JAIST Repository
https://dspace.jaist.ac.jp/

Title	Temporal DecompositionとSTRAIGHTを用いた低ビットレート音声符号化に関する研究
Author(s)	越智，崇夫
Citation	
Issue Date	2002-03
Type	Thesis or Dissertation
Text version	author
URL	http://hdl.handle.net/10119/1569
Rights	
Description	Supervisor:赤木 正人，情報科学研究科，修士

JAIST
JAPAN
ADVANCED INSTITUTE OF
SCIENCE AND TECHNOLOGY

Japan Advanced Institute of Science and Technology

修 士 論 文

Temporal Decomposition と STRAIGHT を用いた低ビットレート音声符号化に関する研究

北陸先端科学技術大学院大学
情報科学研究科情報処理学専攻

越智 崇夫

2002 年 3 月

修 士 論 文

Temporal Decomposition と STRAIGHT を用いた低ビットレート音声符号化に関する研究

指導教官　赤木正人 教授

審査委員主査　赤木正人 教授
審査委員　党建武 助教授
審査委員　小谷一孔 助教授

北陸先端科学技術大学院大学
情報科学研究科情報処理学専攻

010029 越智 崇夫

提出年月: 2002 年 2 月

Copyright © 2002 by Ochi Takao

概 要

近年の携帯電話やマルチメディア通信の発達と普及率の増加に伴い、能率的な伝送または記録を行うことができる音声符号化の需要が増々高まっている。

より低ビットレートの音声符号化システムを構築するためには、音声学的情報を上手く捉えて符号化する必要があり、何が音声情報をよく特徴づけているかということが問題となる。現在、様々な手法を用いた低ビットレートの音声符号化の研究がが行われているが、2 kbit/s 以下のビットレートでは十分な品質の符号化システムは実現されていない。

本研究では、低ビットレート音声符号化における合成音の品質を向上させるために、音声分析・変換・合成方式として高音質な合成音を作成することができる STRAIGHT を用いた。STRAIGHT において、伝送側に送られる情報はスペクトル情報と基本周波数情報である。スペクトル情報は、Temporal Decomposition(TD) を用いて音韻情報に対応したパラメータに分解し、ベクトル量子化を行うことによって情報圧縮を行った。基本周波数情報に対しては、スカラー量子化を行った。最終的にそれらを基にした約 1.2 kbit/s の音声符号化システムを構築した。品質評価実験を行った結果、4.8 kbit/s CELP の品質には及ばないものの、2.4 kbit/s LPC10 よりも明らかに良い品質を持っていることがわかった。

目 次

第 1 章　序論 **1**
1.1 背景 1
1.2 目的 2
1.3 本論文の構成 2

第 2 章　符号化システムの概要 **3**

第 3 章　STRAIGHT **4**
3.1 スペクトルの抽出 4
3.1.1 信号モデル 5
3.1.2 時間方向の位相干渉の効果の軽減 5
3.1.3 周波数方向の平滑化 6
3.1.4 最適な平滑化関数 6
3.2 基本周波数の抽出 6
3.3 音声の再合成 7
3.3.1 オールパスフィルタの設計 8

第 4 章　スペクトルの符号化 **11**
4.1 LSF 11
4.2 制限と修正を加えた時間分解法 [5] 12
4.2.1 イベント関数 13
4.2.2 イベントターゲット 14
4.3 LSF の次数決定 16
4.3.1 次数に対するスペクトル歪みの変動 16
4.3.2 次数に対する音声品質の変動 18
4.4 ベクトル量子化 20
4.4.1 イベント関数の量子化 21
4.4.2 イベントターゲットの量子化 21

第 5 章　基本周波数の符号化 **22**
5.1 符号化方法 1 22

5.1.1 シミュレーション結果 24
5.2 符号化方法 2 26
5.2.1 シミュレーション結果 26
5.3 符号化方法 3 28
5.3.1 シミュレーション結果 28
5.4 考察 30

第 6 章 ゲインの符号化 31
6.1 符号化方法 1 31
6.1.1 シミュレーション結果 32
6.2 符号化方法 2 34
6.2.1 シミュレーション結果 34
6.3 符号化方法 3 36
6.3.1 シミュレーション結果 36
6.4 考察 38

第 7 章 雑音比の符号化 39
7.1 符号化方法 39
7.1.1 シミュレーション結果 40

第 8 章 提案法のビット割り当て 42

第 9 章 品質評価 44
9.1 目的 44
9.2 方法 44
9.3 実験結果と考察 45

第 10 章 結論 47
10.1 まとめ 47
10.2 今後の課題 47

図 目 次

2.1 TD と STRAIGHT を用いた低ビットレート音声符号化システム 3

4.1 スペクトル情報の符号化の流れ 11
4.2 イベントターゲットとイベント関数 13
4.3 2 つの隣接したイベント関数の例 14
4.4 スペクトル歪み評価 (上：LSF のみを用いた場合、LSF と MRTD を用いた場合) 17
4.5 LSF の次数に対するスペクトル歪みの変動 18
4.6 評価表 19
4.7 LSF の次数に対する音声品質の変動 19
4.8 分割ベクトル量子化 21

5.1 基本周波数（上：抽出した基本周波数、中：再構成した基本周波数、下：合成誤差）「原稿の締め切りはいつですか。」 25
5.2 基本周波数の量子化および再構成の流れ 26
5.3 基本周波数（上：抽出した基本周波数、中：再構成した基本周波数、下：合成誤差）「原稿の締め切りはいつですか。」 27
5.4 基本周波数の量子化 28
5.5 基本周波数（上：抽出した基本周波数、中：再構成した基本周波数、下：合成誤差）「原稿の締め切りはいつですか。」 29

6.1 ゲイン（上：抽出したゲイン、中：再構成したゲイン、下：合成誤差）「原稿の締め切りはいつですか。」 33
6.2 ゲインの量子化および線形補間による再構成の流れ 34
6.3 ゲイン（上：抽出したゲイン、中：再構成したゲイン、下：合成誤差）「原稿の締め切りはいつですか。」 35
6.4 ゲインの量子化およびスプライン補間による再構成の流れ 36
6.5 ゲイン（上：抽出したゲイン、中：再構成したゲイン、下：合成誤差）「原稿の締め切りはいつですか。」 37

7.1 雑音比（上：抽出した雑音比、中：再構成した雑音比、下：合成誤差）「原稿の締め切りはいつですか。」 41

9.1 評価表 44
9.2 品質評価実験の結果 46

表 目 次

8.1　ビット割り当て 43

第1章　序論

近年の携帯電話やマルチメディア通信の発達と普及率の増加に伴い、高能率な音声符号化の需要が高まっている。音声波形の符号化において波形を標本化し、量子化してデジタル化するために必要な基本的情報量は、8〜10 kHz サンプリング、10〜12 ビット線形量子化で 80 kbit/s 以上を必要とする。様々な手法により情報量（bit/s）を減らして能率的な伝送または記録を行う技術を低ビットレート符号化という。より低ビットレートの音声符号化システムを構築するためには、音声学的情報を上手く捉えて符号化する必要があり、何が音声情報をよく特徴づけているかということが問題となる。

1.1　背景

音声符号化方式は大きく分けて波形符号化方式、分析合成符号化方式、ハイブリッド符号化方式がある。

波形符号化方式は、基本的に音声波形そのものをできるだけ忠実に再生することを目標とする逐次処理方式を指し、主に 16〜64 kbit/s の符号化のために設計されている。分析合成方式は、符号器側で音声生成モデルに基づいて音声信号からモデルのパラメータのみを抽出し、復号器側でそのパラメータから音声を合成する。この方式では、短区間での平均的なパラメータしか伝送しないので、情報圧縮効果は大きく、2.4〜4.8 kbit/s 程度の明瞭で雑音間の少ない合成音声が得られる。ハイブリッド符号化方式は、主に 4〜16 kbit/s の中帯域の符号化のために設計されている。このビットレートの領域では、従来の波形符号化方式、分析合成方式は共に、特に通信用途の満足できる符号化方法にはならないことが動機になっている。これらの方式において、何らかの意味で入力の波形に近い波形を合成することを目的にしている点は波形符号化と共通で、短い区間のブロックごとにパラメータを抽出する点は分析合成方式と共通している。

低ビットレート音声符号化において、より高い情報圧縮を実現するためには分析合成符号化方式を用いる必要があるが、この方式では他の符号化方式に比べて高品質な合成音を得ることができないという欠点がある。しかし、河原らによって提案された音声分析・変換・合成方式 Speech Transformation and Representation based on Adaptive Interpolation of weiGHTed spectrogram (STRAIGHT)[1, 2, 3] は、分析合成方式ながら高品質な合成音を得られる方式として注目を浴びている。

STRAIGHT において、合成器側に送られる情報は主にスペクトル情報と基本周波数情報である。STRAIGHT のスペクトル情報は、１ ms という分析フレーム周期で、FFT

（1024 ポイント）パワースペクトルを平滑化して得られるものである。よって、1 フレームに対して 513 次という膨大な量の情報を持っており、そのまま量子化したのではビットレートが大きくなる。したがって、STRAIGHT を低ビットレート音声符号化に用いるには、スペクトル情報のパラメータ化が必要である。

STRAIGHT を用いた低ビットレート音声符号化として、東山らは 7.8 kbit/s の符号化システムを構築している [12]。この手法は、スペクトルのおおまかな形を Line Spectral Pairs (LSP) で表現し、その残差を零位相化して表現することで、スペクトル情報をパラメータ化している。スペクトルパラメータは、多段分割ベクトル量子化を用い、基本周波数情報はスカラー量子化を行っている。他の音声符号化方式との品質評価実験を行った結果、ほぼ同等のビットレートを持つ CS-ACELP[10][11] に及ばないが、5bit μ-law PCM の品質と同等程度であることが示されている。しかし、音声符号化方式における分析合成符号化としてはまだまだ情報量が多い。また、STRAIGHT を用いた低ビットレート音声符号化の研究は、まだそれほど行われていないのが現状である。

1.2 目的

現在、様々な手法を用いて低ビットレートの音声符号化が行われているが、2kbit/s 以下のビットレートでは十分な品質の符号化システムは実現されていない。

本研究では、合成音の品質を向上させるために、音声分析・変換・合成方式として高音質な合成音を作成することができる STRAIGHT を用いる。しかし、符号化システムとしては、かなり多くの情報を伝送することになり、情報圧縮という点では不利である。そこで、STRAIGHT により音声データからスペクトル情報と基本周波数情報を抽出した後に、Temporal Decomposition（TD）を用いて音声信号の時間的な変動に極在して現れる音声学的情報を分解する。分解することによって、より低ビットな特徴づけを目指す。さらに、それを基にした低ビットレート音声符号化システムを構築することを目的とする。

1.3 本論文の構成

第 1 章では、STRAIGHT および TD を用いた低ビットレート音声符号化に関する研究の現状と問題点を指摘し、本論文の目的を明らかにする。第 2 章では、本論文で提案する低ビットレート音声符号化システムの概要を示す。第 3 章では、音声分析・変換・合成法 STRAIGHT の構造について説明する。第 4 章では、スペクトル情報の符号化について説明する。第 5 章では、基本周波数情報の符号化について説明する。第 6 章では、ゲイン情報の符号化について説明する。第 7 章では、雑音比情報の符号化について説明する。第 7 章では、提案法におけるパラメータのビット割り当てについて説明する。第 9 章では、提案法と他の符号化方法の合成音を比較することによって品質評価を行う。第 10 章では、全体のまとめを行い、今後の課題を示す。

第2章　符号化システムの概要

図 2.1 にシステムの概要を示す。

符合器側では、STRAIGHT によって、音声信号から平滑化されたスペクトル情報と基本周波数情報（F_0）が抽出される。スペクトル情報は Line Spectral Frequencies (LSF) に変換し、Modified Restricted Temporal Decomposition (MRTD)[5] を用いてスペクトルパラメータの時間変化パターン（イベント関数）とスペクトルの安定する位置におけるスペクトル情報（イベントターゲット）に分解する。分解したパラメータをベクトル量子化することにより、スペクトル情報を圧縮する。符合帳の学習には、LBG アルゴリズム [8][9] を用い、符合帳探索・学習の距離尺度にはユークリッド距離を用いる。その他のパラメータに対しては、スカラー量子化を適用する。

復号器側では、量子化されたパラメータを基にスペクトル情報と基本周波数情報を再構成し、STRAIGHT によって合成音を作成する。

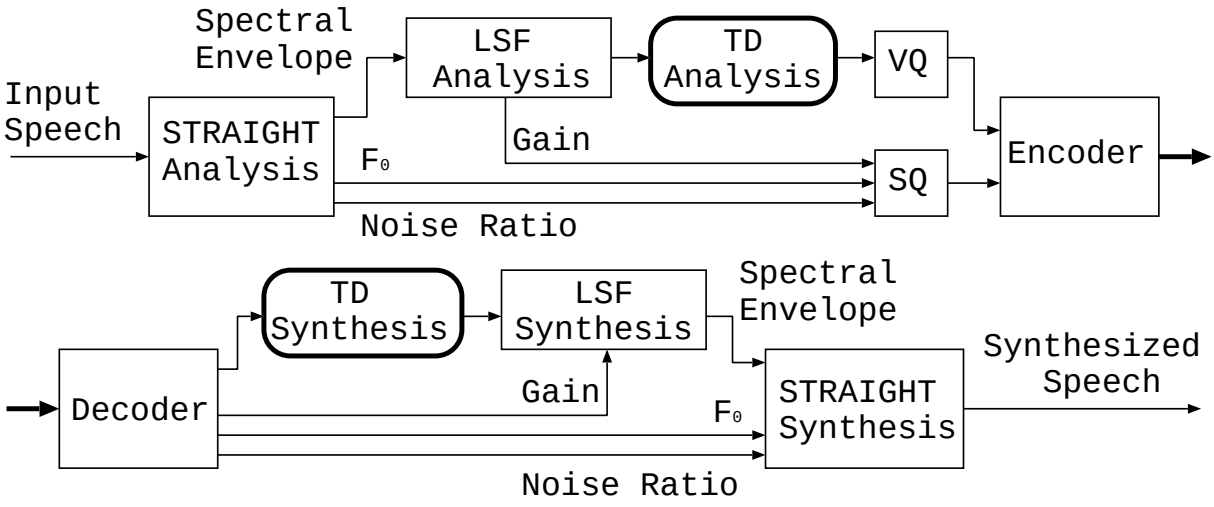

図 2.1: TD と STRAIGHT を用いた低ビットレート音声符号化システム

第3章　STRAIGHT

STRAIGHT は、スペクトル情報を抽出する STRAIGHT-core、基本周波数を推定する Time-domain Excitation extraction based on a Minimum Perturbation Operator (TEMPO)、合成に用いる駆動音源の位相特性を操作する Synthetic Phase Impulese for Keeping Equivalent Sound (SPIKES) の 3 つの主要な部分から構成されている。

STRAIGHT-core は、音声の励振の周波数性による干渉の影響のない時間周波数表現を抽出する方法である。その中心的なアイディアは基本周期、基本周波数を接点とする区分的線形関数による補間と等価な時間周波数領域の平滑化を行うことにある。

TEMPO は、2 つのフィルタ出力の微細の特性を基に、音声の基本周波数を推定する方法である。特別なフィルタ設計と搬送対雑音比（C/N 比）の組み合わせにより、基本周波数の推定が正確なものなっている。

SPIKES は、合成に用いる駆動音源の位相特性を操作することにより、VOCODER 特有の buzzy な音色を軽減する方法である。ここでは、同一のパワースペクトルであっても郡遅延を操作して時間的な微細構造を変えることで音色が変化することを利用している。

