The Computation of Line Spectral Frequencies Using Chebyshev Polynomials

R. P. Ramachandran and P. Kabal

INRS-Télécommunications
3 Place du Commerce
Ile des Soeurs, Qué.
CANADA H3E 1H6

September 1, 1985

Rapport technique de l'INRS-Télécommunications no. 85-27

The Computation of Line Spectral Frequencies Using Chebyshev Polynomials

Abstract

Line spectral frequencies provide an alternate parameterization of the analysis and synthesis filters used in linear predictive coding (LPC) of speech. In this paper, a new method of converting between the direct form predictor coefficients and line spectral frequencies is presented. Both even and odd order LPC systems are considered. The system polynomial for the analysis filter is converted to two even order symmetric polynomials with interlacing roots on the unit circle. The line spectral frequencies are given by the positions of the roots of these two auxiliary polynomials. The response of each of these polynomials on the unit circle is expressed as a series expansion in Chebyshev polynomials. The line spectral frequencies are found using an iterative root finding algorithm which searches for real roots of a real function. The algorithm developed is simple in structure and is designed to constrain the maximum number of evaluations of the series expansions. The method is highly accurate and can be used in a form that avoids the storage of trigonometric tables or the computation of trigonometric functions. The reconversion of line spectral frequencies to predictor coefficients uses an efficient algorithm derived by expressing the root factors as an expansion in Chebyshev polynomials.

Contents

1. Introduction

In many speech coders, the parameters of the all-zero predictor filter or the corresponding all-pole synthesis filter are coded and sent as part of the information stream. Recently, there has been a growing interest in the use of line spectral frequencies (LSF's) to code the filter parameters for linear predictive coding (LPC) of speech [1][2][3][4]. LSF's are an alternative to the direct form predictor coefficients or the lattice form reflection coefficients for representing the filter response.

The direct form coefficient representation of the LPC filters is not conducive to efficient quantization due to the large dynamic range of these coefficients. Instead, non-linear functions of the reflection coefficients (e.g. log-area ratio or inverse sine of the reflection coefficient) are often used as transmission parameters [5]. These parameters are preferable since the reflection coefficients have a well-behaved dynamic range. Coefficient by coefficient quantization of the reflection coefficients results in an efficient representation of the salient spectral features.

Line spectral frequencies are an alternate parameterization of the filter with a one-to-one correspondence with the direct form predictor coefficients. The concept of an LSF was introduced by Itakura [6]. LSF's have a well-behaved dynamic range and have been shown to encode speech spectral information more efficiently than other transmission parameters [2][3][4][7]. This can be attributed to the intimate relationship between the LSF's and the formant frequencies. Accordingly, LSF's can be quantized taking into account spectral features known to be important in perceiving speech signals. For instance, the higher LSF's may be quantized more coarsely or even not transmitted at all. This reduces the required bit rate with no significant effect on speech intelligibility [4]. In addition, LSF's lend themselves to frame-to-frame interpolation with smooth spectral changes because of their frequency domain interpretation.

The emphasis of this work is on the efficient computation of line spectral frequencies. This will involve an iterative root finding algorithm for a series representation in Chebyshev polynomials. The algorithm developed is simple in structure and constrains the maximum number of function evaluations. These considerations are important if LSF's are to be used in a real time environment. The reconversion of LSF's to predictor coefficients is based on reconstructing the expansion in Chebyshev polynomials from the root factors.

In the next section the background framework is set up for computing line spectral frequencies. This includes an explicit formulation for odd order LPC systems as well as the even order ones. Previous work has focussed on coding of pairs of line spectral frequencies (line spectral pairs) and as such, only the case of even order predictors has been explicitly presented in earlier studies.

2. Line Spectral Frequencies

Although LSF's show great promise for coding the LPC filter response, deriving them from predictor coefficients is computationally more complex than deriving reflection coefficients from predictor coefficients. Indeed, the reflection coefficients are a byproduct of solving for the predictor coefficients in an autocorrelation formulation for LPC.

The starting point for deriving the LSF's is the response of the prediction error filter with P coefficients,

$$A(z) = 1 - \sum_{k=1}^{P} a(k) z^{-k} . \tag{1}$$

The $\{a(k)\}$ are the direct form predictor coefficients. The corresponding all-pole synthesis filter is $1/A(z)$. A minimum phase prediction error filter (i.e. one with all its roots within the unit circle) has a corresponding synthesis filter which is stable.

A symmetric polynomial $F_1(z)$ and an anti-symmetric polynomial $F_2(z)$ related to $A(z)$ are formed by adding and subtracting the time-reversed system function,

$$\begin{aligned} F_1(z) &= A(z) + z^{-(P+1)} A(z^{-1}) , \\ F_2(z) &= A(z) - z^{-(P+1)} A(z^{-1}) . \end{aligned} \tag{2}$$

The roots of these two auxiliary polynomials determine the line spectral frequencies. The two polynomials also have the interpretation of being the system polynomials for a $P+1$ coefficient predictor derived from a lattice structure. The first P stages of the lattice have the same response as the original P stage predictor. An additional stage is added with reflection coefficients equal to $+1$ or -1 to give the response $F_1(z)$ and $F_2(z)$ respectively [2].

Soong and Juang [2] have shown that if $A(z)$ is minimum phase, (1) the roots of $F_1(z)$ and $F_2(z)$ are on the unit circle; and (2) the roots are simple and separate each other. For minimum phase predictors, the above conditions assure that the LSF's are well defined. In addition, any procedure which determines an ordered set of LSF's can be used to construct a minimum phase predictor filter. The reconstruction of a predictor from a set of LSF's will be discussed in a later section.

The polynomials $F_1(z)$ and $F_2(z)$ being symmetrical and anti-symmetrical respectively, have roots at $z = +1$ and/or $z = -1$ which can be removed by polynomial division.

$$\begin{aligned} G_1(z) &= \frac{F_1(z)}{1+z^{-1}} \qquad \text{and} \qquad G_2(z) = \frac{F_2(z)}{1-z^{-1}} , \qquad P \text{ even}, \\ G_1(z) &= F_1(z) \qquad\quad \text{and} \qquad G_2(z) = \frac{F_2(z)}{1-z^{-2}} , \qquad P \text{ odd}. \end{aligned} \tag{3}$$

These polynomial divisions can be performed by additions and subtractions of the coefficients of $F_1(z)$ and $F_2(z)$. The resulting $G_1(z)$ and $G_2(z)$ are symmetric polynomials of even order and have all their roots on the unit circle. These unit circle roots are simple and separate each other on the upper semi-circle. Since the roots occur in complex conjugate pairs, it is only necessary to determine the roots located on the upper semi-circle. The roots of interest are $\exp j\omega_i$ for $i = 1, 2, \ldots, P$. The line spectral frequencies are the angular positions of the roots, $0 < \omega_i < \pi$.

Fig. 1 shows the arrangement of zeros of $F_1(z)$ and $F_2(z)$ for both even and odd P. These plots show the actual root positions for a voiced segment of speech (8 kHz sampling rate). The polynomials $G_1(z)$ and $G_2(z)$ have the same zeros as $F_1(z)$ and $F_2(z)$ respectively, except for the zeros at $z = \pm 1$. It can be noted that for any order, the lowest frequency LSF corresponds to a root of $G_1(z)$. These plots show that for roots of $A(z)$ near the unit circle, a pair of LSF's tends to bracket the angular position of the root of $A(z)$. However, it also indicates that the difference between pairs of LSF's is not necessarily a good indicator of how close a root of $A(z)$ is to the unit circle. Interpretation of the LSF's in terms of formant resonances for P odd is more tenuous due to the influence of the real axis root of $A(z)$.

A stability theorem which uses a form similar to the LSF formulation has been formulated by Schussler [8]. The auxiliary symmetric and anti-symmetric polynomials defined by Schussler become the same as $F_1(z)$ and $F_2(z)$ if $A(z)$ is considered to be a polynomial of degree $P+1$ with $a(P+1) = 0$. The root locations of the auxiliary polynomials given by a direct application of Schussler's theorem (i.e. without appending a zero valued coefficient) and those given by a LSF formulation are compared in Appendix A. This examination provides additional insight as to the relationship of the LSF's to the roots of $A(z)$.

