の機能構成例を示す図。
【図３１】復号化装置の実施例１８の機能構成例を示す図。
【図３２】実施例１９における共通乗数の探索手順の例を示す流れ図。
【図３３】その共通乗数のチェックと商信号の変形例を示す流れ図。
【図３４】商信号変形を浮動小数点形式に対し行う例を示す流れ図。
【図３５】浮動小数点形式を整数形式に変換する処理例を示す流れ図。
【図３６】商信号が正の場合における変形手順の処理例を示す流れ図。
【図３７】商信号が負の場合における変形手順の処理例を示す流れ図。
【図３８】有理近似により共通乗数を求める機能構成例を示す図。
【図３９】その他の例を示す図。
【図４０】更にその他の例を示す図。
【図４１】有理近似により共通乗数を求める機能構成の他の例を示す図。
【図４２】図４２Ａは、自己相関により共通乗数を求める数値例を示す図であって、１ビットシフトした自己相関結果を示す図。図４２Ｂは、自己相関により共通乗数を求める数

値例を示す図であって、自己相関のビットが１の数の例を示す図。

【発明を実施するための最良の形態】

【００１２】

以下この発明の実施形態を図面を参照して説明する。各図において、他の図と対応する部分には同一参照番号を付けて重複説明を省略する。またこの発明を、３２ビットの浮動小数点数で表現された音響信号列に対して適用する場合を例とする。

［実施形態１］

実施例１

符号化処理

図２にこの発明の符号化装置の実施形態１の機能的構成例を、図３にその処理手順例をそれぞれ示す。

【００１３】

入力端子１１よりの入力信号列Ｘ＝（ｘ（１），ｘ（２），…）はディジタル化されたサンプル列であり、これが入力されると（ステップＳ１）、区間（フレーム）分割部１２で所定数Ｎ個（例えば１０２４個）のサンプル列に分割され、図に示していないが一旦記憶部に記憶される（ステップＳ２）。サンプル列は、音響信号の場合は、２４ビットの量子化ビット数で量子化された整数サンプル列や３２ｂｉｔ単精度浮動小数点形式サンプル列などが考えられる。カラー画像信号の場合は各色情報要素に分解してラスタ走査したピクセル情報のディジタル化されたサンプル列がある。

【００１４】

分割区間ごとの入力信号としてのサンプルｘ（ｉ）（ｉ＝０，１，…，Ｎ－１）の集合は、剰余分離処理部１００で乗数Ａと、ｘ（ｉ）を乗数Ａで割った商ｙ（ｉ）とに少なくとも分離される。この例ではｘ（ｉ）をＡで割った余り（誤差）ｚ（ｉ）も分離される。

例えば入力信号列Ｘは、分割された１フレームごとに乗数推定部１１０に渡される。乗数推定部１１０では、全てのサンプルｘ（ｉ）（ｉ＝０，１，…，Ｎ－１）に対して、共通の乗数Ａを推定する（ステップＳ３）。例えば、全てのサンプルｘ（ｉ）が共通の値９９．０で割り切れる場合には、共通の乗数Ａ＝９９．０とする。この共通の乗数Ａを推定する方法については、複数種類の方法が考えられるが、例えば近似共通ファクタ（ＡＣＦ：Approximate Common Factor）の有理近似を用いる。ここでは、適切な乗数Ａが与えられるものとする。

【００１５】

乗数推定部１１０で決定された乗数Ａは、除算処理部１２０、乗算部１３０、Ａ符号化部１５に渡される。

除算処理部１２０では、乗数推定部１１０より渡された乗数Ａと、Ｎ個のサンプルｘ（ｉ）を入力とし、ｎ個の商信号ｙ（ｉ）＝ｘ（ｉ）／Ａを算出する（ステップＳ４）。このとき、ｙ（ｉ）は整数形式、浮動小数点形式、固定小数点形式のいずれでもよい。その決められた表現形式に変換するときに、切り捨、切り上げ、四捨五入、nearest tie to even（ニアレストタイツウイーブン）などの丸め処理を行ってもよい。

【００１６】

例えば、ｘ（ｉ）を倍精度の浮動小数点数（ここでは６４ビットの浮動小数点数）に変換して乗数Ａで除算する。得られた倍精度の商を、最も近い単精度（３２ビット）の浮動小数点数に丸めて商信号ｙ（ｉ）とする。

除算処理部１２０で得られたＮ個の商信号ｙ（ｉ）から成る商信号列Ｙ＝（ｙ（０），ｙ（１），…，ｙ（Ｎ－１））は、Ｙ符号化部１３と、この例では乗算部１３０に渡される。

乗算部１３０では、乗数推定部１１０から渡された乗数Ａを除算処理部１２０から渡された商信号列ＹのＮ個の各信号ｙ（ｉ）にそれぞれ乗算して、復元されたＮ個のサンプルｘ’（ｉ）を得る（ステップＳ５）。

【００１７】

このとき、復元サンプルｘ’（ｉ）は３２ビットの浮動小数点表現で表現可能な値の範

囲に丸められる。例えば、ｙ（ｉ）とＡを掛け合わせた結果を倍精度（６４ビット）の浮動小数点数として保持し、得られた倍精度の乗算結果を、最も近い単精度（３２ビット）の浮動小数点数に丸めてｘ’（ｉ）とする。得られたサンプル列Ｘ’＝（ｘ’（０），ｘ’（１），…，ｘ’（Ｎ－１））は、誤差算出部１４０に渡される。

誤差算出部１４０は、入力信号から取り出したＮ個のサンプルｘ（ｉ）それぞれから乗算部１３０から渡されたＮ個の復元サンプルｘ’（ｉ）を差し引いて、Ｎ個の誤差信号ｚ（ｉ）から成る誤差信号列Ｚ＝（ｚ（０），ｚ（１），…，ｚ（Ｎ－１））を得る（ステップＳ６）。この誤差信号の算出は減算を用いる代わりにｘ（ｉ）とｘ’（ｉ）の３２ビットをそのままビット単位の排他的論理和演算（ｘｏｒ）を行ってもよい。要するにｘ（ｉ）とｘ’（ｉ）との差分演算を行えばよい。

【００１８】

得られた誤差信号列Ｚはこの例ではＺ符号化部１４に渡される。

乗数推定部１１０よりの乗数ＡはＡ符号化部１５にも入力される。

Ｙ符号化部１３、Ｚ符号化部１４、Ａ符号化部１５は、既存の情報圧縮手法を用いてそれぞれの入力信号列Ｙ，Ｚ，Ａを圧縮符号化して得られた符号を符号C_Y、符号C_Z、符号C_Aとして出力する（ステップＳ７）。これら符号C_Y，C_Z，C_Aは合成部１６で１フレームごとのビット列として合成され、符号化データC_Xとして出力される（ステップＳ８）。Ｙ符号化部１３には、波形値の相関を利用した高圧縮率の符号化方法、例えば非特許文献１に示す各種の可逆圧縮符号化、もしくはＭＰＥＧ４，ＡＡＣ，ＴwinＶＱなど非可逆の圧縮符号化法を用いてもよい。Ａ符号化部１５は可逆の圧縮符号化方式を用いる。

【００１９】

乗数Ａは、商信号列Ｙや誤差信号列Ｚと比較して情報量が無視できる程度に小さい場合は破線２１で示すようにＡ符号化部１５を設けることなく乗数Ａをそのまま合成部１６へ入力してもよい。

誤差信号列Ｚは、入力信号ｘ（ｉ）の性質によっては十分な圧縮利得が得られず、演算量増加の影響が大きくなる場合がある。そのような場合は、Ｚ符号化部１４を省略して破線２２で示すように誤差信号列Ｚをそのまま合成部１６へ入力してもよい。

Ｚ符号化部１４は例えばエントロピイ符号化法を用いるが、誤差信号ｚ（ｉ）中の上位ビットのみを用いてエントロピイ符号化し、非可逆圧縮符号化としてもよい。更に入力信号ｘ（ｉ）の性質によっては、誤差信号ｚ（ｉ）を必ず０にできると判っている場合や、ｚ（ｉ）が再生品質に実質的に影響を与えない程度に非常に小さくなることが判っている場合は、破線の掛け印×２３で示すように誤差信号列Ｚを破棄してもよい。この場合はＺ符号化部１４は省略され、符号化データC_Xには符号C_Zは含まれない。

復号化処理

実施形態１の復号化装置機能構成例を図４に、復号化方法（処理手順）の例を図５にそれぞれ示す。

【００２０】

入力端子５１からの符号化データC_Xは符号化装置と同じサンプル数を１フレームとし、フレームごとに処理される。符号化データC_Xが入力されると（ステップＳ１１）、分離部５２は符号化データC_Xを、フレームごとに符号C_Yと符号C_Zと符号C_Aとに分離し、対応するＹ復号化部５３、Ｚ復号化部５４、Ａ復号化部５５に渡す（ステップＳ１２）。Ｙ復号化部５３、Ｚ復号化部５４、Ａ復号化部５５には、それぞれＹ符号化部１３、Ｚ符号化部１４、Ａ符号化部１５の各圧縮符号化方法と対応する伸張復号化方法が用いられる。

【００２１】

各復号化部５３～５５は、それぞれ入力された符号を復号化処理して、符号化部１３～１５の各入力に対応した商信号列Ｙ、誤差信号列Ｚ、共通の乗数Ａを求める（ステップＳ１３）。なおこの例では各符号化部１３～１５が可逆符号化方法を用いているものとしている。

剰余結合処理部２００は、それらの復号化信号を結合処理する。

乗算部２１０は、Ａ復号化部５５から渡された乗数Ａを、Ｙ復号化部５３から渡されたＮ個の商信号ｙ（ｉ）に対し、符号化装置の乗算部１３０で行われたのと同じやり方で乗算してＮ個の信号ｘ’（ｉ）を得る（ステップＳ１４）。このとき、ｘ’（ｉ）は３２ビットの浮動小数点表現で表現可能な値の範囲に丸められる。例えば、ｙ（ｉ）とＡを掛け合わせた結果を倍精度（６４ビット）の浮動小数点数として保持し、得られた倍精度の乗算結果を、最も近い単精度（３２ビット）の浮動小数点数に丸めてｘ’（ｉ）とする。この丸めの処理は、符号化装置の乗算部１３０で行われた処理と同じやり方で行う。このようにして所定の有効桁までのｘ’（ｉ）を求める。

【００２２】

誤差補正部２２０は、乗算部２１０からのＮ個のサンプルｘ’（ｉ）を、それぞれＺ復号化部５４からの誤差信号ｚ（ｉ）により補正処理をして、元の信号ｘ（ｉ）を復元する。つまり符号化装置でｘ（ｉ）からｘ’（ｉ）を減算した場合は、各ｘ’（ｉ）に対応する誤差信号ｚ（ｉ）を加算する。符号化装置でｘ（ｉ）とｘ’（ｉ）とのビット単位の排他的論理和演算（ｘｏｒ）をした場合には、対応する誤差信号ｚ（ｉ）の各ビットを参照してｚ（ｉ）が１となった位置にあたるｘ’（ｉ）のビットを反転してｘ（ｉ）を得る。

フレーム連結部５６は、誤差補正部２２０からの復元されたＮ個のサンプルｘ（ｉ）（ｉ＝０，１，…，Ｎ－１）を連結させて出力する（ステップＳ１６）。

【００２３】

符号化装置が乗数Ａを符号化していない場合は、分離部５２で分離された乗数Ａは図４中に破線６１で示すように乗算部２１０に直接入力される。同様に、符号化装置が、誤差信号ｚ（ｉ）を符号化していない場合は、分離部５２で分離されたｚ（ｉ）は破線６２で示すように誤差補正部２２０に直接入力される。符号化装置でｚ（ｉ）が破棄された場合は、分離部５２で符号C_Zと対応するものが得られず、誤差補正部２２０にはなにも入力されない。言い換えると０が入力されることになる。

なお、符号化装置は、乗数Ａが前フレームで得られた乗数Ａと同一であれば、さらに情報量を少なく出来る。具体的には、図２のＡ符号化部１５に示すように、符号化装置にレジスタ１５ａと判定部１５ｂを設ける。レジスタ１５ａには、前フレームの乗数Ａを格納しておき、判定部１５ｂで現フレームの乗数Ａと前フレームの乗数Ａとを比較する。そして、同一であればそのことを示す１ビットのみを符号C_Aとして出力する。これにより符号化データC_Xの情報量を減少することができる。復号化装置では、図４中のＡ復号化部５５内の前フレームの乗数Ａを格納するレジスタ５５ａと、符号C_Aが前フレームと同一乗数であることを示すビットであるか否かを判定し、同一であれば、符号C_Aの復号化処理を行うことなく、レジスタ５５ａ内の乗数Ａを乗算部２１０に出力する判定部５５ｂを設ければよい。この場合は、符号C_Aは乗数Ａが前フレームと同一か否かを示す１ビットでよく、乗数Ａを符号化しない場合よりも、情報量を減少させることができる。

【００２４】

この発明を適用すると圧縮効率を高くすることができることを、図６を参照して説明する。図６は、入力信号ｘ（ｉ）が２進数表現の場合の１例である。入力信号サンプル列ｘ（１），ｘ（２），ｘ（３）が、図６Ａに示すように１０進数表現でｘ（１）＝２５０、ｘ（２）＝５０、ｘ（３）＝３５０の場合、その２進数表現はそれぞれ、「０」と「１」が比較的ランダムに配列されている。

これらサンプルｘ（１），ｘ（２），ｘ（３）を共通の数Ａ＝１．５６２５で割算すると、その商信号ｙ（１），ｙ（２），ｙ（３）は図６Ｂに示すようにそれぞれ１０進数表現でｙ（１）＝１６０、ｙ（２）＝３２、ｙ（３）＝２２４となる。

【００２５】

これら商信号ｙ（１），ｙ（２），ｙ（３）の２進数表現はそれぞれ図６Ｃに示すように、「０」が連続して存在する部分が多い「１」と「０」の配列である。このｙ（１），ｙ（２），ｙ（３）は高圧縮符号化が可能である。乗算Ａも送出する必要があるが、サンプル数が多く、かつ、その大部分が高圧縮可能な商信号の場合は、全体としての圧縮符号

化量を大きく減少させることができる。

図７にＩＥＥＥ－７５４浮動小数点数のサンプル列に適用した例を示す。１０進数表現の数値４７８．４，９５．６８，６６９．７６の浮動小数点表現は、それぞれ図７Ａに示すようにその仮数部の２３ビットの「１」、「０」の配列はかなりランダムである。しかし、これらを共通乗数２．９９で割算すると、その商信号は図７Ｂに示すように１０進数表現が１６０，３２，２２４になる。これらを浮動小数点表現にすると、仮数部の２３ビットの「１」と「０」の配列は、１つ目が上位の２ビット目のみ「１」、２つ目はすべてゼロと、３つ目は上位の１ビット目と２ビット目のみ「１」となる。したがって、図７Ａのまま圧縮するよりも、著しく高く圧縮することができる。