3.1　スペクトルの抽出

原音声のスペクトル情報は STRAIGHT-core によって抽出される。STRAIGHT-core の重要なアイディアは、有声音に見られる周期的な励振を、直接には観測できない仮想的な時間周波数局面を時間周波数領域で組織的にサンプリングする役割を担うものであると解釈するところにある。STRAIGHT-core の原理は、この解釈の下、サンプリングされた限られた局所的情報から局面を復元するために、2 次の cardinal B-spline の基底関数を平滑化関数として用いていることにある。ここで、基底関数を補間関数ではなく平滑化関数として用いることで、周期性の影響を雑音や誤差に強い形で選択的に除去することを狙っている。

3.1.1 信号モデル

音声を、常に周波数の変動する基本波とそれにほぼ同期したイベントに駆動される高次の周波数からなる信号であると考える。

$$s(t) = \sum_{k \in N} \alpha_k(t) \sin\left(\int_{t_0}^{t} k(\omega_0(\tau) + \omega_k(\tau)) d\tau + \phi_k \right) \tag{3.1}$$

ここで、$\omega_0(t)$ は、基本波の角周波数、$\omega_k(t)$ は、k 番目の高次調波成分の角周波数を表す。また、α は、それぞれの成分の強さを表し、ϕ_k は、k 番目の高次調波成分の初期位相を表す。この信号の短時間フーリエ変換は、調波構造と調波間の干渉のため、周波数方向に $f_0(t) = \omega_0(t)/2\pi$、時間方向に $\tau_0 = 1/f_0$ のほぼ周期的な構造を有することになる。

3.1.2 時間方向の位相干渉の効果の軽減

実効的な長さが 1 基本周期上でサイドローブが十分に軽減しているような時間窓を用いれば、分析位置による短時間スペクトルの変動の解析は、隣接する調波の相互作用を考えるだけで良い。例えば、次のように定義される Gauss 型時間窓は、そのような窓の一例である。ここで η は、窓の時間方向の伸長の程度を示すパラメータである。

$$w_G(t) = e^{-\pi \left(\frac{t}{\eta^{t_0}} \right)^2} \tag{3.2}$$

このような窓を用いて周期信号を分析すると、周期的にパワースペクトルが零となる部分が出現する。この零となる部分を埋めて時間的に変動しないパワースペクトルを得ることが最初のステップである。パワースペクトルが零となるのは調波と調波の中間の周波数で上の調波成分と下の調波成分の位相が逆になる部分である。したがって、ある窓 $w(t)$ に対して上下の調波の位相を π だけ回転させるように作った相補的な窓 $w_c(t)$ を用いて計算した短時間スペクトルは、元の窓によるスペクトルが零の部分で最大値を持つようになる。

$$w_c = w(t) \sin\left(\pi \frac{t}{\tau_0} \right) \tag{3.3}$$

時間方向に伸長した時間窓 $(\tau > 1)$ で得られたスペクトル $P_0(\omega, t)$ と、その相補的な窓から求められたスペクトル $P_c(\omega, t)$ とを、次のような加重和として合成することにより、時間標高での周期変動のないスペクトル $P_\tau(\omega, t)$ が求められる。

$$P_\tau(\omega, t) = \sqrt{P_0^2(\omega, t) + \xi(\tau) P_c^2(\omega, t)} \tag{3.4}$$

ここで、$\xi(\tau)$ はスペクトルの時間方向の分散を最小にする混合係数である。なお、時間方向に少しだけ引き延ばすだけで、$P_\tau(\omega, t)$ の時間方向の周期的変動は実質的に無視することができる。

3.1.3 周波数方向の平滑化

基本周波数に応じて適応的に変化する次のような2次のcardinal B-spline基底関数$h_t(\omega)$を周波数方向の平滑化関数とする。

$$h_t(\omega) = 1 - \left|\frac{\omega}{\omega_0(t)}\right| \tag{3.5}$$

ここで、$\omega_0(t) = 2\pi f_0(t)$であり、$-\omega_0(t) \leq \omega \leq \omega_0(t)$である。$P_r(\omega, t)$をこの平滑化関数を用いて次式により平滑化することで、周期的な励振の影響が除かれた時間周波数表現$S(\omega, t)$が得られる。

$$S(\omega, t) = \sqrt{g^{-1}\left(\int_D h_t(\lambda, t) g(|P_r(\omega - \lambda, t)|^2) d\lambda\right)} \tag{3.6}$$

ここでDは、平滑化関数の定義域を表す。式(2.6)の中の$g()$は、平滑化操作によって保存すべき量を定めるのに利用される。

3.1.4 最適な平滑化関数

前節で説明した原理を直接適用しただけでは、再合成音の品質はあまり良くない。これは、時間窓による周波数方向の平滑化と平滑化関数$h_t(\omega$による平滑化が重なることにより、過剰な平滑化が行なわれてしまうためである。この過剰平滑化の影響を避けるために、最適平滑化関数を用いる必要がある。最適平滑化関数は、spline関数の性質を利用すると、窓関数の周波数表現と2次のcardinal B-spline基底関数の畳み込めを基本周波数の間隔で標本化した系列をインパルス応答とみなしたときの逆フィルタの対応を計算することで求めることができる。

3.2 基本周波数の抽出

基本周波数情報はTEMPOによって、より正確に推定することができる。TEMPOは、帯域フィルタの中心周波数とフィルタ出力の瞬時周波数を周波数から周波数への写像とみなし、信号の主要な正弦波の周波数をこのような写像の安定な平衡点に対応する瞬時周波数として求める方法である。

基本周波数推定のために瞬時周波数を使うには、推定に先立って分離され選択される基本周波数成分が必要である。これは、log周波数軸に沿って等しい間隔を保つフィルタからなる帯域通過フィルタ、特別にに設計されたインパルス応答と選択機構によって行なわれる。フィルタのインパルス応答$\omega_s(t, \lambda)$は、STRAIGHT-coreで示したガボール関数（式3.2）とcardinal B-spline基底関数（式3.5）の畳み込みによって得られる。

フィルタの中心周波数 λ からフィルタ出力の瞬時周波数 $\omega_c(t_i, \lambda)$ へ等しく写像される点（不動点）の集合 $\Lambda(t)$ は次のように定義される。

$$\Lambda(t) = \{\lambda | \omega_c(t_i, \lambda) = \lambda, \omega(t_i, \lambda - \epsilon) - (\lambda - \epsilon) > \omega_c(t_i, \lambda + \epsilon) - (\lambda + \epsilon)\} \tag{3.7}$$

ϵ は任意の小さな定数を表す。さらに、$\Lambda(t)$ から、基本周波数に対応する点を選択しなければならない。STRAIGHT では C/N 比を推定し、これが最も低い不動点を用いることにより基本周波数を推定している。

C/N 比と不動点周辺の写像の形状

主要な正弦波成分以外による干渉をモデル化すると、不動点における写像の偏導関数を組み合わせて近似的に C/N 比を求めることができる。C/N 比は、不動点の周波数を λ としたとき、不動点近傍での写像の幾何学的性質に基づいて以下のようにして求められる $\bar{\sigma}$ を用いて $C/N = 1/\bar{\sigma}$ として近似的に推定することができる。

$$\bar{\sigma}^2(t, \lambda) = \int_{-T_w}^{T_w} |w(\tau, \lambda)| \tilde{\sigma}(t - \tau, \lambda) d\tau \tag{3.8}$$

$$\tilde{\sigma}^2 = c_a \left(\frac{\partial \omega(t, \lambda)}{\partial \lambda} \right)^2 + c_b \left(\frac{\partial^2 \omega(t, \lambda)}{\partial t \partial \lambda} \right)^2 \tag{3.9}$$

$$c_a = \frac{1}{\int_{-\infty}^{\infty} \left(\lambda_0 \frac{dg(\lambda)}{d\lambda} |_{\lambda=\lambda_0} \right)^2 d\lambda_0}$$

$$c_b = \frac{1}{\int_{-\infty}^{\infty} \left(\lambda_0^2 \frac{dg(\lambda)}{d\lambda} |_{\lambda=\lambda_0} \right)^2 d\lambda_0}$$

ここで T_w は、関数 $|w(\tau, \lambda)|$ が実質的に 0 でない範囲を覆うことができるように設定する。

3.3 音声の再合成

STRAIGHT-core により求められた時間周波数表現から音声を再合成する方法として、複素ケプストラムを介して最小位相インパルス応答を求め、位相調整して再配置する方法について説明する。このような方法を用いることにより、基本周波数 $f_0(t)$ の精密な制御と、時間的微細構造に依存する音色の制御が可能となる。

この方法による音声の変換と合成は、正式的には次のように表すことができる。合成さ

れた音声波形を $y(t)$ とする。

$$y(t) = \sum_{t_i \in Q} \frac{1}{\sqrt{G(f_0(t_i))}} v_{ti}(t - T(t_i)) \tag{3.10}$$

$$v_{ti}(t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} V(\omega, t_i)\Phi(\omega)\exp(j\omega(t))d\omega \tag{3.11}$$

$$\text{where} \quad T(T_i) = \sum_{t_k \in Q, k<i} \frac{1}{G(f_0(t_k))} \tag{3.12}$$

ここで、Q は合成のための駆動信号を置く位置の集合であり、$G()$ は基本周波数の変換を表す。オールパスフィルタ $\Phi = \Phi_1\Phi_2\Phi_3\Phi_4$ の特性を次節で説明するように操作することにより、基本周波数と音色が制御される。

ここで、$V(\omega, t_i)$ は、$A()$、$u()$、$r()$ をそれぞれ振幅、周波数、時間軸の変換としたとき、変換された振幅スペクトル $A(S(u(\omega), r(t)), u(\omega), r(t))$ から次のようにして複素ケプストラム $h_t(q)$ を介して求めた、最小位相インパルス応答のフーリエ変換である。ただし、q はケフレンシーを表す。

$$V(\omega, t) = \exp\left(\frac{1}{\sqrt{2\pi}} \int_0^{\infty} h_t(q)\exp(j\omega q)dq\right) \tag{3.13}$$

$$h_t(q) = \begin{cases} 0 & (q < 0) \\ c_i(0) & (q = 0) \\ 2c_i(q) & (q > 0) \end{cases} \tag{3.14}$$

$$\text{and} \quad c_i(q) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \exp(j\omega q)\log A d\omega \tag{3.15}$$

3.3.1 オールパスフィルタの設計

オールパスフィルタの第 1 成分 $\Phi_1(\omega)$ は、次式で表される時間遅れに相当する直線位相成分である。$\Phi_1(\omega)$ は、基本周波数の精密な制御に用いられる。

$$\Phi(\omega) = \exp(j\omega f_s T_d) \tag{3.16}$$

ここで、$\omega = 2\pi f/f_s$ は正規化角周波数を表し、T_d は時間遅れを表す。離散時間制系での実装にあたっては、正規化角周波数が 2π のときの値が n を任意の整数とすると $2n\pi$ であるとの拘束条件を満たすように、次式を用いる。

$$\Phi_1(\omega) = \exp(j(\pi f_s T_d + q(\omega))) \tag{3.17}$$

$$p(\omega) = \begin{cases} \dfrac{2\pi a}{1 + \exp((\omega + \pi)/\omega_w)} & \omega \geq 0 \\ \dfrac{2\pi a}{1 + \exp((\omega - \pi)/\omega_w)} & \omega < 0 \end{cases} \tag{3.18}$$

$$a = \lceil f_s T_d \rceil - f_s T_d \tag{3.19}$$

この式では、ナイキスト周波数での位相の不連続を、指数関数を利用して滑らかに接続することにより、特異点の影響の時間領域での局在化を図っている。$f_w = f_s\omega_w/2\pi$ は、位相の不連続を滑らかにつなぐ区間（遷移帯域）の幅を表す。

位相制御の性質は、次のような成分を考えるとわかりやすい。

$$\Phi_2(\omega) = \exp\left(-j\rho(\omega)\sum_{k\in P}\sin(k\omega)\right) \tag{3.20}$$

ここで、P は添字の集合、$\rho(\omega)$ は周波数帯ごとの位相の広がりを制御するための加重である。$\rho(\omega)$ の具体例としては、次のようなものがある。ここで ω_0 は、位相が広がりはじめる領域を定める遷移周波数を表し、b_w は、遷移領域の広さを表す。

$$\rho(\omega) = \frac{1}{1+\exp(-\frac{\omega-\omega_0}{b_w})} \tag{3.21}$$

この位相特性が単一の空間周波数のみからなっており、加重も全ての帯域で一定 $(\rho(\omega)=1)$ であるとしたとき、時間領域の表現は無限個のパルスから構成されることが分かる。ただし、パルスの振幅は急速に減少するので、実質的に 0 とみなすことのできるパルスの高さを ϵ とすると、サンプル数で表したパルスの拡がり T_w は、次式で求められる。

$$\begin{aligned} |\alpha| &\leq (\epsilon\Gamma(\beta+1))^{\frac{1}{\beta}}2^{\frac{\beta-1}{\beta}} \\ \beta &= \frac{T_w}{k} \end{aligned} \tag{3.22}$$

ここで $\Gamma()$ は、ガンマ関数を表す。複数の空間周波数成分が存在する場合には、全体の拡がりと ϵ は、各々の積となる。

乱数を用いたオールパスフィルタの設計では、位相ではなく郡遅延を操作の対称とする。$n(t)$ をガウス雑音とし、$W_s(\tau)$ を空間周波数領域での加重とする。目的とする郡遅延の拡がり（標準偏差）を σ_g とする。すると郡遅延特性 $\tau_g(\omega)$ は次のように計算される。

$$\tau_g(\omega) = \rho(\omega)\frac{\sigma_g x(\omega)}{\sqrt{\frac{1}{2\pi}\int_{-\pi}^{\pi}|x(\omega)|^2 d\omega}} \tag{3.23}$$

$$\begin{aligned} x(\omega) &= F^{-1}(W_s(\tau)N(\tau)) \\ W_s(\tau) &= |\tau|\exp(-\pi(\tau/\tau_{bw})^2) \end{aligned} \tag{3.24}$$

ここで $N(\tau)$ は初期値として用いた乱数 $n(\omega)$ からつくられた郡遅延である。$F^{-1}()$ は、逆フーリエ変換を表す。位相特性 Φ_3 は、この郡遅延 $\tau_g(\omega)$ を積分することで求められる。

これまでに説明した方法で求められたオールパスフィルタの時間波形では、パルスは一般的に時間的に対称的にに分布している。しかし、時間的な対称性は聴覚の情景分析の手がかりとして非常に重要である。実際、人間は 1 ms 以下の半減期の非対称性を明らかな音色の違いとして聞き分けることができる。この音色成分の制御のためのオールパスフィ

ルタの時間応答の非対称性は、次のようにして実現することができる。

$$\Phi_4(\omega) = \exp\left(-j\int_{-pi}^{\omega} r\left(j\frac{d\log\Phi(\lambda)}{d\lambda}\right)d\lambda\right)+c_0 \tag{3.25}$$

ここで $r()$ は、任意の単調で滑らかな偶関数である。一例を以下に示す。

$$r_2(x) = \log(2\cos hx) \tag{3.26}$$

なお、$\Phi_2(\omega)$、$\Phi_3(\omega)$、$\Phi_4(\omega)$ についても、$\Phi_1(\omega)$ の場合と同様にナイキスト周波数において位相が連続になるように補正を行なっている。

第4章　スペクトルの符号化

この章では、スペクトル情報の符号化方法について詳しく述べる。

スペクトル情報は、LSF に変換され、MRTD によってイベントターゲットとイベント関数に分解される。この時、イベント間の距離も求める。イベントターゲットとイベント関数は、ベクトル量子化を行い、イベント間の距離はスカラー量子化を行う。図 4.1 にスペクトル情報の符号化の流れを示す。