The cases of an odd number and an even number of LSF's differ in some details. Let the order of the polynomials $G_1(z)$ and $G_2(z)$ be $2M_1$ and $2M_2$ respectively,

$$\begin{aligned} M_1 &= \frac{P}{2} \quad \text{and} \quad M_2 = \frac{P}{2}, && P \text{ even}, \\ M_1 &= \frac{P+1}{2} \quad \text{and} \quad M_2 = \frac{P-1}{2}. && P \text{ odd}. \end{aligned} \tag{4}$$

Then explicitly showing the symmetry of the polynomial coefficients,

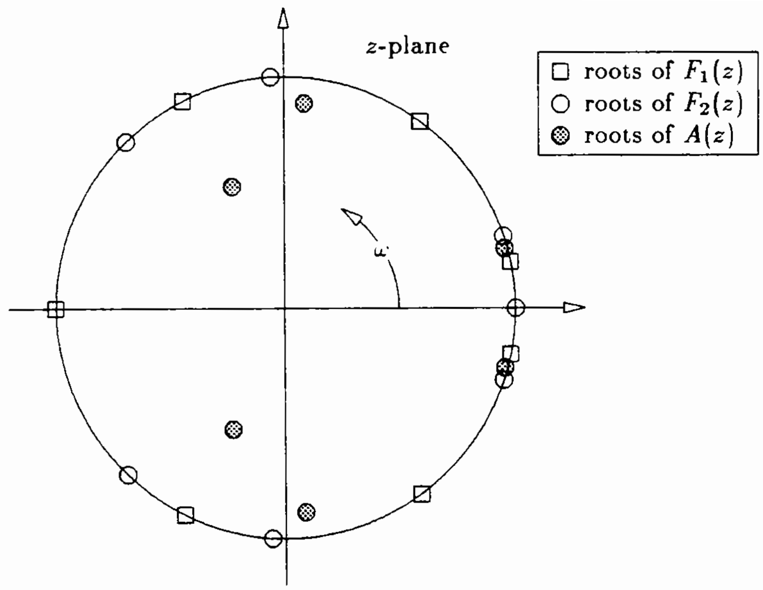
$$\begin{aligned} G_1(z) &= 1 + g_1(1)z^{-1} + \cdots + g_1(M_1)z^{-M_1} + \cdots + g_1(1)z^{-(2M_1-1)} + z^{-2M_1}, \\ G_2(z) &= 1 + g_2(1)z^{-1} + \cdots + g_2(M_2)z^{-M_2} + \cdots + g_2(1)z^{-(2M_2-1)} + z^{-2M_2}. \end{aligned} \tag{5}$$

Both polynomials are of even degree, with $G_1(z)$ contributing M_1 pairs of conjugate zeros and $G_2(z)$ contributing M_2 pairs of conjugate zeros ($M_1 + M_2 = P$). On the unit circle, the linear phase term can be removed to give two zero phase series expansions in cosines,

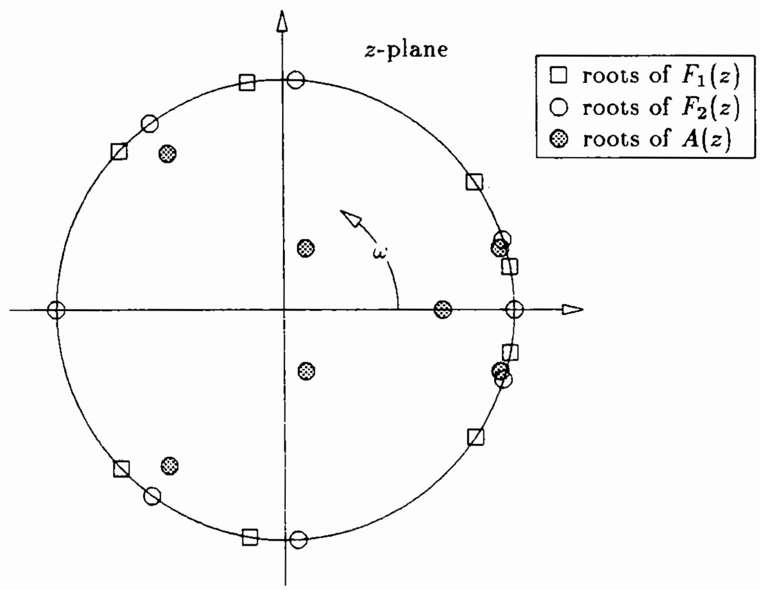
$$\begin{aligned} G_1(e^{j\omega}) &= e^{-j\omega M_1} G_1'(\omega), \\ G_2(e^{j\omega}) &= e^{-j\omega M_2} G_2'(\omega). \end{aligned} \tag{6}$$

where

$$\begin{aligned} G_1'(\omega) &= 2\cos M_1\omega + 2g_1(1)\cos(M_1 - 1)\omega + \cdots + g_1(M_1), \\ G_2'(\omega) &= 2\cos M_2\omega + 2g_2(1)\cos(M_2 - 1)\omega + \cdots + g_2(M_2). \end{aligned} \tag{7}$$



(a) P even (shown for $P = 6$)



(b) P odd (shown for $P = 7$)

Fig. 1 Root locations

Various methods to locate the roots of $G'_1(\omega)$ and $G'_2(\omega)$ have been suggested. Soong and Juang [2] have proposed a numerical technique to find the LSF's. The function values given by evaluation of Eq. (7) are found on an "adequately fine" grid. Evaluation proceeds with a direct calculation of a discrete cosine transform. It is suggested that a cosine table can be stored beforehand to speed up the computation. Sign changes at adjacent grid points isolate intervals containing roots and further bisection of these intervals gives an approximation to the root positions. Soong and Juang also point out that for $P = 8$, a closed form solution for the roots is possible. In this case, the roots of two fourth order polynomials are sought.† However, such a closed form solution involves the computation of transcendental functions and hence may not be appropriate in a real-time environment.

Kang and Fransen [7] have proposed two other methods for finding the LSF's. In one method, the autocorrelation functions of the coefficients of $G'_1(\omega)$ and $G'_2(\omega)$ are used to calculate power spectra. The locations of the local minima of the power spectra give the LSF's. Again, the evaluation of the power spectrum for various values of ω involves a series expansion in cosine terms. Furthermore, the search for local minima can be very time consuming.

The second method proposed by Kang and Fransen to determine the LSF's uses an allpass ratio filter,

$$R(z) = \frac{z^{-(P+1)}A(z^{-1})}{A(z)} . \qquad (8)$$

The phase spectrum of the ratio filter is evaluated and whenever the phase response takes on a value which is a multiple of π, the corresponding frequency is an LSF. The computation of the phase spectrum for various frequencies involves evaluating a series expansion of sine and cosine terms and an inverse tangent operation.

The method proposed in this paper requires no prior storage or calculation of trigonometric functions. Instead, an expansion in Chebyshev polynomials is used. The method is introduced in two steps. In the next section, the use of a Chebyshev polynomial expansion is discussed. Subsequently an efficient numerical algorithm to find the roots with this formulation is established.

† The fourth order polynomials are essentially expanded forms of the Chebyshev polynomial series introduced in the next section.

3. Chebyshev Series Formulation

The use of Chebyshev polynomials eliminates the computation of trigonometric functions and/or the prior storage of trigonometric tables. Frequencies are mapped using $x = \cos\omega$. Then

$$\cos m\omega = T_m(x) \ , \tag{9}$$

where $T_m(x)$ is an m'th order Chebyshev polynomial in x. The Chebyshev polynomials satisfy the order recursion.

$$T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x) \ , \tag{10}$$

with initial conditions, $T_0(x) = 1$ and $T_1(x) = x$. The series expansions in cosines, Eq. (7), can now be expressed in terms of Chebyshev polynomials.

$$\begin{aligned} G_1'(x) &= 2T_{M_1}(x) + 2g_1(1)T_{M_1-1}(x) + \cdots + g_1(M_1) \ , \\ G_2'(x) &= 2T_{M_2}(x) + 2g_2(1)T_{M_2-1}(x) + \cdots + g_2(M_2) \ . \end{aligned} \tag{11}$$

Once the roots $\{x_i\}$ of $G_1'(x)$ and $G_2'(x)$ are determined, the corresponding LSF's are given by $\omega_i = \arccos x_i$. The mapping $x = \cos\omega$ maps the upper semi-circle in the z-plane to the real interval $[-1, +1]$. Therefore, all the roots x_i lie between -1 and $+1$, with the root corresponding to the lowest frequency LSF being the one nearest $+1$.