実施例２（有効桁・丸め選択）

入力信号がどのような有効桁、丸め手段によって生成されたかによって、ｘ（ｉ）の性質は異なる。一般に浮動小数点の演算を行う場合の丸め処理としては、単精度のまま演算を行ったり、倍精度で乗算を行った結果を単精度に丸めたりできる。

【００２６】

丸めの処理としては、所定の有効桁で切り捨てたり、四捨五入したり、ニアレストタイツウイーブン（nearest tie to even）や切り上げがある。そこで商信号ｙ（ｉ）に乗数Ａを乗じてｘ’（ｉ）を生成する際に、入力信号ｘ（ｉ）が生成された際に用いられたのと同じ方法を推定して用いることで、誤差ｚ（ｉ）を最小化する。具体的には、演算精度（有効桁）、丸めの方法として考えられるものを複数ためし、最も誤差ｚ（ｉ）が小さくなる方法を選択する。以下にその具体例を説明する。

図２中の乗算部１３０において、商信号ｙ（ｉ）の有効桁（演算精度）及び丸め方法を選択する例を説明する。その要部の機能構成例を図８に、その処理手順例を図９に示す。剰余分離処理部１００の乗算部１３０は、桁設定部１３０ａと丸め設定部１３０ｂとが設けられる。有効桁・丸め制御部１５０は、有効桁数と丸め方法とを桁設定部１３０ａと丸め設定部１３０ｂに設定する（ステップＬ１）。乗算部１３０では商信号ｙ（ｉ）と乗数Ａとの乗算が設定された有効桁数の倍精度で行われ、その乗算結果に対し、設定された丸め方式により単精度の丸めが行われる（ステップＬ２）。その結果ｘ’（ｉ）と入力信号ｘ（ｉ）との差分が誤差算出部１４０でとられ、その誤差信号ｚ（ｉ）が有効桁・丸め制御部１５０内の記憶部１５０ａに格納される（ステップＬ３）。

【００２７】

次に有効桁・丸め制御部１５０は、予め想定した各有効桁数について、想定した各種の丸め方法の全てについて乗算部１３０における処理が行われたかを調べる（ステップＬ４）。Ｎｏならば、ステップＬ１に戻る。Ｙｅｓならば、有効桁・丸め制御部１５０は、記憶部１５０ａ内の誤差信号ｚ（ｉ）中の最小のものと対応する有効桁数と丸め方法とを、乗算部１３０内に設定する（ステップＬ５）。そして、フレームの全てのサンプルについての誤差信号ｚ（ｉ）を求める処理を行う。またその設定した有効桁数と丸め方法とを示す情報が補助符号C_Bを合成部１６へ入力する（ステップＬ６）。

【００２８】

この場合の復号化装置の剰余結合処理部２００は、図１０に簡単に示すようにＢ復号化部５８と桁設定部２１０ａと丸め設定部２１０ｂを有する乗算部２１０を備える。Ｂ復号化部５８は、分離部５２で分離された符号C_Bを復号化する。その有効桁数と丸め方法とは、乗算部２１０内の桁設定部２１０ａ、丸め設定部２１０ｂにそれぞれ設定される。乗算部２１０は、これら設定情報に基づいて、復号化商信号ｙ（ｉ）と復号化乗数Ａとの乗算を行う。

［実施形態２］

実施例３（変形商信号）

次に商信号ｙ（ｉ）を変形処理することにより、誤差信号ｚ（ｉ）の情報量を減少する他の例を説明する。その要部の機能構成例を図１１に、処理手順例を図１２に示す。

【００２９】

本実施例は、符号化側の誤差算出部１４０から出力される誤差信号ｚ（ｉ）が最小とな

るように、除算処理部１２０で得た商信号ｙ（ｉ）を変形処理する部分が実施例１（図２）と異なる。

例えば、除算処理部１２０で、ｙ（ｉ）を所定の有効桁まで算出する場合に、有効桁より下位の桁で切り捨て、あるいは四捨五入などを行った場合には、ｙ（ｉ）は丸め誤差を含む。同様に復号化装置（図４）の復号化手順で、乗算部２１０で復号化したｙ（ｉ）と復号乗数Ａを掛け合わせる際にも、丸め誤差が生じる場合がある。この２度の丸め処理によって、誤差が累積され、誤差信号ｚ（ｉ）が大きくなる場合がある。

【００３０】

本実施例では、復号化時の乗算処理によって生じる丸め誤差を予め考慮してｙ（ｉ）をｙ’（ｉ）に変形しておくことによって、誤差信号ｚ（ｉ）の情報量が低減するようにする。

各入力サンプルｘ（ｉ）に対して独立に商信号ｙ（ｉ）を変形し、最も誤差信号ｚ（ｉ）が小さくなるような変形誤差信号をｙ’（ｉ）とする。このことにより、乗数推定部１１０において推定された乗数Ａが誤差を含むような場合にも、乗数Ａの推定誤差の影響を低減することが出来る。例えば、１フレームの１０２４サンプルｘ（ｉ）のほとんどが、有効桁の範囲内で乗数Ａで割り切れ、一部のサンプルｘ（ｉ）だけが乗数Ａで割り切れないような場合に、ｙ（ｉ）に対する前記変形操作を行うことで誤差信号ｚ（ｉ）の情報量を低減出来る。

【００３１】

除算処理部１２０は、ｘ（ｉ）をＡで割算し、丸め処理した商信号ｙ（ｉ）を、変形処理部１６０に渡す。

変形処理部１６０は、除算処理部１２０から渡されたｙ（ｉ）を変形して、ｚ（ｉ）が最小となるように変形したｙ’（ｉ）を決定する。ｙ’（ｉ）を決定する手順を図１２により簡略に説明する。（詳細は後で示す。）入力信号サンプルｘ（ｉ）の絶対値が乗算部１３０の乗算結果の絶対値｜Ａｙ’（ｉ）｜より大きいかを調べる（ステップＬ１１）。大きければ、ｙ’（ｉ）の最下位ビットに１を加算する（ステップＬ１２）。たとえば、ｙ’（ｉ）が2進数表現で

＋１．０００００００００００００００００１１１１１１×２＾０

の場合には、最下位ビットに１を加算することでｙ’（ｉ）の値は、

＋１．０００００００００００００００００１０００００×２＾０

となる。ステップＬ１１で｜ｘ（ｉ）｜が｜Ａｙ’（ｉ）｜より大きくなければ、ｙ’（ｉ）の最下位ビットから１を減算する（ステップＬ１３）。ステップＬ１２又はＬ１３の後に、変更されたｙ’（ｉ）と乗数Ａとを乗算部１３０で乗算処理する（ステップＬ１４）。乗算処理結果の絶対値｜Ａｙ’（ｉ）｜が入力サンプルの絶対値｜ｘ（ｉ）｜と等しいかを調べる（ステップＬ１５）。等しくなければ、ステップＬ１１の判定に変化があったかを調べる（ステップＬ１６）。例えば前回の判定が｜ｘ（ｉ）｜＞｜Ａｙ’（ｉ）｜であったが、今回は｜ｘ（ｉ）｜＜｜Ａｙ’（ｉ）｜に変化した場合は、前回の誤差信号ｚ（ｉ）と今回の誤差信号ｚ（ｉ）との小さい方をＺ符号化部１４へ出力し、また対応するｙ’（ｉ）をＹ符号化部１３へ出力する（ステップＬ１７）。従って、誤差算出部１４０からの誤差信号ｚ（ｉ）はステップＬ１７で比較し、出力できるように、前回の分は常に保持しておく。なおｙ’（ｉ）の初期値はｙ（ｉ）を用いる。

【００３２】

ステップＬ１６でステップＬ１１の判定に変化がなければステップＬ１１に戻る。またステップＬ１５で両者が等しければ誤差信号ｚ（ｉ）＝０とし、その入力サンプルｘ（ｉ）に対するｙ（ｉ）の変形処理を終了する。変形処理部１６０にはステップＬ１１，Ｌ１５，Ｌ１６，Ｌ１７の判定を行う判定部１６０ａ、ステップＬ１２，Ｌ１３における前回のｙ’（ｉ）に対する変形とその保持を行う増減部１６０ｂ、ステップＬ１７で必要とする誤差信号の記憶部１６０ｃなどが設けられている。

なお誤差算出部１４０からの出力ｚ（ｉ）の極性（符号）に基づきステップＬ１２又はＬ１３と同様の変形処理を行うようにｚ（ｉ）を変形処理部１６０に帰還して誤差信号ｚ

（ｉ）が最小になるようなｙ’（ｉ）を決定してもよい。要は変形処理部１４０の処理としては、誤差信号ｚ（ｉ）が最小になるようにｙ（ｉ）の有効桁の最小桁を最小単位ずつ増減させてｙ’（ｉ）を決定する。

【００３３】

ここで、誤差信号ｚ（ｉ）が最小となるｙ’（ｉ）の候補が複数存在する場合には、ｙ’（ｉ）の最もＬＳＢ（最下位ビット）側の０でないビットが最もＭＳＢ（最上位）側にあるものを選択する（ｙ’（ｉ）の構成ビットのうち０でないビットの数を少なくすることによって符号化装置の圧縮効率が向上する場合がある。）。

ここで、誤差が最小となるｙ’（ｉ）が複数存在する場合に、構成要素のビットで、１が最も少ないｙ’（ｉ）を選択するようにしてもよい。

以上のようにしてｙ（ｉ）が変形されたｙ’（ｉ）がＹ符号化部１３へ入力される。なお、除算処理部１２０では丸めを行わず、ｙ（ｉ）の所望の有効桁よりも多い有効桁の精度で除算結果を算出しておき、乗算部１３０における乗算結果に対する丸めに基づく誤差を含む全体の誤差信号ｚ（ｉ）が最小となるようにｙ（ｉ）を変形したｙ’（ｉ）を求めてもよい。つまり、符号化装置における乗算部１３０の乗算処理が、復号化装置で行われる乗算部２１０の乗算処理と同じ結果となっていれば、丸め処理を除算処理部１２０で行っても、変形処理部１６０で変形と同時に行ってもどちらでもよい。

実施例４（変形商信号）

この実施例４も商信号ｙ（ｉ）に対し変形処理を行うが、ｙ’（ｉ）を決定する際の評価尺度が実施例３と異なる。変形処理部１７０の処理として、Ｙ符号化部１３、Ｚ符号化部１４で可逆符号化を用いる場合に、符号C_Yと、符号C_Zの合計の符号量が最も小さくなるように、ｙ（ｉ）を変形してｙ’（ｉ）とする。例えば、ｙ’（ｉ）の有効桁を所定の範囲で変化させ、出力符号サイズの総和が最も小さくなるようなｙ’（ｉ）を探索する。

【００３４】

具体的には、その要部の機能構成例を図１３に、処理手順例を図１４にそれぞれ示す。除算処理部１２０からの商信号ｙ（ｉ）は、変形処理部１７０に入力される。まずｙ（ｉ）と対応する誤差信号ｚ（ｉ）とをそれぞれ符号化する（ステップＬ２１）。符号C_Yの符号量V_Yと符号C_Zの符号量V_Zとの加算符号量（ビット数）V_Y+V_Zを求める（ステップＬ２２）。次にｙ（ｉ）の有効桁の最下位ビットに対し、最小単位だけ予め決めた変形、例えば１の加算を行う（ステップＬ２３）。この変形商信号ｙ’（ｉ）と、このｙ’（ｉ）に基づき求めた誤差信号ｚ（ｉ）とをそれぞれＹ符号化部１３、Ｚ符号化部１４で符号化する（ステップＬ２４）。符号C_Y，C_Zの符号量の和（ビット数）を求める（ステップＬ２５）。

【００３５】

この符号量が前回より増加したかを調べる（ステップＬ２６）。増加していれば、ｙ（ｉ）に対する変形を逆にする、つまりステップＬ２３で最下位ビットに１を加算した場合は、最下位ビットから１の減算を行う。この変形ｙ’（ｉ）と対応ｚ（ｉ）を符号化し（ステップＬ２８）、これらの加算符号量を求め（ステップＬ２９）、符号量の変化が減少していたか、増加に変化したかを調べる（ステップＬ３０）。

増加しなければ、ｙ’（ｉ）をそれまでと同一方向に更に変形してステップＬ２８に戻る（ステップＬ３１）。ステップＬ２６で符号量が増加していなければ、ステップＬ３１に移る。ステップＬ３０で符号量の変化が増加に変化したと判定されると、その直前の符号量と今回の符号量との少ない方と対応する符号C_YとC_Zを合成部１６へ出力する（ステップＬ３２）。

【００３６】

変形処理部１７０内には、ステップＬ２２，Ｌ２５，Ｌ２９で符号C_YとC_Zの符号量を加算する符号量加算部１７０ａ、ステップＬ２６，Ｌ３０の判定、ステップＬ３２内の大小判定などを行う判定部１７０ｂ、ステップＬ２４，Ｌ２７，Ｌ３１のｙ（ｉ）に対する変形を行う増減部１７０ｃ、ステップＬ３２の符号量の判定と、前回のC_Y，C_Zを出

力するためなどに利用される記憶部１７０ｄなどが設けられる。

例えばｙ（ｉ）のビット列に１が連続しているような場合に、あえて１を加えると順次桁上りにより０の連続に変換することが出来る。この場合には、加えた１をＡ倍しただけ誤差信号ｚ（ｉ）は増加するが、ｙ’（ｉ）の符号化効率が良くなるので、全体として符号量は減少する。

実施例５（商信号変形）

この実施例では、例えば図１５に要部を示すようにｘ（ｉ）をＡで割った商信号ｙ（ｉ）を変形処理部１８０内の圧縮判定部１８０ａで観測し、例えばＹ符号化部１３においてうまく、つまり効率的に圧縮できるか条件に合っているかを判定し、条件に合っていなかったサンプルについてはｙ’（ｉ）＝０とする。つまり条件に合っていなければ圧縮判定部１８０ａの出力によりスイッチ１８０ｂが０信号源１８０ｃ側に切替えられ、ｙ’（ｉ）＝０に変形される。ｙ（ｉ）’＝０の場合は乗算部１３０の出力が０となり、誤差算出部１４０からはｚ（ｉ）＝ｘ（ｉ）が出力される。ｙ（ｉ）が条件に合っていると圧縮判定部１８０ａで判定されると、スイッチ１８０ｂは除算処理部１２０に切替えられ接続され、ｙ（ｉ）がそのままｙ’（ｉ）として変形処理部１８０から出力される。

【００３７】

このようにして、あるサンプルについてはｙ（ｉ）を圧縮するよりもｘ（ｉ）を圧縮した方が効率よく圧縮することができる場合に、全体の圧縮率を向上させることができる。

なお、実施例４も５も、ｘ（ｉ）からｘ’（ｉ）を差し引いた誤差が最小になるようにｙ（ｉ）を変形しているともいえる。

［実施形態３］

実施例６（非可逆符号化）

実施例６は、商信号ｙ（ｉ）に対し非可逆符号化方法により符号化する例である。その機能構成例を図１６に示す。除算処理部１２０よりの商信号ｙ（ｉ）は非可逆圧縮符号化方法の非可逆Ｙ符号化部３１で符号化される。その符号化された符号C_Yを復号化部３２で、復号化し、その復号化された信号ｙ（ｉ）に対し、乗数Ａを乗算処理し、その結果ｘ’（ｉ）とｘ（ｉ）との誤差信号ｚ（ｉ）を得る。