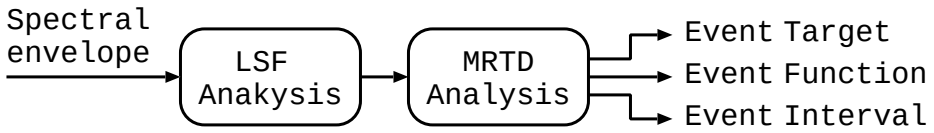

図 4.1: スペクトル情報の符号化の流れ

次のような理由から、スペクトル情報は LSF に変換する。

- 線形補間性が優れている。
- TD に用いられているスペクトルパラメータの中で、より歪みが少なく再現性が良い。

また、MRTD を用いる理由を以下に示す。

- スペクトルパラメータの音韻情報に対応した分解。
- 分解によって実際に量子化するパラメータの削減。

4.1　LSF

STRAIGHT で得られる振幅スペクトル $X[k]$、$0 \leq k \leq N/2$ を用いてパワースペクトル $S[k]$ を計算する。

$$S[k] = |X[k]|^2, \quad 0 \leq k \leq N/2 \tag{4.1}$$

パワースペクトルからフーリエ逆変換することによって相関関数を求めると次のようになる。

$$R[n] = \frac{1}{N}\sum_{k=0}^{N-1} S[k]\exp\left\{j\frac{2\pi kn}{N}\right\} \tag{4.2}$$

ここで、$S[k] = S[N-k]$。この相関関数を有する過程 $x(n)$ が全極フィルタ（次数 L）からの出力と仮定すれば、フィルタの係数を a_l^L、$l = 1, 2, ..., L$、$0 < L < N/2$ として、

$$P_L = R[0] - \sum_{l=1}^{L} a_l^L R[l] \tag{4.3}$$

と書ける。ここで、P_L は誤差（ゲイン）である。P_L が最小となるようにフィルタの係数 a_l^L を決定する。このときのフィルタの係数は、LPC の予測係数と一致する。

予測係数 a_l^L を用いて、次のような Z^{-1} の多項式をつくる。

$$\begin{cases} A_L(Z) = 1 - \sum_{l=1}^{L} a_l^L Z^{-1} \\ B_L(Z) = Z^{-(L+1)} A_L(Z^{-1}) \end{cases} \tag{4.4}$$

LSF への計算は次のようになる。

$$\begin{aligned} P(z) &= A_L(z) - B_L(x) \\ Q(z) &= A_L(z) - B_L(x) \end{aligned} \tag{4.5}$$

ここで、$P(z)$、$Q(z)$ はそれぞれ反対称な係数、対称な係数をもつ $(p+1)$ 次の多項式を表している。L を偶数と仮定すると次のように因数分解される。

$$\begin{aligned} P(z) &= (1 - z^{-1}) \prod_{i=2,4,\ldots,l} (1 - 2z^{-1}\cos\omega_i + z^{-2}) \\ Q(z) &= (1 + z^{-1}) \prod_{i=1,3,\ldots,l-1} (1 - 2z^{-1}\cos\omega_i + z^{-2}) \end{aligned} \tag{4.6}$$

ただし、ω_i は以下の関係を満たすように順序付けるものとする。

$$0 < \omega_1 < \omega_2 < \cdots < \omega_l - 1 < \omega_l \tag{4.7}$$

この係数 $\omega_1, \omega_2, \cdots, \omega_l$ を LSF と呼び、これらを用いて LSF 上での補間が可能となる。

4.2 制限と修正を加えた時間分解法 [5]

LSF に変換されたスペクトル情報は、さらに MRTD[5] を用いてイベント関数とイベントターゲットに分解される。MRTD は、より低ビットレート音声符号化に適応するため、TD[4] に制限と修正を加えた手法である。TD は以下のように、イベントベクトルの線形結合によってスペクトルパラメータの時間変化を近似する。

$$\hat{\boldsymbol{y}}(n) = \sum_{k=1}^{K} \boldsymbol{a}_k \phi_k(n), \quad 1 \le n \le N \tag{4.8}$$

ここで、$\boldsymbol{a}_k$、$\phi_k(n)$ は、それぞれ k 番目イベントターゲット、イベント関数である。$\hat{\boldsymbol{y}}(n)$ は、n 番目スペクトルパラメータ $\boldsymbol{y}(n)$ の近似値である。式 (4.9) を行列表示すると以下のようになる。

$$\hat{\boldsymbol{Y}} = \boldsymbol{A}\boldsymbol{\Phi} \quad \hat{\boldsymbol{Y}} \in R^{P\times N}, \quad \boldsymbol{A} \in R^{P\times K}, \quad \boldsymbol{\Phi} \in R^{K\times N}$$

ここで、P、N そして K は、それぞれスペクトルパラメータの次数、音声区間におけるフレーム数、イベントの数である。

発話中における各イベントは、時間と共に徐々に増加、減少していき、それらは隣同士重なり合う。よって、時間変化パターンを表すイベント関数には以下の特性が考えられる。

- 各イベント関数には始まりと終りの時間が存在する、すなわち時間間隔が存在する。
- 各イベント関数はその存在期間においては非負である。
- 各イベント関数は実際の発話における音声生成と同様、緩かな増加と減少で表せる。

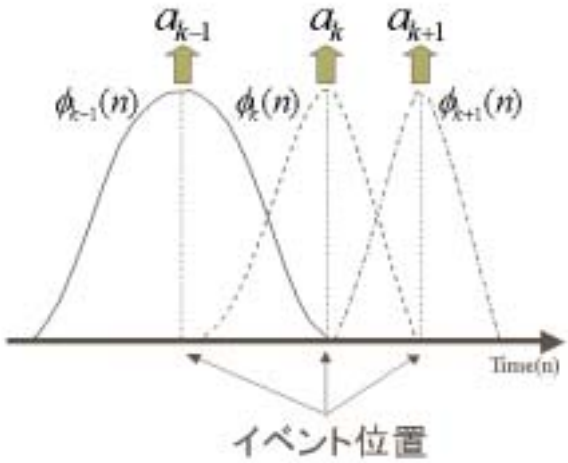

図 4.2: イベントターゲットとイベント関数

4.2.1 イベント関数

MRTD[6] では, イベント関数に 2 つの制約が加えられる。1) 時間のどの瞬間においても、隣接する 2 つのイベント関数だけで記述する。2) どの時刻においても隣接するイベント関数の合計は 1 である。この制約を用いれば式 (1) は次のようになる。$C(k) \leq n \leq C(k+1)$ に対して

$$\begin{aligned} \hat{\boldsymbol{y}}(n) &= \boldsymbol{a}_k\phi_k(n) + \boldsymbol{a}_{k+1}\phi_{k+1}(n) \\ &= \boldsymbol{a}_k\phi_k(n) + \boldsymbol{a}_{k+1}(1-\phi_k(n)) \end{aligned} \tag{4.9}$$

ここで、$C(k)$、$C(k+1)$ は、それぞれイベント k、$k+1$ の中心位置である。ただし、

$$\begin{gathered}\phi_k(C(k)) = 1,\ \phi_k(C(k+1)) = 0 \\ 0 \le \phi_k(n) \le 1 \ \text{ for } \ C(k) \le n \le C(k+1)\end{gathered} \tag{4.10}$$

最終的に $\phi_k(n)$ は、次のように決定される。

$$\phi_k(n) = \begin{cases} 1 - \phi_{k-1}(n), & \text{if } C(k-1) < n < C(k) \\ 1, & \text{if } n = C(k) \\ \min(\phi_k(n-1), & \\ \max(0, \hat{\phi}_k(n))), & \text{if } C(k) < n < C(k+1) \\ 0, & \text{その他} \end{cases} \tag{4.11}$$

ここで

$$\hat{\phi}_k = \frac{\langle (\boldsymbol{y}(n) - \boldsymbol{a}_{k+1}), (\boldsymbol{a}_k - \boldsymbol{a}_{k+1}) \rangle}{\|\boldsymbol{a}_k - \boldsymbol{a}_{k+1})\|^2} \tag{4.12}$$

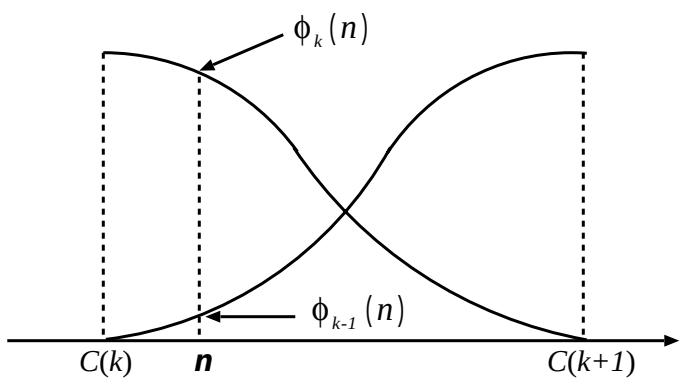

図 4.3: 2 つの隣接したイベント関数の例

4.2.2 イベントターゲット

イベントベクトルは、次の式を用いることによって推定される。

$$\boldsymbol{A} = \boldsymbol{Y}\boldsymbol{\Phi}^T(\boldsymbol{\Phi}\boldsymbol{\Phi}^T)^{-1} \tag{4.13}$$

しかし、推定されたイベントベクトルは、LSF の順序特性を乱す可能性がある。そこで、最小偏差 LSF を ϵ として以下の手順でイベントベクトルを決定する。

はじめに、次数 j $(1 \le j \le P)$ を変化させるより一般的な手順：$a_{i,k},\ a_{i+1,k},\ \cdots, a_{i+j-1,k}$ に対して、それぞれ $\hat{a}_{i,k},\ \hat{a}_{i+1,k} = \hat{a}_{i,k} + \epsilon,\ \cdots,\ \hat{a}_{i+j-1,k} = \hat{a}_{i,k} + (j-1)\epsilon$ を定める。それ

を考慮することによって、変化により引き起こされる誤差 E の増分は、

$$\Delta = \sum_{l=0}^{j-1} [\ a_{i+l,k} - (\hat{a}_{i,k} + l\epsilon)\ ]^2 \sum_n \phi_k(n)^2 \tag{4.14}$$

となる。Δ を最小にするために、$\hat{a}_{i,k} \geq a_{i-1,k} + \epsilon,\ \hat{a}_{i,k}$ は次のように決定される。

$$\hat{a}_{i,k} = \begin{cases} a_{i-1,k} + \epsilon, & \text{if } \tilde{a}_{i,k} < a_{i-1,k} + \epsilon \\ \tilde{a}_{i,k}, & \text{otherwise} \end{cases} \tag{4.15}$$

ここで、

$$\tilde{a}_{i,k} = \frac{\sum_{l=0}^{j-1} a_{i+l,k}}{j} - \frac{(j-1)\epsilon}{2} \tag{4.16}$$

$a_{1,k} > 0$ と $a_{P,k} < \pi$ を想定するため、$a_k = [0, a_1, \cdots, a_{P,k}, \pi]^T$ というように 0 と π を a_k に加える。簡単にするために 0 と π は、それぞれ $a_{o,k}$、$a_{P+1,k}$ とおく。ただし、$a_{o,k}$ と $a_{P+1,k}$ は、正規化する間変化させることはできない。全アルゴリズムを次のように示す。

- ステップ 1：
  初期化 $i \leftarrow 0$

- ステップ 2：
  もし $i < P$ そして $a_{i,k} + \epsilon \leq a_{i+1,k}$ ならば、$i \leftarrow i+1$。このステップを $i = P$ または $a_{i,k} + \epsilon > a_{i+1,k}$ まで繰り返す。もし $i = P$ ならば、ステップ 6 へ。

- ステップ 3：
  もし $i = 0$ ならば、$a_{0,k}$ は変化できないので $i \leftarrow 1$ 、$j \leftarrow 1$。そうでないならば、$j \leftarrow 2$。

- ステップ 4：
  式 (4.16) を使って、$a_{i,k}, \cdots, a_{i+j-1,k}$ を $\hat{a}_{i,k}, \cdots, \hat{a}_{i+j-1,k}$ に変化させる。もし $i+j-1 = P$ ならば、ステップ 6 へ。

- ステップ 5：
  もし $a_{i+j-1,k} + \epsilon > a_{i+j,k}$ ならば、前のステップから a_k を復元し、$j \leftarrow P - i + 1$。そして、ステップ 4 に戻る。そうでないならば、$i \leftarrow i + j$、もし $i < P$ ならばステップ 2 へ。

- ステップ 6：
  もし $a_{P,k} + \epsilon \leq a_{P+1,k}$ ならば、a_k は標準化された。；そうでなければ i と前のステップからそれに一致するベクトル a_k の値を元に戻し、$j \leftarrow P - i + 1$。そして、ステップ 4 へ。

ステップ 6 で、もし i が修正された区分の最終パラメータであるならば、i はその区分の先頭に定められる。特に、$i=0$ のとき、ベクトル a_k は $[0, \pi - P\epsilon, \pi - (P-1)\epsilon, \cdots, \pi]^T$ として定められる。

結局、イベント $C(k)$、$k = 1, \cdots, K$ の中心位置がわかり、対応するイベントベクトルが LSF ベクトル軌道 $\boldsymbol{y}(C(k))$ のサンプルで初期化される場合に、式 (4.12)、(4.14)、(4.16) を用いて繰り返し適当なイベント関数とイベントベクトルを計算することができる。ここで、最大スペクトル安定基準に基づく初期のイベント中心位置を決定する方法が、MRTD 法に採用される。以下にその方法を示す。

Spectral Feature Transition Rate による初期イベント位置の決定

区間 $[n-M, n+M]$ 中におけるスペクトルパラメータ $y_i(n)$ の回帰直線の勾配 $C_i(n)$ と Spectral Feature Transition Rate (SFTR) $s(n)$ を考える。

$$c_r = \frac{\sum_{m=-M}^{M} m y_r(n+m)}{\sum_{m=-M}^{M} m^2}, \quad 1 \le r \le P \tag{4.17}$$

$$s(n) = \sum_{r=1}^{P} c_r(n)^2, \quad 1 \le n \le N \tag{4.18}$$

$s(n)$ の局所的最小値はフレーム内の局所スペクトル安定点を表すことになる。これらの点はイベントの位置を、またそのスペクトルパラメータによってできるベクトルはイベントターゲットを表すことになる。よって、$s(n)$ の局所最小値をそれぞれ $n_1, n_2, \cdots, n_K (n_1 < n_2 < \cdots < n_K)$ をすると、最初のイベントターゲット (イベントの中心位置) の行列 $\boldsymbol{A}^{(0)}$ は以下のようになる。ただし、イベント数は M によって調整が可能である。

$$\begin{aligned} \boldsymbol{A}^{(0)} &= [\boldsymbol{a}_1^{(0)} \boldsymbol{a}_2^{(0)} \cdots \boldsymbol{a}_K^{(0)}] \\ &= [\boldsymbol{y}(n_1) \boldsymbol{y}(n_2) \cdots \boldsymbol{y}(n_K)] \end{aligned} \tag{4.19}$$

4.3 LSF の次数決定

4.3.1 次数に対するスペクトル歪みの変動

LSF の次数を決定するために、符号化システムにおけるスペクトル歪み（SD）を調べた。ここでは、STRAIGHT によって抽出されたスペクトルパラメータと LSF および MRTD を用いて分析合成したスペクトルパラメータの間のスペクトル歪みを評価した。図 4.2 は、スペクトル情報の補間に LSF のみを用いた場合、LSF と MRTD を用いた場合のスペクトル歪み評価を示す。

n 番目のフレームに対するスペクトル歪み D_n は、次のように決定される。

$$D_n^2 = \frac{1}{F_s} \int_0^{F_s} \left[20 \log_{10}(S_n(f)) - 20 \log_{10}(\hat{S}_n(f)) \right]^2 df \tag{4.20}$$