The Chebyshev polynomial series lends itself to an efficient and accurate evaluation which bypasses an expansion in powers of x. Let the series to be evaluated be represented as

$$Y(x) = \sum_{k=0}^{N-1} c_k T_k(x) \ . \tag{12}$$

Consider the backward recurrence relationship

$$b_k(x) = 2x\, b_{k+1}(x) - b_{k+2}(x) + c_k \ . \tag{13}$$

with initial conditions $b_N(x) = b_{N+1}(x) = 0$. This recursion is used to calculate $b_0(x)$ and $b_2(x)$. Then $Y(x)$ can be expressed in terms of $b_0(x)$ and $b_2(x)$,

$$\begin{aligned} Y(x) &= \sum_{k=0}^{N-1} [b_k(x) - 2x\, b_{k+1}(x) + b_{k+2}(x)]\, T_k(x) \\ &= b_0(x)T_0(x) + b_1(x)T_1(x) - 2x\, b_1(x)T_0(x) + \sum_{k=2}^{N-1} b_k(x)\,[T_k(x) - 2x\, T_{k-1}(x) + T_{k-2}(x)] \\ &= b_0(x) - x\, b_1(x) \\ &= \frac{b_0(x) - b_2(x) + c_0}{2} \ . \end{aligned} \tag{14}$$

The benefit of this formulation is that errors in the evaluation of $b_0(x)$ and $b_2(x)$ tend to cancel [9]. This results in a numerically stable evaluation of the Chebyshev polynomial series. Neglecting the factor of 2 which does not affect root locations, each evaluation can be computed with about N multiplies and $2N$ additions.

Using the above procedure to evaluate $G_1'(x)$ and $G_2'(x)$, the numerical algorithm which is used to solve for the roots can avoid altogether the need to compute cosine values. In fact as will be seen later, the roots in the x domain are more convenient than the ω values for reconversion to predictor coefficients

4. Numerical Solution for the Line Spectral Frequencies

In this section, a numerical algorithm to find the roots corresponding to the line spectral frequencies is developed. The basic task is to isolate the roots of $G_1'(x)$ by searching incrementally for intervals in which the sign changes. The search proceeds backwards from $x = 1$ since $G_1'(x)$ has the root nearest $x = 1$. The location of the root in an interval containing a sign change is refined by successive bisection of the root interval. The function values are determined using the backward recursion given in the previous section to compute the Chebyshev polynomials at a given argument value as needed. In this way, only two function values at a time need be stored. Given the interlacing property of the roots, the search for a root of $G_2'(x)$ starts from the position of the root of $G_1'(x)$ just found. The algorithm continues as before, but interchanges the roles of the functions as each root is found.

Two different precisions must be specified for the numerical algorithm. The initial evaluation interval, δ, must be sufficiently small so that two or more roots of the same function do not occur in the same interval. Let the roots be denoted by $\{x_i\}$ for $i = 1, 2, \ldots, P$, and let them be ordered such that $x_i > x_{i-1}$. The roots of $G_1'(x)$ (x_i with i odd) interlace with the roots of $G_2'(x)$ (x_i with i even). Recall that the root nearest $+1$ belongs to $G_1'(x)$. Then the initial evaluation interval must satisfy

$$\delta < \min_i (x_i - x_{i-2}) \ . \tag{15}$$

This guarantees that all roots can be found by examining sign changes.

A second increment, ϵ, specifies the acceptable uncertainty in root position. This value must be small enough that in switching the search for roots from one function to the other, a root is not missed or roots are not interchanged in order. To guarantee this, ϵ must be smaller than the minimum spacing between pairs of roots, one taken from each function,

$$\epsilon < \min_i (x_i - x_{i-1}) \ . \tag{16}$$

Experiments with speech data were conducted to determine reasonable values for δ and ϵ. Five utterances comprising 10 seconds of speech sampled at 8 kHz were used. Three utterances were spoken by males and two by females. In all cases, a 20 ms Hamming window was used to perform a 10'th order autocorrelation analysis. This ensures that the predictor is minimum phase [10]. The root locations corresponding to LSF's were determined to a high precision. A plot of $G_1'(x)$ and $G_2'(x)$ for a voiced segment of speech is shown in Fig. 2. It shows a tendency of roots to pair which requires ϵ to be significantly smaller than δ.

Histograms of root differences are shown in Fig. 3. The extreme values of the root differences are summarized in Table 1. These results indicate that $\delta = 0.02$ is sufficiently small to avoid missing sign changes. Each interval of length δ will be bisected to further resolve the root location. From

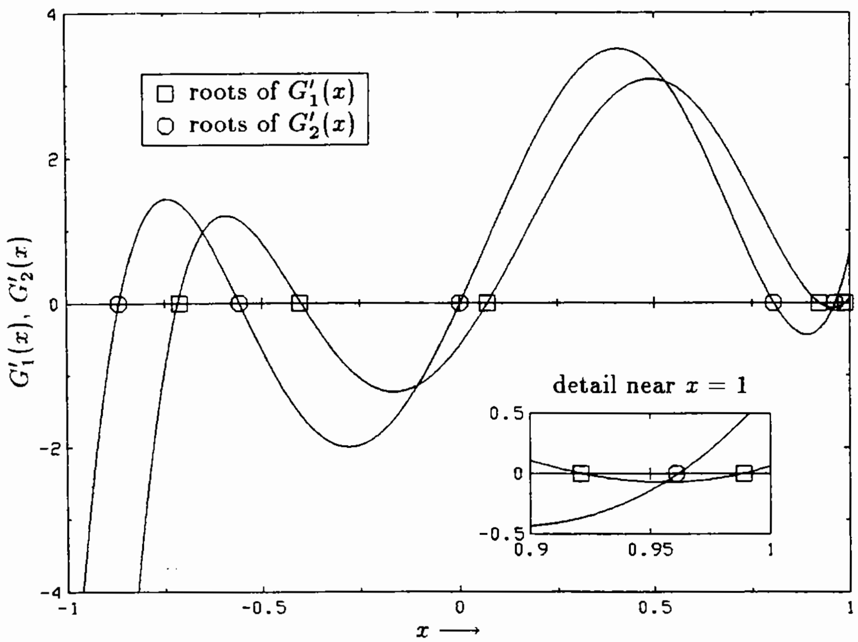


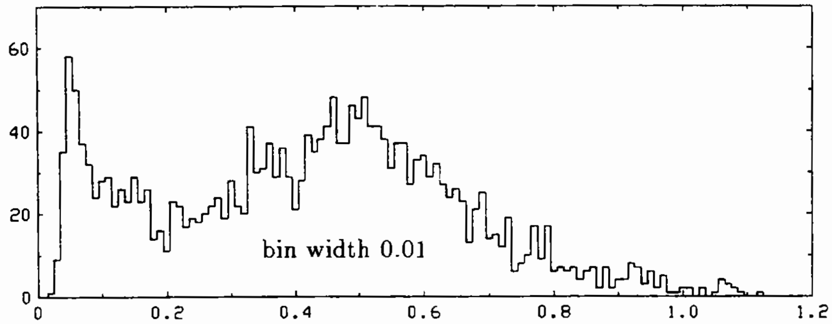
Fig. 2 Plots of $G'_1(x)$ and $G'_2(x)$ $(P = 10)$

the table, ϵ should be chosen to be less than 0.0015, implying that 4 bisections will be sufficient (for $\delta = 0.02$). While the worst case uncertainty in the x-domain is constant, the uncertainty in the ω-domain varies with ω due to the nonlinear relationship between x and ω. For the parameters given above and assuming 8 kHz sampling, the worst case uncertainty in the LSF's varies between 64 Hz at low and high frequencies down to 1.6 Hz at the middle frequencies. However, the uncertainy remains less than 10 Hz for the frequencies between 200 and 3800 Hz. Kang and Fransen [7] suggest a 10 Hz resolution in evaluating LSF's and furthermore find that coarse quantization of LSF's below 300 Hz does not affect speech quality.

