THE DESIGN OF A SERIAL MSK FILTER

by

Mark Earl Flin

B.B., Kansas State University, 1978

A Master' Report

Submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Electrical Engineering

KANSAS STATE UNIVERSITY

Manhattan, Kansas

1980

Approved by:

Donald R. Hummels

Major Professor

TABLE OF CONTENTS

LIST OF FIGURES

Figure

THE DESIGN OF A SERIAL MSK FILTER

I. INTRODUCTION

This paper describes the design of a realizable filter for use in a simple Minimum-Shift-Keying (MSK) transmitter. There are a number of ways to model a transmitter of this type. The model shown in Figure 1 [1] was chosen because of its physical and analytical simplicity.

The transmitter of Figure 1 consists of a simple biphase modulator followed by a linear, time invariant filter, with impulse response h(t). Data in a serial format is input to the balanced modulator and the desired MSK signal is available at the output of the filter.

For MSK generation, the filter of Figure 1 has impulse response given by [1]

$$h(t) = \begin{cases} \sin 2\pi f_2 t & , o \leq t \leq T \\ 0 & , \text{elsewhere} \end{cases} \tag{1}$$

where T is the duration of a message bit and f_2 is described below. During any keying interval of duration T seconds, one of two frequencies, f_1 or f_2, is transmitted, where f_1 and f_2 have a special relationship to the link bit rate 1/T. This special relationship, for some selected integer n, is as follows,

$$f_1 = \frac{n}{2T}$$
$$f_2 = \frac{n+1}{2T} \tag{2}$$

i.e. they differ by half the bit rate.

It should be recognized that the response of (1) is ideal, and that a filter with this impulse response is not realizable. That is, it is not possible to build this filter using non-ideal, real world components. This being the case, an approximation to the ideal filter is required; one which is realizable, with impulse response matching that of the ideal

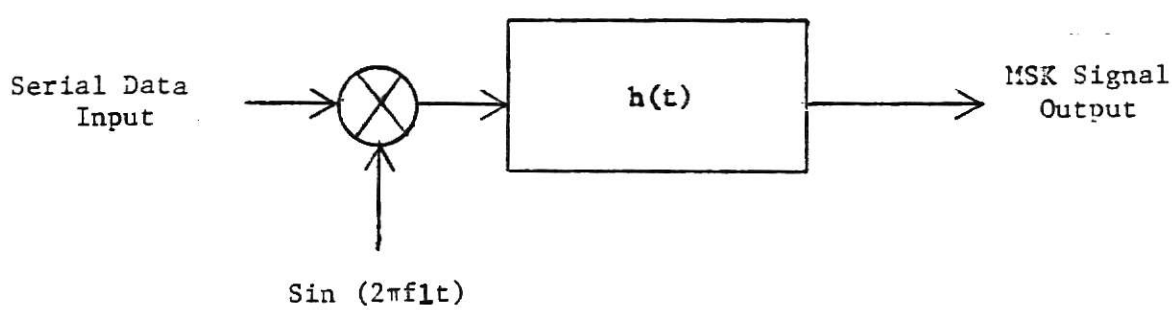

Figure 1 An MSK Transmitter

filter with minimum error. It is convenient for analytic reasons to work with the equivalent low-pass representation [2] rather than the actual impulse response of the filter. In the following sections of this report methods are developed for approximating the equivalent low-pass response of the ideal filter with realizable configurations. The technique employed is to adjust the parameters of the realizable filter so as to minimize an error criterion by using the method of steepest descent A filter design procedure which uses the method of steepest descent is developed in Section II. Some results for filter configurations of interest are given in Section III.

II. FILTER DESIGN USING THE METHOD OF STEEPEST DESCENT

General Description of the Steepest Descent Algorithm

An algorithm useful for filter design, is the method of steepest descent [3]. The method of steepest descent is an iterative method which minimizes a quadratic function of a suitable error vector between a desired and an approximating function. If we denote the quadratic risk function to be minimized as $R(\underline{u})$, then the rule for steepest descent is to let

$$\underline{u}_{n+1} = \underline{u}_n + \Delta\underline{u}_n \tag{3}$$

where
$$\Delta\underline{u}_n = -k \left.\frac{\partial R(\underline{u})}{\partial \underline{u}}\right|_{\underline{u} = \underline{u}_n}$$

which is usually abreviated,

$$\Delta\underline{u}_n = -k \nabla_{\underline{u}} R(\underline{u}_n).$$

In the above expression, $\underline{u}$ is a vector containing the parameters to be fixed. In our case these parameters are the poles and zeros of an approximating function $\hat{H}(\omega,\underline{u})$. k is a constant multiplier which must be optimized, by trial and error, to stimulate the steepest descent algorithm to converge to the desired parameter values as quickly as possible. $\underline{u}_n$ is the current parameter vector. $\Delta\underline{u}_n$ is the change in values, calculated during the current iteration, of the parameter vector. This change is then added to the current vector to provide the set of parameter values, $\underline{u}_{n+1}$, used during the next iteration, which are checked for convergence. The number of iterations, n, is the smallest number possible, for a given k, to achieve convergence. ∇ is the gradient operator.

Matching Power Responses

In analyzing this problem, two approaches were utilized to determine the appropriate approximating function parameters. The first consists of matching the power response of the desired function to the power response of an approximating function, using the method of steepest descent.

For this case, the desired function is an equivalent low-pass representation, of the filter in (1), given as

$$H(\omega) = \frac{\sin \omega T/2}{\omega T/2}$$

with power response,

$$|H(\omega)|^2 = \frac{\sin^2 \omega T/2}{(\omega T/2)^2} \tag{4}$$

which is a function of frequency. Evaluating the function in (4), at each of the frequencies of a vector $\underline{\omega}$, leads to a sequence of values which we denote as the vector $\underline{H}(\underline{\omega})$.

Recalling that the function in (4) goes to zero at integer multiples of the bit rate, 1/T, it would be logical to choose as an approximating function one that has zeros as well as poles. It was arbitrarily decided to approximate the desired function by a realizable function with one real pole, a pair of complex conjugate poles and a pair of imaginary axis zeros. It should be noted that this approach matches only the magnitudes of the power responses without consideration of the phase characteristics.

The approximating function chosen is as follows

$$\hat{H}(s,\underline{u}) = \frac{(s^2+\omega_o^2)(\alpha^2+\beta^2)p_1}{(s+p_1)(s+\alpha+j\beta)(s+\alpha-j\beta)\omega_o^2} \tag{5}$$

or in the frequency domain,

$$\hat{H}(\omega,\underline{u}) = \frac{(j\omega+j\omega_o)(j\omega-j\omega_o)(\alpha^2+\beta^2)p_1}{(j\omega+\alpha-j\beta)(j\omega+\alpha+j\beta)(j\omega+p_1)\omega_o^2}$$

with power response

$$|\hat{H}(\omega,\underline{u})|^2 = \frac{(\omega_o^2-\omega^2)^2(\alpha^2+\beta^2)^2\, p_1^2}{(\omega^2-2\omega\beta+\alpha^2+\beta^2)(\omega^2+2\omega\beta+\alpha^2+\beta^2)(\omega^2+p_1^2)\omega_o^4} \tag{6}$$

which is a function of the parameter vector $\underline{u}$. Evaluating (6), at each of the frequencies of a vector $\underline{\omega}$, leads to a sequence of values which we denote as the vector $\hat{\underline{H}}\ (\underline{\omega},\underline{u})$.

The added factor, $[(\alpha^2+\beta^2)p_1]/\omega_o^2$, is used to normalize the approximating power response to a value of one at zero frequency. This forces the amplitudes of the two functions close enough together at the start, that the method of steepest descent algorithm will converge to the desired parameter values and not erroneous values.

An appropriate measure of the error, in the approximation of the ideal filter response, is the real vector

$$\underline{e} = |\underline{H}(\underline{\omega})|^2 - |\hat{\underline{H}}(\underline{\omega},\underline{u})|^2 \tag{7}$$

which in accordance with our choice above to work with the power responses, is the error between the power responses. In the above expression $\underline{\omega}^T = (\omega_1, \omega_2, \ldots, \omega_m)$ is a set of selected frequencies at which it is desired that the approximating response match the desired function response. The vector $\underline{u}^T = (\alpha, \beta, p_1, \omega_o)$ is the parameter vector containing the values of the poles and zeros of the approximating function.

As was described earlier, a suitable risk function is the magnitude squared of the error vector, i.e.

$$\begin{aligned} R(\underline{u}) &= \underline{e}^T\ \underline{e} = |\underline{e}|^2 \\ &= [|\underline{H}(\underline{\omega})^2| - |\hat{H}(\underline{\omega},\underline{u})|^2]^T[|\underline{H}(\underline{\omega})^2| - |\hat{\underline{H}}(\underline{\omega},\underline{u})|^2]. \end{aligned} \tag{8}$$

Recalling the steepest descent rule, we now need to find

$$\Delta\underline{u}_n = -k\ \nabla_{\underline{u}}\ R(\underline{u}_n).$$

Noting that the error vector $\underline{e}$ is real, and considering

$$\begin{aligned} \Delta\underline{u}\ R(\underline{u}) &= \nabla_{\underline{u}}\ \underline{e}^T\ \underline{e} = 2(\nabla_{\underline{u}}\ \underline{e}^T)\underline{e} \\ &= 2\{\nabla_{\underline{u}}[|\underline{H}(\underline{\omega})^2| - |\underline{H}(\underline{\omega},\underline{u})|^2]\}^T[|\underline{H}(\underline{\omega})|^2 - |\hat{\underline{H}}(\underline{\omega},\underline{u})|^2] \\ &= -2\{\nabla_{\underline{u}}|\underline{H}^T(\underline{\omega},\underline{u})|^2\}\ [|\underline{H}(\underline{\omega})^2| - |\hat{\underline{H}}(\underline{\omega},\underline{u})|^2] \end{aligned} \tag{9}$$

we have at $\underline{u} = \underline{u}_n$, the nth iteration of the algorithm, that

$$\Delta \underline{u}_n = k[\nabla_{\underline{u}} |\underline{\hat{H}}^T(\underline{\omega},\underline{u}_n)|^2]\underline{e}_n.$$

Substituting this result, for $\Delta \underline{u}_n$, into the rule for steepest descent, we have

$$\underline{u}_{n+1} = \underline{u}_n + k\,[\nabla_{\underline{u}}\ |\underline{\hat{H}}^T(\underline{\omega},\underline{u}_n)|^2]\ \underline{e}_n.$$

Carrying out the gradient operation implied above leads to, for our case

$$\nabla_{\underline{u}}|\underline{\hat{H}}^T(\underline{\omega},\underline{u})|^2 = \begin{bmatrix} \frac{\partial|\hat{H}(\omega_1,\underline{u})|^2}{\partial\alpha} & \frac{\partial|\hat{H}(\omega_2,\underline{u})|^2}{\partial\alpha} \cdots & \frac{\partial|\hat{H}(\omega_m,\underline{u})|^2}{\partial\alpha} \\ \frac{\partial|\hat{H}(\omega_1,\underline{u})|^2}{\partial\beta} & \frac{\partial|\hat{H}(\omega_2,\underline{u})|^2}{\partial\beta} \cdots & \frac{\partial|\hat{H}(\omega_m,\underline{u})|^2}{\partial\beta} \\ \frac{\partial|\hat{H}(\omega_1,\underline{u})|^2}{\partial p_1} & \frac{\partial|\hat{H}(\omega_2,\underline{u})|^2}{\partial p_1} \cdots & \frac{\partial|\hat{H}(\omega_m,\underline{u})|^2}{\partial p_1} \\ \frac{\partial|\hat{H}(\omega_1,\underline{u})|^2}{\partial\omega_o} & \frac{\partial|\hat{H}(\omega_2,\underline{u})|^2}{\partial\omega_o} \cdots & \frac{\partial|\hat{H}(\omega_m,\underline{u})|^2}{\partial\omega_o} \end{bmatrix} \tag{10}$$

where m is the number of frequencies at which it is desired to match the approximating and desired power responses. α is the real part of the complex pole pair and β is the imaginary part. p_1 is a real pole and ω_o is an imaginary axis zero. Together α, β, p_1 and ω_o make up the parameter vector $\underline{u}$ of the approximating fuction.