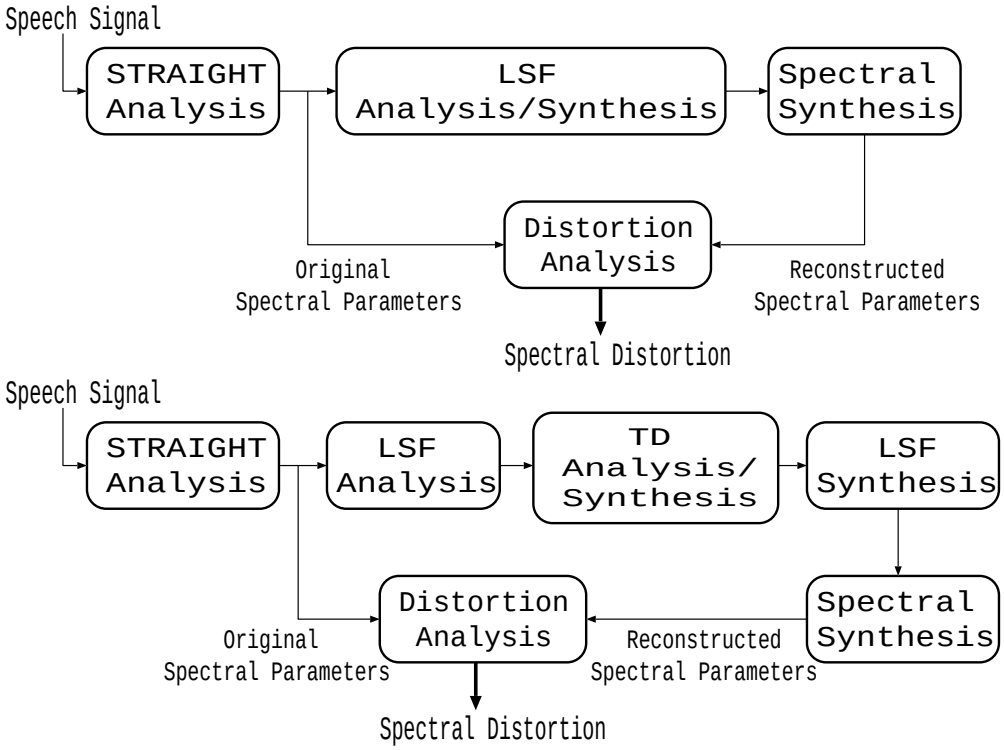

図 4.4: スペクトル歪み評価 (上：LSF のみを用いた場合、LSF と MRTD を用いた場合)

ここで、F_s はサンプリング周波数であり、$S_n(f)$ と $\hat{S}_n(f)$ はそれぞれ n 番目のフレームに対する STRAIGHT によって得られた振幅スペクトル、再構成した振幅スペクトルである。スペクトル歪みは、発話データにおける全てのフレームに対して評価され、その平均値が計算される。

テストデータとして、ATR 日本語音声データベースの話者 MMY による音韻バランス 503 文章中の 112 文を 8 kHz にダウンサンプリングしたものを用いた。スペクトル情報の補間方法に LSF のみを適用した場合、LSF および MRTD を適用した場合の結果を図 4.3 に示す。ただし、量子化は行っていない。横軸は LSF の次数を表し、縦軸は対数スペクトル歪みを表す。図 4.3 において、次数を 22 以上にした場合に LSF によるスペクトル歪みは減少していくが、LSF および MRTD 後のスペクトル歪みはあまり改善されない。これは、次数を増やすにつれて MRTD によるスペクトル歪みが増加しているためだと考えられる。よって、LSF の次数を 22 次以上にしてもスペクトル歪みの著しい改善は期待できない。

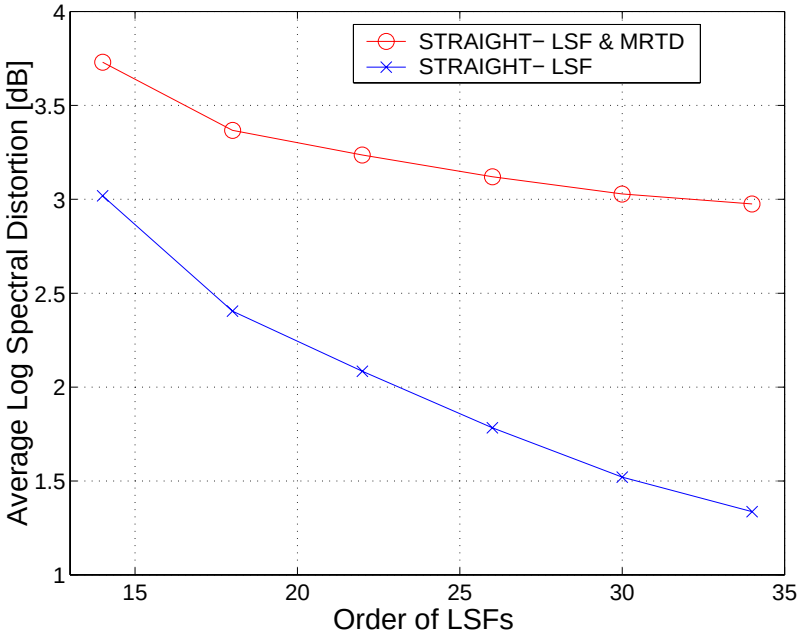

図 4.5: LSF の次数に対するスペクトル歪みの変動

4.3.2 次数に対する音声品質の変動

前節では SD によって、LSF の次数を変化させた場合におけるスペクトル歪みの変動を客観的に評価した。ここでは、同様の場合における合成音の品質を、聴取実験により主観的に評価した。

刺激音

聴取実験には、ATR 音声データベースの話者 MMY による音韻バランス 503 文章中の 2 文章を用いた。音声データを以下に示す。

- 発話文章　「そちらの国際会議に論文を投稿したいのですが。」
  「原稿の締め切りはいつですか。」
- ファイル

  `mmysc102.ad` - ATR 日本語音声データベース
  `mmysc211.ad` - ATR 日本語音声データベース

この 2 文章に対して、LSF の次数を 10、14、18、22、26、30 と変化させたものに MRTD を適用して分析合成を行った。したがって、各文章に対して 6 つの刺激音ができる。ただし、分析合成を行う際に、ベクトル量子化およびスカラー量子化は行っていない。

被験者

正常聴力を有すると認められる大学院生 6 名とした。

実験方法

実験はシェッフェの一対比較法により行った。約 2 秒の間隔で異なる刺激音を一対として呈示し、どちらの音が（前者・後者）が歪みが小さいかを 5 段階で判断させた。評価表を図 4.6 に示す。被験者は防音室内でヘッドホンにより受聴した。受聴は各被験者の聞きやすいレベルによる両耳聴取である。

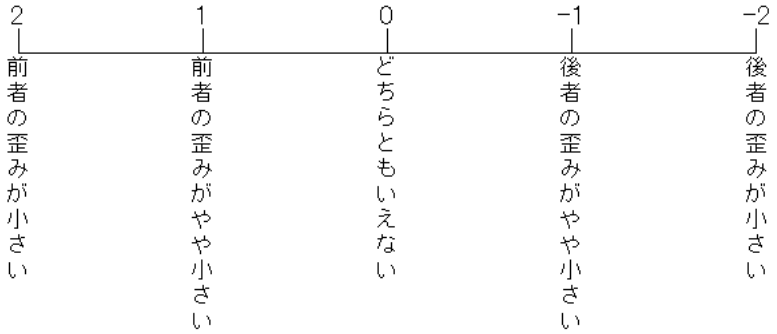

図 4.6: 評価表

実験結果

実験結果を図 4.7 に示す。横軸は母数を表し、その位置は提示した刺激音の相対的な距離を表す。プラス側（右側）にいくほど歪みが小さく、マイナス側（左側）にいくほど歪みが大きいと判断される。矢印の上の数字は、LSF の次数を表す。実験より、LSF の次数が 10、14 では明らかに歪みが大きいことが示された。また、次数を 22 以上にしても聴覚的に歪みの改善は感じられないことが示された。以後、LSF の次数は 22 次とする。

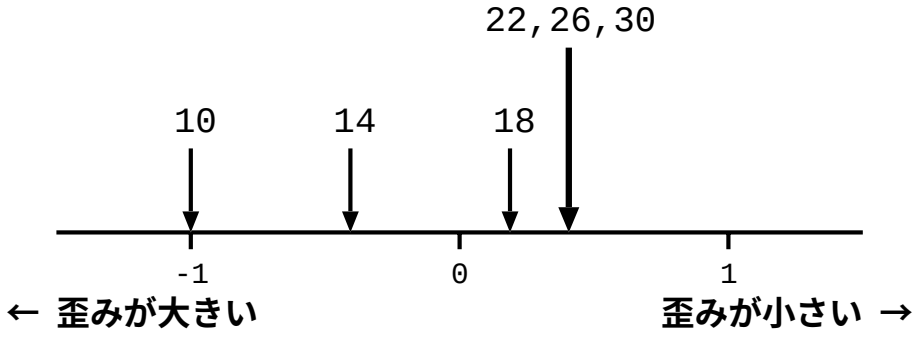

図 4.7: LSF の次数に対する音声品質の変動

4.4 ベクトル量子化

イベントターゲットとイベント関数は、ベクトルの時系列として表現される。各ベクトルはあらかじめ求められている典型的なベクトル集合（コードブック）の一つの要素に置換される。もし、コードブックサイズが $N = 2^k$ であれば、各ベクトルは k ビットで表現されることになる。

今回、ベクトル量子化アルゴリズムとして LBG アルゴリズム [8][9] と 2 分割繰り返しアルゴリズム [9] を用い、符合帳探索・学習の距離尺度にはユークリッド距離を用いた。以下にそのアルゴリズムを示す。

LBG アルゴリズム

1. 初期化
   訓練サンプル集合 : $\{\boldsymbol{x}_\alpha,\ \alpha = 1, 2, \cdots, n\}$
   コードブックサイズ : m
   ステップ数 : $l = 0$
   コードブックの初期集合: $\boldsymbol{A}_0^{(m)} = \{\boldsymbol{y}_1^{(0)}\boldsymbol{y}_2^{(0)} \cdots \boldsymbol{y}_m^{(0)}\}$
   平均ひずみ : $D_{-1} = \infty$
   ひずみ閾値 : ϵ
2. $\{\boldsymbol{x}_\alpha,\ \alpha = 1, 2, \cdots, n\}$ を $\boldsymbol{A}_0^{(m)}$ によって m 個のクラスタ $\{\boldsymbol{C}_i,\ i = 1, 2, \cdots, m\}$ に分割する。すなわち、
   $$\mathrm{dis}(\boldsymbol{x}_\alpha, \boldsymbol{y}_i^{(l)}) < \mathrm{dis}(\boldsymbol{x}_\alpha, \boldsymbol{y}_t^{(l)}),\quad i \neq,\ t = 1, 2, \cdots, m$$
   ならば
   $$\boldsymbol{x}_\alpha \in \boldsymbol{C}_i$$
3. 平均ひずみ D_l を計算
   $$D_l = \frac{1}{n}\sum_{i=1}^{m}\sum_{\alpha=1}^{n}\{\mathrm{dis}(\boldsymbol{x}_\alpha, \boldsymbol{y}_i^{(l)}) \mid \boldsymbol{y}_\alpha \in \boldsymbol{C}_i\}$$
   もし $(D_{l-1} - D_l)/D_l < \epsilon$ ならば終了、$\boldsymbol{A}_l^{(m)}$ をコードベクトルとする。それ以外なら 4 へ。
4. $\boldsymbol{A}_{l+1}^{(m)} = \{\boldsymbol{y}_1^{(l+1)}\boldsymbol{y}_2^{(l+1)} \cdots \boldsymbol{y}_m^{(l+1)}\}$ を求める。ここで、$\boldsymbol{y}_i^{(l+1)}$ はクラスター $\boldsymbol{C}_i$ のセントロイドを再計算をして用いる。$l \leftarrow l + 1$ として 2 へ。

2 分割繰り返しアルゴリズム

1. 初期化
   Δ : 大きさの小さい適当なベクトル
   $\mathbf{A}_0^{(l)}$: 全サンプル $\{\boldsymbol{x}_\alpha,\ \alpha = 1, 2, \cdots, n\}$ のセントロイド
   $s + 1$

2. $\boldsymbol{A}_0^{(s)} = \{\boldsymbol{y}_1^{(0)}\boldsymbol{y}_2^{(0)}\cdots\boldsymbol{y}_s^{(0)}\}$ について各 $\mathbf{y}_k,\ k = 1, 2, \cdots, s$ を 2 つのベクトル

$$\mathbf{y}_k + \Delta,\ \boldsymbol{y}_k - \Delta$$

に分ける。

$$\boldsymbol{A}_0^{(2s)} = \{\boldsymbol{y}_1 + \Delta \quad \boldsymbol{y}_1 - \Delta \quad \cdots \quad \boldsymbol{y}_s + \Delta \quad \boldsymbol{y}_s - \Delta\}$$

3. $\mathbf{A}_0^{(2s)}$ を初期値として LBG アルゴリズムにより $2s$ における最適解

$$\boldsymbol{A}_0^{(2s)} = \{\boldsymbol{y}_k,\ k = 1, 2, \cdots, 2s\}$$

を求める。 $2s = m$ ならば終了。

4. $l = 0, 2 \leftarrow 2s$ として 2 へ。

4.4.1 イベント関数の量子化

イベント関数の時間長は各イベントごとに異なるため、その時間長を 16 次のベクトルに正規化してベクトル量子化を行った。

4.4.2 イベントターゲットの量子化

イベントターゲットは、分割ベクトル量子化を行う。分割ベクトル量子化は、あるベクトルをいくつかに分割し、それぞれの分割したベクトルに対してベクトル量子化を行う方法である。イベントターゲットは、低次の LSF 値の分散が大きいことから 22 次を 6、7、9 次に 3 分割した。

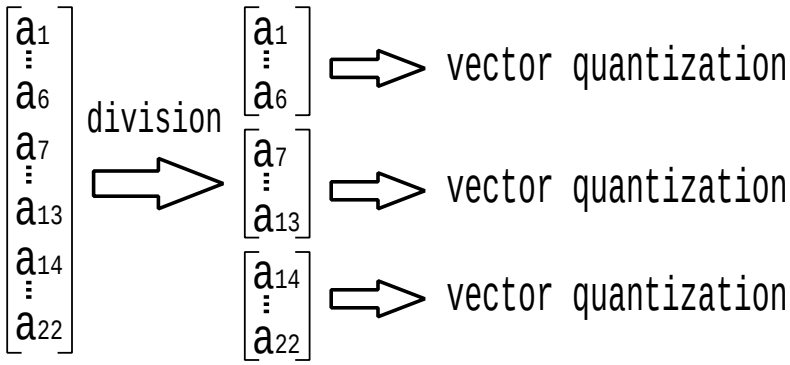

図 4.8: 分割ベクトル量子化

第5章　基本周波数の符号化

この章では、基本周波数情報の符号化方法として、符号化方法1（TDを用いた方法)、符号化方法2（線形補間を用いた方法)、符号化方法3（有声無声区間の時間長を正確に量子化する方法）を検証する。その結果から、3つの方法において最も合成誤差の小さいものを符号化に適用した。

5.1　符号化方法1

基本周波数パラメータは、基本周波数ターゲットとスペクトル用のイベント関数を用いて次のように再構成できる [6]。

$$\hat{p}(n) = \sum_{k=1}^{K} p_k \phi_k(n), \quad 1 \leq n \leq N \tag{5.1}$$

ここで、$\hat{p}(n)$ と p_k は、それぞれ n 番目のフレームに対して再構成した基本周波数パラメータと基本周波数ターゲットである。式 (5.1) は行列表記すると次のように書かれる。

$$\hat{P} = A_p \mathbf{\Phi} \tag{5.2}$$

ここで、$\hat{P}$ と A_p はそれぞれ再構成した基本周波数ベクトルと基本周波数ターゲットベクトルである。さらに

$$\hat{P} \;=\; [\ \hat{p}(n)\ ]_{1 \leq n \leq N} \tag{5.3}$$

そして

$$\begin{aligned} A_p &= [\ p_1\ p_2\ \cdots\ p_K\ ] \\ &= [\ p_k\ ]_{1 \leq k \leq K} \end{aligned} \tag{5.4}$$