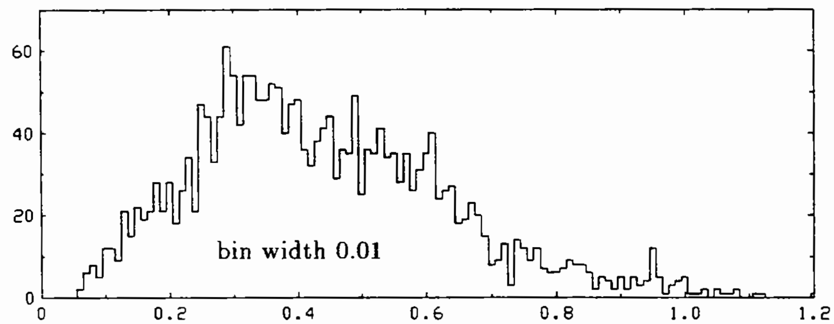
	Minimum Difference	Maximum Difference
$G'_1(x)$ only	0.0232	1.121
$G'_2(x)$ only	0.0564	1.195
$G'_1(x)$ and $G'_2(x)$	0.0015	0.946

Table 1 Root differences

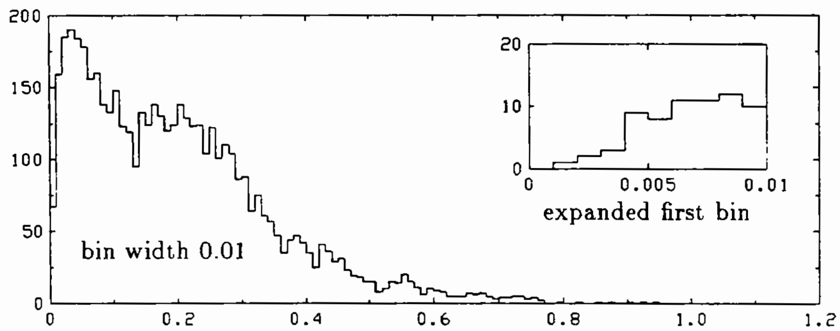
The root finding algorithm described above uses simple bisection to refine the estimates of the root positions. This brings the root position uncertainty below the threshold ϵ. As a last step, the root position is estimated by linearly interpolating between the already known function values. This results in an average error which is significantly smaller than the worst case value given by ϵ. The



(a) Differences for roots of $G_1'(x)$



(b) Differences for roots of $G_2'(x)$



(c) Differences between roots of $G_1'(x)$ and $G_2'(x)$

Fig. 3 Histograms of root differences

given root finding algorithm has been formulated to limit the number of function evaluations. Using the interlacing property of the roots, the root finding algorithm alternates between the polynomials. The initial search for intervals containing roots uses approximately $2/\delta + P$ evaluations. Bisection then uses an additional $\lceil \log_2(\delta/\epsilon) \rceil P$ evaluations. For the values of δ and ϵ given above, the number of function evaluations for a 10'th order LPC system is less than 150. The actual number is somewhat smaller than this value, since the search for roots can be terminated when all roots are found. This generally occurs before the entire interval $[-1, +1]$ has been examined. A listing of a program that finds the LSF's from a given set of predictor coefficients using the algorithm described in this section appears in Appendix B.

The routine to find the LSF's uses an algorithm which is extremely simple in structure and in which the number of function evaluations is relatively constant. The numerical analysis literature abounds with more sophisticated root finding algorithms. These will tend to find the roots with fewer function evaluations on the average. However, for most of these algorithms, the maximum number of function evaluations required is indeterminate. For use in a real-time environment, it is usually the worst case number of evaluations that is of concern. In addition, the more sophisticated algorithms come with a considerable program size penalty.

A slightly more complex root finding algorithm which combines bisection with inverse parabolic interpolation [11] merits consideration for some applications. It can be used to refine the root position when an interval containing a root has been identified. The worst case number of function evaluations for this algorithm is two or three times that for simple bisection, but the average number is smaller.

5. Conversion of LSF's to Predictor Coefficients

The conversion of LSF's to predictor coefficients is less computationally intensive than deriving LSF's from predictor coefficients. Each LSF ω_i gives rise to a second order polynomial factor of the form $1 - 2\cos\omega_i z^{-1} + z^{-2}$. These can be multiplied together to form the auxiliary polynomials directly. In this section, an alternate reconstruction process using the Chebyshev series representation will be formulated. This leads to an efficient reconstruction process which takes symmetries in the auxiliary polynomials into account.

The polynomials $G'_1(x)$ and $G'_2(x)$ are reconstructed from their roots by successive polynomial multiplication of the appropriate first order LSF polynomials,

$$\begin{aligned} G'_1(x) &= \prod_{k=1}^{M_1} 2(x - x_{2k-1}) \; . \\ G'_2(x) &= \prod_{k=1}^{M_2} 2(x - x_{2k}) \; . \end{aligned} \tag{17}$$

However it is not the coefficients of the powers of x that are desired, but the coefficients of the Chebyshev polynomial terms. Consider an N'th order polynomial expressed as a Chebyshev series,

$$Y_N(x) = \sum_{k=0}^{N} c_k T_k(x) \; . \tag{18}$$

Adding one more root factor to form an $N+1$'st order Chebyshev representation,

$$\begin{aligned} Y_{N+1}(x) &= 2(x - x_r) Y_N(x) \\ &= \sum_{k=-1}^{N+1} [c_{k-1} - 2x_r c_k + c_{k+1}] \, T_k(x) \; . \end{aligned} \tag{19}$$

This expression has been put in this form by applying the relation $2xT_k(x) = T_{k-1}(x) + T_{k+1}(x)$ (see Eq. (10)). For simplicity, c_k is defined to be zero for $k < 0$ and $k > N$. In addition, note that $T_{-1}(x) = T_1(x)$, which means that the term for $k = -1$ should be combined with the term for $k = 1$. The bracketed term in Eq. (19) is the coefficient of the k'th Chebyshev polynomial in the representation of $Y_{N+1}(x)$. This equation defines one step in the recursion to determine the coefficients of the Chebyshev representation from the root factors.

The Chebyshev series coefficients for $G'_1(x)$ and $G'_2(x)$ are determined from the root factors using the above recurrence relationship. The coefficients for $G_1(z)$ and $G_2(z)$ can be determined directly from these coefficients (compare Eq. (5) and Eq. (11)). This involves applying a factor of $1/2$ to all but one of the coefficients. In fact, multiplication by this factor can be avoided if the recursion is modified to apply directly to the coefficients of $G_1(z)$ and $G_2(z)$.

As the penultimate step, $G_1(z)$ and $G_2(z)$ must be multiplied by the polynomial terms with roots at ± 1 to give $F_1(z)$ and $F_2(z)$ (see Eq. (3)). This can be carried out on half of the total number of coefficients in these auxiliary polynomials (using symmetry) and involves only additions and subtractions. Finally, the coefficients of the prediction error filter are determined from

$$A(z) = \frac{F_1(z) + F_2(z)}{2} \tag{20}$$

A reconstruction procedure as described above will give a minimum phase prediction error filter. This follows directly from the fact that the reconstruction procedure is the step-by-step inverse of the procedure to find the LSF's. As long as the LSF's are distinct, and $F_1(z)$ and $F_2(z)$ are formed from alternating roots, the minimum phase property of the reconstructed prediction error filter is guaranteed.

A program that determines a minimum phase predictor from a set of LSF's appears in Appendix B. This routine requires about $P^2/4 - P/2$ multiplies and $P^2/2 + 2P - 4$ adds.† This procedure uses less than 1/4 of the number of multiplies and adds cited for the reconstruction procedure suggested by Kang and Fransen [7].

An alternative to converting the LSF's to predictor coefficients. is the use of filter structures that use the LSF's directly as parameters. This kind of structure implements $F_1(z)$ and $F_2(z)$ directly as cascaded second order sections [7]. This structure can be used as the basis of both the analysis (prediction error) filter and the corresponding synthesis filter. However, this form of filter requires more arithmetic operations per sample than a direct form filter using the predictor coefficients. The tradeoff is then between this extra computation which occurs for each sample of data processed and the computation required to convert LSF's to predictor coefficients. Kang and Fransen [7] show that for reasonable frame sizes in an LPC coder, conversion to predictor coefficients and the use of a direct form filter structure results in a lower operations count than the use of an LSF based filter structure. This conclusion is strengthened by the more efficient procedure to convert to direct form coefficients described here.

The reconstruction procedure given in this section has been expressed in terms of the roots in the x-domain. This complements the formulation for the procedure to derive the LSF's. In any transmission system. the LSF's must be quantized and coded. In order to avoid conversion of the LSF's to the ω-domain, the quantization procedure must be modified to work directly on the x_i values. Quantization of line spectral pairs with a position and difference parameter is an efficient coding option. This same strategy can be applied to the x_i values. The cosine non-linearity which relates the ω_i values to the x_i values is not so severe as to change the qualitative nature of the root difference statistics over the range of frequencies important for speech coding. Quantization of the x_i values leads to the possibility of working entirely in the x-domain. This has the merit of

† These counts apply for P even. For P odd, the number of operations is slightly smaller than given by these formulas. Note, also that the counts do not include P multiplies by the factor 1/2 and P multiplies by the factor 2.

completely avoiding the need to evaluate transcendental functions in either the conversion process or the reconversion process.

6. Summary and Conclusions

This paper has reported a method for converting predictor coefficients to a set of line spectral frequencies which can be used for both even and odd order LPC systems. The proposed method with the given interval parameters is highly accurate. The accuracy can be further increased by performing more bisections within the root interval, of course at the cost of more function evaluations. The use of an expansion in Chebyshev polynomials obviates the calculation of trigonometric functions and/or the storage of trigonometric tables calculated on a dense grid. The evaluation of these expansions makes use of an efficient and numerically stable algorithm. The root finding algorithm which determines the LSF's has been structured to limit the maximum number of function evaluations for a given accuracy constraint.