【００３８】

復号化装置での処理は、図４に示した処理と同様に行えばよい。ただし、Ｙ復号化部５３は、非可逆Ｙ符号化部３１の非可逆圧縮符号化方法と対応する非可逆伸張復号化方法のものを用いる。

このように、非可逆Ｙ符号化部３１、Ｙ復号化部として、既知の非可逆符号化方式を用いた場合にも、符号化部１４，１５、復号化部５４，５５の処理にそれぞれ可逆符号化方法とこれと対応する可逆伸張復号化を用いることによって、全体として可逆符号化を行うことが出来る。Ｚ符号化部１４、Ｚ復号化部５４にも非可逆符号化を用いた場合には、全体としても非可逆符号化となる。

【００３９】

図１６中に破線で示すように、除算処理部１２０と非可逆Ｙ符号化部３１との間に変形処理部３５を挿入し、誤差算出部３４からの誤差信号ｚ（ｉ）を変形処理部３５に入力して、誤差信号ｚ（ｉ）が最小になるようにｙ（ｉ）を変形して非可逆Ｙ符号化部３１へ入力するようにしてもよい。同様に全体の符号化サイズ、つまり符号化データC_Xの符号量が最小化するように変形処理部でｙ（ｉ）を、符号化データC_Xのサイズに応じて変形し、非可逆Ｙ符号化部３１へ供給するようにしてもよい。

［実施形態４］

実施例７

先に述べたように、商信号ｙ（ｉ）と誤差信号ｚ（ｉ）、乗数Ａに分離してそれぞれを符号化したほうが、ｘ（ｉ）をそのまま圧縮符号化する場合と比較して圧縮効率が良い場合には、分離して符号化を行い、ｘ（ｉ）をそのまま符号化したほうが圧縮効率が良い場合には、ｘ（ｉ）をそのまま圧縮符号化する例を実施形態４とし、その１例を図１７に符号化装置を、図１８に復号化装置をそれぞれ示す。

【００４０】
図１７の分離符号化部３００は、剰余分離処理部３１０、可逆符号化方式のＹ符号化部３２０、可逆符号化方式のＺ符号化部３３０、必要に応じて設けられるＡ符号化部３４０、合成部３５０を備え、これらの組は実施例１～５に示したいずれかと対応するものである。直接符号化部３６０は、入力信号ｘ（ｉ）を直接符号化するものである。分離符号化部３００で符号化された符号化データC_{SX}のデータ量（情報量、ビット数）V_Sと、直接符号化部３６０で符号化された符号化データC_{DX}のデータ量V_Dとが圧縮率比較部３７０で比較される。そして選択部３８０は、データ量が少ない方の符号化データを選択する。合成部３９０は、選択された符号化データと、いずれを選択したかを示す選択符号（１ビット）C_Cとを合成して、符号化データC_Xとして出力する。

【００４１】
復号化装置においては図１８に示すように、符号化データC_Xは分離部４００で符号化データと選択符号C_Cとに分離される。符号C_Cが分離符号化データC_{SX}であることを示していれば、切替部４１０が制御され、符号化データC_{SX}が分離復号化部４２０に入力される。符号C_Cが直接符号化データC_{DX}であることを示していれば、切替部４１０が制御され、直接符号化データC_{DX}が直接復号化部４３０に入力される。
分離復号化部４２０内の分離部４４０、可逆復号化方式のＹ復号化部４５０、可逆復号化方式のＺ復号化部４６０、必要に応じて設けられるＡ復号化部４７０、乗算部４８０、誤差補正部４９０の組は、実施例１～５の復号化装置のいずれかと対応するものである。切替部５１０は、選択符号C_Cに応じて、分離復号化部４２０からの復号信号または直接復号化部４３０からの復号信号を出力する。なお、切替部５１０は省略し、両復号化部４２０、４３０の両出力端を接続して、復号化処理がなされた信号が出力されるようにしてもよい。

【００４２】
直接符号化部３６０を独立に設けることなくまた選択符号C_Cを省略してもよい。例えば、図１９に示すようにＹ符号化部３２１に剰余分離処理部３１０からの商信号ｙ（ｉ）だけでなく、乗数Ａも入力する。この場合、Ｙ符号化部３２１の判定部３２１ａは、乗数Ａが１．０か否かを判定する。乗数Ａが１．０でなければ切替部３２１ｂが制御され、商信号ｙ（ｉ）が商符号化部３２２に入力され、商信号ｙ（ｉ）に適した符号化が行われる。判定部３２１ａが乗数Ａ＝１．０と判定すると、切替部３２１ｂが制御されて商信号ｙ（ｉ）、つまり入力信号ｘ（ｉ）が直接符号化部３２３に入力される。この場合の商信号ｙ（ｉ）は除算処理部１２０の処理出力はＡが１．０であるからｙ（ｉ）＝ｘ（ｉ）であり、ｘ（ｉ）の符号化に適する符号化が直接符号化部３２３で行われることになる。選択符号C_Cは出力されない。また、選択部３８０、合成部３９０も省略され、図１７に示した構成より簡単になる。

【００４３】
対応する復号化装置を図２０に示す。分離部４４０で分離された符号C_Yは、Ｙ復号化部４５１に入力されると共に、分離された乗数Ａ（又は符号C_A）もＹ復号化部４５１に入力される。判定部４５１ａで乗数Ａが１．０でないと判定されると、切替部４５１ｂは、符号C_Yを商復号化部４５２に入力する。そして、商復号化部４５２が符号C_Yに適した復号を行う。判定部４５１ａがＡ＝１．０と判定されると、切替部４５１ｂは、符号C_Yを直接復号化部４５３に入力する。そして、直接復号化部４５３が符号C_Yに適する復号を行う。Ｙ復号化部４５１からの復号化出力ｙ（ｉ）は、乗算部４８０に入力され、乗数Ａとの乗算処理が行われる。図１８に示した構成より簡単となっている。

実施例８

実施例８は分離符号化と、直接符号化の他の例であってその概略を図２１にて示す。ここで用いられる分離符号化部３０１は、非可逆Ｙ符号化部３１として非可逆圧縮符号化を用いる場合である。この場合は、復号側から考えると、符号化サイズ（符号化量）よりも符号化波形歪が小さいことが重要である。

【００４４】

分離符号化部３０１からのＹ符号C_Y、誤差符号C_Z及び乗数Ａ（又はその符号C_A）からなる符号化データC_{SX}とは、局部復号化部３０２で復号化され、復号化信号列Ｘ’が得られる。直接符号化部３６０からの符号化データC_{DX}は、局部復号化部３０３で復号化され、復号化信号列Ｘ”が得られる。歪計算部３０４は、局部復号化信号列Ｘ’と原入力信号列Ｘとの波形歪を計算する。歪計算部３０５は、局部復号化信号列Ｘ”と原入力信号列Ｘとの波形歪を計算する。これら計算された歪中の小さいものが判定部３０６で判定される。選択部３８０は、その判定結果に基づき、歪が小さい方の符号化データを選択する。合成部３９０は、選択部３８０からの符号化データと、判定部３０６からのいずれを選択したかを示す選択符号C_Cとを合成し、符号化データC_Xを出力する。

［実施形態５］

実施形態５は剰余分離処理により得られた、商信号列Ｙ及び／又は誤差信号列Ｚを再帰的に剰余分離処理する。

実施例９（符号化１）

図２２にその機能構成例を示す。入力信号列Ｘは、剰余分離処理部１００に入力されて、商信号列Y_1、誤差信号列Z_1、乗数A_1に分離される。その商信号列Y_1は剰余分離処理部１０１Ｙで更に商信号列Y_2、誤差信号列Z_2、乗数A_2に分離される。その商信号列Y_2は剰余分離処理部１０２Ｙで商信号列Y_3、誤差信号列Z_3、乗数A_3に分離される。

【００４５】

この例では３回の再帰的分離を行った。最後の商信号列Y_3は、Ｙ符号化部１０３Ｙで可逆圧縮符号化され、符号C_{Y3}として出力される。誤差信号列Z_1，Z_2，Z_3は、それぞれＺ符号化部１０３Z_1，１０３Z_2，１０３Z_3で可逆圧縮符号化され、符号C_{Z1}，C_{Z2}，C_{Z3}としてそれぞれ出力される。乗数A_1，A_2，A_3は、そのまま、あるいは可逆符号化されてC_{A1}，C_{A2}，C_{A3}として出力される。合成部１６は、符号C_{Y3}，C_{Z1}，C_{Z2}，C_{Z3}，A_1（又はC_{A1}），A_2（又はC_{A2}），A_3（又はC_{A3}）を合成し、符号化データC_Xとして出力する。

【００４６】

誤差信号列Z_1について同様に再帰的に符号化してもよい。その例を図２２中に破線で示す。誤差信号列Z_1は、剰余分離処理部１０１Ｚで商信号列Ｙ（Z_1）、誤差信号列Ｚ（Z_1）、乗数Ａ（Z_1）に分離される。この誤差信号列Ｚ（Z_1）は、剰余分離処理部１０２ＺZ_1で商信号列Ｙ（ＺZ_1）、誤差信号列Ｚ（ＺZ_1）、乗数Ａ（ＺZ_1）に分離される。この場合は、合成部１６は、商信号Y_1の可逆符号化符号C_{Y1}と、商信号列Ｙ（ＺZ_1）の可逆符号化部１０４Ｙにより符号化された符号C_{YZZ1}と、誤差信号列Ｚ（Z_1），Ｚ（ＺZ_1）をそれぞれ符号化部１０４Z_1，１０４Z_2で可逆符号化された符号C_{ZZ1}，C_{ZZZ1}と、乗数Ａ（Z_1）（又はその符号C_{AZ1}），Ａ（ＺZ_1）（又はその符号C_{AZZ1}）とを合成し、符号化データC_Xとして出力する。

【００４７】

更に、剰余分離処理部で分離された商信号と誤差信号との両者を、再帰的に分離符号化してもよい。その場合図２２の例では破線で示すように、誤差信号Z_2は剰余分離処理部１０２Ｚで商信号列Ｙ（Z_2）、誤差信号列Ｚ（Z_2）、乗数Ａ（Z_1）に分離される。Ｙ（Z_2），Ｚ（Z_2）は、それぞれ符号化部１０５Ｙ，１０５Ｚで可逆符号化され、符号C_{YZ2}，C_{ZZ2}が出力される。また、剰余分離処理部１０１Ｚで分離された商信号列Ｙ（Z_1）は、剰余分離処理部１０２ＹZ_1で商信号列Ｙ（ＹZ_1）、誤差信号列Ｚ（ＹZ_1）、乗数Ａ（ＹZ_1）に分離される。商信号列Ｙ（ＹZ_1）、誤差信号列Ｚ（ＹZ_1）は、それぞれ符号化部１０６Ｙ，１０６Ｚで符号化されて、可逆符号C_{YYZ1}，C_{ZYZ1}が出力される。

【００４８】

結局、合成部１６は、符号C_{Y3}，C_{Z3}，C_{YZ2}，C_{ZZ2}，C_{Z2}，C_{YYZ1}，C_{ZYZ1}，C_{YZZ1}，C_{ZZZ1}，C_{ZZ1}の他に乗数A_3，Ａ（Z_1），A_2，Ａ（ＹZ_1），Ａ（ＺZ_1），Ａ（Z_1），A_1又はこれらの符号を合成して、符号化データC_Xとする。

実施例１０（符号化２）

このように再帰的分離符号化は、対象とする商信号又は誤差信号が分割できなくなるまで繰り返す。あるいは分割による符号のサイズ（符号化データC_Xのビット数）が減少する効果が得られなくなるまで行う。

【００４９】

実施例１０では、商信号列Ｙを再帰的に分割する場合を示す。図２３にその処理手順を示す。まず、入力信号列Ｘに対し剰余分離処理をして分離符号化してC_Y，C_Z，Ａ（又はC_A）よりなる符号化データC_{SX}を求める。また、入力信号Ｘ列を直接符号化して、符号化データC_{DX}を求める（ステップｓ１）。符号化データC_{SX}とC_{DX}のサイズを比較して、分離符号化を選択するかを判定する（ステップｓ２）。C_{SX}の方が、データ量が少なく、分離符号化が選択されると、その分離された商信号列Ｙに対し剰余分離処理をして分離符号化し商符号C_{Yi}、誤差符号C_{Zi}、乗数A_i（又はその符号C_{Ai}）からなる符号化データC_{Xi}を求める（ステップｓ３）。

【００５０】

この状態での出力符号化データC_Xのデータ量として、現在の処理で求めたC_{Yi}と今までの処理で求めた全ての誤差符号C_{Zi}と全ての乗数A_iとの合計のビット数V_{Si}（符号化データのデータ量）を計算する（ステップｓ４）。データ量V_{Si}と直接符号化の符号化データC_{DX}のデータ量V_Dとを比較し、データ量V_{Si}の方が小さい、かつ前回の分離符号化データ量V_{Si-1}より小さいことを調べる（ステップｓ５）。この条件を満たせば、現在の処理で求めたC_{Zi}とＡを記憶部に記憶してステップｓ３に戻る（ステップｓ６）。

ステップｓ５でV_Dより小さいがV_{Si-1}より大であるか、V_Dより小でなければ、直前処理で得られたC_{Yi}と、直前処理までに得られた全てのC_{Zi}とA_iを符号化データC_Xとして出力する（ステップｓ７）。ステップｓ２でデータ量V_Dの方が小さければ、直接符号化データC_{DX}を符号化データC_Xとして出力する（ステップｓ８）。

実施例１１（復号化１）

図２２中に示した商信号列Ｙを再帰符号化した符号化データC_Xを復号する例を図２４に示す。分離部５２からの符号C_{Y3}は、商復号化部７１で復号化され、Y_3が出力される。乗算部210_1は、このY_3に分離部５２からの乗数A_3を乗算処理する。誤差復号化部72_1は、誤差符号C_{Z3}を復号化し、誤差信号列Z_3を得る。補正部220_1は、乗算部210_1の出力を復号化された誤差信号列Z_3で補正し、商信号列Y_2が得られる。

【００５１】

乗算部210_2は、Y_2と分離部５２からの乗数A_2との乗算処理を行う。誤差復号化部72_2は、分離部５２からの誤差符号C_{Z2}を復号化し、誤差信号列Z_2を得る。補正部220_2は、乗算部210_2からの出力を、誤差信号Z_2で補正し、商信号列Y_1を得る。このY_1に対し、分離部５２よりの乗数A_1が乗算部210_3で乗算処理される。誤差復号化部72_3は、分離部５２よりの誤差符号C_{Z1}を復号化し、誤差信号列Z_1を得る。補正部220_3は、乗算部210_3の出力を誤差信号列Z_1で補正し、原信号列Ｘを出力する。

