Internal Use Only (非公開)
002

TR-IT-0151

Toward a signal processing module

Jean-Christophe L'Hériteau

1996.1

ABSTRACT
Within the system the work was exclusively in the final stages. The main goal was to improve the waveform synthesis modules in CHATR. Using those f0, duration and power information, the Unit Selection module selects the most appropriate unit from the data base to find. However, it is unlikely that a unit with the exact same f0, duration and power will exist in the data base. For that reason we use signal processing to smooth and improve the quality of the final output. For this we use PSOLA. The work reported here consisted in improving this PSOLA module. There were two main axes of research: a) improving the CHATR signal processing module, and b) improving the input to this signal processing module.

©ATR Interpreting Telecommunications
Research Laboratories

©ATR 音声翻訳通信研究所

Toward a signal processing module

Jean-Christophe L'Heriteau

January 31, 1996

Abstract

Within the system the work was exclusively in the final stages. The main goal was to improve the waveform synthesis modules in CHATR. As we all know, CHATR is given some input, typically some text but not necessarily. This text is transformed in a phoneme string. All the available informations goes through the predictions modules, which put outs the f0, duration and power(doesn't exist yet in fact). Using those informations, the Unit Selection module go trough the data base to find the more appropriate unit. However, the unit with the specific f0, duration and power never exists in the data base. For that reason one have to use signal processing tools to smoothen and better the quality of the final output. One of those tool, the one ATR choose to use, is PSOLA.
The work consisted in improving this PSOLA module. In that respect, there to main axes of research:

- Improving the signal processing module
- Improving the input to this signal processing module

Contents

Chapter 1

General presentation

1.1 CHATR

1.1.1 Introduction to CHATR

CHATR is a generic speech synthesis system developed in Dept 2 of ITL at ATR. CHATR offers a uniform environment for developing speech synthesis systems. It offers a simple but powerful architecture for utterances which modules can create and modify. Many modules are currently included offering such facilities as multiple phonemes sets, lexicons, alternative durations modules, intonation systems, as well as multiple methods for waveform synthesis. CHATR also offers display mechanisms for utterances. Parameters and flow of control may done through CHATR's Lisp command interface (either interactively or through files). CHATR also offers text-to-speech for those who are not interested in the internal methods.

CHATR was developed as a generic speech synthesis system. Rather than building a single synthesis system, CHATR was designed to be modular in a way that easily allows a large number of different modules to interact in a defined way. Multiple modules performing equivalent tasks, such as waveform synthesis, duration prediction and intonation are included within the same environment allowing close direct comparison.

CHATR is still under development but already offers a number of synthesis methods (Klatt formant synthesis, non-uniform unit concatenation, and diphone concatenation) and is easily configurable. However, the mean axe of research today is the non-uniform unit concatenation. All the work related in this paper was done in that area.

CHATR can best be thought of as a general system in which *modules* act on *utterances*. Each utterance has a complex internal structure. A module may access (read or write) any part of an utterance though typically they read many parts but write only one type of information. A typical module may be something like a waveform synthesizer, something that takes a description of phonemes, durations and fundamental frequency and generates a waveform. Or it may be something that takes words and looks them up in a lexicon returning their pronunciation. Each module may have its own internal structures if required but its communication with the rest of the system is via an utterance object.

Basically, the overall synthesizer system takes an utterance object where most of its levels are not yet filled in. Various modules are called (depending on parameters) that fill in the other levels of the utterance, eventually leading to a waveform (if requested) that can be played by various mechanisms.

1.1.2 The signal processing module

This paper is concerned with only a small part of the whole speech synthesis process, divided in three major part:

- Text processing (parsing, tagging, phrasing etc.)
- Linguistic processing (prosody and segmental prediction)
- Waveform synthesis (unit selection and signal processing)

Given an input, the system add whatever information it hasn't got to the utterance. After having tagged the phrases, and the word within the phrase (whether it is a verb or a noun), the system look them up in a lexicon to have a phoneme stream. Then the F0 contour of the whole utterance and the duration of each phoneme are predicted. Based on those target values, a module goes through a speech database and select the best fitting units to this target segments.
Even-though this paper doesn't discuss how to select the best possible units, as the database used for selection will always be finite, even the best selection will not in general exactly match the desired utterance. Further signal processing will be required to modify the selected units. We are currently using PSOLA-based techniques to modify the final selection. However the closer the selected units are to the target segments the less signal processing will be required—and hence less distortion will be introduced.
A number of methods are provided for concatenating selected units.

- DUMB

  Simple abutted the selected units together with **no** processing whatsoever.

- DUMB+

  (So-called dumb plus) offers a slightly more sophisticated method of joining but without any real signal processing. The default joins edges by over lapping them and finding the minimum distance between a small window of samples. Other possible join methods are at zero crossings or at pitch marks.

- NUUCEP or CEPLMA

  These are two names for the same process. These specify that the waveform signal is to be reconstruction from improved cepstrum vectors as in the original NUUTALK system. The LMA filter allows modification of pitch and duration. thus allowing the signal to match the target pitch and segments (within on pitch frames 5ms). This concatenation method requires there to be cepstrum files for the database. This method is slow and produces rather buzzy output but it does work reliably.

- PS_PSOLA

  A simple implementation of the time domain pitch synchronous overlap and add method of joining units as described in [2].