The reconversion of the LSF's to predictor coefficients is formulated in terms of a recursive calculation of the coefficients of the Chebyshev expansion. This gives a computationally efficient algorithm which takes into account inherent symmetries in the auxiliary polynomials. If the LSF's are expressed in the cosine domain, trigonometric computations can be avoided altogether. The predictor coefficients derived from a set of distinct LSF's give a minimum phase prediction error filter.

As a test of the overall procedure, LSF's were found using the procedure described in Section 4 for the speech data used previously. The analysis conditions are the same as specified earlier. The 10 LSF's for each frame of speech were reconverted to predictor coefficients by the procedure described in Section 5. The maximum difference between a reevaluated predictor coefficient and the original predictor coefficient was 3.8×10^{-5}.

Appendix A. Relationship Between LSF's and the Predictor Roots

The relationship between the LSF's and the roots of the prediction error filter $A(z)$ is explored in this appendix. In the main text, examples for real speech data have shown that there is indeed a tendency for the LSF's to cluster around the angular positions corresponding to roots of $A(z)$ when these are close to the unit circle. Some insight into the clustering phenomenon can be obtained by examining another formulation related to that for the LSF's.

Schussler [8] has given a stability theorem for a polynomial $A(z)$. The stability condition is expressed in terms of two auxiliary polynomials,

$$\begin{aligned} \tilde{F}_1(z) &= A(z) + z^{-P} A(z^{-1}) , \\ \tilde{F}_2(z) &= A(z) - z^{-P} A(z^{-1}) . \end{aligned} \tag{A.1}$$

The polynomial $A(z)$ has all its roots within the unit circle if and only if: (1) the roots of $\tilde{F}_1(z)$ and $\tilde{F}_2(z)$ are on the unit circle; (2) the roots are simple and separate each other; and (3) $|a(P)| < 1$.† These auxiliary polynomials differ from those used for LSF's by being of order P instead of order P - 1. In Schussler's theorem, the auxiliary polynomials are formed by adding a time-reversed polynomial to the original system polynomial. For the LSF formulation, the auxiliary polynomials are formed by adding the time-reversed polynomial with an additional unit time shift. As will be seen, the formulation derived from Schussler's theorem has drawbacks as a pseudo-LSF representation.

Schussler's auxiliary polynomials become identical to those in the LSF formulation if $A(z)$ is artificially extended with a zero valued coefficient, $a(P+1) = 0$. However, if Schussler's theorem is applied directly (without appending a zero valued coefficient), the resulting symmetric and anti-symmetric polynomials are each of one degree lower than the $F_1(z)$ and $F_2(z)$ polynomials in the LSF formulation. As a result, the roots of Schussler's polynomials cannot be used to uniquely reconstruct $A(z)$. An additional quantity must be specified. This could be the coefficient $a(P)$, which in addition is known to have magnitude less than unity for a minimum phase polynomial.

Consider rewriting $\tilde{F}_1(z)$ and $\tilde{F}_2(z)$ as

$$\begin{aligned} \tilde{F}_1(z) &= A(z)[1 + \tilde{R}(z)] , \\ \tilde{F}_2(z) &= A(z)[1 - \tilde{R}(z)] , \end{aligned} \tag{A.2}$$

where the ratio filter $\tilde{R}(z)$ is defined as

$$\tilde{R}(z) = \frac{z^{-P} A(z^{-1})}{A(z)} . \tag{A.3}$$

This is similar to the ratio filter defined in Section 2 (see Eq. (8)). Note that the only difference between the ratio filter as defined here and that for the LSF formulation is an extra z^{-1} delay term

† The last condition was added Gnanasekaran [12]. Conditions (1) and (2) by themselves also hold if $A(z)$ has all its roots outside the unit circle.

in the latter. In either case, the ratio filter has an allpass response. The auxiliary polynomials have roots at those points on the unit circle at which the phase of the ratio filter passes through multiples of π.

A simple example will point out some of the ramifications of the extra delay term associated with the LSF formulation. Consider an $A(z)$ which has conjugate pairs of roots near the unit circle. It can be shown that the LSF formulation gives pairs of roots bracketing the angle corresponding to the roots of $A(z)$. These LSF's coalesce as the roots of $A(z)$ approach the unit circle. By contrast, the formulation that arises from Schussler's stability theorem gives single roots at the angular position of the roots of $A(z)$, but in addition, roots appear midway between these positions. Fig. A.1 shows a phase plot of both $R(e^{j\omega})$ (LSF formulation) and $\hat{R}(e^{j\omega})$ (Schussler's formulation) for the case of an $A(z)$ which has 3 pairs of conjugate roots, each with magnitude 0.99. The angular positions of the roots correspond to ω equal to $\pi/4$, $\pi/2$ and $3\pi/4$. Symbols are used to mark the places where the phase angle crosses a multiple of π. The frequencies corresponding to these points are the frequencies corresponding to the roots of the auxiliary polynomials. In the vicinity of the roots of $A(z)$, the phase undergoes an excursion through nearly 2π radians for both formulations. However, the phase offset due to the linear phase component in the LSF formulation is enough to substantially change the positions of the roots of the LSF auxiliary polynomials from those for Schussler's formulation.

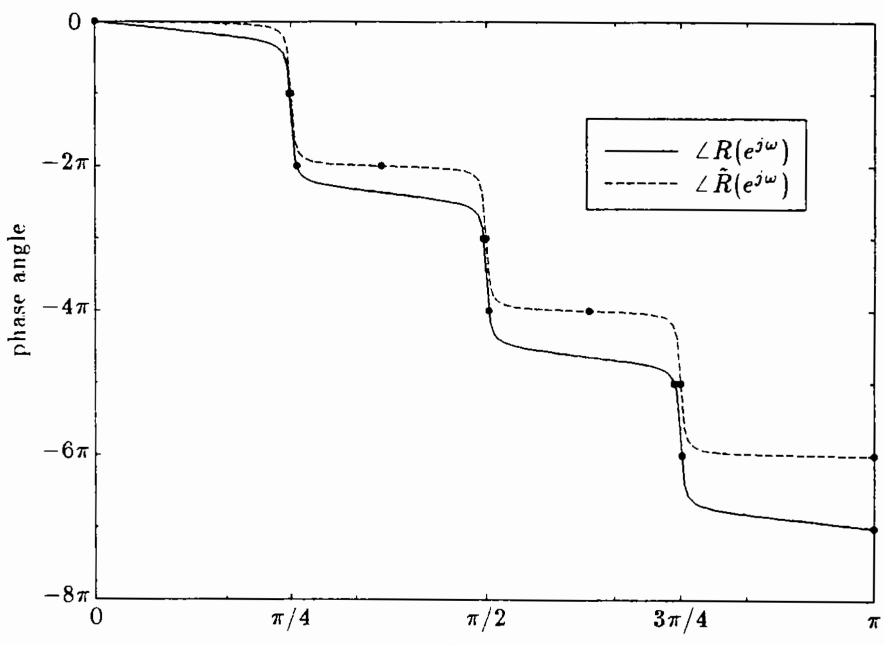


Fig. A.1 Phase responses of $R(e^{j\omega})$ and $\hat{R}(e^{j\omega})$ — $A(z)$ has roots at $0.99\,e^{\pm j\pi/4}$, $0.99\,e^{\pm j\pi/2}$ and $0.99\,e^{\pm j3\pi/4}$

Kang and Fransen [4] point out that the group delay of $R(z)$ increases in the neighbourhood of a pole of $A(z)$. This clearly also applies to $\hat{R}(z)$ since $\hat{R}(z)$ differs from $R(z)$ only by a linear phase term. However, the previous example has shown that peaking of the group delay, or the equivalent rapid change in phase angle, is not sufficient in itself for a pair of LSF's to occur close together. The linear phase component is an important element in keeping the LSF's together in the vicinity of a resonance of $A(z)$.

