音韻モデルの作成

目 次

第 1 章　音韻モデルの仕様 *1*
1.1　モデル構成条件 *1*
1.2　学習条件 *2*
1.3　その他 *2*

第 2 章　納入モデル一覧 *3*

第 3 章　日本語音韻モデル概要 *6*
3.1　音響分析条件 *6*
3.2　音素体系 *6*
3.3　HMM のトポロジー *8*
3.4　Triphone 体系 *8*
3.5　音素環境のクラスタリングによる状態共有 *10*
3.6　混合数の増加 *13*
3.7　Phonetically Tied Mixture(PTM) モデルの作成 *13*
3.8　学習データに対する平均尤度 *13*

第 4 章　納入モデル構造説明 *15*
4.1　HTK 関連ファイルの形式 *15*
4.1.1　音韻モデルファイル *15*
4.1.2　分析条件指定ファイル *16*
4.1.3　ラベルファイル・マスターラベルファイル *19*

第 5 章　モデル作成の実際 *20*
5.1　ディレクトリ構造 *20*
5.2　モデル作成手順 *21*
5.2.1　初期モデル (モノフォン) の作成 *21*
5.2.2　triphone モデルの作成 *21*

5.2.3 状態クラスタリング 21
5.2.4 混合数の増加 24
5.2.5 コンテキストの外挿 26
5.2.6 PTM の作成 27

第 6 章 話者適応プログラムマニュアル 32
6.1 話者適応プログラムについて 32
6.2 話者適応プログラムの仕様 32
6.2.1 インストール手順 32
6.2.2 使用方法 33
6.2.3 使用ファイルの説明 34
6.3 話者適応プログラムの構造 34
6.4 VFS による話者適応 35

参考文献 37

第1章　音韻モデルの仕様†

1.1 モデル構成条件

音響分析：モデルは表 1.1 に示す音響分析により得られる音響パラメータ（２５次元）を用いて作成する。ＨＭＭ構造：ＨＭＭの基本単位は前後の音素文脈に応じて分類された音素すなわちトライフォンとする。音素数は２０～３０とする。ＨＭＭの形式は 3 状態からなる left-to-right 型とする。状態毎の出力確率分布は無相関混合ガウス分布により表現するものとする。トライフォン間で状態の共有を許し、全体で数千程度の状態により数万のトライフォンをモデル化する。ただし，同一中心音素の同一状態では，64 個の無相関ガウス分布を共有し，それぞれの分布に対する重みのみを調整することにより，トライフォンを構築するものとする（Phonetic Tied Mixture 構成）。したがって，音素数ｘ３（音素当たりの状態数）ｘ６４個の無相関ガウス分布と，状態共有化されたトライフォン種類数（数千）X ６４個の重みパラメータ，を学習する。モデル化は、男性・女性音声独立（Gender Dependent）と男性・女性音声混合（Gender Independent）の双方を行なう。

表 1.1. 音響分析条件

サンプリング周波数	16 [kHz]
プリエンファシス	0.97
分析窓	Hamming 窓
分析窓長	25 [ms]
窓間隔	10 [ms]
特徴パラメータ	MFCC(12 次) + Δ MFCC + Δ Pow (計 25 次)
周波数分析	等メル間隔フィルタバンク
フイルタバンク	24 チャネル
CMS	発声単位で実行

† 武田 一哉 (名古屋大学 工学部)

1.2 学習条件

学習データ : 学習に用いる発声は、(１) 出現音韻のバランスを考慮しない新聞記事、(２) 出現音韻のバランスを考慮したテキスト、の 2 種類のテキストの読み上げとする。学習は男女それぞれ１００名以上の音声を用いて行う。話者毎の発声は１００文以上とし、男女それぞれについて合計 2 万文以上を学習に用いる。繰り返し学習回数 : 最終的なトライフォンモデルを得る過程において、モデルの構造を変化させる毎に最低１０回の繰り返し学習を行う

1.3 その他

中間成果物：トライフォンモデル (PTM) と同一の基底分布により表現されたモノフォンモデルを中間成果物とする。

第2章　納入モデル一覧†

```
+ model ---+-- monof --+-- mix4 --+-- gid ---- hmmdefs
|          |           +-- mix8 --+-- gid ---- hmmdefs
|          |           \-- mix16--+-- gid ---- hmmdefs
|          +-- s1000 --+-- mix4 --+-- male --- hmmdefs
|          |           |          \-- female - hmmdefs
|          |           +-- mix8 --+-- male --- hmmdefs
|          |           |          \-- female - hmmdefs
|          |           +-- mix16--+-- male --- hmmdefs
|          |           |          \-- female - hmmdefs
|          |           + triphone,male,1000
|          |           + tdcTree,male1000
|          |           + triphone,female,1000
|          |           \ tdcTree,female1000
|          +-- s2000 --+-- mix4 --+-- male --- hmmdefs
|          |           |          +-- female - hmmdefs
|          |           |          \-- gid ---- hmmdefs
|          |           +-- mix8 --+-- male --- hmmdefs
|          |           |          +-- female - hmmdefs
|          |           |          \-- gid ---- hmmdefs
|          |           +-- mix16--+-- male --- hmmdefs
|          |           |          +-- female - hmmdefs
|          |           |          \-- gid ---- hmmdefs
|          |           + triphone,male,2000
|          |           + tdcTree,male.2000
|          |           + tdcTree,gid.2000
|          |           + triphone,female,2000
```

† 武田 一哉 (名古屋大学 工学部)

```
|          |          \ tdcTree,female,2000
|          +-- s3000 --+-- mix4 --+-- male --- hmmdefs
|          |           |          \-- female - hmmdefs
|          |           +-- mix8 --+-- male --- hmmdefs
|          |           |          \-- female - hmmdefs
|          |           +-- mix16--+-- male --- hmmdefs
|          |           |          \-- female - hmmdefs
|          |           + triphone,male,3000
|          |           + tdcTree,male,3000
|          |           + triphone,female,3000
|          |           \ tdcTree,female,3000
|          +-- PTM ----+-- male --+-- hmmdefs <
|                      |          +-- triphone,male,3000 <
|                      |          \-- tdcTree,male,3000  <
|                      +-- female +-- hmmdefs <
|                      |          +-- triphone,female,3000 <
|                      |          \-- tdcTree,female,3000  <
|                      +-- gid ---+-- tri --+-- hmmdefs <
|                      |          |         +-- triphone,gid,3000 <
|                      |          |         \-- tdcTree,gid,3000 <
|                      |          \-- monof +-- hmmdefs <
|                      +- tools --+ mkptm
|                                 + ext-mixdef
|                                 + ext-statedef
|                                 + ext-sildef
|                                 + ext-hmmdef
+ tools
+ labs
\ params + config.mfcc
         + config.train
         + logicalTri
         + mlf,monof
         + mlf,tri
```

```
+ monophones
+ physicalTri
+ tdc.hed
+ train,female.scp
+ train,gid.scp
\ train,male.scp
```

図 **2.1.** 納入モデルとそのディレクトリ構成

納入モデルならびに関連ツールと格納ディレクトリ構造を図 **2.1** に示す．図中「<」は９９年度版で新たに加えたファイルを，それぞれ示している。model ディレクトリの直下には s1000,s2000, s3000,monof の３つのディレクトリがあり，それぞれ 1000, 2000, 3000 状態の状態共有により構成されたトライフォン音韻モデル及び，文脈非依存音韻モデルが格納されている．s1000, s2000, s3000,monof の直下には，状態あたりの混合数に関するディレクトリ (mix4 ~ mix16) があり，さらに男性用・女性用に別れたディレクトリに納入モデルが格納されている．s2000 と monof には，性別非依存のモデルが gid の下に格納されている。PTM(Phonetically Tied Mixture) モデルモデル作成に利用したツール，データファイルはそれぞれディレクトリ tools, params に格納されている．ツール，データディレクトリの内容は９７年度版解説書 ３．３節において述べるとおりである。