実施例１２（復号化２）

この再帰符号に対する一般的な復号化処理手順の例を、図２５を参照して説明する。まず入力符号化データC_Xを各符号に分離して記憶部に記憶する（ステップｓ１１）。次にその記憶部中の未復号中の符号化における最も後段で得られた各符号を選択する（ステップｓ１２）。これら各符号を復号化する（ステップｓ１３）。最初は商信号と誤差信号と乗数が得られ、次回からは、例えば図２４に示した例では誤差信号と乗数が得られる。

【００５２】

これら復号化された信号を用いて剰余結合処理を行う（ステップｓ１４）。つまり復号化された商信号（又は前回の処理で得られた商信号）は乗算処理される。また、その結果は、復号化された誤差信号により補正される。

次に記憶部中に未選択の符号があるかを調べ、あれば、ステップｓ１２に戻り、なければ処理を終了する（ステップｓ１５）。このようにして元入力信号列Ｘが復号化される。

［実施形態６］

実施例１３

先に示した各実施例では、フレーム単位で剰余分離処理部１００へ入力した。本実施例では、剰余分離処理部１００に入力する前に、図２６Ａに示すように入力判定部４１でｘ（ｉ）を検査し、剰余分離処理部１００へ入力するか否かの判定をする。例えば、入力サンプルが単精度の浮動小数点数で表される場合には、無限大の値や、数値として表すことが出来ないＮａＮという値であることがある。入力判定部４１では、その判断部４１ａがｘ（ｉ）＝ＮａＮであると判断すると、切替部４１ｂを制御して、０源４１ｃから剰余処理部１００にｘ（ｉ）の代わりに０を出力し、ｘ（ｉ）をＸ符号化部４２に出力する。

【００５３】

入力判定部４１は、サンプルｘ（ｉ）が０の場合には、剰余分離処理部１００に０を出力し、同時にＸ符号化部４２にも０を出力する。

Ｘ符号化部４２は、ｘ（ｉ）を符号化して得られた符号を合成部１６に出力する。

復号化装置は、図２６Ｂに示すようにＹ復号化部５７でｙ（ｉ）を復元する。判定部５７ａは、ｙ（ｉ）＝０と判定すると、Ｘ復号化部５８に通知する。Ｘ復号化部５８は、そのｙ（ｉ）に対応し、分離部５２から分離された符号を復号してｘ（ｉ）を出力する。このｘ（ｉ）は誤差補正部２２０に入力される。誤差補正部２２０は、入力されたｙ（ｉ）＝０と対応するｘ’（ｉ）＝０に対し、前記復号したｘ（ｉ）を加算する。

実施例１４

その実施例３（図１１）では、最も誤差ｚ（ｉ）が小さくなるようにｙ（ｉ）の変形ｙ’（ｉ）を求める例を示した。

【００５４】

入力されたサンプルｘ（ｉ）が、そもそも割り算が出来ないような値であったり、乗数Ａで割ると誤差が大きくなりすぎるような場合がある。例えば、入力サンプルが単精度の浮動小数点数で表される場合には、無限大の値や、数値として表すことが出来ないＮａＮという値であることがある。

入力信号ｘ（ｉ）がＮａＮであった場合には、ｘ（ｉ）を乗数Ａで除算した結果得られたｙ（ｉ）もＮａＮとなる。本実施例では、符号化装置は、図１１中の変形処理部１６０に代り図２７に変形処理部１６１を備える。変形処理部１６１は、判定部１６１ａが、除算処理部１２０から渡されたｙ（ｉ）が、ＮａＮであったと判定すると、切替部１６１ｂを制御して、０源１６１ｃからのｙ’（ｉ）＝０を、Ｙ符号化部１３および乗算部１３０に出力する。

【００５５】

乗算部１３０では、ｙ’（ｉ）に乗数Ａを乗じてｘ’（ｉ）を算出する。よって、ｙ’（ｉ）＝０であることから乗算部１３０の出力は、ｘ’（ｉ）＝０となり、誤差算出部１４０の出力ｚ（ｉ）はｘ（ｉ）＝ＮａＮとなる。この場合、誤差算出部１４０では、誤差を算出する。実施例３では、誤差算出に当たってｘ（ｉ）とｘ’（ｉ）の減算処理を行うこととしていたが、ｘ（ｉ）＝ＮａＮの場合には減算は未定義であるため、正しく減算を行うことが出来ない。

そこで、本実施例では、ｙ’（ｉ）＝０の場合には、誤差信号ｚ（ｉ）としてｘ（ｉ）をそのまま設定する。なお、図２７の変形処理部１６１中の処理部１６１ｄは図１１中の変形処理部１６０と同様な機能をもつ。

【００５６】

復号化処理装置の誤差補正部２２０では、上記と逆の処理を行う必要がある。

すなわち、復元したｙ’（ｉ）＝０であった場合には、ｙ’（ｉ）に乗数Ａを乗じて得られたｘ’（ｉ）も０となる。誤差補正部２２０は、ｘ’（ｉ）が０であるから、Ｚ復号化部５４で復号化されたｚ（ｉ）＝ｘ（ｉ）をそのまま出力する。

この場合も、図２６Ｂ中に示すように判定部５７ａが、ｙ（ｉ）＝０と判定すると、スイッチ２６１を制御してＺ復号化部５４からの復号化出力ｚ（ｉ）＝ｘ（ｉ）がｘ（ｉ）として出力するように設定してもよい。この場合、乗算部１３０及び誤差算出部１４０に

おける処理を行う必要がなく、処理が簡単になる。

【００５７】

なお、図２７では、判定部１６１ａがｙ（ｉ）はＮａＮであると判定すると、ｙ’（ｉ）＝０とする。また、スイッチ２７１はオンとなり、Ｘ符号化部２７２が入力信号ｘ（ｉ）を符号化する。その符号化出力C_Zとスイッチ２７３を通じて合成部１６に入力することとしてもよい。この場合、復号化装置は、図２６Ｂ中に示すように、分離部５２で分離された符号C_ZをＸ復号化部５８にも入力する。そして、判定部５７ａが復号したｙ（ｉ）が０であると判定すると、スイッチ２６２を制御してＸ復号化部５８の復号結果を復号したｘ（ｉ）として出力してもよい。

【００５８】

Ｚ符号化部１４としては、例えばＲｉｃｅ符号化法、代数符号化法などが適している。Ｘ符号化部２７２としては、ハフマン符号化法、ＬＺ符号化法などが適している。Ｚ復号化部５４、Ｘ復号化部２６Ｂ１は、対応する符号化部の符号化法に対するものである。

浮動小数点表現では、正規化して１．……となり、この小数点以下の２３ビットが仮数部とされている。また指数の最小値は－１２７であり、指数が－１２７の場合は０か非正規化数を表す。指数が－１２６で、仮数部の最下位ビットのみが１の場合は浮動小数点で正規化表現できる絶対値の最小値である。絶対値がこれ以下の数値は、浮動小数点表現のための正規化を行うことができない。入力信号ｘ（ｉ）が浮動小数点表現されたサンプル列であった場合には、ＮａＮの他に、ｘ（ｉ）が非正規化数である場合がある。つまり指数が－１２７であり、かつ仮数部の最上位ビットの１つ上位ビットが１でない場合（ＩＥＥＥ－７５４形式の２進浮動小数点表現で正規化された形式により表わせる最小値より小さい場合）がある。ｘ（ｉ）が非正規化数であるような場合に、さらにｘ（ｉ）を乗数Ａで割るとアンダーフローにより情報が失われる。したがって、ｙ（ｉ）として符号化する際に効率よく圧縮符号化が出来ない場合がある。

【００５９】

これを避けるために、変形処理部１６１中の判定部１６１ａとして、ｘ（ｉ）が非正規化数も判定できるようにする方法がある。この方法では、ｘ（ｉ）が非正規化数であったと判定されると、変形処理部１６１はｙ（ｉ）をｙ’（ｉ）＝０に変形してＹ符号化部１３および乗算部１３０に出力する。

乗算部１３０は、ｙ’（ｉ）に乗数Ａを乗じてｘ’（ｉ）を算出する。したがって、ｘ’（ｉ）＝０となり、復号化したｚ（ｉ）＝ｘ（ｉ）がＺ符号化部１４に入力される。先の場合と同様に、非正規化数であったと判定してｙ’（ｉ）＝０にした場合には、スイッチ２７１，２７３を制御して復号化されたｚ（ｉ）＝ｘ（ｉ）をそのまま復号化ｘ（ｉ）として設定してもよい。この場合の復号化装置の処理は、図２６Ｂを参照して述べたと同様である。

【００６０】

また、入力信号ｘ（ｉ）が整数で、乗数がＡ＜１である場合に、ｘ（ｉ）を乗数Ａで割って得られた商信号ｙ（ｉ）が、表現可能な範囲を超えてしまい、オーバーフロー状態となってしまう場合がある。従って、例えば図２７中の変形処理部１６１の判定部１６１ａは、ｙ（ｉ）のオーバーフローが検出された場合にも、ｙ’（ｉ）＝０とする。この場合、復号化装置で復号した商信号ｙ’（ｉ）が０であれば、前述と同様に、符号C_Zの復号化信号ｚ（ｉ）をそのまま復号化信号ｘ（ｉ）とする。

［実施形態７］

この発明と非特許文献２に示した技術とを組み合わせた実施形態７を説明する。

実施例１５（符号化処理）

図２８に実施例１５の符号化装置の機能構成例を示す。この実施例１５では、除算処理部１２０からの出力ｙ（ｉ）が整数化部２８１に入力される。整数化部２８１は、ｙ（ｉ）を整数形式の信号ｙ１（ｉ）に変換する。なお、ｙ（ｉ）をそのまま整数形式に変換するのではなく、変形処理部で変形処理してから変換してもよい。整数形式信号はｙ１（ｉ）は、出力する最大振幅の桁数が２４ビット、あるいは１６ビットと定められている場合

もある。その整数形式信号ｙ１（ｉ）は、その最下位桁に対し丸め処理がなされる。

【００６１】

この丸め処理された整数形式信号ｙ１（ｉ）が、前述した各種実施例と同様にY符号化部１３に出力される。

この実施例では、浮動小数点化部２８２はｙ１（ｉ）を浮動小数点表現に変換し、乗算部１３０へ出力する。そして、この浮動小数点表現の整数に対し、乗数Ａが乗算される。誤差算出部１４０は、その乗算結果によって誤差信号生成処理を前述と同様に行う。

本実施例では、剰数判定部２８３は、乗数Ａが１．０であるかの判定を行う。Ａ＝１．０であればスイッチ２８４が切替えられて、入力信号ｘ（ｉ）が誤差算出部１４０ではなく、差分生成部２８５に入力される。整数化部２８１内に設けられた桁数計算部２８１ｂは、整数形式信号の桁数ｎを計数する。この桁数ｎも差分生成部２８５に入力される。差分生成部２８５は、浮動小数点入力信号ｘ（ｉ）の極性Ｓと指数部Ｅ、仮数部Ｍの下位（２３－ｎ）ビットはそのままとして、仮数部中の上位ｎビット分をすべて０とした出力信号ｚ’（ｉ）を生成する。なお仮数部の整数部分の桁数ｎ＝Ｅ－$Ｅ_0$は整数形式信号ｙ１（ｉ）の最上位の“１”より下位の桁数と等しい。

【００６２】

この差分信号ｚ’（ｉ）は、Ｚ’符号化部２８６で可逆符号化される。

従って、合成部１６には、符号$Ｃ_Y$と、符号$Ｃ_A$（又はＡ）と、符号$Ｃ_Z$（Ａ＝１．０であればＺ’符号化部２８６の出力、Ａ＝１．０でなければＺ符号化部１４の出力）が入力されることになる。

実施例１６（復号化処理）

図２８に示した符号化装置と対応する復号化装置を図２９に示す。分離部５２で分離された符号$Ｃ_Y$は、Ｙ復号化部２９１で復号化され、整数形式信号ｙ１（ｉ）が得られる。整数形式信号ｙ１（ｉ）は、浮動小数点化部２９２で浮動小数点信号とされる。Ａ復号化部５５からの乗数Ａは、剰数判定部２９３でＡ＝１．０か否かが判定される。Ａが１．０でなければ、スイッチ２９４がＺ復号化部５４側に切替えられ、このＺ復号化部５４で復号された誤差信号ｚ（ｉ）が補正部２２０に供給される。

【００６３】

Ａが１．０の場合は、スイッチ２９４はＺ’復号化部２９５側に切替えられ、符号$Ｃ_Z$が復号化される。Ｚ’復号化部２９５には、Ｙ復号化部２９１内の桁数計算部２９１ａで計算された整数形式信号の桁数ｎも入力される。Ｚ’復号化部２９５は、浮動小数点表現における２３ビットの仮数部の上位ｎビットに“０”を埋め、下位２３－ｎビットに符号$Ｃ_Z$の復号信号を設定する。組立部２９５ａは、前記仮数部と、復号された符号ビットＳと、復号された指数部Ｅ－$Ｅ_0$から、浮動小数点形式の差分信号（誤差信号）ｚ（ｉ）を組立てる。この差分信号ｚ（ｉ）が補正処理部２２０に入力される。

【００６４】

なお、スイッチ２９４、Ｚ復号化部５４、Ｚ’復号化部を一体として、Ｚ復号化部５４’としても良い。この場合は、乗数判定部２９３は図２９のように独立した構成部としても良いし、Ｙ復号化部２９１の一部とし、整数形式信号の桁数ｎと一緒に乗数Ａの値をＺ復号化部５４’に供給しても良い。

図２８の変形処理部２８１ａによりｙ（ｉ）＝０に変形処理される場合は、図２８に破線で示すように、ｙ変形判定部２８７でｙ１（ｉ）＝０に変形したことが判定されると、スイッチ２８８を制御して、入力信号ｘ（ｉ）をＸ符号化部２７２で直接符号化し、その符号化結果を$Ｃ_Z$として合成部１６へ供給してもよい。

【００６５】

図２９中に破線で示すように、変形判定部２９６が、復号化されたｙ１（ｉ）が０かを判定する。ｙ１（ｉ）＝０と判定された場合は、スイッチ２９７が、補正処理部２２０側からＸ’復号化部２６Ｂ１側に切替えられる。そして、Ｘ’復号化部２６Ｂ１で復号化された符号ｚ（ｉ）が復号化信号ｘ（ｉ）として出力される。

実施例１７（符号化処理）

この実施例は、剰余分離処理部６００の出力に対して非特許文献２に示した方法を適用した例である。図３０にこの実施例の符号化装置の機能構成例を示す。剰余分離処理部６００内の（入力判定）剰数推定部３０１で乗数Ａを推定する。除算処理部１２０は、入力信号を除算処理する。変形処理部３０２は、除算結果ｙ（ｉ）を変形処理し、処理結果ｙ’（ｉ）を出力する。乗算部１３０は、ｙ’（ｉ）と乗数Ａとを乗算する。更に、誤差算出部１４０は、その乗算結果と入力信号との誤差算出処理を行い、誤差信号ｚ（ｉ）を出力する。

