TR-I-0238

Automatic Phoneme Segmentation Using Continuous Hidden Markov Models

Richard Lengagne   Yasuhiro Komori

January 1992

Abstract

This report proposes a method for automatic phoneme segmentation of Japanese using continuous mixture density hidden Markov models (HMMs). Different kinds of training methods have been performed: word, phrase (bunsetsu) and sentence models are tested after training using single and multiple male speaker word data. The main experiments of this study, performed on 2 male speakers' utterances use HMMs trained on data from 8 other male speakers, and yield an average success rate of 95 % in segmentation within a deviation of 30 ms from the "hand"-determined boundaries.

The problem of word-spotting using "keyword"-segmentation will also be discussed.

ATR 自動翻訳電話研究所
ATR Interpreting Telephony Research Laboratories

Contents

1 Introduction

Speech recognition experiments have clearly demonstrated the usefulness of well-segmented speech data. An efficient automatic segmentation system is a basic requirement for obtaining good recognition results, and is also necessary for building large speech data-bases, without which satisfactory recognition experiments cannot be achieved. The main purpose of an automatic segmentation system is to obtain segmented data without human effort. Formerly, speech segmentation and label generation were carried out "by hand", reading the sound spectrograms and then matching a part of the speech wave to a given label. This therefore required much time and resources. Hence, the interest of automatic segmentation and labelling is obvious.

The present work investigates the efficiency of a hidden Markov model (HMM) based automatic labelling system applied to the Japanese language. The HMM technique has been broadly used in speech processing, for it provides a parametric model which can deal with many speech signal units, such as phonemes and syllables. Here, we use phoneme models trained by word utterances. Moreover, a HMM based automatic segmentation system can easily be applied to continuous speech. Therefore, the comparison between hand-labelling and automatic labelling is here performed not only with word utterances, but also with "bunsetsu" [1] and full sentence utterances.

The model considered here is a continuous mixture density HMM. In spite of its computational complexity, and the number of parameters to be estimated, this type of model is of great interest, since it provides better results than a discrete HMM (using Vector Quantization), as far as speaker-independent experiments (requiring the modelling of different speaker characteristics) are concerned. The software used is this study is HmmToolKit (HTK), which is a set of modules developed at Cambridge University, very convenient to handle such models. Segmentation results using continuous HMMs will be compared to discrete HMM-experiments performed previously in ATR. The influence of the number of HMM mixtures on the segmentation results is also investigated.

Two types of HMM training have been carried out. Training with one male speaker has been used for tests on the same male speaker (" close" testing), and on another one ("open" testing). Training with eight male speakers has been applied to test male speaker utterances (close and open), and female speaker utterances.

The segmentation system has also been used in several word-spotting experiments, for accurate segmentation of "Keywords" in spontaneous utterances.

2 HMM theory

2.1 Introduction

All the uncertainties about speech, including different speaking styles and rates, use of non-standard language, have led to model speech in a probabilistic or statistical way. Matching a set of acoustic observations to a given language pattern such as a word, a syllable, a phoneme, can indeed be considered as the result of a probabilistic process.

Moreover, the variability of speech signals with time suggests that a speech-unit model should take into account the evolution of the signal between two states where the signal properties can be considered as steady. This leads to the use of hidden Markov models (HMMs) in speech processing.

The HMM theory is based on Markov processes, in which a set of states with output probabilities representing random events is associated with a transition probability matrix; these two sets of probabilities enable us to model in a satisfactory way the variability in time and in the observation space which characterizes the speech signals. The particularity of HMMs is that the sequence of states is "hidden", that is to say only the output symbols (either discrete or continuous) associated to each hidden state can be known.

We thus assume that the observed signal is a stochastic function of the state sequence in that Markov chain.

2.2 Definition

A HMM is defined by the following parameters [1, 2, 3, 4] :

[1]the smallest semantically independant unit in a Japanese sentence

T: length of the observation sequence $(O_t)_{1\leq t\leq T}$

N: number of states in the model

S: a set of states $\{s_t\}$

V: a set of output symbols

A: a state transition probability matrix; $A = \{a_{ij}|a_{ij} = Pr(s_{t+1} = j|s_t = i)\}$

B: an output probability matrix; $B = \{b_j(O_t)|b_j(O_t) = Pr(O_t|s_t = j)\}$; if the observation sequence consists in symbols from a finite L-sized alphabet, the HMM is said to be "discrete". In that case, B is a matrix $\{b_{ij}\}_{1\leq i\leq N, 1\leq j\leq L}$ where b_{ij} represents the probability that symbol j occurs if the current state is i.

If the observed values can belong to a continuous set, the HMM is said to be "continuous"; in such a case, B is a one-dimensional matrix $\{b_j(\mathrm{x})\}$, where $b_j(\mathrm{x})d\mathrm{x} = Pr(\mathrm{x} \leq O_t \leq \mathrm{x} + d\mathrm{x})$ and x is a d-dimensional observation vector.

π: an initial state distribution. Generally, π is set to (1,0,0,...,0) since the entry state is the first one.

Fig.1 shows a 3-state left-to-right HMM, in which $a_{ij} = 0$ if $i \geq j$.

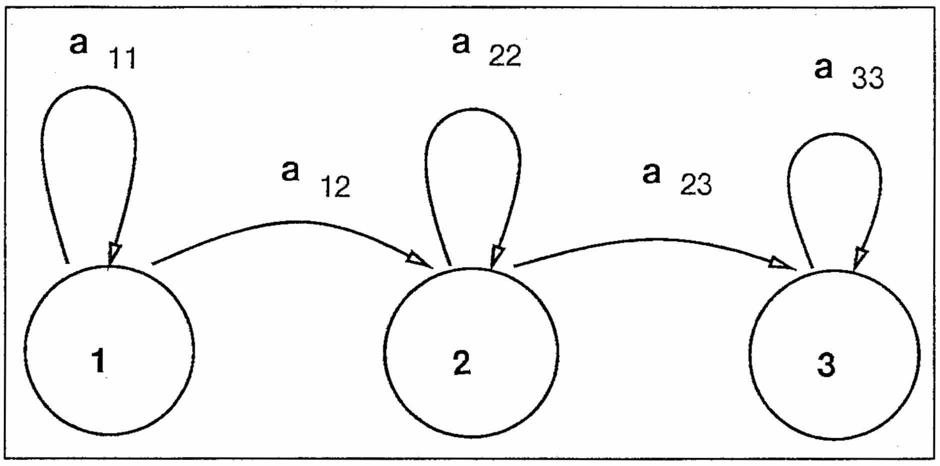


Figure 1: A 3 state left-to-right HMM

We must make two assumptions about the HMMs we will use:

- The transition probability from state i to state j only depends on state i (*Markov assumption*).
- The output probability related to one state only depends on that state, no matter when or how the state is entered (*output-independance assumption*).

We will now refer to λ as the set (A, B, π) which defines the model.

2.3 Basic algorithms

Once the model is defined, three types of problems have to be solved.

2.3.1 The evaluation problem

Given a model λ and an observation sequence (O_t), how can we easily compute $Pr(O|\lambda)$, which is the probability that the model λ will produce these observations ? The model which will eventually be chosen will have to maximise this probability.

If $S = \{s_t\}$ is a possible state sequence, we can write:

$Pr(O|\lambda) = [\sum_{all S} Pr(O|S,\lambda)Pr(S|\lambda)]$.

Moreover,

$$Pr(O|S,\lambda) = [\prod_{t=1}^{T}(b_{s_t}(O_t))], \text{ and } Pr(S|\lambda) = \prod_{t=1}^{T}(a_{s_{t-1}s_t}).$$

Consequently,

$$Pr(O|\lambda) = \sum_{all S} \prod_{t=1}^{T} a_{s_{t-1}s_t} b_{s_t}(O_t) \ .$$

A straightforward calculation shows that such a computation is of the order of $O(N^T)$, which is quite huge. A more efficient algorithm has thus to be found.

The following *Forward-backward algorithm* can reduce that computational cost.

First, we define the forward-variable as:

$\alpha_t(i) = Pr(O_1, O_2, ..., O_t, s_t = i|\lambda)$

which represents the probability of getting the observation sequence $(O_1, ..., O_t)$ and being in state i at time t, given the model λ.

We can then compute $Pr(O|\lambda)$ this way:

- **Step 1 :** $\forall i \in \{1, 2, ...N\}, \alpha_1(i) = \pi_i b_i(O_1)$
- **Step 2 :** $\forall j \in \{1, 2, ...N\}, \forall t \in \{1, 2, ..., T\}, \alpha_t(j) = [\sum_{i=1}^{N} \alpha_{t-1}(i)a_{ij}]b_j(O_t)$
- **Step 3 :** $Pr(O|\lambda) = \sum_{i \in S_F} \alpha_T(i)$ where S_F is the set of all possible final states.

The computational cost is now reduced to the order of $O(N^T)$.

We can easily consider a backward-variable $\beta_t(i) = Pr(O_{t+1}, O_{t+2}, ..., O_t|s_t = i, \lambda)$, which would represent the probability of getting the sequence $(O_{t+1}, O_{t+2}, ..., O_T)$, given the state i at time t and the model λ. A similar algorithm can be processed to compute $Pr(O|\lambda)$.

2.3.2 The Estimation Problem

The parameters A, B, π of the model λ have then to be adjusted in order to maximise $Pr(O|\lambda)$. The algorithm which is ordinarily used on that purpose is the Baum-Welch re-estimation algorithm. Assuming that the initial parameters can be chosen randomly, or judiciously guessed, this algorithm solves the HMM-training problem.

Considering the previous notations, we can define $\gamma_t(i,j) = Pr(s_t = i, s_{t+1} = j|O,\lambda)$ as the probability of being in state i at time t and in state j at time $t+1$, given the observation sequence O and the model λ (γ_{ij} is in fact an *a posteriori* transition probability from state i to state j.)

Thus,

$\gamma_t(i,j) = \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{Pr(O|\lambda)} = \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{\sum_{k \in S_F} \alpha_T(k)}$

We can also define an *a posteriori* probability of being in state i at time t, given O andλ.

$\gamma_t(i) = Pr(s_t = \text{i}|O,\lambda) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{k \in S_F} \alpha_T(k)}$

We can notice that $\gamma_t(i) = \sum_j \gamma_t(i,j)$.

Moreover, a_{ij} is the general transition probability from state i to state j, no matter the time those states are reached. Therefore, an estimate of a_{ij} can be:

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \gamma_t(i,j)}{\sum_{t=1}^{T-1} \sum_j \gamma_t(i,j)} = \frac{\sum_{t=1}^{T-1} \gamma_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}.$$

Besides, the coefficient of the matrix $\{B\}$ (in case of a dicrete HMM, for instance) $b_j(k)$ represents the probability of observing the symbol $v_k \in V$, while in state j. From the training data, we can consider as an estimate of $b_j(k)$ the frequency of occurrence of v_k relative to the frequency of occurrence of any symbol while in state j.

Thus, an estimate of $b_j(k)$ is:

$$\bar{b}_j(k) = \frac{\sum_{t \in O_t = v_k} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}.$$

Eventually, an estimate of the initial state probability can be given by $\bar{\pi}_i = \gamma_1(i)$.
We can show that:
- either the initial model was the optimal one, in which case the estimates would be equal to the initial probabilities;
- either the replacement of λ by $\bar{\lambda} = (\bar{A}, \bar{B}, \bar{\pi})$ increases the probability $Pr(O|\lambda)$.

2.3.3 The decoding problem

We want to know the best state sequence $S = s_1 s_2 ... s_T$ according to the observation sequence. In other words, we have to choose the states which maximize $Pr(O, S|\lambda)$. It can be a means of interpreting the hidden state sequence of the HMM. One famous method for that purpose is the *Viterbi algorithm*. Here are the outlines of this algorithm:

Step 1: Initialisation. $\forall i \in \{1, 2, ..., N\}$,
$\delta_t(i) = \pi_i b_i(O_1)$
$\psi_1(i) = 0;$

Step 2: Recursion. $\forall t \in \{2, ..., T\}, \forall j \in \{1, 2, ..., N\}$,
$\delta_t(j) = \max_i[\delta_{t-1}(i)a_{ij}]b_j(O_t)$
$\psi_t(j) = \arg\max_i[\delta_{t-1}(i)a_{ij}]$

Step 3: Termination. $\bar{P} = \max_{s \in S_F}[\delta_T(s)]$
$\bar{s}_T = \arg\max_{s \in S_F}[\delta_T(s)]$
where $\bar{P}$ and $\bar{s}$ represent the optimized values.

Step 4: Path backtracking. for $t = T - 1$ to $t = 1$, $\bar{s}_t = \psi_{t+1}(\bar{s}_{t+1})$

This algorithm is used in state segmentation.

2.4 Continuous HMMs

The previous explanations considered a discrete-type HMM, that is to say the observation was a symbol from a L-sized finite alphabet. But all those algorithms can be adapted to the continuous HMM, which has been used in this study. In that case, the observation $\mathbf{x}$ can be any point of a d-dimensional vector space to which we assign an occurrence probability. We thus define output probability functions.

Let $\mathbf{X}$ be the observation sequence from the continuous set. The goal is then to maximize $f(\mathbf{X}|\lambda)$ over all the parameters of the model λ.
By using the same kind of formulas as in the discrete case:

$$f(\mathbf{X}|\lambda) = \sum_{all S} f(\mathbf{X}, S|\lambda) = \sum_{all S} \prod_{t=1}^{T} a_{s_{t-1}s_t} b_{s_t}(\mathbf{x}_t).$$

Using continuous HMMs enables to get rid of Vector Quantization; matching an observation to the closest element of a finite codebook according to a given distance indeed implied some quantization errors, which were one drawback of discrete HMMs. The use of continuous models can then be assumed to improve the accuracy of the method.

If the vectors are scattered in the observation space, it can be necessary to define several density probabilities and to consider the global density function as a summation of M densities which are assigned some weights. We thus have a mixture density HMM, for each state j of which we can write:
$b_j(\mathbf{x}_t) = \sum_{k=1}^{M} c_{jk} b_{jk}(\mathbf{x}_t)$.
c_{jk} is the weight of mixture k in state j.
The c_{jk} coefficients must verify the essential condition
$\sum_{k=1}^{M} c_{jk} = 1$,

in which case
$\int_R b_j(\mathrm{x})d\mathrm{x} = 1$,
for $b_j(\mathrm{x})$ is a density probability.

Therefore,
$f(X,S|\lambda) = \prod_{t=1}^{T} a_{s_{t-1}s_t} b_{s_t}(\mathrm{x}_t)$

Several kinds of density probabilities can be used; nevertheless, the most common ones are the Gaussian distributions. As a matter of fact, from the *central limit theorem* , we know that the probability density function of a sum of independant random variables tend to a Gaussian distribution if the number of these variables tends to infinity. We can thus consider as a Gaussian distribution the function $b_j(x)$ if the number of mixtures is large enough.

Using a large number of mixtures can improve the accuracy of the model (especially if the training vectors are sparse in the vector space), but also requires a great amount of training data; otherwise, all the parameters of the model cannot be well-estimated.

In order to reduce the computational complexity of such a model and the number of free parameters to be estimated, we have to make some assumptions about the mixtures; for example, we can suppose that the random variables represented by each mixture are independant. In such a case, the covariance matrix of these variables is diagonal. Of course, on the other hand, this simplification can reduce the accuracy of the model.

The training problem will be to find a balance between that accuracy and the simplicity.

3 The segmentation system

3.1 Introduction

The main goal of this segmentation system is to build automatically a large speech database. The system will have to determine with the greatest accuracy as possible the boundaries of already known labels.

3.2 HMM theory applied to phoneme segmentation

We have seen that hidden Markov modelling was very convenient to treat speech signals. Each predefined speech unit (word, phoneme, allophone[2]) is represented by an HMM which will store its features.

The elementary speech units which are to be modelled are phonemes, since labelling is performed with phoneme units.

Here, we use a list of 48 phoneme models, including silence (model "+") at the beginning and the end of the utterances. These phonemes can be clustered in some classes:

vowels: a aa e ee i ii o oo u uu N

voiced consonants: b b0 d d0 g g0 m m0 n n0 r r0 z z0

unvoiced consonants: ch ch0 k k0 p p0 s sh ss ssh t t0 ts ts0

silence: +

pause: Q

other phonemes: h w w0 y y0 Ui Uu

This list takes into account the short vowels (ex: "a") and the long ones (ex: "aa") in the Japanese language. The "N"model represents the nasal sound, like in "d-e-N-w-a" for example. The symbol "0" following a consonant model indicates that this consonant was uttered at the beginning of a word utterance or after a word-medial pause, denoted by "Q", like in "m-o-Q-t0-o", transcribed in Latin alphabet as "motto". The utterances are indeed different after a pause or inside a word, where coarticulation effects can appear. "Ui" and "Uu" are respectively unvoiced "i" and unvoiced "u"; the "i" and "u" sounds are generally unvoiced when between two unvoiced consonants, like in "d0-e-k-Ui-t-a", trancribed as "dekita";

[2]environment-dependent phoneme

the "u"sound is also often unvoiced at the end of a word, like in "g0-o-z-a-i-m-a-s-Uu", transcribed as "gozaimasu".

48 phoneme HMMs have then to be trained, using phoneme labels from the label files; but those labels do not distinguish "Ui" and "i", or "k0" and "k", for example. An automatic label conversion program has then to be applied to the original labels, in order to produce the final ones.

3.3 Speech coding

The original speech data come from words, bunsetsu and sentences uttered by several male or female speakers and sampled at the rate of 12 kHz.

The speech data must be analysed and coded in order to train the HMMs. The result of this speech analysis is the observation sequence mentioned in the previous section.
The speech samples $\{s_i\}_{1\leq i\leq I}$, if I is the length of the sample sequence, are first pre-emphasized to a sequence $\{s'_n\}$ using the formula:

$$s'_i = s_i - ks_{i-1}$$

In the z-transform domain, this is equivalent to applying a filter, the transfer function of which is $H(z) = 1 - kz^{-1}$. In the following experiments, k will be set to 0.97.
A Hamming window is then applied to the sequence in the time domain, following the formula:

$$s''_i = 0.54 - 0.46\cos(\frac{2\pi i}{I-1})s'_n$$

The basic goal of this analysis is to get a sequence of overlapping frames defined by those Hamming windows. The window is applied every 5 ms, and its length is set to 25.6 ms in the following experiments.

Then we perform the Linear Predictive Coding (LPC) [5], and the computation of the LPC Cepstra. Assuming that, if $\{x_n\}$ is the time sequence, and $X(z)$ its z-transform,

$$X(z) = \frac{1}{1+\sum_{n=1}^{p}\alpha_n z^{-n}}$$

where $\{\alpha_i\}_{1\leq i\leq p}$ are the p-order linear predictive coefficients, computed with the autocorrelation method.

With defining the cepstral coefficients $\{c_i\}$ by:

$$\sum_{n=0}^{\infty} c_n z^{-n} = C(z) = \log[X(z)]$$

It can be shown that

$$c_n = -a_n + \frac{1}{n}\sum_{i=1}^{n-1}(n-i)a_i c_{n-i}$$

These coefficients, which can be recursively computed from the LPC coefficients, are widely used in speech processing.

For each frame (every 5 ms), p coefficients are computed. We also calculate the difference coefficients (delta-cepstra), according to a $2N$-sized window, using the formula:

$$d_t = \frac{\sum_{\tau=1}^{N}\tau(c_{t+\tau}-c_{t-\tau})}{2\sum_{\tau=1}^{N}\tau^2},$$

c_t being the cepstral coefficient at time t. Here, N is set to 6 ms.

The energy $\sum_n x_n^2$ for n in the Hamming window and the corresponding difference coefficient (in fact, their logarithms) are also added to these elements.

The observation vector will therefore be composed of 26 coefficients. To each speech file will correspond a feature file containing those vectors.

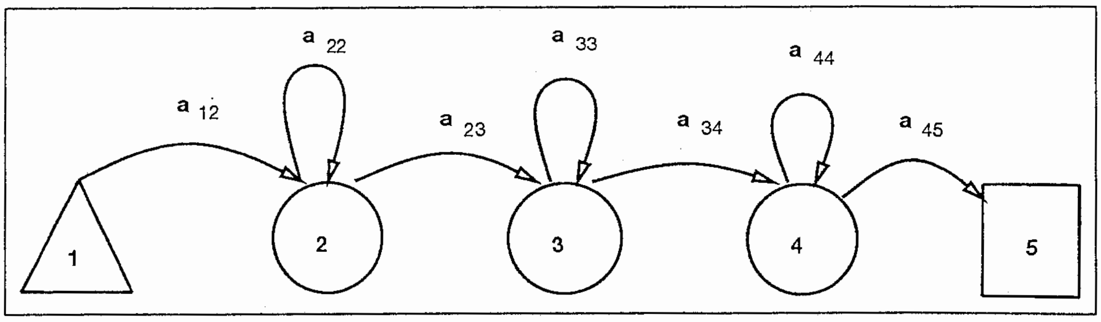


Figure 2: The model

3.4 HMMToolKit

HMMToolKit (HTK) [6] is a set of modules and programs developed at Cambridge University which handle continuous mixture density HMMs with any number of states and mixtures, any observation vector size, full or diagonal covariance matrices.

3.5 The model

There is no means of knowing in advance either the optimal number of states or the optimal number of mixtures a HMM has to contain. Therfore, experience can only give an idea of these numbers to use. All the phoneme models in these experiments will be 3 state- models, but HTK requires an entry state and an exit state to build an HMM. The model is thus the one shown on Fig.2.

3.6 Training

The HMMs are trained using feature files (outputs of the HTK program named "HCode") and label files which give, for each label, its start point and end point determined by hand. These files have to be changed into HTK format, as follows:

```
2550000   3550000      +
3550000   3900000      h
3900000   5600000      a
5600000   6650000      i
6650000   7650000      +
```

where each label (model) is preceded by its start and end points in hundreds of ns.
The HTK training module named "HInit" reads all the label and feature files used for training, and matches one label to its corresponding part in the feature file. This initialises the parameters of each model.

A "proto-HMM" which a priori random initial values for means and variances of the observation vectors, the weights of each mixture, and the transition probability matrix has first to be written. The accuracy of the final model depends on the parameters before Baum-Welch iterative reestimation, and especially the means of each vector; the purpose of "HInit" is thus to provide good initial estimates of these means.
After mapping a label to a temporal part of the feature file, the program has to segment each observation of this label into the required number of states, using the Viterbi alignment. Each observation vector corresponding to each state has to be clustered into a set of M clusters, if M is the required number of mixtures of the probability density function. For that purpose, a VQ algorithm is iteratively applied.
The mixture weights previously denoted as c_{jk} are in fact the number of vectors pooled in mixture k in state j divided by the number of vectors in state j.

This procedure computes a new mean matrix, a new covariance matrix, a new output probability matrix (B), but keeps unchanged the old transition probability matrix (A).

A limited number of training data (e.g. 1000 occurrences of the same label in the training files) can be enough for this initialisation.

The iterative reestimation procedure ("HRest"), implementing Baum-Welch algorithm formulas, gives new estimates of A and B (weights, means and covariance matrices); the new model $\bar{\lambda}$ enables to compute $Pr(O|\bar{\lambda})$, to compare it to $Pr(O|\lambda)$ and thus to decide either to stop the procedure (in which case an optimal model has been found) or to iterate once more. The largest number of data as possible must be used in the reestimation part.

3.7 Segmentation

The segmentation is performed on testing speech files coded into feature files, and produces label files with the same format as those used during the training part. It is achieved using "HVite" program, which is a Viterbi recognizer (also used for recognition) given a deterministic grammar. As a matter of fact, in segmentation problem, we already know which phonemes were uttered and in which order; the grammar needed by HVite is thus the sequence of phonemes corresponding to each speech file. The goal will then be to compare the original ("hand"-determined) start points and end points of each phoneme to those determined automatically by the program.

3.8 Evaluation

The outputs of HVite program will be compared to the original label files, considered as the reference points. For each start point and end point, the difference between the automatic boundary and the "hand"-determined one will be the segmentation error. The segmentation rate will represent, for each model, the percentage of boundaries for which this error is smaller than 50, 30, 10 ms respectively. This evaluation has some drawbacks: first, the figures 50, 30, 10 ms may seem arbitrary; then, it does not take into account the average length of the phonemes (a vowel is obviously longer than a consonant, for instance) and uses the same measurements for all phonemes. In spite of those drawbacks, this method enables the comparison with other methods. The probability density function of error will also be evaluated, with the computation of the mean error, the mean absolute error, and the standard deviation.

4 Experiments

4.1 Main questions

The main points we have to focus on are the following ones:

- Is the segmentation rate dependent on the kind of tested data, that is to say: are, for instance, words "better-segmented"than sentences, in some sense? Therefore, all the different experiments will be applied to test words, phrases and sentences.
- How can the training conditions influence the segmentation results for one given speaker and upon the difference between two speakers? Single and multiple speaker training will thus be used. The influence of the number of mixtures will also be investigated.
- How robust is the segmentation system to different speakers? In which case can we consider a model as "speaker-dependent", or rather "speaker-independent"? Tests will be performed on both male and female speakers, in both "close" and "open" speaker cases.
- Are some phonemes better segmented than others? Is it possible to find a general trend for some phonemes or some phoneme classes?

Therefore, the following experiments were performed:

Testing conditions

Training conditions

		MAU	MNM	MHT	MTK	FSU
single		C	O			
multiple	3 mixt.	O	O	C	C	O
	9 mixt.	O	O			

Single speaker training: MAU (2620 words)

Multiple speaker training: MHT MMS MMY MSH
MTK MTM MTT MXM
(8 * 1310 words)

"C": close speaker

"O": open speaker

tests on words, phrases (sets DSA and DSB), and sentences (set DSC).

4.2 Single speaker training

The first experiments used 3 mixture-HMMs trained with 2620 word-utterances from one male speaker (speaker MAU).

4.2.1 Close speaker testing

The segmentation was first performed on the same speakers' utterances.

Close context The 2620 training words were tested, and the phoneme boundaries were compared to the hand-determined boundaries.

The segmentation results have been gathered into phoneme classes: vowels, voiced consonants (denoted as "Vcons"), unvoiced consonants (denoted as "Ucons"), and in "all" class, including all the models except the silence (+) model. These results distinguish start point comparisons and end point comparisons. The "segmentation rate" is the percentage of boundaries correctly determined within a given error interval from the "hand"-boundaries.

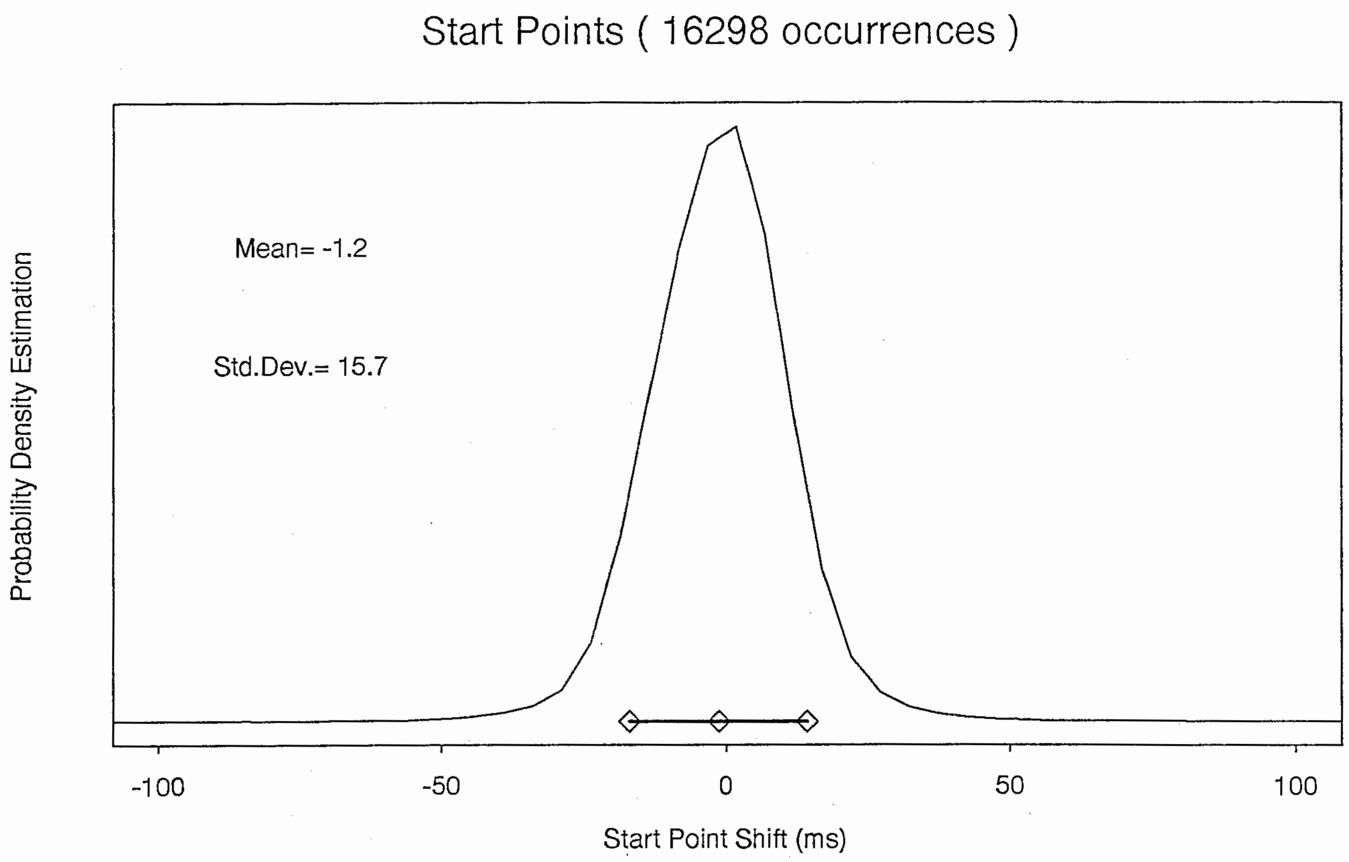
Model class	Nbr. of occ.	Segmentation rate (%) (Error $\leq$)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	6766	99.4	98.0	77.8	3.7	8.6	16.2
Vcons	2349	99.6	98.7	68.7	-8.2	10.1	19.9
Ucons	3205	99.9	98.2	61.2	-5.7	10.4	11.8
all	16298	99.6	98.2	73.6	-1.2	8.9	15.7

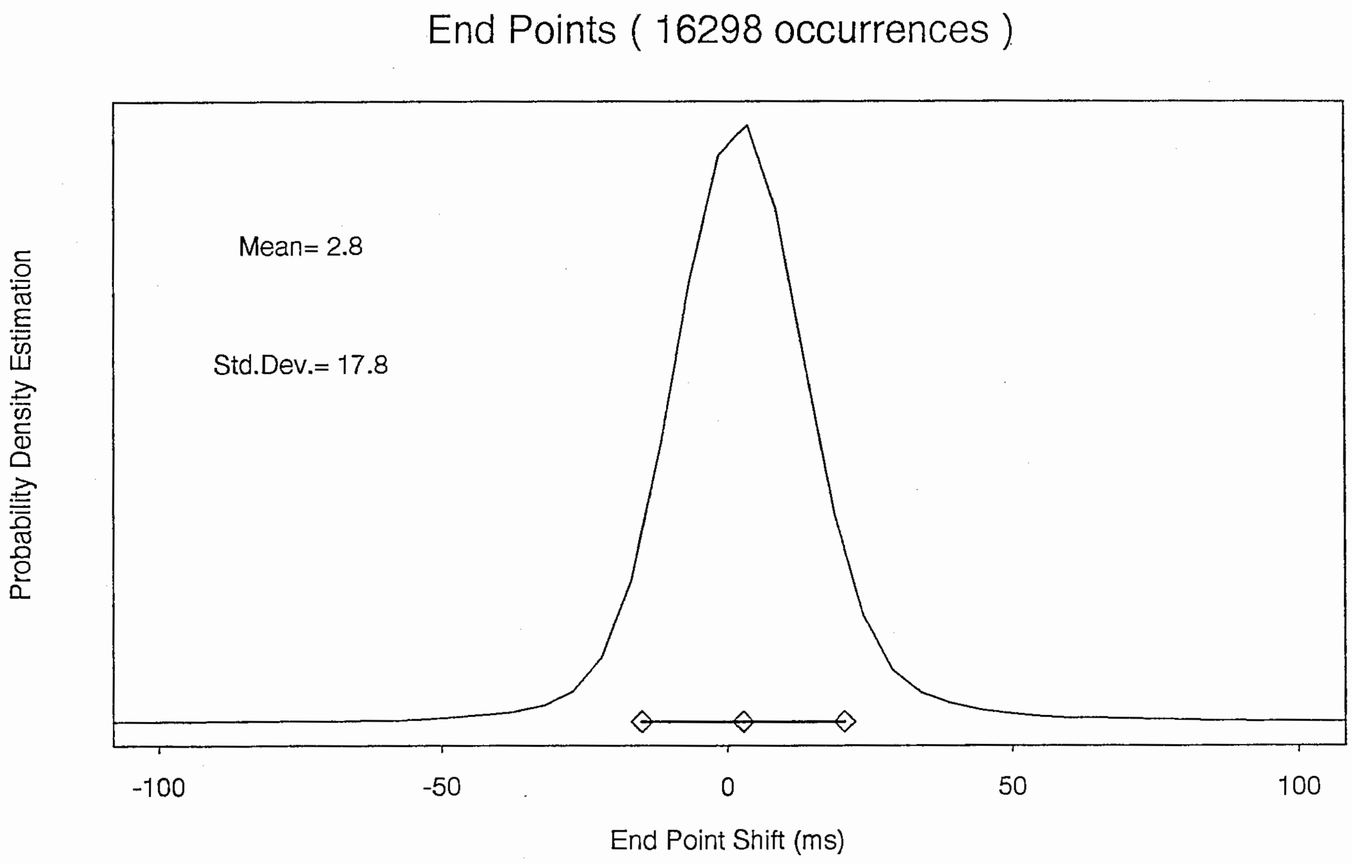
Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error $\leq$)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	6681	98.4	95.6	70.4	-0.1	10.7	20.7
Vcons	3067	99.2	96.9	70.4	7.1	10.3	20.5
Ucons	2763	100.0	100.0	93.2	1.6	5.1	6.3
all	16298	99.1	96.6	72.0	2.8	9.8	17.8

End points

The figure below shows the distribution of the boundary errors, that is to say the difference between the automatically found boundaries and the "hand"-determined ones.


