An Automated Framework for Multicriteria Optimization of Analog Filter Designs

Niranjan Damera-Venkata, *Student Member, IEEE,* and Brian L. Evans, *Senior Member, IEEE*

***Abstract*— This paper presents an extensible framework for designing analog filters that exhibit several desired behavioral properties *after* being realized in circuits. In the framework, we model the constrained nonlinear optimization problem as a sequential quadratic programming (SQP) problem. SQP requires real-valued constraints and objective functions that are differentiable with respect to the free parameters (pole–zero locations). We derive the differentiable constraints and a weighted differentiable objective function for simultaneously optimizing the behavioral properties of magnitude response, phase response, peak overshoot, and the implementation property of quality factors. We use Mathematica to define the algebraic equations for the constraints and objective function, compute their gradients symbolically, and generate standalone MATLAB programs to perform the multicriteria optimization. Providing closed-form gradients prevents divergence in the SQP procedure. The automated approach avoids errors in algebraic calculations and errors in transcribing equations into software. The key contributions are: 1) an extensible, automated, multicriteria filter optimization framework; 2) an analytic approximation for peak overshoot; and 3) three novel filter designs. We have released the source code for the framework on the Internet.**

***Index Terms*—Analog filter optimization, hybrid filters, multicriteria optimization.**

I. Introduction

CLASSICAL elliptic Chebyshev, Butterworth, and Bessel analog filter designs yield desirable behavioral properties subject to constraints on the magnitude response. For example, the step response of Bessel filters exhibits low overshoot, and its phase response is nearly linear over the passband. Bessel filters have been used as antialiasing filters, since antialiasing filters require a minimum deviation in the phase response from linear phase, subject to a set of magnitude specifications [1]. Classical elliptic filter designs have minimal order, but for a given implementation technology, minimal-order filters may either not be realizable or may not have minimal complexity [2]. In designing analog filters for implementation, multiple behavioral properties (e.g., magnitude response, phase response, peak overshoot, rise time, and settling time) *and* implementation properties (e.g., quality factors and capacitance spread) may be important.

In this paper, we present a formal extensible framework for simultaneously optimizing analog filter designs for multiple behavioral and implementation properties. We demonstrate the framework using the behavioral properties of magnitude response, phase response, and peak overshoot, and the implementation property of quality factors. The framework takes an initial filter design, e.g., one designed using a classical numeric approach or a modern symbolic approach [2], and finds the pole–zero locations that optimize a weighted combination of properties subject to constraints on the properties. Some of the previous multicriteria filter optimization techniques, such as [3], only optimize for one property subject to constraints on multiple properties. Another technique applies sequential quadratic programming (SQP) methods to optimize loss and delay in digital filter designs [3].

Our framework models the constrained nonlinear optimization problem as a SQP problem. SQP requires that the objective function [4] and the constraints [5] be real valued and twice continuously differentiable, with respect to the free parameters. The free parameters are the pole and zero locations. When closed-form formulas for the gradients of the objective function and constraints are not provided for SQP routines, the SQP routines must approximate the gradient, which often leads to divergence. We develop Mathematica [6] software to compute the gradients and translate the entire SQP formulation into working MATLAB [7] programs that optimize analog filter designs. The generated MATLAB code is combined with the SQP procedure in the MATLAB optimization toolbox [8] to produce stand-alone programs that solve the constrained nonlinear optimization problem.

The framework is flexible because it is formulated at an algebraic level. At the algebraic level, a designer can use a symbolic mathematics environment such as Mathematica to change the objective measure for a given property or add, delete, and change constraints. Our symbolic software will then recompute the gradient and regenerate the numerical optimization code. We have bridged the gap between the symbolic work designers often do on paper and the working computer implementation, thereby eliminating algebraic errors in hand calculations and bugs in coding the equations in software.

Section II reviews notation. Section III derives a family of weighted differentiable objective functions to measure the deviation in magnitude response, deviation in linear-phase response, quality factors, and peak overshoot of the step

Manuscript received July 14, 1998; revised March 19, 1999. This work was supported by the National Science Foundation under CAREER Award MIP-9702707 and by the Defense Advanced Research Projects Agency on the Composite CAD Program under DARPA Contract DAAB07-97-C-J007, through a subcontract from the Ptolemy Project, University of California at Berkeley. This paper was recommended by Associate Editor W. Kao.

The authors are with the Embedded Signal Processing Laboratory, Department of Electrical and Computer Engineering, The University of Texas, Austin, TX 78712 USA.

Publisher Item Identifier S 1057-7130(99)06536-2.

response, of an analog filter. In the derivation, we find a new analytic approximation for the peak overshoot. Section IV converts filter specifications on the magnitude response, quality factors, and peak overshoot into differentiable constraints. Section V reports three novel design examples found by our filter optimization framework. Section VI describes the process by which we verified the formulas in Section III, generated MATLAB code for the objective function and constraints as well as their gradients, and validated the generated MATLAB code. Section VII concludes the paper. The automated design framework, including the generated MATLAB code, is available online.[1]

II. Notation

We represent an analog filter by its n complex conjugate pole pairs and r complex conjugate zero pairs, such that $r \leq n$. We denote the kth pole pair as $p_k = a_k \pm jb_k$, where $a_k < 0$ for stability, and the lth zero pair as $z_l = c_l \pm jd_l$. We arbitrarily choose b_k and c_l to be negative and d_l to be nonpositive. The magnitude response $|G(j\omega)|$ and unwrapped phase response $\angle G(j\omega)$ of an all-pole filter, expressed as real-valued differentiable functions, are

$$|G(j\omega)| = \prod_{k=1}^{n} \frac{a_k^2 + b_k^2}{\sqrt{a_k^2 + (\omega + b_k)^2}\sqrt{a_k^2 + (\omega - b_k)^2}} = \prod_{k=1}^{n} \frac{a_k^2 + b_k^2}{\sqrt{(\omega^2 + 2(a_k^2 - b_k^2))\omega^2 + (a_k^2 + b_k^2)^2}} \quad (1)$$

$$\angle G(j\omega) = \sum_{k=1}^{n} \arctan\left(\frac{\omega - b_k}{a_k}\right) + \arctan\left(\frac{\omega + b_k}{a_k}\right). \quad (2)$$

We factor the polynomial under the square root in (1) into Horner's form because it has better numerical properties. Together with the zero pairs, the magnitude and unwrapped phase responses, respectively, are

$$|H(j\omega)| = |G(j\omega)| \cdot \prod_{l=1}^{r} \frac{\sqrt{(\omega^2 + 2(c_l^2 - d_l^2))\omega^2 + (c_l^2 + d_l^2)^2}}{c_l^2 + d_l^2} \quad (3)$$

$$\angle H(j\omega) = \angle G(j\omega) - \sum_{l=1}^{r} \arctan\left(\frac{\omega - d_l}{c_l}\right) + \arctan\left(\frac{\omega + d_l}{c_l}\right). \quad (4)$$

We assume that the filter is low pass. Its dc response ($\omega = 0$) is normalized to be 1.