基本周波数ターゲット p_k、$1 \leq k \leq K$ は次のように、元の基本周波数パラメータと再構成した基本周波数パラメータの二乗誤差を最小にするように決定される。

$$E_p = \sum_{n=1}^{N} \Big(p(n) - \sum_{k=1}^{K} p_k \phi_k(n) \Big)^2 \tag{5.5}$$

p_r に関する E_p の偏導関数を 0 とおくと

$$\begin{aligned}\frac{\partial E_p}{\partial p_r} &= \sum_{n=1}^{N}\Big(p(n)-\sum_{k=1}^{K}p_k\phi_k(n)\Big)(-2\phi_r(n)) \\ &= 0, \quad 1 \le r \le K \end{aligned} \tag{5.6}$$

$$\sum_{k=1}^{K}p_k\sum_{n=1}^{N}\phi_k(n)\phi_r(n)=\sum_{n=1}^{N}p(n)\phi_r(n), \quad 1 \le r \le K \tag{5.7}$$

式（5.1）によって再構成される基本周波数は、無声区間を正確に再現していない。そこで、以下に説明する方法で有声無声情報を抽出・再構成する。この有声無声情報を用いて、再構成される基本周波数における無声区間を再構成する。ただし、有声無声情報も量子化することになり、符号化全体のビットレートが増加する。

有声無声ターゲットの決定

有声無声パラメータは、有声無声ターゲットとスペクトル用のイベント関数を用いて次のように再構成できる。

$$v_{int}(n)=\sum_{k=1}^{K}v_k\phi_k(n), \quad 1 \le n \le N \tag{5.8}$$

$$\hat{v}(n)=\begin{cases} 1, & v_{int}(n) \ge 0.5 \\ 0, & \text{otherwise} \end{cases}$$

ここで、$\hat{v}(n)$ と v_k は、それぞれ n 番目のフレームに対して再構成した有声無声パラメータと有声無声ターゲットである。式 (5.8) は行列表記すると次のように書かれる。

$$V_{int}=A_v\mathbf{\Phi} \tag{5.9}$$

ここで、$\hat{V}$ と A_v はそれぞれ再構成した有声無声ベクトルと有声無声ターゲットベクトルである。さらに

$$\hat{V} = [\ \hat{v}(n)\ ]_{1\le n\le N} \tag{5.10}$$

そして

$$\begin{aligned} A_v &= [\ v_1\ v_2\ \cdots\ v_K\ ] \\ &= [\ v_k\ ]_{1\le k\le K} \end{aligned} \tag{5.11}$$

有声無声ターゲット v_k、$1 \leq k \leq K$ は次のように、元の有声無声パラメータと再構成した有声無声パラメータの二乗誤差を最小にするように決定される。

$$E_v = \sum_{n=1}^{N} \Big(v(n) - \sum_{k=1}^{K} v_k \phi_k(n) \Big)^2 \tag{5.12}$$

v_r に関する E_v の偏導関数を 0 とおくと

$$\begin{aligned} \frac{\partial E_v}{\partial v_r} &= \sum_{n=1}^{N} \Big(v(n) - \sum_{k=1}^{K} v_k \phi_k(n) \Big) (-2\phi_r(n)) \\ &= 0, \quad 1 \leq r \leq K \end{aligned} \tag{5.13}$$

$$\sum_{k=1}^{K} v_k \sum_{n=1}^{N} \phi_k(n)\phi_r(n) = \sum_{n=1}^{N} v(n)\phi_r(n), \quad 1 \leq r \leq K \tag{5.14}$$

最後に、有声無声ターゲット v_k、$1 \leq k \leq K$ は、次のように決定される。

$$v_k = \begin{cases} 1, & \hat{v}_{int}(n) \geq 0.5 \\ 0, & \text{otherwise} \end{cases}$$

5.1.1 シミュレーション結果

ここでは、5.1 節で示した方法の合成精度を検証する。イベント数を 15 events/s に設定し、基本周波数ターゲットを 5 bit、有声無声ターゲットを 1 bit で対数スカラー量子化した後、基本周波数を再構成した。ビットレートは 90 bit/s である。図 5.1 は、STRAIGHT によって抽出した基本周波数、符号化方法 1 によって再構成した基本周波数、合成誤差を表している。テストデータを以下に示す。

- 発話文章　「原稿の締め切りはいつですか。」
- ファイル　mmysc211.ad - ATR 日本語音声データベース

ただし、データは 8 kHz にダウンサンプリングしたものを用いた。RMS 誤差は 37.0 Hz であった。

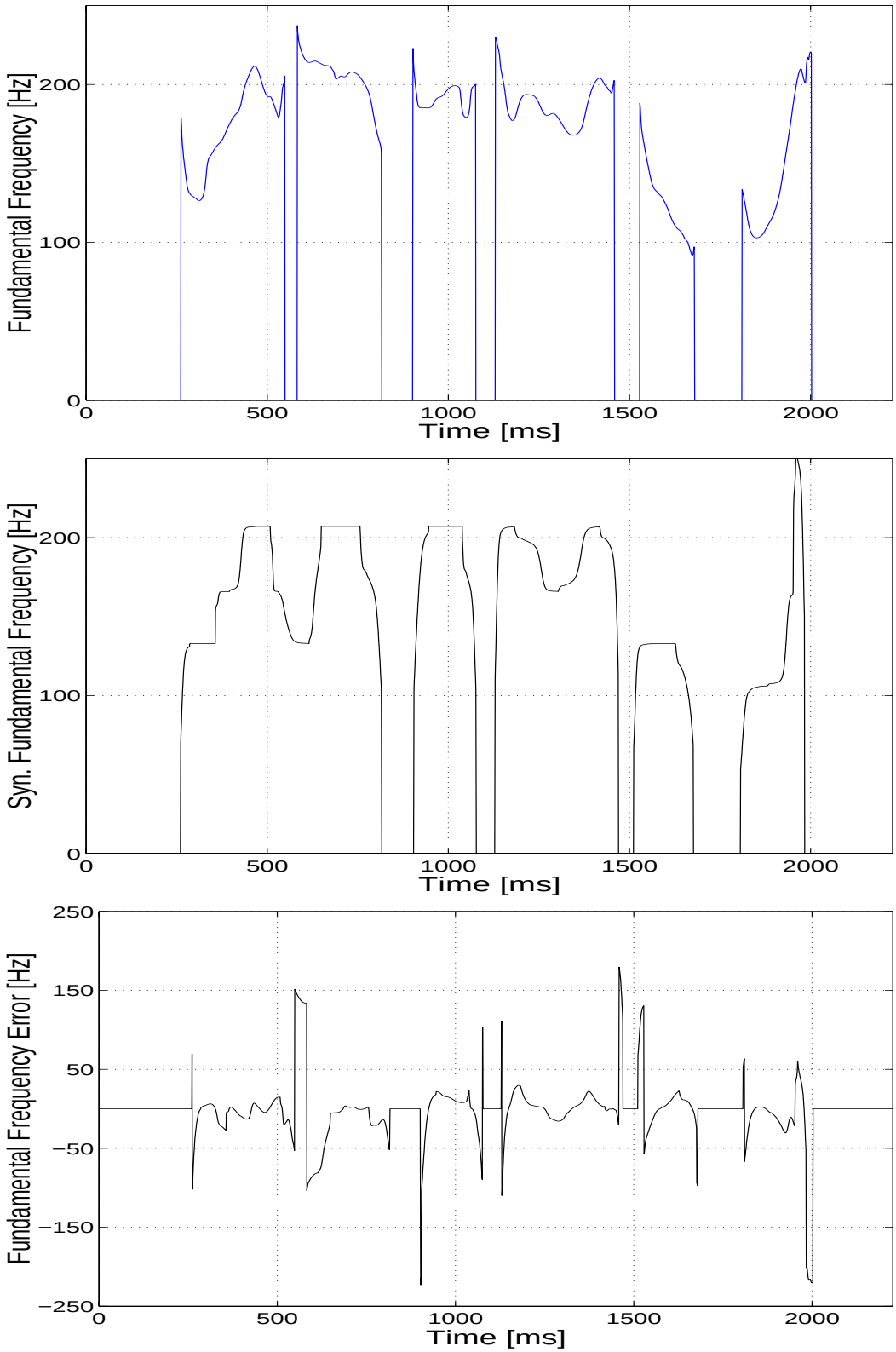

図 5.1: 基本周波数（上：抽出した基本周波数、中：再構成した基本周波数、下：合成誤差）「原稿の締め切りはいつですか。」

5.2 符号化方法2

抽出した基本周波数情報に対してダウンサンプリングを行い、対数スカラー量子化を行った。量子化されたパラメータは、合成側で線形補間を用いて元のサイズに復元した。基本周波数パラメータの量子化および再構成の流れを図 5.2 に示す。

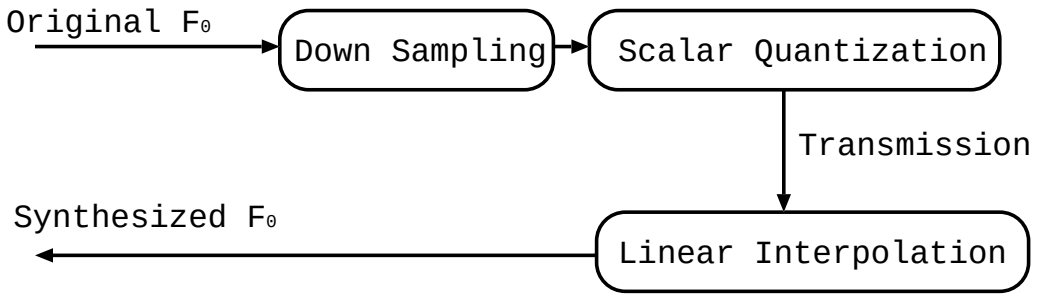

図 5.2: 基本周波数の量子化および再構成の流れ

5.2.1 シミュレーション結果

ここでは、5.2 節で示した方法の合成精度を検証する。基本周波数パラメータは、ダウンサンプリングの間隔を 28 ms とし、6 bit で対数スカラー量子化を行った。ビットレートは約 216 bit/s である。図 5.3 は、STRAIGHT によって抽出した基本周波数、符号化方法 2 によって再構成した基本周波数、合成誤差を表している。テストデータを以下に示す。

- 発話文章　「原稿の締め切りはいつですか。」
- ファイル　mmysc211.ad - ATR 日本語音声データベース

ただし、データは 8kHz にダウンサンプリングしたものを用いた。RMS 誤差は約 28.8 Hz であった。

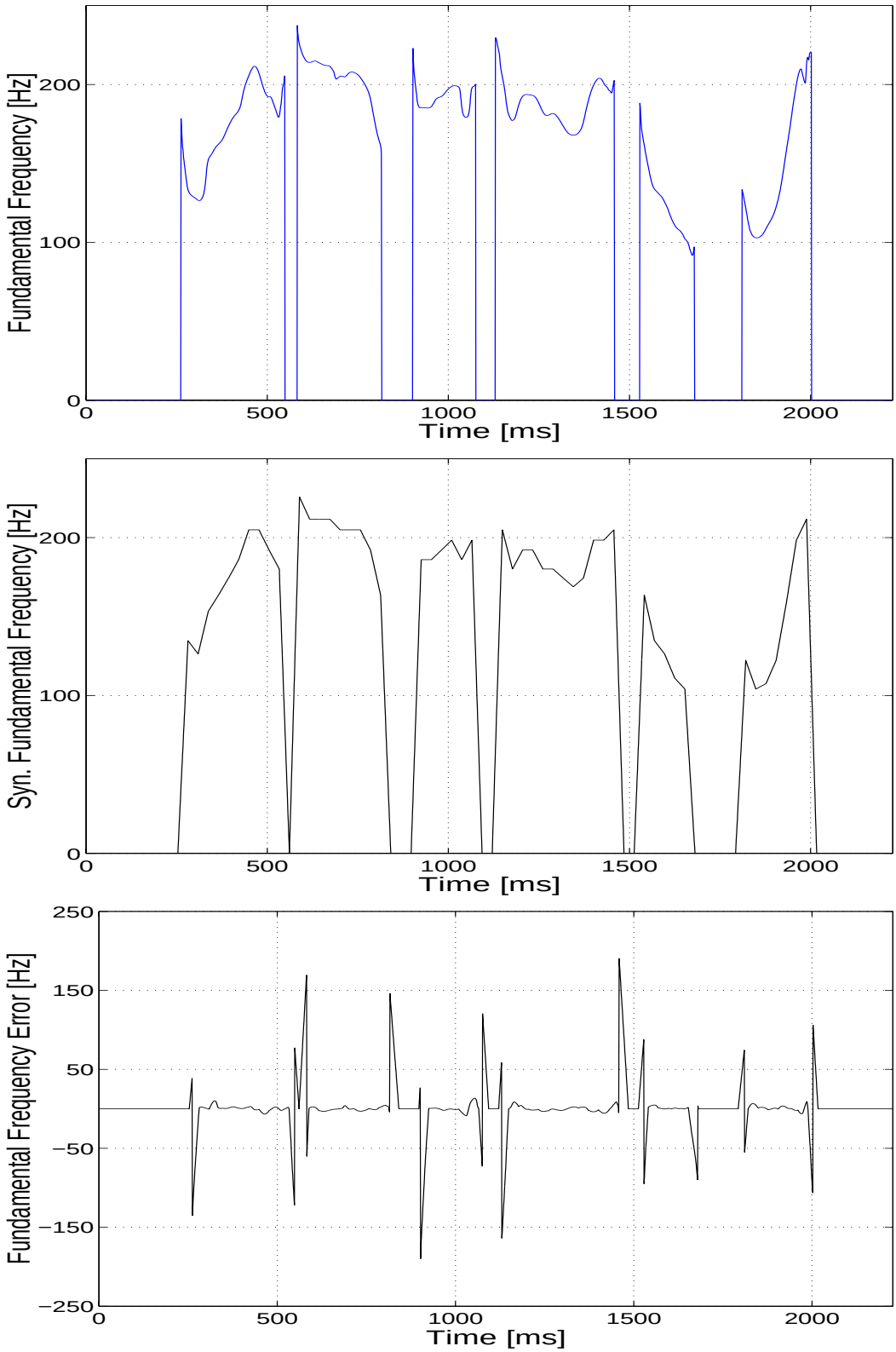

図 5.3: 基本周波数（上：抽出した基本周波数、中：再構成した基本周波数、下：合成誤差）「原稿の締め切りはいつですか。」

5.3 符号化方法3

図5.4に示すように、STRAIGHTにより求めた基本周波数から、無声区間と有声区間の時間長をスカラー量子化した。これにより、有声区間の立ち上がり位置と立ち下がり位置の正確な情報を伝送する。有声区間に対しては、ダウンサンプリングを行い、対数スカラー量子化を行った。この時、立ち上がり位置と立ち下がり位置のデータも含める。

合成側において、これらの情報をもとに無声区間では0として補間を行い、有声区間では線形補間を用いて元の長さに復元し、基本周波数を再構成した。

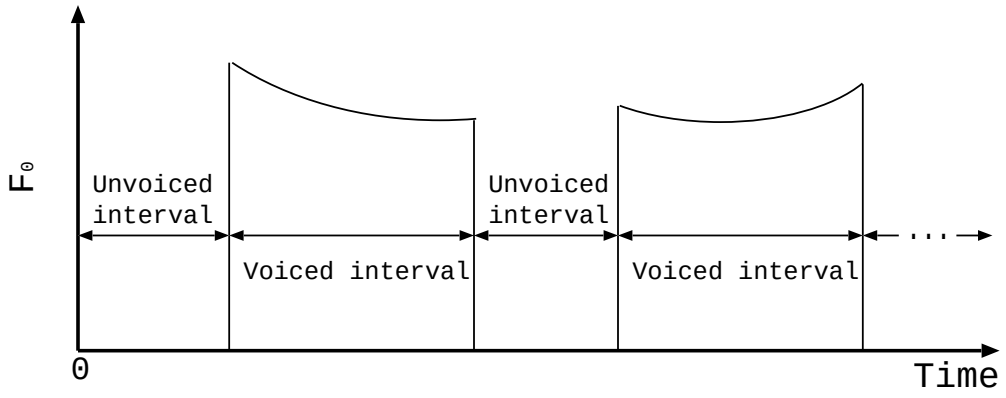

図 5.4: 基本周波数の量子化

5.3.1 シミュレーション結果

ここで、5.3節で示した方法の合成精度を検証する。有声無声区間の時間長を10ビットで量子化し、有声区間は28msの間隔でダウンサンプリングして、5ビットで量子化した。ビットレートは約215 bit/sである。図5.5は、STRAIGHTによって抽出した基本周波数、符号化方法3によって再構成した基本周波数、合成誤差を表している。テストデータを以下に示す。