Start Points (16298 occurrences)
Mean= -1.2
Std.Dev.= 15.7
Probability Density Estimation
-100
-50
0
50
100
Start Point Shift (ms)


End Points (16298 occurrences)
Mean= 2.8
Std.Dev.= 17.8
Probability Density Estimation
-100
-50
0
50
100
End Point Shift (ms)

Open context The next tests were performed on utterances different from the ones used for testing.

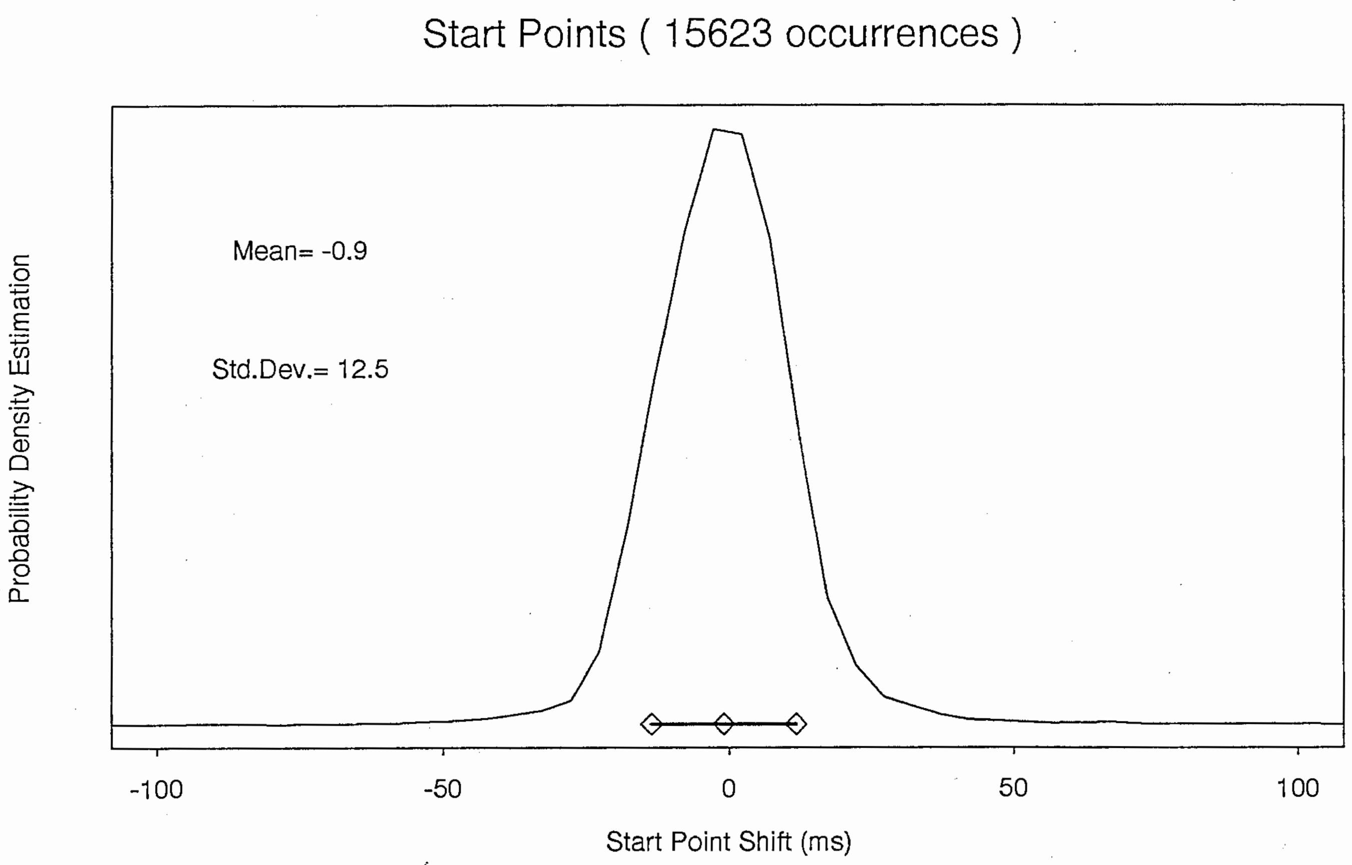
words

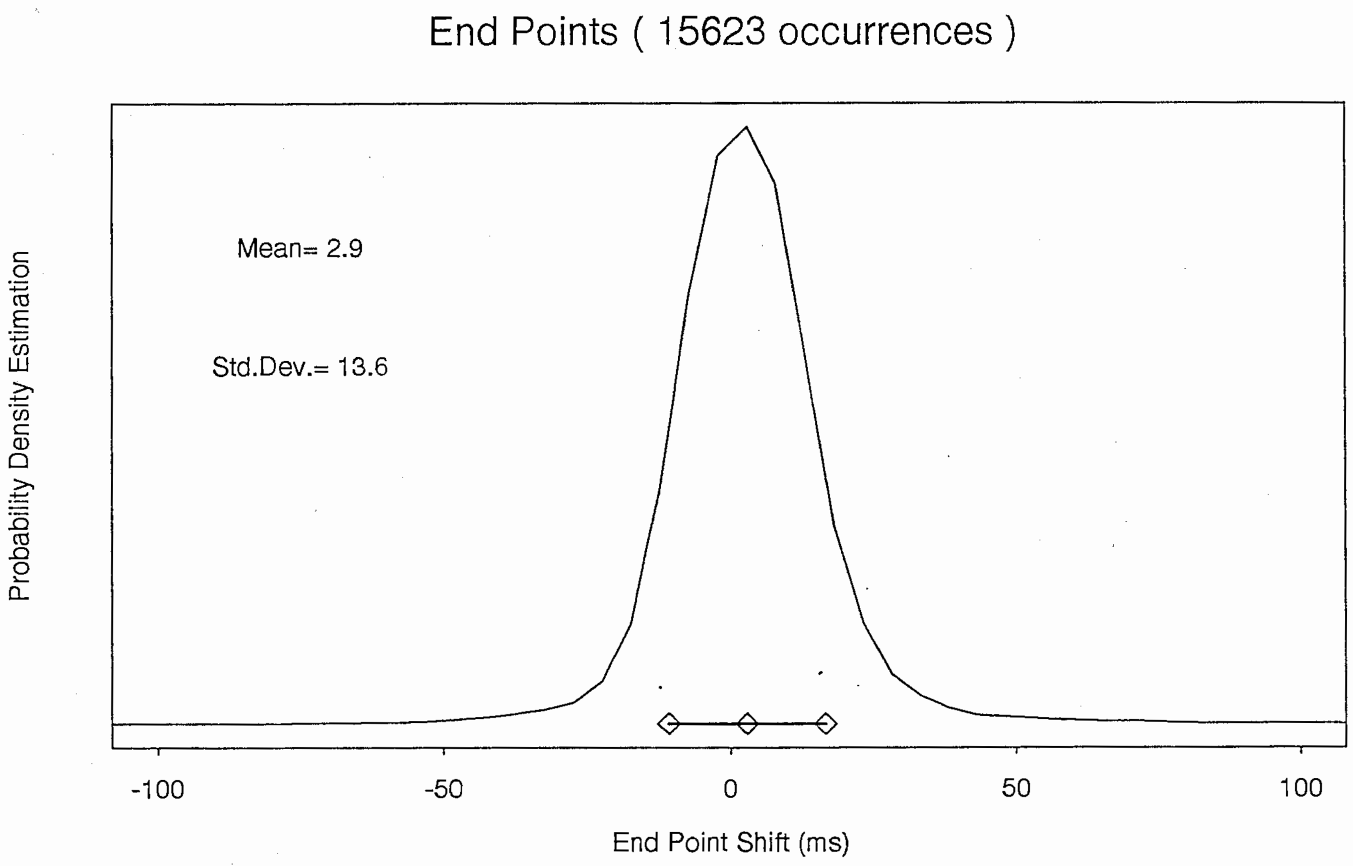
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	6747	99.5	97.8	77.7	4.0	8.4	11.5
Vcons	2495	99.7	98.5	67.6	-7.2	9.9	11.3
Ucons	3127	99.9	98.0	59.9	-5.2	12.2	10.7
all	15623	99.6	97.9	72.4	-0.9	9.1	12.5

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	6653	98.8	95.8	69.3	0.5	10.6	15.5
Vcons	2551	99.2	97.2	72.3	7.2	9.5	11.9
Ucons	2972	100.0	100.0	92.2	1.6	5.2	6.7
all	15623	99.2	96.8	72.0	2.9	9.6	13.6

End points


Start Points (15623 occurrences)
Mean= -0.9
Std.Dev.= 12.5
Probability Density Estimation
-100
-50
0
50
100
Start Point Shift (ms)


End Points (15623 occurrences)
Mean= 2.9
Std.Dev.= 13.6
Probability Density Estimation
-100
-50
0
50
100
End Point Shift (ms)

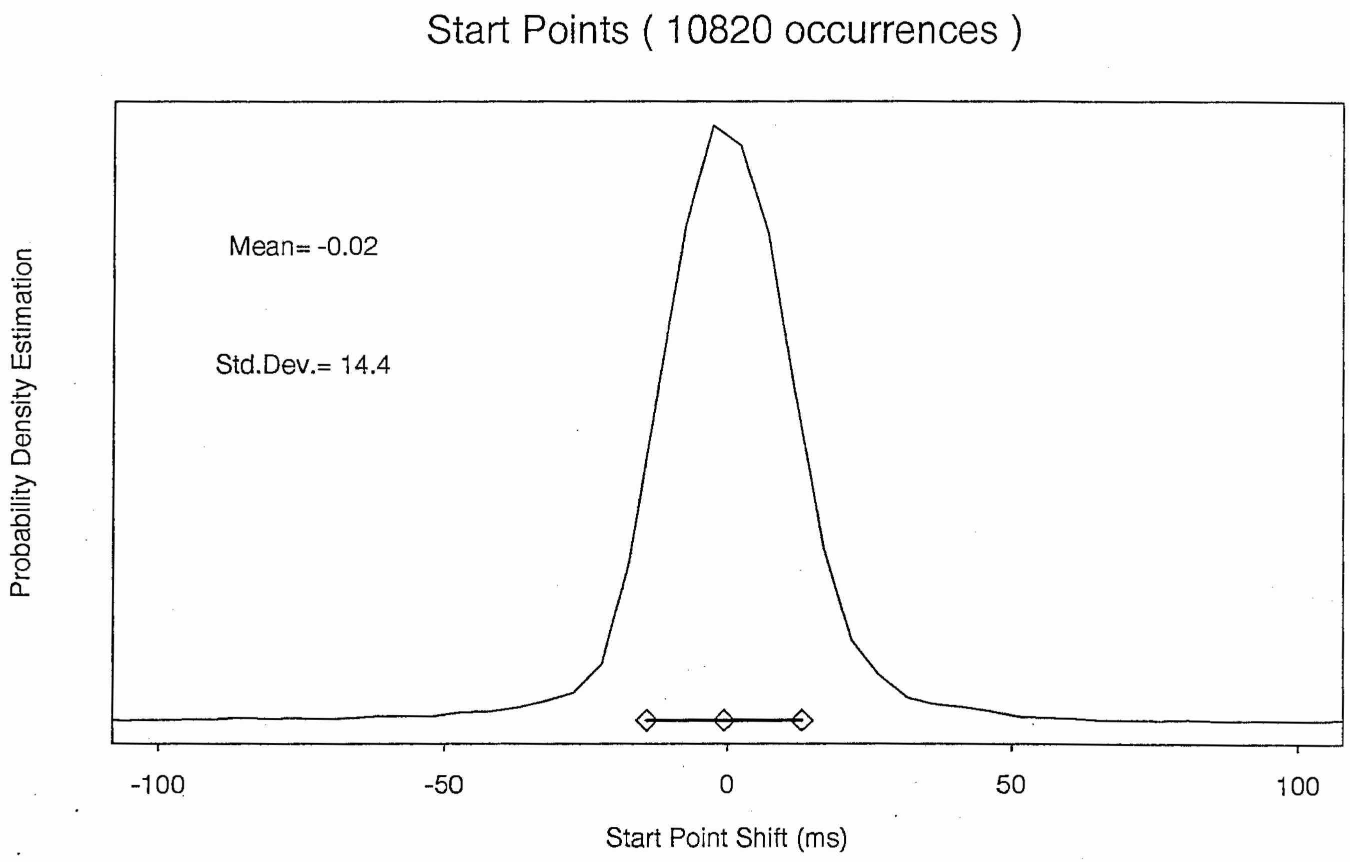
phrases

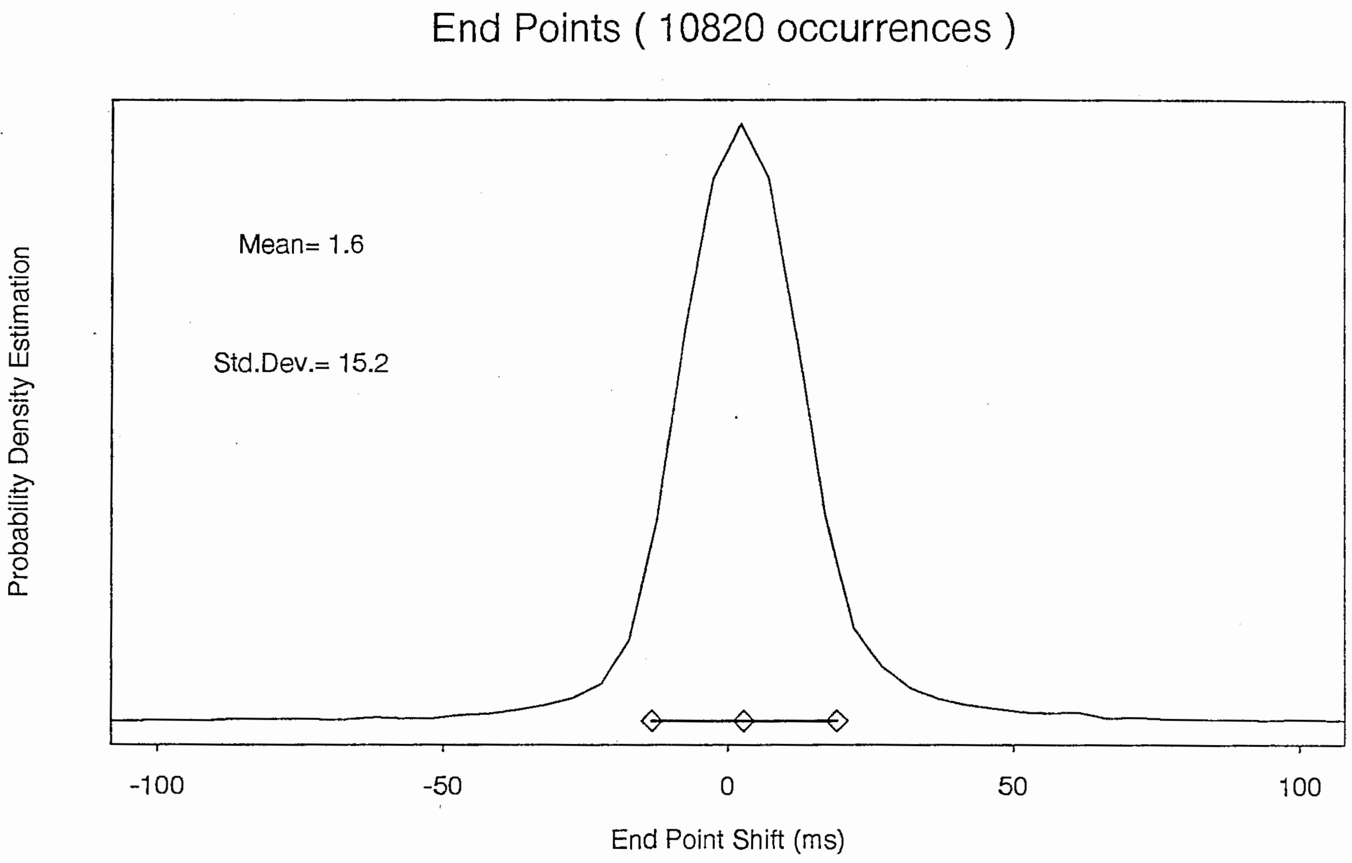
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	5456	99.3	97.0	75.4	3.3	8.9	13.6
Vcons	2068	98.8	96.7	65.2	-4.1	11.1	16.2
Ucons	2336	99.8	98.7	72.1	-4.5	8.8	11.1
all	10820	99.3	97.1	71.9	0.0	9.4	14.4

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	5386	98.8	95.6	65.1	0.2	11.0	16.4
Vcons	2147	98.6	95.0	69.0	6.3	10.7	16.2
Ucons	2017	99.9	99.9	90.6	1.0	5.5	8.8
all	10820	99.1	96.5	70.5	1.6	9.8	15.2

End points


Start Points (10820 occurrences)
Probability Density Estimation
Mean= -0.02
Std.Dev.= 14.4
-100
-50
0
50
100
Start Point Shift (ms)


End Points (10820 occurrences)
Probability Density Estimation
Mean= 1.6
Std.Dev.= 15.2
-100
-50
0
50
100
End Point Shift (ms)

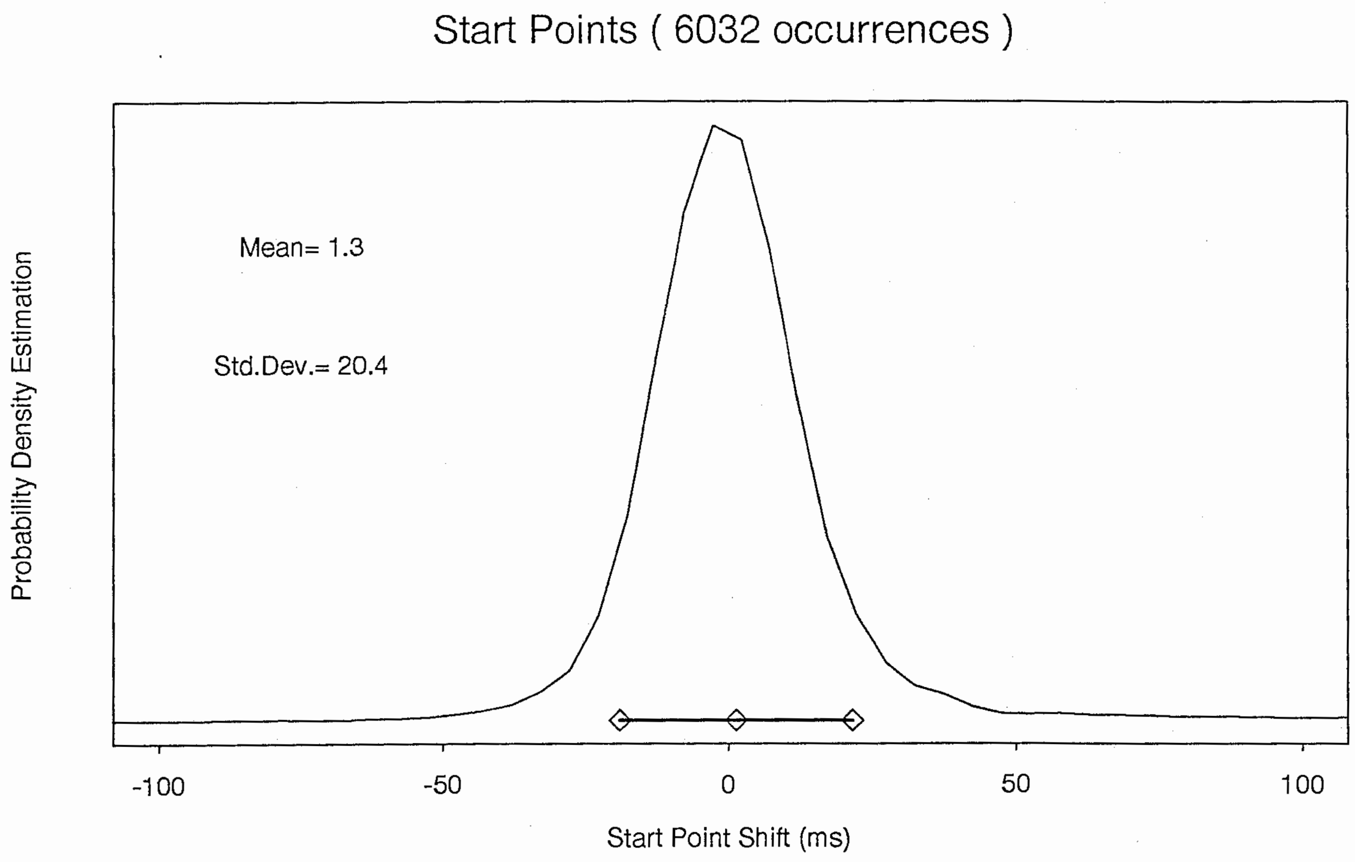
sentences

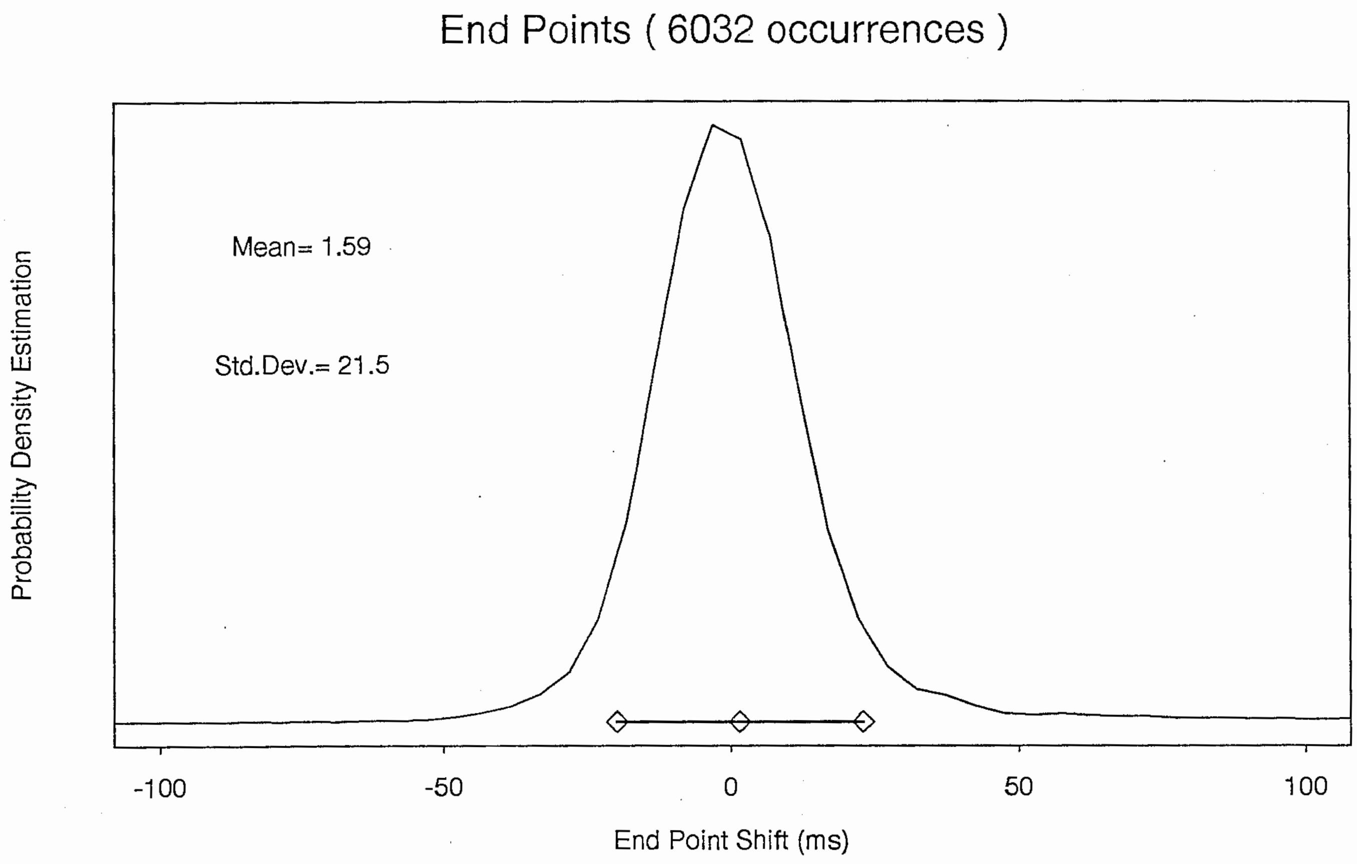
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3078	97.9	95.4	68.8	5.4	11.2	19.6
Vcons	1184	97.6	93.0	55.6	-5.1	13.7	19.8
Ucons	1247	98.2	95.2	62.9	-4.7	11.9	18.8
all	6032	97.9	94.7	64.6	1.3	12.0	20.4

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3029	97.4	93.9	58.9	-1.2	13.6	23.7
Vcons	1224	97.2	93.1	66.2	7.6	12.0	18.0
Ucons	1145	98.7	98.2	81.1	3.9	8.3	17.3
all	6032	97.8	94.6	64.6	1.6	12.2	21.5

End points


Start Points (6032 occurrences)
Mean= 1.3
Std.Dev.= 20.4
Probability Density Estimation
-100
-50
0
50
100
Start Point Shift (ms)


End Points (6032 occurrences)
Mean= 1.59
Std.Dev.= 21.5
Probability Density Estimation
-100
-50
0
50
100
End Point Shift (ms)

4.2.2 Open speaker testing

The experiments are now performed on speaker MNM (open speaker).