We now consider finding the derivatives of the approximating function, $|\underline{\hat{H}}^T(\underline{\omega},\underline{u})|^2$ shown in (10), with respect to each element of the vector $\underline{u}$. Recalling from (6) that

$$|\hat{H}(\underline{\omega},\underline{u})|^2 = \frac{(\omega_o{}^2-\omega^2)^2\ (\alpha^2+\beta^2)^2 p_1{}^2}{(\omega^2-2\omega\beta+\alpha^2+\beta^2)(\omega^2+2\omega\beta+\alpha^2+\beta^2)(\omega^2+p_1{}^2)\omega_o{}^4}$$

we determine the derivatives with respect to p_1, ω_o, α, and β.

These derivatives are summarized in Table 1.

We now have everything needed to implement the steepest descent algorithm. The implementation is accomplished by a computer program. The program first accepts values for the frequencies at which it is desired to match the power response of the approximating and desired filter functions. It next accepts initial values for the poles and zeros. Using this initial information, the derivatives of the approximating function are calculated. The balance of the program is a recursive section in which the error vector between the two functions is calculated, the risk function is evaluated, and $\Delta \underline{u}_n$ is calculated and added to $\underline{u}_n$, to obtain the next set of values for the poles and zeros of the approximating function. This current set of values for the poles and zeros of the approximating function is then checked for convergence. If the algorithm has not converged, the approximating function derivatives are recalculated, using current prarmeter values, and another iteration of the recursive section is made. If the algorithm has converged, the optimum parameter values are output. The program also stores the values, for each iteration, of the desired and approximating filters power response as well as the values of the risk function. A plot routine may be executed to obtain plots of these various functions.

The program listing is shown in the Appendix.

Matching Transfer Functions

The second approach, used to arrive at the optimum values for the parameters of the approximating function, consists of matching the transfer functions of the desired and approximating filter functions. Again, the method of steepest descent is utilized. In this case, however, the desired function is given by,

TABLE 1

DERIVATIVES OF APPROXIMATING FUNCTION WITH RESPECT TO PARAMETERS

$$\frac{\partial}{\partial p_1}\left|\underline{\hat{H}}(\underline{\omega},\underline{u})\right|^2 = \frac{2p_1(\omega_o^2-\omega^2)^2(\alpha^2+\beta^2)^2\omega^2}{(\omega^2-2\omega\beta+\alpha^2+\beta^2)(\omega^2+2\omega\beta+\alpha^2+\beta^2)(\omega^2+p_1^2)^2\,\omega_o^4} \tag{11}$$

$$\frac{\partial}{\partial \alpha}\left|\underline{\hat{H}}(\underline{\omega},\underline{u})\right|^2 = \frac{2p_1^2(\alpha^2+\beta^2)(\omega_o^2-\omega^2)^2\left[2\alpha(\omega^2-2\omega\beta+\alpha^2+\beta^2)(\omega^2+2\omega\beta+\alpha^2+\beta^2)-(\alpha^2+\beta^2)(2\omega^2\alpha+2\alpha\beta^2+2\alpha^3)\right]}{(\omega^2-2\omega\beta+\alpha^2+\beta^2)^2\,(\omega^2+2\omega\beta+\alpha^2+\beta^2)^2\,(\omega^2+p_1^2)\omega_o^4} \tag{12}$$

$$\frac{\partial}{\partial \beta}\left|\underline{\hat{H}}(\underline{\omega},\underline{u})\right|^2 = \frac{2p_1^2(\alpha^2+\beta^2)(\omega_o^2-\omega^2)^2\left[2\beta(\omega^2-2\omega\beta+\alpha^2+\beta^2)(\omega^2+2\omega\beta+\alpha^2+\beta^2)-(\alpha^2+\beta^2)(-2\omega^2\beta+2\alpha^2\beta+2\beta^3)\right]}{(\omega^2-2\omega\beta+\alpha^2+\beta^2)^2(\omega^2+2\omega\beta+\alpha^2+\beta^2)^2(\omega^2+p_1^2)\omega_o^4} \tag{13}$$

$$\frac{\partial}{\partial \omega_o}\left|\underline{\hat{H}}(\underline{\omega},\underline{u})\right|^2 = \frac{4p_1^2(\omega_o^2-\omega^2)(\alpha^2+\beta^2)^2\,\omega^2}{(\omega^2-2\omega\beta+\alpha^2+\beta^2)(\omega^2+2\omega\beta+\alpha^2+\beta^2)(\omega^2+p_1^2)\omega_o^5} \tag{14}$$

$$H(\omega) = -j \frac{\sin \frac{\omega T}{2} e^{-j\frac{\omega T}{2}}}{\frac{\omega T}{2}} \tag{15}$$

which is a complex number that may also be written as

$$H(\omega) = -j \frac{\sin\frac{\omega T}{2}}{\frac{\omega T}{2}} [\cos \frac{\omega T}{2} -j \sin \frac{\omega T}{2}]$$

Expanding we obtain

$$H(\omega) = - \frac{\sin\frac{\omega T}{2}}{\frac{\omega T}{2}} -j \frac{(\sin\frac{\omega T}{2})(\cos\frac{\omega T}{2})}{\frac{\omega T}{2}} \tag{16}$$

which is a function of frequency. Evaluating the function in (16), at each of the frequencies of a vector $\underline{\omega}$, leads to a sequence of values which we refer to as the vector $\underline{H}(\underline{\omega})$. It should be observed that, in this case, in addition to matching the amplitude responses of the approximating and desired functions, we are also matching the phases.

Since the more complex a filter is, the harder it is to build, it was decided to use a simpler approximating filter in this case. The filter chosen was an all pole filter with two complex pole pairs, whose function is written as

$$\hat{H}(s,\underline{u}) = \frac{-j\ (\alpha_1^2+\beta_1^2)(\alpha_2^2+\beta_2^2)}{(s+\alpha_1+j\beta_1)(s+\alpha_1-j\beta_1)(s+\alpha_2+j\beta_2)(s+\alpha_2-j\beta_2)} \tag{17}$$

or in the frequency domain,

$$\hat{H}(\omega,\underline{u}) = \frac{-j\ (\alpha_1^2+\beta_1^2)(\alpha_2^2+\beta_2^2)}{(s+\alpha_1+j\beta_1)(s+\alpha_1-j\beta_1)(s+\alpha_2+j\beta_2)(s+\alpha_2-j\beta_2)}$$

$$= \frac{-j\ (\alpha_1^2+\beta_1^2)(\alpha_2^2+\beta_2^2)}{(-\omega^2+2j\omega\alpha_1+\alpha_1^2+\beta_1^2)(-\omega^2+2j\omega\alpha_2+\alpha_2^2+\beta_2^2)} \tag{18}$$

which is a function of a parameter vector $\underline{u}$. Evaluating (18), at each of the frequencies of a vector $\underline{\omega}$, leads to a sequence of values which we refer to as the vector $\hat{\underline{H}}(\underline{\omega},\underline{u})$. The added factor $(\alpha_1^2+\beta_1^2)(\alpha_2^2+\beta_2^2)$ as before, is used to normalize approximating transfer function to a value of one at zero frequency, thus facilitating convergence of the steepest descent algorithm.

An appropriate measure of the error, in the approximation of the ideal filter response, is the complex vector

$$\underline{e} = [\underline{H}_r(\underline{\omega})+j\underline{H}_i(\underline{\omega})]-[\hat{\underline{H}}_r(\underline{\omega},\underline{u})+j\hat{\underline{H}}_i(\underline{\omega},\underline{u})] \tag{19}$$

which in accordance with the above choice to match transfer functions is the error between the transfer functions of the desired and approximating functions, respectively. In (19) $\underline{\omega}^T = (\omega_1,\omega_2,\ldots,\omega_m)$ is again a set of selected frequencies at which it is desired that the approximating response match the desired response. The vector $\underline{u}^T = (\alpha_1,\beta_1,\alpha_2,\beta_2)$ is the parameter vector containing the values of the poles of the approximating functions. $\underline{H}_r(\underline{\omega})$ and $\underline{H}_i(\underline{\omega})$ are the real and imaginary parts of the desired function and $\hat{\underline{H}}_r(\underline{\omega},\underline{u})$ and $\hat{\underline{H}}_i(\underline{\omega},\underline{u})$ are the real and imaginary parts of the approximating function.

A suitable risk function may again be written as the magnitude squared of the error vector, i.e.,

$$R(\underline{u}) = \underline{e}^T\underline{e}^* = (\underline{e}_r+j\underline{e}_i)^T(\underline{e}_r+j\underline{e}_i)^* \tag{20}$$

where $\underline{e}_r = [\underline{H}_r(\underline{\omega})-\hat{\underline{H}}_r(\underline{\omega},\underline{u})]$ and $\underline{e}_i = [\underline{H}_i(\underline{\omega})-\hat{\underline{H}}_i(\underline{\omega},\underline{u})]$ are the real and imaginary parts of $\underline{e}$.

To use the steepest descent rule, we now need to find

$$\Delta\underline{u}_n = -k\,\nabla_{\underline{u}}R(\underline{u}_n).$$

Noting that the error vector $\underline{e}$ is a complex vector, and considering

$$\nabla_{\underline{u}} R(\underline{u}) = \nabla_{\underline{u}} \underline{e}^T \underline{e}^* = (\nabla_{\underline{u}} \underline{e}^T)\underline{e}^*$$

$$= (\nabla_{\underline{u}} \underline{e}_r + j\nabla_{\underline{u}} \underline{e}_i)^T (\underline{e}_r + j\underline{e}_i)^*$$

$$= [(\nabla_{\underline{u}} \underline{e}_r^T)\underline{e}_r^* - (\nabla_{\underline{u}} \underline{e}_i^T)\underline{e}_i^*] + j[(\nabla_{\underline{u}} \underline{e}_i^T)\underline{e}_r^* + (\nabla_{\underline{u}} \underline{e}_r^T)\underline{e}_i^*]$$

$$= 2(\nabla_{\underline{u}} \underline{e}_r^T)\underline{e}_r^* + 2(\nabla_{\underline{u}} \underline{e}_i^T)\ \underline{e}_i^*$$

$$= 2(\nabla_{\underline{u}} \underline{e}_r^T)\underline{e}_r + 2(\nabla_{\underline{u}} \underline{e}_i^T)\underline{e}_i$$

from which observing (20) we have

$$\nabla_{\underline{u}} R(\underline{u}) = -2\nabla_{\underline{u}} \underline{\hat{H}}_r(\underline{\omega},\underline{u})\underline{e}_r - 2\nabla_{\underline{u}} \underline{\hat{H}}_i(\underline{\omega},\underline{u})\underline{e}_i. \tag{21}$$

