

ROBUST FORMANT ANALYSIS FOR SPEECH SYNTHESIS APPLICATIONS

L.F. Willems¹

ABSTRACT

For speech synthesis applications a formant description of the speech signal has a number of advantages over other parametrizations. The analysis of formant frequencies and bandwidths from the LPC coefficients has two drawbacks: sometimes the number of formants detected is smaller than is needed for the synthesizer and sometimes due to numerical instability, the analysis fails completely. A method is described to first derive the formant frequencies by means of the Split Levinson Algorithm and second, to find optimal bandwidth values from a table.

INTRODUCTION

Formants are defined as the acoustical resonances of the vocal tract. In the case of a simple vocal tract without side branches such as the nasal cavity, the transfer characteristic of the vocal tract consists of a number of resonances and the transfer function has poles at the position of these acoustical resonances. During articulation other, more complex vocal tract configurations also occur, such as in nasals, plosives and fricatives. Then the acoustic tube is not a simple one but has side tubes. In these cases the transfer characteristic is also more complex and the transfer function then has zeroes and a varying number of poles.

In a practical speech synthesizer, use is often made of a rather simple filter structure consisting of a fixed number of poles representing the formants in speech production. These practical speech synthesizers are nowadays available as ICs such as the MEA8000 and PCF8200. The problem of determining a fixed number of formant frequencies and bandwidth values is often tackled by means of an LPC-analysis followed a root solving procedure in order to find the poles of the all pole filter found in the LPC-analysis. This analysis method suffers from a few drawbacks.

This paper describes an algorithm, which determines the necessary synthesis data for a fixed synthesis filter structure in two steps: in the first, a fixed number of formant frequencies is determined in a robust way by means of the Split Levinson Algorithm, and in a second step optimum bandwidth values are chosen from a small table with eight values.

FORMANT FREQUENCIES FROM LPC AND ROOT SOLVING

We will briefly describe this LPC and root-solving analysis method because it is related to the new algorithm. An autocorrelation LPC analysis is performed on a speech segment of 25 ms. The classical algorithm to solve for the predictor coefficients is the Levinson algorithm. It calculates recursively from the autocorrelation coefficients and the predictor coefficients of the previous recursion step the new predictor coefficients. The resulting polynomial $A(z)$ is

$$A(z) = 1 + a_1 z^{-1} + a_2 z^{-2} + \dots + a_M z^{-M} \quad (1)$$

where M is the order of the filter polynomial and is equal to twice the number of formants. This polynomial can be decomposed in $M/2$ second-order terms with, for instance, Bairstow's method (Hildebrand, 1956).

$$A(z) = \prod_{j=1}^{M/2} (1 + p_j z^{-1} + q_j z^{-2}) \quad (2)$$

¹Institute for Perception Research, POBox 513, 5600 MB Eindhoven, Netherlands

The formant frequency F_j and the bandwidth value B_j in Hz can be determined with

$$q_j = e^{-2\pi \frac{B_j}{F_s}} \quad (3)$$

$$p_j = 2e^{-\pi \frac{B_j}{F_s}} \cdot \cos(2\pi \frac{F_j}{F_s}) \quad (4)$$

where F_s is the sampling frequency used. This determination of F_j and B_j is only successful if the second-order term has complex conjugate zeroes, or if $p_j^2 - 4q_j < 0$. This is not always the case, because the production model deviates from the synthesis model. In practical situations we found in 5–10% of the number of analysis frames that at least one of the second order terms has no complex conjugate zeroes. This results in gaps in the formant tracks, as can be seen in Figure 2, lower box.

A second problem is that the Bairstow method does not always converge to a solution. If the starting value for the solution which has to be given to the Bairstow algorithm is not close enough to the actual zeroes, then the algorithm may converge or may not. This analysis procedure thus cannot be called robust.

USING THE SPLIT LEVINSON ALGORITHM

The Split Levinson Algorithm (SLA) which was developed by Delsarte and Genin (Delsarte & Genin, 1986) produces the same result: $A(z)$ of (1) as the classical Levinson Algorithm but with less computational effort (almost half). Instead of the predictor polynomials as used in the classical algorithm, the SLA uses the so-called singular predictor polynomial (SPP). These polynomials have the property that their zeroes lie on the unit circle (the polynomials being singular).