第3章　日本語音韻モデル概要†

3.1 音響分析条件

作成した音素モデルの音響分析条件は表 3.1 に示すとおりである．特徴パラメーターには MFCC[3] を用い，動的な特徴パラメータとして ΔMFCC ならびに、ΔLogPow を用いている．音響分析は HTK version2.0 [4] の HCopy コマンドを用いて行なった．さらにマイクロフォン等，入力環境の異る様々なシステムの評価に利用することを考慮して，学習データの発声を単位としたケプストラム平均除去，(CMS) [5] を行なっている．これらの音響分析を HTK により行なう際のコマンドファイル (config.mfcc) はツールディレクトリに格納されている．

3.2 音素体系

音素体系及び仮名表記との対応は，日本音響学会連続音声データベース [6] のローマ字表記に準じた．作成された音素モデルは表 3.2 に示す合計 43 種類である．silB/E は文頭 /

表 3.1. 音響分析条件

サンプリング周波数	16 [kHz]
プリエンファシス	0.97
分析窓	Hamming 窓
分析窓長	25 [ms]
窓間隔	10 [ms]
特徴パラメータ	MFCC(12 次) + Δ MFCC + Δ Pow (計 25 次)
周波数分析	等メル間隔フィルタバンク
フイルタバンク	24 チャネル
CMS	発声単位で実行

† 武田 一哉 (名古屋大学 工学部)

表 3.2. 音素表

a i u e o a: i: u: e: o: N w y j my ky dy by gy ny hy ry py p t k ts ch b d g z m n s sh h f r q sp silB silE

```
0 2000000 sil
2000000 2500000 sp
2500000 3100000 a
3100000 3400000 r
3400000 4400000 a
   . . . .

9700000 10500000 j
10500000 10800000 i
10800000 12300000 ts # 音素 u が脱落
12300000 14600000 o
14600000 14900000 sp
14900000 16300000 s  # 音素 u が脱落
16300000 16900000 b
16900000 17900000 e
17900000 18400000 t
18400000 19200000 e
   . . . .
```

発声内容：あらゆる現実を全て....

図 3.1. 学習データの音素表記例

文末の，sp は文節間の無音モデルを表しており，q には促音に伴う無音に対応させている ．

学校 => g a q k o u

また a: ~ i: 等は長母音を表している．

音素モデルの連結学習は，発声の音素表記 (音素ラベル) に従って HTK version2.0 の HERest を用いて行なった．音素ラベルは，発声内容のローマ字表記から以下の手順で作成した．

- ローマ字表記から，発声のバリエーション (無声化，ポーズの挿入位置など) を含む音素ネットワークを生成する．
- 音素ネットワークを構文規則として，事前に用意した不特定話者ＨＭＭにより学習音声の認識を行ない，発声に対応する音素系列と音素境界を決定する．(ただし，モデルの繰り返し推定に用いられるのは音素系列に関する情報のみで，音素境界は用いない．)

作成された音素ラベルの一例を図 3.1 に示す．このラベルは無声化に伴う母音 (u) の脱落を反映している．ただし後述するトライフォンには子音の連続を認めていないため，上の例の音素ラベルは，トライフォンラベルでは

sp s b → sp s+u u-b+e

と展開される．

3.3 HMM のトポロジー

ＨＭＭの状態数は 5 状態とし，状態間の遷移は自己遷移と「左から右」への遷移のみを許す．ただし第 1 状態 (開始状態) では第 2 状態への状態のみを許し第 5 状態 (終了状態) からの状態遷移は許していない．また，第 1 状態と第 5 状態からの遷移に伴う出力はない．(ヌル遷移)

3.4 Triphone 体系

提供されるモデルは，隣接する音素の影響を考慮して詳細に音韻の音響特徴をモデル化することが可能なトライフォンモデルである．トライフォンモデルでは中心音素 X 先行音素 L 後続音素 R の 3 つ組を L-X+R と表記して一つの単位としてあつかう．以下は「あらゆる現実を..」を音素とトライフォンで表記した例である．

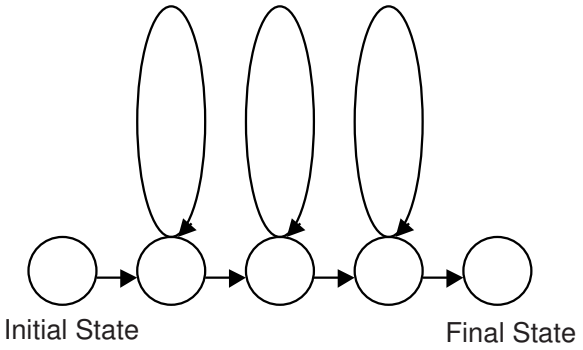


図 3.2. ＨＭＭの状態構造

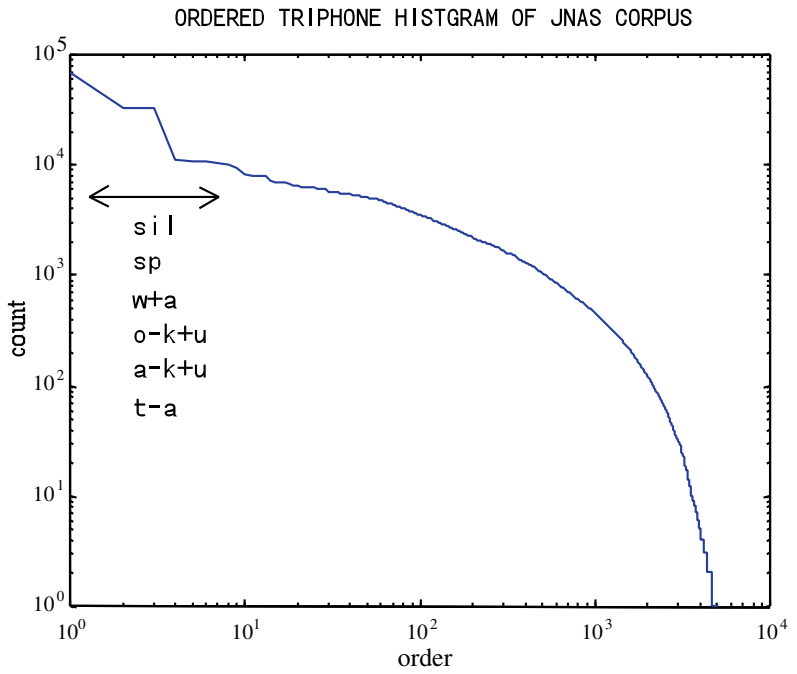


図 3.3. トライフォンの出現数と累積種類．

音素	a r a y u r u g N j i ts u o
トライフォン	a+r a-r+a r-a+y a-y+u y-u+r u-r+u r-u+g u-g+e ...

上記の音素体系では，子音の連鎖を除外すると約 21,000 種類 (コンテキストを考慮しない通常の音韻モデル (monophone) と，先行あるいは後続コンテキストのみを考慮したモデル (biphone) を含む) のトライフォンの出現が可能である．これらを「論理的」に存在可能なトライフォンであるという意味で， logical triphone と呼ぶ．

logical triphone は以下のヒューリスティックな規則：

- 文脈において長母音と通常の母音との違いを無視する

```
a:-k+a => a-k+a
```

- 右音素文脈では拗音を区別しない

```
*-a+ky => *-a+k
```

- 拗音の左音素文脈を共通化する

```
ky-a+* => y-a+*
```

を用いて書き換えることで，約 $8{,}000$ 種類にまとめることができる．以降このトライフォンセットを physical triphone と呼ぶ．physical triphone は 3 ．3 節に示すとおり、ファイル `physicalTri` に格納されている．一方，学習コーパスに出現するトライフォン (以下では corpus triphone と呼ぶ) は physical triphone のうち約 $5{,}000$ 種類であり，実際のＨＭＭはこれらのトライフォンについてだけしか学習できない．