- NULL

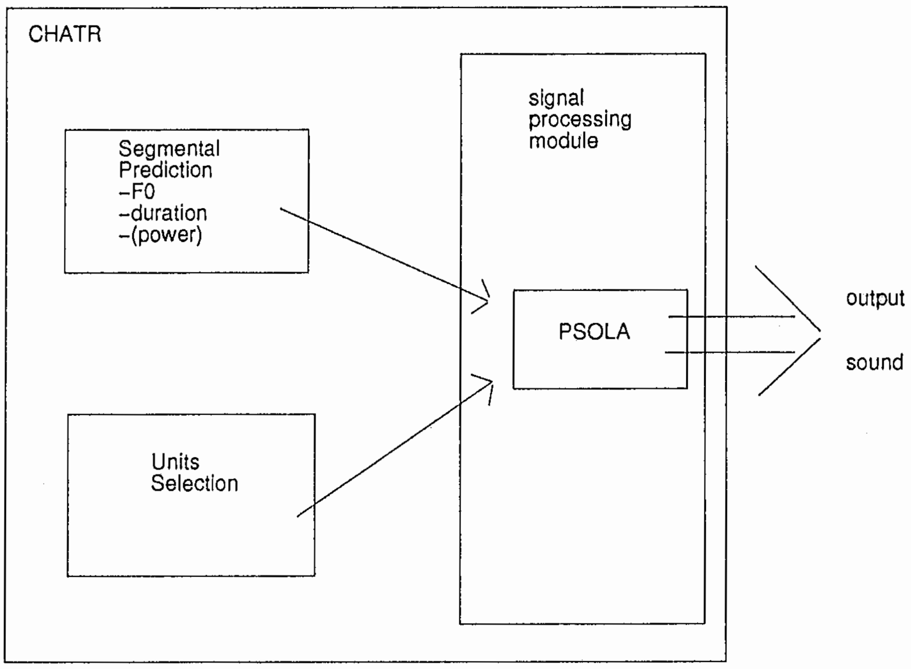
  This concatenation method creates a empty wave irrespective of the units selected. The result wave is smooth and has no clicks or pops in it, though it is not easy to recognize the meaning of the synthesized utterance from this output. This option was very useful while debugging the unit selection algorithm.

- XPSOLA

  This module offers an alternative time domain pitch synchronous overlap and add module with more options that the `ps_psola` module described above.

- PSOLA

  The original PSOLA module is probably due for deletion its was too slow, contains too many bugs and now no longer supports preloaded pitch marks.



1.2 Pitch Synchronous OverLap and Add

PSOLA stands for Pitch Synchronous OverLap and Add, and designates an algorithm for performing prosodic modifications in concatenative synthesis.

The PSOLA (Pitch Synchronous Overlap and Add) algorithm was invented by Charpentier in 1988 [1]. This algorithm, which worked in the frequency domain, was very cost full(time consuming). But the algorithm was extended with a similar quality in the time domain by Eric Moulines[2].

The PSOLA synthesis scheme involves the three following steps: an analysis of the original speech waveform in order to produce an intermediate non-parametric representation of the signal, modification brought to this intermediate representation, and finally the synthesis of the modified signal from the modified intermediate representation. Here is a presentation of these three steps, and then a detailed one of the specific features of the time-scaling and the pitch-scaling algorithm.

1.2.1 Pitch-synchronous analysis

The intermediate representation of the digitized speech waveform consists of a sequence of short-term signals, obtained by multiplying the signal by a sequence of pitch-synchronous analysis windows.

The windows are centered around the successive instant t_m, called pitch-marks, which are set at a pitch synchronous rate on the voiced portions of the signal and at a constant rate on the unvoiced portions. The windows are of Hanning type and they are always longer than one single pitch period, so that neighboring short term signals (ST-signals) always involve a certain overlap. Their lengths are usually set to be proportional to the local pitch period, with a proportionality factor ranging from 2 to 4.

1.2.2 Pitch-synchronous modifications

The stream of analysis ST-signals is converted into a modified stream of synthesis ST-signals synchronized on a new set of synthesis pitch marks t_q. Such a conversion involves three basic operations: a modification of the number of ST-signals, a modification of the delays between the ST-signals, and possibly, a modification of the waveform of each individual ST-signal.
The number of synthesis pitch marks depends on the pitch-scale and time-scale modifications factors. The delays between two successive pitch-marks must be equal to the local synthesis pitch period.
In the Time-Domain PSOLA (TD-PSOLA) approach, the synthesis ST-signals are obtained by simply copying a version of the corresponding analysis signal. In the Frequency-Domain PSOLA (FD-PSOLA) approach, the synthesis ST-signals are obtained by a frequency-domain transformation of the adequate (translated) analysis ST-signal.

1.2.3 Pitch-synchronous overlap-add synthesis

Several overlap-add synthesis procedures are available to obtain the final synthetic speech. In one of its simplest scheme (the one used in ATR) the synthetic signal appears as a simple linear combination of windowed and translated versions of the original signal. All the operations involved are linear except the windowing operation. Subsequently, when combining the PSOLA speech modifications scheme with a linear filter such as an LPC-filter or a low-pass filter, the order of the operations cannot be changed without modifying the behavior of the overall system.

1.2.4 Time-scale modifications

Time-scale modifications of speech can be performed either in combination in with pitch-scaling or as a separate transformation in itself. In the latter case, no frequency-domain modifications are required and only the TD-PSOLA algorithm is used, independently of the size of the analysis window.
In the very simple case where the time-scale modification factor γ is constant, the $t_q \rightarrow t_m$ pitch-mark mapping associates t_q with the analysis pitch mark t_m being the nearest to the instant γt_q. When slowing down the speech signal, the pitch-mark mapping results in the repetition of several analysis ST-signals. Inversely, a selective elimination of the analysis ST-signals leads to an acceleration of the speech signal.
The acoustical distortions involved by the TD-PSOLA scheme are negligible for accelerating the speech rate and for slowing it down by moderate factors. However, when slowing down unvoiced portions by factors in the range of 2 and above, the regular repetition of unvoiced ST-signals introduces a short-term correlation in the synthesized signal, which is perceived as a tonal noise. A practical solution for a factor 2 is to reverse the time-axis of every repeated version of an ST-signal. Higher factors are generally not required in applications such as diphone synthesis. However, if necessary, it is possible (although costlier) to use an FD-PSOLA scheme in order to randomize the phase spectrum of repeated unvoiced ST-signals.

1.2.5 Pitch-scale modifications

Pitch-scale modifications are slightly more complicated because they interfere with time-scale modifications. The simplest case is when the signal is to be simultaneously time and pitch-scaled by the same factor β=γ. There is then a one-to-one mapping between the synthesis and analysis pitch-marks:$t_q \rightarrow t_m$. But generally, independent time and pitch-scaling factors must be applied, so the mapping is not one-to-one and results in either duplicating or eliminating of some analysis ST-signals.