【００６６】

整数誤差分離部３０３内の整数化部３０３ａは、変形処理されたｙ’（ｉ）を、必要に応じて最大振幅が予め決められた整数形式信号ｙ１（ｉ）に変換する。また、ｙ’（ｉ）中の整数値未満（小数点以下）の部分が浮動小数点誤差信号ｙ２（ｉ）として出力される。Ｙ符号化部１３は、整数形式信号ｙ１（ｉ）を符号C_Yに符号化する。誤差符号化部３０４は、浮動小数点誤差信号ｙ２（ｉ）を可逆符号化し、符号C_{2Z}として出力する。剰余分離処理部６００からの誤差信号列Ｚは、Ｚ符号化部１４へ入力される。また、乗数Ａは必要に応じて符号化されて合成部１６に入力される。

実施例１８（復号化処理）

実施例１７の符号化装置に対応する復号化装置の機能構成例を、図３１に示す。Ｙ復号化部５３は、整数形式信号である復号化信号ｙ１（ｉ）を、浮動小数点化部３１２に出力する。誤差復号化部３１１は、分離部５２からの符号C_{2Z}を、浮動小数点形式の復号化信号ｙ２（ｉ）に復号化し、浮動小数点化部３１２に出力する。浮動小数点化部３１２は、ｙ１（ｉ）とｙ２（ｉ）とを組み合わせ、最大振幅が前記予め決められた浮動小数点信号ｙ’（ｉ）を生成する。剰余結合処理部２００は、ｙ’（ｉ）に乗数Ａを乗算し、Ｚ復号化部５４で復号化された誤差信号を加算する。その他の処理は先の実施例と同じである。

【００６７】

図３０の誤算信号ｙ２（ｉ）と誤差信号ｚ（ｉ）とは、いずれも小さい振幅の信号であり、比較的似た性質があるため、誤差符号化部３０４とＺ符号化部１４は破線で囲んで示すように１つのエントロピイ符号器で符号化してもよい。例えば、フレーム中のｙ２（ｉ）を全て符号化した後、ｚ（ｉ）を符号化する方法がある。このように１つの符号化器を用いる場合は、復号化装置でも、図３１中に破線で囲んで示しているように、１つの対応する復号化器で復号してもよい。

図２８の変形処理部２８１ａや図３０の変形処理部３０２としては、実施例２～７、９～１４に説明した各種変形方法のいずれか１つ又はその中のいくつかを同時に用いるものであってもよい。更に、（入力判定）乗数推定部３０１では、実施例１３（図２６Ａ）の入力判定部４１と同様に、入力信号ｘ（ｉ）が無限大の値やＮａＮの場合、これを判定して、ｘ（ｉ）＝０とし、乗数推定を行う。つまり、剰余分離処理部６００からは、ｙ’（ｉ）＝０を出力する。また、スイッチ３０５をオンにしてその時のｘ（ｉ）をＸ符号化部４２で直接符号化し、合成部１６へ出力する。復号化装置では、図２６Ｂと同様に、判定部５７ａが復号化したｙ（ｉ）が０であると判定すると、スイッチ３１３をオンにする。Ｘ復号化部５８は、分離部５２からの分離符号を復号化して、ｘ（ｉ）を出力する。また、剰余分離処理部６００内の変形処理部３０２でも実施例２～７、９～１４で説明した処理と同様のことがなされ、これと対応する処理が復号化装置においても、図２９に示したと同様に行われる。

【００６８】

実施例１５～１８では、可逆符号化について説明した。実施例６（図１６）、８（図２１）でそれぞれ説明した非可逆符号化についても、実施例１５～１８に適用することもできる。また、実施例９～１２（図２２～図２５）の再帰処理も、実施例１５～１８に適用することもできる。符号Ｚ’、符号Ｚはそれぞれ符号化することなく上記差分信号、誤差信号をそのまま出力してもよい。つまり符号化しても圧縮効果がほとんど得られない場合もある。

［実施形態８］
乗数Ａを有理近似で決定する手法の実施形態を以下に説明する。
実施例１９
図３２に、複数の数値の集合の共通乗数を求める処理手順を示す。まずフレームごとに乗数Ａを新たに推定するか否かを決定する。現フレームの全入力信号ｘ（ｉ）を入力する（ステップ３２－１）。前フレームの乗数Ａが１．０でないかを調べる（ステップ３２－２）。１．０でなければその現フレームの乗数Ａの候補とする（ステップ３２－３）。全ての入力信号ｘ（ｉ）についてその乗数Ａが妥当であるかを調べ、妥当でなければその乗数をＡ＝１．０に変更する（ステップ３２－４）。乗数Ａが１．０でないかを調べる（ステップ３２－５）。１．０でなければそのフレームの乗数候補を現フレームの乗数Ａとする（ステップ３２－６）。
【００６９】
ステップ３２－２で前フレームの乗数Ａが１．０、またはステップ３２－５で乗数Ａが１．０であれば、各入力信号を用いて有理近似で乗数Ａを推定する（ステップ３２－７）。その乗数Ａが１．０でないかを調べ、１．０でなければステップ３２－４に移る（ステップ３２－８）。
次に図３２中のステップ３２－７で推定した乗数Ａについて調べ、かつ除算処理結果ｙ（ｉ）を変形する手順の例を、図３３を参照して説明する。全てのｘ（ｉ）のチェックを完了したかを調べる（ステップ３３－１）。調べてないものがあればその入力信号ｘ（ｉ）がＮａＮであるか、無限大であるか、０であるか、非正規化数であるかを調べる（ステップ３３－２）。Ｙｅｓであればｙ（ｉ）を０にし、ステップ３３－１に戻る（ステップ３３－３）。
【００７０】
ステップ３３－２で前記条件外であれば（例外の値でなければ）、その信号ｘ（ｉ）を推定した乗数Ａで割算してｙ１とする（ステップ３３－４）。ｙ１の仮数部の最下位ビット（ＬＳＢ）を０でマスクしてｙ２とする（ステップ３３－５）。ｙ２に乗数Ａを掛算し、その結果をｘ１とする（ステップ３３－６）。信号ｘ（ｉ）からｘ１を引算し、その結果をｚとする（ステップ３３－７）。
この引算の余りｚが０でないかを調べる（ステップ３３－８）。０でないならば、ｘ１の絶対値が信号ｘ（ｉ）の絶対値より小さいかを調べる（ステップ３３－９）。ｘ１の絶対値の方が小さければｙ１の仮数部最下位ビット（ＬＳＢ）に１を足してその値をｙ１とおく（ステップ３３－１０）。そのｙ１の仮数部最下位ビットを０でマスクしてｙ２とする（ステップ３３－１１）。そのｙ２に乗数Ａを掛算してその結果をｘ１とし、ステップ３３－９に戻る（ステップ３３－１２）。
【００７１】
ステップ３３－９で、ｘ１の絶対値の方が小さくなければ、ｘ１の絶対値がｘ（ｉ）の絶対値よりも大きいかを調べる（ステップ３３－１３）。ｘ１の絶対値の方が大きければ、ｙ１の仮数部最下位ビットから１を引算し、その結果をｙ１とする（ステップ３３－１４）。そのｙ１の最下位ビットを０でマスクした値をｙ２とする（ステップ３３－１５）。そのｙ２に乗数Ａを掛算して、その結果をｘ１とする（ステップ３３－１６）。次に信号ｘ（ｉ）からｘ１を引算し、結果をｚとする（ステップ３３－１７）。その引算した結果ｚが０かを確認する（ステップ３３－１８）。ｚ≠０ならば、その時の乗数Ａを１．０として終了する（ステップ３３－１９）。ステップ３３－１３で、ｘ１の絶対値が大きくなければステップ３３－１７に移る。
【００７２】
ステップ３３－８及びステップ３３－１８のいずれかで、ｚ＝０ならば、ステップ３３－２０に移る。ステップ３３－２０では、信号ｘ（ｉ）の仮数部の最下位ビットの８ビットがすべて０であり、かつｙ１の最下位ビットが０でないかを調べる（ステップ３３－２０）。この両条件を満たせば、ｙ１の最下位ビットが０でないサンプル（信号ｘ（ｉ））の数がしきい値（例えば１０）を超えたかを調べる（ステップ３３－２１）。しきい値を

超えていれば、ステップ３３－１９に移り、超えていなければステップ３３－２２に移る。ステップ３３－２２では信号ｘ（ｉ）の仮数部の最下位ビットがすべて０であり、かつｙ１の最下位ビットが０でないかを調べる（ステップ３３－２２）。

【００７３】

ステップ３３－２２の両条件を満たせば、ｙ１の最下位ビットが０でないサンプル（信号ｘ（ｉ））の数がしきい値（例えば１００）を超えたかを調べる（ステップ３３－２３）。しきい値を超えていれば、ステップ３３－１９に移る。

ステップ３３－２２でいずれかの条件を満たしていなければ、ｙ（ｉ）をｙ１としてステップ３３－１に移る（ステップ３３－２４）。ステップ３３－８で、ｚが０であればステップ３３－２０に移る。ステップ３３－１ですべての信号ｘ（ｉ）についての乗数のチェックを完了すればその当該乗数Ａを正しいと決定する（ステップ３３－２５）。この際ｙ（ｉ）も求まっていることになる。ステップ３３－９ないしステップ３３－１６は、ｙに対する変形処理である。ステップ３３－９～３３－１２、ステップ３３－１３～３３－１６は、各々ｙ（ｉ）のダイナミックレンジがなるべく小さくなるような処理である。

【００７４】

ｙの変形の基本処理を図３４に示す。図３３と異なるところのみを説明する。ステップ３３－８で、ｚが０であれば、直ちにステップ３３－２４に移る。同様にステップ３３－１８で、ｚが０であれば、直ちにステップ３３－２４に移る。ステップ３３－１８で、ｚが０でなければ、ｚが所定ビット数（例えば２３ビット）以下で表現できるかを調べる（ステップ３４－１）。表現できれば、ステップ３３－２４に直ちに移り、表現できなければステップ３４－２に移る。ステップ３４－２では、ｙ（ｉ）を０としステップ３３－１に移る（ステップ３４－２）。ステップ３３－１ですべての信号ｘ（ｉ）について処理を完了すれば、このｙ（ｉ）を変形する処理は終了する。

【００７５】

次に浮動小数点を整数に変換する処理手順の例を、図３５を参照して説明する。実施例１５（図２８）の整数化部２８１や、実施例１７（図３０）の整数化部３０３ａにこの処理は適用できる。

乗数Ａ、信号ｘ（ｉ）、除算処理結果ｙ（ｉ）に対し、図３４で変形処理したもの（これもｙ（ｉ）と記載する）を入力する（ステップ３５－１）。ｙ（ｉ）の絶対値の最大値からシフト係数Ｓｃを求める（ステップ３５－２）。符号化装置が想定する量子化ビット数をＢＭＡとし、スケール係数S_Fを次式により求める（ステップ３５－３）。

【００７６】

$$S_F = 2^{(BMA-1+Sc)}$$

すべてのｘ（ｉ）について処理が完了したかを調べる（ステップ３５－４）。完了していなければ、ｙ（ｉ）が０であるかを調べる（ステップ３５－５）。０であれば、ｙ’（ｉ）＝０としてステップ３５－４に戻る（ステップ３５－６）。ステップ３５－５で、ｙ（ｉ）が０でなければ、ｙ（ｉ）が正であるかを調べる（ステップ３５－７）。ｙ（ｉ）が正であれば、ｙ（ｉ）が正の場合の整数化処理を行ってｙ’（ｉ）を求め、ステップ３５－４に戻る（ステップ３５－８）。ステップ３５－７でｙ（ｉ）が正でなければ、ｙ（ｉ）が負の場合の整数化処理を行ってｙ’（ｉ）を求め、ステップ３５－４に戻る（ステップ３５－９）。ステップ３５－４ですべてのｘ（ｉ）について処理が完了すれば、ｙ’（ｉ）を求める処理は終了する。

【００７７】

図３５のステップ３５－８のｙ（ｉ）が正の場合の整数化処理の手順例を、図３６を参照して説明する。ｙ（ｉ）にスケールS_Fをかけた値に対して小数点以下を切り捨ててｙ’（ｉ）を求める（ステップ３６－１）。なお

【数２】

$$\lceil A \rceil$$

はＡの小数点以下の切捨てを表わす。ｙ’（ｉ）をスケールＳ$_F$で割算し、その結果をｙ１とする（ステップ３６－２）。ｙ１に乗数Ａを掛算してその結果を１とする（ステップ３６－３）。信号ｘ（ｉ）からｘ１を引算し、ｚを求める（ステップ３６－４）。ｚが０でないかを調べる（ステップ３６－５）。Ｚが０でなければ、ｘ１がｘ（ｉ）より小さいかを調べる（ステップ３６－６）。ｘ１の方が小さければ、ｙ’（ｉ）に１を加算してｙ’（ｉ）とする（ステップ３６－７）。そのｙ’（ｉ）をスケールＳ$_F$で割算してｙ１とする（ステップ３６－８）。そのｙ１に乗数Ａを掛算しｘ１としてステップ３６－６に戻る（ステップ３６－９）。

【００７８】

ステップ３６－６でｘ１の方が小さくなければ、ｘ１がｘ（ｉ）より大きいかを調べる（ステップ３６－１０）。ｘ１の方が大きければ、ｙ’（ｉ）から１を減算しその結果をｙ’（ｉ）とする（ステップ３６－１１）。そのｙ’（ｉ）をスケールＳ$_F$で割算し、その結果をｙ１とする（ステップ３６－１２）。そのｙ１に乗数Ａを掛算し、その結果をｘ１とする（ステップ３６－１３）。そのｘ１を信号ｘ（ｉ）から引算し、その結果をｚとする（ステップ３６－１４）。そのｚが０でないかを調べる（ステップ３６－１５）。ｚが０でなければｚが所定ビット数（例えば２３ビット）以下で表現可能かを調べる（ステップ３６－１６）。表現可能でなければ、ｙ’（ｉ）を０としてステップ３６－１８に移る（ステップ３６－１７）。ステップ３６－５でｚが０であれば、その時のｙ’（ｉ）が求めるｙ’（ｉ）である。ステップ３６－１０でｘ１の方が大きくなければ、ステップ３６－１８に移る。ステップ３６－１５でｚが０であれば、ステップ３６－１８に移る。ステップ３６－１６でｚを所定ビットＮ以下で表現できれば、ステップ３６－１８に移る。

【００７９】

なおステップ３６－１６で、ｚがＮビット（この例では２３ビット）以下で表現できない場合はｙ’（ｉ）を０とするかわりに乗数Ａを１．０としてもよい。この場合はｙ’（ｉ）はｘ（ｉ）から実施例１６又は実施例１７で行っている手法によって再度求め直す。

次にｙ（ｉ）が負の場合のｙ（ｉ）の変形手順の例を、図３７を参照して説明する。ｙ（ｉ）にスケールＳ$_F$をかけた値の小数点以上を切り上げて整数ｙ’（ｉ）とする（ステップ３７－１）。ここで