トライフォンモデルを用いた連続音声認識を任意の語彙に対して動作させるためには，全ての logical triphone に対応するＨＭＭが存在する必要がある．そのため学習データに存在しない logical triphone に corpus triphone を適切に対応させる規則が必要となる．ファイル `logicalTri` は，logical triphone と physical triphone との対応を定めており，第 1 フィールドのトライフォンが logical triphone を表している．上述の規則により，当該のトライフォンが physical triphone に置き換えられる場合には，第 2 フィールドに置き換え後のトライフォン名が記される．physical triphone の corpus triphone による置き換えは，次節に述べる音素環境のクラスタリングに基づいて行なわれる．

3.5 音素環境のクラスタリングによる状態共有

同一の中心音素を持つトライフォンモデルの各状態をクラスタリングし，クラスタ毎に状態間でパラメータを共有することでロバストなパラメータの推定を行なうことが一般的である．汎用の音素モデルを作成するという観点からは，語彙によらず精度の高いトライフォンモデルを提供することが望まれるため，トップダウンのクラスタリングにより文脈分類木を作成することが有効である [7]．トップダウンクラスタリングは，HTK version2.0 の HHEd コマンドにより行なった．基本的なアルゴリズムは，以下のとおりである．

1)　予め用意された分類条件にしたがって全ての状態クラスタを 2 分割する．

2)　1) で得られた分割のうち，2 分割後の状態クラスタ間の距離が最も大きい分割を用い，状態クラスタ数を 1 つ増やす．

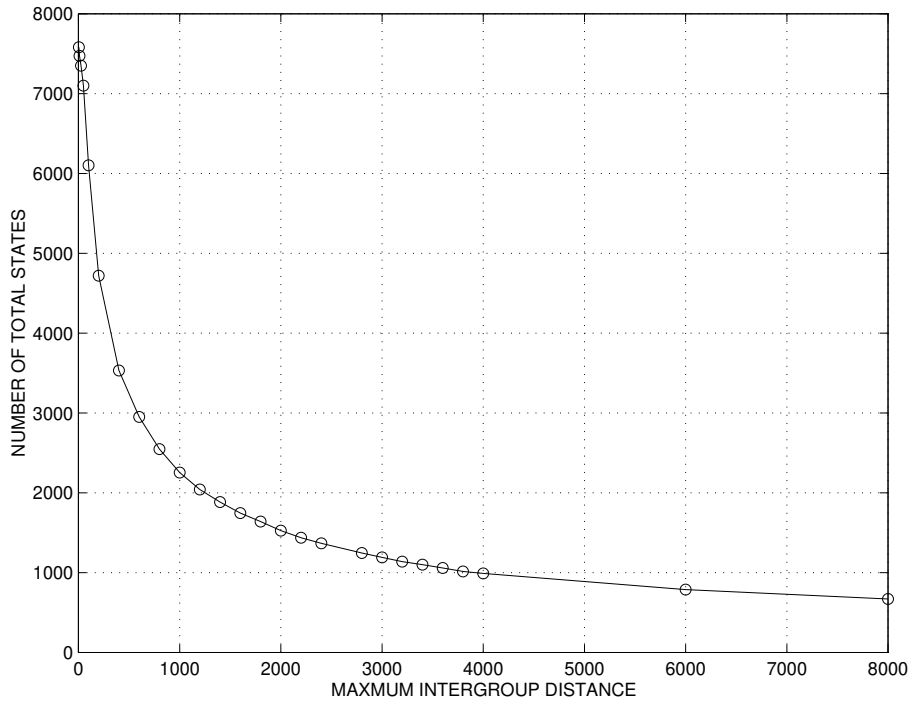


図 **3.4.** クラスタリングによる状態数の削減．

3)　2) の分割に伴う状態クラスタ間の距離が定められた閾値を下回らない限り 1) に戻り分割を続ける．

状態のクラスタ i,j 間の距離は，分散で正規化された平均ベクトル間のユークリッド距離として，以下に従い計算した．

$$d(i,j) = \frac{1}{S}\sum_{s=1}^{S}\left[\frac{1}{V}\sum_{k=1}^{Vs}\frac{(\mu_{isk}-\mu_{jsk})^2}{\sigma_{isk}\sigma_{jsk}}\right]^{\frac{1}{2}}$$

上式において s はストリーム (MFCC,ΔMFCC など個々の特徴パラメータベクトル) に対応するインデックスであり，S はストリーム総数 (本モデルの場合 1) である．μ，σ はそれぞれのクラスタに対応する正規分布の平均と分散を表している．またコンテキストの分類条件は主として調音位置に着目して表 2．3 のとおり設定した．

図 **3.4** に，トップダウンクラスタリングによるトライフォン状態数の削減の程度を示す．今回作成したモデルは，状態数がそれぞれ約 3000, 2000, 1000 となるように男性・女性独立にしきい値設定している．

状態毎に得られたコンテキスト分類木を用いて，未知の (学習データに出現しない) コンテキストを学習済みのトライフォンで置き換えることが可能となる．学習データに存在しないトライフォン状態を，コンテキスト分類木を用いて既学習のトライフォン状態に割り当てて得られた結果が，最終トライフォン状態セットである．

表 3.3. コンテキスト定義規則

規則名	該当トライフォン
L_Nasal	N-*,n-*,m-*
R_Nasal	*+N,*+n,*+m
L_Bilabial	p-*,b-*,f-*,m-*,w-*
R_Bilabial	*+p,*+b,*+f,*+m,*+w
L_DeltalAlveolar	t-*,d-*,ts-*,z-*,s-*,n-*
R_DeltalAlveolar	*+t,*+d,*+ts,*+z,*+s,*+n
L_PalatoAlveola	ch-*,j-*,sh-*
R_PalatoAlveola	*+ch,*+j,*+sh
L_Velar	k-*,g-*
R_Velar	*+k,*+g
L_Glottal	h-*
R_Glottal	*+h
L_YOUON	y-*
L_SOKUON	q-*
R_SOKUON	*+q
L_R	r-*
R_R	*+r
L_N	N-*
R_N	*+N
L_A	a-*
R_A	*+a
L_I	i-*
R_I	*+i
L_U	u-*
R_U	*+u
L_E	e-*
R_E	*+e
L_O	o-*
R_O	*+o

表 3.4. 作成モデルのクラスタリングしきい値

	3 000 状態	2 000 状態	1 000 状態
男性	500	1000	3500
女性	600	1200	3800
混合	800	2000	6000

3.6 混合数の増加

トライフォン状態のクラスタリングまでは，単一ガウス分布によりモデルの作成を行なった．これらのモデルの混合数を 2 ，4 ，8 ，１６の順で増やし，最終的な汎用トライフォン HMM を作成した．

3.7 Phonetically Tied Mixture(PTM) モデルの作成

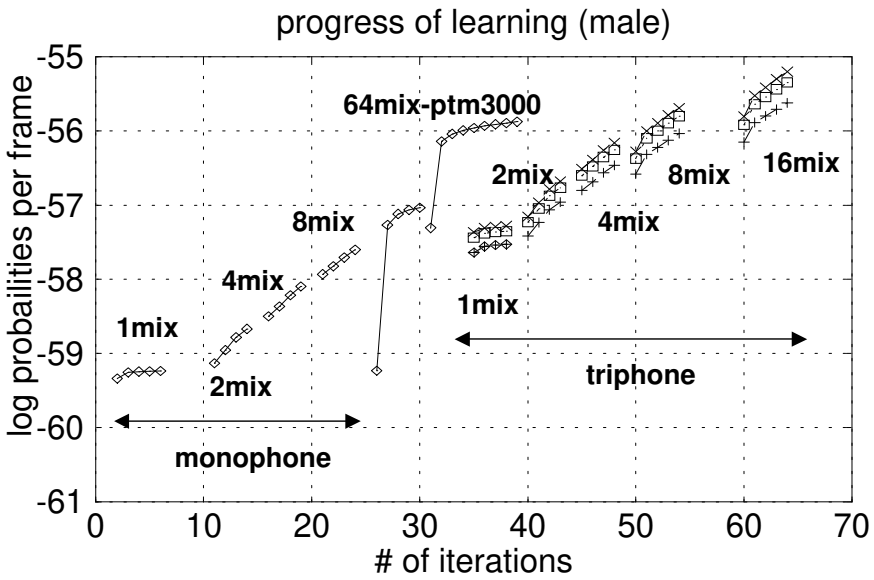
モノフォンモデルの混合数を，トライフォン同様に 2、４，8 と増加させることで作成した６４混合分布モデルを初期値として、PTM モデルを作成した。PTM モデルは、同一中心音素を持つ全てのトライフォンにおいて、状態毎に基底分布を共有する特殊なトライフォンモデルである。PTM に含まれる基底分布の数は　64 (混合数) X ３ (音素毎の状態数) X ４３ (音素種類数) = 約 8000 分布である。分布重みは、クラスタリングの結果得られた 3000 種類の状態それぞれについて学習された．