Chapter 2

Improving the signal processing module

2.1 PSOLA within CHATR

The first implementation of PSOLA under CHATR was done by Hélène Valbret, who did a PhD Thesis at ENST (Ecole National Supérieur des Télécommunication) under the guidance of Dr. Moulines, one of the author of PSOLA. She developed the RUC (random Units Concatenation) module in CHATR, within the random unit concatenative synthesis module, the major synthesis method developed in ITL today.
As this first implementation's performance was found not enough satisfactory, a second implementation was thus done by Christian Lelong, an intern student from INT (Institut National des Télécommunications). This new implementation was done by using the previous module as a source of inspiration, while avoiding any of its mistakes. A total freedom concerning any personal, supplementary algorithm was given to the author. Therefore, the final algorithm was much more complex than the initial one. A lot of options that permitted to gradually prune the final output were added.
Unfortunately, a change in the format of the pitch mark implied the rewriting of part of the code. Despite a lot of serious work, the version which was delivered at the end of the intern-ship did not run and had design flows.
My task was to debug this version and make it run. This implied understanding and debbuging the code, and then only continuing to improve the algorithm.

2.2 XPSOLA

The code within CHATR had been written by different people. This is why different styles of programming and quite different qualities of the code coexist in this implementation. As one could expect, the problematic part of the code, which needed debugging was the one with very poor quality in design, documentation and writing.
The major flows were:

- No explicit variable name. Even if an effort seems to have occurred in that section, most of the variable had short diminutive or initial-based names.

- Absence of factorization for similar parts of code. A mean to achieve maintability and readability is by factorising similar pieces of code into small and well defined functions. This wasn't done in the code we had to deal with.

- Lack of comments. As code was very long and one readability decreases with the length of functions, this lack was felt particularly impeding.

2.2.1 Flows and lacks in XPSOLA

In debugging somebody else's code, understanding of the role of a function is crucial. For that, one may rely on the name of the function in the absence of comments. In doubt, the only way is to test the function on small examples. Because the functions were very long in our case, this was almost impossible. As a matter of fact it was unclear what some functions were suppose to do and whether they really did what they were supposed to do was a total mystery.
Verification of the initial and ending conditions of loops were the only checking we could achieve in many cases.

2.2.2 Requirements for an improved version

The solutions lay in any good lesson on programming, but it seem necessary to remind them:

- Explicit variable names.

- Object oriented approach. In the modern way of programming, we want to have a clear objects and all the necessary functions to create or access it. In our case, even if the structures of the objects were quite simple to understand, the basic access functions were not defined, which was badly felt during the debugging process.

- Comments. The code was very long and with a very poor percentage of comments.

2.3 Conclusion

For all those reasons it was decided to, at least temporally, abandon the debugging of this code.
For the time being CHATR uses the PS_PSOLA module. This is a basic PSOLA module written by Dr. Alan black.

Chapter 3

Improving the input quality

The previous duration prediction module hadn't got sufficient results in figures and sound (in correlation and in hearing). But we weren't sure of the reasons and of the possibility of improving it. However, Arman Magboulet's talk showed that he could predict duration on ATR database with a linear regression module, much better than it was done at that time.
So the new task was very simple: Find out why the actual module is so poor and fix it. This was mostly due to an implementation error as we are going to see later.

3.1 The former duration prediction module

The former duration prediction module was based on Dr. Nick Campbell PhD. Thesis[4]. This duration method basically breaks the task into two levels. First syllable durations are predicted and then based on that value the phonemes within that syllable are predicted.
In this implementation both the syllable durations and the phoneme durations are predicted by a neural net. This is not what was done in Dr. Nick Cambell's thesis.

Given the results this method achieves in the synthetisor it was originally implemented, it should give much better results than a simple linear regression on the phoneme set.

3.2 The Neural Nets

3.2.1 The Neural Nets module in CHATR

A subsystem for training, testing and using neural nets within CHATR is included. Although it offers some flexibility it has been specifically designed to implement Campbell's duration theory. Thus the basic structure is fixed. Although a choice in number of inputs, hidden nodes and outputs is available. All these nets consist of an input layer, a hidden layer and an output layer. All input nodes are connected to all hidden nodes and all hidden nodes are connected to all output nodes. The nets are used to produce a single value between 0 and 1 (which I am told is a slightly unusual use of neural nets).

Input values must be single digits 0-9. If all inputs for all training data are 0 or 1 the inputs are treated as binary and are rescaled, the actual inputs (and outputs) are rescaled to be greater than 0 but less that 1 but the scale factors are internal to the implementation so users need not worry about them. The outputs are linearly scaled from the examples in the training set, by the function

$$O_{net} = (O_{actual} - Min_{out} + \epsilon)/(Max_{out} - Min_{out} + \epsilon)$$

However, some other mapping function may be more appropriate.

3.2.2 Training, testing and evaluation

The module was based on two neural net. One predicting syllable duration and then one predicting the phone duration, knowing the syllable duration. A check of those nets in training and running conditions was done. The reason of the bad output layed in the "difference" between the training input feature vector and the run-time input feature vector. The two feature vectors had the same size but the order of the features weren't the same... Of course this difference was a total error, that should not exist.

So with Dr. Alan Black, we trained a number of Neural Nets, for phones and syllables, with different size of the unique hidden layer.
Note that a more powerful neural net set would provide us different number of hidden layer, which might lead to better results in duration prediction.

3.3 Database processing

3.3.1 Methodology and its limits

For some database (wnc) the syllable information wasn't available, so it was impossible to train any syllable duration prediction net for them. For those database the usual solution was to take the two nets train on the f2b corpus. This solution had defenetly lower the performances. So it was needed to add the syllable information to those databases.
The method used was to implement the code and run it on f2b data base that had all the information ever needed. Then check that the generated information and the original information in the database was the same, or close enough. Maybe "close enough" should be define.
One of the problem in ATR databases is that the different file doesn't always align. Say one have the word "Hello" in the database. The phoneme are "hh, ax, l, ow" and the syllable "hhax" and "low". The problem is that the word end time, the "ow" phoneme end time and the "low" syllable end time are not the same. Here the close solution was within 5 milliseconds distance or to align them all on the label file (phoneme), the only file that exist in all databases.
This is one possibility of "close enough". Other example might be given later.