At $\underline{u} = \underline{u}_n$, the n^{th} iteration of the algorithm, we now have

$$\nabla\underline{u}_n = k\ [\nabla_{\underline{u}} \underline{\hat{H}}_r(\underline{\omega},\underline{u})\underline{e}_r + \nabla_{\underline{u}} \underline{\hat{H}}_i(\underline{\omega},\underline{u})\underline{e}_i]$$

Substituting the result for $\Delta\underline{u}_n$, into the rule for steepest descent, we obtain

$$\underline{u}_{n+1} = \underline{u}_n + k[\nabla_{\underline{u}} \underline{\hat{H}}_r(\underline{\omega},\underline{u})\underline{e}_r + \nabla_{\underline{u}} \underline{\hat{H}}_i(\underline{\omega},\underline{u})\underline{e}_i]$$

From the formula for the gradient of a function with respect to a real vector we have

$$\nabla_{\underline{u}} \underline{\hat{H}}(\underline{\omega},\underline{u}) = \begin{bmatrix} \frac{\partial\hat{H}(\omega_1,\underline{u})}{\partial\alpha_i} & \frac{\partial\hat{H}(\omega_2,\underline{u})}{\partial\alpha_1} & \cdots & \frac{\partial\hat{H}(\omega_m,\underline{u})}{\partial\alpha_1} \\ \frac{\partial\hat{H}(\omega_1,\underline{u})}{\partial\alpha_2} & \frac{\partial\hat{H}(\omega_2,\underline{u})}{\partial\alpha_2} & \cdots & \frac{\partial\hat{H}(\omega_m,\underline{u})}{\partial\alpha_2} \\ \frac{\partial\hat{H}(\omega_1,\underline{u})}{\partial\beta_1} & \frac{\partial\hat{H}(\omega_2,\underline{u})}{\partial\beta_1} & \cdots & \frac{\partial\hat{H}(\omega_m,\underline{u})}{\partial\beta_1} \\ \frac{\partial\hat{H}(\omega_1,\underline{u})}{\partial\beta_2} & \frac{\partial\hat{H}(\omega_2,\underline{u})}{\partial\beta_2} & \cdots & \frac{\partial\hat{H}(\omega_m,\underline{u})}{\partial\beta_2} \end{bmatrix} \tag{22}$$

where m is the number of frequencies at which it is desired to match the approximating and desired transfer functions. α_1 is the real part of the first complex pair and β_1 is the imaginary part of the first complex pair. Similarily, α_2 is the real part of the second complex pair and β_2 is the imaginary part of the second complex pair. Together $\alpha_1, \beta_1, \alpha_2, \beta_2$ make up the parameter vector $\underline{u}$ of the approximating function.

We now consider finding the derivatives of the approximating function $\hat{\underline{H}}^T(\underline{\omega},\underline{u})$, shown in (22), with respect to each element of the vector $\underline{u}$. Recalling from (18) that

$$\hat{\underline{H}}(\underline{\omega},\underline{u}) = \frac{-j(\alpha_1^2+\beta_1^2)(\alpha_2^2+\beta_2^2)}{(-\omega^2+2j\omega\alpha_1+\alpha_1^2+\beta_1^2)(-\omega^2+2j\omega\alpha_2+\alpha_2^2+\beta_2^2)}$$

we obtain for the derivative with respect to α_1

$$\frac{\partial \underline{H}(\underline{\omega},\underline{u})}{\partial \alpha_1} = \frac{-j2[(-\omega^2+2j\omega\alpha_1+\alpha_1^2+\beta_1^2)(\alpha_2^2+\beta_2^2)\alpha_1-(\alpha_1^2+\beta_1^2)(\alpha_2^2+\beta_2^2)(j\omega+\alpha_1)]}{(-\omega^2+2j\omega\alpha_1+\alpha_1^2+\beta_1^2)^2(-\omega^2+2j\omega\alpha_2+\alpha_2^2+\beta_2^2)}$$

$$= \frac{2[(\omega\alpha_1^2\alpha_2^2+\omega\alpha_1^2\beta_2^2-\omega\alpha_2^2\beta_1^2-\omega\beta_1^2\beta_2^2)+j(\omega^2\alpha_1\alpha_2^2+\omega^2\alpha_1\beta_2^2)]}{\text{DENOM}} \qquad (23)$$

DENOM = DENOMR + jDENOMI which is made up of the real and imaginary parts of the denominator of (23) where,

$$\text{DENOMR} = (-\omega^6+6\omega^4\alpha_1^2+2\omega^4\beta_1^2+2\omega^2\alpha_1^2\beta_1^2-\omega^2\alpha_1^4-\omega^2\beta_1^4+8\omega^4\alpha_1\alpha_2-8\omega^2\alpha_1^3\alpha_2-$$

$$8\omega^2\alpha_1\alpha_2\beta_1^2+\omega^4\alpha_2^2-6\omega^2\alpha_1^2\alpha_2^2-2\omega^2\alpha_2^2\beta_1^2+2\alpha_1^2\alpha_2^2\beta_1^2+\alpha_1^4\alpha_2^2+\alpha_2^2\beta_1^4+\omega^4\beta_2^2-$$

$$6\omega^2\alpha_1^2\beta_2^2-2\omega^2\beta_1^2\beta_2^2+2\alpha_1^2\beta_1^2\beta_2^2+\alpha_1^4\beta_2^2+\beta_1^4\beta_2^2)$$

is the real part and the imaginary part is given as

$$\mathrm{DENOMI} = (4\omega^5\alpha_1 - 4\omega^3\alpha_1^3 - 4\omega^3\alpha_1\beta_1^2 + 2\omega^5\alpha_2 - 12\omega^3\alpha_1^2\alpha_2 - 4\omega^3\alpha_2\beta_1^2 + 4\omega\alpha_1^2\alpha_2\beta_1^2 +$$

$$2\omega\alpha_1^4\alpha_2 + 2\omega\alpha_2\beta_1^4 - 4\omega^3\alpha_1\alpha_2^2 + 4\omega\alpha_1^3\alpha_2^2 + 4\omega\alpha_1\alpha_2^2\beta_1^2 - 4\omega^3\alpha_1\beta_2^2 + 4\omega\alpha_1^3\beta_2^2 + 4\omega\alpha_1\beta_1^2\beta_2^2)$$

Letting $\mathrm{NUM1R} = (\omega\alpha_1^2\alpha_2^2 + \omega\alpha_1^2\beta_2^2 - \omega\alpha_2^2\beta_1^2 - \omega\beta_1^2\beta_2^2)$ and $\mathrm{NUM1I} = (\omega^2\alpha_1\alpha_2^2 + \omega^2\alpha_1\beta_2^2)$, we obtain for the derivative that

$$\frac{\partial\underline{\hat{H}}(\underline{\omega},\underline{u})}{\partial\alpha_1} = \frac{2[\mathrm{NUM1R} + j\mathrm{NUM1I}]}{\mathrm{DENOM}}$$

Multiplying numerator and denominator by the conjugate of the denominator to rationalize the derivative results in

$$\frac{\partial\underline{H}(\underline{\omega},\underline{u})}{\partial\alpha_1} = \frac{2[\mathrm{NUM1R} + j\mathrm{NUM1I}][\mathrm{DENOMR} - j\mathrm{DENOMI}]}{[\mathrm{DENOMR} + j\mathrm{DENOMI}][\mathrm{DENOMR} - j\mathrm{DENOMI}]}$$

from which (24) in TABLE 2 is found as the final form of the derivative. In (24), $\mathrm{DENOM1} = (\mathrm{DENOMR})^2 + (\mathrm{DENOMI})^2$.

Similarly, for the derivative with respect to β_1 we find

$$\frac{\partial\underline{\hat{H}}(\underline{\omega},\underline{u})}{\partial\beta_1} = \frac{-j2[-\omega^2+2j\omega\alpha_1+\alpha_1^2+\beta_1^2)(\alpha_2^2+\beta_2^2)\beta_1 - (\alpha_1^2+\beta_1^2)(\alpha_2^2+\beta_2^2)\beta_1]}{(-\omega^2+2j\omega\alpha_1+\alpha_1^2+\beta_1^2)^2(-\omega^2+2j\omega\alpha^2+\alpha_2^2+\beta_2^2)}$$

$$= \frac{2[(2\omega\alpha_1\alpha_2^2\beta_1 + 2\omega\alpha_1\beta_1^2\beta_2^2) + j(\omega^2\alpha_2^2\beta_1 + \omega^2\beta_1\beta_2^2)]}{\mathrm{DENOM}} \qquad (25)$$

Letting $\mathrm{NUM2R} = (2\omega\alpha_1\alpha_2^3\beta_1 + 2\omega\alpha_1\beta_1^2\beta_2^2)$ and

$\mathrm{NUM2I} = (\omega^2\alpha_2^2\beta_1 + \omega^2\beta_1\beta_2^2)$, we obtain for the derivative that

$$\frac{\partial \hat{H}(\underline{\omega},\underline{u})}{\partial \beta_1} = \frac{2[\mathrm{NUM2R} + j\mathrm{NUM2I}]}{\mathrm{DENOM}}$$

Rationalizing and simplifying, the derivative is written as (26) in TABLE 2.

The derivatives with respect to α_2 and β_2 are found, using the procedure above, and shown as (27) and (28) respectively, in TABLE 2.

In (27), $\mathrm{NUM3R} = (\omega\alpha_1^2\alpha_2^2+\omega\alpha_2^2\beta_1^2-\omega\alpha_1^2\beta_2^2-\omega\beta_1^2\beta_2^2)$

$\mathrm{NUM3I} = (\omega^2\alpha_1^2\alpha_2+\omega^2\alpha_2\beta_1^2)$

$\mathrm{DENOMR1} = (-\omega^6+6\omega^4\alpha_2^2+2\omega^4\beta_2^2-2\omega^2\alpha_2^2\beta_2^2-\omega^2\alpha_2^4-\omega^2\beta_2^4+8\omega^4\alpha_1\alpha_2-8\omega^2\alpha_1\alpha_2^3-$
$8\omega^2\alpha_1\alpha_2\beta_2^2+\omega^4\alpha_1^2-6\omega^2\alpha_1^2\alpha_2^2-2\omega^2\alpha_1^2\beta_2^2+2\alpha_1^2\alpha_2^2\beta_2^2+\alpha_1^2\alpha_2^4+\alpha_1^2\beta_2^4+\omega^4\beta_1^2-$
$6\omega^2\alpha_2^2\beta_1^2-2\omega^2\beta_1^2\beta_2^2+2\alpha_2^2\beta_1^2\beta_2^2+\alpha_2^4\beta_1^2+\beta_1^2\beta_2^4)$

$\mathrm{DENOMI1} = (4\omega^5\alpha_2-4\omega^3\alpha_2^3-4\omega^3\alpha_2\beta_2^2+2\omega^5\alpha_1-12\omega^3\alpha_1\alpha_2^2-4\omega^3\alpha_1\beta_2^2+4\omega\alpha_1\alpha_2^2\beta_2^2+$
$2\omega\alpha_1\alpha_2^4+2\omega\alpha_1\beta_2^4-4\omega^3\alpha_1^2\alpha_2+4\omega\alpha_1^2\alpha_2^3+4\omega\alpha_1^2\alpha_2\beta_2^2-4\omega^3\alpha_2\beta_1^2+4\omega\alpha_2^3\beta_1^2+4\omega\alpha_2\beta_1^2\beta_2^2)$