One can show (see Soong & Juang, 1984) that the zeroes of the SPP are related to the zeroes of the classical predictor polynomial $A(z)$, and that this correspondence is the better the closer the zeroes of the classical predictor polynomials are to the unit circle. Hence, for zeroes close to the unit circle, which correspond to formants with low bandwidth values and thus to perceptually important ones, the approximation to the zeroes of the SPP is closest. In the formant frequency analysis method proposed here we approximate the formant frequencies by the frequencies at which the zeroes in the SPP occur.

In this way the required number of formant frequencies is always found, since the zeroes of the SPP are on the unit circle and thus occur in complex conjugate pairs and can always be converted into formant frequencies by application of (4). Apart from the fact that a fixed number of formant frequencies is always found, another advantage of using the SPP is that it allows for a robust method of determining the zeroes numerically. During the recursion steps of the SLA the positions of the zeroes shift along the unit circle in a regular and predictable way. The SPP in the recursion steps can be written as:

$$\begin{aligned} k=0 & \quad P_0(z) = 1 \\ k=1 & \quad P_1(z) = 1 + z^{-1} \\ k=2 & \quad P_2(z) = 1 + p_{2,1}z^{-1} + z^{-2} \\ k=3 & \quad P_3(z) = 1 + p_{3,1}z^{-1} + p_{3,1}z^{-2} + z^{-3} \\ k=4 & \quad P_4(z) = \dots \end{aligned} \quad (5)$$

It is a property of these polynomials that the interval in which the zeroes of $P_k(z)$ are found can be derived from the zeroes of $P_{k-1}(z)$. A zero of $P_k(z)$ is in an interval which is bounded by two zeroes of $P_{k-1}(z)$ (and the bounds $z^{-1} = +1$ and $z^{-1} = -1$). One zero in a known interval can always be determined numerically. The polynomials in (5) can be simplified

furthermore through the substitution $z^{-t} + z^t = 2 \cos \omega t$ mapping the unit circle on the interval $[+1, -1]$ and writing $\cos \omega t$ as powers of $\cos \omega$ (see also Soong & Juang, 1984). The resulting data are of the same kind as the (p, q) coefficients in (3) and (4). In Figure 1 the positions of the SPP zeroes up to $k = 15$ are sketched for a frame of speech. One can easily ascertain that the zeroes travel in a regular fashion.

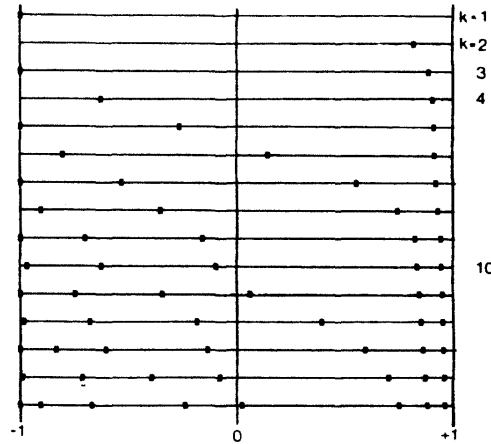


Figure 1: The zeroes for all the SPP during a recursion of the SLA up to $k = 15$

OPTIMAL BANDWIDTH VALUES

The last step in the analysis is to find bandwidth values associated with the already determined formant frequencies in such a way that the all-pole filter $1/A(z)$ minimizes the error between the predicted data and the input speech. This filter can be built with the formant frequencies and the estimated bandwidth values. The total error E can now be written as

$$E = \sum_{n=1}^N (s_n + \sum_{k=1}^M \tilde{a}_k s_{n-k})^2 \quad (6)$$

where s_n are the speech samples.

N is the number of speech samples in the window ($N \approx 250$).

\tilde{a}_k are the coefficients of the polynomial $\tilde{A}(z)$ composed from the formant frequencies and the bandwidth values chosen. The latter are varied in order to minimize the error E .

M is the order of $\tilde{A}(z)$ and is two times the number of formants.

Note: In our analysis method we use a sampling frequency of 10 kHz and then we choose $M = 10$ for male voices looking for five formants in the region of 0 to 5000 Hz. This is optimal for a vocal tract length of 17 cm. For female voices we choose $M = 8$.

The expression for the total error E (6) can be written as (with $\tilde{a}_0 = 1$)

$$E = \sum_{k=0}^M \tilde{a}_k^2 r_k + 2 \sum_{k=0}^M \tilde{a}_k \sum_{j=k+1}^M \tilde{a}_j r_{k-j} \quad (7)$$

where $r_k = \sum_{j=1}^{N-k} s_j s_{j+k}$ are the autocorrelation coefficients which have already been determined and have served as input to the SLA.