3.3.2 Adding syllable information

A syllable was define as, given a phoneme stream, it has one and only one vowel (vowel being vowel + syllabic consonant). The boundary will be at the minimum onset of two

such vowel.
The syllable information was added by a Perl code based on Arman Magbouleh code. This code took a label file as an input and computing a minimum onset gave the syllable information.
The main problem faced was that the word boundaries weren't always syllable boundaries, which is a problem. Typically things like "Hello welcome" would give "hhax loww ehl kaxm" syllables. This was over come by adding the word boundary information to the code.

3.3.3 Reformating word information

This was easily over come by adding the word boundary information to the code. Unfortunately, this information was not available in the wnc database.
Using the .pron file, that has the word stream with different reading of the word (ie. different phoneme stream) and the label file, that has the actual phone string in the speech waveform, I could write a Perl script that created the word boundary information.
Finally reformat the word information for wnc and 2 Japanese databases (MHT and FMP).

3.3.4 Human vs machine-generated syllable assessment

Using the code already existing and the new word boundaries information, finally correct syllable information could be added to the database. Nevertheless, this generated information (machine one) wasn't exactly the same as the previously existing one (Human one). It was impossible to avoid things like "wehlk axm" as syllables for the word "welcome" to happen. It was decided that the same number of syllable, given correct word boundaries would be "close enough". The number of syllable was counted in two different methods to minimize the possible errors.
Finally, after correcting some unusual cases, the sỳllable information was added to the wnc database.

3.4 Improved duration module

Finally, the best Neural Net we had gave an 0.82 correlation, which is the best we could ever get.

Model	Train (30000)			Test (8360)		
	Mean	Std	Cor	Mean	Std	Cor
Phone Average	25.4	23.3	0.57	25.2	23.3	0.54
NNet (syl/ph)	20.0	17.2	0.81	21.3	19.0	0.78
NNet (ph)	17.7	16.9	0.82	18.5	18.7	0.76
Linear Reg	21.8	19.5	0.71	22.0	19.7	0.69

- This is on the f2b corpora
- Silences and pauses are removed from the set

The Nets were very long to train (typically one week for the phone nets and one day for the syllable ones). There is still a lot of freedom to vary the network architecture and future research might concentrate on optimising in this area.

On the other hand, we train the same Neural Net on power prediction and had promising results.

3.5 Power prediction

3.5.1 Use of a power prediction module

If power was predictable, this should improve the output sound quality. However nobody has predicted power and in ITL no promising results had been produced. Nevertheless power would be of great help to smooth the joint between two consecutive units. Indeed in the actual system, one can often see a jump in power in two consecutive units. This is perceived by a click while listening or by a strange accentuation.
A power prediction module would add this feature to the segment and therefore to the unit selection, leading to less jumps of power in selected units. More over this information could be used by PSOLA to remove the remaining power difference.

3.5.2 Power prediction based on neural nets

We tried to predict power, per phoneme, using the same input feature vector as for the duration prediction module.
The first net had the same topology as the best duration net (80 hidden nodes on a unique hidden layer). This net gave a promising result:

- Global variance of 0.4191424
- Global correlation of 0.8774434

As the same work with a linear regression method gave around 0.5 correlation, this was felt promising.
However, as the data size is 40000 phonemes, divided 90-10% train test set. It was pointed out that the neural net was much too big, compare to the size of the training data (4 data per weights!).
Moreover the figures were calculated on the whole data (training + testing) which is not representative. The test set was also felt to small.

A new experiment was carried on :

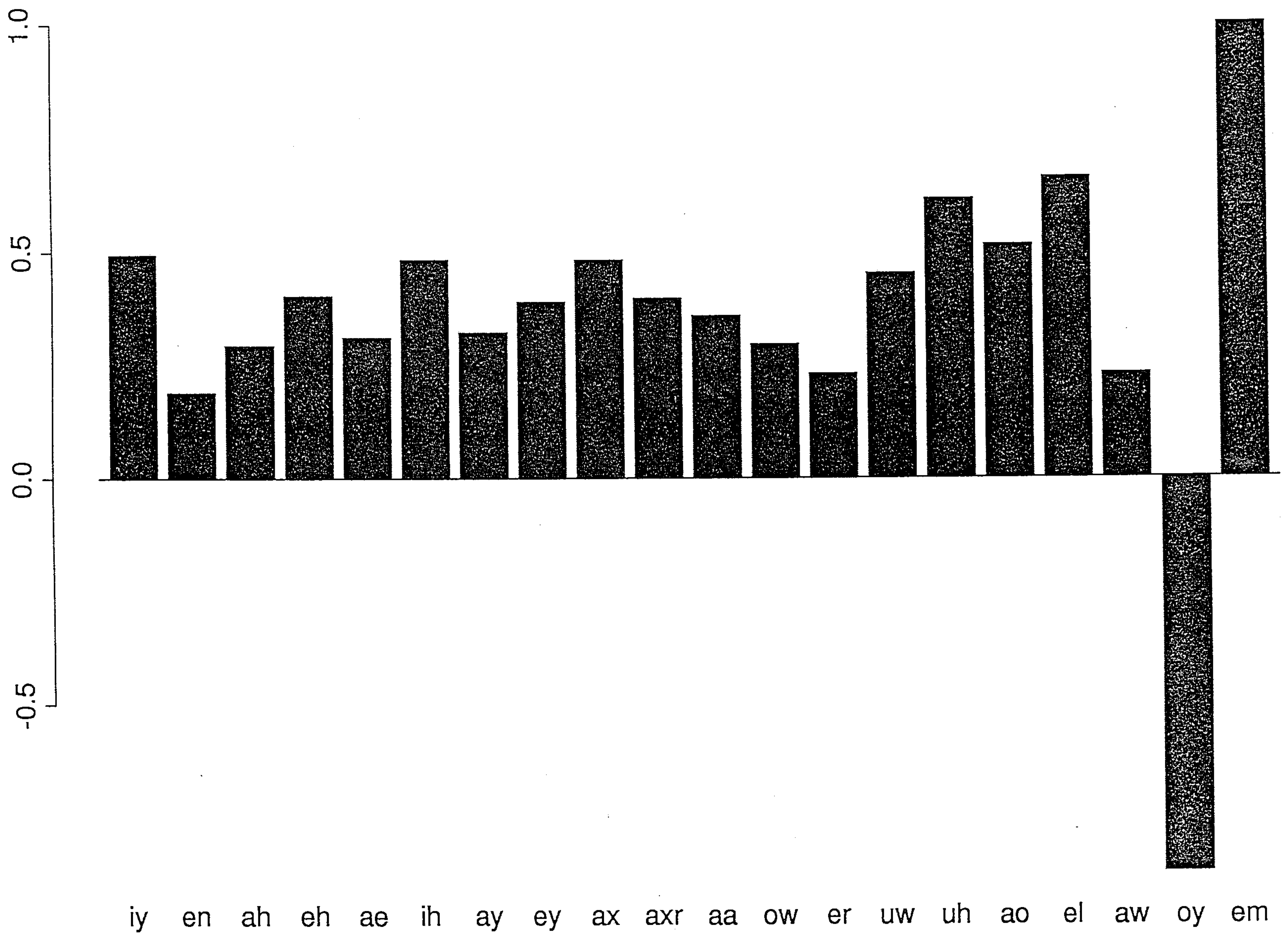
- A 20 nodes hidden layer
- 80-20% train-test set