$\mathrm{DENOM2} = (\mathrm{DENOMR1})^2 + (\mathrm{DENOMI1})^2$

In (28), $\mathrm{NUM4R} = (2\omega\alpha_1^2\alpha_2\beta_2+2\omega\alpha_2\beta_1^2\beta_2^2)$

$\mathrm{NUM4I} = (\omega^2\alpha_1^2\beta_2+\omega^2\beta_1^2\beta_2)$

One step remains to be done before we have everything needed to implement the algorithm. The approximating function needs to be rationalized into its real and imaginary parts, as was done with the derivatives, to facilitate the analytical analysis. Recalling the equation for the approximating function from (18),

TABLE 2

DERIVATIVES OF APPROXIMATING FUNCTION WITH RESPECT TO PARAMETERS

$$\frac{\partial \underline{\hat{H}}(\underline{\omega},\underline{u})}{\partial \alpha_1} = \frac{2\{[(\mathrm{NUM1R})(\mathrm{DENOMR})+(\mathrm{NUM1I})(\mathrm{DENOMI})]+j[(\mathrm{DENOMR})(\mathrm{NUM1I})-(\mathrm{DENOMI})(\mathrm{NUM1R})]\}}{\mathrm{DENOM1}} \quad (24)$$

$$\frac{\partial \underline{\hat{H}}(\underline{\omega},\underline{u})}{\partial \beta_1} = \frac{2\{[(\mathrm{NUM2R})(\mathrm{DENOMR})+(\mathrm{NUM2I})(\mathrm{DENOMI})]+j[(\mathrm{NUM2I})(\mathrm{DENOMR})-(\mathrm{NUM2R})(\mathrm{DENOMI})]\}}{\mathrm{DENOM1}} \quad (26)$$

$$\frac{\partial \underline{\hat{H}}(\underline{\omega},\underline{u})}{\partial \alpha_2} = \frac{2\{[(\mathrm{NUM3R})(\mathrm{DENOMR1})+(\mathrm{NUM3I})(\mathrm{DENOMI1})]+j[(\mathrm{DENOMR1})(\mathrm{NUM3I})-(\mathrm{DENOMI1})(\mathrm{NUM3R})]\}}{\mathrm{DENOM2}} \quad (27)$$

$$\frac{\partial \underline{\hat{H}}(\underline{\omega},\underline{u})}{\partial \beta_2} = \frac{2\{[(\mathrm{NUM4R})(\mathrm{DENOMR1})+(\mathrm{NUM4I})(\mathrm{DENOMI1})]+j[(\mathrm{NUM4I})(\mathrm{DENOMR1})-(\mathrm{NUM4R})(\mathrm{DENOMI1})]\}}{\mathrm{DENOM2}} \quad (28)$$

$$\hat{\underline{H}}(\underline{\omega},\underline{u}) = \frac{-j(\alpha_1^2+\beta_1^2)(\alpha_2^2+\beta_2^2)}{(-\omega^2+2j\omega\alpha_1+\alpha_1^2+\beta_1^2)(-\omega^2+2j\omega\alpha_2+\alpha_2^2+\beta_2^2)}$$

we obtain after rationalizing

$$\hat{\underline{H}}(\underline{\omega},\underline{u}) = \frac{-[(ANUM)(ADENI) + j(ANUM)(ADENR)]}{DENOM3} \qquad (29)$$

where $ANUM = (\alpha_1^2+\beta_1^2)(\alpha_2^2+\beta_2^2)$

$ADENR = (\omega^4-\omega^2\alpha_2^2-\omega^2\beta_2^2-4\omega^2\alpha_1\alpha_2-\omega^2\alpha_1^4\alpha_2^2+\alpha_1^2\beta_2^2-\omega^2\beta_1^2+\alpha_2^2\beta_1^2+\beta_1^2\beta_2^2)$ is the real part of the denominator and

$ADENI = (-2\omega^3\alpha_2-2\omega^3\alpha_1+2\omega\alpha_1\alpha_2^2+2\omega\alpha_1\beta_2^2+2\omega\alpha_1^2\alpha_2+2\omega\alpha_2\beta_1^2)$ is the imaginary part of the denominator and

$$DENOM3 = (ADENR)^2 + (ADENI)^2$$

The final step of implementing the steepest descent algorithm is again the writing of a computer program. As before, the program accepts values for the frequencies at which it is desired to match the transfer functions of the approximating and desired filter functions. It next accepts initial values for the poles. Using this initial information, the derivatives of the approximating function are calculated. The balance of the program is a recursive section in which the error vector between the two functions is calculated, the risk function is evaluated and $\Delta\underline{u}_n$ is calculated and added to $\underline{u}_n$, to obtain the next set of values for the poles of the approximating function. This current set of values for the poles is then checked for convergence. If the algorithm has not converged, the approximating function derivatives are recalculated, using current pole values, and another iteration of the recursive section is performed. If the algorithm has converged, to a set of pole values, these optimum values are output. In addition, the program stores the values on disk, for each iteration, of the desired and approximating filters transfer function amplitude and phase response, as well as the risk function. A plot routine may be executed to obtain plots of these various functions.

The program listing is shown in the Appendix.

III. RESULTS

Matching Power Responses

Following the procedure set forth above, the following results were obtained on a Data General Nova computer. In the first case, of matching the power responses, it was arbitrarily decided to match the two functions at 17 frequencies evenly spaced between 0 and 4π radians. By trial and error, a good value for the multiplying constant k was found to be 20.0. Using this value of k and 500 iterations of the iterative loop of the first program in the Appendix, the pole and zero values in TABLE 3 were obtained. Various initial values for the poles and zeros were used to verify that the values shown in TABLE 3 are optimum. A plot of the desired and approximating function power responses is found in Figure 2. Figure 3 shows that the risk function, or the error squared between the desired and approximating power responses, is equal to 0.43×10^{-2}, after the last iteration.

To observe how closely the approximating functions impulse response matches that of the desired function, we use the parameter values in (30) and perform an inverse Laplace transformation [4] on the approximating function to obtain

$$\underline{\hat{h}}(t) = 15.86e^{-6.14t} + 12.52e^{-3.66t}\cos(3.431t-3.0) \tag{31}$$

Figure 4 shows this response along with the ideal response.

Matching Transfer Functions

For the case of matching the transfer function of the approximating function to that of the desired, it was decided to match the two

TABLE 3

Pole and Zero Values for the Case of Matching Power Responses

(In Radians)

Real pole, p_1	= 6.14	
Real part of complex pole pair, α	= 3.66	
Imaginary part of complex pole pair, β	= 3.43	(30)
Imaginary axis zero, ω_o	= 6.71	

TABLE 4

Pole Values for the Case of Matching Transfer Functions

(In Radians)

Real part of 1st complex pole pair, α_1	= 2.33	
Imaginary part of 1st complex pole pair, β_1	= 1.91	
Real part of 2nd complex pole pari, α_2	= 50.00	(32)
Imaginary part of 2nd complex pole pair, β_2	= 50.80	