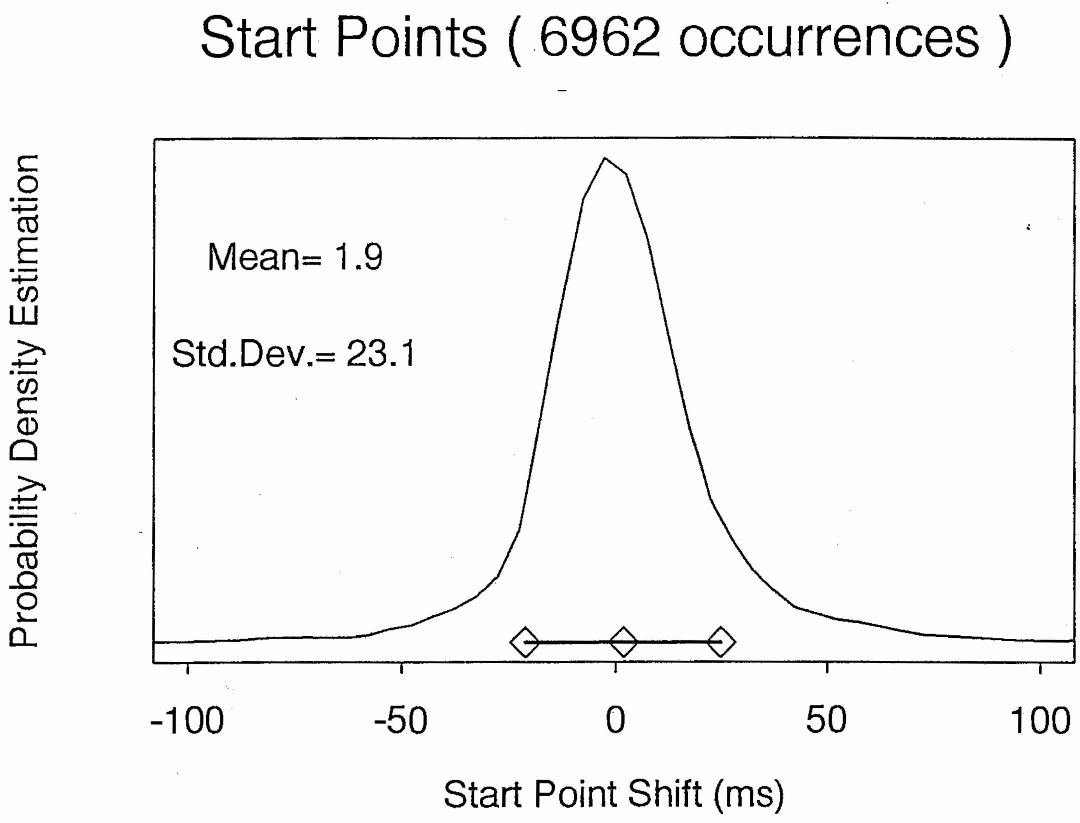
words

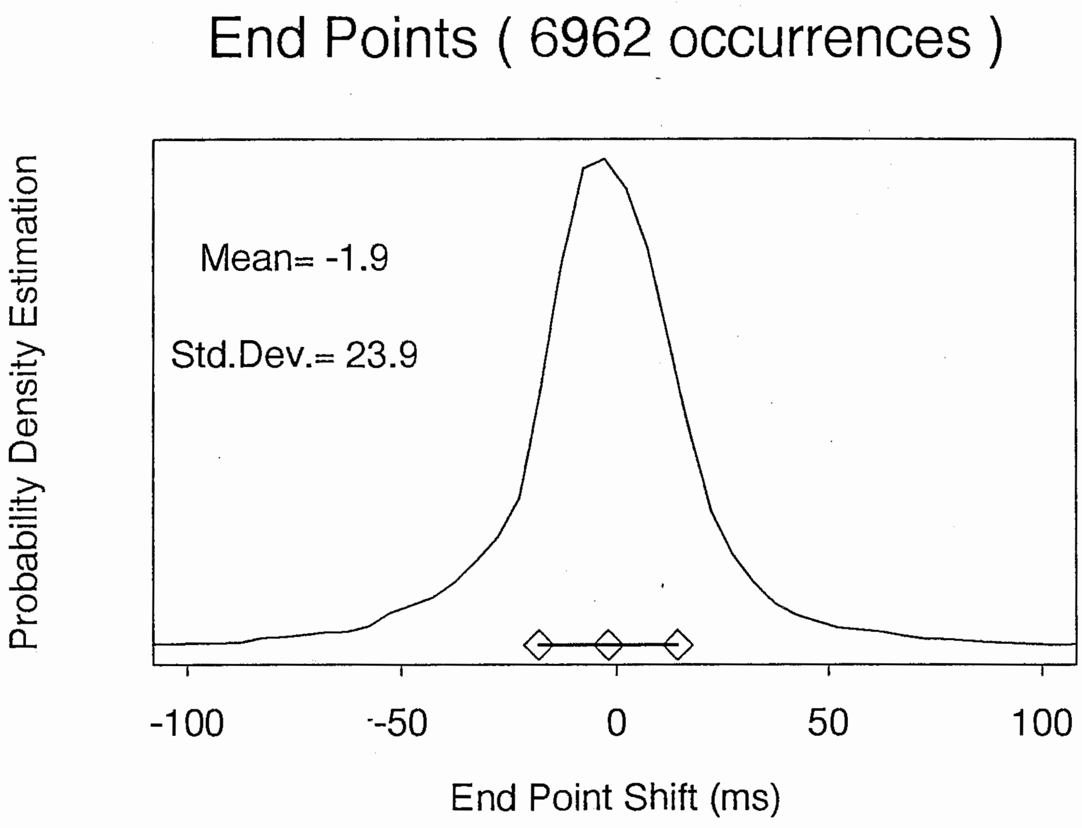
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3364	96.5	91.8	59.1	2.7	14.1	21.7
Vcons	1272	94.0	85.1	50.6	3.8	18.0	26.2
Ucons	1498	96.5	85.5	49.7	-2.6	17.1	24.7
all	6962	96.1	89.0	56.1	1.9	15.4	23.1

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	2761	95.0	82.8	46.9	-4.6	18.9	26.3
Vcons	1307	93.1	84.7	51.0	5.4	18.0	26.7
Ucons	1416	99.8	99.2	74.4	0.8	8.8	13.4
all	6962	95.7	87.3	53.8	-1.9	16.3	23.9

End points


Start Points (6962 occurrences)
Mean= 1.9
Std.Dev.= 23.1
Probability Density Estimation
-100
-50
0
50
100
Start Point Shift (ms)


End Points (6962 occurrences)
Mean= -1.9
Std.Dev.= 23.9
Probability Density Estimation
-100
-50
0
50
100
End Point Shift (ms)

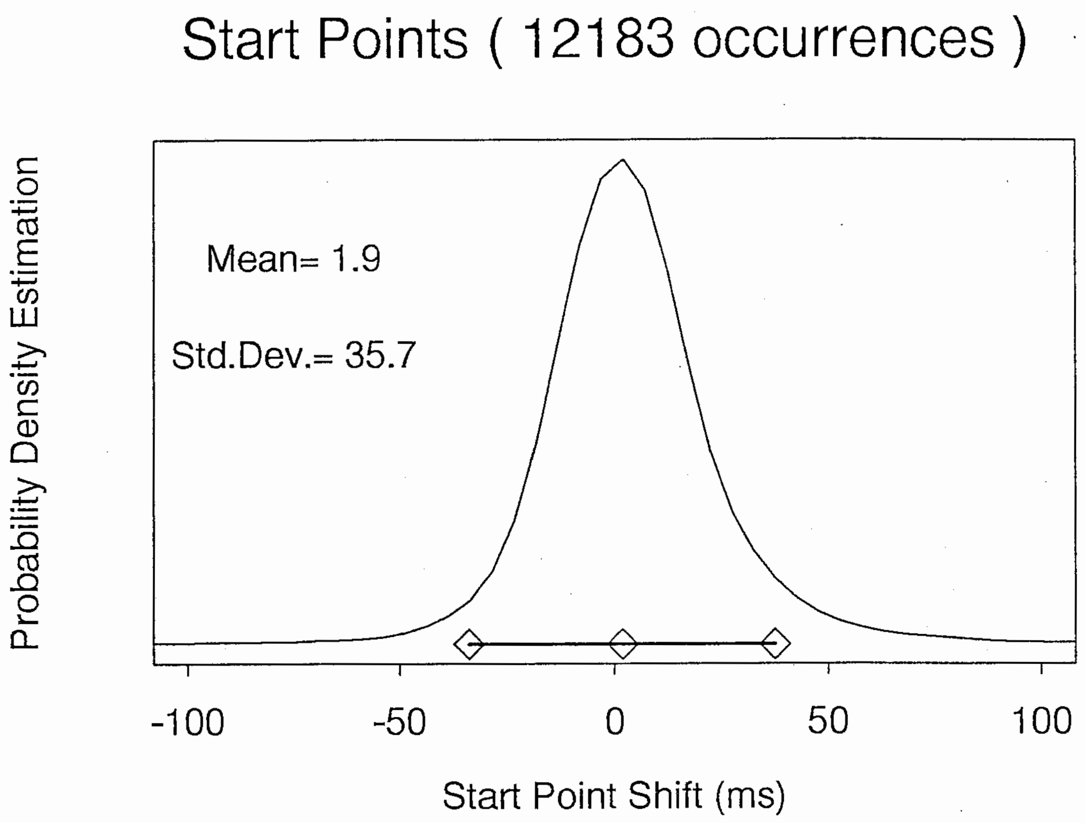
phrases

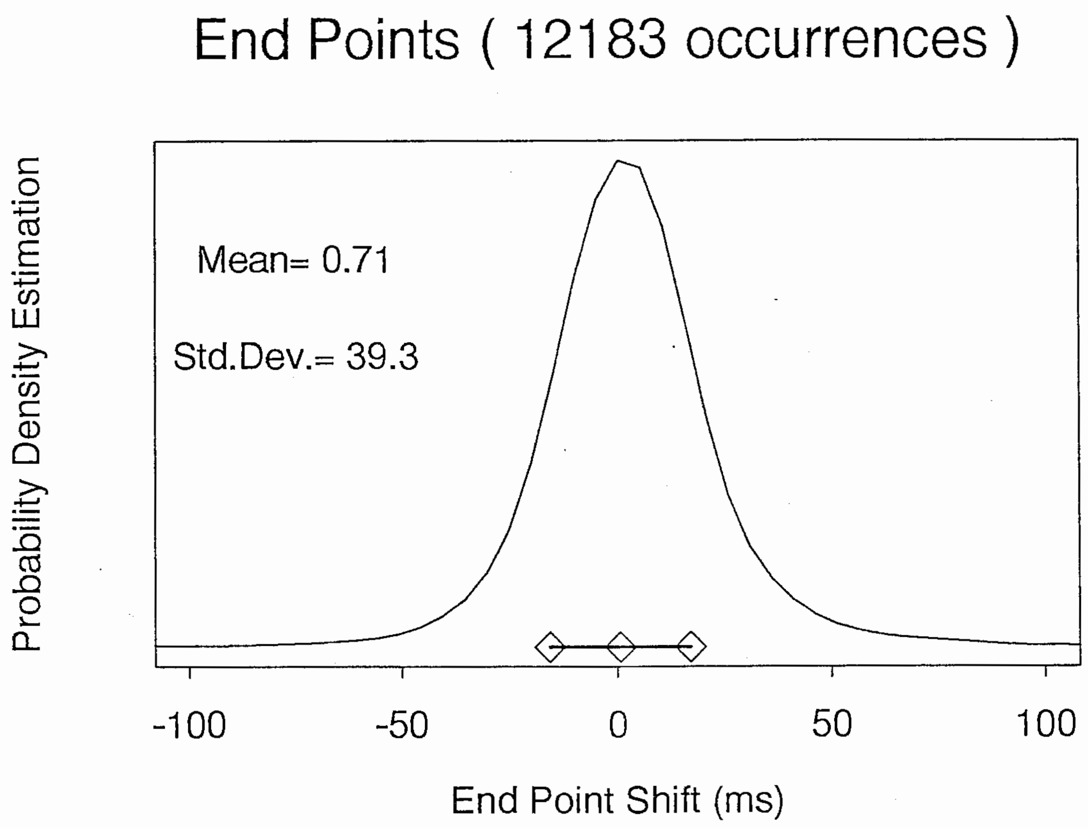
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	6252	97.1	92.3	60.8	3.4	15.0	33.6
Vcons	2324	95.2	87.9	57.4	2.9	17.3	37.3
Ucons	2495	96.7	87.3	47.5	-4.5	18.4	39.8
all	12183	96.3	89.6	56.8	1.9	16.5	35.8

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	6158	95.0	86.0	50.4	-0.4	19.1	41.1
Vcons	2471	96.0	89.6	57.2	2.2	16.9	38.6
Ucons	2252	99.5	98.8	75.8	2.2	10.0	29.0
all	12183	96.4	89.7	56.7	0.7	16.7	39.3

End points


Start Points (12183 occurrences)
Probability Density Estimation
Mean= 1.9
Std.Dev.= 35.7
-100
-50
0
50
100
Start Point Shift (ms)


End Points (12183 occurrences)
Probability Density Estimation
Mean= 0.71
Std.Dev.= 39.3
-100
-50
0
50
100
End Point Shift (ms)

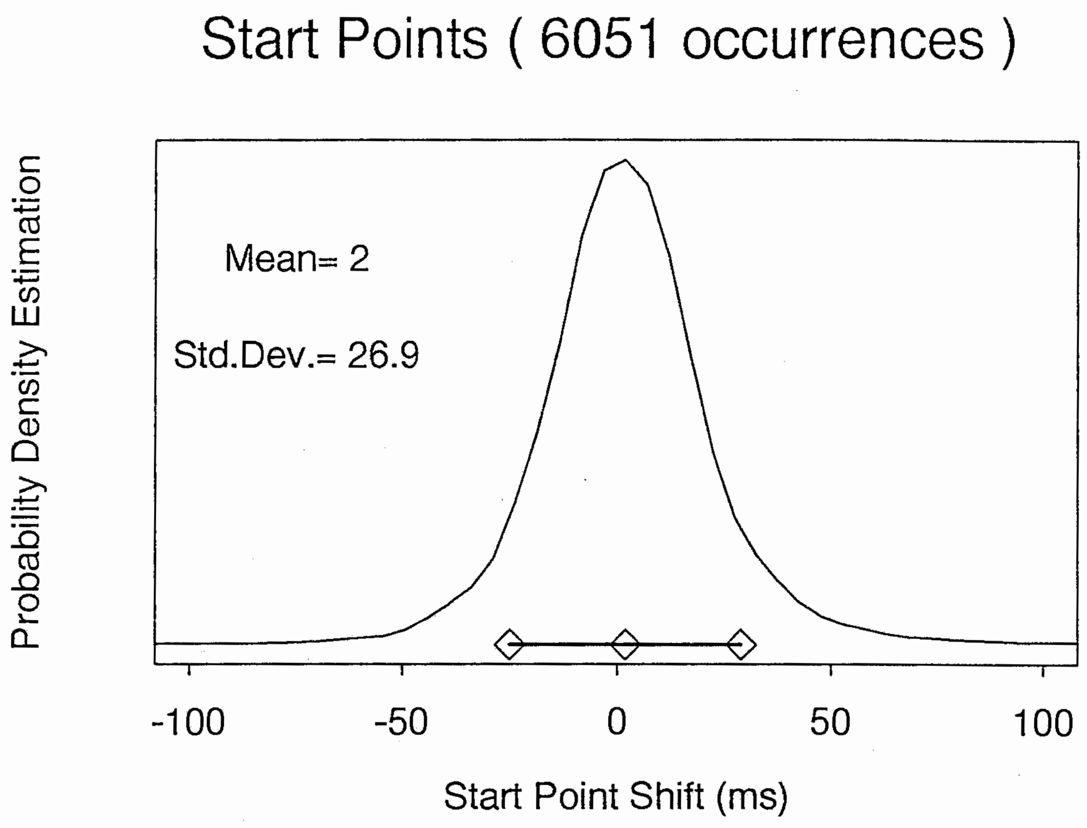
sentences

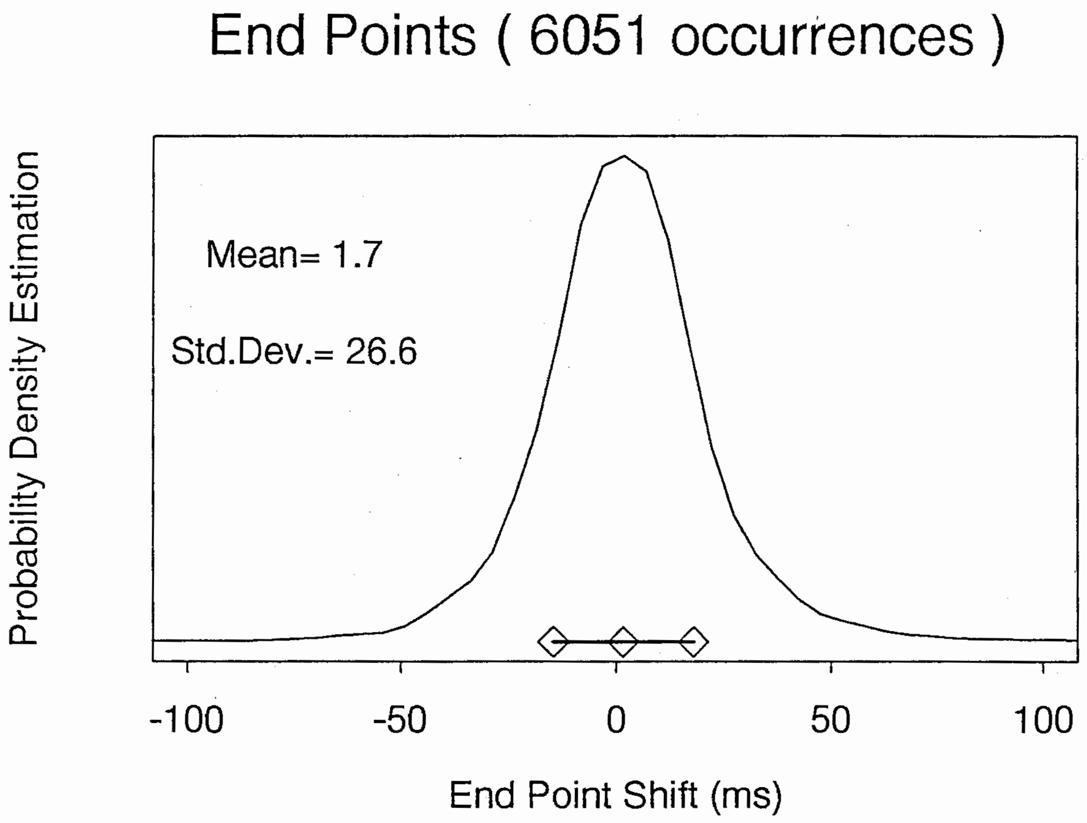
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3089	97.1	91.2	57.8	4.5	14.8	24.5
Vcons	1162	95.3	87.7	55.1	3.4	17.5	31.4
Ucons	1251	97.8	86.5	45.1	-6.9	17.2	22.0
all	6051	96.3	88.8	54.2	2.0	16.4	26.9

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3034	95.1	84.7	47.0	0.3	19.0	29.7
Vcons	1238	95.8	89.2	54.9	3.3	16.3	27.9
Ucons	1137	100.0	98.9	72.2	4.9	9.4	10.6
all	6051	96.4	88.8	54.0	1.7	16.4	26.6

End points


Start Points (6051 occurrences)
Probability Density Estimation
Mean= 2
Std.Dev.= 26.9
-100
-50
0
50
100
Start Point Shift (ms)


End Points (6051 occurrences)
Probability Density Estimation
Mean= 1.7
Std.Dev.= 26.6
-100
-50
0
50
100
End Point Shift (ms)

4.3 Multiple speaker training

4.3.1 Close speaker training

speaker MHT

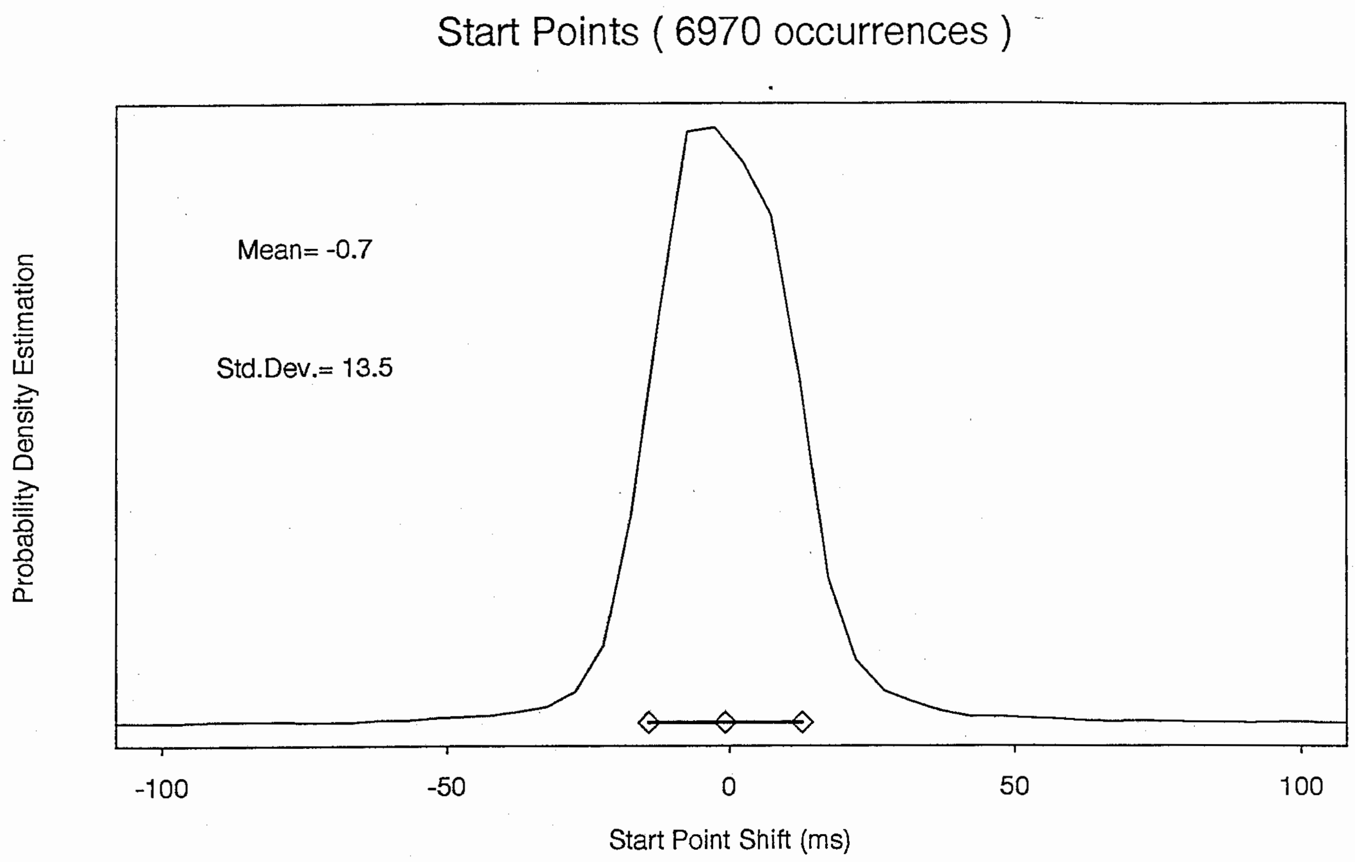
words

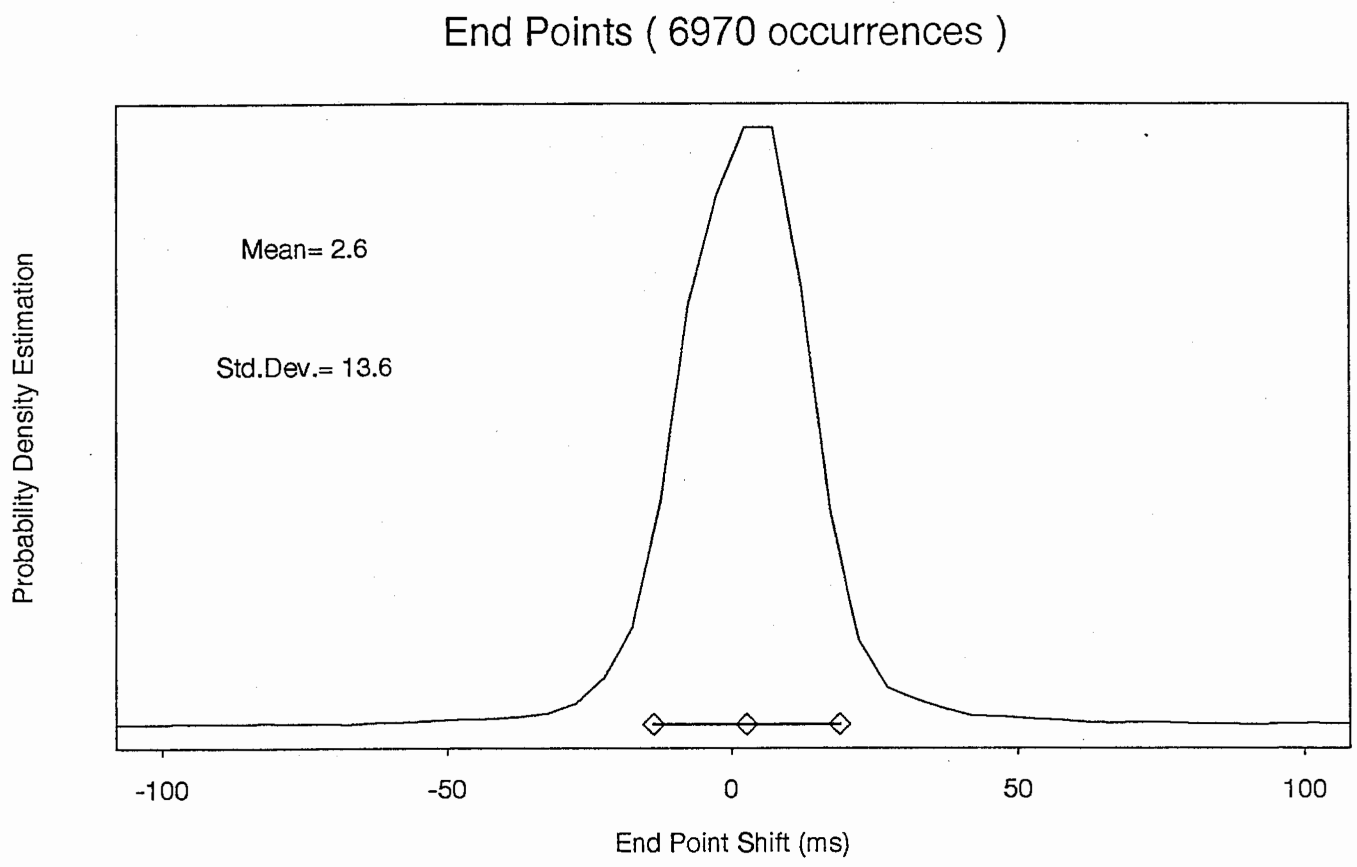
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3355	99.1	96.7	75.4	4.4	9.7	13.6
Vcons	1249	99.4	98.0	78.9	-6.7	9.1	11.2
Ucons	1516	99.7	98.7	66.9	-7.2	10.3	10.0
all	6970	99.3	97.5	74.4	-0.7	13.5	9.6

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3304	99.0	96.4	74.3	-0.9	10.0	14.7
Vcons	1289	99.0	97.1	75.8	6.8	9.6	12.6
Ucons	1447	99.6	99.5	87.2	4.3	6.8	8.3
all	6970	99.2	97.1	75.2	2.6	9.6	13.6

End points


Start Points (6970 occurrences)
Probability Density Estimation
Mean= -0.7
Std.Dev.= 13.5
-100
-50
0
50
100
Start Point Shift (ms)


End Points (6970 occurrences)
Probability Density Estimation
Mean= 2.6
Std.Dev.= 13.6
-100
-50
0
50
100
End Point Shift (ms)

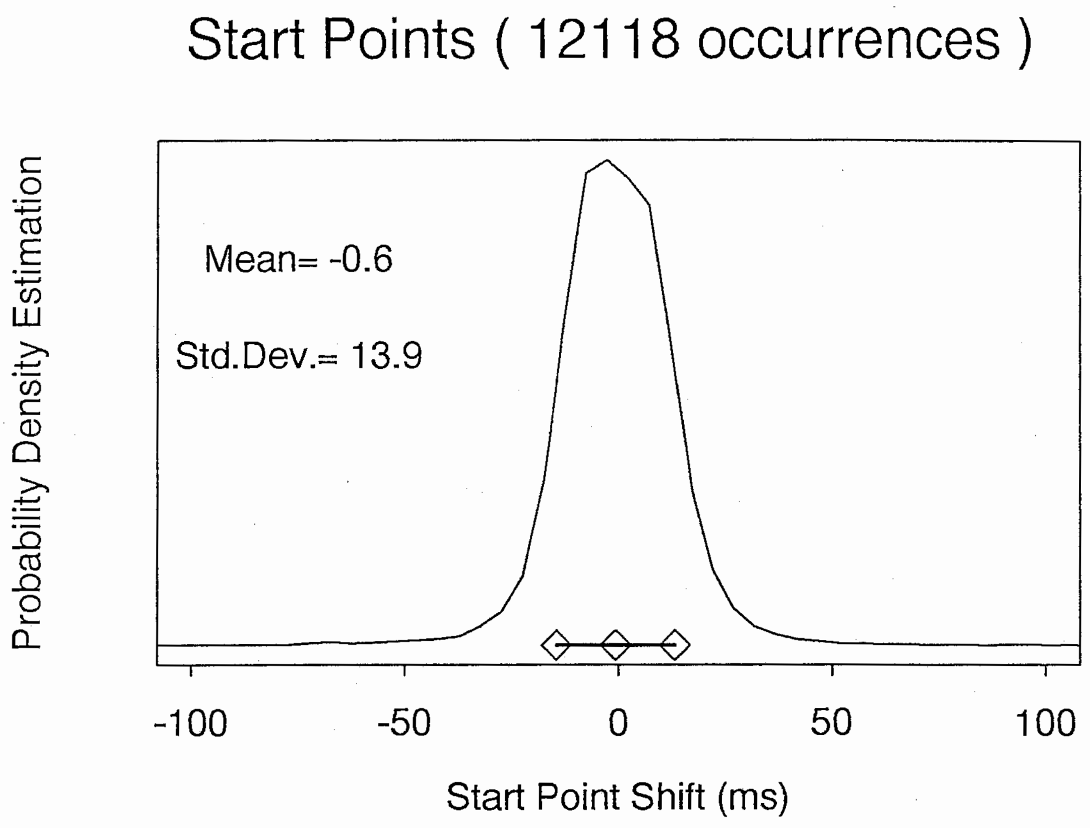
phrases

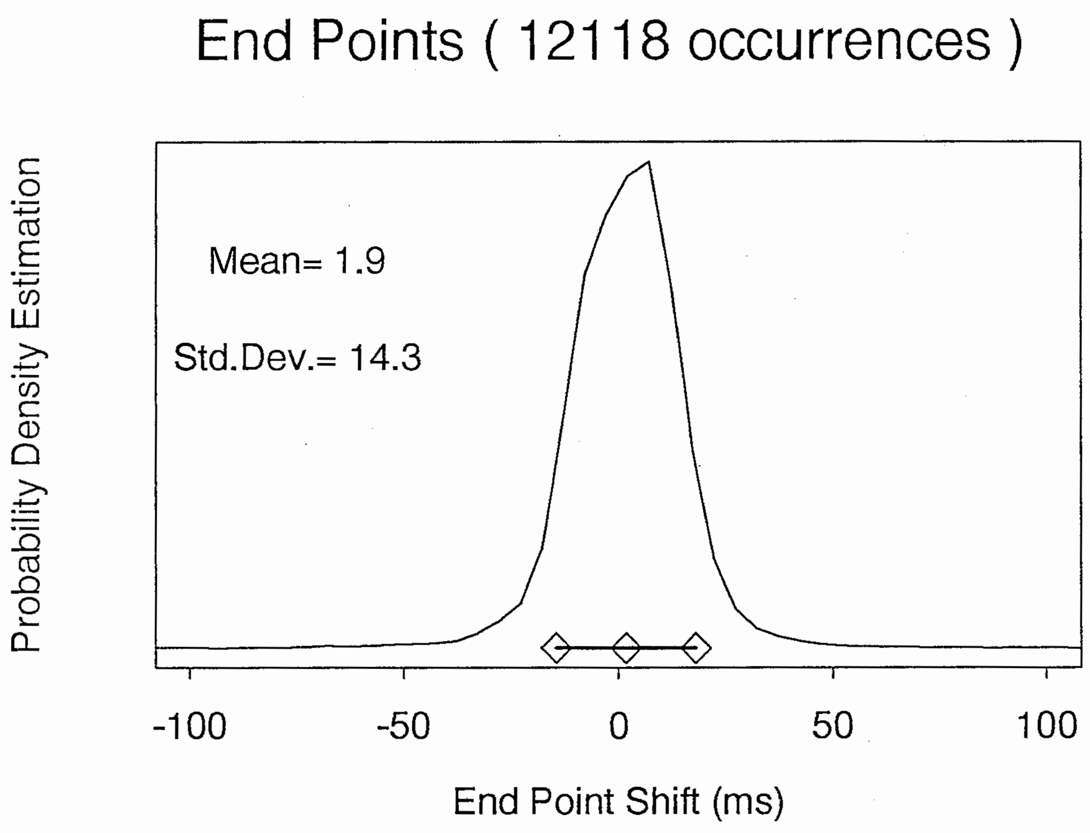
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	6200	99.2	97.4	73.9	4.9	9.7	12.8
Vcons	2326	99.4	98.1	74.7	-5.8	9.8	11.5
Ucons	2518	99.8	99.3	69.4	-7.8	10.0	10.0
all	12118	99.3	97.5	72.4	-0.6	10.0	13.9