In this paper, Q represents quality factors, ϵ represents a small positive number, σ denotes deviation, m represents slope of a line, t is time, and W is a weighting factor.

III. Objective Function

In this section, we derive an objective function that is real and twice continuously differentiable to match an SQP formulation. The objective function is a weighted combination of the deviation from the ideal filter for the desired properties. Properties are quantified by objective measures. Based on the objective measures for magnitude and phase responses given in the previous section, Sections III-A and III-B define deviation from an ideal magnitude response and deviation from a phase response that is linear in the passband, respectively. Section III-C defines quality factors and deviation from ideal quality factors. Section III-D derives a new analytic approximation to measure peak overshoot in the step response and defines deviation from the ideal filter overshoot. Section III-E defines the objective function. The objective function is nonnegative so that a value of zero represents the ideal filter. Using the automated SQP-based framework, a designer may change the objective measures and distance measures to form new objective functions, and regenerate the new MATLAB programs to perform the optimization.

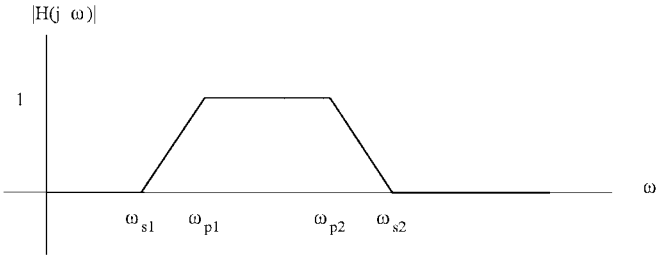


Fig. 1. The ideal magnitude response.

A. Deviation from an Ideal Magnitude Response

We use (3) as the real differentiable measure of magnitude response. We measure the deviation from the ideal magnitude-response separately in the passband, transition bands, and stopband(s). Based on the five regions of the ideal magnitude response, shown in Fig. 1, the five components of the objective function relate to the deviation from an ideal magnitude response in the least-squares sense in each region. Assuming that the ideal filter is low-pass or bandpass with the first stopband located on $\omega \in (0, \omega_{s1})$, the passband located on $\omega \in (\omega_{p1}, \omega_{p2})$, and the second stopband located on $\omega \in (\omega_{s2}, \infty)$

$$\sigma_{\mathrm{sb1}} = \int_0^{\omega_{s1}} F_{s1}(\omega)|H(j\omega)|^2\, d\omega \quad (5)$$

$$\sigma_{\mathrm{tb1}} = \int_{\omega_{s1}}^{\omega_{p1}} F_{t1}(\omega)(|H(j\omega)| - (m_1\omega - m_1\omega_{s1}))^2\, d\omega \quad (6)$$

$$\sigma_{\mathrm{pb}} = \int_{\omega_{p1}}^{\omega_{p2}} F_p(\omega)(|H(j\omega)| - 1)^2\, d\omega \quad (7)$$

$$\sigma_{\mathrm{tb2}} = \int_{\omega_{p2}}^{\omega_{s2}} F_{t2}(\omega)(|H(j\omega)| - (m_2\omega - m_2\omega_{s2}))^2\, d\omega \quad (8)$$

$$\sigma_{\mathrm{sb2}} = \int_{\omega_{s2}}^{\infty} F_{s2}(\omega)|H(j\omega)|^2\, d\omega \quad (9)$$

where $F_p(\omega)$, $F_{t1}(\omega)$, $F_{t2}(\omega)$, and $F_s(\omega)$ are integrable weighting functions, and m_1 and m_2 are the slopes of

[1] [Online]. Available HTTP: http://www.ece.utexas.edu/~bevans/projects/syn_filter_software.html

the ideal response in the transition regions defined as $m_1 = 1/(\omega_{p1} - \omega_{s1})$ and $m_2 = 1/(\omega_{p2} - \omega_{s2})$. These five integral quantities represent deviation from the ideal magnitude response in the least squares sense over the five regions shown in Fig. 1.

B. Deviation from an Ideal Phase Response

We use (4) as a real differentiable measure of the phase response. We measure the deviation of a filter from linear phase over some range of frequencies (usually over the passband)

$$\sigma_{\text{phase}} = \int_{\omega_1}^{\omega_2} \left(\angle H(j\omega) - m_{lp}\omega\right)^2 d\omega. \tag{10}$$

Here, m_{lp} is the slope of the linear phase response. Unfortunately, one does not know the value of m_{lp} *a priori*. We can compute it as the slope of the line in ω that minimizes (10)

$$\min_{m_{lp}} \int_{\omega_1}^{\omega_2} \left(\angle H(j\omega) - m_{lp}\omega\right)^2 d\omega. \tag{11}$$

In (11), the $H(j\omega)$ term does not depend on m_{lp}, so the integrand is quadratic in m_{lp}. To find the minimum, we take the derivative with respect to m_{lp}, set it to zero, and solve for m_{lp}

$$m_{lp} = \frac{\displaystyle\int_{\omega_1}^{\omega_2} \angle H(j\omega)\omega \, d\omega}{\displaystyle\int_{\omega_1}^{\omega_2} \omega^2 \, d\omega}. \tag{12}$$

After evaluating the integrals

$$m_{lp} = \frac{3}{2(\omega_2^3 - \omega_1^3)} \cdot \left[\sum_{k=1}^{n} \left[f_{lp1}(\omega_2) - f_{lp1}(\omega_1)\right] - \sum_{l=1}^{r} \left[f_{lp2}(\omega_2) - f_{lp2}(\omega_1)\right]\right] \tag{13}$$

where $f_{lp1}(\omega)$ is

$$f_{lp1}(\omega) = 2\omega a_k + (b_k^2 - a_k^2 - \omega^2) \cdot \left(\arctan\left(\frac{\omega - b_k}{a_k}\right) + \arctan\left(\frac{\omega + b_k}{a_k}\right)\right) + a_k b_k \left(\log\left(1 + \frac{(\omega - b_k)^2}{a_k^2}\right) - \log\left(1 + \frac{(\omega + b_k)^2}{a_k^2}\right)\right)$$

and $f_{lp2}(\omega)$ is

$$f_{lp2}(\omega) = 2\omega c_l + (d_l^2 - c_l^2 - \omega^2) \cdot \left(\arctan\left(\frac{\omega - d_l}{c_l}\right) + \arctan\left(\frac{\omega + d_l}{c_l}\right)\right) + c_l d_l \left(\log\left(1 + \frac{(\omega - d_l)^2}{c_l^2}\right) - \log\left(1 + \frac{(\omega + d_l)^2}{c_l^2}\right)\right).$$

Using Mathematica, we compute the definite integrals in (12) to verify the answers. Now that we have a closed-form solution for m_{lp}, we can substitute (13) into (10) to obtain a rather complicated but differentiable expression for the deviation from linear phase.