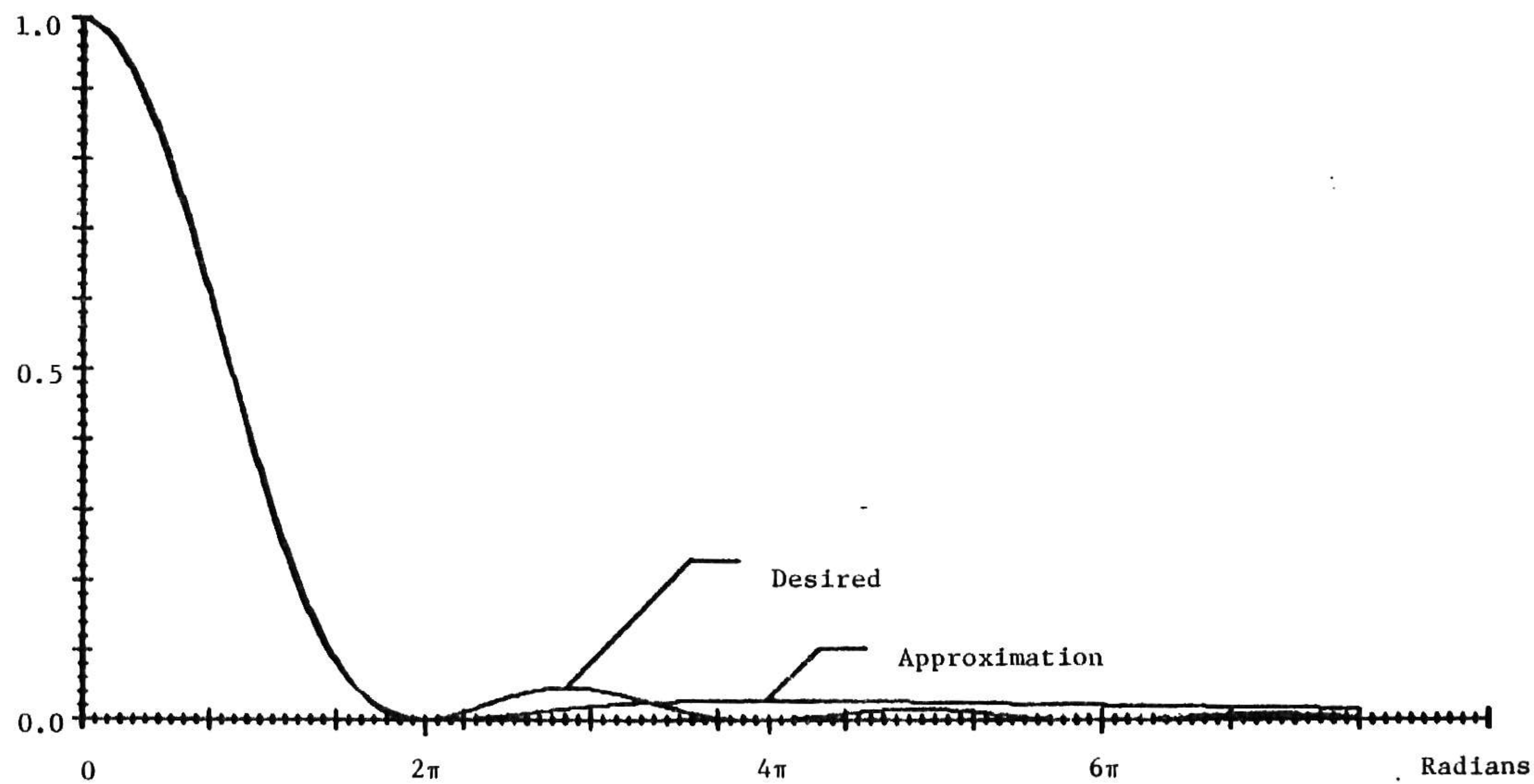

Figure 2 Desired and approximating function power responses

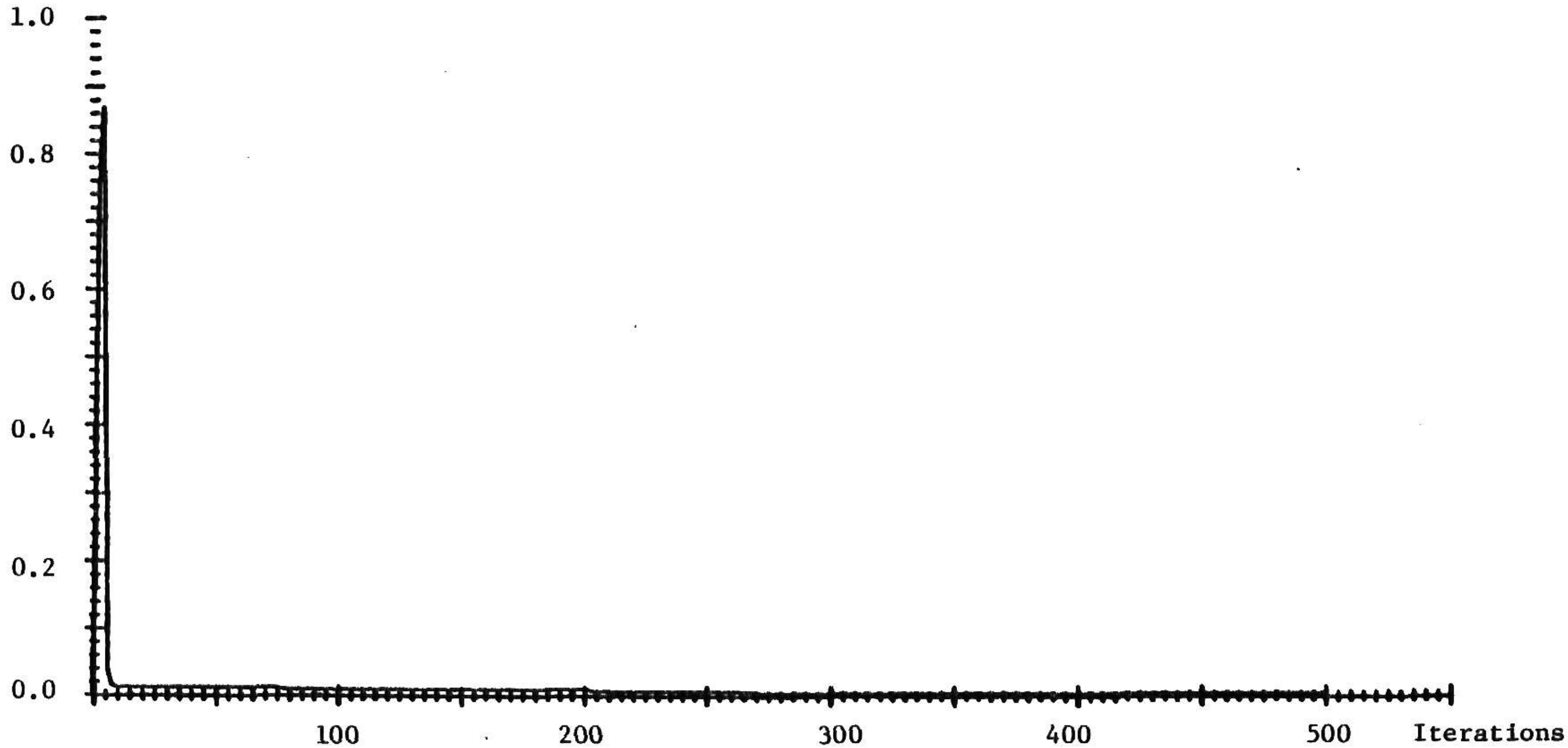

Figure 3 Risk function for matching power response case

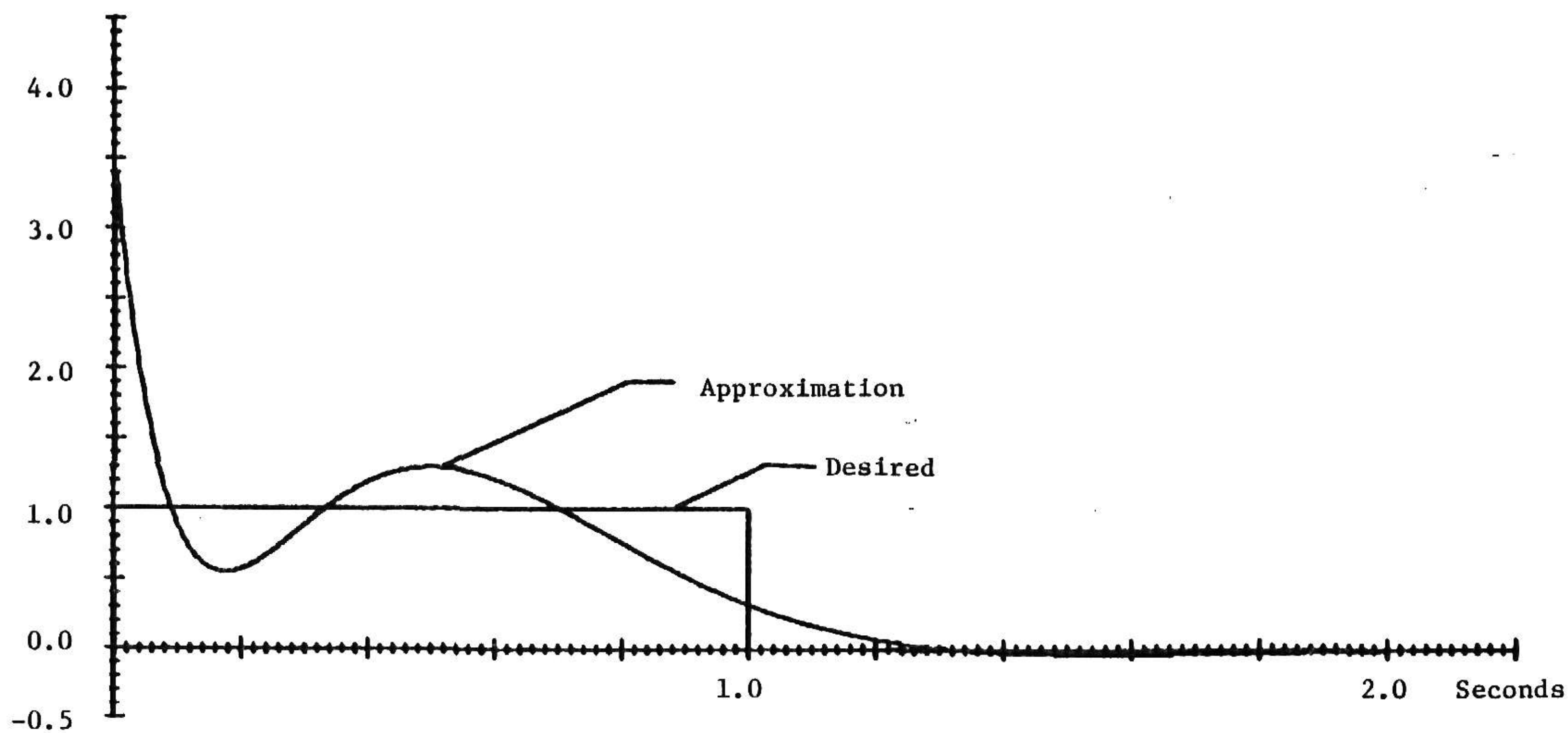

Figure 4 Ideal and approximating impulse responses

functions at nine frequencies evenly spaced between 0 and 2π radians. This decision was made because the largest portion of the energy in the signal is contained in the area between zero and the first null of the amplitude spectrum, which is at 2π radians. Again trail and error was used to arrive at 2.0 for a good value of k. Using this value for k, and 200 iterations of the iterative loop of the second program in the Appendix, the following pole values, shown in TABLE 4, were found. Various initial values for the poles were used to varify that the values shown in TABLE 4 are optimum. Plots of the desired and approximating functions amplitude responses are shown in Figure 5. Figure 6 shows the phase responses of the two functions. From Figure 7, the risk function, which is the error squared between the desired and approximating functions amplitude and phase responses, has a value of 0.098, after the last iteration.

To observe how closely the impulse response of the approximating function matches that of the desired function, we use the parameter values in (32) and perform an inverse Laplace transformation on the approximating function to obtain

$$\hat{h}(t) = 4.98e^{-2.33t}\cos(1.91t - 1.608) + .186e^{-50.0t}\cos(50.8t + .063) \tag{33}$$

Figure 8 shows this response along with the ideal response, h(t).

It is observed from Figure 2 through 4 above, that even though the power response of the approximating function matches that of the desired function exceptionally well, the impulse responses vary some what. It remains to be seen how this variance will affect the performance of the filter in generating an MSK signal.

From Figures 5 through 8, we observe that even though the approximation of the transfer function is not as close as the approximation of the power response, the impulse response is closer to the ideal response.