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	6091	99.0	97.0	73.2	-1.5	10.2	15.1
Vcons	2474	99.4	97.7	73.0	6.0	9.8	12.5
Ucons	2251	100.0	99.8	84.6	5.2	7.1	7.2
all	12118	99.3	97.4	72.8	1.9	10.0	14.3

End points


Start Points (12118 occurrences)
Probability Density Estimation
Mean= -0.6
Std.Dev.= 13.9
-100
-50
0
50
100
Start Point Shift (ms)


End Points (12118 occurrences)
Probability Density Estimation
Mean= 1.9
Std.Dev.= 14.3
-100
-50
0
50
100
End Point Shift (ms)

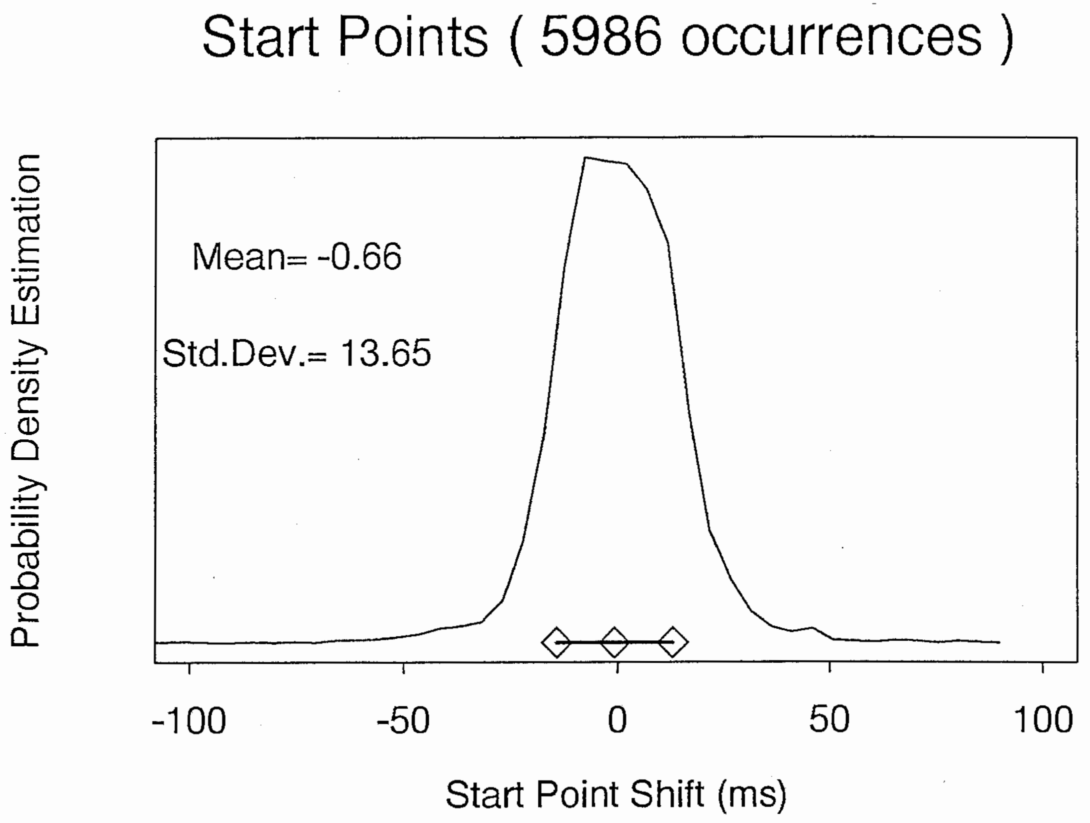
sentences

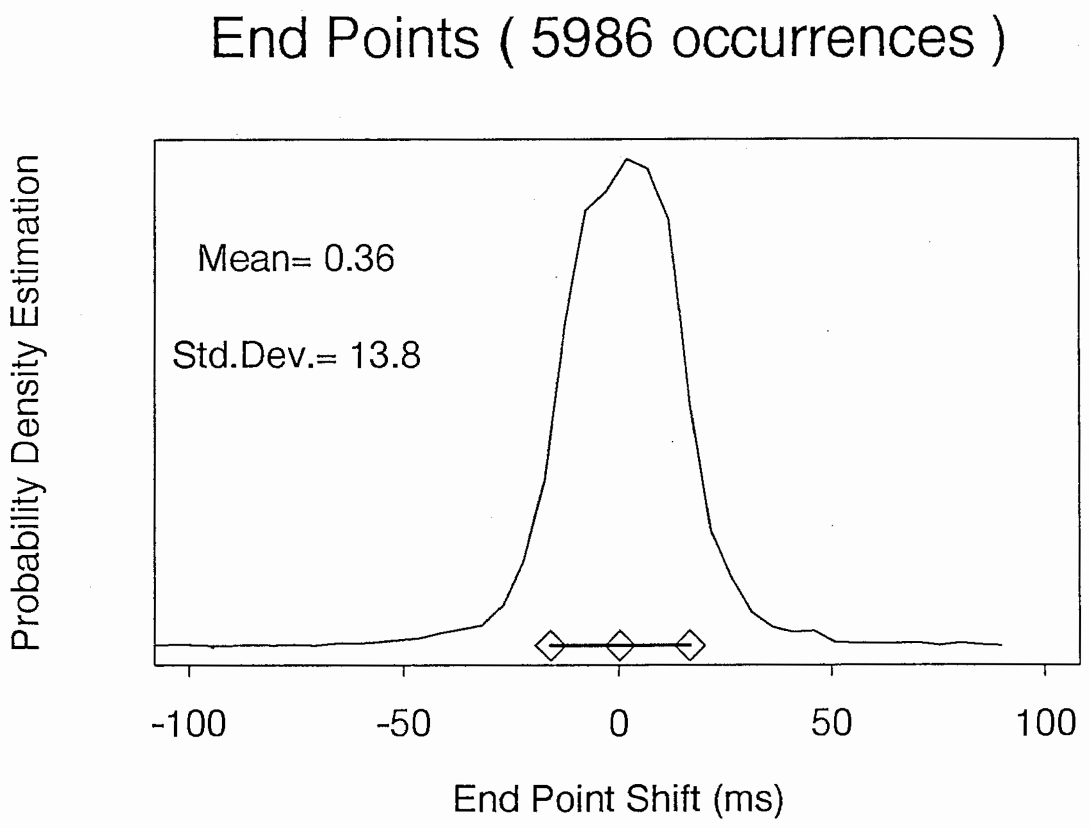
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3059	99.5	96.7	70.5	5.0	10.1	12.9
Vcons	1161	99.6	97.8	71.7	-6.0	10.5	11.9
Ucons	1239	100.0	99.4	66.9	-8.6	10.1	9.0
all	5986	99.5	97.3	70.0	-0.7	10.3	13.7

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	2992	99.3	96.6	70.8	-4.0	10.6	13.8
Vcons	1235	99.6	97.6	76.6	4.4	8.9	11.7
Ucons	1115	99.9	99.8	79.5	6.1	7.8	8.1
all	5986	99.5	97.1	70.8	0.4	10.3	13.8

End points


Start Points (5986 occurrences)
Probability Density Estimation
Mean= -0.66
Std.Dev.= 13.65
-100
-50
0
50
100
Start Point Shift (ms)


End Points (5986 occurrences)
Probability Density Estimation
Mean= 0.36
Std.Dev.= 13.8
-100
-50
0
50
100
End Point Shift (ms)

speaker MTK

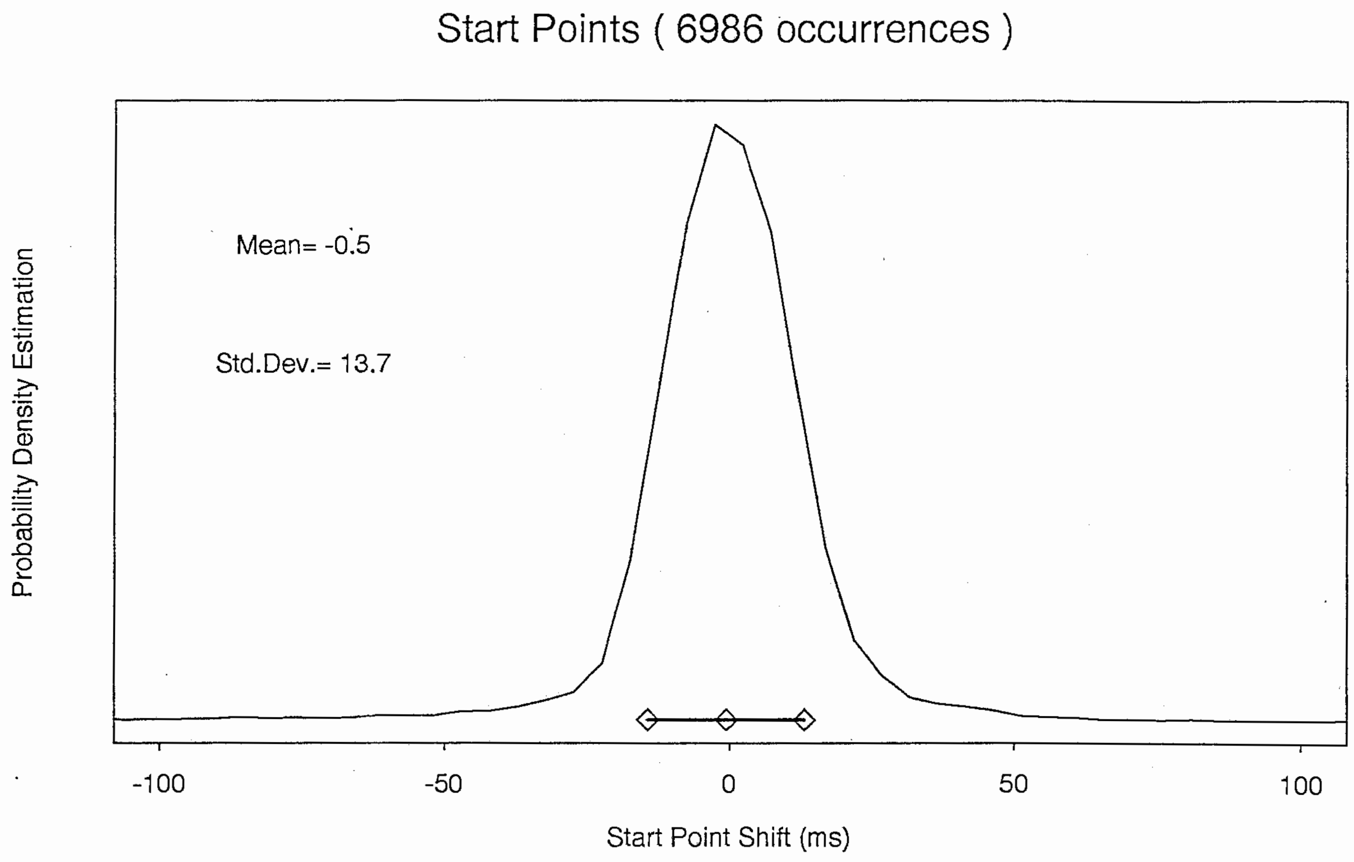
words

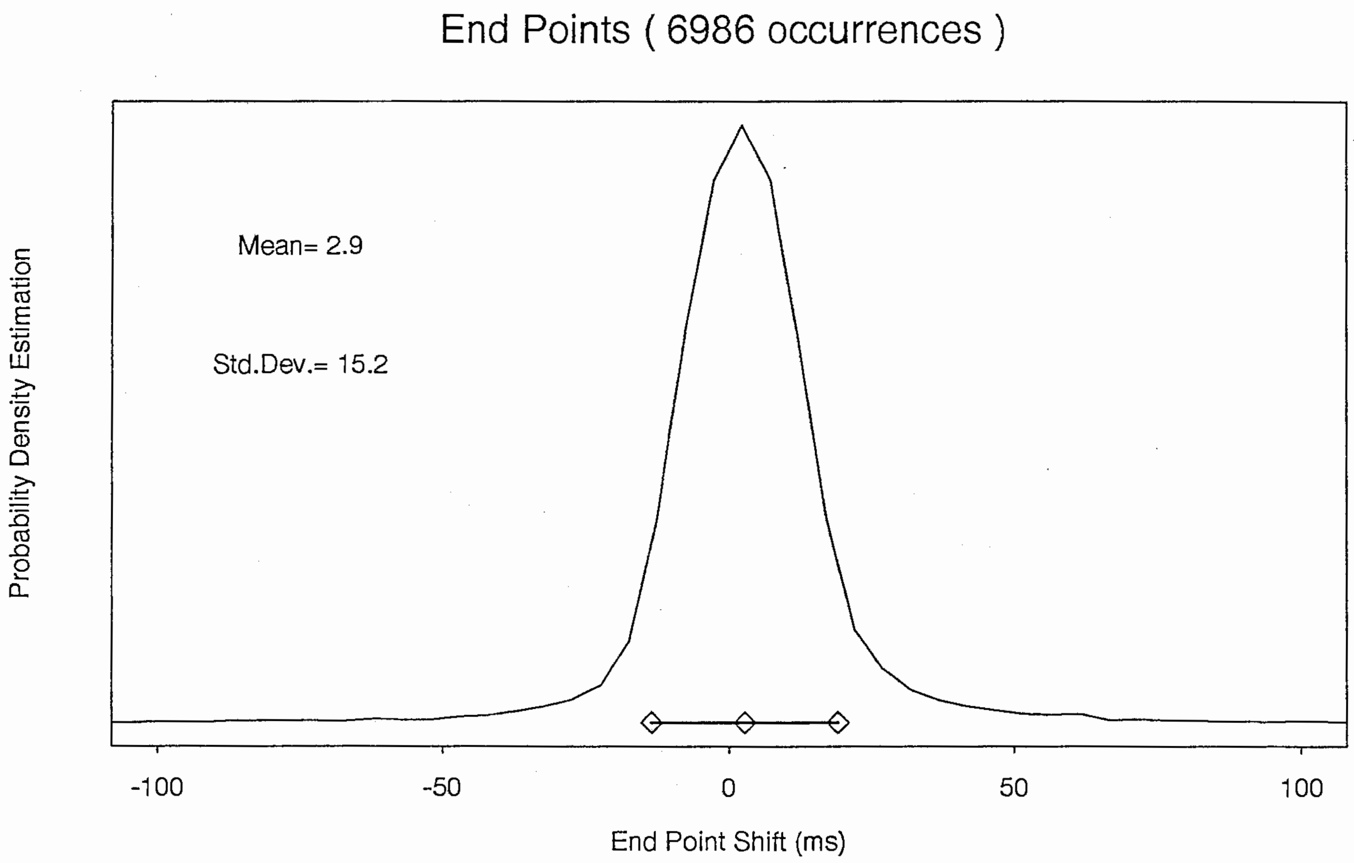
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3371	99.3	96.6	74.3	5.4	9.6	12.8
Vcons	1257	99.0	96.6	76.9	-7.7	9.6	11.4
Ucons	1521	99.3	97.2	71.7	-7.6	9.7	11.3
all	6986	99.2	96.9	75.0	-0.5	9.5	13.7

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3308	97.1	94.0	74.2	0.1	11.1	17.9
Vcons	1282	100.0	98.8	72.9	7.3	9.0	9.2
Ucons	1435	100.0	99.9	88.9	3.3	6.2	7.2
all	6986	98.5	95.7	74.9	2.9	10.1	15.2

End points


Start Points (6986 occurrences)
Mean= -0.5
Std.Dev.= 13.7
Probability Density Estimation
-100
-50
0
50
100
Start Point Shift (ms)


End Points (6986 occurrences)
Mean= 2.9
Std.Dev.= 15.2
Probability Density Estimation
-100
-50
0
50
100
End Point Shift (ms)

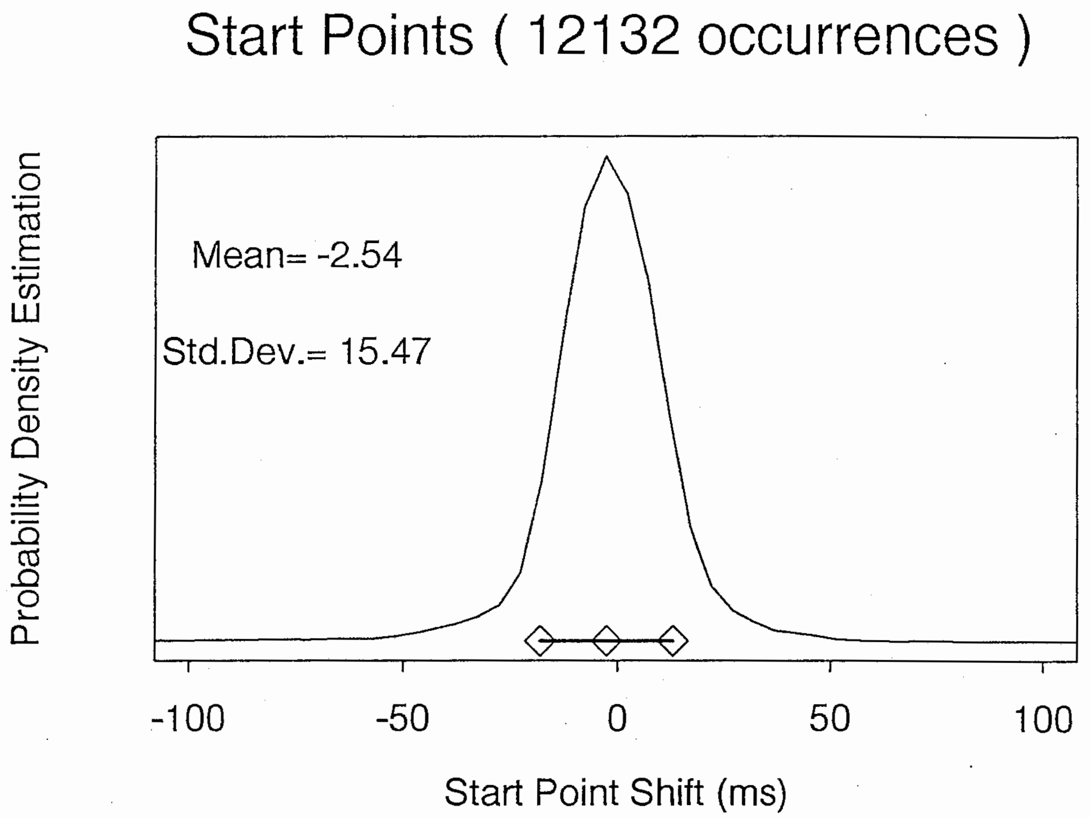
phrases

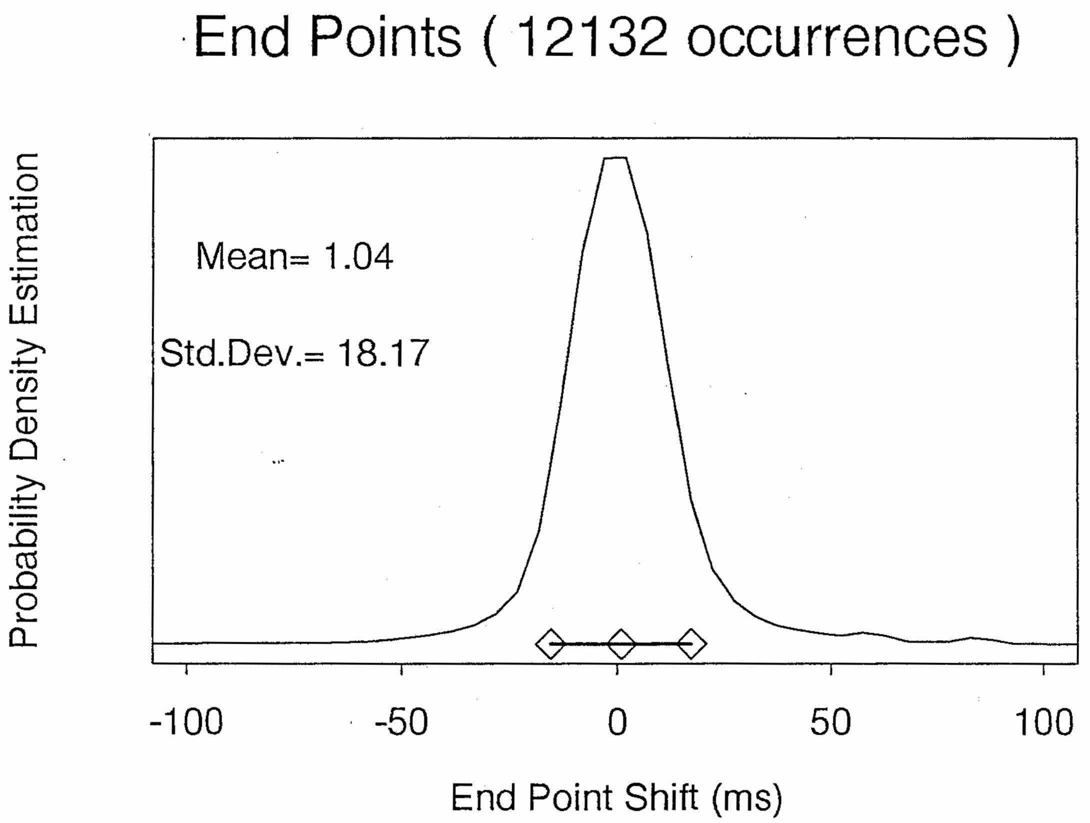
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	6192	99.4	96.8	75.9	3.3	9.2	13.0
Vcons	2319	99.2	97.1	70.2	-7.8	10.6	12.4
Ucons	2515	99.8	96.4	64.5	-9.0	11.1	11.6
all	12132	99.1	96.2	71.6	-2.5	10.3	15.5

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	6081	96.0	92.0	67.6	-1.1	13.2	22.0
Vcons	2470	99.7	99.0	80.7	3.3	7.8	10.2
Ucons	2247	100.0	99.8	87.6	2.8	6.2	7.6
all	12132	97.8	94.4	71.8	1.0	11.2	18.2

End points


Start Points (12132 occurrences)
Probability Density Estimation
Mean= -2.54
Std.Dev.= 15.47
-100
-50
0
50
100
Start Point Shift (ms)


End Points (12132 occurrences)
Probability Density Estimation
Mean= 1.04
Std.Dev.= 18.17
-100
-50
0
50
100
End Point Shift (ms)

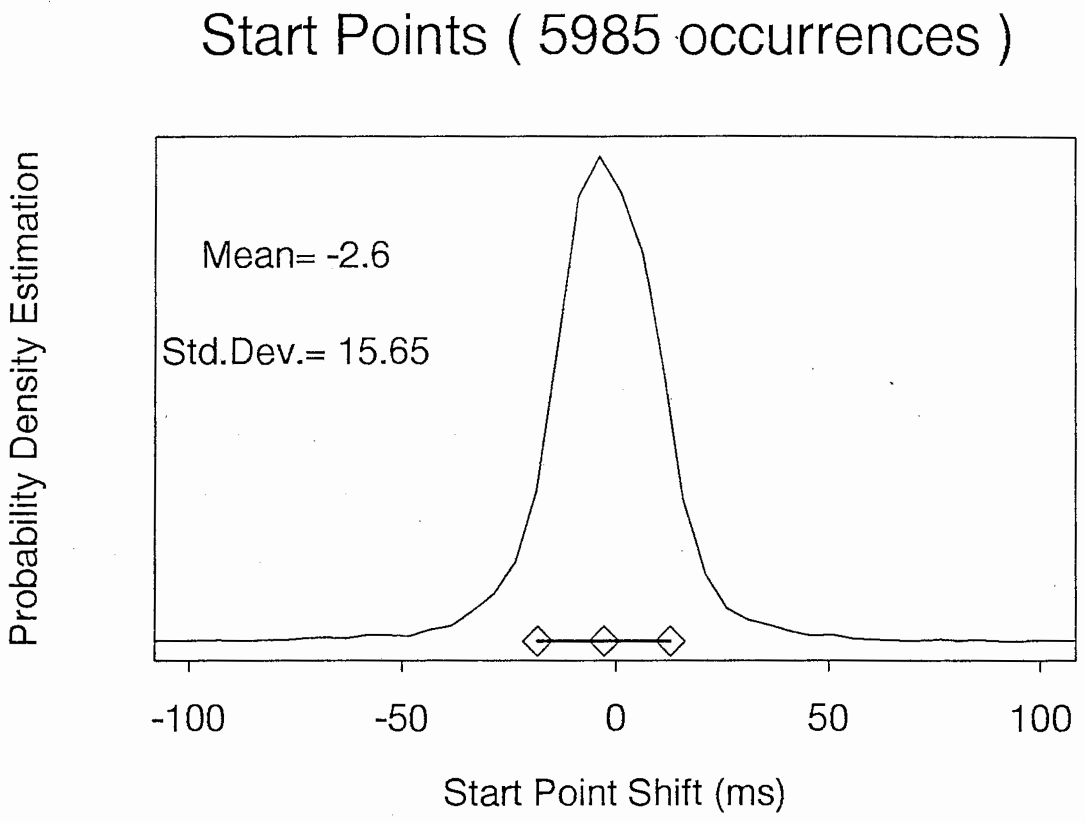
sentences

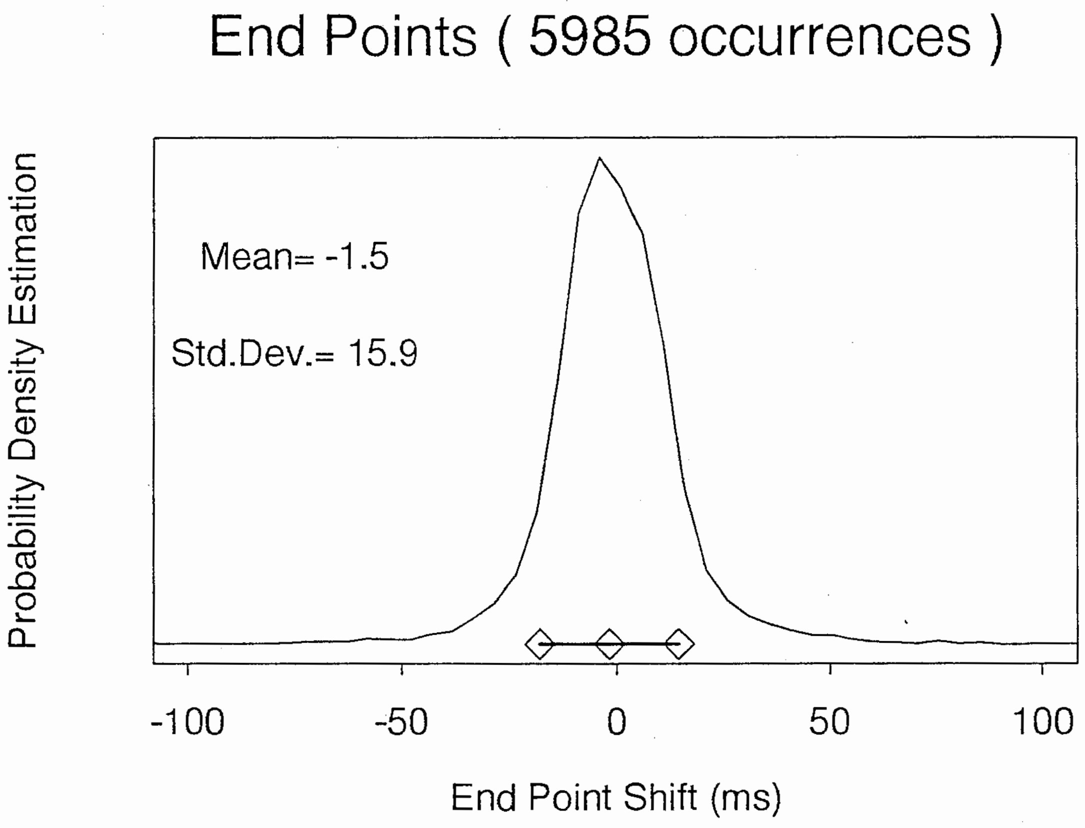
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3061	98.9	95.6	72.8	3.4	10.1	14.3
Vcons	1160	98.9	96.7	66.2	-8.2	11.4	13.7
Ucons	1240	99.4	96.5	59.5	-10.6	12.1	11.2
all	5985	98.7	95.5	68.4	-2.6	11.0	15.7

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3000	98.1	94.3	63.0	-5.4	12.5	17.0
Vcons	1237	99.6	98.6	80.8	2.1	8.0	11.1
Ucons	1114	99.9	99.4	85.6	3.2	6.9	8.4
all	5985	98.7	95.4	69.0	-1.5	11.0	15.9