【数３】

$$\lfloor A \rfloor$$

はＡの少数点以下を切り上げることを表わす。ｙ’（ｉ）をスケールＳ$_F$で割算し、ｙ１とする（ステップ３７－２）。そのｙ１に乗数Ａを乗算してｘ１とする（ステップ３７－３）。信号ｘ（ｉ）からｘ１を引算し、結果をｚとする（ステップ３７－４）。ｚが０でないかを調べる（ステップ３７－５）。ｚが０でなければｘ１がｘ（ｉ）よりも大きいかを調べる（ステップ３７－６）。ｘ１の方が大きければ、ｙ’（ｉ）から１を減算し、その結果をｙ’（ｉ）とする（ステップ３７－７）。そのｙ’（ｉ）をスケールＳ$_F$で割算し、その結果をｙ１とする（ステップ３７－８）。そのｙ１に乗数Ａを掛算し、その結果をｘ１として、ステップ３７－６に戻る（ステップ３７－９）。

【００８０】

ステップ３７－６でｘ１の方が大きくなければ、ｘ１がｘ（ｉ）より小さいかを調べる（ステップ３７－１０）。ｘ１の方が小さければ、ｙ’（ｉ）に１を加算して、ｙ’（ｉ）とする（ステップ３７－１１）。そのｙ’（ｉ）をスケールＳ$_F$で割算して、割算結果ｙ１を求める（ステップ３７－１２）。そのｙ１に乗数Ａを掛算し、結果をｘ１とする（ステップ３７－１３）。ｘ１をｘ（ｉ）から引算して、結果をｚとする（ステップ３７－１４）。ｚが０でないかを調べる（ステップ３７－１５）。ｚが０でなければ、そのｚが所定値（この例では２３）ビット以下で表現できるかを調べる（ステップ３７－１６）。表現できなければ、ｙ’（ｉ）を０としてステップ３７－１８に移る（ステップ３７－１７）。ステップ３７－５でｚが０であれば、ステップ３７－１８に移る。同様にステップ

３７－１５でｚが０であれば、ステップ３７－１８に移る。それぞれその時得られているｙ’（ｉ）を、求めるｙ’（ｉ）とする。ステップ３７－１０でｘ１の方が小さくなければステップ３７－１８に移る。ステップ３７－１６でｚがＮビット以下で表現できればステップ３７－１８に移る。このようにしてｙ’（ｉ）を変形する処理が終了する。

【００８１】

なおステップ３７－１６でｚがＮビット以下で表現できない場合は、ｙ’（ｉ）を０とする代わりにＡ＝１．０としてもよい。この場合は、ｙ’（ｉ）は実施例１５又は実施例１７で説明した手法によってｘ（ｉ）から再度求め直す。

入力信号ｘ（ｉ）が浮動小数点形式ではなく整数形式表現の場合に共通の乗数を求めるには、その入力信号ｘ（ｉ）の極性をまず判断する。正であればｙ（ｉ）に対する変形処理を図３６に示した手法で行い、負であればｘ（ｉ）がｙ（ｉ）に対する変形処理は図３７に示した手法で行えばよい。

［実施形態９］

以下に前述の乗数推定部１１０における乗数Ａの決定について説明する。

実施例２０（有理近似）

有理近似による乗数Ａを推定する方法を、図３８を参照して説明する。仮数部変換部３８１は、入力された浮動小数点数ｘ（ｉ）の各仮数部の最上位ビット（ＭＳＢ）の１ビット上位に、１を配して符号なし２４ビット整数ｍｘ（ｉ）（ｉ＝０，１，・・・，Ｎ）とする。ただしｘ（ｉ）が、ＮａＮや非正規化数の場合はｍｘ（ｉ）＝０に変換する。この場合、全入力信号を入力することなく、例えばｘ（ｉ）を絶対値表現で小さい順に配列し、１フレーム（例えば２０４８サンプル）中の０以外の複数個（例えば５１２個）をソート部３８０で選出して仮数部変換部３８１に入力しても良い。

【００８２】

ＧＣＤ計算部３８２では、必要に応じて、通常用いられているユークリッドの互除法を用いて最大公約数ｇｃｄを計算する。ｇｃｄが判定部３８２ａの基準値よりも大きければ、その値を乗数として出力しても良い。しかし、この例ではそのｇｃｄをＡ０とし、分母選定部３８３が、例外値（ｍｘ（ｉ）＝０）以外のｍｘ（ｉ）をＡ０で割算し、この結果ｍｙ（ｉ）を出力する。そして、以下のアルゴリズムを使い、乗数Ａの推定値を計算する。その際にＡ０分の調整を加える場合もある。

このような乗数Ａの推定方法はいくつか考えられる。

第一の方法（最初の方法．基本）

前述したように例外値をｍｘ（ｉ）としたｍｘ（ｉ）を求めることは同様である。

（１）分母選定部３８３は、ｍｘ（ｉ）中の最大値を選択し、その値をＸとする。

（２）例外値（ｍｘ（ｉ）＝０）以外のｍｘ（ｉ）を一つずつ取り、下記式（１）を有理近似計算部３８４で計算する。

（ｍｘ（ｉ）－１／２）／（Ｘ＋１／２）＜ｍｙ（ｉ）／Ｙ（ｉ）
＜（ｍｘ（ｉ）＋１／２）／（Ｘ－１／２）　（２）

この式（２）を満たす分母最小の既約分数ｍｙ（ｉ）／Ｙ（ｉ）を求める。

【００８３】

ここで連分数展開を利用する。

有理近似計算部３８４における処理手順を以下に示す。

ステップ１．(mx(i)－1/2)/(X＋1/2)=nlow/dlow,(mx(i)+1/2)/(X－1/2)=nhigh/dhigh と既約分数で書く。

ステップ２．nlow/dlow＜１＜nhigh/dhighならば1/1を出力する。

ステップ３．dn=max{dlow,dhigh}とし、dnに対応する分子をnmとする。

ステップ４．ｎ０＝ｄ１＝０，ｎ１＝ｄ０＝１，ｉ＝０とおく。

【００８４】

ステップ５．k=[nm/dn]（nm/dnを超えない最大の整数），r=（nmをdnで割ったあまり）、nm＝dn，dn＝rとおく。

ステップ６．ｓ＝ｎ０＋ｎ１×ｋ，ｔ＝ｄ０＋ｄ１×ｋ，ｎ０＝ｎ１，ｄ０＝ｄ１とおく。

ステップ７．nm（i）＝n１＝s，dn（i）＝d１＝tとおく。dn＞０ならばiを１増やしステップ５へ戻り、dn＝０ならばステップ８へ移る。

ステップ８．nlow/dlow＜nm（i）/dn（i）＜nhigh/dhighとなる最小のiを求める（dn=dlowの場合は奇数のみ、dn=dhighの場合は２以上の偶数のみ調べれば十分）。

【００８５】

ステップ９．i＝１のときは、（１＋k×nm（０））/k＜nhigh/dhighを満たす最小の整数kを求め、（１＋k×nm（０））/kを出力する。

iが１より大きい奇数のときは、（nm（i－2）＋k×nm（i－1））/（dn（i－2）＋k×dn（i－1））＜nhigh/dhighを満たす最小の整数kを求め、（nm（i－2）＋k×nm（i－1））/（dn（i－2）＋k×dn（i－1））を返す。

iが偶数のときは、（nm（i－2）＋k×nm（i－1））/（dn（i－2）＋k×dn（i－1））＞nlow/dlowを満たす最小の整数kを求め、（nm（i－2）＋k×nm（i－1））/（dn（i－2）＋k×dn（i－1））を返す。

【００８６】

以上の処理が可能なことは高木貞治著「初等整数論理講義」第２版、２００３年９月１５日共立出版株式会社発行、１２４～１４５頁参照（以下非特許文献３という）から次のように理解される。この非特許文献３の１４０頁の定理２．７から以下が成立つ。

実数ωに対して誤差範囲ε＞０を指定したとき、

ω－ε＜ａ／ｂ＜ω

（またはω＜ａ／ｂ＜ω＋ε）

となる分母最小の分数ａ／ｂは、ωの奇数番目（または偶数番目）の主あるいは中間近似分数である。これは、定理２．７のすぐ下にある証明（全体で４行）の、「また」から始まる後半３行を言い換えたものである。

【００８７】

上記主張のωが前記有理近似計算部３８４のnm/dn，

εがnm/dn－nlow/dlow（またはnhigh/dhigh－nm/dn）である。

定理２．７のある第２章ではωを無理数としているが、１４０頁中程の［注意］にある通り、場合分けなどに注意すれば、結論は有理数の場合にも成立する。今の場合、有理近似計算部３８４のステップ３の、

「dn=max{dlow,dhigh}とし、dnに対応する分子をnmとする。」

がその注意にあたる（分母が大きい方をωとして連分数展開）。

（３）共通分母確認部３８５で例外値以外のｍｘ（ｉ）の数だけ得られたＹｉの最大値Ｙを求め、

（４）このＹの約数となるＹｉの数Ｎが、あらかじめ決めてある条件（下記に示す）を満たせば、共通因数計算部３８６で乗数Ａの推定値として、Ｙをシフトして１＜＝Ｘ／Ｙ＜２となるようにしたときのＸ／Ｙを、浮動小数点表現で計算したものを出力する。

【００８８】

条件を満たさなければ乗数Ａとして１を出力する。

前記あらかじめ決めてある条件の具体例は、例外値（ｍｘ（ｉ）＝０）を除いたデータ数を、ｍとすると下記の通りである。

Ｎ≧Ｃ（あらかじめ決めた数）

Ｎ≧ｃｍ（０＜ｃ＜１はあらかじめ決めておく）

入力信号によっては桁上りが生じる確率が知られており、１フレーム中で例えば２０％程度が桁上りとなる場合は、Ｎは７０％の値とする。また乗数Ａをなるべく利用したほうが符号量を小さくできる点からＮは２０％程度でも良い。またＡの精度をよくしたい場合は８０％程度とする。なおＡの正当性は後で調べる。

変形

前記（４）で、条件を満たすとき、Ｙの約数となるＹｉについて、ｍｘｉ／Ｙｉ（Ｙｉは条件を満たすようにシフトした上で）の平均をＡとして出力する。この平均によりＡの誤差が少なくなることが期待できる。条件を満たさないときは上記と同じ様にＡ＝１．０

を出力する。

【００８９】

前記（４）で、Ｙｉの最小公倍数Ｙを求め、それがＸ以下なら、Ｘ／Ｙ（Ｙをシフトした上で）を浮動小数点表現で計算した値を出力する。つまりＹｉは、必ず最大公約数となるか不明であるが、Ｙｉの最小公倍数を求めれば、必ず全体に対する最大公約数になる。あるいは、Ｙの約数となるＹｉから計算したものの平均を用いる。条件を満たさないときは上記と同様にＡ＝１．０を出力する。

このようにしてこれまでよりも広い範囲からＹｉを選択できる。

前記（１）で、最大値Ｘの代わりに、大きい方からＭ個（あらかじめ決めた数、２以上、好ましくは、３～５０内の数）を取り、それぞれについてＡを推定し、多数決で勝ったものつまりｍｘｉ中に最も多く共通に使用できるものを出力する。

【００９０】

この場合は、一個だけで求めると、たまたま素姓の悪いデータ（例えば桁上りしたものとしないものが混在した場合に対するＸ）を回避する。

前記（１）でランダムにＭ個（あらかじめ決めた数）を取り、以下同様に各Ａを推定し、多数決で決める。

この理由は大きい方からＭ個とる理由と同様である。

前記（１）で最大値の代わりに、大きい方から順に取り、それぞれについてＡを推定する。

【００９１】

１以外の推定値ＡがＭ個（あらかじめ定められた数）集まったところで、それを出力する。Ｍ個集まる前にデータがなくなれば、足りない分はＡ＝１として出力する。これら複数候補でいずれがよいかは検算ルーチンにまかせた方がよい。例えば、前記実施例３（図１１、図１２）で説明した変形処理、あるいは実施例４（図１３、図１４）で説明した変形処理を各乗数Ａごとに処理して、最も好ましいＡを選択する。

図３９に示すように、図３８に示した処理を次のように変更してもよい。分母選定部３９１は、ＧＣＤ計算部３８２で得られたｇｃｄＡ０で各ｍｘ（ｉ）を割算する。これは、分母選定部３９１で、ＧＣＤ計算部３８２で得られたｇｃｄを後の処理で再び求めないため、つまり無駄な処理を少なくするためである。有理近似計算部３９２は、ｍｘ（ｉ）／Ａ０を、有理近似計算部３８４でのｍｘ（ｉ）と同様に扱った処理を行う。またｍｘ（ｉ）／Ａ０として処理をしているため、共通因数計算部３８６は、得られた乗数候補をＡ０倍して乗数Ａとする。

【００９２】

図４０に示すように、図３８に示した処理を次のように変更してもよい。共通分母確認部４０２は、Ｙｉが見つからないと判定部４０２ａで判定すると、分母選定部４０１で他の１個の分母Ｘを選定させる。この選定は、ｍｘ（ｉ）の大きい順、あるいはランダムに行う。分母Ｘの選定をあらかじめ決めた回数、例えば５回行ってもＹｉがみつからない場合はＡ＝１．０を出力する。

図３９中に破線３９４で示すように、共通分母確認部３８５でＹｉが求まらない場合は、図４０で説明したと同様に、分母選定部３９１で他の分母Ｘを選択するようにしてもよい。分母選定部３９１ではＸをＭ個ではなく、１個ずつ選定する。

【００９３】

図４１の変形例では、有理近似計算部４１１は、各ｍｘ（ｉ）と対応したｍｙ（ｉ）／Ｙｉを求める。ここで、乗数Ａは、除算処理結果ｙ（ｉ）がなるべく小さいくなるものがよい。従って共通分母確認部４１２では、分母Ｙの最も小さい１個を選択し、共通分母となり得るかを確認する。共通分母となり得ない場合は、有理近似計算部４１１が、次に大きいＹとｙ（ｉ）の組を選択する。共通分母確認部４１２は、この組に対して共通分母となり得るかを確認する。このようなことを繰り返し、最も好ましいｙ（ｉ），Ｙ（ｉ）の組を決定する。図４１中に破線で示すように、最も大きい共通分母が得られなければ、最初に入力されたＹｉ，ｙｉをそれぞれ１単位ずつ増加させて、共通分母を求めるようにし