C. Deviation from Ideal Filter Quality

The objective measure of filter quality, known as a quality factor, measures the relative distance of a filter pole from the imaginary frequency axis. The lower the quality factor, the less likely that the pole will cause oscillations in the output. A conventional definition for the quality factor Q_k for the kth second-order section with conjugate poles $a_k \pm jb_k$ (with $a_k < 0$) is given by

$$Q_k = \frac{\sqrt{a_k^2 + b_k^2}}{-2a_k}$$

where $Q_k \geq 0.5$. $Q_k = 0.5$ corresponds to a double real-valued pole ($b_k = 0$), and $Q_k = \infty$ corresponds to an ideal oscillator ($a_k = 0$). For the effective overall measure of filter factor Q_{eff}, we use a geometric mean of the individual pole-pair quality factors

$$Q_{\text{eff}} = \left(\prod_{k=1}^{n} Q_k\right)^{1/n} \tag{14}$$

where $Q_{\text{eff}} \geq 0.5$. Other objectives measures could be used. To measure the distance from the ideal quality factor of 0.5, we simply use

$$Q_{\text{eff}} - 0.5.$$

This distance measure rewards low quality factors because they are essential in damping oscillatory behavior in the time response of the filter.

D. Deviation from Ideal Peak Overshoot in the Step Response

From the step response, we can numerically compute the peak overshoot and the time t_{peak} at which it occurs. In order to make the peak overshoot calculation differentiable for the SQP-based framework, we derive a new analytic expression that approximates t_{peak} in terms of the pole–zero locations. The derivation assumes that there are no duplicate poles. The assumption of no duplicate poles is enforced by means of constraints, as explained in Section IV.

The Laplace transform of the step response is

$$\frac{H(s)}{s} = \frac{1}{s}\left[\prod_{k=1}^{n} \frac{a_k^2 + b_k^2}{s^2 - 2a_k s + a_k^2 + b_k^2}\right] \cdot \left[\prod_{l=1}^{r} \frac{s^2 - 2c_l s + c_l^2 + d_l^2}{c_l^2 + d_l^2}\right]. \tag{15}$$

Assuming no duplicate poles, partial fractions expansion yields

$$\begin{aligned} \frac{H(s)}{s} &= \left[\frac{A}{s} + \sum_{k=1}^{n} \frac{C_k s + D_k}{s^2 - 2a_k s + a_k^2 + b_k^2}\right] \\ A &= [H(s)]_{s=0} = 1 \\ B_k &= \left[(s - p_k)\frac{H(s)}{s}\right]_{s=p_k} = |B_k| e^{j\angle B_k} \\ C_k &= 2|B_k| \cos(\angle B_k) \\ D_k &= -2|B_k|(a_k \cos(\angle B_k) + b_k \sin(\angle B_k)) \end{aligned} \tag{16}$$

where $|B_k|$ and $\angle B_k$ can be expressed as real-valued differentiable functions of the pole and zero locations. After inverse transforming (16), the step response is

$$h_{\text{step}}(t) = 1 + \sum_{k=1}^{n} e^{a_k t}\left[C_k \cos(b_k t) + \left(\frac{D_k + C_k a_k}{b_k}\right)\sin(b_k t)\right]. \tag{17}$$

By substituting the definitions for D_k and C_k from (16) into (17), the step response simplifies to

$$h_{\text{step}}(t) = 1 + 2\sum_{k=1}^{n} |B_k| e^{a_k t}\cos(b_k t + \angle B_k). \tag{18}$$

The Appendix presents a rigorous derivation of the step response.

The overall step response, given by either (17) or (18), is one plus a sum of n second-order responses. Each second-order response is a function of all of the filter poles and zeros. By analyzing the kth term in the summation in (17), the kth peak overshoot occurs at time

$$t_{\text{peak}}^k = -\frac{1}{b_k}\left[\arctan\left(\frac{(D_k + 2C_k a_k)b_k}{C_k(a_k^2 - b_k^2) + D_k a_k}\right) + \pi\right] \tag{19}$$

provided that $b_k \neq 0$. By substituting for C_k and D_k in (19), the kth peak overshoot occurs at time

$$t_{\text{peak}}^k = -\frac{1}{b_k}\left[\arctan\left(\frac{b_k \tan(\angle B_k) - a_k}{b_k + \tan(\angle B_k)}\right) + \pi\right] \tag{20}$$

provided that $b_k \neq 0$. The additive π term in both (19) and (20) corrects the quadrant of the arctangent. Each second-order time response in (18) either increases monotonically from time zero until the peak time and then decreases to oscillate around the steady-state value of zero, or decreases monotonically from time zero to the peak time and then increases to oscillate around the steady-state value of zero. Therefore, the least and greatest peak times for the second-order responses provide a bound on the possible times at which the peak overshoot occurs, i.e.,

$$t_{\text{peak}} \in \left[\min_k t_{\text{peak}}^k, \max_k t_{\text{peak}}^k\right]. \tag{21}$$

In the SQP-based framework, (21) helps in two important areas. First, at each iteration, (21) gives a range of time over which to perform a one-dimensional search for the actual t_{peak} value. In the implementation, we search a broader interval. Second, from the observation that (21) states that t_{peak} is dependent on the values of t_{peak}^k, we construct the following differentiable function to approximate t_{peak} for the sole purpose of computing gradients for the peak overshoot of the filter

$$t_{\text{peak}} \approx \frac{1}{n}\sum_{k=1}^{n} t_{\text{peak}}^k \Rightarrow t_{\text{peak}} = \beta\frac{1}{n}\sum_{k=1}^{n} t_{\text{peak}}^k. \tag{22}$$

This analytic approximation is inferred from (17), in which the step response is written as a sum of a constant plus an equal additive contribution from each second-order section. At each iteration of the optimization procedure, β is set to the true value of t_{peak} divided by the approximation $1/n\sum_{k=1}^{n} t_{\text{peak}}^k$. Section VI-C validates the accuracy of the analytic approximation.

The deviation in peak overshoot is measured by a differentiable function that measures overshoot and undershoot. For low-pass filters, one measure of deviation in peak overshoot is the percent overshoot defined as $100\% \times (h_{\text{step}}(t_{\text{peak}}) - 1)$. This formula, however, assumes that the step response will rise above 1. We measure the deviation in step response amplitude when the peak overshoot occurs from the ideal amplitude of one, which corresponds to a peak overshoot of zero

$$(h_{\text{step}}(t_{\text{peak}}) - 1)^2.$$

E. The Complete Objective Function

The complete objective function is a weighted sum of the distance measures developed earlier in this section. Since the objective function will ultimately be handed off to a numerical optimizer, each infinity that appears in the limit of the definite integrations, such as in (9), must be approximated. We approximate ∞ as 10^d multiplied by the highest frequency specified (e.g., ω_{s2} in the case of low-pass and bandpass filters), where d represents the number of decades beyond the highest specified frequency. In assembling the composite objective function, we normalize the integrals by dividing by the length of the integration interval and scale the distance measure for the peak overshoot, so that when the weights are equal, each distance measure will contribute more equally. Note that the weighted objective function is nonnegative and twice differentiable

$$\begin{aligned} &W_{\text{sb1}}\frac{1}{\omega_{s1}}\sigma_{\text{sb1}} + W_{\text{tb1}}\frac{1}{\omega_{p1} - \omega_{s1}}\sigma_{\text{tb1}} + W_{\text{pb}}\frac{1}{\omega_{p2} - \omega_{p1}}\sigma_{\text{pb}} \\ &\quad + W_{\text{tb2}}\frac{1}{\omega_{s2} - \omega_{p2}}\sigma_{\text{tb2}} + W_{\text{sb2}}\frac{1}{10^d\omega_{s2} - \omega_{s2}}\sigma_{\text{sb2}} \\ &\quad + W_{lp}\frac{1}{\omega_{p2} - \omega_{p1}}\sigma_{lp} + W_Q(Q_{\text{eff}} - 0.5) \\ &\quad + W_{\mathcal{O}}1000(h_{\text{step}}(t_{\text{peak}}) - 1)^2. \end{aligned} \tag{23}$$