End points


Start Points (5985 occurrences)
Probability Density Estimation
Mean= -2.6
Std.Dev.= 15.65
-100
-50
0
50
100
Start Point Shift (ms)


End Points (5985 occurrences)
Probability Density Estimation
Mean= -1.5
Std.Dev.= 15.9
-100
-50
0
50
100
End Point Shift (ms)

4.3.2 Open speaker testing

3 mixtures

- speaker MAU

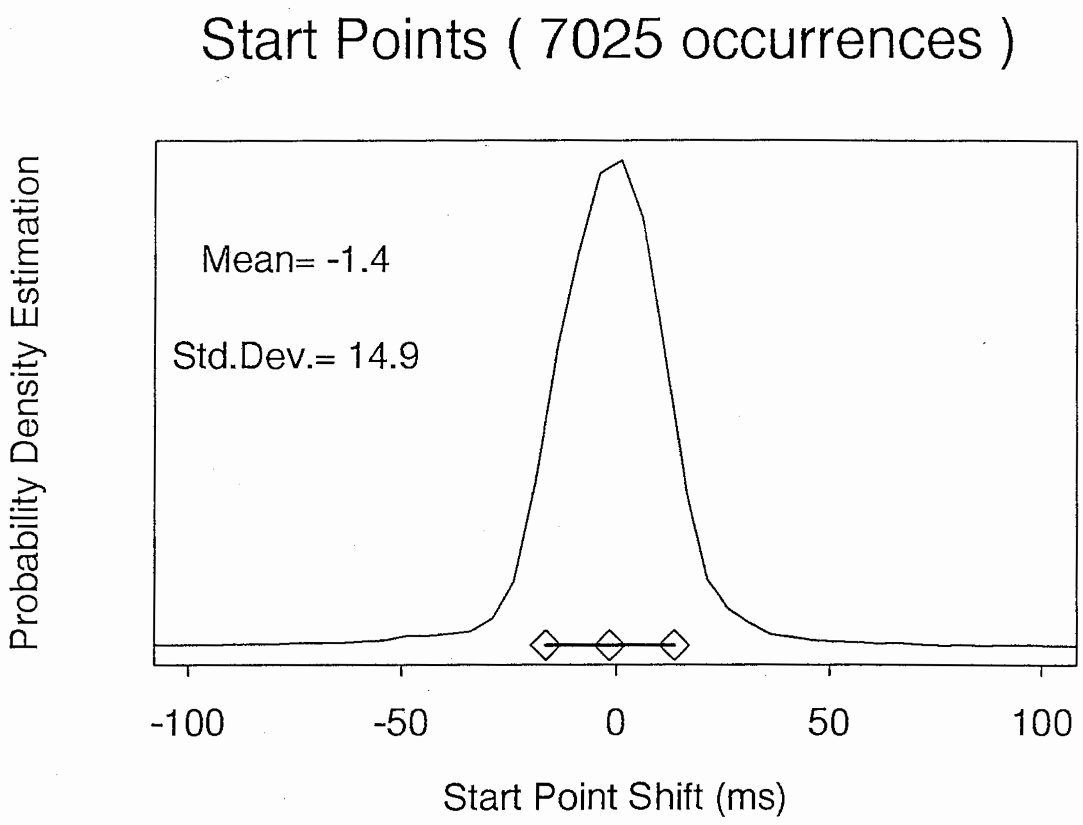
words

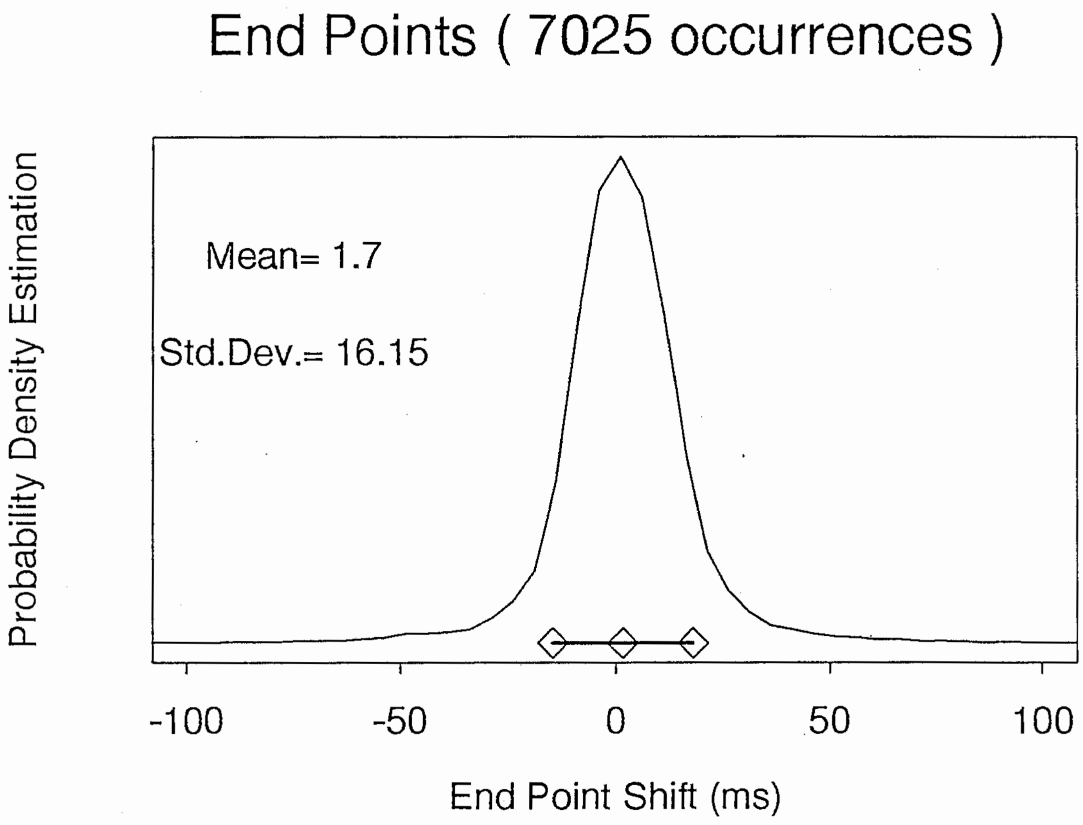
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3376	98.7	96.7	75.1	3.2	9.0	13.4
Vcons	1276	98.6	95.7	62.2	-7.6	11.8	14.5
Ucons	1521	99.5	95.6	50.0	-6.9	13.2	15.2
all	7025	98.8	96.4	67.0	-1.4	10.5	14.9

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3331	97.4	93.5	62.8	-0.3	12.6	18.7
Vcons	1306	98.5	95.4	69.5	5.7	10.3	13.9
Ucons	1445	99.9	99.8	89.8	3.0	5.6	7.7
all	7025	98.3	94.9	69.1	1.7	10.8	16.2

End points


Start Points (7025 occurrences)
Probability Density Estimation
Mean= -1.4
Std.Dev.= 14.9
-100
-50
0
50
100
Start Point Shift (ms)


End Points (7025 occurrences)
Probability Density Estimation
Mean= 1.7
Std.Dev.= 16.15
-100
-50
0
50
100
End Point Shift (ms)

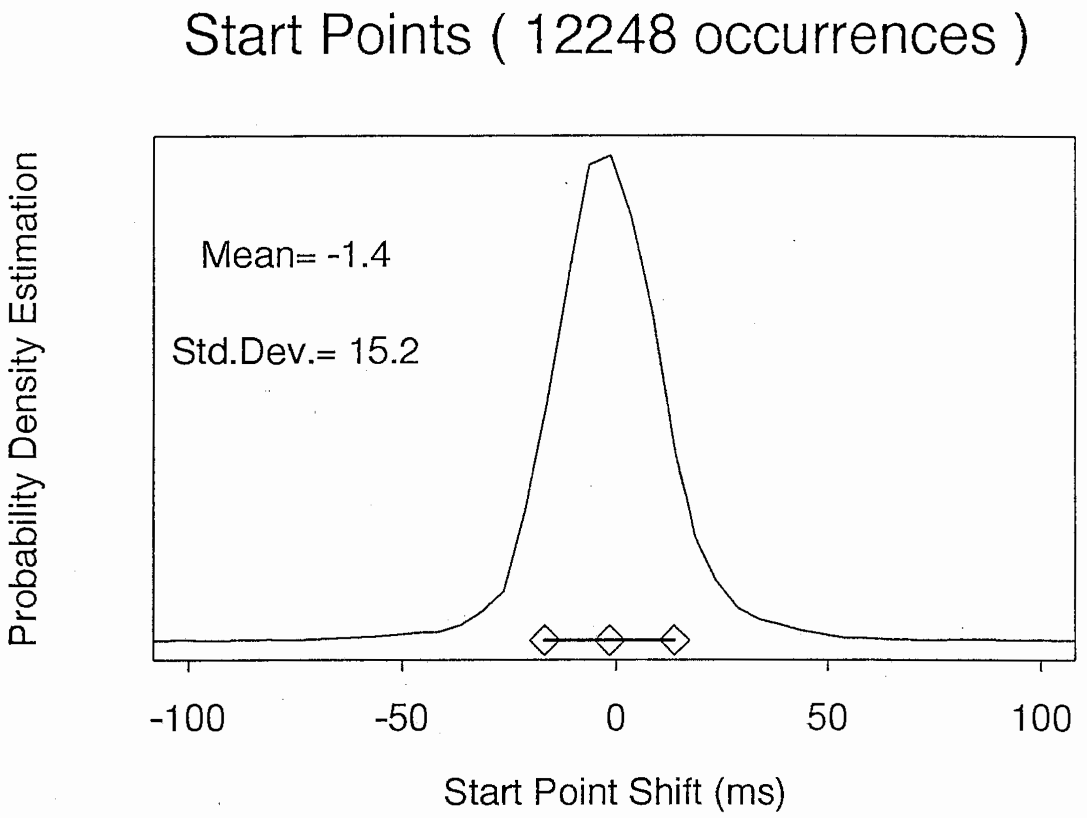
phrases

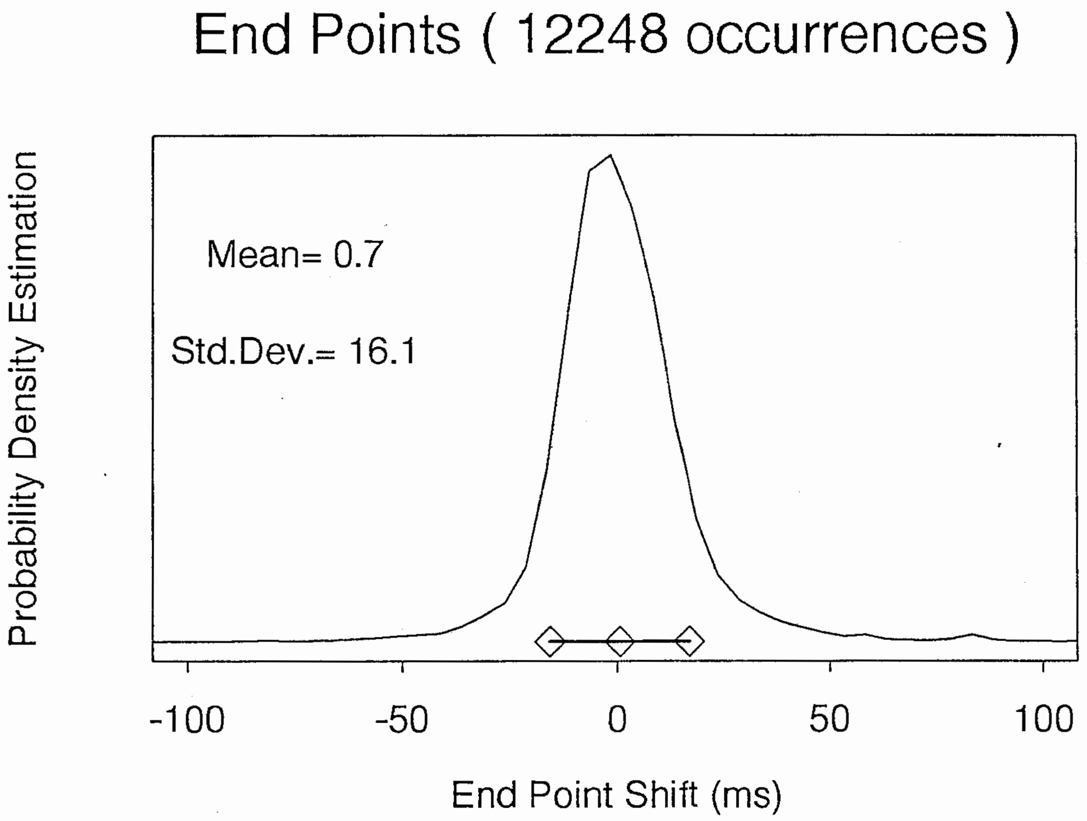
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	6255	99.2	96.1	72.0	3.4	9.5	13.4
Vcons	2385	98.7	95.7	59.7	-6.7	11.8	14.5
Ucons	2512	99.6	97.5	55.3	-8.1	12.6	11.9
all	12248	99.1	95.8	65.1	-1.4	10.8	15.2

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	6187	97.7	93.3	60.1	-0.9	12.6	18.4
Vcons	2466	99.3	94.7	67.5	5.4	10.7	14.0
Ucons	2297	99.9	99.7	89.9	1.5	5.4	7.3
all	12248	98.6	94.7	65.9	0.7	11.1	16.1

End points


Start Points (12248 occurrences)
Mean= -1.4
Std.Dev.= 15.2
Probability Density Estimation
-100
-50
0
50
100
Start Point Shift (ms)


End Points (12248 occurrences)
Mean= 0.7
Std.Dev.= 16.1
Probability Density Estimation
-100
-50
0
50
100
End Point Shift (ms)

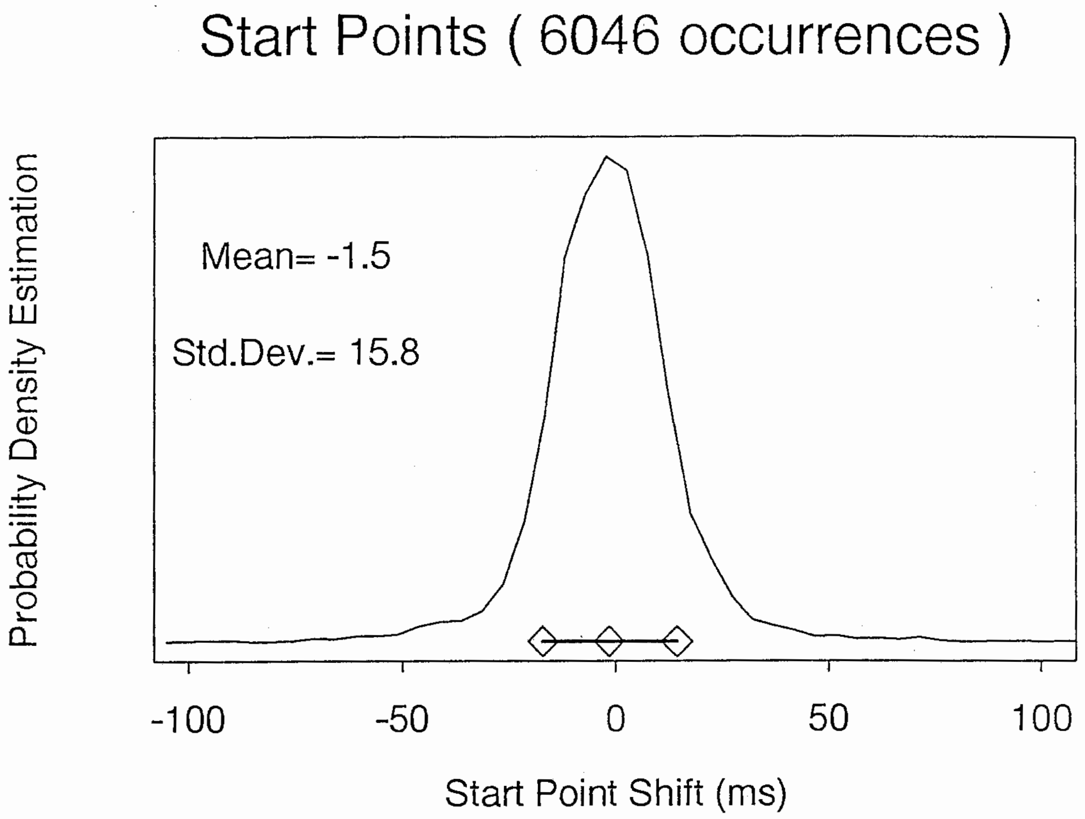
sentences

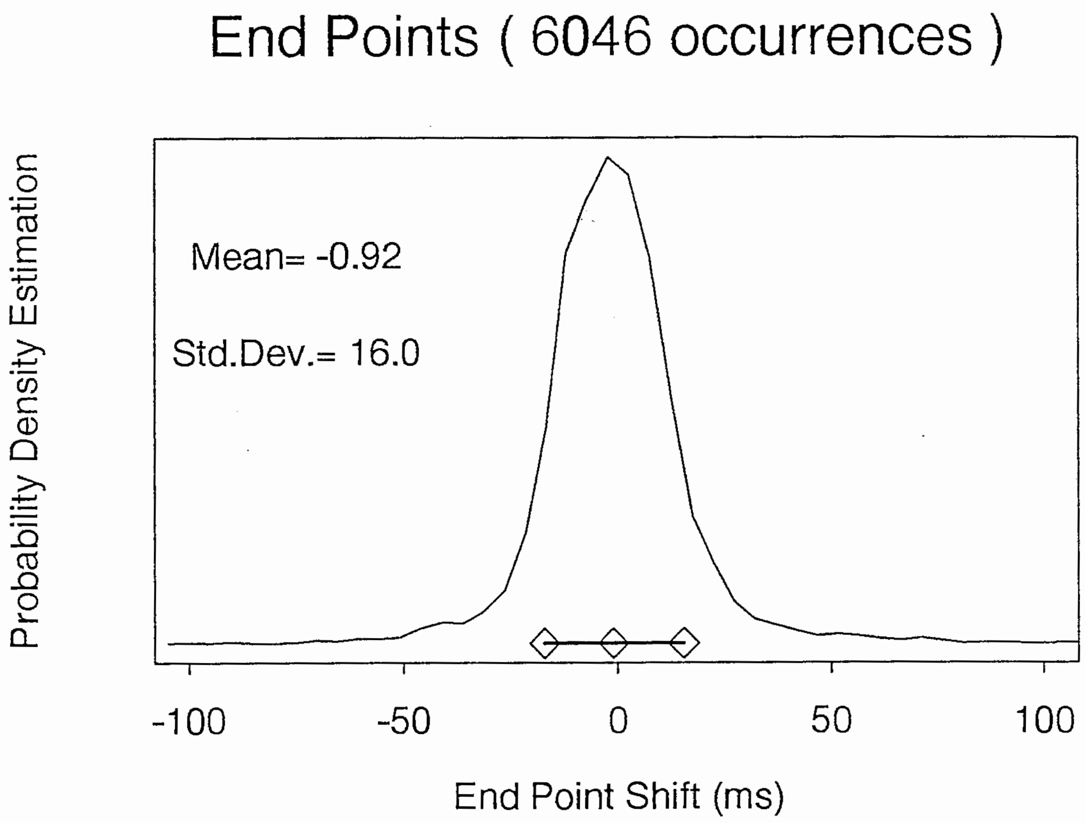
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3085	99.0	96.1	70.0	4.0	10.0	13.8
Vcons	1191	97.9	94.7	53.4	-8.6	13.2	16.0
Ucons	1248	98.9	94.7	56.9	-8.4	12.5	14.4
all	6046	98.7	95.2	63.6	-1.5	11.3	15.8

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3039	97.8	94.0	57.1	-4.5	12.8	17.1
Vcons	1233	98.8	95.6	69.5	5.0	10.5	14.4
Ucons	1146	99.7	99.0	85.0	3.3	6.6	8.5
all	6046	98.5	94.8	63.9	-0.9	11.4	16.0

End points


Start Points (6046 occurrences)
Probability Density Estimation
Mean= -1.5
Std.Dev.= 15.8
-100
-50
0
50
100
Start Point Shift (ms)


End Points (6046 occurrences)
Probability Density Estimation
Mean= -0.92
Std.Dev.= 16.0
-100
-50
0
50
100
End Point Shift (ms)

- speaker MNM

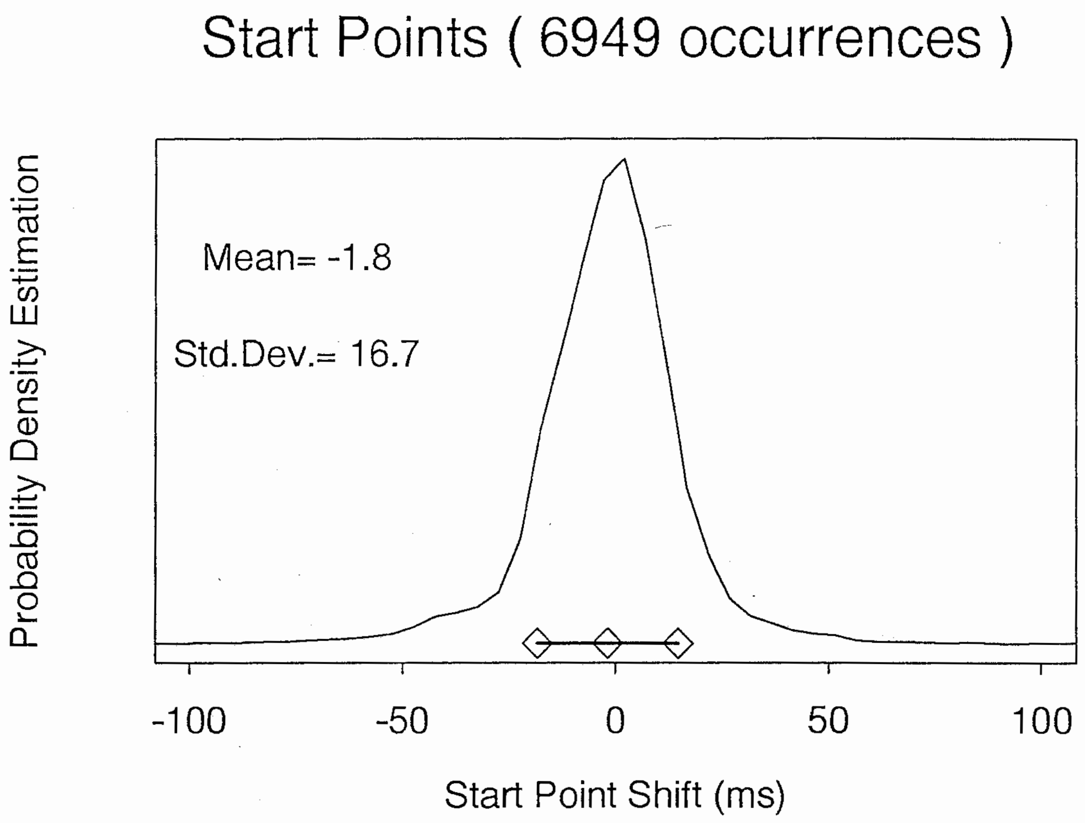
words

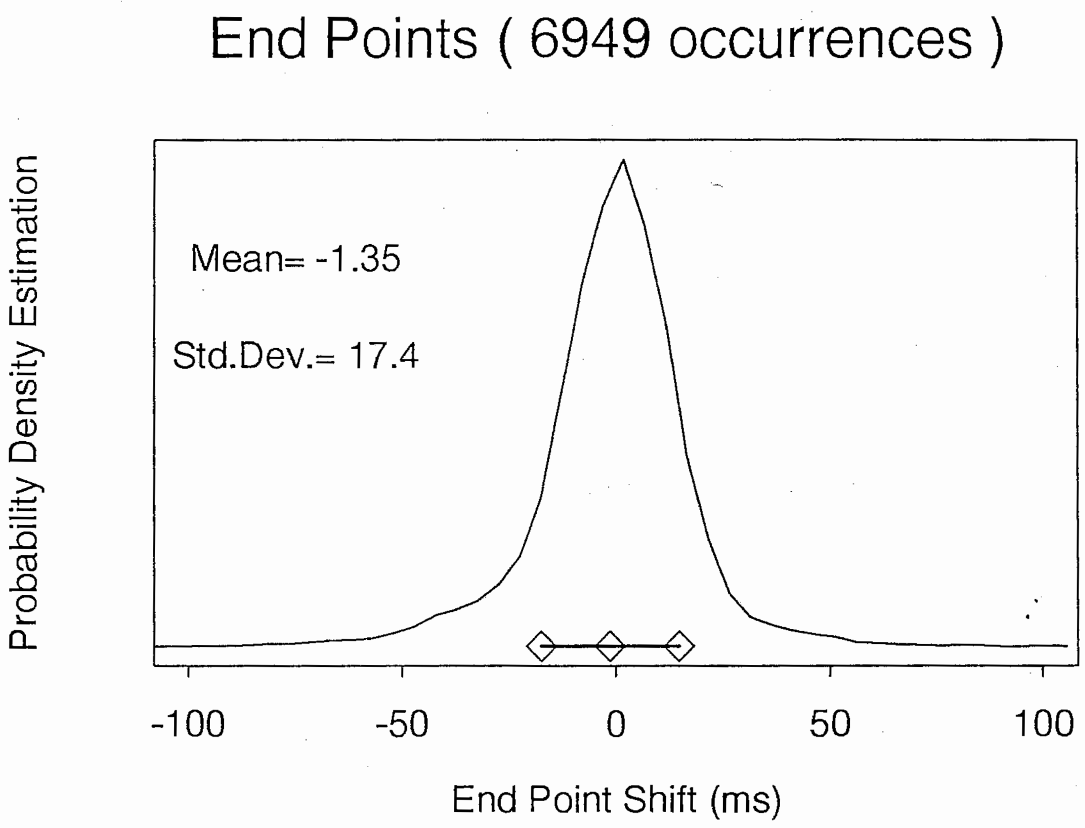
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3361	99.0	95.9	72.1	2.5	10.1	14.7
Vcons	1268	98.5	93.9	61.0	-0.2	12.5	17.6
Ucons	1494	98.5	88.4	45.3	-13.4	15.7	15.0
all	6949	98.7	93.8	63.9	-1.8	11.8	16.7

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3314	98.1	90.8	56.9	-4.4	14.1	19.1
Vcons	1304	98.9	95.2	71.4	3.3	10.3	14.6
Ucons	1414	100.0	99.7	82.1	2.1	7.0	8.9
all	6949	98.5	92.8	64.0	-1.4	12.2	17.4

End points


Start Points (6949 occurrences)
Probability Density Estimation
Mean= -1.8
Std.Dev.= 16.7
-100
-50
0
50
100
Start Point Shift (ms)


End Points (6949 occurrences)
Probability Density Estimation
Mean= -1.35
Std.Dev.= 17.4
-100
-50
0
50
100
End Point Shift (ms)

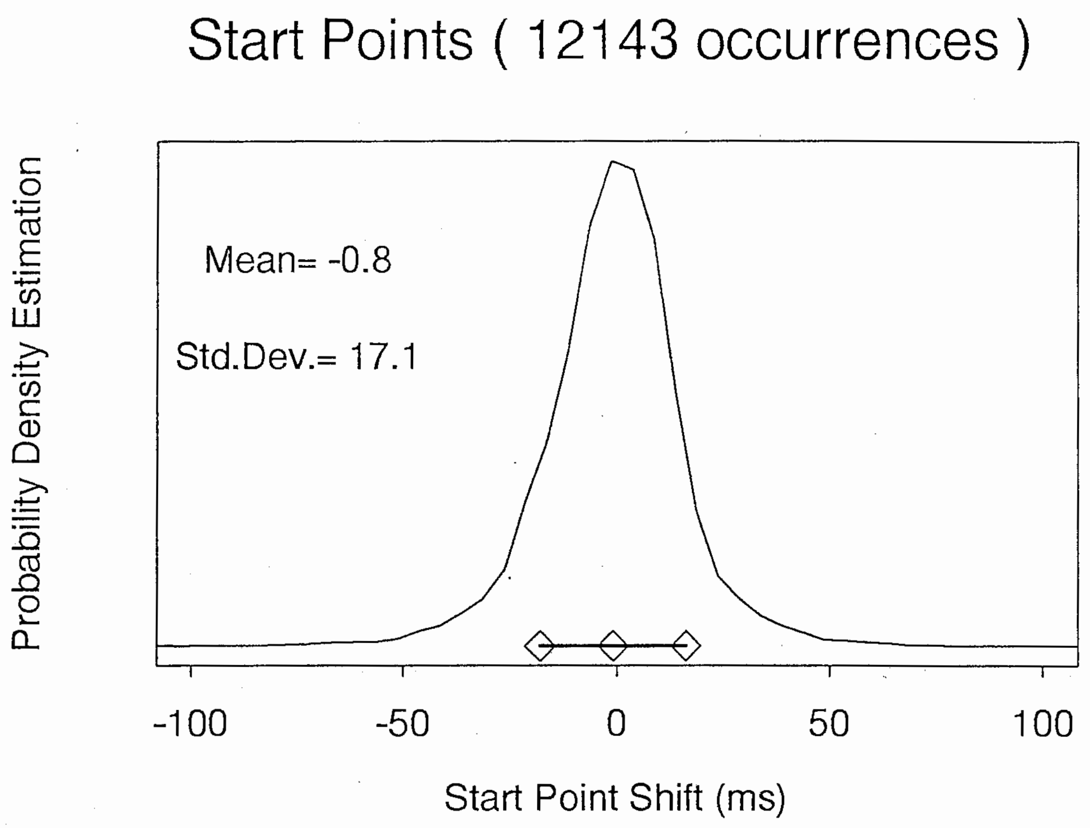
phrases

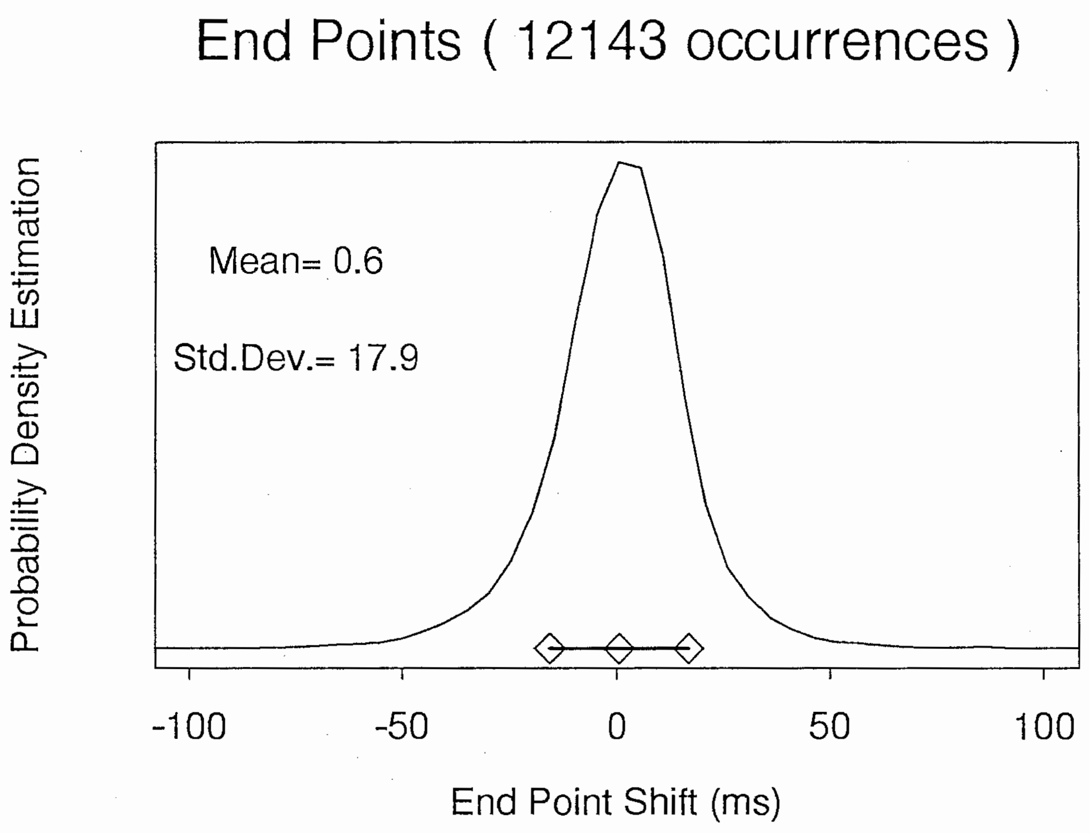
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	6233	99.2	96.0	71.3	4.7	10.2	13.6
Vcons	2312	98.7	94.5	63.6	-2.1	11.8	16.2
Ucons	2486	99.4	91.7	48.9	-11.9	14.8	14.6
all	12143	98.7	94.1	64.0	-0.8	12.0	17.1