てもよい。

第二の方法（いくつか足してグループ作成）

この方法は、ｘ０，…，ｘｎをあらかじめ定めた数ずつ足して、相対的に誤差を減らす方法である。

【００９４】

最初に仮数部を符号なしの２４ビット整数形式ｍｘ（ｉ）に変換し、例外値はｍｘ（ｉ）＝０とする。そして、

（１）ｍｘ（ｉ）を大きい順に配列し、大きい方からＭ個ずつ（Ｍはあらかじめ決めておく）加えたものをＸｉとする（仮数部変換部３８１で）。

（２）Ｘｉの最大値を取りＸとする（分母選定部３８３で）。

（３）Ｘｉを一つずつ取り、

（Ｘｉ－１／（２√Ｍ））／（Ｘ＋１／（２√Ｍ））＜ｙｉ／Ｙｉ

＜（Ｘｉ＋１／（２√Ｍ））／（Ｘ－１／（２√Ｍ））

を満たす分母最小の既約分数ｙｉ／Ｙｉを求める（有理近似計算部３８４で）。

【００９５】

以下は第一の方法の（４）以下の処理と、その変形処理を行う。

このようにすると、第一の方法よりも相対的にデータの誤差が減るので、正しいＹを求める確率が上がる。

第三の方法（全探索その１．ｙベース）

この方法も、ｍｘ（ｉ）を求める処理は同じである。

（１）ｍｘ（ｉ）の最大値を取り、Ｘとする。

（２）ｍｘ（ｉ）＜Ｘとなるｉ最小の例外値（ｍｘ（ｉ）＝０）ではないｘｉを用い、

（ｘｉ－１／２）／（Ｘ＋１／２）＜ｙｉ／Ｙｉ＜（ｘｉ＋１／２）／（Ｘ－１／２）

を満たす分母最小の既約分数ｙｉ／Ｙを求める。

（３）例外値（ｍｘ（ｉ）＝０）以外のすべてのｘｊに対し、

（ｘｉ－１／２）／（Ｘ＋１／２）＜ｙｊ／Ｙ＜（ｘｉ＋１／２）／（Ｘ－１／２）

を満たす分母がＹとなる分数ｙｊ／Ｙが、存在するかどうか調べる。

（４）すべての場合に存在すれば、Ａの推定値としてＸ／Ｙを出力する。あるいは、ｘｊ／ｙｊの平均を出力する。

【００９６】

存在しないｊがあれば、

（ｘｉ－１／２）／（Ｘ＋１／２）＜ｙｉ／Ｙｉ＜（ｘｉ＋１／２）／（Ｘ－１／２）

を満たす分母であって、Ｙのつぎに小さい分数を求め、その分母を改めてＹとする。

もし、Ｙ＜Ｘなら（３）へ戻り、もし、Ｙ＝Ｘなら１を出力する。

第四の方法（全探索その２．Ａベース）

この方法では、Ａについての全探索を行う。

まず、前述の有理近似を用いた共通乗数を求める考え方を、図３８を参照して説明する。前述したように、ＧＣＤ計算部３８２では、入力信号ｘ（ｉ）に誤差が含まれていると正しい値を求めることができない。分母選定部３８３で、１フレーム中のｍｘ（ｉ）＝０以外の適当な代表値をＸ（分母）として選ぶ。この場合、最大公約数という点からなるべく情報量が多い信号（前記では最大値）が好ましい。

【００９７】

有理近似計算部３８４では、代表値Ｘとそのフレーム中の全てのｍｘ（ｉ）（前述したように、ｍｘ（ｉ）＝０を処理から除くことは、以後の説明では省略する。）を分子とする分数ｍｘ（ｉ）／Ｘにおいて、ｍｘ（ｉ）中に含まれている誤差を考慮した時に、ｍｘ（ｉ）／Ｘが、正しい値に対して負の誤差か正の誤差を含むが、その最大誤差範囲内にあるｙｉ／Ｙを求める。前述では四捨五入による誤差としたため、１／２を±したが、誤差が±αであるとすれば、次式を満たす分母最小の既約分数ｙｉ／Ｙｉを求めることになる。

【００９８】

（ｍｘ（ｉ）－α）／（Ｘ＋α）＜ｙｉ／Ｙｉ＜（ｍｘ（ｉ）＋α）／（Ｘ－α）

例えばｍｘ（ｉ）に±２の誤差があるとすればα＝２とする。正負の誤差が非対称の場合は＋α、－βを用いればよい。

この既約分数ｙｉ／Ｙｉは、先に示した非特許文献3中の連分数展開を利用した定理から必ず求まる。

このようにして求めたｙｉ／Ｙｉは、フレーム中のｍｘ（ｉ）の数だけある。従って、共通分母確認部３８５でＹｉ中の各ｍｘ（ｉ）に共通な値を求める。この場合、同一のＹｉの数があらかじめ決めた数以上であれば、そのＹｉを共通のＹとし、共通因数計算部３８６で、そのＹｉでＸを割算して、共通乗数Ａとして出力する。

実施例２１（自己相関利用）

乗数Ａを求める他の例を、図４２を参照して説明する。

【００９９】

各信号ｘ（ｉ）の仮数部の最上位ビット（ＭＳＢ）の１ビット上位に、“１”を配して２４ビットしたものをｍｘ（ｉ）とする。これとｍｘ（ｉ）をｊビット（ｊ＝１，２，…，２３）下位にシフトしたものとの自己相関を求める。図４２Ａは、ｍｘ（ｉ）と、これを１ビットシフトしたものｍｘ（ｉ）（１）とこれらの自己相関結果の例を示す。図４２Ａは、ｚ（ｉ）が１２ビット構成を例としている。

この自己相関値中の１のビット数を計数する。このことをｊ＝１，…，２３について行う。ｊごとに全てのｚ（ｉ）についての相関値中の１のビット数を合計する。この合計値がしきい値を超えたｊについての合計値を１とし、しきい値以下の場合を０とする。この判定結果のビット列が求める乗数Ａである。

【０１００】

例として、ｚ（ｉ）のビット数が８の場合の信号ｘ（０）～ｘ（４）について、ｊ＝１，２，…，９での各自己相関のビットが“１”の数を、図４２Ｂに示す。図４２Ｂで、ｊ＝１とした場合の自己相関値におけるビットが“１”の数は、ｘ（０）は５、ｘ（１）は３、ｘ（２）は２…である。ｊ＝２の場合は、ｘ（０）～ｘ（４）の自己相関値のビットが“１”の数は、全て０である。ｊ＝３の場合は、ｘ（０），ｘ（１），ｘ（２）…に対する自己相関値のビットが“１”の数は、２，４，１…である。これら自己相関値のビットが“１”の数を各ｊ＝１，２，３…ごとに合計した値が、２３，０，１５，…である。しきい値を１０とした場合、しきい値を超えたか否かの判定結果が、ｊ＝１，２，３…に対し１，０，１，…となる。この判定結果が、ｘ（０）～ｘ（４）に対する共通乗数Ａとされる。なおｊが大きくなるに従って、相関値中で１となるビット数が減少するから、しきい値をｊが大きくなる程、共通乗数Ａを小さくしてもよい。

実施例２２

音響信号としては、例えば２４ビット整数形式で信号化された信号素材を変形、振幅調整、効果付加、混合などの編集処理したものを浮動小数点形式に変換されたものが入力信号ｘ（ｉ）として入力されることがある。このような場合、元の２４ビット整数形式信号に対して行った加減乗除処理が記録されて、その加減乗除処理を知ることができる場合がある。この場合、その入力信号ｘ（ｉ）と共に加減乗除の演算順序と係数を符号化装置に与えられる。乗数推定部では、その入力された加減乗除情報をもとに乗数Ａを決定する。例えば定数Ａを元信号に乗算した場合は乗数Ａとしてその入力された定数Ａを出力すればよい。

【０１０１】

上述した符号化装置、復号化装置、有理近似共通乗数計算装置は、それぞれコンピュータにより機能させることもできる。この場合はコンピュータに、目的とする装置（各種実施例で図に示した機能構成をもつ装置）として機能させるためのプログラム、又はその処理手順（各実施例で示したもの）の各過程をコンピュータに実行させるためのプログラムを、ＣＤ－ＲＯＭ、磁気ディスク、半導体記録装置などの記録媒体から、あるいは通信回線を介してそのコンピュータ内にダウンロードし、そのプログラムを実行させればよい。

【図１】

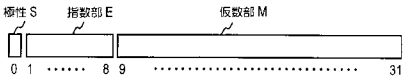


図1

【図２】

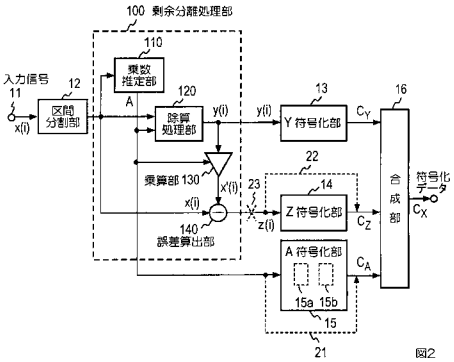


図2

【図３】

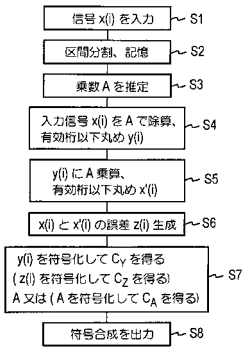


図3

【図４】

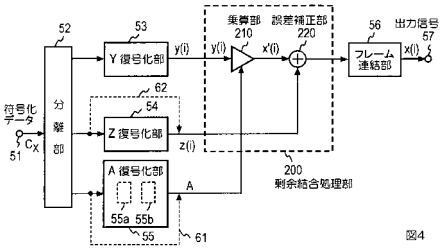


図4

【図５】

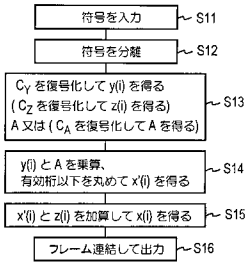


図5

【図６】

		10進数表現	2進数表現
A	x(1)	250	11111010
	x(2)	50	110010
	x(3)	350	101011110

B
- y(1) = x(1)÷A = 250÷1.5625 = 160
- y(2) = x(2)÷A = 50÷1.5625 = 32
- y(3) = x(3)÷A = 350÷1.5625 = 224

		10進数表現	2進数表現
C	y(1)	160	10100000
	y(2)	32	100000
	y(3)	224	11100000

図6

【図７】

	数値	符号	指数部	仮数部 (23bit)
A	478.4	0	8	11011110011001100110011
	95.68	0	6	01111110101110000101001
	669.76	0	9	01001110111000010100100

	数値	符号	指数部	仮数部 (23bit)
B	160 (= 478.4/2.99)	0	7	01000000000000000000000
	32 (= 95.68/2.99)	0	5	00000000000000000000000
	224 (= 669.76/2.99)	0	7	11000000000000000000000