For a low-pass filter, $\omega_{s1} = 0$, $\omega_{p1} = 0$, $W_{\text{tb1}} = 0$, and $W_{\text{sb1}} = 0$. For a high-pass filter, $\omega_{p2} = \infty \approx 10^d\omega_{p1}$, ω_{s2} is undefined, $W_{\text{tb2}} = 0$, and $W_{\text{sb2}} = 0$.

IV. Constraints

The first set of constraints are on the magnitude response, peak overshoot, and quality; the second set prevents numerical instabilities in the calculations and enforces assumptions about the poles and zeros. For the magnitude-response constraints, we sample the magnitude response given by (3) at a set of passband frequencies $\{\omega_k\colon \omega_k \leq \omega_p\}$ and stopband frequencies $\{\omega_l\colon \omega_l \geq \omega_s\}$

$$1 - \delta_p \leq |H(j\omega_k)| \leq 1\ \forall\, k, \qquad |H(j\omega_l)| \leq \delta_s\ \forall\, l \tag{24}$$

where δ_p, ω_p, δ_s and ω_s are the magnitude specifications. For the peak-overshoot constraint, we compute the peak overshoot by searching a larger interval than $t \in [\min_k t_{\text{peak}}^k, \max_k t_{\text{peak}}^k]$ to find the maximum value of the

step response in (17). Before finding the gradient of this constraint, we substitute $t = t_{\text{peak}}$ in (17) by using the analytic approximation for t_{peak} given by (22).

Lower quality factors mean better noise immunity and lower chances of an oscillating time response. When the analog filter is implemented, the second-order sections will typically be cascaded in order of ascending quality factors. The earlier sections will attenuate input signals so as to minimize the oscillatory behavior of the final sections. The implementation technology imposes an upper limit on the quality factors, $Q_{\max}$, for each second-order section

$$\frac{\sqrt{a_k^2 + b_k^2}}{-2a_k} < Q_{\max}, \quad \text{for} \quad k = 1, \cdots, n. \tag{25}$$

The designer is free to set the value of $Q_{\max}$. Section VI-B explains how to set $Q_{\max}$ in the synthesized MATLAB code.

Since a_k appears in the denominators in (2) and (13), b_k appears in the denominators in (17) and (19), and c_l appears in the denominators in (4) and (13), we constrain these negative-valued parameters to be a neighborhood away from zero

$$\begin{aligned} a_k &< -\epsilon_{\text{div}} < 0, \quad \text{for} \quad k = 1, \cdots, n \\ b_k &< -\epsilon_{\text{div}} < 0, \quad \text{for} \quad k = 1, \cdots, n \\ c_l &< -\epsilon_{\text{div}} < 0, \quad \text{for} \quad l = 1, \cdots, r. \end{aligned}$$

Here, ϵ_{div} is the distance from 1.0 to the next largest floating point number. In MATLAB, it is defined by the `eps` constant, which is 2.2204×10^{-16} on a Sun Ultra-2 workstation. To ensure the numerical stability of the denominators of $|B_k|$ and $\angle B_k$ in (16)

$$\sqrt{a_k - a_m} > \epsilon_{\text{div}}, \quad \text{for } k = 1, \cdots, n \text{ and } m = k+1, \cdots, n.$$

This set of constraints also prevents duplicate poles and poles from becoming too close to one another relative to the available numerical precision.

It is possible that an initial filter design does not meet all of the constraints. For example, classic filter design algorithms often yield filters with extraordinarily large quality factors which may be exceed $Q_{\max}$. When an initial guess is infeasible, the SQP procedure in MATLAB will update the free parameters until the constraint or constraints that were initially violated are satisfied. This relaxation occurs in the design example discussed in Section V-B.

V. Example Filter Designs

This section presents three new low-pass filter designs found by the automated filter optimization framework. In Sections V-A and V-B, the filters have minimized deviation from linear phase over the passband and peak overshoot in the step response. In Section V-C, the filter has minimized magnitude response in the stopband, deviation from linear phase in the passband, peak overshoot, and quality factors. All execution times are given for MATLAB 5 on a 167-MHz Sun Ultra workstation.

TABLE I
Quality Factors and Pole Locations for the Two Second-Order Sections of the Initial and Optimized Filters

Q	Poles	Q	Poles
0.54	$-20.3153 \pm j\,8.4149$	0.50	$-19.5623 \pm j\,0.6255$
1.31	$-8.4149 \pm j\,20.3153$	1.55	$-7.7918 \pm j\,22.8984$
(a)		(b)	

The optimized filter has minimal deviation from linear phase in the passband and peak overshoot. Fig. 2 compares the behavior of the two filters.

A. All-Pole Filter with Near-Linear-Phase Response and Minimal Peak Overshoot

We apply the framework to optimize an all-pole filter to obtain near-linear-phase response *and* minimal peak overshoot. The magnitude specifications are $\omega_p = 20$ rad/s, $\delta_p = 0.21$, $\omega_s = 30$ rad/s, and $\delta_s = 0.31$. The initial design is a fourth-order Butterworth filter. We jointly optimize the pole locations to achieve minimal peak overshoot and minimal deviation from linear phase in the passband phase response. In the objective function, we weight the deviation from linear phase by 0.1 and deviation in peak overshoot by one. All other weights are zero. We set $Q_{\max}$ to 10. The nonnegative objective function is reduced from 1.17 to 4.7×10^{-5}. The peak overshoot of the resulting filter is reduced from 16% to 8%, and the phase becomes approximately linear in the passband. Since the magnitude response was not optimized, it is traded off for better phase response and lower overshoot, but kept within specifications. Table I lists the initial and optimized pole locations. Fig. 2 plots the magnitude, phase, and step responses for the initial and optimized filters. The plots illustrate that the optimization procedure effectively traded off magnitude response in the passband for a more linear-phase response in the passband and a lower overshoot. The optimization takes 13 s to run.