3.8 学習データに対する平均尤度

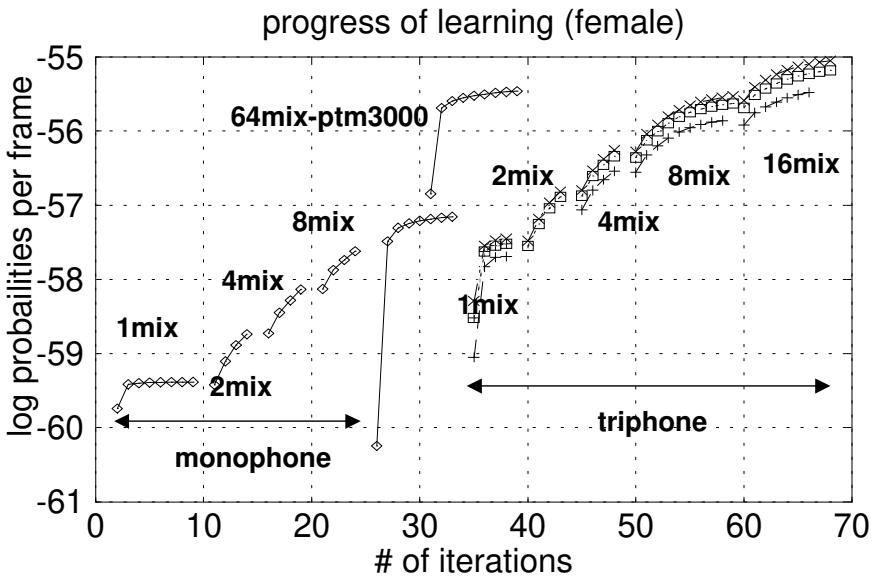
図 3.5 に上記の学習過程それぞれの段階における，学習データに対する平均尤度を示す．1mix,2mix 等は状態当たりのガウス分布の混合数を表しており，triphone に関しては，トップダウンのクラスタリング結果の状態数毎に尤度を示している．

トップダウンのクラスタリングに基づき状態共有を行なった 2 混合分布モデルで，状態共有を行なわない単一分布モデルと同等の平均対数確率密度が得られた．これらの結果から，状態の共有化による分布数の制御よりも混合数を増加させる方がモデルの尤度向上により効果的であることがわかる。

さらに，64 混合の PTM を用いた場合、４ ~ 8 混合のトライフォンを用いる場合とほぼ同様の尤度が得られた。



(a) 男性



(b) 女性

図 3.5. 学習回数毎の学習データに対する平均対数確率密度 (フレーム当たり) .

第4章　納入モデル構造説明†

4.1 HTK関連ファイルの形式

4.1.1 音韻モデルファイル

提供される音韻モデルはＨＴＫの音韻モデルファイルの形式に準拠しており，以下の構造からなる。

ファイル先頭部 o

ファイルの先頭部では、異なる音韻モデル間で共通なパラメータ (globalOpts)) が格納される。下に示す配布モデルの例では、特徴パラメータは次元数は $1 \sim 25$ 次元まで単一のベクトルとしてあつかう。

```
<STREAMINFO> 1 25
```

全体のベクトルの次元数は 25 次元である。

```
<VECSIZE> 25
```

状態継続時間の制御は行なわない。

```
<NULLD>
```

利用する特徴パラメータは、MFCC $+\Delta$ MFCC と ΔPower であり、ＣＭＳを発声単位で行なっている。

```
<MFCC_E_D_Z>
```

といった内容が書かれている。

† 武田 一哉 (名古屋大学 工学部)

マクロ定義部 t, s

提供されるモデルでは状態遷移確率は、同一の中心音素を持つトライフォン全てに対して同一の値が与えられている。下の定義は、中心音素が N である全てのトライフォンに共有される状態遷移確率を行列形式に定義し、TIt_N という名前を与えている。

```
~t "TIt_N"
<TRANSP> 5
 0.000000e+00 1.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
 0.000000e+00 6.516595e-01 3.483405e-01 0.000000e+00 0.000000e+00
 0.000000e+00 0.000000e+00 7.266717e-01 2.733283e-01 0.000000e+00
 0.000000e+00 0.000000e+00 0.000000e+00 5.833134e-01 4.166866e-01
 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
```

状態共有化処理が行なわれたモデルでは、同一の状態が異なる音韻モデル間で共有される。そのため、クラスタリングにより得られた全ての状態を名前とともに、予め定義する。図の例は、音素 i の第 2 状態の一つを定義している。

```
<NUMMIXES> 2
```

は、当該状態が 2 混合分布からなっていることを表しており、

```
<MIXTURE> 1 5.000000e-01
```

は 1 番目の分布の分岐確率が 0.5 であることに対応している。

```
<MEAN> 25 ..........
<VARIANCE> 25 ......
<GCONST> 8.544662e+01
```

の 3 行はそれぞれ（１）多次元正規分布の平均ベクトル、（２）分散ベクトル、（３）確率密度関数の定数部分 $\frac{1}{(2\pi)^{n/2}\prod_{i=1}^{25}\sigma_i}$ を予め計算した値、に対応している．

4.1.2 分析条件指定ファイル

音響分析に用いた条件指定ファイルは図に示すとおり．

```
~s "TC_i3_2"
<NUMMIXES> 2
<MIXTURE> 1 5.000000e-01
<MEAN> 25
  2.688612e-01 8.211005e+00 1.627891e+01 -9.258192e+00 -7.451896e+00
 -1.448812e+00 -7.768027e+00 -7.621338e+00 -9.995598e-01 3.628927e+00
 -1.476366e+00 2.528450e+00 3.503191e-01 2.016549e-01 2.403739e-01
  3.950355e-01 2.007166e-01 4.091309e-01 -3.240004e-03 3.393190e-01
  5.975805e-01 3.879358e-01 2.803141e-01 5.337226e-01 -2.467386e-01
<VARIANCE> 25
  1.202541e+01 2.051532e+01 2.044661e+01 3.172720e+01 2.725772e+01
  2.422007e+01 2.708308e+01 2.127524e+01 2.756099e+01 3.125399e+01
  2.650542e+01 2.840686e+01 7.975645e-01 1.116986e+00 1.306246e+00
  2.231485e+00 1.449811e+00 1.320115e+00 1.236644e+00 1.146589e+00
  1.288562e+00 1.270534e+00 1.260544e+00 1.101223e+00 2.257046e-01
<GCONST> 8.544662e+01
<MIXTURE> 2 5.000000e-01
<MEAN> 25
 -1.118246e+00 6.399251e+00 1.447019e+01 -1.151127e+01 -9.540254e+00
 -3.417368e+00 -9.849683e+00 -9.466341e+00 -3.099502e+00 1.392716e+00
 -3.535702e+00 3.965263e-01 -6.906733e-03 -2.210953e-01 -2.167906e-01
 -2.024907e-01 -2.809158e-01 -5.045412e-02 -4.480580e-01 -8.899660e-02
  1.435211e-01 -6.293614e-02 -1.687816e-01 1.139660e-01 -4.367720e-01
<VARIANCE> 25
  1.202541e+01 2.051532e+01 2.044661e+01 3.172720e+01 2.725772e+01
  2.422007e+01 2.708308e+01 2.127524e+01 2.756099e+01 3.125399e+01
  2.650542e+01 2.840686e+01 7.975645e-01 1.116986e+00 1.306246e+00
  2.231485e+00 1.449811e+00 1.320115e+00 1.236644e+00 1.146589e+00
  1.288562e+00 1.270534e+00 1.260544e+00 1.101223e+00 2.257046e-01
<GCONST> 8.544662e+01
```