図7

【図８】

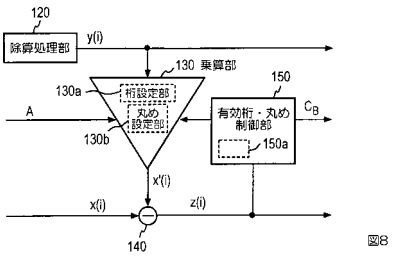


図8

【図９】

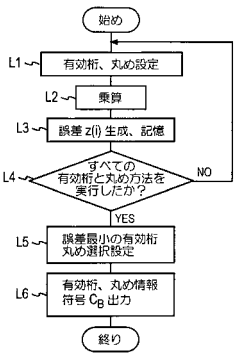


図9

【図１０】

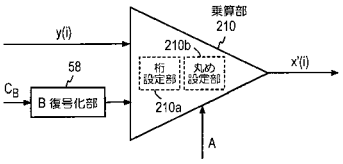


図10

【図１１】

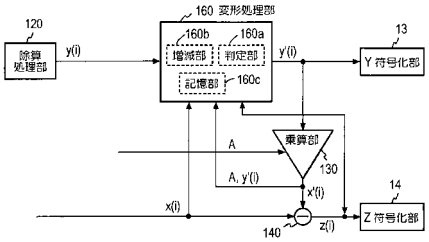


図11

【図１２】

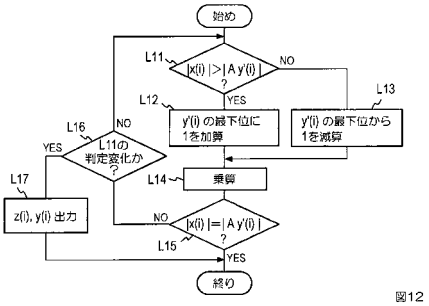


図12

【図１３】

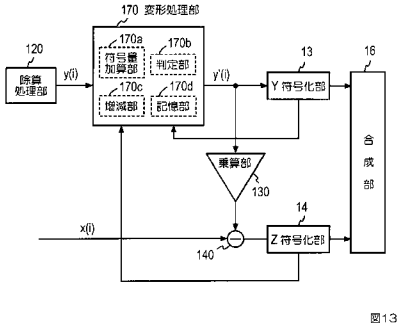


図13

【図１４】

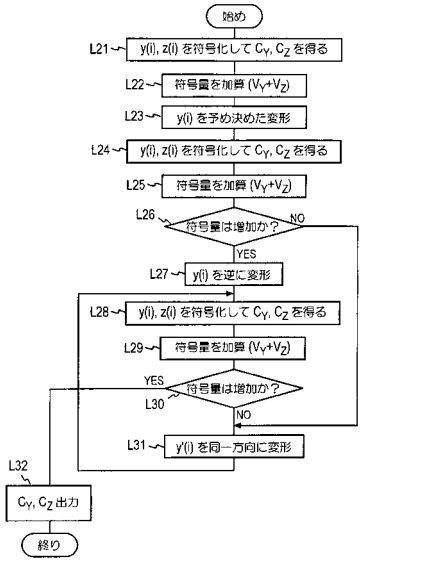


図14

【図１５】

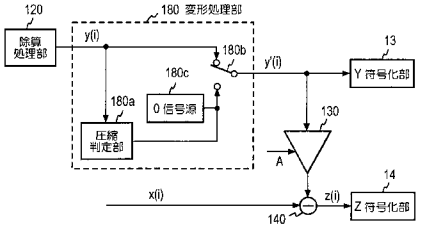


図15

【図１６】

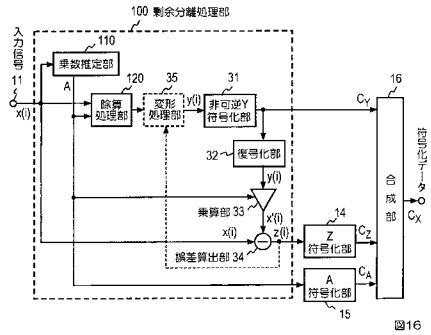


図16

【図１７】

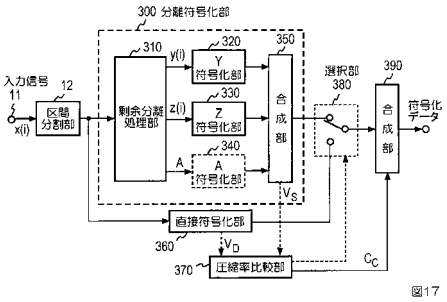


図17

【図１８】

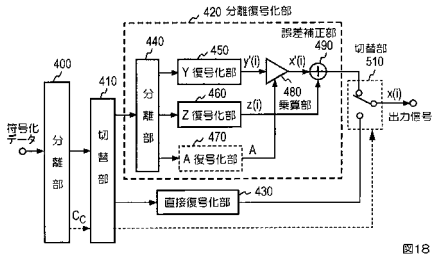


図18

【図１９】

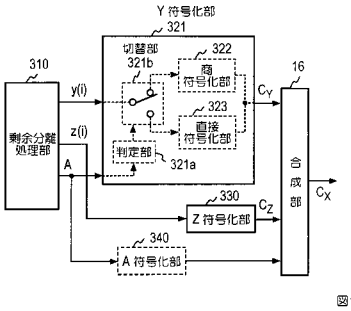


図19

【図２０】

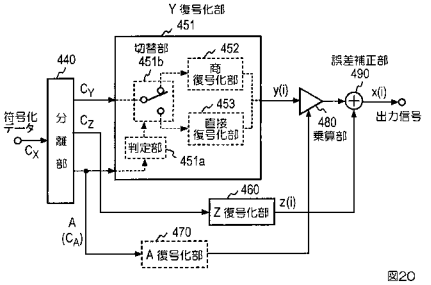


図20

【図２１】

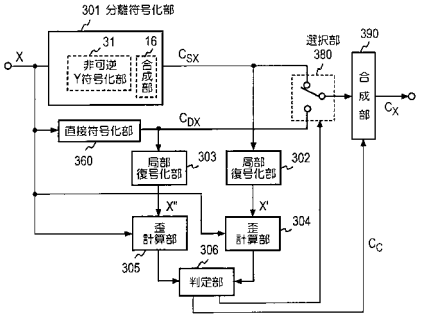


図21

【図２２】

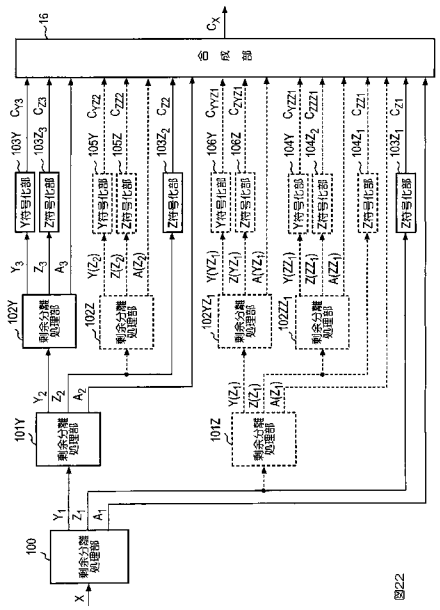


図22

【図２３】

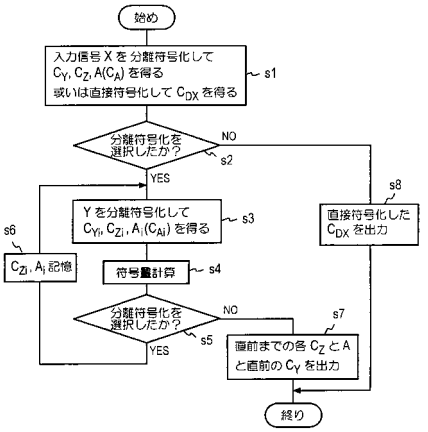


図23

【図２４】

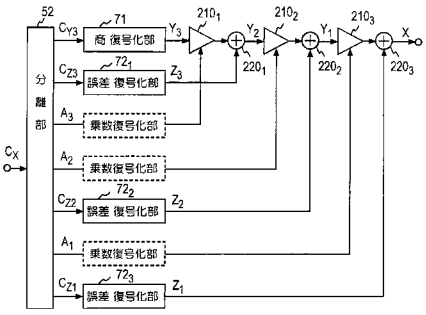


図24

【図２５】

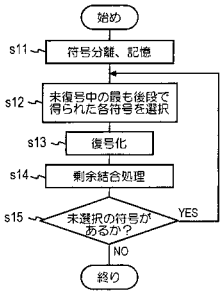


図25

【図２６】

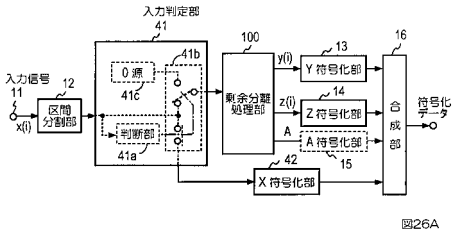


図26A

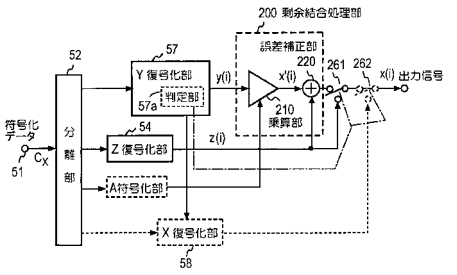
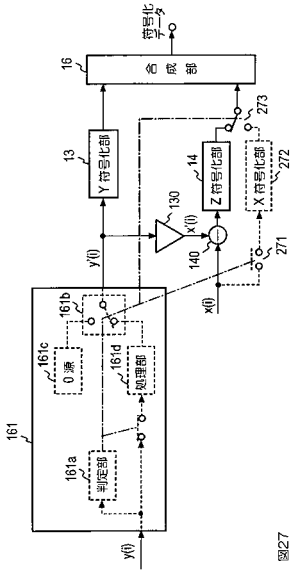
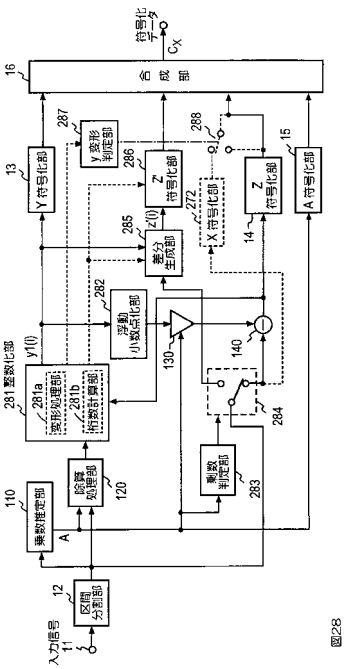


図26B

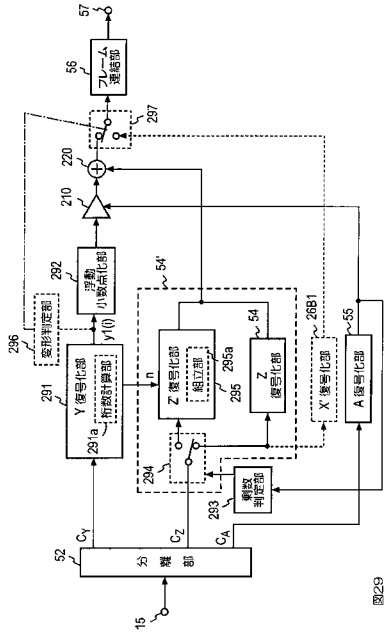
【図２７】



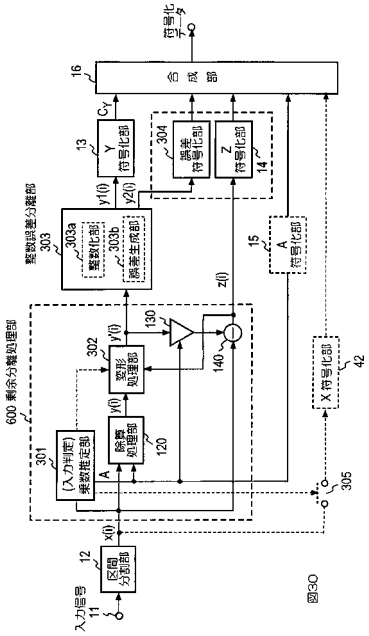
【図２８】



【図２９】



【図３０】



【図３１】

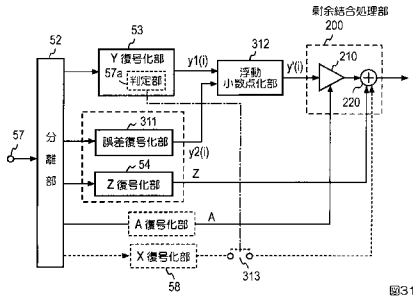


図31

【図３２】

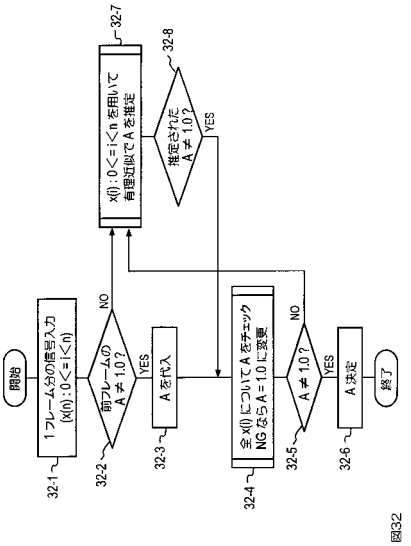


図32

【図３３】

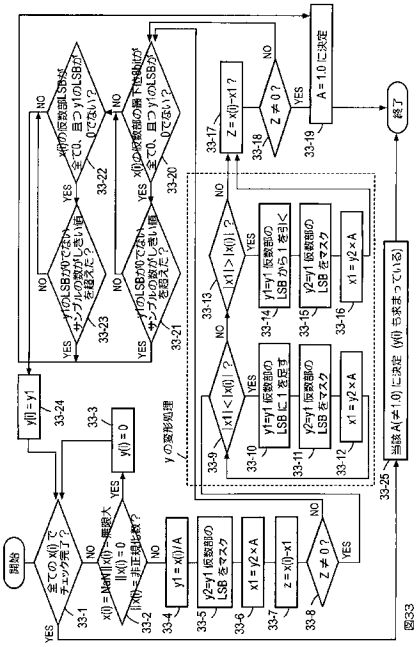


図33

【図３４】

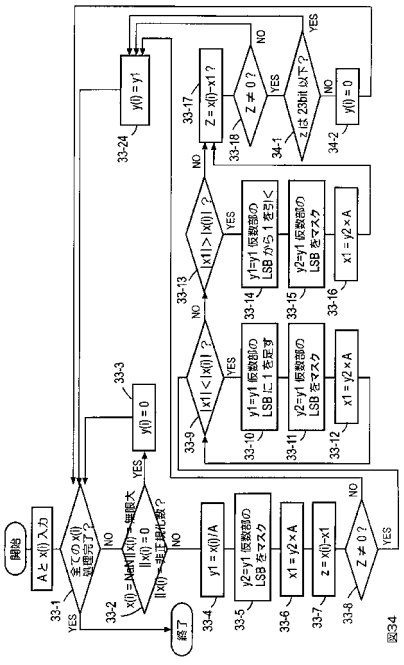
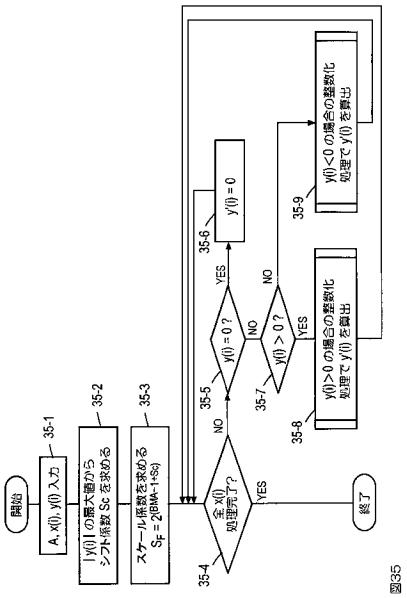
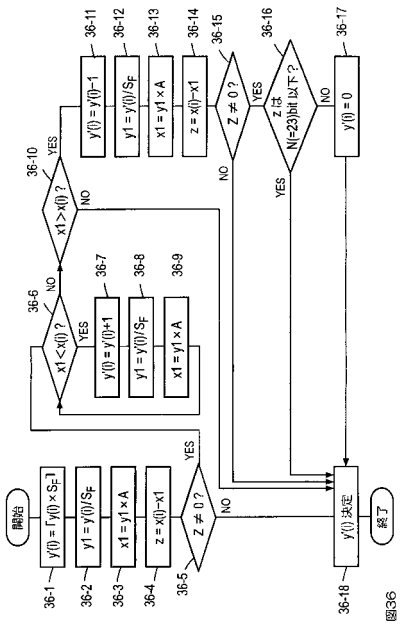


図34

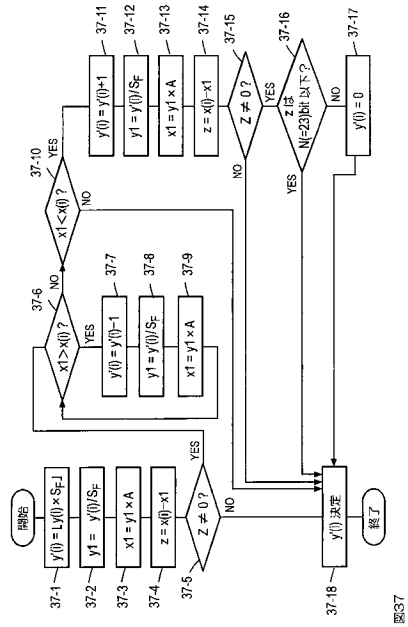
【図３５】



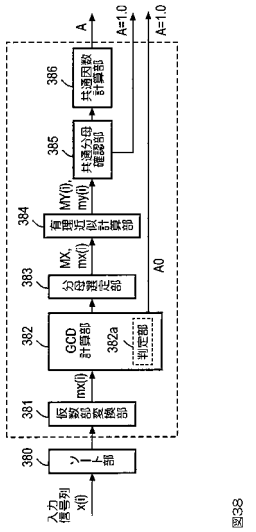
【図３６】



【図３７】



【図３８】



【図３９】

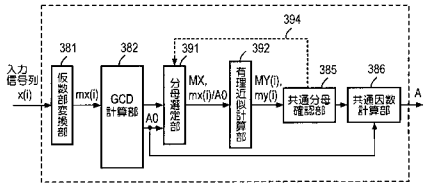


図39

【図４０】

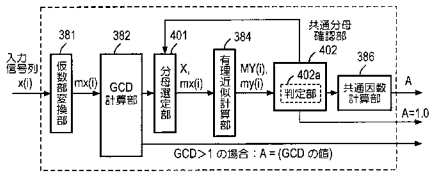


図40

【図４１】

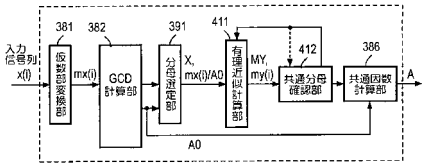


図41

【図４２】

A

z(i)	100100100100
z(i)(1)	100100100100
相関値	000000000000

B

j：	1	2	3	4	5	6	7	8	9
x(0)(j)：	5	0	2	0	0	0	3	2	0
x(1)(j)：	3	0	4	1	0	0	0	8	0
x(2)(j)：	2	0	1	0	0	3	1	1	0
x(3)(j)：	4	0	2	1	0	8	0	4	0
x(4)(j)：	9	0	6	0	0	8	0	6	6
合計：	23	0	15	2	0	19	4	21	6
判定：	1	0	1	0	0	1	0	1	0

図42

フロントページの続き

(72)発明者　守谷　健弘
東京都千代田区大手町二丁目３番１号　日本電信電話株式会社内
(72)発明者　白柳　潔
東京都千代田区大手町二丁目３番１号　日本電信電話株式会社内

審査官　田中　友章

(56)参考文献　特開２００３－３３２９１４（ＪＰ，Ａ）
特開２００４－２８９１９６（ＪＰ，Ａ）
特開平１１－１４５８４３（ＪＰ，Ａ）
特開平１０－０７８８６３（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
G06F　5/00
G10L　19/00
H03M　7/24
H03M　7/30
H03M　7/36