B. Filter with Near-Linear-Phase Response and Minimal Peak Overshoot

As in the previous example, we minimize the peak overshoot and deviation from linear phase of a low-pass filter given the same magnitude specifications $\omega_p = 20$ rad/s, $\delta_p = 0.21$, $\omega_s = 30$ rad/s, and $\delta_s = 0.31$, except that we allow poles and zeros. We use a fourth-order elliptic filter as the initial guess. In the objective function, we weight the deviation from linear phase by 0.1 and deviation in peak overshoot by 1. All other weights are zero. We set $Q_{\max}$ to 10. The nonnegative objective function is reduced from 2.87 to 4.33×10^{-5}. Table II lists the initial and final poles and zeros. Fig. 3 plots the magnitude, phase, and step responses for the initial and final filters. Fig. 3 illustrates that the optimization procedure effectively trades off transition bandwidth in the magnitude response for more linear phase in the passband and a lower overshoot. The peak overshoot is reduced from 25% to 10%. The gradient of the objective function with respect to the poles $\{a_1, b_1, a_2, b_2\}$ is $\{-1.99, -0.81, -17.74, 2.11\} \times 10^{-5}$ and with respect to the zeros, $\{c_1, d_1, c_2, d_2\}$ is $\{2.07, 2.38, 0.55, 1.97\} \times 10^{-5}$. Since the second filter section is more sensitive to

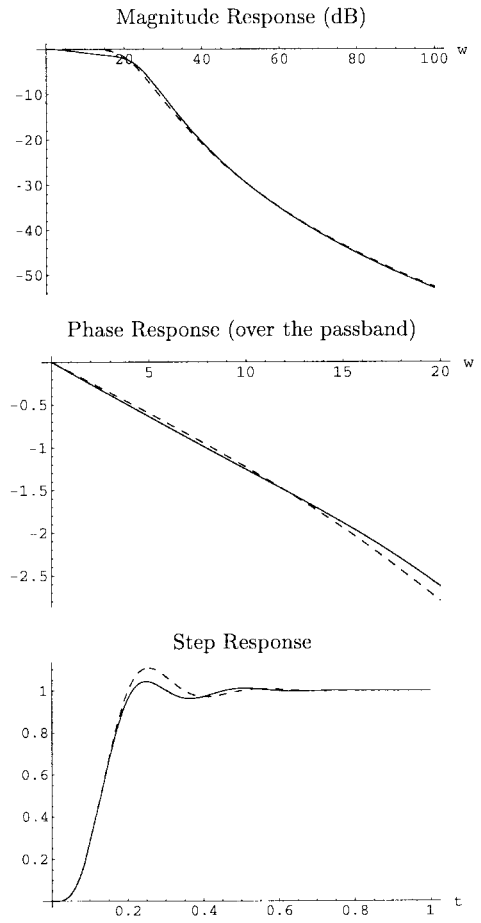


Fig. 2. Fourth-order low-pass filter with optimized phase and step responses. The specification of the magnitude response is $\omega_p = 20$ rad/s and $\delta_p = 0.21$ for the passband and $\omega_s = 30$ rad/s, and $\delta_s = 0.31$ for the stopband. The dashed lines represent the initial Butterworth filter, and the solid lines represent the filter optimized for linear phase response in the passband and for overshoot of the step response.

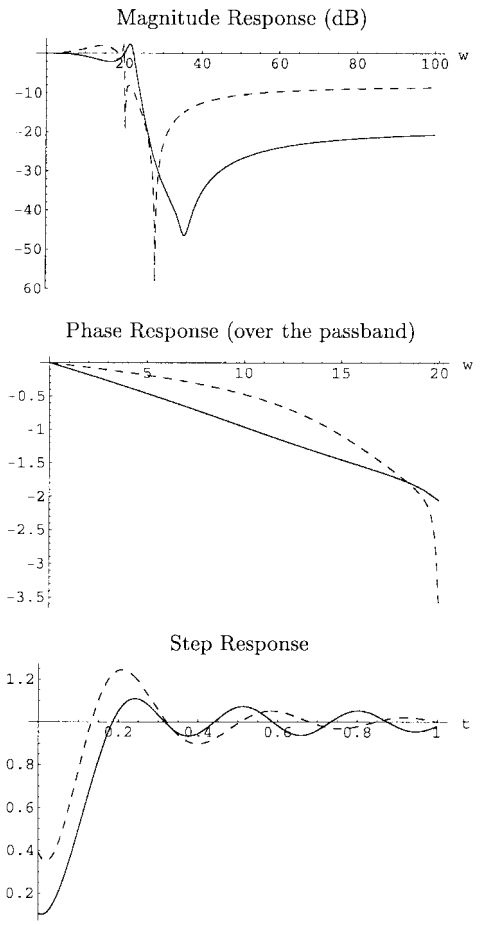


Fig. 3. Two fourth-order low-pass filters to meet the magnitude specifications $\omega_p = 20$ rad/s, $\delta_p = 0.21$, $\omega_s = 30$ rad/s, and $\delta_s = 0.31$. The initial filter is an elliptic filter (dashed lines) and the final filter is optimized for phase and step response (solid lines). We are trading linear-phase response over the passband and peak overshoot in the step response for magnitude response, while keeping the magnitude response within specification. For the optimization, we set the maximum quality factor $Q_{\max}$ to be 10. Even though the initial guess is infeasible because its maximum Q value is 61, the SQP procedure in MATLAB adjusted the initial guess to be a feasible solution.

perturbations in the pole locations, better components should be used for the second section. The optimization takes 20 s to run.

C. Filter Simultaneously Optimized for Four Criteria

We optimize three behavioral properties and one implementation property simultaneously. The specifications on the magnitude response are $\omega_p = 30$ rad/s, $\delta_p = 0.2$, $\omega_s = 50$ rad/s, and $\delta_s = 0.3$. In the objective function, we weight the deviation from linear phase by 1, ideal peak overshoot by 1, ideal filter quality by 0.5, and ideal stopband magnitude response by 1. All other weights are zero. We set $Q_{\max}$ to 10. The initial guess is a sixth-order elliptic filter. It was designed using a stricter stopband criterion than specified to illustrate the tradeoffs obtained by searching the infinite design space. The nonnegative objective function is reduced from 4.25 to 0.34. The peak overshoot is reduced from 20% to 15%. Table III lists the initial and final poles and zeros. Table III and Fig. 4 illustrate that the optimization procedure effectively trades off transition bandwidth and extra stopband attenuation for more linear phase in the passband, lower overshoot, and lower quality factors. The optimization takes 65 s to run.

TABLE II
QUALITY FACTORS AND POLE–ZERO LOCATIONS FOR THE TWO SECOND -ORDER SECTIONS OF THE INITIAL AND OPTIMIZED FILTERS

Q	Poles	Zeros
1.7	$-5.3553 \pm j16.9547$	$0 \pm j20.2479$
61.1	$-0.1636 \pm j19.9899$	$0 \pm j28.0184$

(a)

Q	Poles	Zeros
0.68	$-11.4343 \pm j10.5092$	$-3.4232 \pm j28.6856$
10	$-1.0926 \pm j21.8241$	$-1.2725 \pm j35.5476$

(b)

The filter was minimized for the deviation from linear phase in the passband and peak overshoot. Fig. 3 compares the behavior of the two filters.