図 4.1. 状態定義の一例

```
# HTK Configuration Parameters for Generating MFCC_D_E_Z from JNAS
# Continuous SPEECH Corpus.
# Copyright 1997 Kazuya TAKEDA
# takeda@nuee.nagoya-u.ac.jp

SOURCEFORMAT=NOHEAD # ASJ Copus has no header part
SOURCEKIND = WAVEFORM
SOURCERATE = 625    # surce sampling frequency is 16 [kHz]

TARGETKIND = MFCC_E_D_Z
TARGETRATE=100000.0 # frame interval is 10 [msec]
SAVECOMPRESSED=F    # set T, if you like to save disk storage
SAVEWITHCRC=F
WINDOWSIZE=250000.0 # window length is 25 [msec]
USEHAMMING=T        # use HAMMING window
PREEMCOEF=0.97      # apply highpass filtering
NUMCHANS=24         # # of filterbank for MFCC is 24
NUMCEPS=12          # # of parameters for MFCC presentation
ZMEANSOURCE=T
ENORMALISE=F
ESCALE=1.0
TRACE=0
RAWENERGY=F
BYTEORDER=SUN
```

図 **4.2.** 音響分析条件指定ファイルの内容

```
#!MLF!#
"*/m001/pb/*" -> "/lwork1/JNAS/lab/tri,JNAS/m001/pb"
"*/m001/np/*" -> "/lwork1/JNAS/lab/tri,JNAS/m001/np"
"*/m002/pb/*" -> "/lwork1/JNAS/lab/tri,JNAS/m002/pb"
"*/m002/np/*" -> "/lwork1/JNAS/lab/tri,JNAS/m002/np"
"*/m003/pb/*" -> "/lwork1/JNAS/lab/tri,JNAS/m003/pb"
"*/m003/np/*" -> "/lwork1/JNAS/lab/tri,JNAS/m003/np"
"*/m004/pb/*" -> "/lwork1/JNAS/lab/tri,JNAS/m004/pb"
"*/m004/np/*" -> "/lwork1/JNAS/lab/tri,JNAS/m004/np"
"*/m005/pb/*" -> "/lwork1/JNAS/lab/tri,JNAS/m005/pb"
```

図 4.3. マスターラベルファイルの内容

4.1.3 ラベルファイル・マスターラベルファイル

納入物にはモデルの学習に用いたラベルファイルを添付した．ラベルファイルは，音声ファイルに対応するモデルの系列を記述するものであり，モノフォンモデル用のファイルと，トライフォンモデル用のファイルが，全ての発声ファイルに作成さいれている．一方，マスターラベルファイルは音声ファイル名から当該のラベルファイルが格納されているディレクト等を指定する．下の例では，話者 m001 について pb というサブディレクトリに格納された発声のラベルが，/lwork1/JNAS/lab/tri,JNAS/m001/pb に格納されていることを示している．

第5章　モデル作成の実際†

5.1 ディレクトリ構造

```
model +--+ monof ---+- hmmXXYY - hmmdefs
          |           |    (XX; 混合数 = 01,02,04,08,16
          |           |     YY; 繰り返し学習回数 = 00,01,...10)
          |           +- log --+-- log,hmmXXYY
          |                     +-- stat,hmmXXYY
          + tri---- -+- hmmXXYY - hmmdefs
          |           |    (XX; 混合数 = 01
          |           |     YY; 繰り返し学習回数 = 00,01,...10)
          |           +- log --+-- log,hmmXXYY
          |                     +-- stat,hmmXXYY
          + tri2000 -+- hmmXXYY - hmmdefs
          |           |    (XX; 混合数 = 01,02,04,08,16
          |           |     YY; 繰り返し学習回数 = 00,01,...10)
          |           +- log --+-- log,hmmXXYY
          |                     +-- stat,hmmXXYY
          + work --+ monophones
                   + triphones
                   + allTriphones
                   + tri.hed
                   + clustering.hed
                   + train.monof.mu
                   + train.tri2000.mu
                   + tdcList
                   + tdcTree
```

† 武田 一哉 (名古屋大学 工学部)

```
+ config.train
+ gid.scp
+ gid.tri.mlf
```

本章で解説するモデル作成では、上記のディレクトリ構造を仮定する。

5.2 モデル作成手順

5.2.1 初期モデル (モノフォン) の作成

5.2.2 triphone モデルの作成

学習した単一ガウス分布のモノフォンモデルを複製し、単一ガウス分布のトライフォンモデルを作成する。

```
H2HEd -H ../monof/hmm0110/hmmdefs -w ../tri/hmm0100/hmmdefs tri.hed monophones
```

単一ガウス分布のトライフォンモデルを１０回繰り返し学習を行なう。

5.2.3 状態クラスタリング

学習された、トライフォンモデルをクラスタリングにより、状態共有を行なう。

```
cp ../tri/log/stat,hmm0110 stat
H2HEd -T 00407 \
      -D \
      -H ../tri/hmm0110/hmmdefs \
      -w ../tri2000/hmm0100/hmmdefs \
      clustering.hed triphones > ../log/log,tdc,${hmmid}
```

clustering.hed の内容は，

```
LS stats

QS "L_Nasal"   { N-*,n-*,m-* }
QS "R_Nasal"   { *+N,*+n,*+m }
```

```
QS "L_Bilabial"       { p-*,b-*,f-*,m-*,w-* }
QS "R_Bilabial"       { *+p,*+b,*+f,*+m,*+w }

QS "L_DeltalAlveolar" { t-*,d-*,ts-*,z-*,s-*,n-* }
QS "R_DeltalAlveolar" { *+t,*+d,*+ts,*+z,*+s,*+n }

QS "L_PalatoAlveolar"  { ch-*,j-*,sh-* }
QS "R_PalatoAlveolar"  { *+ch,*+j,*+sh }

QS "L_Velar"          { k-*,g-* }
QS "R_Velar"          { *+k,*+g }

QS "L_Glottal"        { h-* }
QS "R_Glottal"        { *+h }

QS "L_YOUON"      { y-* }

QS "L_SOKUON"     { q-* }
QS "R_SOKUON"     { *+q }

QS "L_R"          { r-* }
QS "R_R"          { *+r }

QS "L_N"          { N-* }
QS "R_N"          { *+N }

QS "L_A"          { a-* }
QS "R_A"          { *+a }
QS "L_I"          { i-* }
QS "R_I"          { *+i }
QS "L_U"          { u-* }
QS "R_U"          { *+u }
QS "L_E"          { e-* }
```

```
QS "R_E"          { *+e }
QS "L_O"          { o-* }
QS "R_O"          { *+o }

UF macro

TB 2000 "TC_N2_" {("N","*-N+*","N+*","*-N").state[2]}
TB 2000 "TC_a2_" {("a","*-a+*","a+*","*-a").state[2]}
TB 2000 "TC_a:2_" {("a:","*-a:+*","a:+*","*-a:").state[2]}
TB 2000 "TC_b2_" {("b","*-b+*","b+*","*-b").state[2]}
TB 2000 "TC_by2_" {("by","*-by+*","by+*","*-by").state[2]}
TB 2000 "TC_ch2_" {("ch","*-ch+*","ch+*","*-ch").state[2]}
TB 2000 "TC_d2_" {("d","*-d+*","d+*","*-d").state[2]}
TB 2000 "TC_dy2_" {("dy","*-dy+*","dy+*","*-dy").state[2]}
TB 2000 "TC_e2_" {("e","*-e+*","e+*","*-e").state[2]}
TB 2000 "TC_e:2_" {("e:","*-e:+*","e:+*","*-e:").state[2]}
TB 2000 "TC_f2_" {("f","*-f+*","f+*","*-f").state[2]}
TB 2000 "TC_g2_" {("g","*-g+*","g+*","*-g").state[2]}
TB 2000 "TC_gy2_" {("gy","*-gy+*","gy+*","*-gy").state[2]}

                .
                .
                .
                .

TB 2000 "TC_p4_" {("p","*-p+*","p+*","*-p").state[4]}
TB 2000 "TC_py4_" {("py","*-py+*","py+*","*-py").state[4]}
TB 2000 "TC_q4_" {("q","*-q+*","q+*","*-q").state[4]}
TB 2000 "TC_r4_" {("r","*-r+*","r+*","*-r").state[4]}
TB 2000 "TC_ry4_" {("ry","*-ry+*","ry+*","*-ry").state[4]}
TB 2000 "TC_s4_" {("s","*-s+*","s+*","*-s").state[4]}
TB 2000 "TC_sh4_" {("sh","*-sh+*","sh+*","*-sh").state[4]}
TB 2000 "TC_t4_" {("t","*-t+*","t+*","*-t").state[4]}
```

```
TB 2000 "TC_ts4_" {("ts","*-ts+*","ts+*","*-ts").state[4]}
TB 2000 "TC_u4_" {("u","*-u+*","u+*","*-u").state[4]}
TB 2000 "TC_u:4_" {("u:","*-u:+*","u:+*","*-u:").state[4]}
TB 2000 "TC_w4_" {("w","*-w+*","w+*","*-w").state[4]}
TB 2000 "TC_y4_" {("y","*-y+*","y+*","*-y").state[4]}
TB 2000 "TC_z4_" {("z","*-z+*","z+*","*-z").state[4]}

CO "tdcList"
ST "tdcTree"
```