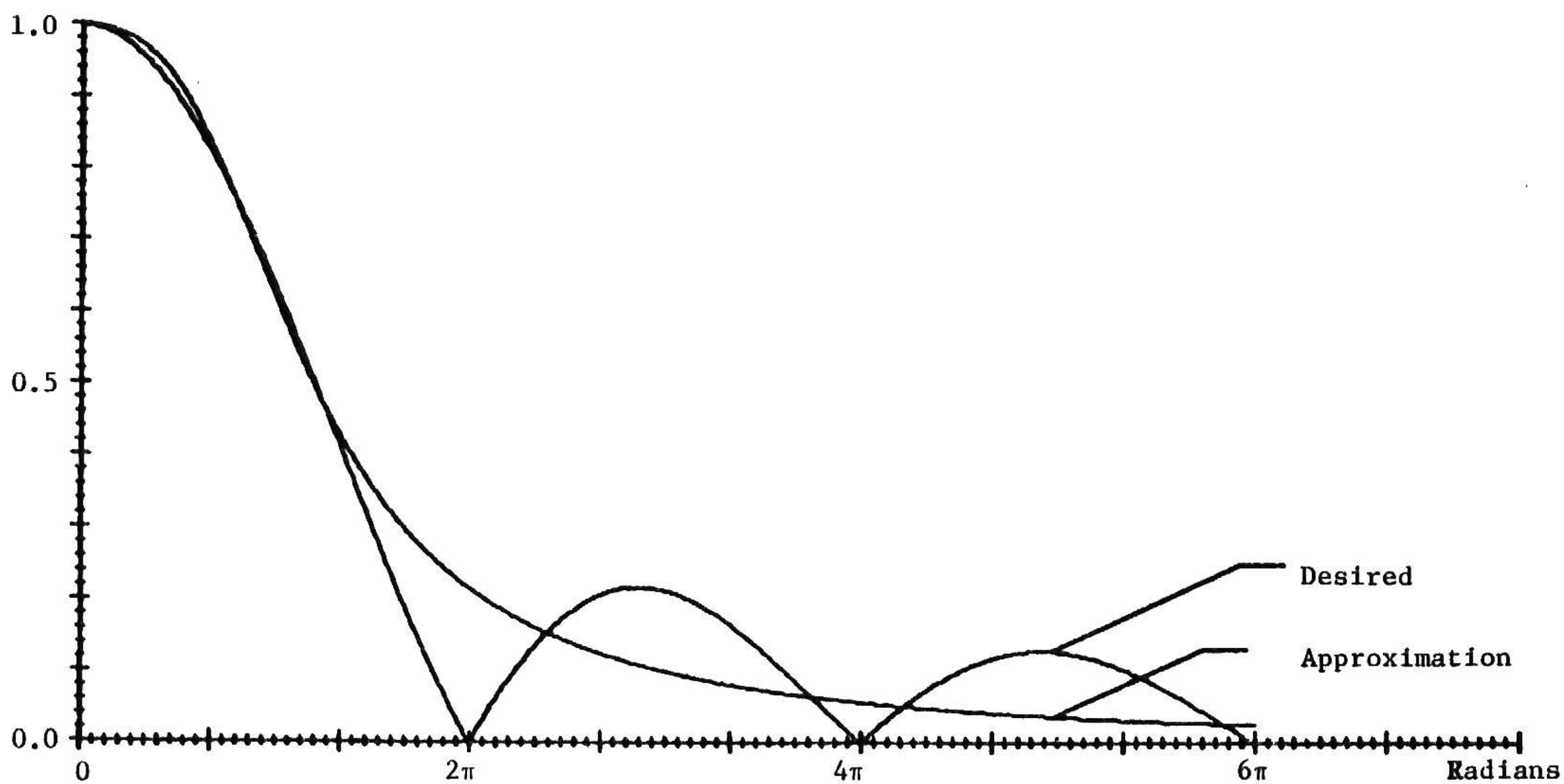

Figure 5 Amplitude response for matching transfer function case

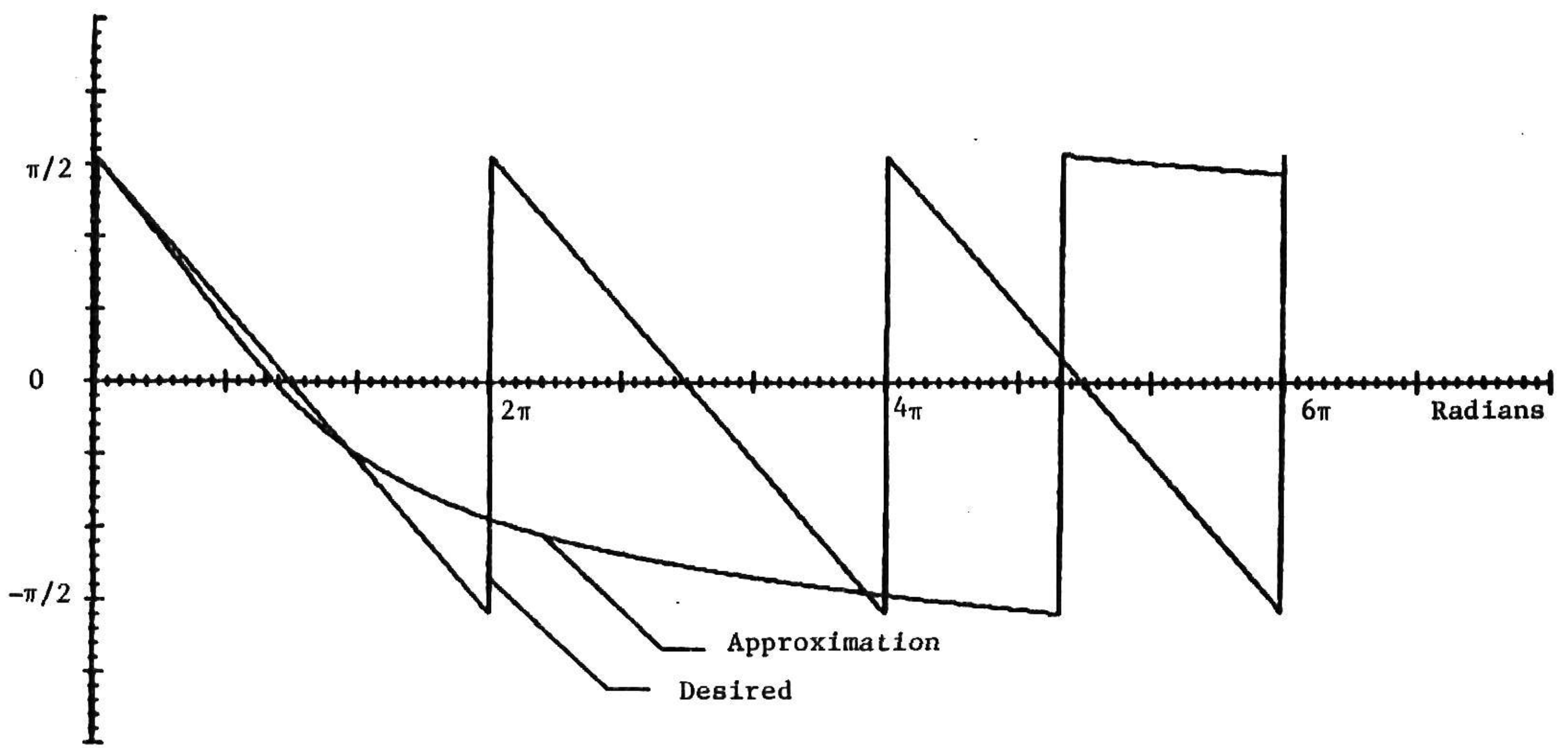

Figure 6 Phase response for matching transfer function case

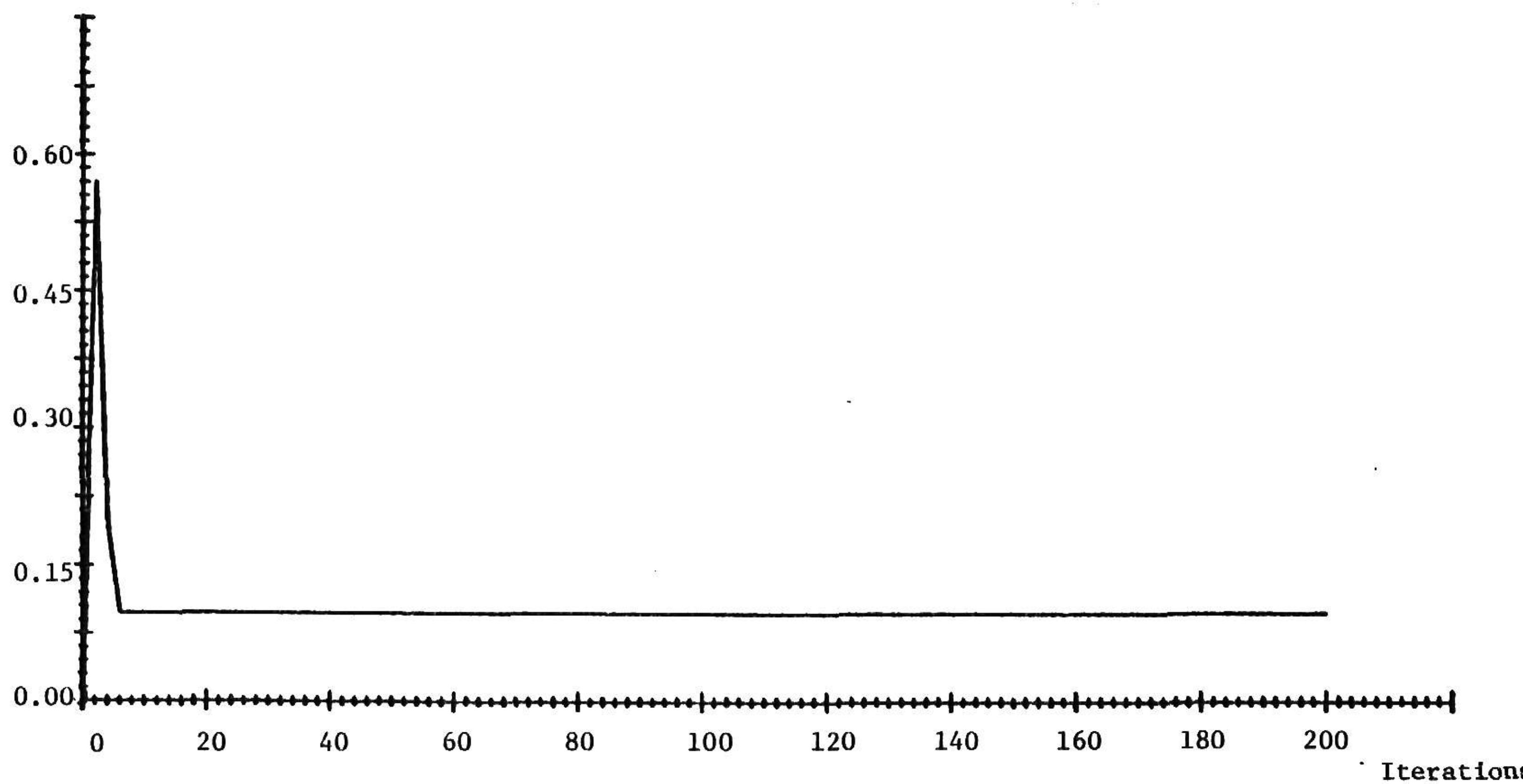

Figure 7 Risk function for matching transfer function case

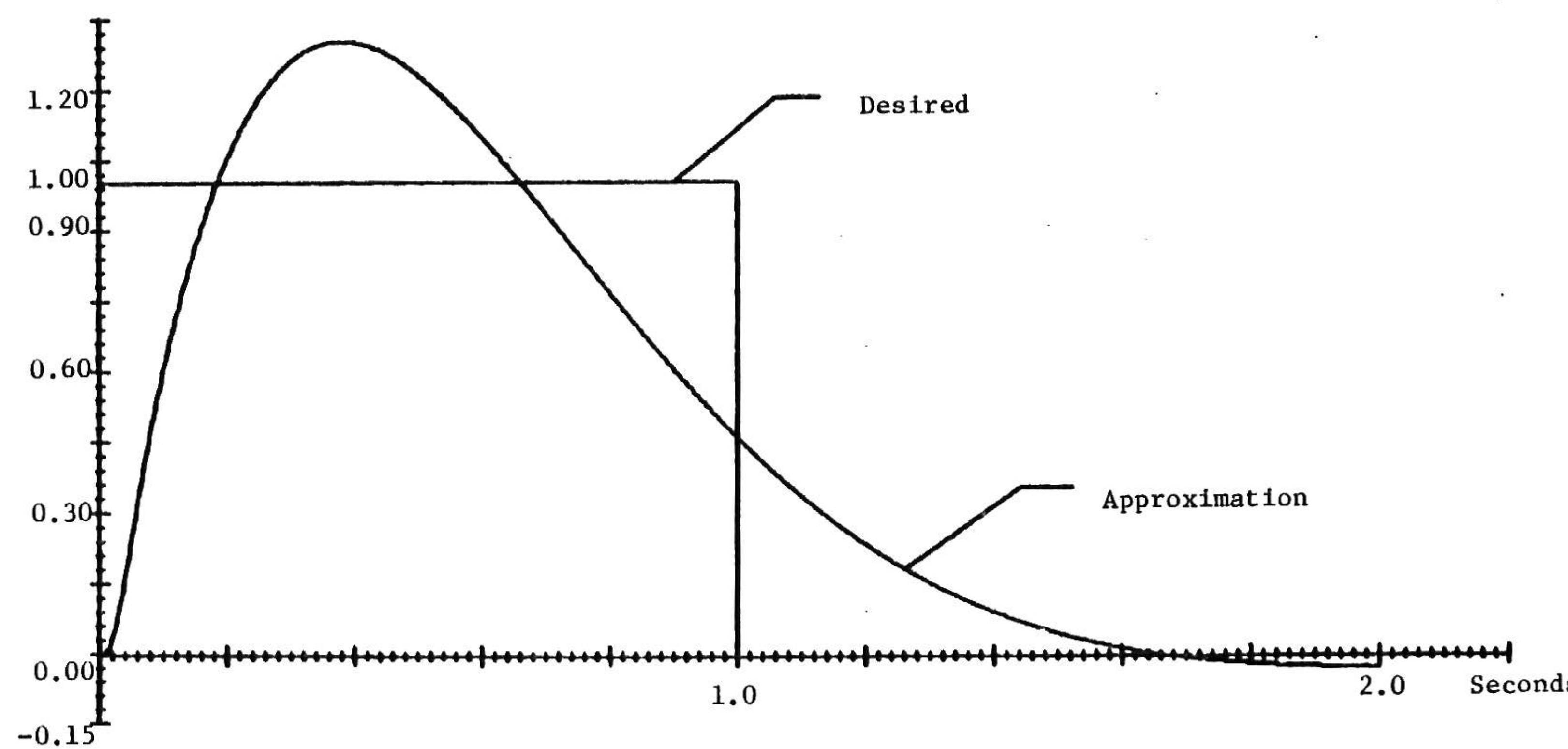

Figure 8 Ideal and approximating impulse responses

This could be accounted for by the fact that two filter types, each having a different characteristic, were used. It could also be due to the fact that in the first case we didn't consider the phase and in the second case, the phase as well as the amplitudes of the equivalent low-pass representation were matched. How this filter performs in the generation of MSK signals and how it compares to the filter of the first case remains to be analyzed.