VI. VALIDATION AND VERIFICATION OF THE AUTOMATED FRAMEWORK

We use the Mathematica symbolic mathematics environment to collect the formulas for objective measures of properties,

TABLE III
QUALITY FACTORS AND POLE-ZERO LOCATIONS FOR THE TWO SECOND-ORDER SECTIONS OF THE INITIAL AND OPTIMIZED FILTERS

Q	Poles	Zeros
0.63	$-16.821 \pm j12.881$	$0 \pm j37.082$
4.22	$-3.4232 \pm j28.686$	$0 \pm j110.17$
9.05	$-1.7224 \pm j31.139$	$0 \pm j45.114$

(a)

Q	Poles	Zeros
0.51	$-38.21 \pm j5.8056$	$-16.407 \pm j45.045$
1.57	$-13.30 \pm j39.507$	$-4.8726 \pm j52.405$
1.74	$-9.84 \pm j32.906$	$-916.20 \pm j109.78$

(b)

The filter was minimized for deviation from linear phase in the passband, peak overshoot, quality factors, and magnitude response in the stopband. Fig. 4 compares the behavior of the two filters.

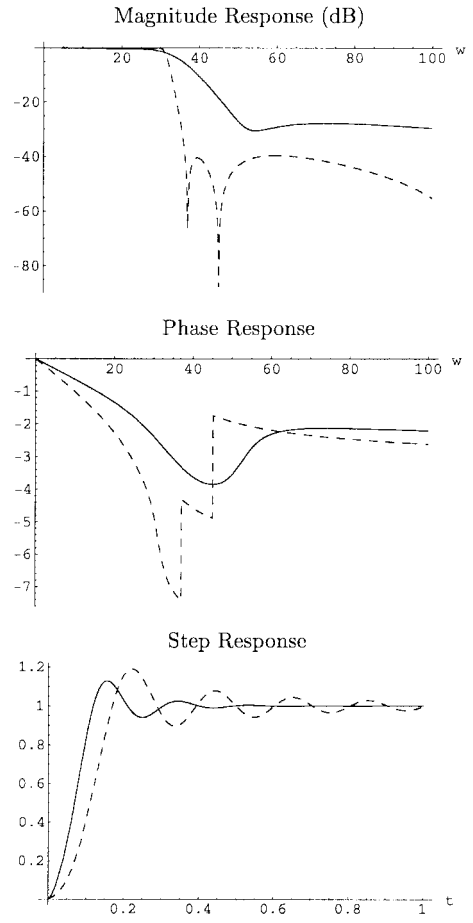


Fig. 4. Two fourth-order low-pass filters to meet the magnitude specifications $\omega_p = 20$ rad/s, $\delta_p = 0.20$, $\omega_s = 30$ rad/s, and $\delta_s = 0.30$. The initial filter is an elliptic filter (dashed lines), and the final filter is optimized for phase, step response magnitude response in the stopband and quality factor (solid lines).

distance measures, objective function, and constraints, and to automate the conversion of the formulas to working filter optimization programs in MATLAB. Section VI-A describes the steps in using Mathematica to verify the formulas we entered into it for the equations in Section III. It also introduces our approach to converting the objective function and constraints, as well as their gradients, into source code. Section VI-B discusses the automatic synthesis of complete MATLAB programs to optimize analog filter designs. Finally, Section VI-C summarizes the ways in which we validate the generated code.

A. Verification of Formulas and Code Generation

We have written functions in the Mathematica to return the formulas in Section III given the names of the free parameters of the pole–zero locations, the number of conjugate pole pairs, and the number of conjugate zero pairs (i.e., `a`, `b`, `c`, `d`, `n`, and `r`). We then use the symbolic mathematics environment to verify that we encoded the key equations properly as follows. For (1) and (2), we validated them by comparing their formulas to the absolute value and argument, respectively, applied directly to several example frequency responses $H(j\omega)$. We verified the formula for the partial fractions coefficients in (16) by comparing the formula to the partial fractions decomposition obtained by the symbolic mathematics environment. In validating the encoding of the step response in (17), we compared the formula with the inverse Laplace transform of s multiplied by several transfer functions.

Now that the formulas as entered into the symbolic mathematics environment have been verified, we encode the objective function given in (23). The next step is to compute the gradient of the objective function and the constraints so that the problem formulation can be passed onto optimization software. In this step, we discovered the need for an "inert" integral operator (one that does not try to evaluate the integrand) to represent equations such as (7). The reason is that we want the optimization software to compute the integral rather than the symbolic mathematics environment. Therefore, we introduce an inert integration operator called `integrate`. By using `integrate`, we were able to automate the calculation of gradients for the objective function and constraints by augmenting Mathematica's built-in derivative operator. The gradients are computed with respect to the free parameters (pole–zero locations).

Next, we convert the objective function and constraints and their gradients into source code. In order to generate efficient source code, we reuse the intermediate results of code that has already been generated. Before we can reuse computations, we must make sure that the computations are context-free, i.e., that they are not dependent on a parameter that can change value. For the objective function and constraints, all of the computations are context-free except for the integral calculations. The integral calculations can be performed by substituting the variable of integration with some unique variable. Now we can employ the equivalent of subexpression elimination by means of hash tables or some other method. Although we can generate C and Fortran code, we focus on generating MATLAB code to take advantage of its optimization toolbox.

TABLE IV
MATLAB FILES GENERATED BY OUR PROGRAMS

Filename	File Type	Purpose
stepresp.m	function	returns the value of the step response at time t (works for all filter orders)
filcost$n$.m	function	returns the value of objective function (a scalar) and values of the constraints (a vector)
costder$n$.m	function	returns the gradient of the objective function (a vector) and gradient of constraints (a matrix)
go$n$.m	script	runs numerical optimization to design "best" analog filter

Here, n is the number of conjugate pole pairs in the analog filter. All functions take the pole–zero locations as arguments.

B. Generating Working MATLAB Programs

The first step in generating MATLAB programs is to find the analogous operations in MATLAB. The MATLAB functions corresponding to the Mathematica functions `Cos`, `Log`, `Max`, `Min`, `Sign`, `Sin`, and `Tan` are the same, except that the MATLAB names are in lower case. Similarly, the constant `Pi` becomes pi, but the constant `E` becomes exp(1). `Arg` maps to angle. We map `ArcTan[x, y]`, which takes the sign of `x` and `y` into account, to angle $(x+j*y)$. Likewise, `ArcTan[z]` becomes angle $(1+z)$. We map symbolic definite integrals into numerical integrations via MATLAB's trapz function.

MATLAB's syntax is different from Mathematica's, but the conversion is possible by a direct mapping of characters. In delimiting functions, Mathematica uses square brackets whereas MATLAB uses parentheses. Unlike Mathematica, which performs its computations in terms of formulas, MATLAB is based on matrix–vector operations. In Mathematica, the multiplication, division, and power operators work in a pointwise fashion and are therefore mapped into MATLAB's $.*$, $./$, and $.\hat{}$ operators. The complete mapping of an expression in Mathematica to MATLAB is carried out by first generating a string representing the internal form of the Mathematica expression via `InputForm` and then performing the substitutions listed above via `StringReplace`.