上で，2000 はクラスタリングの閾値であり，大きいほど少ない状態数にクラスタリングされる。tdcList は，クラスタリングの結果，全く同一内容のトライフォンが生成された場合のリストファイル名を示す。tdcTree は，クラスタリングの結果生成される音素環境木の出力ファイルを示す。

コマンドの結果出力されるログファイルには，音素・状態毎にクラスタリング後の状態数が出力される。出力中

```
TB 2000.00 TC_u:4_ {}
 TB: Stats 22->3 [13.6%]  { 14993->407 [2.7%] total }
```

が，状態数の削減に関する情報である。22->3 は，２２あった音素 u:の第４状態が３状態に統合されたことを示している。さらに，14993 ->407 は全状態数が４０７まで削減されたことを，示している。

閾値を変化させ，クラスタリング後の状態数が所望の状態数になるように調整する。

5.2.4 混合数の増加

クラスタリングが終了した単一混合のモデルの，混合数を増加させながら学習を行なう。具体的な学習コマンドを以下に示す。

```
#!/bin/csh
# train triphones with increasing mixture
# IPA Japanese dictation software
```

```
# Copyright 1998 Kazuya Takeda, takeda@nuee.nagoya-u.ac.jp
#

source /data/schroeder4/HTK21/env/htkrc.linux;
set triphoneList = triphones;
set dir          = ../tri2000;
set orgmix       = 01;

foreach mix (02 04 08 16 32)
if (! -d ${dir}/hmm${mix}00) mkdir ${dir}/hmm${mix}00;
gzip -dc ${dir}/log/stat,hmm${orgmix}10.gz > stat
H2HEd -H ${dir}/hmm${orgmix}10/hmmdefs \
      -w ${dir}/hmm${mix}00/hmmdefs \
         mixup${mix}.hed ${triphoneList}

set org = hmm${mix}00
foreach id (01 02 03 04 05 06 07 08 09 10)
 set dst = hmm${mix}${id};
 echo -n "${org}=>${dst}"; date;
 set dstD = ${dir}/${dst};
 set orgD = ${dir}/${org};
 if (! -d ${dstD}) mkdir ${dstD};
 H2ERest -H ${orgD}/hmmdefs -M $dstD \
         -s ${dir}/log/stat,$dst \
         -D \
         -t 250.0 150.0 1000.0 \
         -C config.train \
         -S gid.scp \
         -T 1 \
         -I gid.tri.mlf \
          ${triphoneList} >& $dir/log/log,$dst
 gzip $dir/log/log,$dst
 gzip $dir/log/stat,$dst
```

```
 set org = ${dst}
end
 gzip ${dir}/hmm${mix}0[0-9]/hmmdefs
 chomd -w ${dir}/hmm${mix}??/hmmdefs*
 chmod -w ${dir}/hmm${mix}??/hmmdegs.gz
 set orgmix = ${mix};
end
```

5.2.5 コンテキストの外挿

任意の語彙を認識するためには，学習データに出現しないトライフォンを作成する必要がある。これは，全てのトライフォンを環境分類木により分類し，学習された状態との対応を取ることで行なわれる。下の例では，最終的に hmmdefs,all という allTriphones に記入された全てのコンテキストに対応可能なモデルセットが生成される。

```
#!/bin/csh -f
# exttri: csh script
# extrapolate triphone models using topdown clustring tree
# Copyright 1997 Kazuya Takeda, takeda@nuee.nagoya-u.ac.jp
#
source /data/schroeder4/HTK21/env/htkrc.linux

 set orghmm     = ../tri2000/hmm1610/hmmdefs
 set dsthmm     = ../tri2000/hmmdefs,all
 set TreeRule   = tdcTree # Tree clusturing rules
 set msthmmList = allTriphones
 set atlhmmList = triphones   # actual hmm list

 echo "LT $TreeRule" > exttri.hed
 echo "AU $msthmmList" >> exttri.hed

 H2HEd \
  -w ${dsthmm} \
  -H ${orghmm} \
```

```
      exttri.hed ${atlhmmList}
```

5.2.6 PTM の作成

64混合モノフォンモデルと、状態共有されたトライフォンモデルから、PTM(Phonetically Tied Mixture) モデルの初期値を作成する。具体的な作成プログラムを以下に示す。ここでは、トライフォンモデルとして，../tri/hmm0110/hmmdefs をモノフォンモデルとして ../monof/hmm6410/hmmdefs を想定している．作成された，初期モデルを用いて，学習を行なうことで PTM が作成される．

PTM 作成スクリプト (mkptm)

```
#!/bin/csh
# mkptm: generate initial ptm from monof and tree
# Copyrith 2000 Kazuya Takeda, takeda@nuee.nagoya-u.ac.jp
# This is a part of IPA Japanese dictation software
#

set tridef = "../tri/hmm0110/hmmdefs";
set monof  = "../monof/hmm6410/hmmdefs";

if ($#argv != 2) then
 echo "Usage: mkptm trihmm monohmm";
else
 set tridef = $1;
 set monof = $2;
endif

if (! -f $tridef) then
 echo "can not open triphone file $tridef";
 exit;
endif
if (! -f $monof) then
 echo "can not open monophone file $monof";
```

```
 exit;
endif

# prin header
echo "~o";
echo "<STREAMINFO> 1 25";
echo "<VECSIZE> 25<NULLD><MFCC_E_D_N_Z>";

# print transitinal probabilities (except for sp, silB, silE)
perl -npe 'exit if (/~[smh]/);' $tridef | grep '~t' -A7

# print mixture definitions
ext-mixdef $monof

# print shared state definitions
grep '~s' $tridef | sort -u | ext-statedef

# print triphone definitions
ext-hmmdef $tridef

# generate sp and sil models
ext-sildef $monof
```