- Evaluation only on the test set.

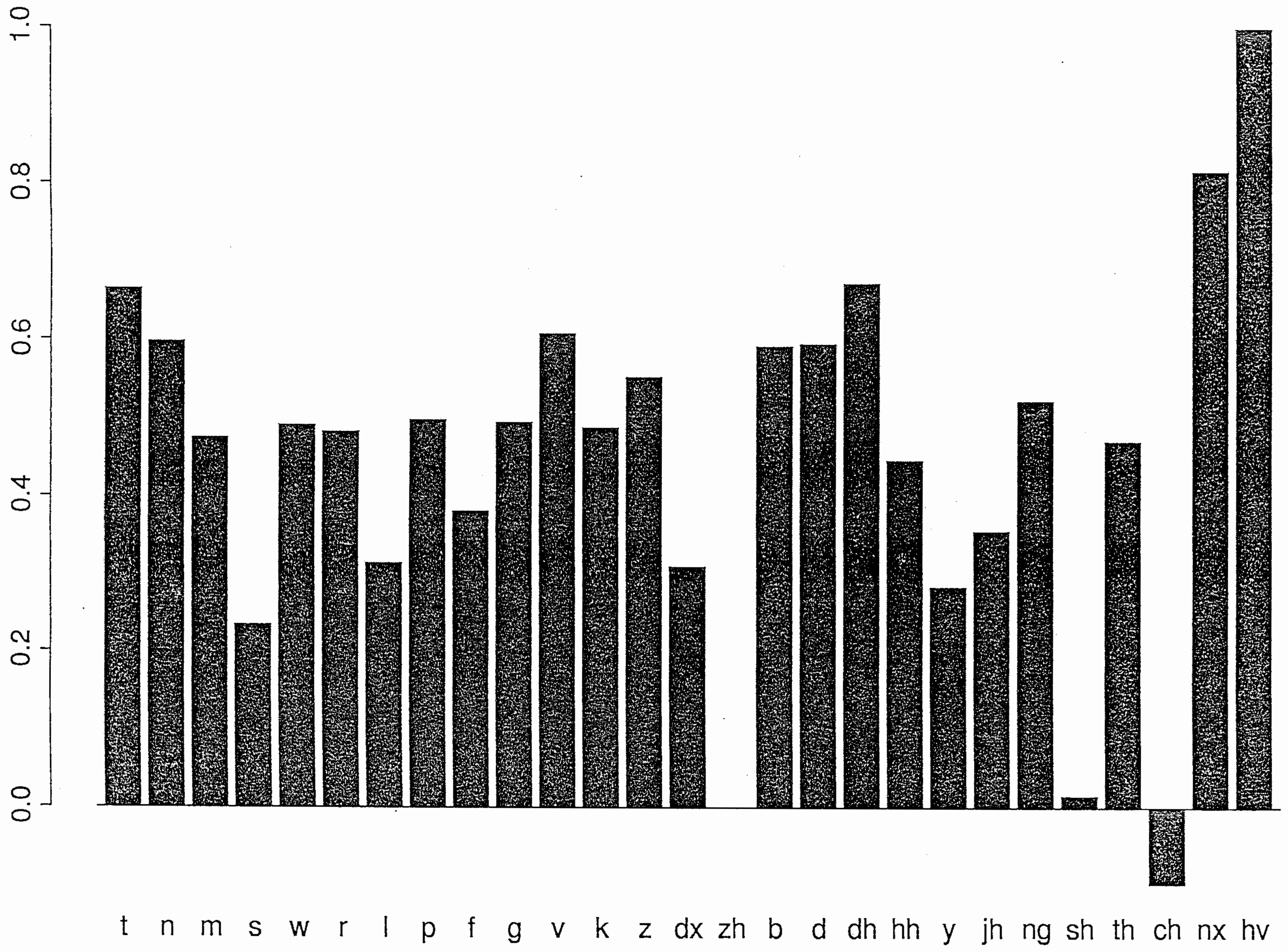
We had the excellent result of a global correlation on the test set of 0.8845843. A per phoned figures and plots are given in the following pages.

The facts remains that those figures, given the pruning of the method we can do, shows that we can do power prediction with similar input and similar methods.