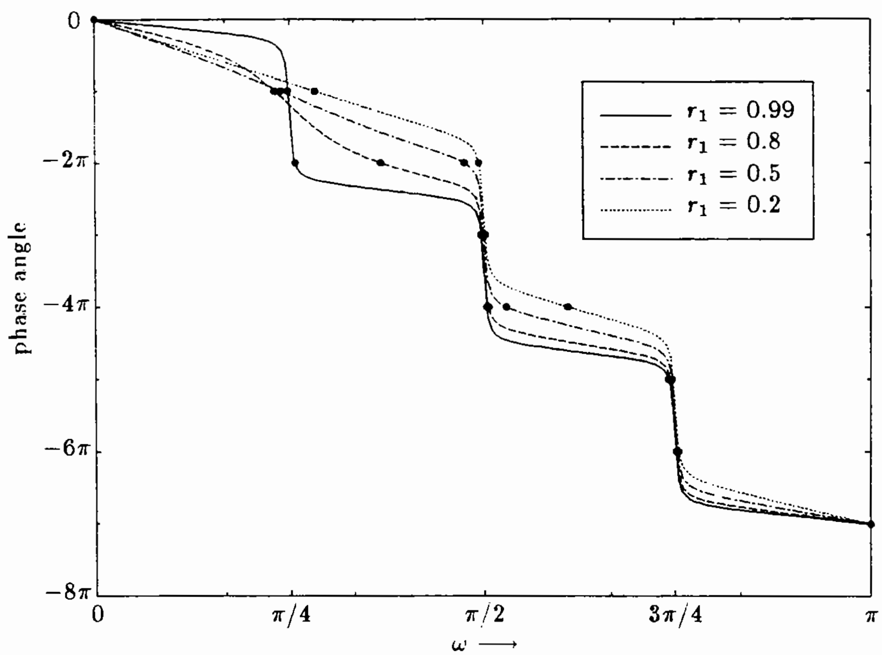


Fig. A.2 Phase response of $R(e^{j\omega})$ — $A(z)$ has roots at $r_1 e^{\pm j\pi/4}$, $0.99\,e^{\pm j\pi/2}$ and $0.99\,e^{\pm j3\pi/4}$

The previous example can be explored further. For this discussion, only the LSF formulation will be considered. Consider changing the radial position of the first pair of roots of $A(z)$ (at angular position $\omega = \pi/4$) while keeping the other root positions constant. As the radius of this pair of roots is reduced to 0.8, the phase change occurs more gently. The lowest frequency LSF's are now spread apart (see Fig. A.2). When the radius is reduced to 0.5, the phase change is even more gradual and effects the overall phase in the vicinity of $\omega = \pi/2$. There are now three closely spaced LSF's near $\omega = \pi/2$. As the radius is further reduced to 0.2, the LSF's take on a different configuration. The lowest LSF has moved to just above $\omega = \pi/4$, the second and third LSF's occur close together near $\omega = \pi/2$, and the fourth LSF lies midway between these LSF's and the two paired LSF's near $\omega = 3\pi/4$. This root configuration shows that the sharp resonance at $\omega = \pi/2$ is signalled by a close spacing between the second and third LSF's. In the coding scheme proposed by Kang and Fransen [7], LSF's are coded in pairs specified by a centre frequency and an offset frequency. In this case, the coding would be applied to two pairs of LSF's, each of which has a relatively large offset. After quantization of the offset, this coding scheme may not adequately represent the fact that the upper LSF from one pair and the lower LSF from another pair are closely spaced.

This appendix has shown examples that point out that the relationship between the positions of the roots of $A(z)$ and the LSF configuration is more fragile than the literature would lead one to believe. Nonetheless, this does not invalidate the use of LSF's in speech coding. Clearly, the extensive perceptual testing carried out by Kang and Fransen shows that in spite of hurdles in the interpretation of the LSF's in terms of the spectral features or roots of $A(z)$, LSF's do efficiently represent the perceptually important features of speech spectra.

Appendix B. Programs to Calculate LSF's

This appendix gives listing for programs which implementing the conversion from direct form predictor coefficients to LSF's and vice versa. The subroutines have been coded in standard FORTRAN 77 for the most part, except that structured DO constructs have been used for enhanced readibility. The routines have been written so as to keep the code compact and maximize clarity by paralleling the analysis given in the main text. This style does compromise execution time to some extent.

The routines given here represent the LSF's as $\omega_i/2\pi$ in order to be compatible with the standard description of the LSF's. The invokation of the inverse cosine function in subroutine PTOLSF and the cosine function in subroutine LSFTOP can be eliminated if the LSF's are retained in the x domain.

B.1 Predictor Coefficients to LSF's

B.1.1 PTOLSF

```
* MODULE:
*       SUBROUTINE PTOLSF (PRCOF, FRLSF, NPOLE)
*
*
* PURPOSE:
*       This subroutine converts a set of predictor coefficients to
*       a set of normalized line spectral frequencies.
*
*
* DESCRIPTION:
*       The transfer function of the predictor filter is transformed
*       into two reciprocal polynomials having roots on the unit circle.
*       These roots of these polynomials interlace. It is these roots
*       that determine the line spectral frequencies. The two reciprocal
*       polynomials are expressed as series expansions in Chebyshev
*       polynomials with roots in the range -1 to +1. The inverse cosine
*       of the roots of the Chebyshev polynomial expansion gives the line
*       spectral frequencies. If NPOLE line spectral frequencies are not
*       found, this routine signals an error condition.
*
*
* PARAMETERS:
*       PRCOF  - Predictor coefficients (NPOLE values)
*   (*) FRLSF  - Array of NPOLE line spectral frequencies (in ascending
*                order). Each line spectral frequency lies in the range
*                range 0 to 0.5.
*       NPOLE  - Order of the system (at most 20)
*
*
```

```
* ROUTINES REQUIRED:
*       CHEBPS - Evaluates a series expansion in Chebyshev polynomials
*       HALT   - Signals an error condition
*
*
      SUBROUTINE PTOLSF (PRCOF, FRLSF, NPOLE)


      PARAMETER (MXPOLE=20,PI2=6.283185307,RESL=0.02,NBIS=4)

      REAL PRCOF(NPOLE),F1(0:(MXPOLE+1)/2),F2(0:(MXPOLE+1)/2)
      REAL T(0:(MXPOLE+1)/2,2),FRLSF(NPOLE)
      INTEGER NC(2)
      LOGICAL ODD


* Determine the number of coefficients in each of the polynomials
* with coefficients T(.,1) and T(.,2).
* ODD is true when NPOLE is odd
      ODD=MOD(NPOLE,2).NE.0
      IF (ODD) THEN
        NC(2)=(NPOLE+1)/2
        NC(1)=NC(2)+1
      ELSE
        NC(2)=NPOLE/2+1
        NC(1)=NC(2)
      END IF

* Let D=z**(-1), the unit delay, then the predictor filter with
* N coefficients is
*
*           N        n
*   P(D) = SUM p(n) D   .
*          n=1
*
* The error filter polynomial is A(D)=1-P(D) with N+1 terms.
* Two auxiliary polynomials * are formed from the error filter polynomial,
*   F1(D) = A(D) + D**(N+1) A(D**(-1))  (N+2 terms, symmetric)
*   F2(D) = A(D) - D**(N+1) A(D**(-1))  (N+2 terms, anti-symmetric)

* Establish the symmetric polynomial F1(D) and the anti-symmetric
* polynomial F2(D)
* Only about half of the coefficients are evaluated since the
* polynomials are symmetric and will later be reduced in order by
* division by polynomials with roots at +1 and -1
      F1(0)=1.0
      J=NPOLE
      DO I=1,NC(1)-1
        F1(I)=-PRCOF(I)-PRCOF(J)
        J=J-1
      END DO
```

```
      F2(0)=1.0
      J=NPOLE
      DO I=1,NC(2)-1
        F2(I)=-PRCOF(I)+PRCOF(J)
        J=J-1
      END DO

* N even,  F1(D)  includes a factor 1+D,
*          F2(D)  includes a factor 1-D
* N odd,   F2(D)  includes a factor 1-D**2
* Divide out these factors, leaving even order symmetric polynomials,
* M is the total number of terms and Nc is the number of unique terms,
*   N       polynomial            M         Nc=(M+1)/2
* even,  G1(D) = F1(D)/(1+D)     N+1          N/2+1
*        G2(D) = F2(D)/(1-D)     N+1          N/2+1
* odd,   G1(D) = F1(D)           N+2        (N+1)/2+1
*        G2(D) = F2(D)/(1-D**2)   N          (N+1)/2
      IF (ODD) THEN
        DO I=2,NC(2)-1
          F2(I)=F2(I)+F2(I-2)
        END DO
      ELSE
        DO I=1,NC(1)-1
          F1(I)=F1(I)-F1(I-1)
          F2(I)=F2(I)+F2(I-1)
        END DO
      END IF

* To look for roots on the unit circle, G1(D) and G2(D) are
* evaluated for D=exp(ja). Since G1(D) and G2(D) are symmetric,
* they can be expressed in terms of a series in cos(na) for D on
* the unit circle. Since M is odd and D=exp(ja)
*
*          M-1        n
*   G1(D) = SUM f1(n) D             (symmetric, f1(n) = f1(M-1-n))
*          n=0
*                                   Mh-1
*        = exp(j Mh a) [ f1(Mh) + 2 SUM f1(n) cos((Mh-n)a) ]
*                                   n=0
*                       Mh
*        = exp(j Mh a) SUM t1(n) cos(na) ,
*                       n=0
*
* where Mh=(M-1)/2=Nc-1. The Nc=Mh+1 coefficients t1(n) are defined as
*   t1(n) =    f1(Nc-1) ,     n=0,
*         = 2 f1(Nc-1-n) ,   n=1,...,Nc-1.
```

```
* The next step is to identify cos(na) with the Chebyshev polynomial
* T(n,x). The Chebyshev polynomials satisfy T(n,cos(x)) = cos(nx).
* Then omitting the exponential delay term which does not affect the
* positions of the roots on the unit circle, the series expansion in
* terms of Chebyshev polynomials is
*
*           Nc-1
*   T1(x) = SUM t1(n) T(n,x)
*           n=0
*
* The domain of T1(x) is -1 < x < +1. For a given root of T1(x), say
* x0, the corresponding position of the root of F1(D) on the unit
* circle is exp(j arccos(x0)).

* Establish the coefficients of the series expansion in Chebyshev
* polynomials
      T(0,1)=F1(NC(1)-1)
      J=NC(1)-2
      DO I=1,NC(1)-1
        T(I,1)=2.0*F1(J)
        J=J-1
      END DO
      T(0,2)=F2(NC(2)-1)
      J=NC(2)-2
      DO I=1,NC(2)-1
        T(I,2)=2.0*F2(J)
        J=J-1
      END DO

* Sample at equally spaced intervals between -1 and 1 to look for sign
* changes. RESL is chosen small enough to avoid problems with multiple
* roots in an interval. After detecting a sign change, successive
* bisections and linear interpolation are used to find roots corresponding
* to LSF frequencies. Since the roots of the two polynomials interlace,
* the search alternates between the polynomials T(.,1) and T(.,2).
* IP is either 1 or 2 depending on which polynomial is being examined.
      NF=0
      IP=1
      XLOW=1.0
      YLOW=CHEBPS(XLOW,T(0,IP),NC(IP))

      DO WHILE (XLOW.GT.-1. .AND. NF.LT.NPOLE)
        XHIGH=XLOW
        YHIGH=YLOW
        XLOW=MAX(XHIGH-RESL,-1.0)
        YLOW=CHEBPS(XLOW,T(0,IP),NC(IP))
        IF (YLOW*YHIGH.LE.0.0) THEN
          NF=NF+1
```

```
* Bisections of the interval containing a sign change
            DO I=1,NBIS
              XMID=0.5*(XLOW+XHIGH)
              YMID=CHEBPS(XMID,T(0,IP),NC(IP))
              IF (YLOW*YMID.LE.0.0) THEN
                YHIGH=YMID
                XHIGH=XMID
              ELSE
                YLOW=YMID
                XLOW=XMID
              END IF
            END DO

* Linear interpolation in the subinterval with a sign change
            XINT=XLOW-YLOW*(XHIGH-XLOW)/(YHIGH-YLOW)
            FRLSF(NF)=ACOS(XINT)/PI2

* Start the search for the roots of the next polynomial at
* the estimated location of the root just found
            IP=3-IP
            XLOW=XINT
            YLOW=CHEBPS(XLOW,T(0,IP),NC(IP))
          END IF
        END DO

* Halt if NPOLE frequencies have not been found
        IF (NF.NE.NPOLE)
     -     CALL HALT ('PTOLSF - Too few frequencies computed')

        RETURN


        END
```