ミクスチャー、状態、モデルの定義を抽出するためのスクリプトを以下に示す。

混合要素定義の抽出スクリプト (ext-mixdef)

```
#!/usr/local/bin/perl
# ext-mixdef: extract mixture definitions
# Copyrith 2000 Kazuya Takeda, takeda@nuee.nagoya-u.ac.jp
# This is a part of IPA Japanese dictation software

while (<>) {
 if (/^~h \"(.*)\"/) { $p=$1; }
 if (/<[Nn][Uu][Mm][Ss][Tt][Aa][Tt][Ee]> (.*)/) { $ns=$1; };
```

```
 if (/<[Ss][Tt][Aa][Tt][Ee]> (.*)/) { $s=$1; };
 if (/<[Nn][Uu][Mm][Mm][Ii][Xx][Tt][Uu][Rr][Ee][Ss]> (.*)/) { $nm=$1; };
 if (/<[Mm][Ii][Xx][Tt][Uu][Rr][Ee]> (.*) (.*)/) { $id=$1; $w=$2 };

 if (/<[Mm][Ee][Aa][Nn]> (.*)/)
 {
   print "~m \"".$p.$s."m".$id."\"\n";
   print;        # Copy mean
   $_=<>; print;
   $_=<>; print; # Copy var
   $_=<>; print;
   $_=<>; print; # Copy Gcons
 };
}
```

状態定義の抽出スクリプト (ext-statedef)

```
#!/usr/local/bin/perl
# ext-statedef: extract state definitions
# Copyrith 2000 Kazuya Takeda, takeda@nuee.nagoya-u.ac.jp
# This is a part of IPA Japanese dictation software

while (<>) {
 if (/^~s \"TC_(.*)([234])_(.*)\"/) { $p=$1; $s=$2; $id=$3;
   print;
   print "<NUMMIXES> 64"."\n";
   foreach $m (1...64) {
    printf "<MIXTURE> %d %le\n", $m, 1/64;
    print "~m \"".$p.$s."m".$m."\"\n";
   }
 }
}
```

モデル定義抽出スクリプト (ext-hmmdef)

```
#!/usr/local/bin/perl
# ext-hmmedef: extract hmm definitions
# Copyrith 2000 Kazuya Takeda, takeda@nuee.nagoya-u.ac.jp
# This is a part of IPA Japanese dictation software

# skip everything before the first definition
$/ = "~h";
$_=<>;

# read until the ENDHMM appears
$/ = "<ENDHMM>";
$_= <>;

# The first record does not have '~h'
if (/^ \"(.*)\"/) {
 $p=$1;
 if ($p =~ /sp/) {
 } elsif ($p =~ /sil/) {
 } else {print '~h'; print;}
}

# copy out except for sil and sp
while (<>) {
 if (/~h \"(.*)\"/) {
 $p=$1;
 if ($p =~ /sp/) {
 } elsif ($p =~ /sil/) {
 } else {print;}
 }
}
print "\n";
```

無音定義生成スクリプト (ext-sildef)

```
#!/usr/local/bin/perl
# ext-sildef: extract sp and sil definitions
# Copyrith 2000 Kazuya Takeda, takeda@nuee.nagoya-u.ac.jp
# This is a part of IPA Japanese dictation software

# skip the header lines

$_=<>;
$_=<>;
$_=<>;

# read until the ENDHMM appears
$/ = "<ENDHMM>";

while (<>) {
 if (/~h \"(.*)\"/) {
 $p=$1;
 if ($p =~ /sp/) {
 print;
 } elsif ($p =~ /sil/) {
 print;
 } else {}
 }
}
print "\n";
```

第6章　話者適応プログラムマニュアル†

6.1 話者適応プログラムについて

このプログラムは、移動ベクトル場平滑化法 (Vector Field Smoothing: VFS) による話者適応プログラムである。HTK 形式の HMM 定義ファイル形式の音響モデルと、適応対象話者の音声波形サンプルより、話者適応した音響モデルを生成することができる。また、音響モデルは、モノフォンとトライフォン両方を話者適応することができる。第 2 節でプログラムの仕様、第 3 節ではプログラムの内部構造、第 4 節では VFS による話者適応のアルゴリズムについて詳しく説明する。

6.2 話者適応プログラムの仕様

6.2.1 インストール手順

1 準備

JULIUS のソースコードの存在するディレクトリで話者適応プログラムのソースパッケージを展開し、展開先のディレクトリへ移動する。

```
% uncompress -c adapt-1.0.tar.Z | tar -xvf -
% cd adapt
```

2 コンパイル

話者適応プログラムは make コマンドによりコンパイルされる。ディフォルトのコンパイラは gcc、コンパイルオプションは-O2 が設定されている。また、使用するコンパイラやコンパイルオプションは、環境変数 CC と CFLAGS により指定することができる。

```
% setenv CC cc
% setenv CFLAGS '-n32 -O'
% make -e
```

† 嵯峨山 茂樹 (北陸先端科学技術大学院大学 情報科学研究科)

6.2.2 使用方法

話者適応プログラムは、コマンドライン上では次のようにして使用する。

```
% adapt \
   -h /phone_m/model/s1000/mix4/male/hmmdefs.Z  \
   -hlist /phone_m/parms/logicalTri.added  \
   -adic word.adic -input rawfile -it 10 -tau 5.0 -lambda 5.0 -K 6 \
   -o hmmdefs
```

各オプションの意味は次の通りである。各係数の意味は第 4 節で説明する。

-h `hmmdeffile`
　適応元音響モデルのファイル名 (HTK の HMM 定義ファイル形式)

-hlist `hmmmapfile`
　HMMList ファイル名 (JULIUS 独自形式)

-adic `adaptdicfile`
　適応用音声波形の音素連結ファイル名

-input {`rawfile`,`mfcfile`} (`default:rawfile`)
　入力する音声データ形式の選択 (音声波形ファイルの他、MFCC ファイルの読み込みも可能)

-it (`default:10`)
　平均値ベクトルの連結学習の回数

-tau (`default:5.0`)
　適応元音響モデルの分布推定サンプル数に対する重み係数 ($\tau \approx MAP$)

-lambda (`default:5.0`)
　スムージング係数 (λ)

-K (`default:6`)
　考慮する近傍数 (K)

-o `adapthmmdeffile`
　話者適応された音響モデルのファイル名 (HTK の HMM 定義ファイル形式で出力)

話者適応プログラムの入出力ファイル及び各種係数のまとめを示す。

入力
1. 適応元音響モデルファイル (HTK の HMM 定義ファイル形式)
2. hmmlist ファイル (JULIUS 独自形式) (triphone モデルの場合は必要)
3. 音素連結ファイル
4. 音声波形ファイル (もしくは特徴パラメータファイル)
5. 各係数値 (-it -tau -lambda -K)

出力　話者適応された音響モデルファイル (HTK の HMM 定義ファイル形式)

6.2.3 使用ファイルの説明

適応元音響モデルファイルは、JULIUS で読み込み可能な制限付きの HTK の HMM 定義ファイル形式である。また音素連結ファイルは、話者適応プログラム独自の形式である。音素連結ファイルの例を次に示す。

```
FILE（word.adic）
[間]            silB a i d a silE
sm001a.ad
sm001b.ad
[医者]          silB i sh a silE
sm002a.ad
[うどん]        silB u d o n silE
sm003a.ad
EOF
```

一行目の [ ] 内は、適応用波形ファイルの識別名である。またそれに続いて、音素連結が記述されている。二行目は、その適応用音声波形ファイルである。ファイル名は、音素連結ファイルからの相対ディレクトリで指定する。この音声波形ファイルは 1 つ以上必要である。音声波形のファイル形式は RAW(16kHz,16bit(signed short),mono,big-endian) である。また、オプション指定 (-input) で mfcfile を指定すれば、HTK 形式のパラメータファイルが入力可能である。

6.3 話者適応プログラムの構造

図 **6.1** に、話者適応プログラムの内部構造を示す。(具体的な話者適応のアルゴリズムについては第 4 節を参照) 話者適応プログラムの処理の流れを次に示す。