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. berror (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	6137	97.8	91.9	55.7	-2.7	14.2	20.0
Vcons	2460	99.7	97.6	76.1	2.5	9.1	15.0
Ucons	2248	100.0	99.9	79.0	6.1	7.7	7.3
all	12143	98.6	93.7	63.0	0.6	12.3	17.9

End points


Start Points (12143 occurrences)
Probability Density Estimation
Mean= -0.8
Std.Dev.= 17.1
-100
-50
0
50
100
Start Point Shift (ms)


End Points (12143 occurrences)
Probability Density Estimation
Mean= 0.6
Std.Dev.= 17.9
-100
-50
0
50
100
End Point Shift (ms)

sentences

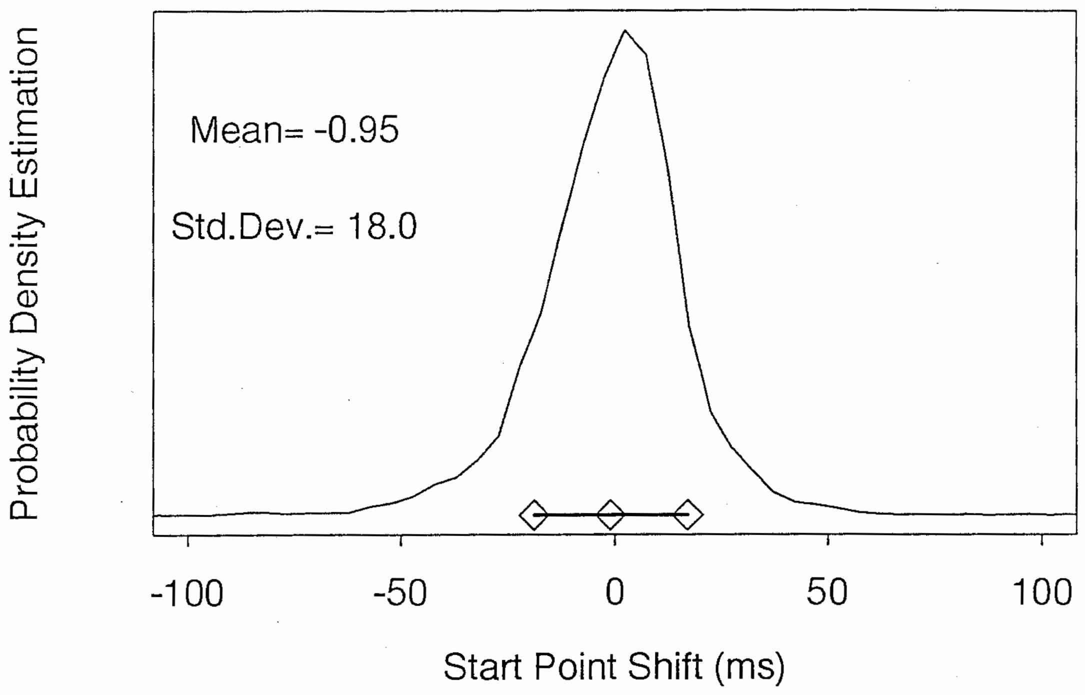
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3076	99.4	95.7	65.9	5.3	11.2	14.2
Vcons	1162	98.3	94.8	64.8	-2.6	11.7	16.3
Ucons	1244	99.2	88.5	42.3	-13.1	16.5	16.4
all	6028	98.7	93.3	60.3	-1.0	12.8	18.0

Start points

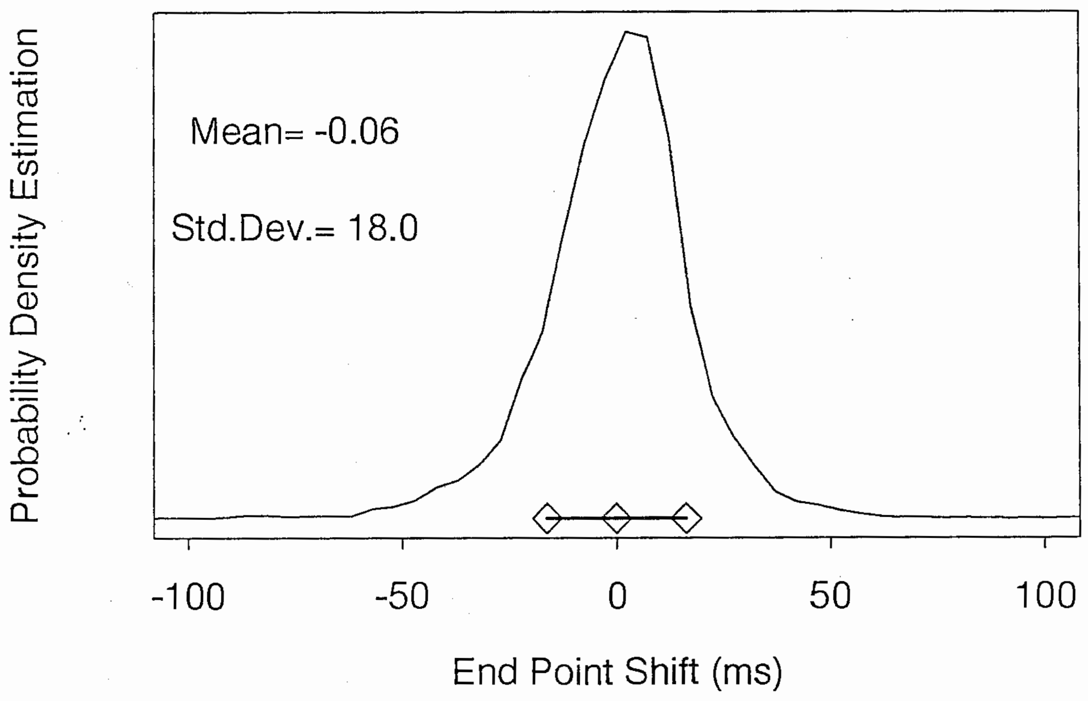
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3027	98.1	91.4	52.7	-4.6	14.7	19.9
Vcons	1232	99.4	97.7	71.4	3.0	9.8	13.2
Ucons	1134	100.0	99.4	76.8	7.1	8.3	7.7
all	6028	98.7	93.2	59.8	-0.1	12.9	18.0

End points

Start Points (6028 occurrences)



End Points (6028 occurrences)



- speaker FSU

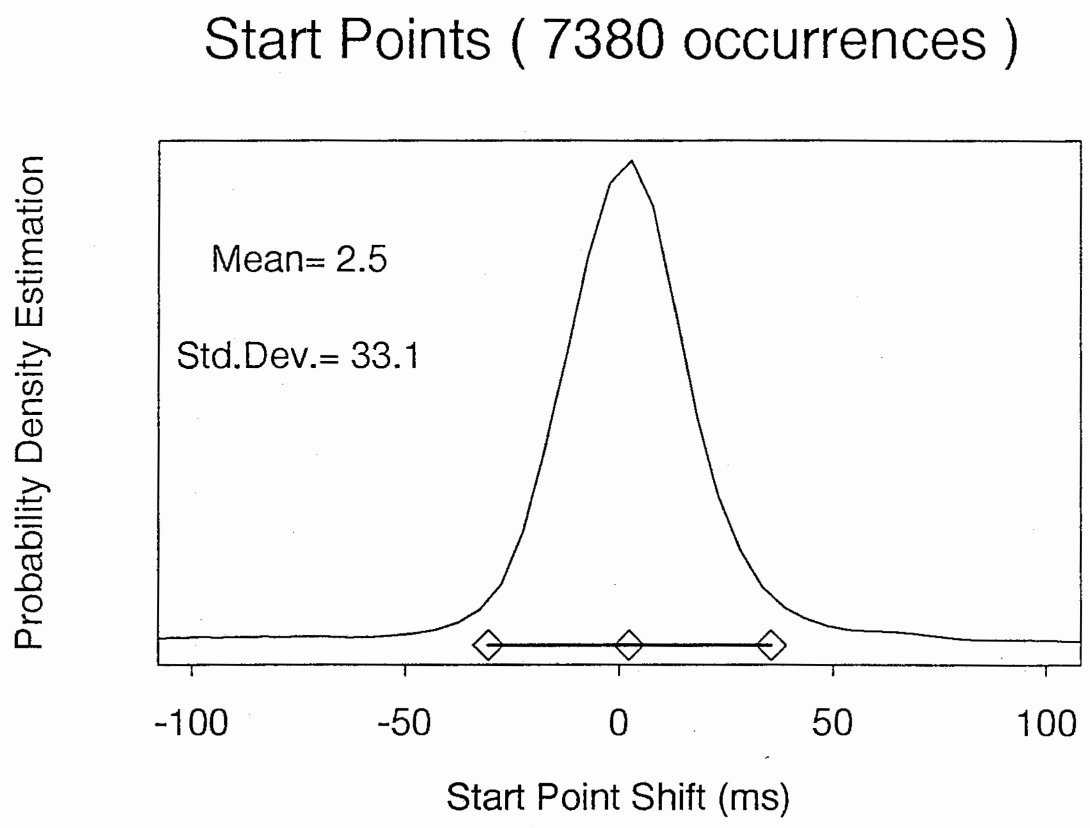
words

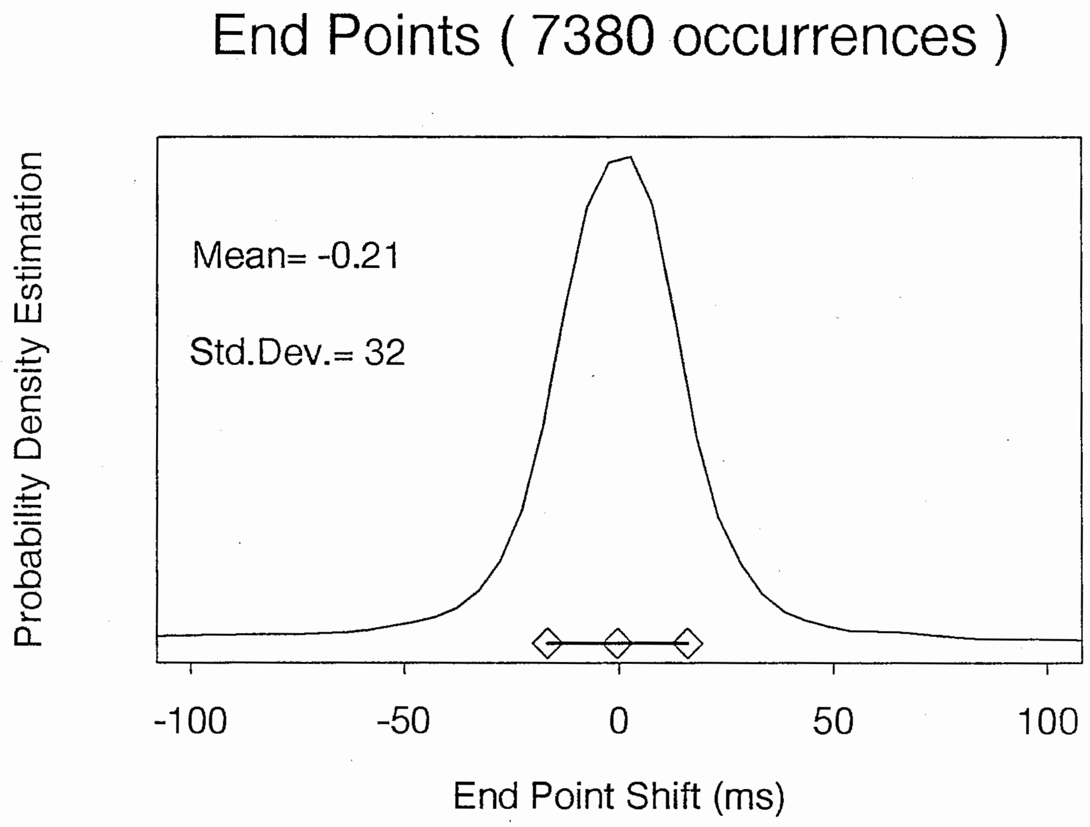
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3369	90.8	85.6	56.5	5.8	20.3	37.4
Vcons	1243	89.9	83.5	54.6	-4.8	21.3	36.6
Ucons	1482	97.8	94.0	55.9	-0.8	23.1	14.0
all	7380	93.0	88.2	56.8	2.5	18.1	33.1

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3733	91.5	85.9	55.1	-6.4	19.5	33.1
Vcons	1298	91.7	85.6	57.4	8.2	18.5	31.6
Ucons	1420	99.8	98.9	75.6	6.0	9.0	15.1
all	7380	93.5	88.2	58.1	-0.2	17.7	32.0

End points


Start Points (7380 occurrences)
Probability Density Estimation
Mean= 2.5
Std.Dev.= 33.1
-100
-50
0
50
100
Start Point Shift (ms)


End Points (7380 occurrences)
Probability Density Estimation
Mean= -0.21
Std.Dev.= 32
-100
-50
0
50
100
End Point Shift (ms)

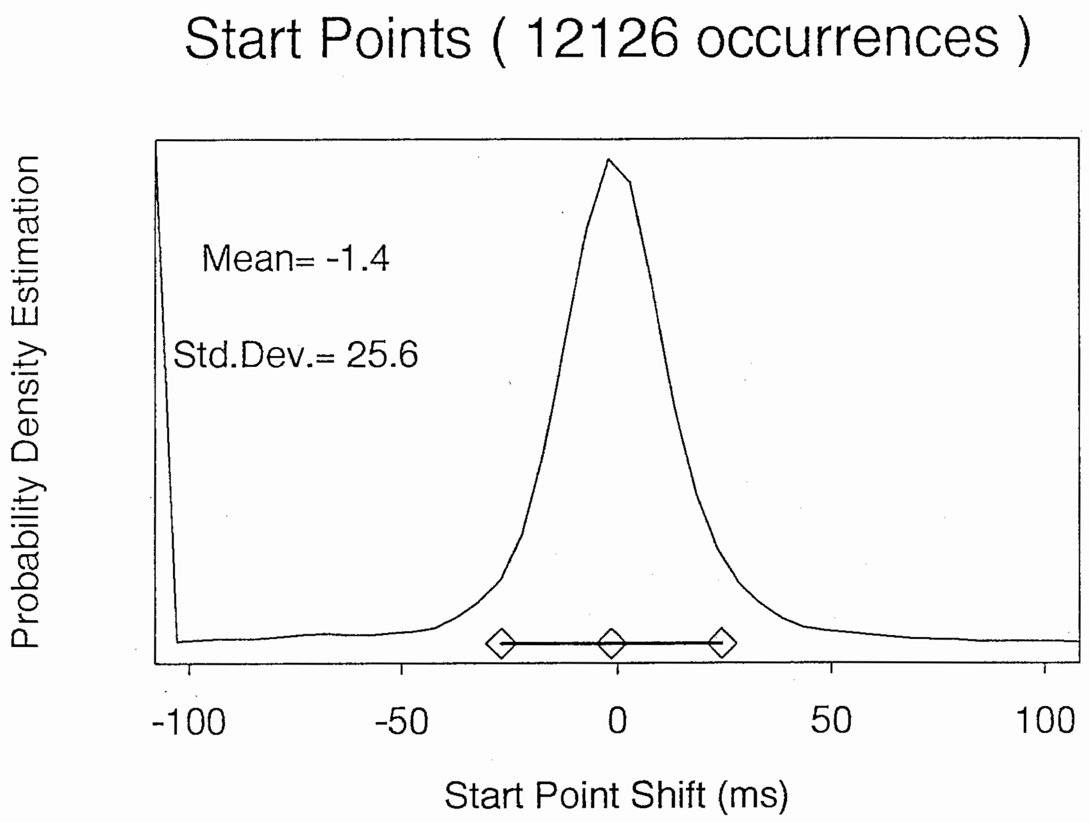
phrases

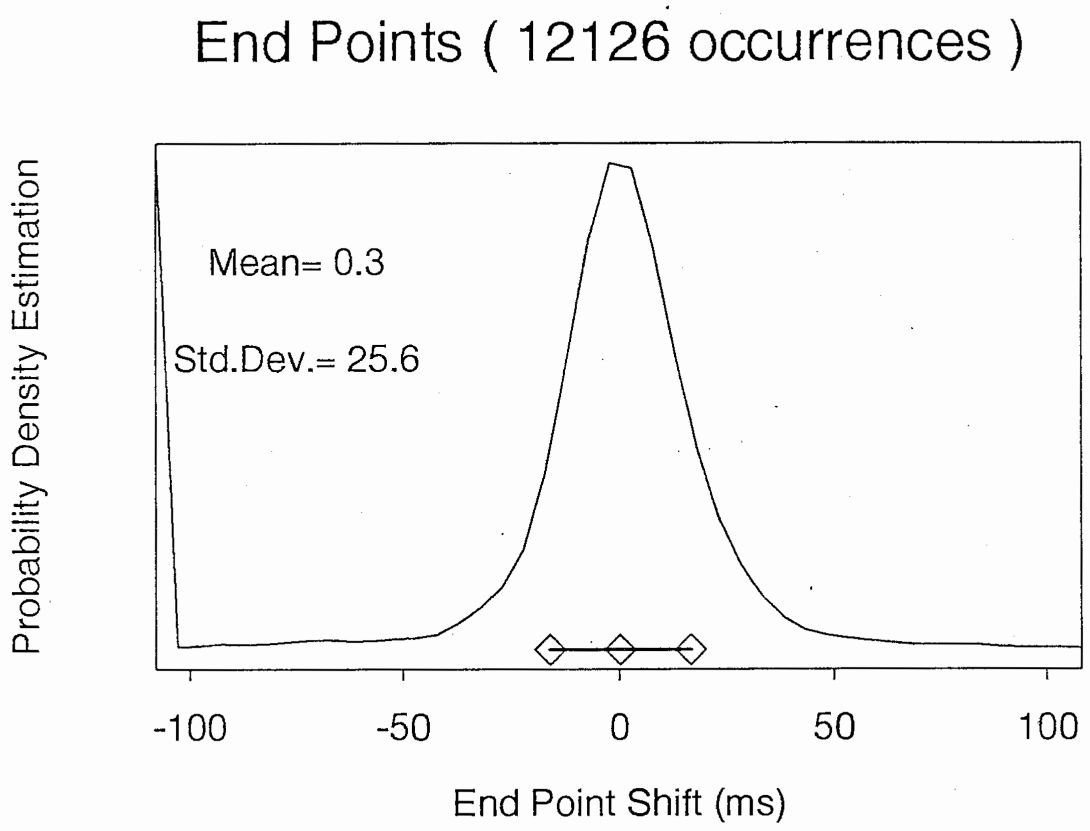
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	5846	94.2	88.7	64.5	2.4	15.1	27.5
Vcons	2303	94.5	89.1	56.3	-9.8	16.0	23.8
Ucons	2502	98.8	94.9	56.3	-1.9	12.7	16.8
all	12126	95.2	90.2	60.7	-1.4	14.9	25.6

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	6034	93.3	88.3	53.2	-3.6	17.6	29.4
Vcons	2479	96.1	89.3	62.4	6.9	14.0	21.9
Ucons	2319	100.0	99.2	80.0	3.5	7.2	9.2
all	12126	95.4	90.1	58.4	0.3	15.2	25.6

End points


Start Points (12126 occurrences)
Mean= -1.4
Std.Dev.= 25.6
Probability Density Estimation
-100
-50
0
50
100
Start Point Shift (ms)


End Points (12126 occurrences)
Mean= 0.3
Std.Dev.= 25.6
Probability Density Estimation
-100
-50
0
50
100
End Point Shift (ms)

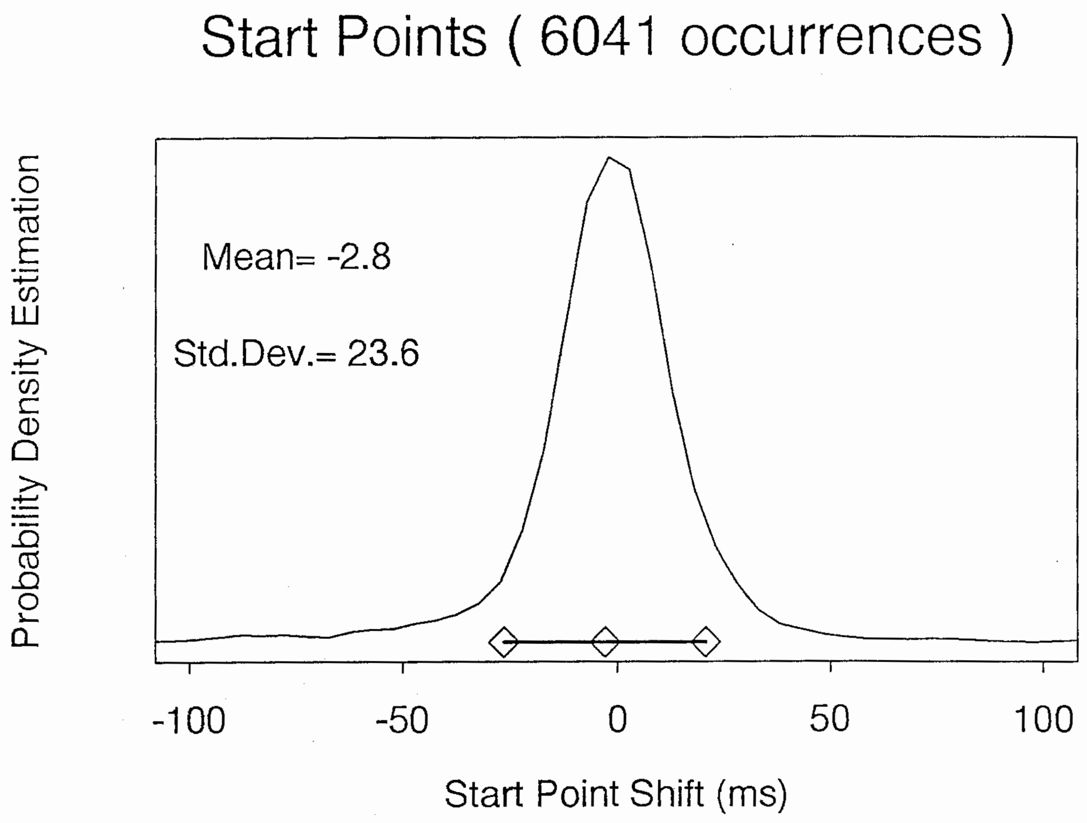
sentences

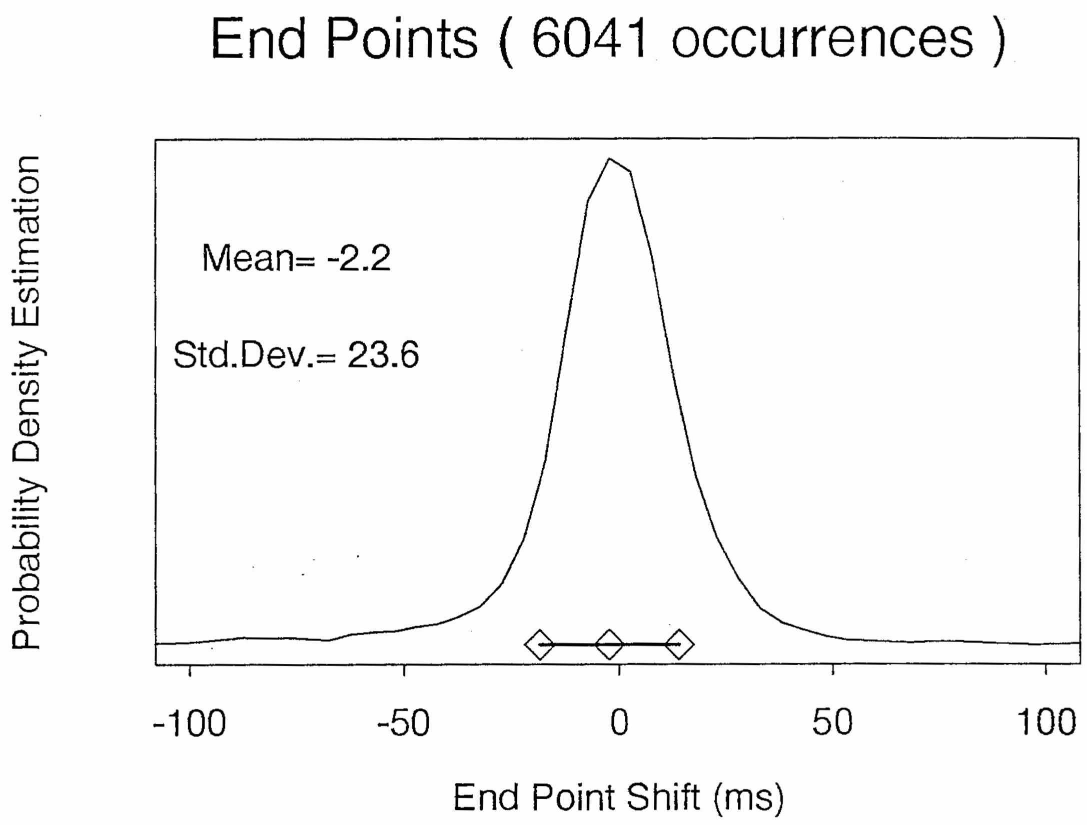
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3111	95.3	90.0	64.0	0.3	14.2	25.0
Vcons	1166	95.7	90.4	55.5	-10.8	15.3	22.1
Ucons	1266	99.2	96.8	59.4	-2.4	12.0	15.7
all	6041	96.2	91.5	61.5	-2.8	14.1	23.6

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	2981	94.1	88.6	54.2	-6.6	16.9	27.3
Vcons	1260	98.5	94.7	66.4	3.2	11.5	19.0
Ucons	1156	99.9	99.5	85.3	1.8	6.5	9.5
all	6041	96.3	91.4	60.8	-2.2	14.1	23.6

End points


Start Points (6041 occurrences)
Probability Density Estimation
Mean= -2.8
Std.Dev.= 23.6
-100
-50
0
50
100
Start Point Shift (ms)


End Points (6041 occurrences)
Probability Density Estimation
Mean= -2.2
Std.Dev.= 23.6
-100
-50
0
50
100
End Point Shift (ms)