B.1.2 CHEBPS

```
* MODULE:
*       FUNCTION CHEBPS (X, COF, N)
*
*
* PURPOSE:
*       This function evaluates a series expansion in Chebyshev
*       polynomials.
*
*
```

```
* DESCRIPTION:
*       The series expansion in Chebyshev polynomials is defined as
*
*                  N-1
*         Y(x) = SUM c(i) T(i,x) ,
*                  i=0
*
*       where Y(x) is the resulting value (Y(x) = CHEBPS(...)),
*             N is the order of the expansion,
*             c(i) is the coefficient for the i'th order Chebyshev
*                  polynomial (c(i) = COF(i+1)), and
*             T(i,x) is the i'th order Chebyshev polynomial
*                  evaluated at x.
*       The Chebyshev polynomials satisfy the recursion
*         T(i,x) = 2x T(i-1,x) - T(i-2,x),
*       with the initial conditions T(0,x)=1 and T(1,x)=x. This
*       routine evaluates the expansion using a backward recursion
*       to obtain a numerically stable solution.
*
*
* PARAMETERS:
*   (*) CHEBPS - Resulting function value
*
*       X      - Input value
*       COF    - Array of coefficient values. COF(i) is the coefficient
*                corresponding to the Chebyshev polynomial of order i-1
*       N      - Order of the polynomial and number of coefficients
*
*
      FUNCTION CHEBPS (X, COF, N)


      REAL COF(0:N-1)


* Consider the backward recursion b(i,x)=2xb(i+1,x)-b(i+2,x)+c(i),
* with initial conditions b(N,x)=0 and b(N+1,x)=0.
* Then dropping the dependence on x, c(i)=b(i)-2xb(i+1)+b(i+2).
*
*         N-1
*  Y(x) = SUM c(i) T(i)
*         i=0
*
*         N-1
*       = SUM [b(i)-2xb(i+1)+b(i+2)] T(i)
*         i=0
*                                          N-1
*       = b(0)T(0)+b(1)T(1)-2xb(1)T(0)  + SUM b(i)[T(i)-2xT(i-1)+T(i-2)]
*                                          i=2
```

```
* The term inside the sum is zero because of the recursive relationship
* satisfied by the Chebyshev polynomials. Then substituting the values
* T(0)=1 and T(1)=x, Y(x) is expressed in terms of the difference between
* b(0) and b(2) (errors in b(0) and b(2) tend to cancel),
*
*   Y(x) = b(0)-xb(1) = [b(0)-b(2)+c(0)] / 2

      B1=0.0
      B0=0.0
      DO I=N-1,0,-1
        B2=B1
        B1=B0
        B0=(2.*X)*B1-B2+COF(I)
      END DO
      CHEBPS=0.5*(B0-B2+COF(0))

      RETURN


      END
```

B.2 LSF's to Predictor Coefficients

B.2.1 LSFTOP

```
* MODULE:
*       SUBROUTINE LSFTOP (FRLSF, PRCOF, NPOLE)
*
*
* PURPOSE:
*       This subroutine converts a set of normalized line spectral
*       frequencies to the equivalent set of predictor coefficients.
*
*
* DESCRIPTION:
*       The line spectral frequencies are assumed to be frequencies
*       corresponding to roots on the unit circle. Alternate roots on
*       the unit circle belong to two polynomials. These polynomials
*       are formed by polynomial multiplication of factors representing
*       conjugate pairs of roots. Additional factors are used to give
*       a symmetric polynomial and an anti-symmetric polynomial. The sum
*       (divided by 2) of these polynomials gives the predictor
*       polynomial.
*
*
```

```
* PARAMETERS:
*       FRLSF  - Array of NPOLE line spectral frequencies (in ascending
*                order). Each line spectral frequency lies in the range
*                0 to 0.5.
*   (*) PRCOF  - Output of array of NPOLE predictor coefficients
*       NPOLE  - Order of the system (at most 20)
*
*
* ROUTINES REQUIRED:
*       CONVSM - Convolves coefficients for symmetric polynomials
*
*
      SUBROUTINE LSFTOP (FRLSF, PRCOF, NPOLE)


      PARAMETER (MXPOLE=20,PI2=6.283185307)

      REAL FRLSF(NPOLE),PRCOF(NPOLE)
      REAL F1(0:(MXPOLE+1)/2),F2(0:MXPOLE/2)
      DATA F1(0)/1.0/,F2(0)/1.0/


* Each line spectral frequency w contributes a second order
* polynomial of the form Y(D)=1-2*cos(w)*D+D**2. These polynomials
* are formed for each frequency and then multiplied together.
* Alternate line spectral frequencies are used to form two polynomials
* with interlacing roots on the unit circle. These two polynomials
* are again multiplied by 1+D and 1-D if NPOLE is even or by 1 and
* 1-D**2 if NPOLE is odd. This gives the symmetric and anti-symmetric
* polynomials that in turn are added to give the predictor coefficients.

* Form a symmetric F1(D) by multiplying together second order
* polynomials corresponding to odd numbered LSF's.
* This procedure is equivalent to the reconstruction of a Chebyshev
* polynomial representation from its root factors.
      NC=0
      DO I=1,NPOLE,2
        A=-2.0*COS(PI2*FRLSF(I))
        CALL CONVSM(F1(1),NC,A)
      END DO

* Form a symmetric F2(D) by multiplying together second order
* polynomials corresponding to even numbered LSF's
      NC=0
      DO I=2,NPOLE,2
        A=-2.0*COS(PI2*FRLSF(I))
        CALL CONVSM(F2(1),NC,A)
      END DO
```

```
* Both F1(D) and F2(D) are symmetric, with leading coefficient
* equal to unity. Exclusive of the leading coefficient, the
* number of coefficients needed to specify F1(D) and F2(D) is:
*  NPOLE     F1(D)        F2(D)
*  even     NPOLE/2     NPOLE/2
*   odd   (NPOLE+1)/2  (NPOLE-1)/2

       IF (MOD(NPOLE,2).NE.0) THEN

***** NPOLE odd
* F2(D) is multiplied by the factor (1-D**2)
         M=(NPOLE-1)/2
         DO I=M,2,-1
           F2(I)=F2(I)-F2(I-2)
         END DO

* Form the predictor filter coefficients
* Note that F1(D) is symmetric and F2(D) is now anti-symmetric.
* Since only the first half of the coefficients are available,
* symmetries are used to get the other half.
         K=NPOLE
         DO I=1,M
           PRCOF(I)=-0.5*(F1(I)+F2(I))
           PRCOF(K)=-0.5*(F1(I)-F2(I))
           K=K-1
         END DO
         PRCOF(K)=-0.5*F1(K)
       ELSE

***** NPOLE even
* F1(D) is multiplied by the factor (1+D)
* F2(D) is multiplied by the factor (1-D)
         M=NPOLE/2
         DO I=M,1,-1
           F1(I)=F1(I)+F1(I-1)
           F2(I)=F2(I)-F2(I-1)
         END DO

* Form the predictor filter coefficients
* Note that F1(D) is symmetric and F2(D) is now anti-symmetric.
         K=NPOLE
         DO I=1,M
           PRCOF(I)=-0.5*(F1(I)+F2(I))
           PRCOF(K)=-0.5*(F1(I)-F2(I))
           K=K-1
         END DO

       END IF

       RETURN


       END
```