- 発話文章　「原稿の締め切りはいつですか。」
- ファイル　mmysc211.ad - ATR日本語音声データベース

ただし、データは8 kHzにダウンサンプリングしたものを用いた。RMS誤差は約3.7 Hzであった。

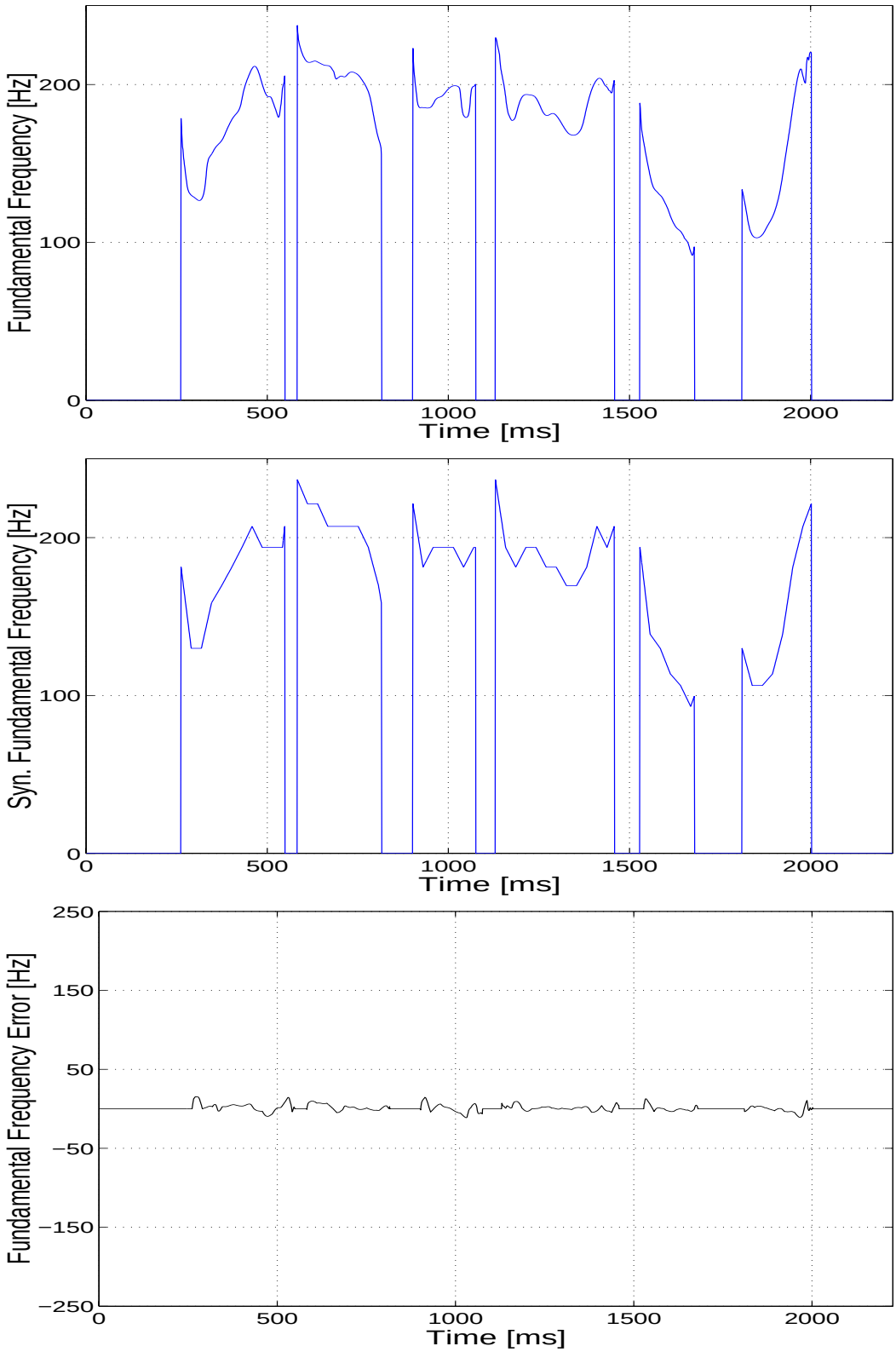

図 5.5: 基本周波数（上：抽出した基本周波数、中：再構成した基本周波数、下：合成誤差）「原稿の締め切りはいつですか。」

5.4 考察

シミュレーションの結果から、符号化方法 3 が最も精度良く基本周波数情報を再構成できることがわかった。

TD を用いた符号化方法 1 は、ビットレートを低く抑えることはできるが、合成誤差が大きくなった。この理由の一つに、有声無声情報が曖昧になっていることが挙げられる。さらには、再構成した音声にも音色の変化などの影響がみられた。線形補間を用いた符号化方法 2 は、符号化方法 1 よりもビットレートが 2 倍以上になるにもかかわらず、合成誤差はあまり変らなかった。符号化方式 2 と同程度のビットレートを持つ符号化方法 3 は、情報圧縮の面では符号化方法 1 にかなわない。しかし、合成精度は他の 2 つに比べて著しく改善されており、再構成した音声も改善がみられた。

以上のことから、基本周波数情報の符号化には、符号化方法 3 を用いる。

第6章　ゲインの符号化

この章では、ゲインの符号化方法として、符号化方法1（TDを用いた方法)、符号化方法2（線形補間を用いた方法)、符号化方法3（スプライン補間を用いた方法）を検証する。その結果から、3つの方法において最も合成誤差の小さいものを符号化に適用した。

6.1　符号化方法1

ゲインパラメータは、ゲインターゲットとスペクトル用のイベント関数を用いて次のように再構成できる [6]。

$$\hat{g}(n) = \sum_{k=1}^{K} g_k \phi_k(n), \quad 1 \leq n \leq N \tag{6.1}$$

ここで、$\hat{g}(n)$ と g_k は、それぞれ n 番目のフレームに対して再構成したゲインパラメータとゲインターゲットである。式 (6.1) は行列表記すると次のように書かれる。

$$\hat{G} = A_g \mathbf{\Phi} \tag{6.2}$$

ここで、$\hat{G}$ と A_g はそれぞれ再構成したゲインベクトルとゲインターゲットベクトルである。さらに

$$\hat{G} \;=\; [\ \hat{g}(n)\ ]_{1 \leq n \leq N} \tag{6.3}$$

そして

$$\begin{aligned} A_g &= [\ g_1\ g_2\ \cdots\ g_K\ ] \\ &= [\ g_k\ ]_{1 \leq k \leq K} \end{aligned} \tag{6.4}$$

ゲインターゲット g_k、$1 \leq k \leq K$ は次のように、元のゲインパラメータと再構成したゲインパラメータの二乗誤差を最小にするように決定される。

$$E_g = \sum_{n=1}^{N} \Big(g(n) - \sum_{k=1}^{K} g_k \phi_k(n) \Big)^2 \tag{6.5}$$

g_r に関する E_g の偏導関数を 0 とおくと

$$\begin{aligned}\frac{\partial E_g}{\partial g_r} &= \sum_{n=1}^{N}\Big(g(n)-\sum_{k=1}^{K} g_k\phi_k(n)\Big)(-2\phi_r(n)) \\ &= 0, \quad 1 \le r \le K \end{aligned} \tag{6.6}$$

$$\sum_{k=1}^{K} g_k \sum_{n=1}^{N} \phi_k(n)\phi_r(n) = \sum_{n=1}^{N} g(n)\phi_r(n), \quad 1 \le r \le K \tag{6.7}$$

6.1.1 シミュレーション結果

6.1 節で示した方法の合成精度を検証する。ベント数を 15 events/s に設定し、ゲインターゲットを 6 bit で対数スカラー量子化した後、ゲインを再構成した。ビットレートは 90 bit/s である。図 6.1 は、STRAIGHT によって抽出したゲイン、符号化方法 1 によって再構成したゲイン、合成誤差を表している。テストデータを以下に示す。

- 発話文章　「原稿の締め切りはいつですか。」
- ファイル　mmysc211.ad - ATR 日本語音声データベース

ただし、データは 8 kHz にダウンサンプリングしたものを用いた。RMS 誤差は 8.5 dB であった。

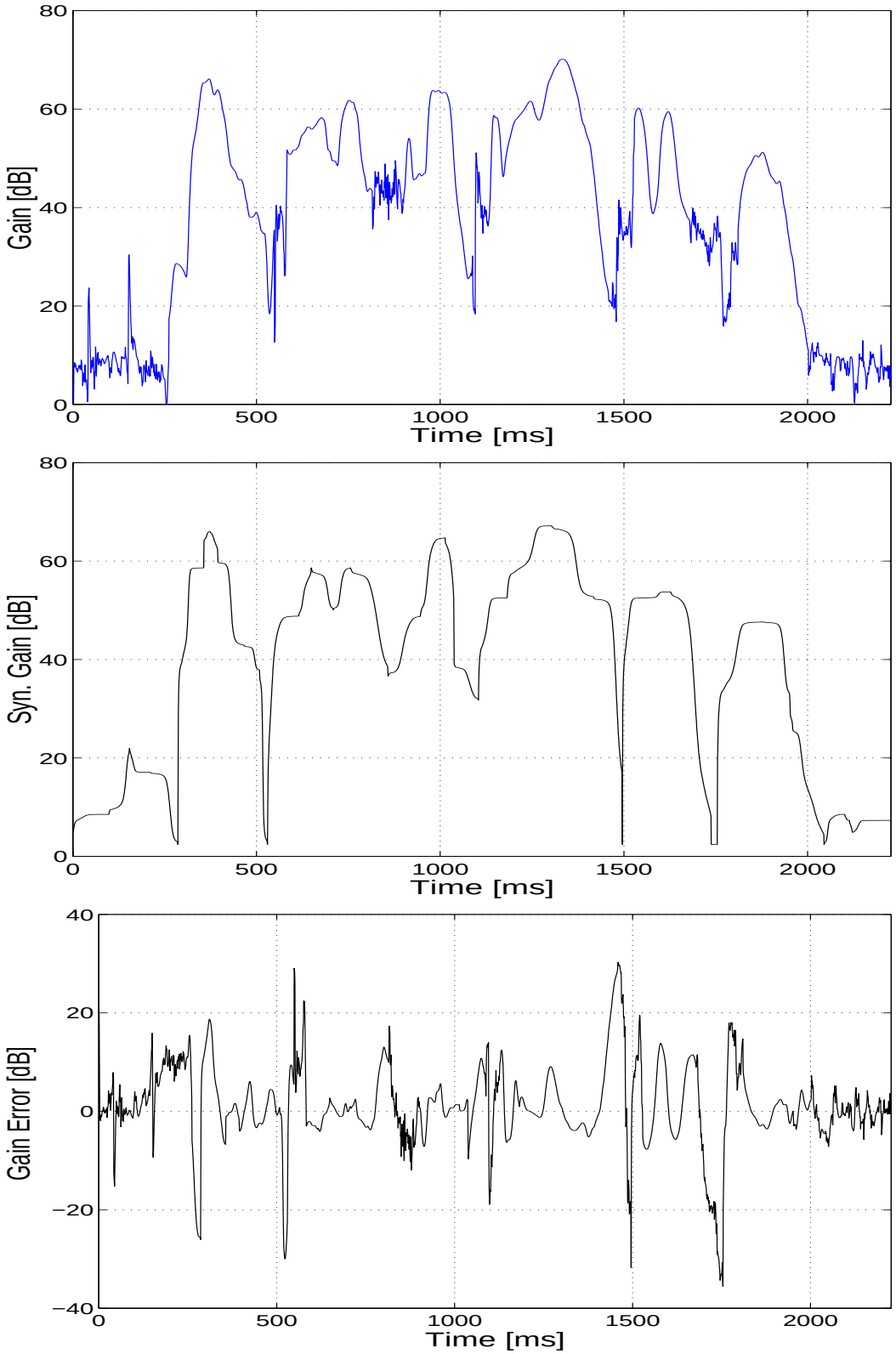

図 6.1: ゲイン（上：抽出したゲイン、中：再構成したゲイン、下：合成誤差）「原稿の締め切りはいつですか。」

6.2　符号化方法2

ゲインパラメータに対してダウンサンプリングを行い、対数スカラー量子化を行う。量子化されたパラメータは、合成側で線形補間を用いて元のサイズに復元した。ゲインパラメータの量子化および再構成の流れを図6.2に示す。

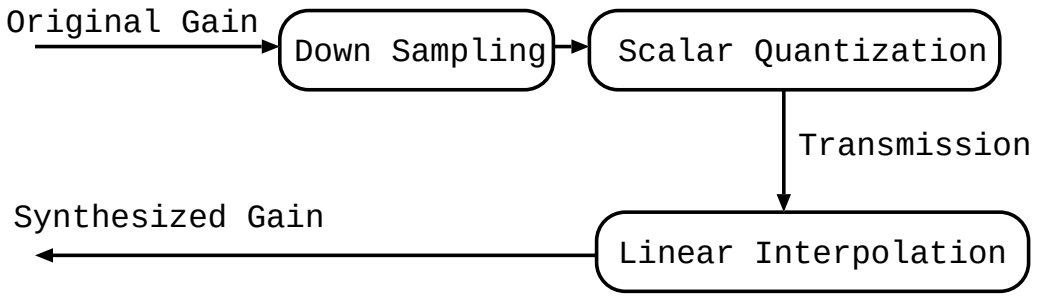

図 6.2: ゲインの量子化および線形補間による再構成の流れ

6.2.1　シミュレーション結果

ここでは、6.2節で示した方法の合成精度を検証する。ゲインパラメータは、ダウンサンプリングの間隔を20 msとし、6 bitで対数スカラー量子化を行った。ビットレートは300 bit/sである。図6.3は、抽出したゲイン、再構成したゲイン、合成誤差を表している。テストデータを以下に示す。