9 mixtures

- speaker MAU

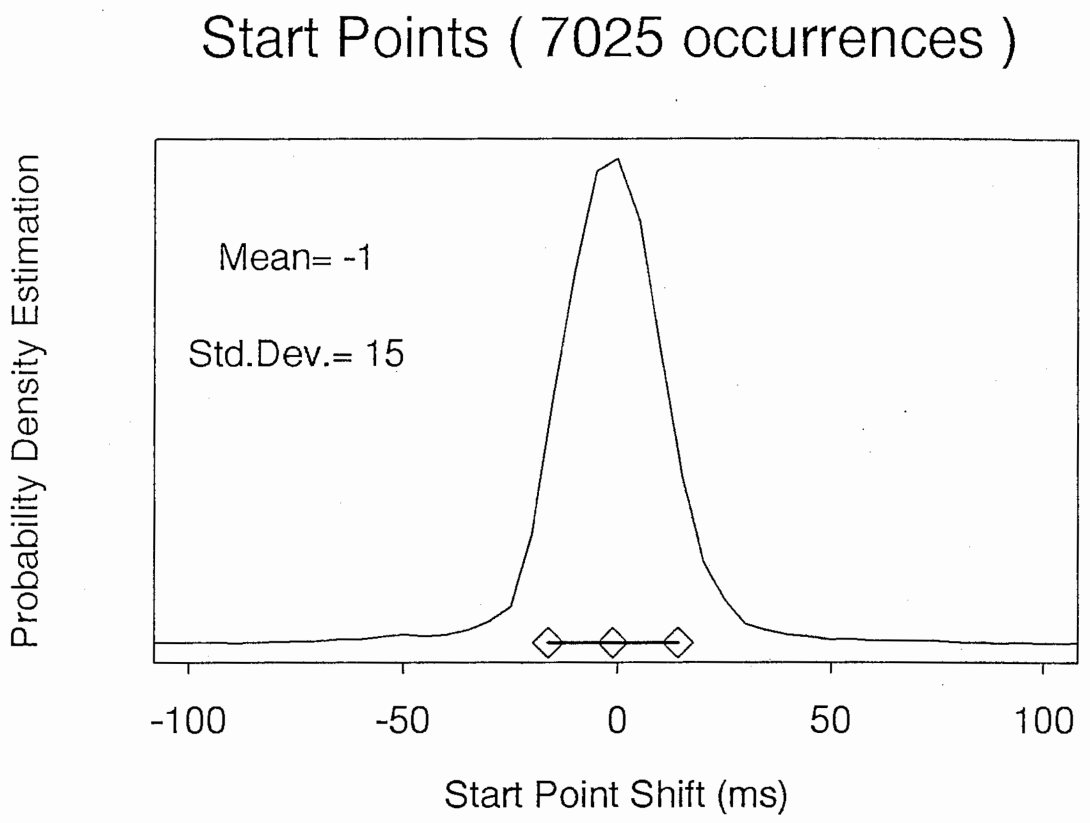
words

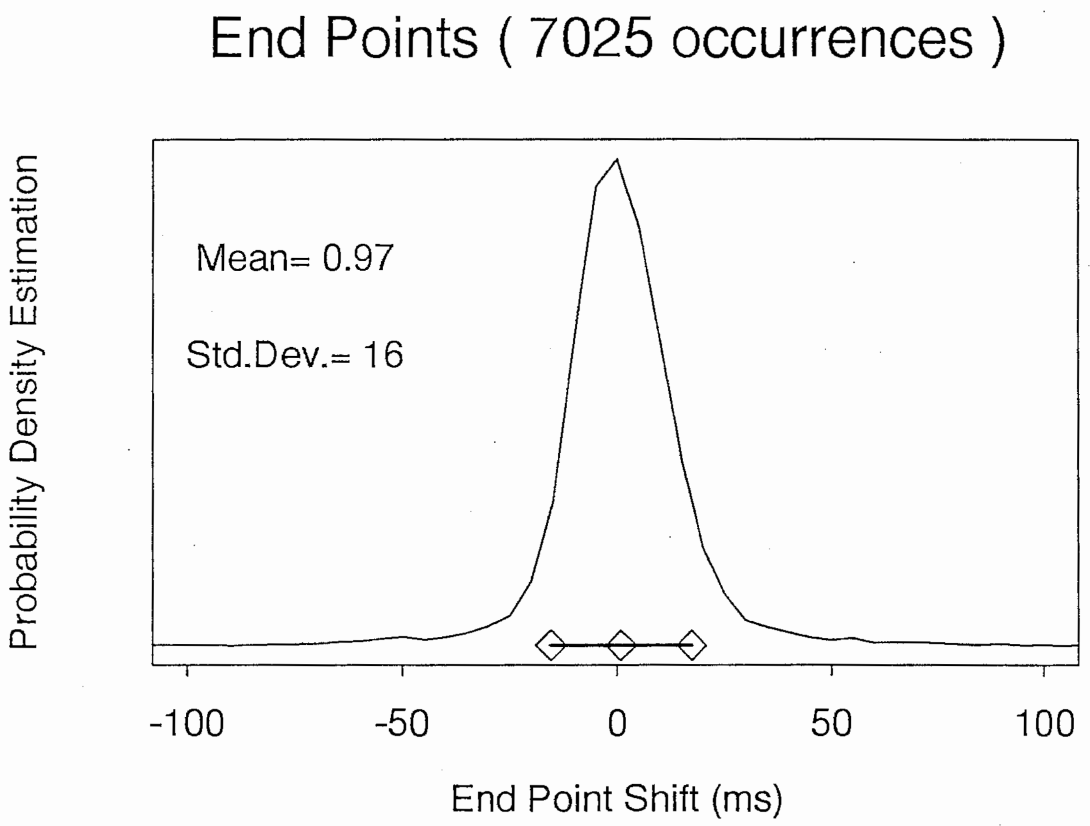
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3376	98.5	96.5	74.1	3.3	9.4	14.1
Vcons	1276	98.8	96.2	68.0	-4.7	10.7	14.7
Ucons	1521	99.0	95.1	54.2	-6.3	12.5	15.5
all	7025	98.7	96.3	69.0	-1.1	10.2	15.0

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3331	97.7	93.8	65.0	-0.4	11.9	17.5
Vcons	1306	98.2	95.9	70.1	5.8	10.4	14.9
Ucons	1445	99.9	99.9	91.1	1.5	5.4	7.8
all	7025	98.3	95.2	71.0	1.0	10.3	15.7

End points


Start Points (7025 occurrences)
Probability Density Estimation
Mean= -1
Std.Dev.= 15
-100
-50
0
50
100
Start Point Shift (ms)


End Points (7025 occurrences)
Probability Density Estimation
Mean= 0.97
Std.Dev.= 16
-100
-50
0
50
100
End Point Shift (ms)

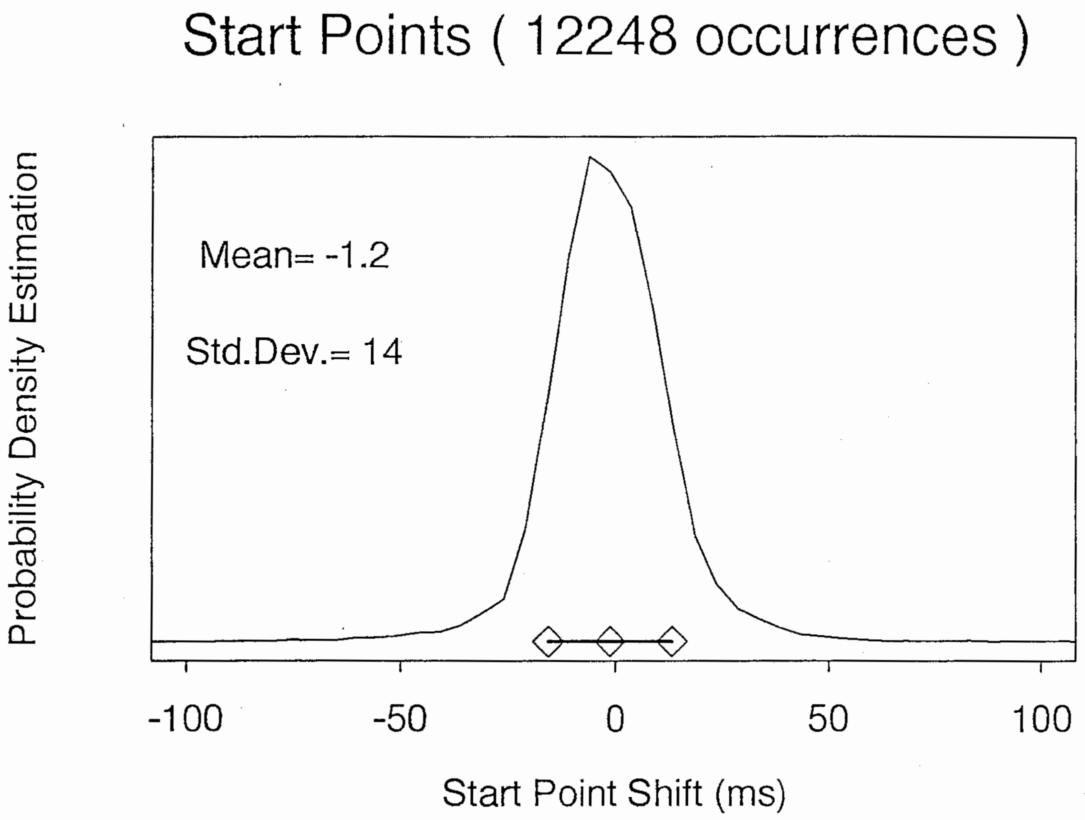
phrases

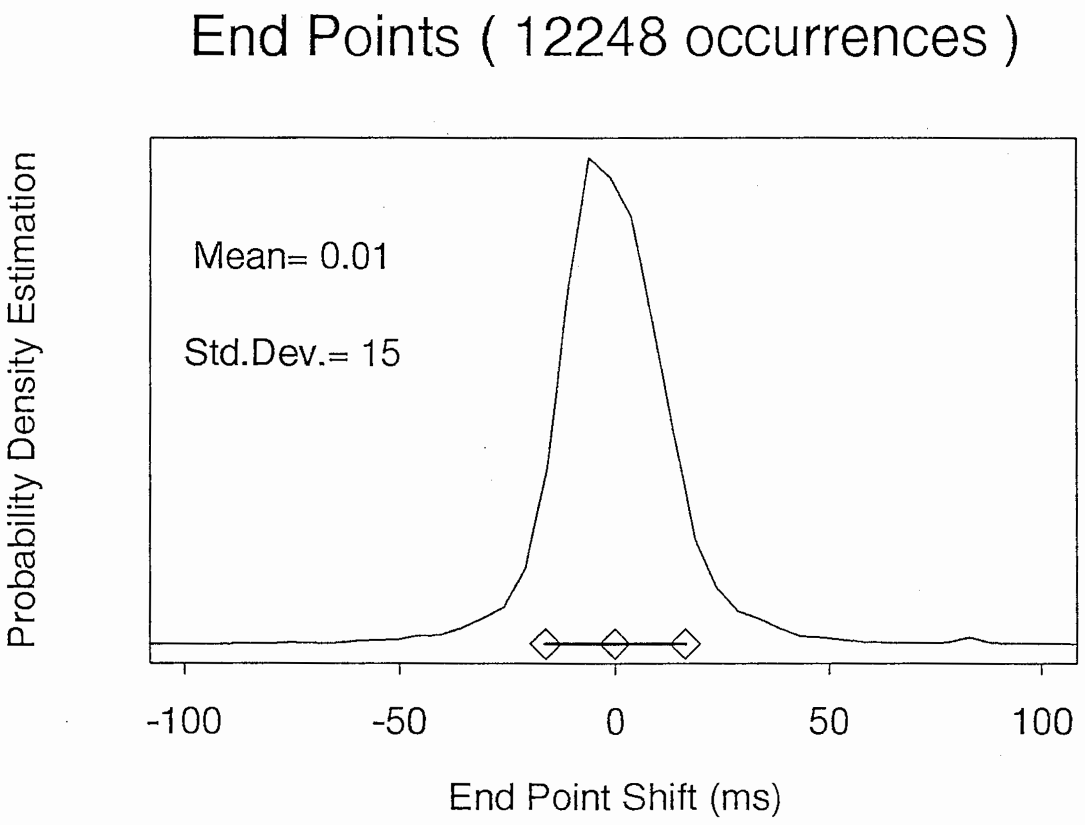
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	6255	99.2	96.4	70.9	2.8	9.6	13.5
Vcons	2385	98.9	96.5	63.6	-4.1	10.8	14.1
Ucons	2512	99.8	96.3	55.5	-8.3	12.0	12.7
all	12248	99.2	96.2	65.6	-1.2	10.6	14.5

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	6187	98.2	94.2	62.2	-1.0	11.9	17.1
Vcons	2466	99.2	96.4	66.9	4.7	10.2	13.1
Ucons	2297	100.0	99.9	85.8	1.1	6.1	7.8
all	12248	98.8	95.5	67.1	0.0	10.6	15.2

End points


Start Points (12248 occurrences)
Mean= -1.2
Std.Dev.= 14
Probability Density Estimation
-100
-50
0
50
100
Start Point Shift (ms)


End Points (12248 occurrences)
Mean= 0.01
Std.Dev.= 15
Probability Density Estimation
-100
-50
0
50
100
End Point Shift (ms)

sentences

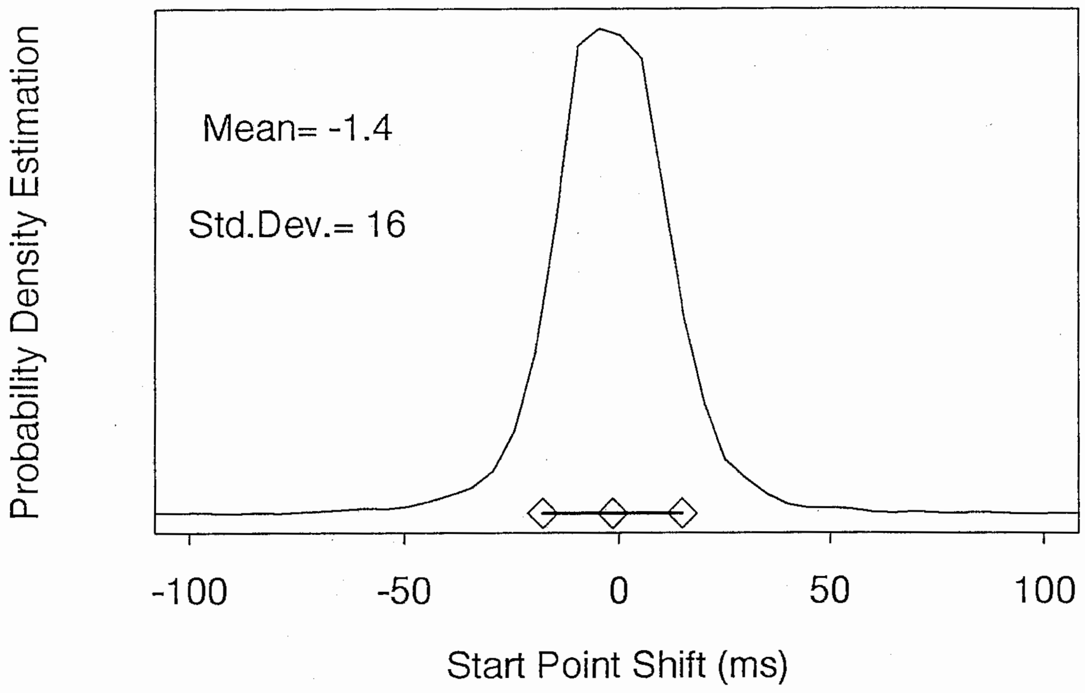
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3085	99.1	96.8	68.9	3.3	10.0	14.5
Vcons	1191	98.3	96.0	57.6	-7.0	12.8	17.9
Ucons	1248	98.9	93.0	56.0	-8.1	12.9	15.2
all	6046	98.9	95.7	63.7	-1.4	11.3	16.3

Start points

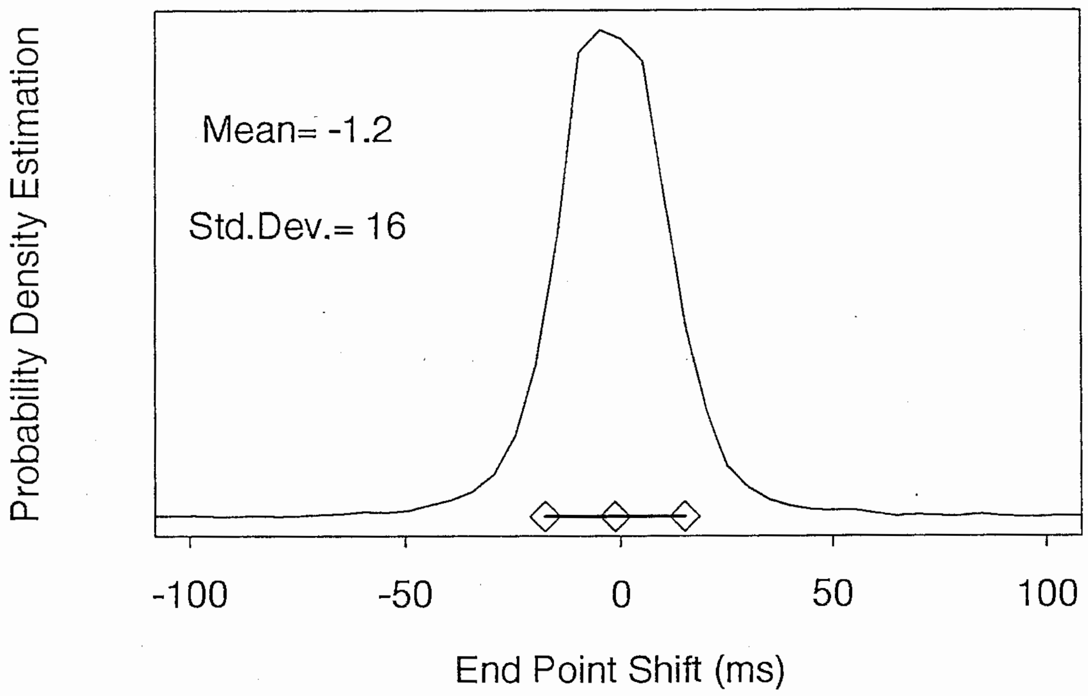
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3039	98.5	95.3	58.5	-4.1	12.4	17.1
Vcons	1233	99.0	97.0	68.4	4.5	10.2	15.6
Ucons	1146	99.7	99.3	81.0	2.2	7.3	9.6
all	6046	98.8	95.7	64.2	-1.2	11.2	16.3

End points

Start Points (6046 occurrences)



End Points (6046 occurrences)



- speaker MNM

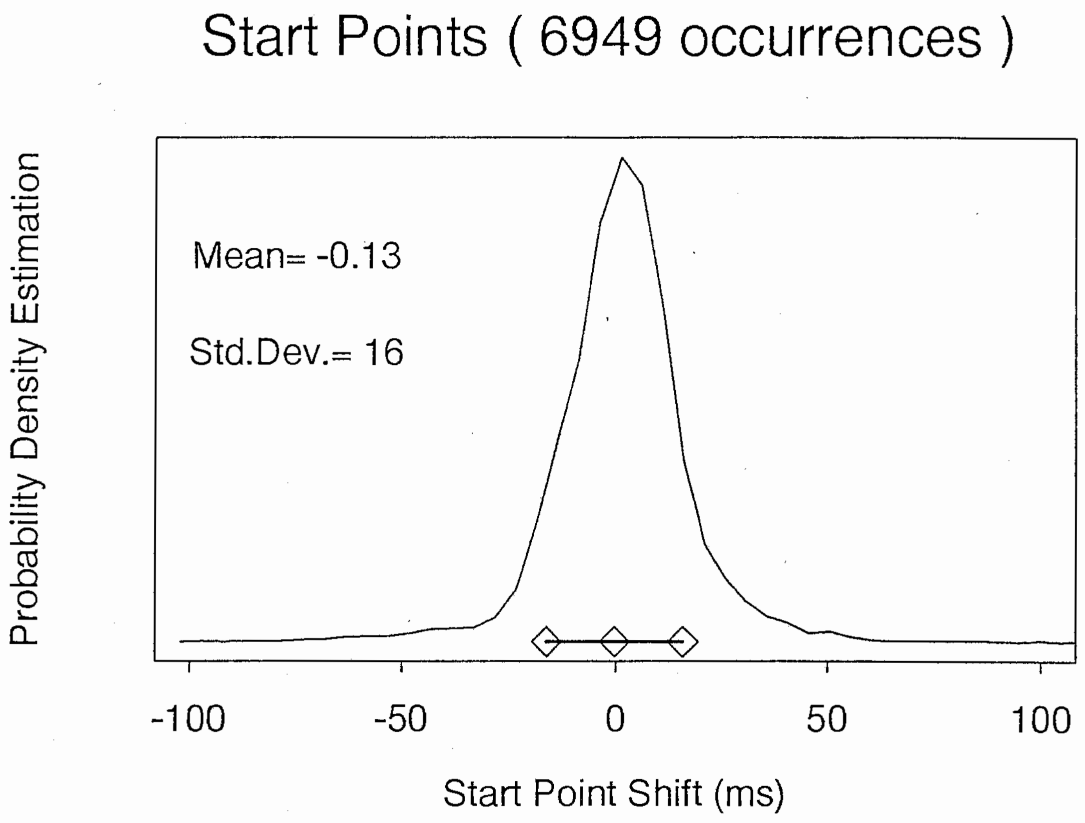
words

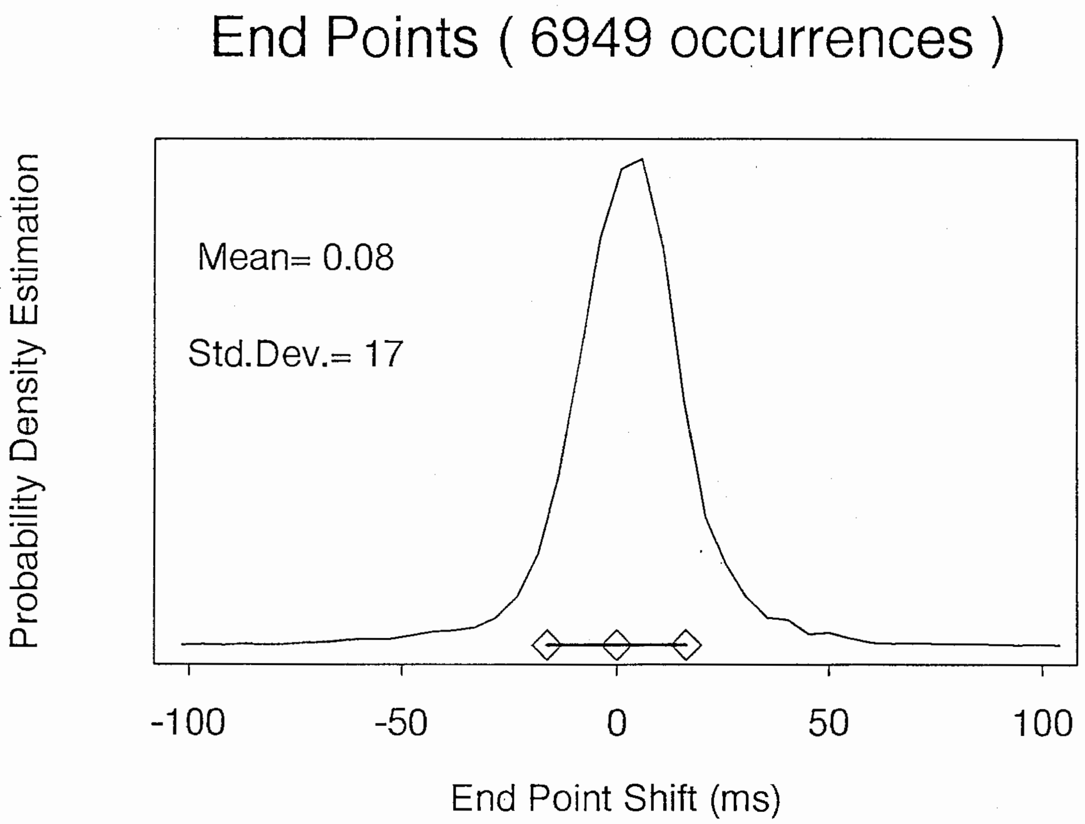
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3361	98.9	95.9	71.3	2.9	10.1	14.5
Vcons	1268	98.8	94.1	62.5	1.9	12.1	16.9
Ucons	1494	97.7	91.7	58.0	-8.5	13.4	16.6
all	6949	98.6	94.8	67.3	-0.1	11.1	16.0

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3314	98.3	93.5	61.8	-1.8	17.5	12.7
Vcons	1304	99.1	95.3	70.6	4.3	10.4	14.3
Ucons	1414	100.0	99.7	81.4	1.9	7.1	9.0
all	6949	98.4	94.1	66.2	0.1	11.6	16.6

End points


Start Points (6949 occurrences)
Probability Density Estimation
Mean= -0.13
Std.Dev.= 16
-100
-50
0
50
100
Start Point Shift (ms)


End Points (6949 occurrences)
Probability Density Estimation
Mean= 0.08
Std.Dev.= 17
-100
-50
0
50
100
End Point Shift (ms)

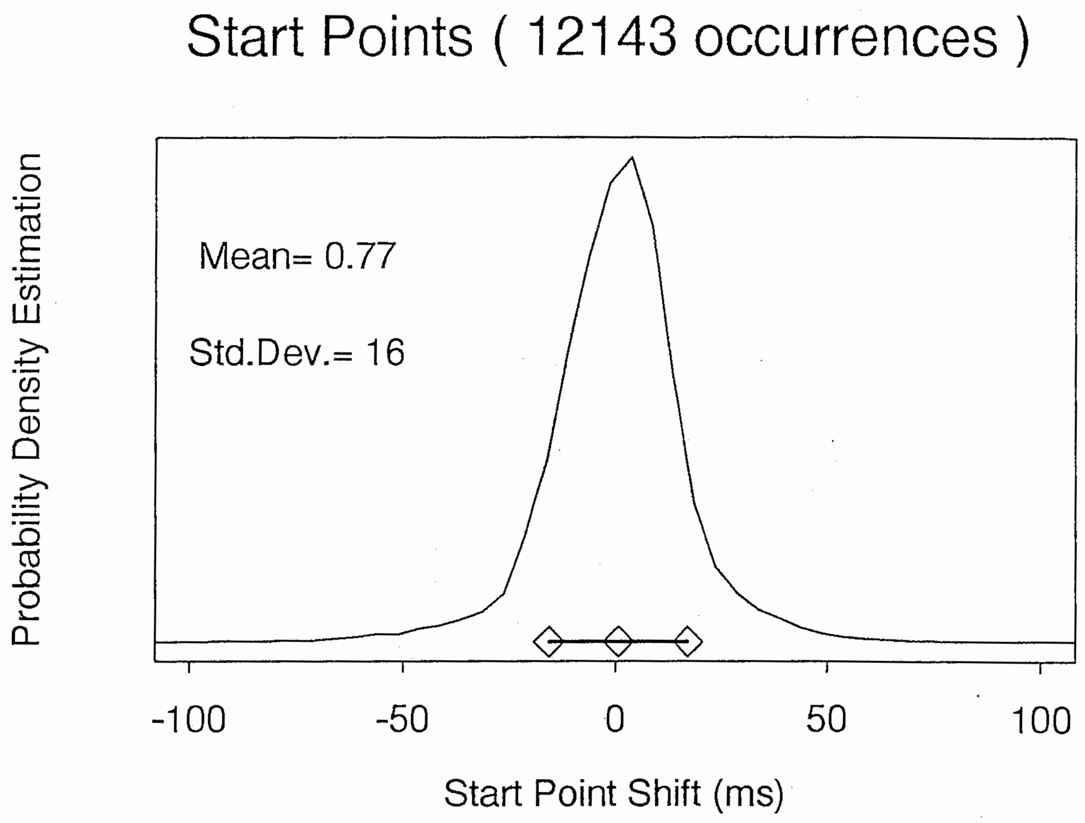
phrases

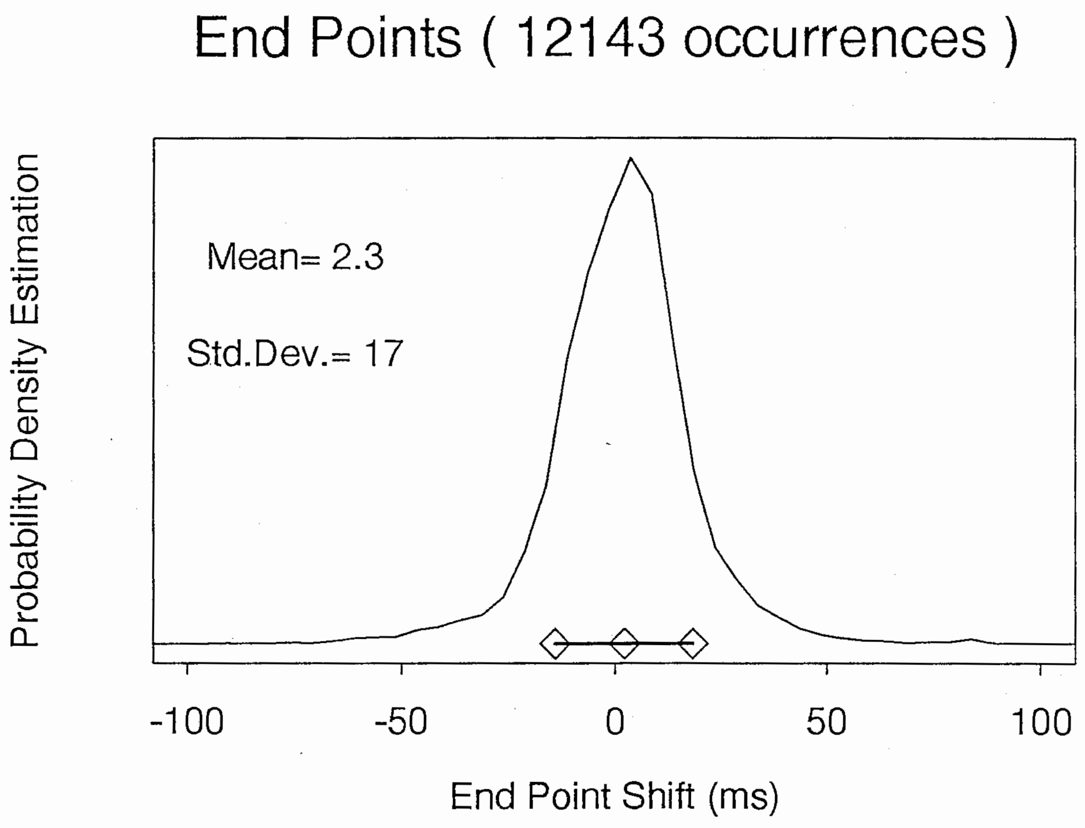
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	6233	99.2	95.8	69.5	4.8	10.5	13.9
Vcons	2312	99.1	93.5	63.3	0.2	12.0	16.4
Ucons	2486	98.4	92.8	56.8	-8.0	13.1	15.9
all	12143	98.7	94.3	65.5	0.8	11.6	16.5

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	6137	98.1	92.9	58.3	0.3	13.4	18.9
Vcons	2460	99.8	97.8	71.8	3.2	9.5	12.1
Ucons	2248	100.0	99.9	78.9	5.9	7.8	7.7
all	12143	98.6	93.9	63.4	2.3	12.1	17.0

End points


Start Points (12143 occurrences)
Mean= 0.77
Std.Dev.= 16
Probability Density Estimation
-100
-50
0
50
100
Start Point Shift (ms)


End Points (12143 occurrences)
Mean= 2.3
Std.Dev.= 17
Probability Density Estimation
-100
-50
0
50
100
End Point Shift (ms)