IV. CONCLUSIONS

In this report, methods have been developed for arriving at an approximation to the equivalent low-pass response, of an ideal MSK transmitter filter. The methods consist of adjusting the pole and zero locations of simple realizable filter configurations. The technique of minimizing an error criterion, by the method of steepest descent, is then employed to arrive at a good set of realizable filter parameters.

It is shown that using these methods, it is possible to closely approximate the power response or the transfer function of the equivalent low-pass representation, of an ideal MSK filter. The real-world performance of these approximations, in generating the desired MSK signal, is a problem that has not yet been analyzed. If, as it is hoped, these simple realizable configurations do succeed in acceptable generation of the MSK signal, a significant reduction in transmitter complexity and implementation cost will have been achieved.

APPENDIX

Computer Programs

```
C* * * * * * * * * * * * * * * * * * * * *
C
C              MATCHING POWER RESPONSE PROGRAM
C
C                         MARK89.FR
C
C     PROGRAMMED BY MARK FLIN                   12/20/79
C
C* * * * * * * * * * * * * * * * * * * * *
      DIMENSION FREQ(17),ERROR(17),APXM(68),DELUN(4),DTNFR(17),
     1 ATNFR(17),RSK(500),ESQD(17),CATNFR(1500),CDTNFR(1500)
      REAL K
      DUMMY1=0.
      DELUN(1)=0.
      DELUN(2)=0.
      DELUN(3)=0.
      DELUN(4)=0.
      DO 25 I=1,17
              FREQ(I)=0.
              ERROR(I)=0.
25    CONTINUE
      DO 50 I=1,68
              APXM(I)=0.
50    CONTINUE
      P1=0.
      A=0.
      B=0.
      WO=0.
C
C     ENTER FREQUENCIES AT WHICH IT IS DESIRED TO MATCH
C     DESIRED AND APPROXIMATING POWER RESPONSES
C
      FREQ(1)=1.E-06
      DO 100 I=2,17
              FREQ(I)=FREQ(I-1)+(3.1415)/4.
100   CONTINUE
C
C     INITIALIZE MULTIPLICATION CONSTANT OF STEEPEST DESCENT RULE
C
              ACCEPT " CONSTANT K=?",K
C
C     OBTAIN INITIAL VALUES FOR PARAMETERS
C
              ACCEPT " REAL POLE=?",P1
              ACCEPT " REAL PART OF COMPLEX POLE PAIR=?",A
              ACCEPT " IMAGINARY PART OF COMPLEX POLE PAIR=?",B
              ACCEPT " IMAGINARY AXIS ZERO=?",WO
              ACCEPT " VALUE FOR PERIOD=?",T
      M=1
C
```

```
C       ASSIGN REDUNDANT PORTIONS OF DERIVATIVE EQUATIONS TO DUMMY
C       VARIABLES
C
200     CONTINUE
                W02=W0**2
                W03=W02*W0
                P12=P1**2
                A2=A**2
                A3=A2*A
                B2=B**2
                B3=B2*B
                ANUM1=(A2+B2)
                ANUM2=ANUM1**2
C
C       EVALUATE INITIAL DERIVATIVE MATRIX
C
        DO 300 I=1,17
                W=FREQ(I)
                W2=W**2
                WB=2*W*B
                ANUM=(W02-W2)**2
                DENOM=(W2-WB+A2+B2)*(W2+WB+A2+B2)*(W2+P12)*W02
                DENOM2=DENOM**2
C
        CALCULATE PARTIAL OF POWER RESPONSE WITH RESPECT TO P1
                APXM(I)=2.*P1*ANUM*ANUM2*((W2+P12)-1.*P12)/(DENOM*
     1          (W2+P12)*W02)
C
C       CALCULATE PARTIAL WITH RESPECT TO WO
                APXM(I+17)=4.*P12*(W02-W2)*ANUM2*W2/(DENOM*W03)
C
C       CALCULATE PARTIAL WITH RESPECT TO A
                APXM(I+34)=2.*ANUM1*ANUM*P12*(W2+P12)*((W2-WB+A2+B2)*
     1          (W2+WB+A2+B2)*2.*A-(A2+B2)*(2.*W2*A+2.*A*B2+2.*A3))
     2          /DENOM2
C
C       CALCULATE PARTIAL WITH RESPECT TO B
                APXM(I+51)=2.*ANUM1*ANUM*P12*(W2+P12)*((W2-WB+A2+B2)*
     1          (W2+WB+A2+B2)*2.*B-(A2+B2)*(-2.*W2*B+2.*A2*B+2.*B3))
     2          /DENOM2
C
C       CALCULATE THE ERROR VECTOR BETWEEN APPROXIMATING AND DESIRED
C       POWER RESPONSES
C
                DTNFR(I)=((SIN(W*T/2.))**2)/(W*T/2.)**2
                ATNFR(I)=ANUM*ANUM2*P12/(DENOM*W02)
                ERROR(I)=DTNFR(I)-ATNFR(I)
                IF(I.EQ.1.AND.M.GT.1) GO TO 250
                GO TO 275
250     CONTINUE
```

```
                RSK(M)=DUMMY1
                DUMMY1=0.
275     CONTINUE
                DUMMY1=DUMMY1+(ERROR(I)**2)
300     CONTINUE
C
C       CALCULATE DELTAUN, THE CHANGE IN PARAMETERS TO BE TRIED NEXT
C
        L=1
        J=1
        DUMMY=0.
        DO 500 I=1,68
                DUMMY=DUMMY+(K*APXM(I)*ERROR(L))
                IF(I.EQ.17.OR.I.EQ.34.OR.I.EQ.51.OR.I.EQ.68) GO TO 400
                L=L+1
                GO TO 500
400             DELUN(J)=DUMMY
                J=J+1
                L=1
500     CONTINUE
C
C       CALCULATE NEW PARAMETERS TO BE CONSIDERED
C
        P1=P1+DELUN(1)
        W0=W0+DELUN(2)
        A=A+DELUN(3)
        B=B+DELUN(4)
        IF(M.EQ.500) GO TO 600
        M=M+1
        GO TO 200
C
C       USING NEW FOUND PARAMETERS COMPARE CONTINUOUSLY, DESIRED AND
C       APPROXIMATING POWER RESPONSES FROM 0 TO 6.28 RAD.
C
600     CONTINUE
        TYPE "P1=",P1,"W0=",W0,"A=",A,"B=",B
        W02=W0**2
        P12=P1**2
        A2=A**2
        A3=A2*A
        B2=B**2
        B3=B2*B
        ANUM1=A2+B2
        ANUM2=ANUM1**2
        W=0.
        DO 700 I=1,1500
                W=W+(3.14159)/200
                W2=W**2
                WB=2*W*B
                ANUM=(W02-W2)**2
```

```
              DENOM=(W2-WB+A2+B2)*(W2+WB+A2+B2)*(W2+P12)*W02
              CATNFR(I)=ANUM*ANUM2*P12/(DENOM*W02)
              CDTNFR(I)=((SIN(W*T/2.))**2)/(W*T/2.)**2
700       CONTINUE
C
C         STORE DATA ON DISK
C
          CALL IOPEN(0,3,2,0,6000,"APPROXIMATION?")
          CALL WRITR(0,0,CATNFR,1,IERR0)
          CALL CLOSE(0,IERR0)
          CALL IOPEN(1,3,2,0,6000,"DESIRED?")
          CALL WRITR(1,0,CDTNFR,1,IERR1)
          CALL CLOSE(1,IERR1)
          CALL IOPEN(2,3,2,0,M*4,"RISK?")
          CALL WRITR(2,0,RSK,1,IERR2)
          CALL CLOSE(2,IERR2)
          STOP
          END
```

```
C*  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
C
C           MATCHING TRANSFER FUNCTION PROGRAM
C
C                       MARK103.FR
C
C     PROGRAMMED BY MARK FLIN                 12/20/79
C
C*  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *
      DIMENSION FREQ(9),ERROR(18),APXM(72),DTNFR(18),DELR(4),
     1 ATNFR(18),RSK(200),ESQD(18),CATNFR(900),CDTNFR(900),
     2 CATNFRP(900),DELI(4),CDTNFRP(900)
      REAL K
      DUMMY1=0.
      DO 25 I=1,9
              FREQ(I)=0.
25    CONTINUE
      DO 35 I=1,18
              ERROR(I)=0.
35    CONTINUE
      DO 50 I=1,72
              APXM(I)=0.
50    CONTINUE
      A1=0.
      A2=0.
      B1=0.
      B2=0.
C
C     ENTER FREQUENCIES AT WHICH IT IS DESIRED TO MATCH THE
C     DESIRED AND APPROXIMATING TRANSFER FUNCTIONS
C
      FREQ(1)=1.E-06
      DO 100 I=2,9
              FREQ(I)=FREQ(I-1)+(3.1415)/4.
100   CONTINUE
C
C     INITIALIZE MULTIPLICATION CONSTANT K
C
      ACCEPT " CONSTANT K=?",K
C
C     OBTAIN INITIAL PARAMETER VALUES
C
      ACCEPT " REAL PART OF 1ST COMPLEX PAIR=?",A1
      ACCEPT " IMAGINARY PART OF 1ST COMPLEX PAIR=?",B1
      ACCEPT " REAL PART OF 2ND COMPLEX PAIR=?",A2
      ACCEPT " IMAGINARY PART OF 2ND COMPLEX PAIR=?",B2
      ACCEPT " VALUE FOR PERIOD=?",T
      M=1
C
C     ASSIGN REDUNDANT PORTIONS OF EQUATIONS TO DUMMY
```

```
C       VARIABLES
C
200     CONTINUE
        DO 210 I=1,4
                DELI(I)=0.
                DELR(I)=0.
210     CONTINUE
        HNUM1=(A1**2)+(B1**2)
        HNUM2=(A2**2)+(B2**2)
        A12=A1**2
        A13=A12*A1
        B12=B1**2
        B13=B12*B1
        A22=A2**2
        A23=A22*A2
        B22=B2**2
        B23=B22*B2
C
C       EVALUATE INITIAL DERIVATIVE MATRIX
C
        DO 300 I=1,18
                IF(I.GT.9) GO TO 215
                W=FREQ(I)
                GO TO 220
215     CONTINUE
                W=FREQ(I-9)
220     CONTINUE
                W2=W**2
                W3=W2*W
                W4=W3*W
                DENOMR=(-1.*W3*W3+6.*W4*A12+2.*W4*B12-2.*W2*A12*B12
     1  -1.*W2*A12*A12-1.*W2*B12*B12+8.*W4*A1*A2-8.*W2*A13*A2