STEP 1 適応元音響モデルファイルと素連結ファイルから、平均値学習用の音響モデルを生成する。

STEP 2 平均値ベクトルのみの Viterbi 学習により標準話者と適応対象話者の間の移動ベクトルを計算する。

STEP 3 得られた移動ベクトルと適応元音響モデルファイルから、VFS 法により新しい音響モデルファイルを生成する。

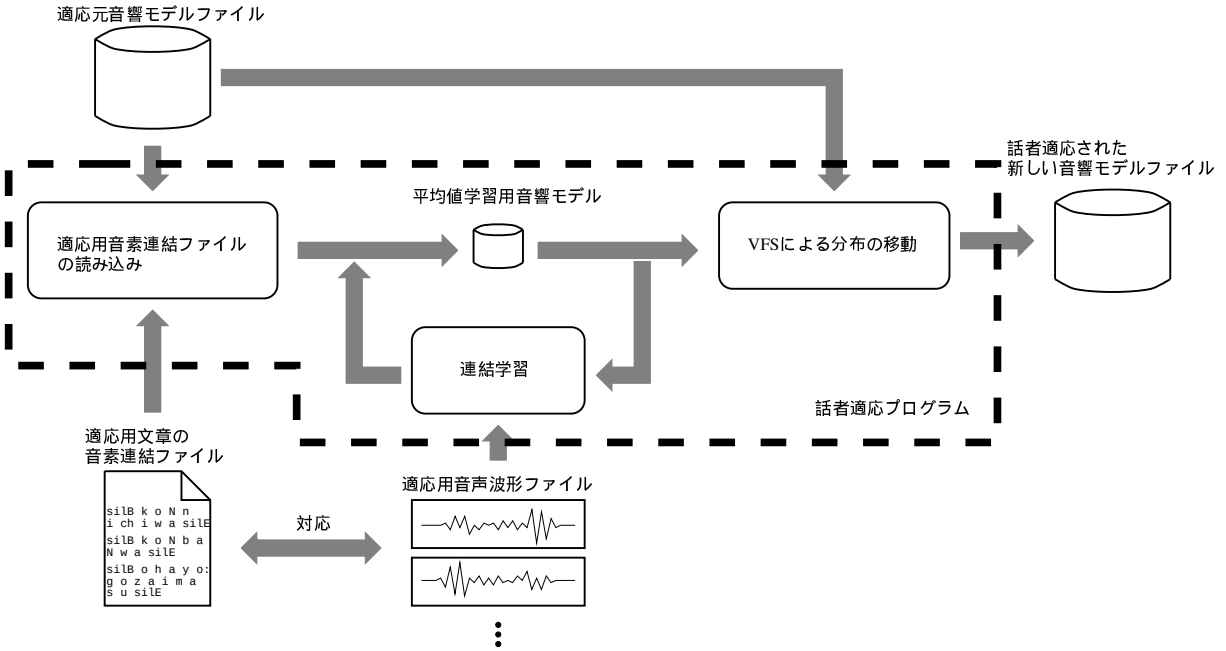


図 **6.1.** 話者適応プログラムの内部構造

6.4 VFS による話者適応

VFS による話者適応法は、標準話者の音響空間に対して、滑らかな収縮や並行移動などを行うことにより、適応対象話者の音響空間へ変換する手法である。VFS は、小量の適応用文章内に存在した各音素の音響空間内での変化量 (移動ベクトル) から適応対象話者の音響空間のゆがみを求めることにより適応を行っている。

次に VFS のアルゴリズムについて説明する。図 **6.2** は、VFS による分布移動の模式図である。

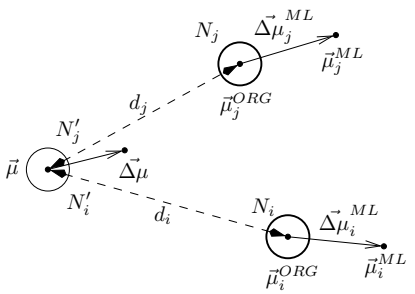


図 **6.2.** VFS による分布の移動

STEP 1 まず初めに、標準話者と適応対象話者との間のずれを調べるため、小量の適応サンプルを用いて、モデルパラメータの連結学習を行う。連結学習によって得られた分布の平均値ベクトルから、移動ベクトル $\vec{\Delta\mu}_i^{ML}$ を式 (1) により求める。$\vec{\mu}_i^{ORG}$ は標準

話者の平均値ベクトル、$\vec{\mu}_i^{ML}$ は連結学習によって得られた平均値ベクトルである。

$$\vec{\Delta\mu}_i^{ML} = \vec{\mu}_i^{ORG} - \vec{\mu}_i^{ML} \tag{6.1}$$

STEP 2 標準話者の分布と、その近傍に存在する移動ベクトルの得られた (K 個の) 分布の間の距離を式 (2) により計算する。

$$d_i = \frac{1}{D}\sum_{d=1}^{D}\frac{(\vec{\mu}_{id} - \vec{\mu}_{id}^{ORG})^2}{\sigma_{id}^2} \tag{6.2}$$

STEP 3 移動ベクトルを推定するために用いられたサンプル数 N_i と標準話者の分布までの距離 d_i から、それぞれの移動ベクトルが計算された分布の寄与 N_i'(サンプル数換算) を式 (3) により計算する。λ はスムージング係数である。

$$N_i' = \exp(-d_i/\lambda)N_i \tag{6.3}$$

STEP 4 各移動ベクトル $\vec{\Delta\mu}_i^{ML}$ とそれらの分布の寄与 N_i' から、分布の移動量 $\vec{\Delta\mu}$ を式 (4) により求める。またこの時、寄与したサンプル数の総和に対して $\tau(\approx MAP)$ の重みを付ける。

$$\vec{\Delta\mu} = \frac{\sum_n N_n' \vec{\Delta\mu}_n^{ML}}{\sum_n N_n' + \tau} \tag{6.4}$$

STEP 5 計算された $\vec{\Delta\mu}$ より、標準話者の分布の移動を式 (5) により行う。

$$\vec{\mu} = \vec{\mu} + \vec{\Delta\mu} \tag{6.5}$$

上記の処理により、すべての平均値ベクトルを適応対象話者の空間に変換することができる。

参考文献

[1] 伊藤克亘，武田一哉，竹沢寿幸，松岡達雄，鹿野清宏，“大語彙連続音声認識のための読み上げ文コーパスの構築,” 情報処理学会第 54 回 (平成 9 年前期) 全国大会，5H-10, Vol.2, pp.225–226 (1997-03).

[2] 板橋秀一，山本幹男，竹沢寿幸，小林哲則，“日本音響学会新聞記事読み上げ音声コーパスの構築,” 日本音響学会講演論文集 (1997-09).

[3] S.B.Davis and P.Mermelstein, “Comparison of parametric representations for monosyllabic word recognition in countinuously spoken sentences,” IEEE Trans. on Speech and Signal Processing, pp.357–366 (1980).

[4] S. Young, J.Jansen, J.Odell, D.Ollason, P.Woodland, “The HTK Book,” Entropic Research Lab. (1995).

[5] B.Atal, “Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification,” J. Acoust. Soc. Am., Vol.55, 6, pp.1304–1312 (1974).

[6] 小林哲則, 板橋秀一, 速水 悟, 竹沢寿幸, “日本音響学会研究用連続音声データベース”, 日本音響学会誌, Vol.48, No.12, pp.888–893 (1992).

[7] S.J. Young and P.Woodland, “The use of state tying in continuous speech recognition,” Proc. of Eurospeech’93, pp.2203–2206 (1993).

[8] J.R.Bellegarda and D.Nahamoo, “Tied Mixture Continuous Parameter Modelis for Large Vocabulary Isolated Speech Recognition,” Proc. of ICASSP’89, pp.13–16 (1989).

[9] 鷹見淳一，嵯峨山茂樹，“逐次状態分割による隠れマルコフ網の自動生成,” 信学論 (D-II), J76-DII, Vol.10, pp.2155–2164, (1993).