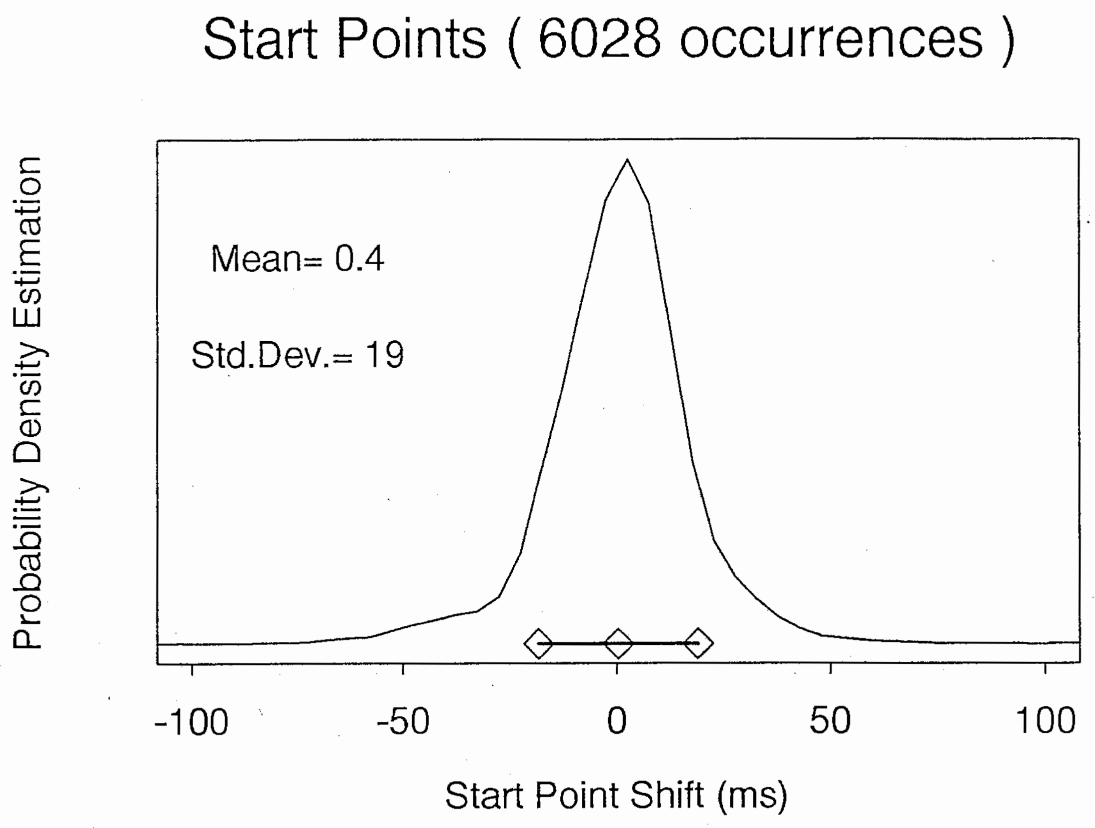
sentences

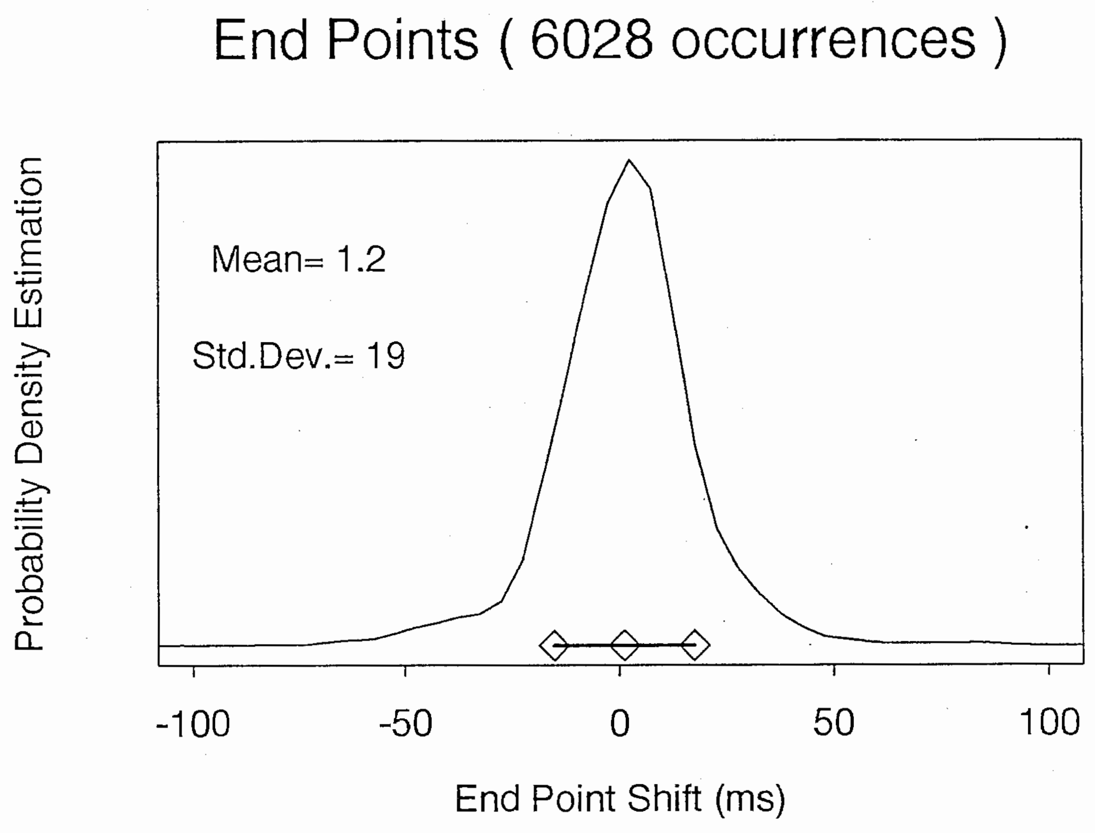
Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3076	99.1	95.4	65.1	5.4	11.4	15.3
Vcons	1162	97.8	93.3	67.0	-0.8	12.0	18.7
Ucons	1244	98.3	88.9	51.5	-9.8	14.8	17.8
all	6028	98.4	93.1	62.6	0.4	12.5	18.6

Start points

Model class	Nbr. of occ.	Segmentation rate (%) (Error ≤)			Mean error (ms)	Mean abs. error (ms)	Std. dev. (ms)
		50 ms	30 ms	10 ms			
vowels	3027	97.5	91.5	57.9	-2.2	14.0	20.9
Vcons	1232	99.4	97.6	70.9	3.9	9.9	13.9
Ucons	1134	100.0	99.4	75.5	7.3	8.5	7.8
all	6028	98.3	93.0	61.9	1.2	12.7	18.6

End points


Start Points (6028 occurrences)
Probability Density Estimation
Mean= 0.4
Std.Dev.= 19
-100
-50
0
50
100
Start Point Shift (ms)


End Points (6028 occurrences)
Probability Density Estimation
Mean= 1.2
Std.Dev.= 19
-100
-50
0
50
100
End Point Shift (ms)

4.4 Discussion

From the previous experiments, we can draw several conclusions:

- Generally, the segmentation results are very similar, whatever the kind of data used for the tests. The difference between words, phrases and sentences is usually very small (less than 1.5 % on average, when we consider an error of 30 ms).
  Yet, there are two cases where that trend is not confirmed.
  In case of single speaker training and testing on the same speaker, the results drop quite much from close words (tests on the training set)(98.2 % for 30 ms) to sentences (94.7 %) on start points. This model is very dependent on the training data.
  In case of multiple male-speaker training and female speaker testing, the trend is rather opposite: 88.2 % for words and 91.5 % for sentences. But this case is very particular, since we do not expect a female testing to have the same "behaviour" as a male testing.

- Multiple speaker trained models yield worse results than single spaker ones for MAU, but much better ones for MNM; MAU and MNM are both open speakers in the multiple case, which appears to reduce much the differences between two speakers.
  Moreover, it seems to be no need increase a lot the number of mixtures (which also increases a lot the computation time), since the results do not vary in a meaningful way between 3 and 9 mixtures, either for one speaker's results, or for the difference between two speakers. As soon as the results with 3 mixtures are already satisfactory, we can reasonably expect those with 9 mixtures not to increase a lot.
  Unfortunately, there are no results for one single Gaussian density model, but we can guess it would certainly not be enough to model the characteristics of different speakers. In such a case, we could assume that the results would be rather worse.
  However, we must not forget that all these spakers are professional speakers, so that the testing data are well-uttered. Maybe more mixtures would be necessary to test less-trained speakers.

  Besides, the continuous HMMs give better results than discrete ones, since, in case of single speaker training (MAU), discrete HMMs yield on average 98.0% and 93.9% as segmentation rates, respectively for 50 and 30 ms in the close word-testing, and 98.0% and 93.6% in the open word-testing.

- In the multiple training case, the difference between two speakers in the same testing conditions (open/close speaker) certainly comes from the speaking style, since this difference is generally rather small. As, even in the open speaker tests, the results are quite satisfactory, we can say that this kind of model has a good speaker-independency property.

- Distinguishing start points and end points leads to this conclusion, which agrees with results from other experiments: the best segmentation rate is reached for vowels and the worst one for unvoiced consonants considering the start points, while the trend is exactly opposite for end points. The start points of vowels are usually slightly shifted towards the right side (around 3 ms), while the start points of unvoiced consonants are rather shifted towards the left side (around 6-7 ms). As for the end points of the vowels, the error seems to have a bigger standard deviation than the other phoneme classes.

5 Word-spotting experiments

5.1 Introduction

Further speech recognition experiments will require data from continuous and spontaneous speech.
The purpose of this study is thus to determine as precisely as possible the boundaries of

predefined keywords in continuous speech. We will assume in the following experiments that, in the tested sentences, we know

- if
- which
- in which order

keywords were uttered.

The eight keywords chosen in these experiments are:
- denwa
- honyaku
- kaigi
- kokusai
- moushikomi
- touroku
- tsuuyaku
- sanka

The tests are performed on 25 sentences from speaker MAU (set DSC3).

5.2 The models

5.2.1 The keywords

All the keywords will be considered as a sequence of phonemes, modelled like in the previous part with 3 state, 3 mixture, multiple male speaker-trained models.

5.2.2 The background speech

Any part of the utterance different from a keyword will be considered as "background speech", or "garbage". Background speech will be modelled with a "Garbage Model", or "Filler Model"[7]. Assuming that we already have good models for the phonemes from the keywords, we will focus on the garbage models. We will apply different kinds of grammars to the segmentation system (to HVite, if using HTK), according to the different garbage models.
The following garbage models were used:
- Type 1
  1 model for all phonemes with 1 state, 27 mixtures, trained with word-utterances (8 speakers).
- Type 2
  1 model for all phonemes with 3 states, 138 mixtures, trained with word-utterances, built with merging all the phoneme models (use of HMerge from HTK).
- Type 3
  1 model of all phonemes with 3 states, 138 mixtures, trained with 90 sentence utterances (SC1, SC2, SC3) from 8 male speakers.
- Type 4
  any of the 48 phoneme models (3 states, 3 mixtures, trained with words).

Let us assume that one testing sentence contains the only keyword "kaigi", and denote as "GB" the garbage model. The types of grammars given to the system are:
- Type 1
  (GB k a i g i GB)
  In that case, we only allow one repetition of the garbage model before the keyword.
- Type 2
  $\$k = k|k0$
  $\$GBM = GB|+$

$(< \$GBM > \$$k a i g i $< \$GBM >)$
This grammar allows several repetitions of the garbage model.

- Type 3
  $\$phn = a|aa|i|ii|...$
  $\$GBM =< \$phn >$
  $(< \$GBM > \$$k a i g i $< \$GBM >)$

The following experiments have been performed:

grammar \ model	1	2	3	4
1	■			
2	■	■	■	
3				■

■ : **experiment performed**

5.3 Results

KEYWORD	nbr.	50 ms	30 ms	10 ms	mean err.	mean. abs.err.	std.dev.
kaigi	14	85.7	85.7	78.6	-9.5	13.0	25.2
kokusai	12	75.0	58.3	41.7	-34.0	34.0	38.3
denwa	8	100	100	75.0	-8.8	8.8	3.5
sanka	7	100	100	57.1	-5.4	11.1	11.3
touroku	5	100	80.0	80.0	-10.5	14.5	20.6
moushikomi	7	71.4	57.1	14.3	10.7	76.4	136.6
tsuuyaku	6	83.3	16.7	16.7	-13.8	34.6	40.1
honyaku	5	60.0	60.0	40.0	-362.5	707.3	372.5

Start Points

kaigi	14	71.4	71.4	42.9	24.5	33.0	48.5
kokusai	12	100	100	91.7	0.2	4.0	5.9
denwa	8	87.5	87.5	75.0	61.3	63.8	159.4
sanka	7	85.7	85.7	71.4	9.3	17.9	31.3
touroku	5	40.0	40.0	40.0	196.0	196.0	229.1
moushikomi	7	85.7	71.4	28.6	92.9	95.0	200.0
tsuuyaku	6	100	100	66.7	-8.3	8.3	4.1
honyaku	5	60.0	60.0	60.0	-342.5	342.5	674.7

End points

(model 1, grammar 1)

KEYWORD	nbr.	50 ms	30 ms	10 ms	mean err.	mean. abs.err.	std.dev.
kaigi	14	85.7	85.7	78.6	-9.5	13.0	25.2
kokusai	12	75.0	66.7	41.7	-32.3	32.3	38.2
denwa	8	100	100	75.0	-8.8	8.8	3.5
sanka	7	100	100	51.7	-5.4	11.1	11.3
touroku	5	100	80.0	80.0	-10.5	14.5	20.6
moushikomi	7	71.4	57.1	14.3	10.7	76.4	136.6
tsuuyaku	6	83.3	50.0	50.0	-2.1	22.9	35.7
honyaku	5	60.0	60.0	40.0	-72.5	82.5	112.7

Start Points

kaigi	14	71.4	71.4	42.9	24.5	33.0	48.5
kokusai	12	100	100	91.7	0.2	4.0	5.9
denwa	8	87.5	87.5	75.0	60.0	62.5	155.9
sanka	7	85.7	85.7	71.4	9.3	17.9	31.3
touroku	5	40.0	40.0	40.0	196.0	196.0	229.1
moushikomi	7	85.7	71.4	28 .6	92.9	95.0	200..0
tsuuyaku	6	100	100	66.7	-8.3	8.3	4.1
honyaku	5	80.0	80.0	60.0	-37.5	37.5	58.9

End points

(model 1, grammar 2)

keyword	nbr.	50 ms	30 ms	10 ms	mean err.	mean. abs.err.	std.dev.
kaigi	14	92.9	92.9	78.6	-4.5	10.2	20.1
kokusai	12	91.7	75.0	58.3	-19.8	20.2	28.3
denwa	8	100	100	75.0	-8.8	8.8	3.5
sanka	7	100	100	71.4	-1.8	10.4	13.0
touroku	5	100	100	40.0	4.5	16.5	19.2
moushikomi	7	100	71.4	42.9	-17.9	20.7	17.9
tsuuyaku	6	100	100	50.0	-3.8	11.3	14.6
honyaku	5	80.0	80.0	60.0	-26.5	34.5	66.7

(%) (ms)

Start Points

kaigi	14	71.4	64.3	14.3	15.9	39.1	51.9
kokusai	12	100	100.0	91.7	1.5	5.2	9.7
denwa	8	87.5	87.5	75.0	30.0	30.6	75.0
sanka	7	85.7	85.7	42.9	15.0	22.9	34.8
touroku	5	80.0	60.0	40.0	36.0	37.0	48.3
moushikomi	7	100	85.7	57.1	7.1	13.9	21.5
tsuuyaku	6	100	100	66.7	-8.3	8.3	4.1
honyaku	5	100	80.0	60.0	-14.5	14.5	10.7

End points

(model 2, grammar 2)

KEYWORD	nbr.	50 ms	30 ms	10 ms	mean err.	mean. abs.err.	std.dev.
kaigi	14	85.7	85.7	78.6	70.2	82.7	227.0
kokusai	12	58.3	50.0	33.3	-126.5	126.9	158.3
denwa	8	100	100	75.0	-8.8	8.8	3.5
sanka	7	100	100	85.7	-1.1	11.1	14.6
touroku	5	100	100	40.0	4.5	16.5	19.2
moushikomi	7	85.7	42.9	14.3	-38.6	42.9	43.9
tsuuyaku	6	83.3	83.3	50.0	2.1	18.8	30.8
honyaku	5	20.0	20.0	20.0	187.5	412.5	738.0

Start Points

kaigi	14	57.1	57.1	42.9	113.8	116.3	304.2
kokusai	12	100	100	91.7	1.5	5.2	9.7
denwa	8	87.5	87.5	75.0	62.5	63.1	160.8
sanka	7	71.4	71.4	57.1	23.6	30.0	39.2
touroku	5	40.0	40.0	20.0	263.0	263.0	241.6
moushikomi	7	100	71.4	57.1	4.3	16.1	25.4
tsuuyaku	6	100	100	66.7	-8.3	8.3	4.1
honyaku	5	80.0	60.0	60.0	309.5	338.5	732.7

End points

(model 3, grammar 2)

KEYWORD	nbr.	50 ms	30 ms	10 ms	mean err.	mean. abs.err.	std.dev.
kaigi	14	100	100	92.9	1.6	4.8	9.0
kokusai	12	100	100	91.7	-5.6	6.0	5.9
denwa	8	100	100	75.0	-8.8	8.8	3.5
sanka	7	100	71.4	71.4	7.5	15.4	21.0
touroku	5	100	100	40.0	6.5	16.5	18.8
moushikomi	7	100	100	57.1	-4.3	8.6	10.1
tsuuyaku	6	83.3	83.3	33.3	5.4	20.4	30.9
honyaku	5	100	100	60.0	5.5	12.5	13.5

Start Points

kaigi	14	78.6	71.4	7.1	4.8	37.0	49.3
kokusai	12	100	100	91.7	1.5	5.2	9.7
denwa	8	87.5	87.5	62.5	-26.9	30.6	72.4
sanka	7	100	100	71.4	-0.7	12.1	15.0
touroku	5	100	80.0	40.0	16.0	20.0	20.9
moushikomi	7	100	85.7	71.4	3.6	11.4	21.2
tsuuyaku	6	100	100	66.7	-8.3	8.3	4.1
honyaku	5	100	80.0	60.0	-16.5	16.5	9.9

End points

(model 4, grammar 3)

5.4 Discussion

Because of the very few number of occurrences for each keyword, it is difficult to draw general conclusions from those experiments. There does not seem to be any real optimal garbage model. However, it seems that several repetitions of the garbage model yield better results than only one (with many self-loops). Besides, the results are very dependent on the keywords; for example, it is natural that the boundaries of "honyaku" should not be determined very accurately, because of the"h" and unvoiced "u" sounds; there seems to be no real correlation between the length of the keywords and the segmentation results. On the other hand, a keyword in a long sentence will certainly be relatively badly segmented ("honyaku", in sentence SC23, for example).

5.5 Future research

The previous experiments were preliminary studies on keyword-segmentation. The next step would maybe consist in increasing the size of the testing set, so that the number of occurrences for each keyword would enable more conclusions. Although, on average, the models 2 and 4 (respectively associated with grammars 2 and 3) seem to give the best results, some experiments with garbage models of the first type with more mixtures (64 or 128) should be performed. Moreover, it could be interesting to investigate the case where phoneme class garbage models are used.

6 Conclusion

This report showed that the use of continuous HMMs gives good results in phoneme segmentation. Multiple male speaker training can successfully be applied, in a rather speaker-independent way, to male speaker testing. The first steps of keyword segmentation have been investigated, and show that the main problem is to find a satisfactory means of modelling "non-keyword" speech.

References

[1] X.D.Huang, Y.Ariki, M.A.Jack, *Hidden Markov Models for Speech Recognition*, Edinburgh University Press, 1990.

[2] L.R.Rabiner, B.H.Juang, *An introduction to Hidden Markov Models*, IEEE ASSP Magazine, January 1986.

[3] L.R.Rabiner, B.H.Juang, S.E.Levinson, M.M.Sondhi, *Recognition of Isolated Digits Using Hidden Markov Models With Continuous Mixture Densities*, AT&T Technical Journal, July-August 1985.

[4] L.R.Rabiner, B.H.Juang, S.E.Levinson, M.M.Sondhi, *Some Properties of Continuous Hidden Markov Model Representations*, AT&T Technical Journal, July-August 1985.

[5] L.R.Rabiner, R.W.Schafer, *Digital Processing of Speech Signals*, Prentice Hall Signal Processing Series, 1978.

[6] S.J.Young, *HTK: Hidden Markov Model Toolkit V1.2, Installation Guide*, Cambridge University Engineering Department, Speech Group, December 1990.

[7] Richard C.Rose and Douglas B.Paul, *A Hidden Markov Model Based Keyword Recognition System*, Lincoln Laboratory, MIT, 1990.

ACKNOWLEDGEMENTS

I would like to thank ATR Interpreting Telephony Laboratories for enabling me to achieve this research, and especially Dr. Akira Kurematsu for welcoming me in these laboratories, Shigeki Sagayama, Dr. Masahide Sugiyama, Yasuhiro Komori for their continuous support, Dr. David Rainton for his explanations about the softwares, and Harald Singer for his valuable help and comprehension.

APPENDIX:

Final Talk

Automatic Phoneme Segmentation

Using

Continuous Hidden Markov Models

1. Phoneme labelling experiments

goal: building automatically large speech databases.

$\longrightarrow$ evaluation of automatic segmentation,

compared to hand-segmentation.

2. Word-spotting experiments

goal: keyword-segmentation in continuous speech.

Phoneme Labelling

Evaluation method

Reference = hand-labelling (boundaries)

Measurements: segmentation rate (50, 30, 10 ms comparison window)

mean error

mean absolute error

standard deviation

probability density function of error

error = automatic boundary - hand boundary

distinction between start points and end points.

TOOLS

48 models including:

46 phoneme models

silence

word-medial pause

Continuous Mixture Density HMMs

$\longrightarrow$ use of HTK software

3 states

Diagonal covariance matrices

HMMs always trained by word-utterances using labels

Phoneme Models

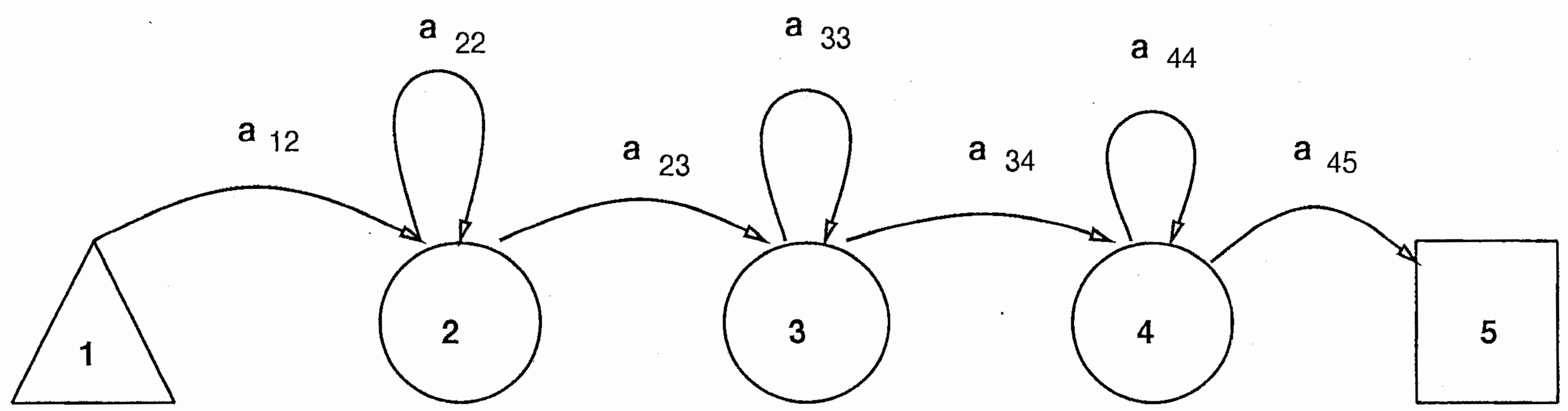
-vowels:	a, aa, e, ee, i, ii, o, oo, u, uu , N
-voiced consonants:	b, b0, d, d0, g, g0, m, m0, n, n0, r, r0, z, z0
-unvoiced consonants:	ch, ch0, k, k0, p, p0, s, sh, ss, ssh, t, t0, ts, ts0
-other phonemes:	Ui, Uu, h, w, w0, y, y0
-pause:	Q
-silence:	+

after Q or + $\xrightarrow{\qquad}$ "0"

automatic label conversion (conversion table):

k-i-k-a-i $\xrightarrow{\qquad}$ k0-Ui-k-a-i

g-o-z-a-i-m-a-s-u $\xrightarrow{\qquad}$ g0-o-z-a-i-m-a-s-Uu



THE MODEL

The coding conditions

- 5 ms frame shift

-26 analysis coefficients:

12 cepstra (LPC)

12 delta cepstra

log power

delta log power

Main questions

Influence on segmentation results of:

- The type of tested data

  $\longrightarrow$ experiments on words

  bunsetsu

  sentences

- The training conditions:

  - single / multiple speaker training
  - number of mixtures

- The testing conditions

  tests on different speakers (male and female)

  $\longrightarrow$ speaker dependency?

  independency?

- The segmented phoneme

Testing conditions

Training conditions

		MAU	MNM	MHT	MTK	FSU
single		C	O			
multiple	3 mixt.	O	O	C	C	O
	9 mixt.	O	O			

Single speaker training: MAU (2620 words)

Multiple speaker training: MHT MMS MMY MSH
MTK MTM MTT MXM
(8 * 1310 words)

"C":	close speaker
"O":	open speaker

tests on words, phrases (sets DSA and DSB), and sentences (set DSC).

General points

Very similar results on words, bunsetsu and sentences

(difference < 1.5 % on average for the segmentation rate)

EXCEPT

single male-speaker training

tests on the same speaker:

close words: 98.2 %

sentences: 94.7 %

on start points with a 30 ms comparison window

multiple male speaker training

tests on a female speaker

words: 88.2 %

sentences: 91.5 %

Single/Multiple speaker training

	MAU	MNM
Single	97.1%	89.6%
Multiple (3 mixt.)	95.8%	94.1%

$\longrightarrow$ speaker dependent

$\longrightarrow$ speaker independent

Number of mixtures (bunsetsu results)

	MAU	MNM
3 mixt.	95.8%	94.1%
9 mixt.	96.2%	94.3%

no significant change for one speaker

the difference between two speakers does not change much.

General results (speaker independency?)

	MHT	MTK	MAU	MNM	FSU
Single			97.1	89.6	
Multiple	97.5	96.2	95.8	94.1	90.2

(bunsetsu results)

single speaker-trained model $\longrightarrow$ speaker-dependent
(data dependant)

multiple speaker-trained model $\longrightarrow$ good (male) speaker-independency

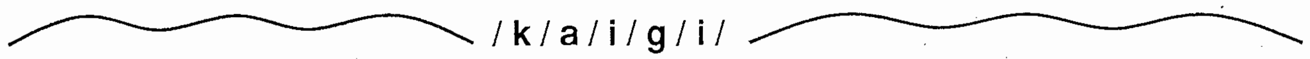
Influence of the phoneme on segmentation results:

	Best	Worst
Start Points	vowels	Unvoiced csnts.
End points	Unvoiced csnts.	vowels

Word-spotting

determine as precisely as possible

the boundaries of predefined "keywords" in continuous speech (sentences).



assumption:

we know	if	keywords were uttered.
	which	
	in which order	

set of 8 keywords:	deNwa
	hoNyaku
	kaigi
	kokusai
	moushikomi
	touroku
	tsuuyaku
	saNka

tests on 25 sentences (DSC3) from speaker MAU

MODELS

KEYWORDS:

set of 48 phoneme models

3 states

3 mixtures

multiple male-speaker training

MODELS

BACKGROUND SPEECH:

Different types of "Garbage models" have been tried.

- 1 model for all phonemes with 1 state, 27 mixtures

  trained with word-utterances.

- 1 model for all phonemes with 3 states, 138 mixtures

  trained with word-utterances

- 1 model for all phonemes with 3 states, 138 mixtures

  trained with 90 sentence utterances (SC1, SC2, SC4) from 8 speakers.

- 48 phoneme models with 3 states, 3 mixtures

  trained with words.

Different kinds of grammars

(allowing or not some repetitions of garbage models)

keyword	nbr.	50 ms	30 ms	10 ms	mean err.	mean. abs.err.	std.dev.
kaigi	14	92.9	92.9	78.6	-4.5	10.2	20.1
kokusai	12	91.7	75.0	58.3	-19.8	20.2	28.3
denwa	8	100	100	75.0	-8.8	8.8	3.5
sanka	7	100	100	71.4	-1.8	10.4	13.0
touroku	5	100	100	40.0	4.5	16.5	19.2
moushikomi	7	100	71.4	42.9	-17.9	20.7	17.9
tsuuyaku	6	100	100	50.0	-3.8	11.3	14.6
honyaku	5	80.0	80.0	60.0	-26.5	34.5	66.7

(%) (ms)

Start Points

kaigi	14	71.4	64.3	14.3	15.9	39.1	51.9
kokusai	12	100	100.0	91.7	1.5	5.2	9.7
denwa	8	87.5	87.5	75.0	30.0	30.6	75.0
sanka	7	85.7	85.7	42.9	15.0	22.9	34.8
touroku	5	80.0	60.0	40.0	36.0	37.0	48.3
moushikomi	7	100	85.7	57.1	7.1	13.9	21.5
tsuuyaku	6	100	100	66.7	-8.3	8.3	4.1
honyaku	5	100	80.0	60.0	-14.5	14.5	10.7

End points

(138 mixtures, trained with words)

KEYWORD	nbr.	50 ms	30 ms	10 ms	mean err.	mean. abs.err.	std.dev.
kaigi	14	85.7	85.7	78.6	70.2	82.7	227.0
kokusai	12	58.3	50.0	33.3	-126.5	126.9	158.3
denwa	8	100	100	75.0	-8.8	8.8	3.5
sanka	7	100	100	85.7	-1.1	11.1	14.6
touroku	5	100	100	40.0	4.5	16.5	19.2
moushikomi	7	85.7	42.9	14.3	-38.6	42.9	43.9
tsuuyaku	6	83.3	83.3	50.0	2.1	18.8	30.8
honyaku	5	20.0	20.0	20.0	187.5	412.5	738.0

Start Points

kaigi	14	57.1	57.1	42.9	113.8	116.3	304.2
kokusai	12	100	100	91.7	1.5	5.2	9.7
denwa	8	87.5	87.5	75.0	62.5	63.1	160.8
sanka	7	71.4	71.4	57.1	23.6	30.0	39.2
touroku	5	40.0	40.0	20.0	263.0	263.0	241.6
moushikomi	7	100	71.4	57.1	4.3	16.1	25.4
tsuuyaku	6	100	100	66.7	-8.3	8.3	4.1
honyaku	5	80.0	60.0	60.0	309.5	338.5	732.7

End points

(138 mixtures, trained with sentence-utterances)

Difficult to find an optimal model

The results also depend on:

-the keywords (length, type of initial and final phonemes)

-the sentence (length)

-the grammar

(GB $key GB) < (<GB> $key <GB>)

Future research

- extend the number of keywords

- enlarge the testing set (the number of occurrences for each keyword)