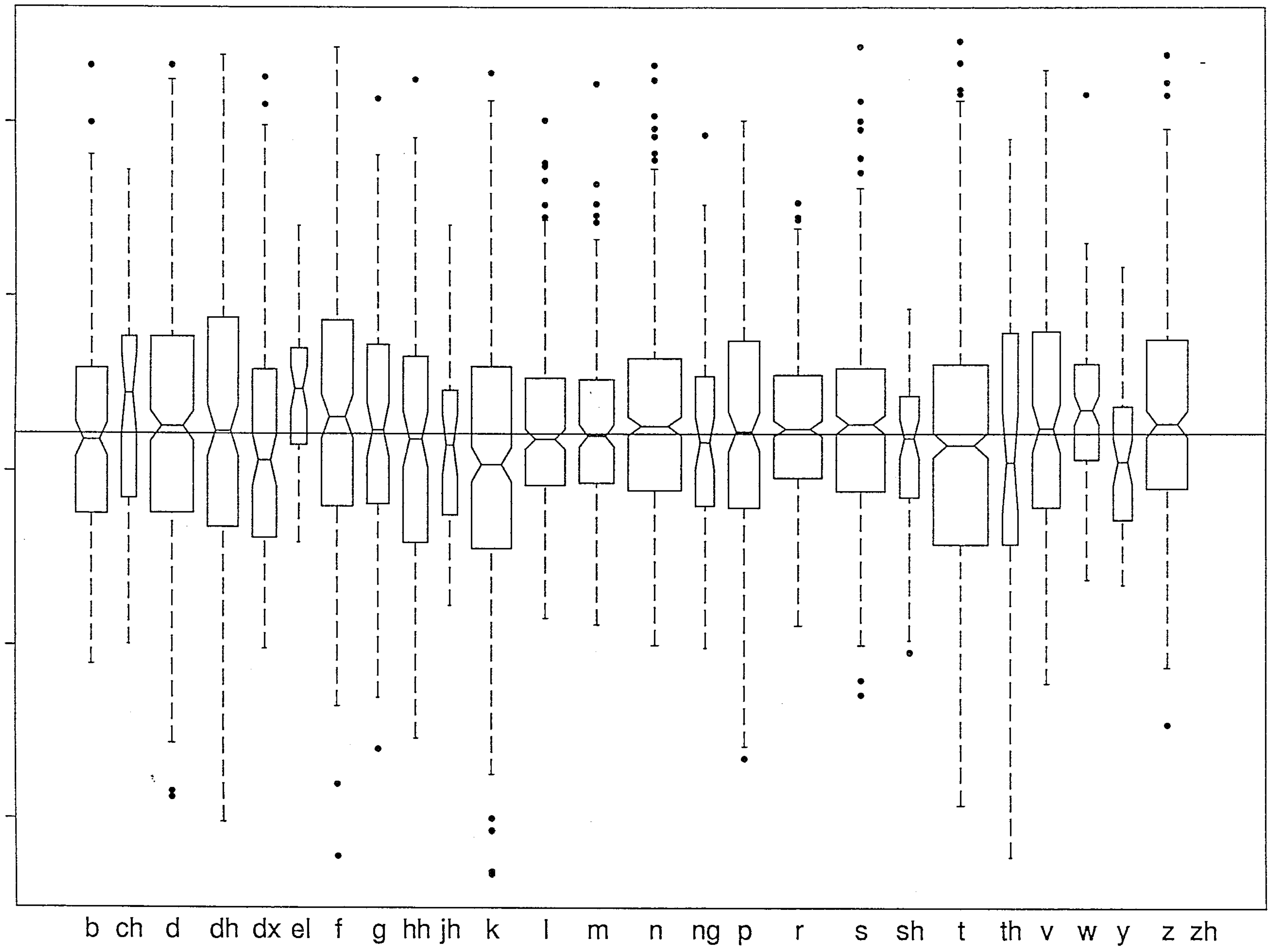
Corelation per vowel for power



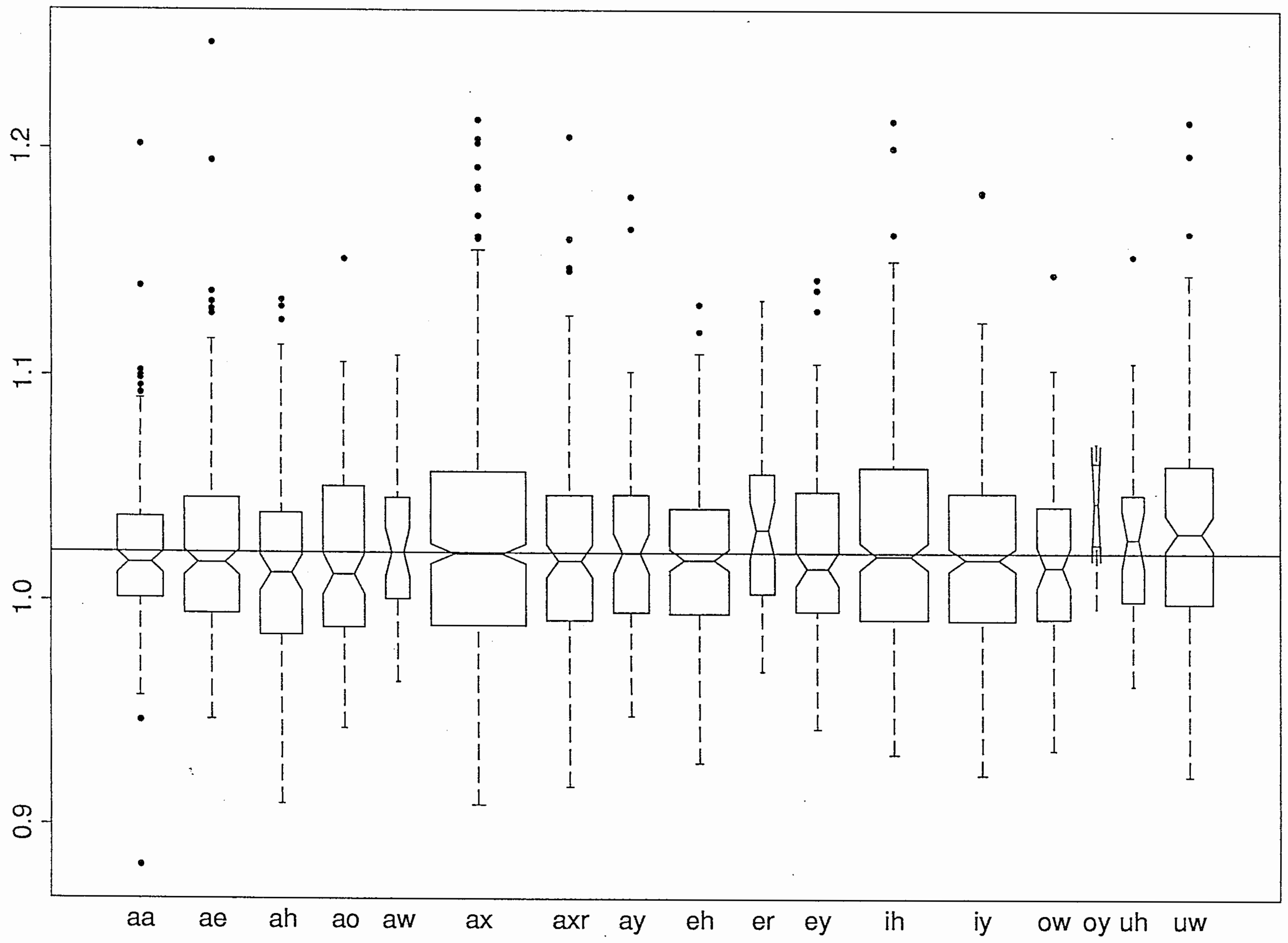
corelation per consonant for power



difference: predicted/observed power



14c



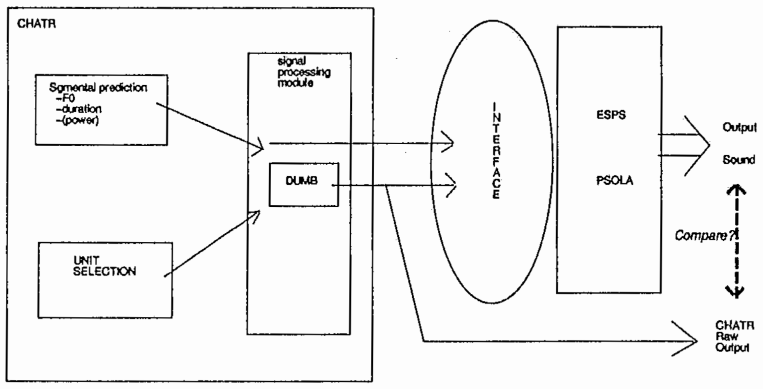
14d

Chapter 4

Changing a module

4.1 A new PSOLA from Stuttgart university

This PSOLA, the ESPS PSOLA, was developed by Gregor Moehler in Stuttgart university. Even though this PSOLA is not yet released, it is supposed to be very soon by Entropics. As it has a commercial aim, it is suppose to be very good. The only default of this PSOLA is that it is outside CHATR, meaning a need to built an interface between CHATR and it.



A lot of laboratories in the world use PSOLA as a final signal processing module in their speech synthetiser, with very good result. However the unit used are completely different from the ones in ATR. All other laboratories work on diphones or triphones, which are very clean, very carefully labeled unit. And most of all, the pitch mark are very good. Indeed on a limited amount of data they can hand correct them. On the other hand ATR uses random units which are selected from a natural speech database.
To suppose that the quality of the unit affect the quality of PSOLA output seems logical. Unfortunately we don't know in which extent this is true.

4.2 Interface CHATR to ESPS PSOLA

4.2.1 Specification

As ATR never had a probing improvement in sound quality with the various PSOLA implemented within CHATR, it was decided to create the "best" sound quality possible with CHATR's natural speech units using ESPS PSOLA.

So, the task was to:

- Build a robust interface between CHATR and ESPS PSOLA
- Use this tool to test the validity of PSOLA on natural speech units
- Use this tool to test the improvement in sound quality a power modification could bring