In the minimization procedure we take a bandwidth value from a table arranged with in-

creasing values for every formant. Now this value is systematically varied until the minimum error is found. This is done successively for every formant in increasing order from F_1 to F_5 and repeated until the bandwidth values no longer change or a maximum number of iterations is reached for all the formants. We have observed that this suboptimal search procedure leads to correct bandwidth data, provided the table is searched from low bandwidth values to higher ones. This convergence behaviour was checked in a number of full searches.

RESULTS

For all frames the analysis method described produces ordered formant tracks with no data missing. In Figure 2 the new analysis method is compared with the LPC analysis + root solving. In the latter case the formants found are not ordered (cannot be seen in this picture). If the speech is resynthesized with a direct form filter (of the type $A(z)$) hardly any difference

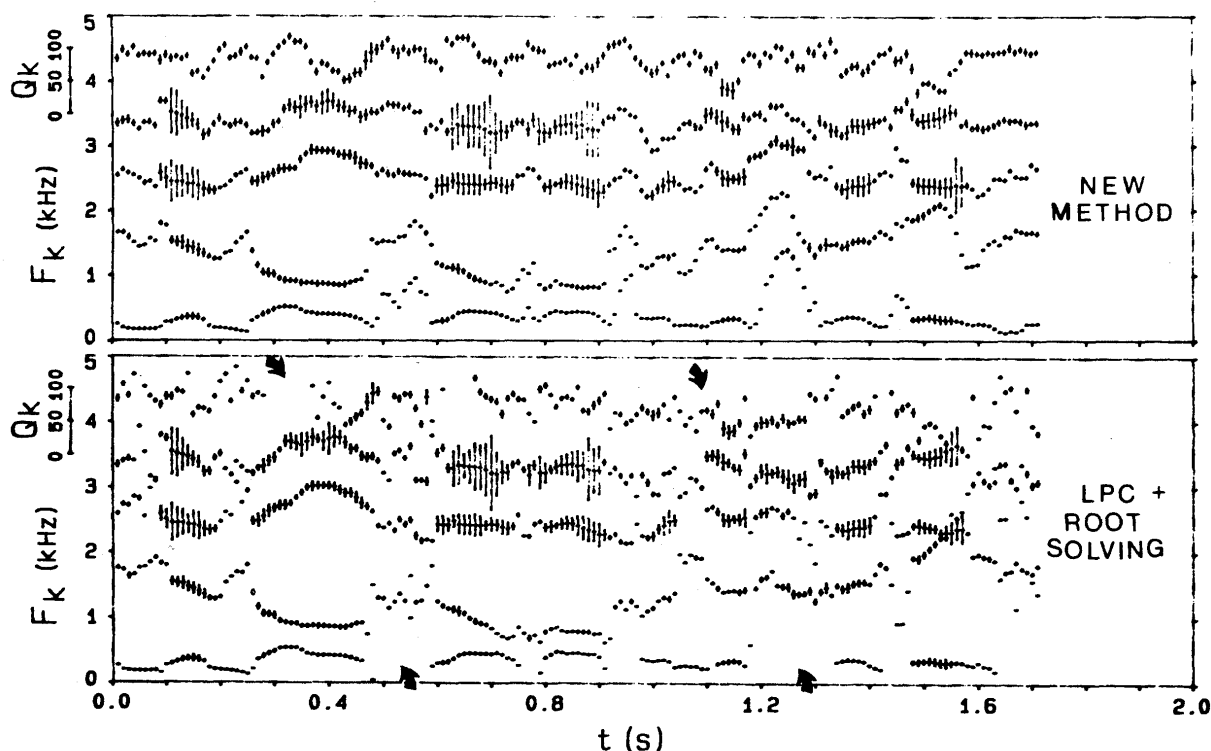


Figure 2: Comparison of analysis data from both methods. In the lower box some formant tracks show gaps (see arrows).

can be heard between the results of the two methods, but when a formant synthesizer is used, or the data have to be manipulated as is done, for instance, in concatenating diphones, the advantage of the formant analysis method here described is obvious.

REFERENCES

- Delsarte, P. & Genin, Y. (1986) The Split Levinson Algorithm. *IEEE Transactions ASSP*, 34, 470-478.
- Hildebrand, F.B. (1956) *Introduction to numerical analysis*. New York: McGraw Hill.
- Soong, F.K. & Juang, B.H. (1984) Line Spectrum Pair (LSP) and Speech Data Compression. *ICASSP 1984*, paper 1.10.