Now that expressions in Mathematica can be mapped directly into MATLAB code, we can add the syntax for function definitions and create a MATLAB script that directs the design procedure. In the synthesis of MATLAB programs, our programs generate four files as shown in Table IV. The "go$n$.m" file directs the optimization procedure. First, it initializes several constant optimization parameters as global variables. Then, it calls the constrained nonlinear optimization routine *constr* in the MATLAB optimization toolbox [8] to minimize the objective function given by (23).

The *constr* function calls the filcost$n$ function to compute the objective function and constraints, and the costder$n$ function to compute the gradients. As stated previously, the *constr* routine is sequential quadratic programming method [5], and not a conjugate-gradient technique.

The designer may change the constant optimization parameters in the MATLAB code. The maximum quality factor $Q_{\max}$ is the `qmax1` variable in the "filcost$n$.m" file. The other constant parameters—weights of the objective function, magnitude response specifications, and maximum overshoot—are defined at the beginning of the "go$n$.m" file.

The "stepresp.m" file computes the step response. It is independent of the number of conjugate pole pairs n, whereas the other three files are not. In order for Mathematica to compute the gradient of the objective function, we unravel the product and summation terms in the objective function for fixed values of n and r. The effect on the generated MATLAB code is that the loops that would have depended on n and r have been unrolled (i.e., flattened). In the freely distributable release of the framework, we have generated the MATLAB programs for four poles and zero zeros, four poles and two zeros, four poles and four zeros, six poles and six zeros, and eight poles and eight zeros.

C. Validating the MATLAB Programs

This section discusses the verification of synthesized MATLAB programs. Because we generated the step response as a separate MATLAB file, we compare it directly to the step response generated in Mathematica for sampled values. MATLAB implements the step response according to (17), whereas Mathematica computes the step response by using the inverse Laplace transform. We compare the objective function generated in Mathematica directly with the objective function generated in MATLAB as it is also a separate file. The Mathematica and MATLAB versions agree.

MATLAB's *constr* SQP routine enables the checking of the symbolic form of the gradients. We expect strong agreement because symbolic differentiation by a symbolic mathematics environment is highly reliable. In numerically computing the gradients, the SQP routine uses finite difference techniques. The finite difference techniques only use the generated objective function, which we have already validated. If the analytic and finite difference gradients agree, then the symbolic form of the gradient is correct because we already know that the code generation is working properly. For the design examples in Section V, the maximum deviation in the components of the gradient is less than 0.06%. The symbolic form of the gradients are important for the stability of the SQP procedure. We found design examples in which MATLAB's *constr* routine diverged when not using the symbolic gradients but converged when using the symbolic gradients.

By validating the symbolic form of the gradients, we validate the analytic approximation of t_{peak} by (19) and (22). The analytic approximation is solely used to compute the gradient of the objective measure of the peak overshoot of a filter. We validate the accuracy of the analytic approximation by running 10 design examples requiring at least 40 iterations of the SQP procedure each. At each iteration, we have the SQP routine compare gradients. In all cases, the gradient of

the objective measure of the peak overshoot given by (19) and (22) never varied more than 0.06% of the numerically approximated value.

VII. Conclusion

We have developed a formal, extensible framework for optimizing multiple behavioral and implementation properties of analog filter designs. We have implemented the framework as a set of Mathematica programs that generate MATLAB programs to perform the simultaneous optimization of magnitude response, phase response, peak overshoot, and quality factors. In developing the framework, we derive a new analytic approximation for peak overshoot, which we validate using the MATLAB implementation. We demonstrate the framework by finding three new low-pass filter designs optimized for multiple criteria.

In the framework, both the algebraic derivations and programming tasks would be nearly impossible for a human to carry out correctly. By performing both processes together, we can validate that the assumptions in the algebraic derivations are legitimate and that the source code is generated properly. Furthermore, the algebraic abstraction empowers the researcher to create new filter design programs by simply redefining the objective function—our software will take care of recomputing the derivatives and regenerating the source code.

Appendix
Derivation of the Step Response

For an analog filter with n conjugate pole pairs and r conjugate zero pairs, the transfer function of the impulse response is

$$\left[\prod_{m=1}^{n}\frac{a_m^2+b_m^2}{s^2-2a_ms+a_m^2+b_m^2}\right]\left[\prod_{l=1}^{r}\frac{s^2-2c_ls+c_l^2+d_l^2}{c_l^2+d_l^2}\right].$$

In the continuous time domain, the step response is

$$h_{\text{step}}(t)=h(t)*u(t).$$

The Laplace transform of $h_{\text{step}}(t)$ is

$$H_{\text{step}}(s)=\frac{1}{s}\left[\prod_{m=1}^{n}\frac{a_m^2+b_m^2}{s^2-2a_ms+a_m^2+b_m^2}\right]\cdot\left[\prod_{l=1}^{r}\frac{s^2-2c_ls+c_l^2+d_l^2}{c_l^2+d_l^2}\right]. \tag{26}$$

Since the order of the denominator will always be greater than the order of the numerator even for $n=r$, we can express the transfer function as a partial fractions expansion

$$H_{\text{step}}(s)=\frac{A}{s}+\sum_{k=1}^{n}\frac{C_ks+D_k}{s^2-2a_ks+a_k^2+b_k^2}.$$

Splitting up (26) in terms of individual poles and zeros

$$H_{\text{step}}(s)=\frac{1}{s}\left[\prod_{m=1}^{n}\frac{a_m^2+b_m^2}{(s-(a_m+jb_m))(s-(a_m-jb_m))}\right]\cdot\left[\prod_{l=1}^{r}\frac{(s-(c_l+jd_l))(s-(c_l-jd_l))}{c_l^2+d_l^2}\right].$$

The partial fractions expansion may be expressed as

$$\frac{A}{s}+\sum_{k=1}^{n}\left[\frac{B_k}{s-(a_k+jb_k)}+\frac{B_k^*}{s-(a_k-jb_k)}\right]$$

where the second-order sections have been split up into first-order sections. Thus

$$C_k=2|B_k|\cos(\angle B_k) \tag{27}$$

$$D_k=-2|B_k|(a_k\cos(\angle B_k)+b_k\sin(\angle B_k)) \tag{28}$$

where the partial fraction coefficients A and B_k can be obtained as follows:

$$A=[H_{\text{step}}(s)s]_{s=0} \tag{29}$$

$$B_k=[H_{\text{step}}(s)(s-(a_k+jb_k))]_{s=a_k+jb_k}=|B_k|e^{j\angle B_k}. \tag{30}$$

$$B_k=\frac{1}{2jb_k(a_k+jb_k)}\frac{\prod_{m=1}^{n}a_m^2+b_m^2}{\prod_{l=1}^{r}c_l^2+d_l^2}\cdot\frac{\prod_{l=1}^{r}[(a_k-c_l)+j(b_k-d_l)][(a_k-c_l)+j(b_k+d_l)]}{\prod_{\substack{m=1\\m\neq k}}^{n}[(a_k-a_m)+j(b_k-b_m)][(a_k-a_m)+j(b_k+b_m)]} \tag{31}$$

$$|B_k|=\frac{1}{-2b_k\sqrt{a_k^2+b_k^2}}\frac{\prod_{m=1}^{n}a_m^2+b_m^2}{\prod_{l=1}^{r}c_l^2+d_l^2}\cdot\frac{\prod_{l=1}^{r}\sqrt{[(a_k-c_l)^2+(b_k-d_l)^2][(a_k-c_l)^2+(b_k+d_l)^2]}}{\prod_{\substack{m=1\\m\neq k}}^{n}\sqrt{[(a_k-a_m)^2+(b_k-b_m)^2][(a_k-a_m)^2+(b_k+b_m)^2]}} \tag{31a}$$