4.2.2 A user friendly interface

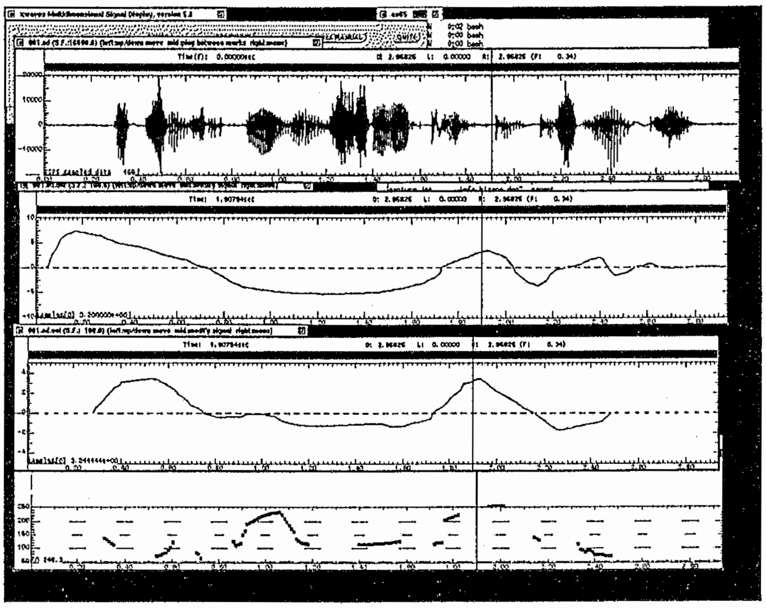


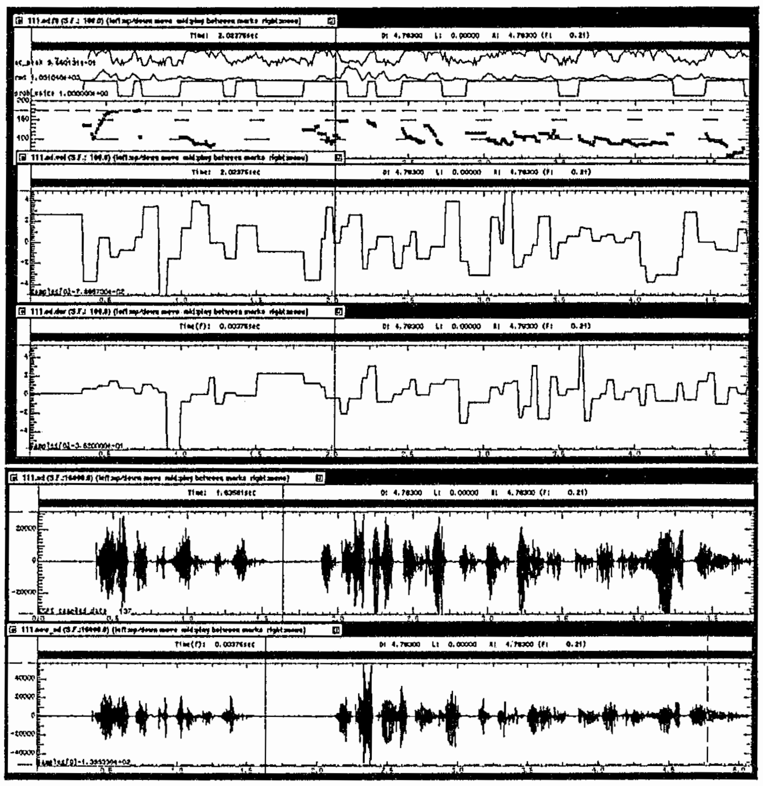
Figure 1

This PSOLA has a nice machine-to-human interface,but no program-to-program interface was thought of. You display your waveform with XWAVES. To click on PSOLA activates three windows, the .ed.dur, .ed.f0 and the .ed.vol files. The .ed.f0 file has got the ESPS representation of the f0 contour of the waveform. With the mouse you modify it to the contour you want. For the .ed.vol and the .ed.dur file, you draw the modification, which is in decibels (cf. Figure 1).

To create those .ed files turned out to be very difficult.

Witting and reading ESPS header was difficult. One of the major problem faced was a good data file with a bad header. However, the PSOLA program has a check on

the header (the .ed files and the input waveform are of the same length), therefore the waveform could not go through the algorithm.
Finally we worked on a phoneme base.



The research was then axed in the following first two main directions. Psola was performed on different kind of input:

- CHATR units + CHATR segmental prediction.
  Equivalent to having a new signal processing module. Everything is performed on CHATR output.
- CHATR units + Natural targets.
  This is to test the best possible PSOLA on CHATR units.
- Natural speech original waveform + Natural targets.
  The natural targets are the segmental features of the same phrase said by another speaker. This is probably the best PSOLA ever possible on natural speech units.

4.3 Perceptual evaluation

The best we could obtain in the first two cases, the ones we are really interested in, is maybe only slightly better than CHATR output. However, in most of the cases, the duration modification seems to give a metallic sound to the output speech.
Some of the tested utterance were quite good. We can state that they were slightly better

than CHATR raw output. However, in most of the case the sound was metallic due to the distortion caused by the process. In many cases removing the duration modification, and do only f0 and power modification, gave good output sound. Unfortunately the sound quality was very close to CHATR output, and decide whether it was better or worst was very subjective.
It is not an astonishing result that the duration modification gives a bad sound quality. It is stated by the author of PSOLA that a greater modification than 3db with TD_PSOLA might give bad result. The threshold we put is at -5 and 5 db. Moreover, combining a pitch modification and a duration modification can increase the risk because pitch modification affects duration modification.

Chapter 5

Conclusion

5.1 Summary

5.1.1 Improvement of the component: PSOLA module

Fixing the flows of X_PSOLA turned out to be very difficult. Probably a good programmer should be able to debug it, but I think it might take a long time, and a good programmer probably might prefer to work from scratch or from PS_PSOLA. The lesson we can take from that is whenever an intern comes to ATR with a knowledge which is not programming, probably it might be better to form (or re-instil) him a little to the basic notions of maintainable programming. His work then might be useful to the later researcher.
An interface between CHATR output or natural speech waveform to ESPS PSOLA is now built. As the sound quality is not that better, this interface was used to try to locate the problems. One of them is the quality of the pitch marking. This tool can be used to continue this work.

5.1.2 Improvement of the input

We now have a more robust and reliable duration prediction module. While doing this work I also add the syllable information to wnc data base, and have the code that can add it to any other data base. The word information was reformat to the new standard for wnc, FMP and MHT data base.
A first draft of power prediction using neural nets was done, showing that this is possible(the results were not satisfactory with a linear regression method). However the results should be analyzed more carefully and the power prediction module designed in respect of that analysis.

5.2 Future work and discussion

We now have a good interface between CHATR And ESPS PSOLA. However the output sound quality is not that better. This is probably due to the specificity of the units used in CHATR (natural speech units). My point of vue is that we can not improve that

much this interface. The problem being CHATR specific, I think it should be treated in a CHATR specific way, within CHATR. PS_PSOLA is a good base for that.
The problems with the implementation of the duration module have been found and fixed. The framework of this module has been used for prediction of power and the first steps toward a power prediction module have been done. However a lot of work still needs to be done.
One of the main work is finding a clever way to represent power. The work was done in predicting phone power. This is probably not the best way to represent power. Maybe the global contour of an utterance or specification of three points, such as joint and maximum with slopes, etc, might be better.This power representation must have a perceptual meaning, so that improving it improves the output sound quality.
The power prediction have been done on the same input features as the duration prediction. Even if those two sets of features must have a lots in common, a proper analysis of the contribution of each feature should be done. Probably unneeded features exists in the actual set and some important one might be missing. We could then have an improved input feature set. The analysis of the data created (the different prediction nets and the first figures extracted from them) should be a reasonable departure point.

Bibliography

[1] F. Charpentier.
Traitement de la parole par analyse-synthèse de Fourier: Application à la synthèse par diphones. *Doctoral Thesis, Ecole Nationale Supérieure des Tlécommunications*, 1988.

[2] E. Moulines.
Algorithmes de codage et de modification des paramètres prosodiques pour la synthèse de parole à partir du texte. *Doctoral thesis, Ecole Nationale Supérieure des Tlécommunications*, 1990.

[3] E. Moulines and F. Charpentier.
Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones. *Speech Communication,* vol. 9, Nos. 5/6, pp.453-467, December 1990.

[4] N. Campbell.
ATR Tech Rept TR-IT-0035