B.2.2 CONVSM

```
* MODULE:
*       SUBROUTINE CONVSM (X, N, A)
*
*
* PURPOSE:
*       This routine convolves the coefficients of a symmetric polynomial
*       with the coefficients of a three term polynomial.
*
*
* DESCRIPTION:
*       This routine convolves two sets of coefficients to form
*       an output coefficient array. If the coefficients are
*       considered to be polynomial coefficients, this operation
*       is equivalent to polynomial multiplication.
*
*       The input coefficient array contains the coefficients for
*       a monic (leading coefficient equal to one), symmetric
*       polynomial. The polynomial has an odd number of terms,
*         X(D) = 1 + x(1)D + x(2)D**2 + ... + x(N)D**N
*                + x(N-1) D**(N+1) + ... + x(1) D**(2N-1) + D**(2N)
*       Note that the polynomial is specified by only N coefficients,
*       x(1),...,x(N). The other polynomial is
*         Y(D) = 1 + aD + D**2 ,
*       which is also a monic, symmetric polynomial with an odd number
*       of coefficients. It is specified by the single coefficient a.
*       The product of X(D) and Y(D) has 2N+3 coefficients. However,
*       the product is also a monic, symmetric polynomial and hence can
*       be specified by N+1 terms.
*
*
* PARAMETERS:
*   (*) X      - Input / output coefficient array. On input, X
*                contains N coefficients specifying a monic,
*                symmetric polynomial with a total of 2N+1 terms.
*                On output, X contains the N+1 coefficients resulting
*                from the convolution of the input polynomial
*                coefficients with those for a three term monic,
*                symmetric polynomial. The resulting N+1 coefficients
*                specify a 2N+3 term monic, symmetric polynomial.
*   (*) N      - On input N is the number of coefficients in the
*                input array X. On output, N is the number of
*                coefficients in X, (N <-- N+1).
*       A      - Input coefficient for a three term monic, symmetric
*                polynomial
*
*
      SUBROUTINE CONVSM (X, N, A)


      REAL X(*)
```

```
* Consider a monic, symmmetric polynomial with an odd number of
* coefficients (2N+1). This polynomial can be specified by N unique
* coefficients, x(1),...,x(N). Let
*  X(D) = x(0) + x(1)D + x(2)D**2 + ... + x(2N) d**(2N)
*       x(0) = x(2N) = 1  (monic and symmetric)
*       x(1) = x(2N-1)
*        ...    ...
*       x(n) = x(2N-n)    (general term)
*        ...    ...
*       x(N-1) = X(N+1)
*       x(N)               (not paired)
* Consider another monic, symmetric polynomial with 3 coefficients
*  Y(D) = 1 + aD + D**2
* The convolution of the coefficients of X(D) and Y(D) gives another
* monic, symmetric polynomial with an odd number of coefficients.
* Let Z(D) = Y(D) X(D), then
*       z(0) = x(0) = 1
*       z(1) = x(1) + ax(0) = x(1) + a
*       z(2) = x(2) + ax(1) + x(0) = x(2) + ax(1) + 1
*       z(3) = x(3) + ax(2) + x(1)
*       ...     ...
*       z(n) = x(n) + ax(n-1) + x(n-2)  ... general term
*       ...     ...
*       z(N-1) = x(N-1) + ax(N-2) + x(N-3)
*       z(N) = x(N) + ax(N-1) + x(N-2)
*       z(N+1) = x(N+1) + ax(N) + x(N-1) = 2x(N+1) + ax(N)
*  notes:
*  1) z(0) need not be calculated
*  2) terms z(N+2),...,z(2N+2) need not be calculated since they
*     can be obtained by symmetry
*  3) terms z(3),...z(N) are of the same form as the general term
*  4) term z(N+1) uses the symmetry of X(D), x(N+1)=x(N-1)

* Convolve the coefficients by summing shifted versions
* of X(.) weighted by a. By choosing the order of the operations
* appropriately, the result can overlay the input array X(.).
        IF (N.GE.2) THEN
          X(N+1)=X(N-1)
          DO K=N+1,3,-1
            X(K)=X(K)+A*X(K-1)+X(K-2)
          END DO
          X(2)=X(2)+A*X(1)+1.0
          X(1)=X(1)+A
        ELSE IF (N.EQ.1) THEN
          X(2)=2.0+A*X(1)
          X(1)=X(1)+A
        ELSE IF (N.EQ.0) THEN
          X(1)=A
        END IF
```

```
* Update the number of coefficients in X
      N=N+1

      RETURN

      END
```

References

1. H. Wakita. "Linear prediction voice synthesizers: line spectrum pairs (LSP) is the newest of several techniques", *Speech Technology*, Fall 1981.
2. F. K. Soong and B-W Juang, "Line spectrum pair (LSP) and speech data compression", *Proc. Int. Conf. on Acoustics, Speech and Signal Processing*, San Diego, California, pp. 1.10.1–1.10.4, March 1984.
3. J. R. Crosmer and T. P. Barnwell, III, "A low bit rate segment vocoder base on line spectrum pairs", *Proc. Int. Conf. on Acoustics, Speech and Signal Processing*, Tampa, Florida, pp. 7.2.1–7.2.4, April 1985.
4. G. S. Kang and L. J. Fransen. "Application of line spectrum pairs to low bit rate speech encoders", *Proc. Int. Conf. on Acoustics, Speech and Signal Processing*, Tampa, Florida, pp. 7.3.1–7.3.4, April 1985.
5. J. D. Markel and A. H. Gray, Jr., *Linear Prediction of Speech*, Springer-Verlag, 1976.
6. F. Itakura. "Line spectrum representation of linear predictor coefficients of speech signals", *Journal of the Acoustical Society of America*. vol. 57. S35(A). 1975.
7. G. S. Kang and L. J. Fransen, "Low bit rate speech encoders based on line spectrum frequencies (LSFs)", *Naval Research Laboratory Report 8857*. November 1984.
8. H. W. Schussler. "A stability theorem for discrete systems", *IEEE Trans. Acoust., Speech, Signal Processing*. vol. ASSP-24, pp. 87–89, February 1976.
9. L. Fox and I. B. Parker. *Chebyshev Polynomials in Numerical Analysis*, Oxford University Press, London, 1968.
10. L. R. Rabiner and R. W. Schafer, *Digital Processing of Speech Signals*. Prentice-Hall, 1978.
11. R. P. Brent. "An algorithm with guaranteed convergence for finding a zero of a function", *The Computer Journal*, vol. 14, pp. 422–425, 1971.
12. R. Gnanasekaran, "A note on the new 1-D and 2-D stability theorems for discrete systems", *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-29, pp. 1211–1212, December 1981.