Assuming that the analog filter is a low-pass prototype filter, the dc value is 1, so

$$A = 1.$$

By substituting $s = a_k + jb_k$ into (30), we have (31), shown at the bottom of the previous page. Next, we compute the magnitude and phase for B_k based on (31) for use in (27) and (28). The magnitude of B_k is shown in (31a), at the bottom of the previous page, where the negative sign before b_k is because b_k is taken to be negative in our formulation. Similarly, the unwrapped phase of B_k is

$$\begin{aligned}\angle B_k =& \frac{\pi}{2} - \sum_{\substack{m=1 \\ m\neq k}}^{n} \\ & \cdot \left[\arctan\left(\frac{b_k - b_m}{a_k - a_m}\right) + \arctan\left(\frac{b_k + b_m}{a_k - a_m}\right)\right] \\ & + \sum_{l=1}^{r}\left[\arctan\left(\frac{b_k - d_l}{a_k - c_l}\right) + \arctan\left(\frac{b_k + d_l}{b_k - c_l}\right)\right] \\ & - \arctan\left(\frac{b_k}{a_k}\right).\end{aligned}$$

C_k and D_k can be computed from $|B_k|$ and $\angle B_k$. Thus, $H_{\text{step}}(s)$ has the form

$$H_{\text{step}}(s) = \frac{1}{s} + \sum_{k=1}^{n} \frac{C_k s + D_k}{s^2 - 2a_k s + a_k^2 + b_k^2}$$

which can be rewritten as

$$H_{\text{step}}(s) = \frac{1}{s} + \sum_{k=1}^{n}\left[\frac{C_k s}{(s-a_k)^2 + b_k^2} + \frac{D_k}{(s-a_k)^2 + b_k^2}\right].$$

Therefore, taking their inverse Laplace transform, we obtain the step response

$$\begin{aligned}&h_{\text{step}}(t) \\ &\quad = 1 + \sum_{k=1}^{n} e^{a_k t}\left[C_k \cos(b_k t) + \left(\frac{D_k + C_k a_k}{b_k}\right)\sin(b_k t)\right]\end{aligned} \tag{32}$$

provided that $b_k \neq 0$. By substituting (27) and (28) into (32), we obtain

$$h_{\text{step}}(t) = 1 + 2\sum_{k=1}^{n} |B_k| e^{a_k t} \cos(b_k t + \angle B_k).$$

ACKNOWLEDGMENT

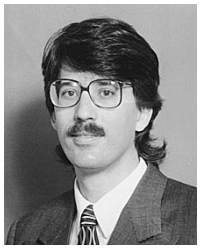
The authors would like to thank the following for their valuable suggestions: D. R. Firth of Precision Filters, Ithaca, NY; Prof. E. A. Lee of the University of California at Berkeley; Prof. M. D. Lutovac and Prof. D. V. Tosic of the University of Belgrade, Serbia, Yugoslavia; and K. D. White of Diva Communications, Berkeley, CA. This entire joint optimization approach for multiple filter characteristics had its roots in a September 1993 discussion between B. Evans and D. Firth.

REFERENCES

[1] T. Saramaki and K.-P. Estola, "Design of linear-phase partly digital anti-aliasing filters," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Tampa, FL, Mar. 1985.
[2] M. Lutovac, D. V. Tosic, and B. L. Evans, "Algorithm for symbolic design of elliptic filters," in *Int. Workshop Symbolic Methods and Applications to Circuit Design*, Leuven, Belgium, Oct. 1996, pp. 248–251.
[3] S. Lawson and T. Wicks, "Improved design of digital filters satisfying a combined loss and delay specification," in *Inst. Elect. Eng. Proceedings G*, June 1993, vol. 140, pp. 223–229.
[4] S. Wright, "Convergence of SQP-like methods for constrained optimization," *SIAM J. Contr. Optim.*, vol. 27, pp. 13–26, Jan. 1989.
[5] K. Schittkowski, "NLPQL: A Fortran subroutine solving constrained nonlinear programming problems," *Annals Oper. Res.*, vol. 5, no. 1/2/3/4, pp. 485–500, 1986.
[6] S. Wolfram, *The Mathematica Book*, 3rd ed. Champaign, IL: Wolfram Media Inc., 1996.
[7] *Matlab 5 User's Guide.* Natick, MA: The MathWorks Inc., 1997.
[8] A. Grace, *Optimization Toolbox.* Natick, MA: The MathWorks, Inc., 1992.

Niranjan Damera-Venkata (S'99) received the B.S.E.E. degree from the University of Madras, India, in July 1997, and the M.S.E.E. degree from The University of Texas (UT) at Austin in May 1999. He is currently working toward the Ph.D. in electrical engineering at UT at Austin.

He was awarded the 1998–1999 Texas Telecommunications Engineering Consortium Graduate Fellowship from UT. His research involves phase-locked loop design, optimal analog and digital filter design, and digital image halftoning.



Brian L. Evans (S'88–M'93–SM'97) received the B.S.E.E.C.S. degree from the Rose–Hulman Institute of Technology, Terre Haute, IN, in May 1987, and the M.S.E.E. and Ph.D. degrees from the Georgia Institute of Technology, Atlanta, GA, in December 1988 and September 1993, respectively.

From 1993 to 1996, he was a Postdoctoral Researcher at the University of California at Berkeley, where he worked with the Ptolemy Project, a research project and software environment focused on design methodology for signal processing, communications, and controls systems. In addition, he has played a key role in the development and release of six other computer-aided design frameworks. He is the primary architect of the *Signals and Systems Pack* for Mathematica (1995). He is currently an Assistant Professor in the Department of Electrical and Computer Engineering, The University of Texas (UT) at Austin. He is the Director of the Embedded Signal Processing Laboratory within the Center for Vision and Image Sciences. His research interests include real-time embedded systems, signal, image, and video processing systems, system-level design, symbolic computation, and filter design. At UT, he developed and currently teaches multidimensional digital signal processing, embedded software systems, and a real-time digital signal processing laboratory, both to respond to the needs of industry and recruit students for his research group.

Dr. Evans is an Associate Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING and a member of the Design and Implementation of the Signal Processing Systems Technical Committee of the IEEE Signal Processing Society. He was the recipient of a 1997 National Science Foundation CAREER Award.