     2  -8.*W2*A1*B12*A2+W4*A22-6.*W2*A12*A22-2.*W2*B12*A22
     3  +2.*A12*B12*A22+A12*A12*A22+B12*B12*A22+W4*B22-6.*W2*A12*B22
     4  -2.*W2*B12*B22+2.*A12*B12*B22+A12*A12*B22+B12*B12*B22)
                DENOMI=(4.*W4*W*A1-4.*W3*A13-4.*W3*A1*B12+2.*W4*W*A2
     1  -12.*W3*A12*A2-4.*W3*B12*A2+4.*W*A12*B12*A2+2.*W*A13*A1*A2
     2  +2.*W*B13*B1*A2-4.*W3*A1*A22+4.*W*A13*A22+4.*W*A1*B12*A22
     3  -4.*W3*A1*B22+4.*W*A13*B22+4.*W*A1*B12*B22)
                DENOM1=(DENOMR**2)+(DENOMI**2)
                ANUMAR=(W*A12*A22+W*A12*B22-1.*W*A22*B12-1.*
     1  W*B12*B22)
                ANUMAI=(W2*A1*A22+W2*A1*B22)
                ANUMBR=(2.*W*A1*A22*B1+2.*W*A1*B12*B22)
                ANUMBI=(W2*A22*B1+W2*B1*B22)
                ANUM=(HNUM1*HNUM2)
C
C       CALCULATE PARTIAL OF APPROX FUNCTION WITH RESPECT TO A1/A2
C       REAL PART
C
```

```
                 APXM(I)=2.*((ANUMAR*DENOMR)+(ANUMAI*DENOMI))
      1   /DENOM1
C         IMAGINARY PART
C
                 APXM(I+18)=2.*((ANUMAI*DENOMR)-(ANUMAR*DENOMI))
      1   /DENOM1
C
C         CALCULATE PARTIAL OF APPROX FUNCTION WITH RESPECT TO B1/B2
C         REAL PART
C
                 APXM(I+36)=2.*((ANUMBR*DENOMR)+(ANUMBI*DENOMI))
      1   /DENOM1
C
C         IMAGINARY PART
C
                 APXM(I+54)=2.*((ANUMBI*DENOMR)-(ANUMBR*DENOMI))
      1   /DENOM1
C
C         CALCULATE ERROR VECTOR BETWEEN APPROX & DESIRED FUNCTIONS
C         ASSIGN REDUNDANT PARTS OF FUNCTIONS TO DUMMY VARIABLES
C
                 ADENR=(W4-1.*W2*A22-1.*W2*B22-4.*W2*A1*A2
      1   -1.*W2*A12+A12*A22+A12*B22-1.*W2*B12+A22*B12+B12*B22)
                 ADENI=(-2.*W3*A2-2.*W3*A1+2.*W*A1*A22
      1   +2.*W*A1*B22+2.*W*A12*A2+2.*W*A2*B12)
                 DENOM2=(ADENR**2)+(ADENI**2)
C
C         CALCULATE REAL PART OF APPROX AND DESIRED FUNCTIONS AND
C         ERROR VECTOR BETWEEN THEM
C
                 IF(I.GT.9) GO TO 235
                 DTNFR(I)=(-1.*T)*((SIN(W*T/2.))**2)/(W*T/2.)
                 ATNFR(I)=(-1.*ANUM*ADENI)/DENOM2
                 ERROR(I)=DTNFR(I)-ATNFR(I)
                 GO TO 245
C
C         CALCULATE IMAGINARY PARTS OF ABOVE FUNCTIONS AND VECTORS
C
235       CONTINUE
                 DTNFR(I)=(-1.*T)*SIN(W*T/2.)*COS(W*T/2.)/(W*T/2.)
                 ATNFR(I)=(-1.*ANUM*ADENR)/DENOM2
                 ERROR(I)=-1.*(DTNFR(I)-ATNFR(I))
C
C         ERROR CONTAINS THE CONJUGATE OF THE ERROR VECTOR
245       CONTINUE
                 IF(I.EQ.1.AND.M.GT.1) GO TO 250
                 GO TO 275
250       CONTINUE
                 RSK(M)=DUMMY1
                 DUMMY1=0.
```

```
C
C       THE RISK FUNCTION IS EQUAL TO THE SUM OF THE SQUARES OF
C       THE ELEMENTS OF THE ERROR VECTOR, I.E. THE ERROR VECTOR
C       TIMES ITS CONJUGATE
C
275     CONTINUE
                DUMMY1=DUMMY1+(ERROR(I)**2)
                IF(I.NE.9) GO TO 300
C
C       DUE TO SYMMETRY, THE PARTIALS WITH RESPECT TO A2&B2 CAN
C       BE FOUND USING THE SAME EQUATIONS USED FOR A1&B1, BY
C       SWAPPING THE VALUES OF A1 AND B1 WITH A2 AND B2
C       SWAP A1 WITH A2
C
                C=A1
                D=A2
                A1=D
                A2=C
C
C       SWAP B1 WITH B2
                E=B1
                F=B2
                B1=F
                B2=E
C
C       RECALCULATE REDUNDANT PORTIONS OF EQUATIONS
C       USING A2&B2 INSTEAD OF A1&B1
C
                A12=A1**2
                A13=A12*A1
                B12=B1**2
                B13=B12*B1
                A22=A2**2
                A23=A22*A2
                B22=B2**2
                B23=B22*B2
300     CONTINUE
C
C       SWAP BACK PARAMETERS
C
        C=A1
        D=A2
        A1=D
        A2=C
        E=B1
        F=B2
        B1=F
        B2=E
C
C       CALCULATE THE PARAMETER CHANGE, DELR(I)
```

```
C       REAL PART OF CHANGE
C
        DO 500 I=1,9
                DELR(1)=DELR(1)+K*((APXM(I)*ERROR(I))-
      1 (APXM(I+18)*ERROR(I+9)))
                DELR(2)=DELR(2)+K*((APXM(I+9)*ERROR(I))-
      1 (APXM(I+27)*ERROR(I+9)))
                DELR(3)=DELR(3)+K*((APXM(I+36)*ERROR(I))-
      1 (APXM(I+54)*ERROR(I+9)))
                DELR(4)=DELR(4)+K*((APXM(I+45)*ERROR(I))-
      1 (APXM(I+63)*ERROR(I+9)))
C
C       IMAGINARY PART OF CHANGE
C
                DELI(1)=DELI(1)+K*((APXM(I)*ERROR(I+9))+(APXM(I+18)*
      1 ERROR(I)))
                DELI(2)=DELI(2)+K*((APXM(I+9)*ERROR(I+9))+
      1 (APXM(I+27)*ERROR(I)))
                DELI(3)=DELI(3)+K*((APXM(I+36)*ERROR(I+9))+
      1 (APXM(I+54)*ERROR(I)))
                DELI(4)=DELI(4)+K*((APXM(I+45)*ERROR(I+9))+
      1 (APXM(I+63)*ERROR(I)))
500     CONTINUE
C
C       CALCULATE NEW PARAMETERS
C
        A1=A1+DELR(1)
        A2=A2+DELR(2)
        B1=B1+DELR(3)
        B2=B2+DELR(4)
        IF(M.EQ.200) GO TO 600
        M=M+1
        DO 550 I=1,4
550     CONTINUE
        GO TO 200
C
C       USING NEW PARAMETERS, COMPARE CONTINUOUSLY DESIRED AND
C       APPROXIMATING TRANSFER FUNCTIONS
C
600     CONTINUE
        W=0.
        A12=A1**2
        A22=A2**2
        B12=B1**2
        B22=B2**2
        HNUM1=(A12+B12)
        HNUM2=(A22+B22)
        ANUM=HNUM1*HNUM2
        TYPE " RISK=",RSK(M)
        TYPE "A1=",A1,"B1=",B1,"A2=",A2,"B2=",B2
```

```
      DO 700 I=1,900
              W=W+(3.1415)/150.
              W2=W**2
              W3=W2*W
              W4=W3*W
              ADENR=(W4-1.*W2*A22-1.*W2*B22-4.*W2*A1*A2-1.*W2*A12
     1 +A12*A22+A12*B22-1.*W2*B12+A22*B12+B12*B22)
              ADENI=(-2.*W3*A2-2.*W3*A1+2.*W*A1*A22+2.*W*A1*B22
     1 +2.*W*A12*A2+2.*W*A2*B12)
              DENOM2=(ADENR**2)+(ADENI**2)
              ATNFRR=(-1.*ANUM*ADENI)/DENOM2
              DTNFRR=(-1.*T)*((SIN(W*T/2.))**2)/(W*T/2.)
              ATNFRI=(-1.*ANUM*ADENR)/DENOM2
              DTNFRI=(-1.*T)*SIN(W*T/2.)*COS(W*T/2.)/(W*T/2.)
              CATNFR(I)=SQRT((ATNFRR**2)+(ATNFRI**2))
              CDTNFR(I)=SQRT((DTNFRR**2)+(DTNFRI**2))
              CATNFRP(I)=ATAN(ATNFRI/ATNFRR)
              CDTNFRP(I)=ATAN(DTNFRI/DTNFRR)
              ATNFRR=0.
              DTNFRR=0.
              ATNFRI=0.
              DTNFRI=0.
700   CONTINUE
C
C     STORE DATA ON DISK
C
      CALL IOPEN(0,3,2,0,3600,"APPROXIMATION?")
      CALL WRITR(0,0,CATNFR,1,IERR0)
      CALL CLOSE(0,IERR0)
      CALL IOPEN(1,3,2,0,3600,"DESIRED?")
      CALL WRITR(1,0,CDTNFR,1,IERR1)
      CALL CLOSE(1,IERR1)
      CALL IOPEN(2,3,2,0,M*4,"RISK?")
      CALL WRITR(2,0,RSK,1,IERR2)
      CALL CLOSE(2,IERR2)
      CALL IOPEN(3,3,2,0,3600,"APPROX PHASE?")
      CALL WRITR(3,0,CATNFRP,1,IERR3)
      CALL CLOSE(3,IERR3)
      CALL IOPEN(3,3,2,0,3600,"DESIRED PHASE?")
      CALL WRITR(3,0,CDTNFRP,1,IERR3)
      CALL CLOSE(3,IERR3)
      STOP
      END
```

REFERENCES

[1]. F. Amoroso and J. Kivett, "Simplified MSK Signaling Techniques," IEEE Transactions on Communications, Vol COM-25, No. 4, pp. 433-434, April 1977.

[2]. M. Schwartz, W. Bennett, and S. Stein, Communication Systems and Techniques, New York, N.Y.: MacGraw-Hill, 1966.

[3]. A. Sage, Optimum Systems Control, Englewood Cliffs, N.J.: Printice Hall, Inc., 1968.

[4]. M. Javid and E. Brenner, Analysis, Transmission, and Filtering of Signals, New York, N.Y.: MacGraw-Hill, 1963

THE DESIGN OF A SERIAL MSK FILTER

by

Mark Earl Flin

B.S., Kansas State University, 1978

A Master's Report

Submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Electrical Engineering

KANSAS STATE UNIVERSITY

Manhattan, Kansas

1980

ABSTRACT

This report considers the development of methods for designing a realizable Minimum Shift Keying (MSK) transmitter filter, by application of the method of steepest descent. Two realizable configurations are analyzed. It is determined how closely each configuration approximates the equivalent low-pass response of an ideal MSK filter. Results are tabulated and responses of interest are graphed. Listings of computer programs developed for the analysis are included.