- 発話文章　「原稿の締め切りはいつですか。」
- ファイル　mmysc211.ad - ATR日本語音声データベース

ただし、データは8kHzにダウンサンプリングしたものを用いた。RMS誤差は約4.3 dBであった。

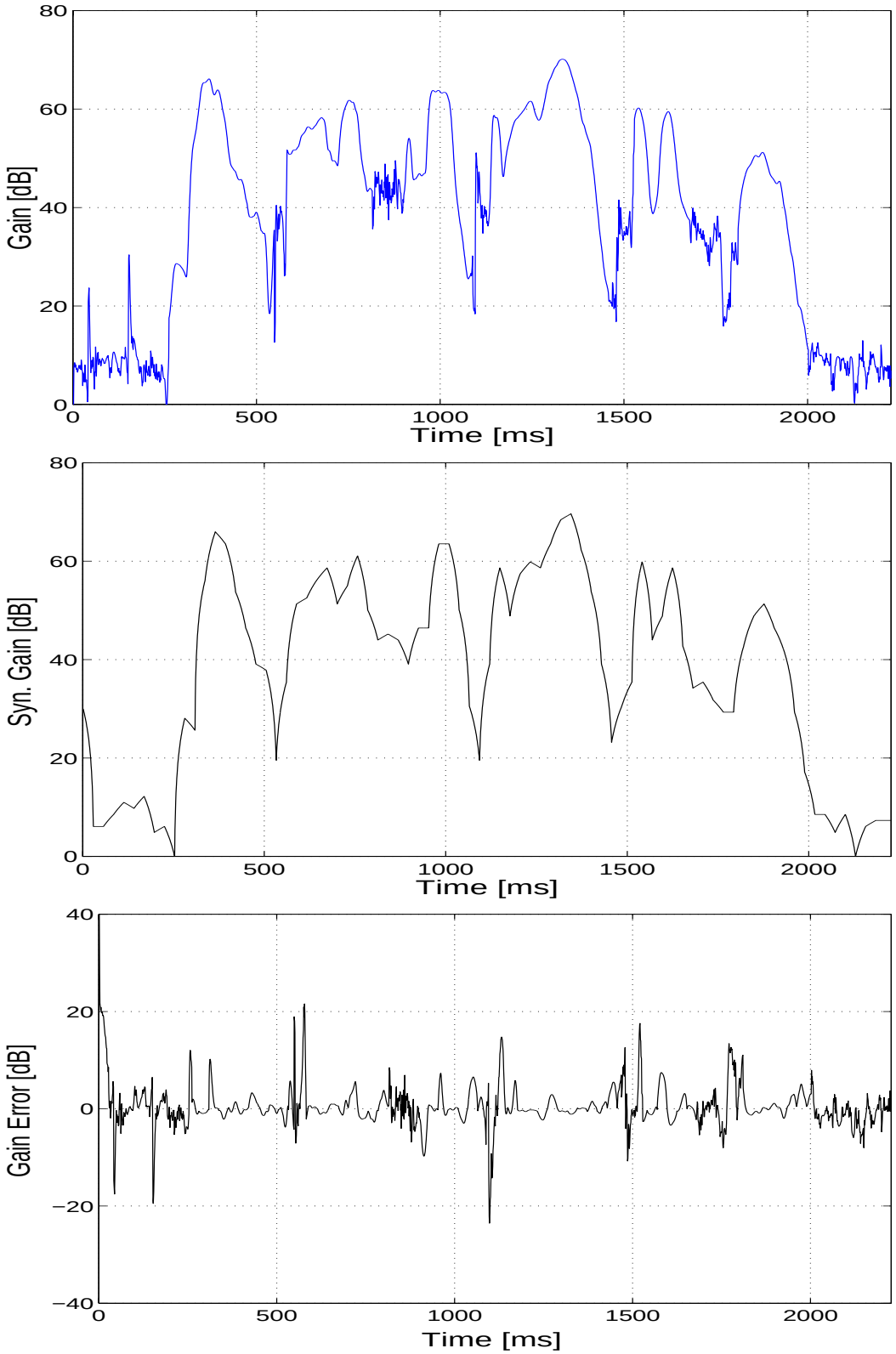

図 6.3: ゲイン（上：抽出したゲイン、中：再構成したゲイン、下：合成誤差）「原稿の締め切りはいつですか。」

6.3 符号化方法3

ゲインパラメータに対してダウンサンプリングを行い、対数スカラー量子化を行う。量子化されたパラメータは、合成側でスプライン補間を用いて元のサイズに復元した。ゲインパラメータの量子化および再構成の流れを図 6.4 に示す。

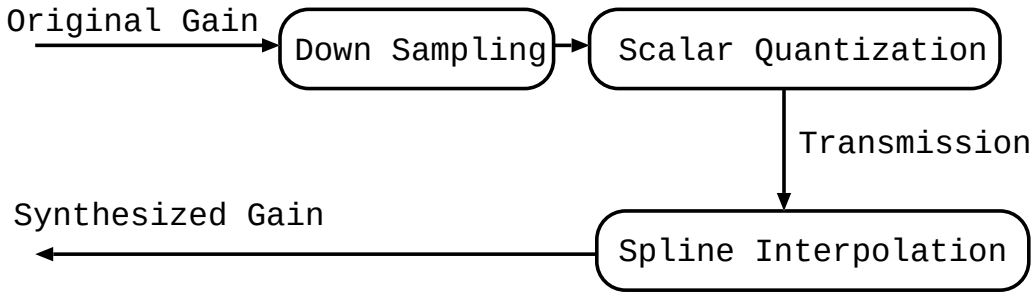

図 6.4: ゲインの量子化およびスプライン補間による再構成の流れ

6.3.1 シミュレーション結果

ここでは、6.3 節で示した方法の合成精度を検証する。ゲインパラメータは、ダウンサンプリングの間隔を 20 ms とし、6 bit で対数スカラー量子化を行った。ビットレートは 300 bit/s である。図 6.5 は、抽出したゲイン、再構成したゲイン、合成誤差を表している。テストデータを以下に示す。

- 発話文章　「原稿の締め切りはいつですか。」
- ファイル　mmysc211.ad - ATR 日本語音声データベース

ただし、データは 8kHz にダウンサンプリングしたものを用いた。RMS 誤差は約 3.5 dB であった。

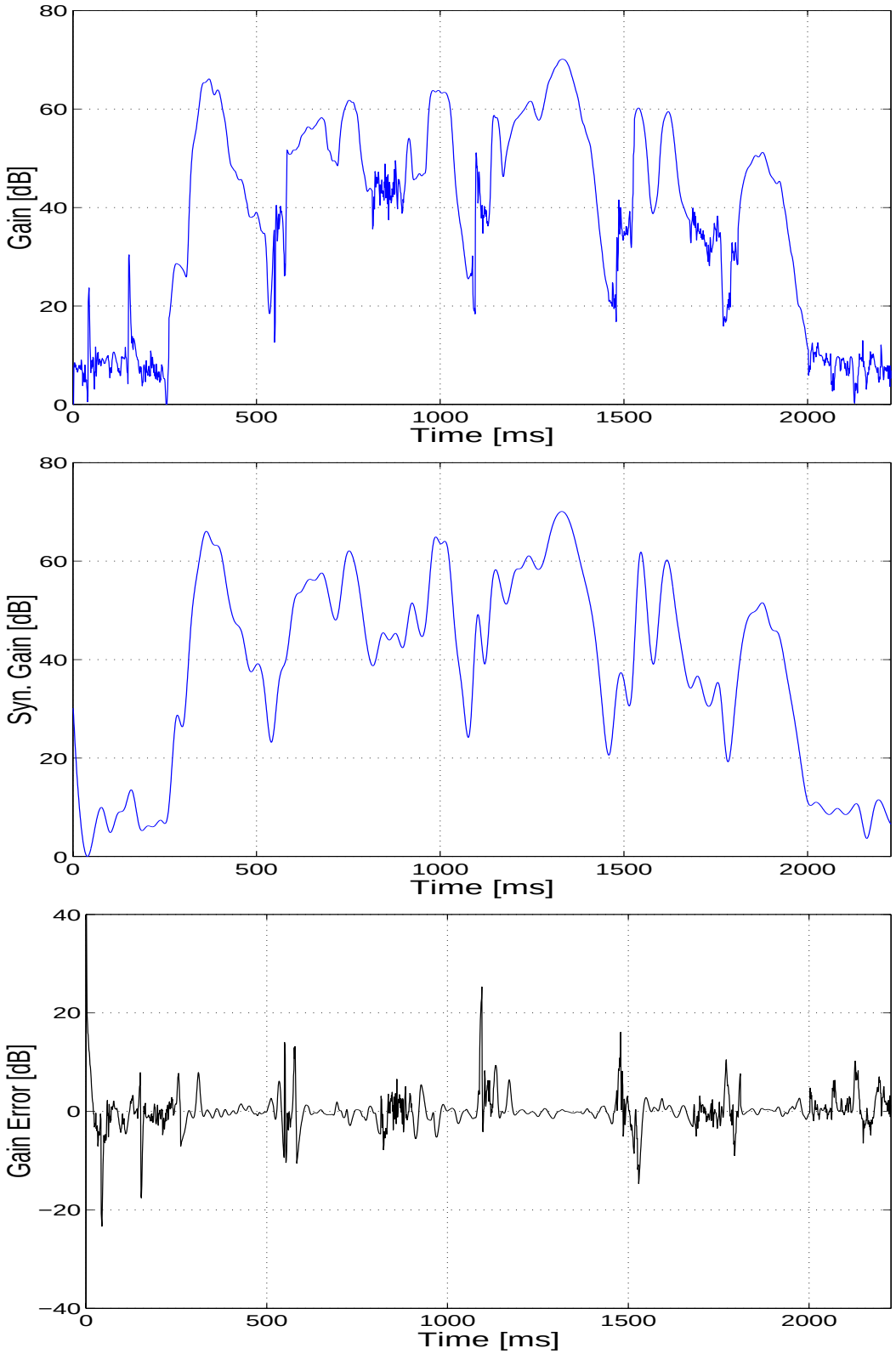

図 6.5: ゲイン（上：抽出したゲイン、中：再構成したゲイン、下：合成誤差）「原稿の締め切りはいつですか。」

6.4 考察

シミュレーションの結果から、符号化方法3が最も精度良くゲインを再構成できることがわかった。

TDを用いた符号化方法1は、ビットレートを低く抑えることはできるが、合成誤差が大きくなった。さらには、再構成した音声にも影響がみられた。これは、本手法で求めるゲインがスペクトルを再構成する時に用いられるため、再構成したスペクトルの構造に大きな影響を与えるためである。線形補間を用いた符号化方法2は、符号化方法1よりもビットレートが2倍以上になるが、合成誤差は約1/2に改善された。符号化方式2と同程度のビットレートを持つ符号化方法3は、さらに符号化方法2よりも合成誤差が改善された。これは、スプライン補間を用いた方が線形補間を用いるよりも、ゲインのなだらかな変化を上手く表現できるためだと考えられる。

以上のことから、ゲインの符号化には、符号化方法3を用いる。

第7章　雑音比の符号化

7.1　符号化方法

雑音比パラメータは、雑音比ターゲットとスペクトル用のイベント関数を用いて次のように再構成できる。

$$\hat{i}(n) = \sum_{k=1}^{K} i_k \phi_k(n), \quad 1 \leq n \leq N \tag{7.1}$$

ここで、$\hat{i}(n)$ と i_k は、それぞれ n 番目のフレームに対して再構成した雑音比パラメータと雑音比ターゲットである。式 (7.1) は行列表記すると次のように書かれる。

$$\hat{I} = A_i \mathbf{\Phi} \tag{7.2}$$

ここで、$\hat{I}$ と A_i はそれぞれ再構成した雑音比ベクトルと雑音比ターゲットベクトルである。さらに

$$\hat{I} \;=\; [\ \hat{i}(n)\ ]_{1 \leq n \leq N} \tag{7.3}$$

そして

$$\begin{aligned} A_i &= [\ i_1\ i_2\ \cdots\ i_K\ ] \\ &= [\ i_k\ ]_{1 \leq k \leq K} \end{aligned} \tag{7.4}$$

雑音比ターゲット i_k、$1 \leq k \leq K$ は次のように、元の雑音比パラメータと再構成した雑音比パラメータの二乗誤差を最小にするように決定される。

$$E_i = \sum_{n=1}^{N} \Big(i(n) - \sum_{k=1}^{K} i_k \phi_k(n) \Big)^2 \tag{7.5}$$

i_r に関する E_i の偏導関数を 0 とおくと

$$\begin{aligned} \frac{\partial E_i}{\partial i_r} &= \sum_{n=1}^{N} \Big(i(n) - \sum_{k=1}^{K} i_k \phi_k(n) \Big)(-2\phi_r(n)) \\ &= 0, \quad 1 \leq r \leq K \end{aligned} \tag{7.6}$$

$$\sum_{k=1}^{K} i_k \sum_{n=1}^{N} \phi_k(n)\phi_r(n) = \sum_{n=1}^{N} i(n)\phi_r(n), \quad 1 \leq r \leq K \tag{7.7}$$

再構成した雑音比パラメータは、全体を通して比較的なだらかな変動で表現される。しかし、STRAIGHT によって実際に抽出された雑音比パラメータは、有声区間での値がどうであれ、無声区間では 1 になる。この無音区間での誤差をなくすため、再構成した基本周波数情報から有声無声情報を抽出し、その有声無声情報を用いて、再構成した雑音比パラメータの無音区間を 1 に置き換える操作を加えた。

7.1.1 シミュレーション結果

ここでは、7.1 節で示した方法の合成精度を検証する。イベント数を 15 events/s に設定し、雑音比ターゲットを 5 bit で対数スカラー量子化した後、雑音比パラメータを再構成した。ビットレートは 75 bit/s である。図 7.1 は、抽出した雑音比、再構成した雑音比、合成誤差を表している。テストデータを以下に示す。

- 発話文章　「原稿の締め切りはいつですか。」
- ファイル　mmysc211.ad - ATR 日本語音声データベース

ただし、データは 8 kHz にダウンサンプリングしたものを用いた。RMS 誤差は 0.1 であった。

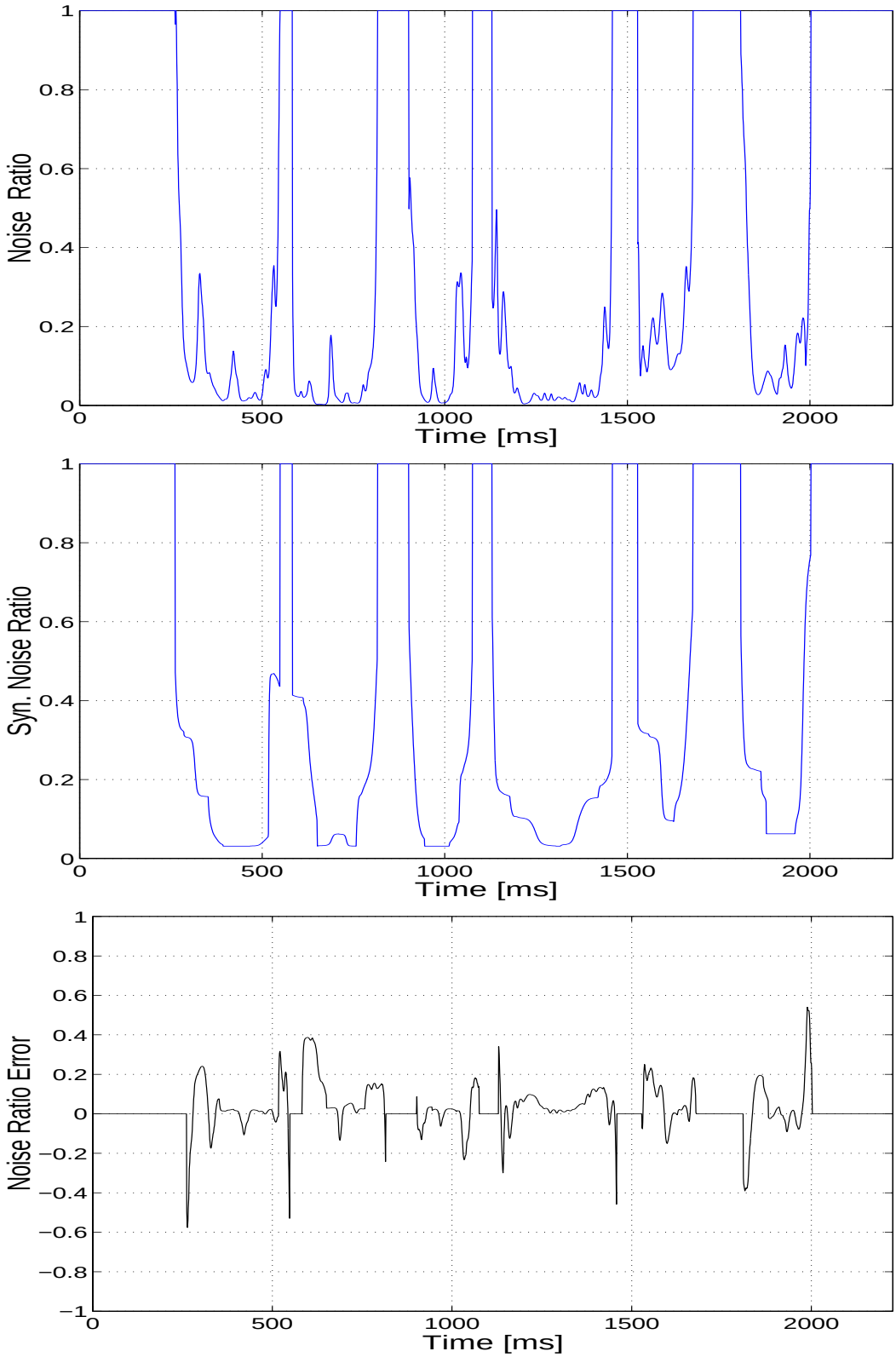

図 7.1: 雑音比（上：抽出した雑音比、中：再構成した雑音比、下：合成誤差）「原稿の締め切りはいつですか。」

第8章　提案法のビット割り当て

これまでに示した手法を用いて、ビットレートの異なる2つの低ビットレート音声符号化システムを構築した。各パラメータのビット割り当てを表1に示す。ただし、イベント数は約15 events/sになるように設定した。提案法1は、スペクトルに関するイベントターゲットに27 bitを割り当てたものである。提案法2は、スペクトルに関するイベントターゲットに24 bitを割り当てたものである。イベントターゲットに対するビット割り当ての括弧内の値は、分割したイベントターゲットにそれぞれ割り当てたビット数である。基本周波数およびゲインのビット割り当ての詳細を以下に示す。

- 基本周波数
  有声無声区間の時間長に10ビットを割り当てた。有声区間においては、ダウンサンプリングの間隔を28 msとし、量子化の符合帳に5ビットを割り当てた。ただし、ビットレートは、1秒間に時間長の量子化が7回行われ、有声区間が約800 ms存在するものとして計算している。有声区間では約800[ms]/28[ms]個のデータを量子化することになるので、ビットレートは次の通りになる。

$$7 \times 10\ [\mathrm{bit}] + 29 \times 5\ [\mathrm{bit}] = 215\ [\mathrm{bit/s}]$$

- ゲイン
  ダウンサンプリングの間隔を20 msとし、量子化の符合帳に6ビットを割り当てた。1秒間では50個のデータをサンプリングすることになるので、ビットレートは次の通りになる。

$$50 \times 6\ [\mathrm{bit}] = 300\ [\mathrm{bit/s}]$$

表 8.1: ビット割り当て

パラメータ	提案法 1	提案法 2
イベントターゲット	24(8+8+8)	27(9+9+9)
イベント関数	7	7
イベント間の距離	8	8
雑音比	5	5
小計 A (合計×イベント数)	660 [bit/s]	705 [bit/s]
基本周波数	215	215
ゲイン	300	300
入力音声の最大値	9	9
小計 B	524 [bit/s]	524 [bit/s]
総計 (A+B)	1194 [bit/s]	1229 [bit/s]

第9章　品質評価

9.1　目的

提案法の品質を評価するために、約 1.2 kbit/s に設定した提案法による合成音と、他の低ビットレート音声符号化方式による合成音との品質比較実験を行う。

9.2　方法

実験はシェッフェの一対比較法により行った。約 2 秒間ごとに異なる刺激音を一対として呈示し、どちらの音（前者・後者）の歪みが小さいかを 5 段階で判断させた。評価表を図 9.1 に示す。被験者は防音室内でヘッドホンにより受聴した。受聴は各被験者の聞きやすいレベルによる両耳聴取である。

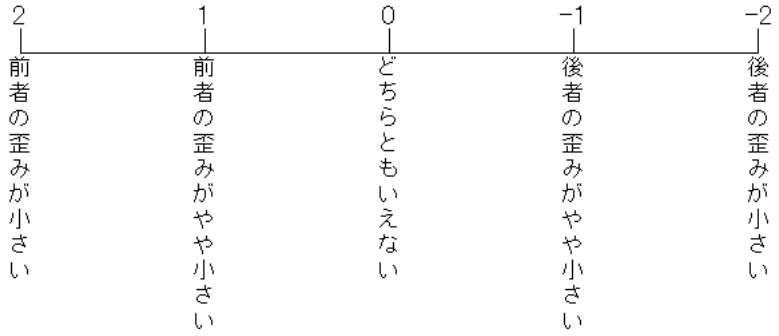

図 9.1: 評価表

符合帳の学習データ

学習データは、ATR 日本語音声データベースにおける音韻バランス 503 文章中の連続発話された約 108 文章を用いた。ただし、データは 8kHz にダウンサンプリングしたものを用いている。話者は男女各 3 名（MHT、MMY、MSH、FKN、FTK、FYM）である。

刺激音

音声データは、学習外男女各 1 名の発話音声 2 文章（学習外）を用いた。音声データを

以下に示す。

- 発話文章　「原稿の締め切りはいつですか。」
  「会議に発表するのではなくて聴講するだけだと、いくらかかりますか。」

- ファイル

  `mtksc211.ad` - ATR 日本語音声データベース
  `mtksc305.ad` - ATR 日本語音声データベース
  `fkssc211.ad` - ATR 日本語音声データベース
  `fkssc305.ad` - ATR 日本語音声データベース

この各データに対して、STRAIGHT に LSF を適用したもの（量子化なし）、STRAIGHT に LSF と MRTD を適用したもの（量子化なし）、4.8 kbit/s CELP、2.4 kbit/s LPC10、1.19 kbit/s に設定した提案法、1.23 kbit/s に設定した提案法の 6 つの方法によって刺激音を作成した。したがって、各データに対して 6 つの刺激音ができる。

被験者
正常聴力を有すると認められる大学院生 8 名とした。

9.3　実験結果と考察

実験結果を図 9.2 に示す。横軸は母数を表す。プラス側（右側）にいくほど歪みが小さく、マイナス側（左側）にいくほど歪みが大きいと判断される。矢印の上の表記は、それぞれ音声符号化方式を表す。ただし、STRAIGHT-LSF は STRAIGHT に LSF を適用したもの、STRAIGHT-LSF, MRTD は STRAIGHT に LSF と MRTD を適用したものを表す。

実験の結果、低ビットレートの音声符号化方式の中では、2.4 kbit/s LPC10 が最も歪みが大きく、4.8 kbit/s CELP が最も歪みが小さいという結果となった。2.4 kbit/s LPC10 は、明らかに歪みが大きく、さらに音色の変化も感じられるために最も悪い結果になったと考えられる。4.8 kbit/s CELP は、歪みおよび音色の変化が感じられるものの、音声の明瞭度がよいことから最も良い結果になったと考えられる。提案法は、2.4 kbit/s LPC10 と比べて、ビットレートが低いにもかかわらず明らかに歪みが小さいと判断された。また、2.4 kbit/s LPC10 よりも高いビットレートをもつ 4.8 kbit/s CELP には劣るものの、その差はわずかであった。さらに音色という点では、他の 2 つの低ビットレート音声符号化方式を用いたものよりも原音に近く、音声によっては評価が 4.8 kbit/s CELP に優るものもあった。しかし、被験者に意見を聞いたところ、「合成音にポコやピコなどといった音が発生した。」、「背景雑音が大きい場合がある。」、「子音の部分での歪みが大きい。」といった意見が出た。これは、MRTD を用いたときのスペクトルパラメータの時間分解に

おいて、分解による歪みや母音部分と子音部分との分解能が完全ではないことよる音質劣化の影響が考えられる。

量子化を行っていない方法、STRAIGHT-LSF と STRAIGHT-LSF, MRTD を比べると、かなりの差があることがわかる。これは上記の影響が考えられる。だだし、分解による歪みについては、MRTD の分析幅を変化させることによりイベント数を調節することで、ある程度改善することができる。改善の結果によっては、提案法が 4.8 kbit/s CELP を上回る可能性がある。

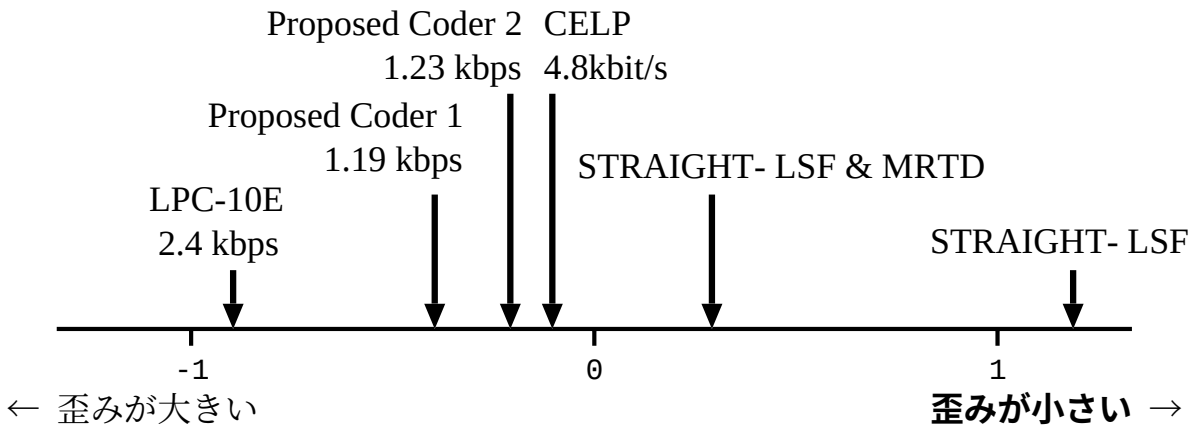

図 9.2: 品質評価実験の結果

第10章　結論

10.1　まとめ

本研究では、低ビットレート音声符号化における合成音の品質を向上させるために、音声分析・変換・合成方式として高音質な合成音を作成することができる STRAIGHT を用いた。STRAIGHT において、伝送側に送られる情報はスペクトル情報と基本周波数情報である。スペクトル情報は、MRTD を用いて音韻情報に対応したパラメータに分解し、ベクトル量子化を行うことによって情報圧縮を行った。基本周波数情報に対しては、スカラー量子化を行った。最終的に約 1.2 kbit/s の音声符号化システムを構築し、聴取実験により品質評価を行った。

品質評価実験を行った結果、4.8 kbit/s CELP の品質には及ばないものの、2 kbit/s 以下のビットレートで 2.4 kbit/s LPC10 よりも明らかに良い品質を持っていることがわかった。よって、提案法は、低ビットレート音声符号化において高品質ではないが、2 kbit/s 以下のビットレートでも十分な品質の合成音を作成できる可能性があると言える。

10.2　今後の課題

本研究の課題として次のことがあげられる。

- MRTD による影響
  MRTD を用いて音声を再構成した場合に、不自然な音や背景雑音が合成音に混じることがあった。また、合成音において子音部分の歪みが感じられた。このようなことは、スペクトル情報を LSF に変換した後に、量子化せずに音声の再構成を行った場合には発生しなかった。原因として、MRTD を用いたときのスペクトルパラメータの時間分解において、分解による歪みや母音部分と子音部分との分解能が完全ではないことよる音質劣化やベクトル量子化による量子化誤差などの影響が考えられる。低ビットレート音声符号化において、より品質の良い合成音を作成するためには、MRTD の改善が必要である。

- イベント数について
  本研究では、より低ビットレートを実現するため、イベント数を約 15 events/s に設定している。しかし、このイベント数では、MRTD を用いた時に合成音を上手く表

現できない可能性がある。よって、イベント数を増やした場合、つまりビットレート増加させた場合における提案法の検討が必要である。

- STRAIGHT の分析フレーム長について
  本手法では、STRAIGHT における分析フレームシフトを 1ms としている。分析フレームが 1ms では、かなり多くの情報を伝送することになり、情報圧縮という面では不利である。文献 [12] によれば、5ms まで分析フレームシフトをのばしても、STRAIGHT による合成音の劣化は生じないことが示されている。よって、分析フレームシフトの長さについても検討の余地がある。

謝辞

本研究を遂行するにあたり、数多くの貴重な御助言、御指導を頂きました北陸先端科学技術大学院大学 情報科学研究科 赤木 正人 教授、党 建武 助教授、並びに本学の教官の皆様に深く感謝致します。本研究を進めるにあたって、貴重な助言をして頂きましたNTTサイバースペース研究所 守谷 健弘 氏に深く感謝致します。日頃から大いなる議論と激励を頂きました北陸先端科学技術大学院大学 情報科学研究科 赤木研究室の皆様に深く感謝致します。最後に日頃からあたたかく見守り、支えてくれた妻に心より感謝します。

関連図書

[1] 河原英紀, “聴覚の情景分析と高品質音声分析変換合成法 STRAIGHT,” 音響学会講演論文集, 1-2-1, pp.186-192, 1997-9.

[2] 河原英紀, Alain de Cheveigne, “原理的に抽出誤りの存在しないピッチ抽出法とその評価について,” 電子情報通信学会, SP96-96, pp.9-18, 1997.

[3] 河原英紀, 勝瀬郁代, “音声分析・変換・合成法 STRAIGHT-TEMPO における相補的な時間窓の利用について,” 電子情報通信学会, SP97-32, pp.21-28, 1997.

[4] B.S.Atal, “Efficient coding of LPC parameters by temporal decomposition,” Proc. ICASSP ’83, pp.81-84, 1983.

[5] P.C.Nguyen and M.Akagi, “Improvement of the restricted temporal decomposition method for LSF parameters,” Proc. 2001 Autumn Meeting of ASJ, pp.267-268, 2001.

[6] A.C.R.Nandasena, P.C.Nguyen and M.Akagi, “Spectral stability based event localizing temporal decomposition,” Computer Speech and Language, Vol.15, No.4, pp.-, 2001.

[7] S.J.Kim, S.H.Lee, W.J.Han and Y.H.Oh, “Efficient quantization of LSF parameters based on temporal decomposition,” Proc. ICLP98, Vol.6, pp.2575-2578, 1998.

[8] Y.Linde, A.Buzo, R.M.Gray, “An algorithm for vector quantiser design,” IEEE Trans. on Communication, Vol.28, pp.84-95, 1980.

[9] 電子情報通信学会, “確率モデルによる音声認識,” pp.27-28, コロナ社, 1988.

[10] “Draft Recommendation G.729 : Coding of Speech at 8 kbit/s using Conjugate-Structure Algebraic-Code-Excited Linear-Predictive (CS-ACELP) Coding,” ITU-T Study Group 15 Contribution Q12/15, 1995.

[11] 片岡章俊, 林伸二, “ITU-T 標準 8-kbit/s 音声符号化（CS-ACELP）,” 音響学会講演論文集, 3-1-16, pp.299-300, 1995-9.

[12] 東山恵祐, 陸金林, 中村哲, 鹿野清, 河原英紀, “音声分析・変換・合成法 STRAIGHT の音声符号化への適用について,” 電子情報通信学会, SP98-10, pp.13-18, 1998-4.

[13] 東山恵祐, 陸金林, 中村哲, 鹿野清, 河原英紀, “4kHz 帯域の STRAIGHT の品質評価と情報圧縮について,” 音響学会講演論文集, 1-2-6, pp.207-208, 1997-9.

[14] 東山恵祐, 陸金林, 中村哲, 鹿野清, 河原英紀, “STRAIGHT の声道情報の量子化についての検討,” 音響学会講演論文集, 2-7-5, pp.243-244, 1998-3.

[15] 平井啓之, 橋本誠, 大西宏樹, “STRAIGHT を用いたテキスト音声合成の開発と評価,” 音響学会講演論文集, 1-P-7, pp.369-370, 2001-10.

学会発表リスト

- 越智崇夫, Nguyen Phu Chien, 赤木正人, “Temporal Decomposition を用いた STRAIGHT 用低ビットレート音声符号化,” 音響学会春季発表会, 3-10-24, 2002.