Solving the Selesnick-Burrus Filter Design Equations

John B. Little

Department of Mathematics and Computer Science

College of the Holy Cross

Worcester, MA 01610

email: little@mathcs.holycross.edu

AMS Sectional Meeting

Stevens Institute of Technology

April 29, 2001

Outline of Talk

We will discuss an application of computational commutative algebra to a question in *signal processing*.

1. Background on digital filters

2. A filter design problem

3. Results

4. Methods

Digital Signals

A *digital* signal is a quantized function of a *discrete* variable.

- only have signal values at $n\Delta t$, n an integer, Δt called the *sampling interval*

- signal values are elements of a discrete set.

For simplicity, we ignore quantization effects and study *discrete-time, real-valued* signals.

Can write a signal as a real-valued function of an integer variable, $x[n]$, or equivalently, a list of real numbers indexed by the integers:

$$\cdots, x[-2], x[-1], x[0], x[1], x[2], \cdots$$

Filters

Signal processing operations can be described mathematically by means of functions Γ:

$$\Gamma : \{\text{signals}\} \to \{\text{signals}\}.$$

We consider only operations with the following properties

1. (Linearity) If $z[n] = cx[n] + y[n]$, then

$$\Gamma(z[n]) = c\Gamma(x[n]) + \Gamma(y[n])$$

2. (Shift-invariance) If $y[n] = x[n+k]$ for all n, k fixed, and the outputs are $w[n] = \Gamma(x[n])$, $z[n] = \Gamma(y[n])$, then

$$z[n] = \Gamma(y[n]) = \Gamma(x[n+k]) = w[n+k]$$

A Γ satisfying both of these properties is said to be a *filter*.

Example – A "Moving Average" Filter

For any input signal $x[n]$, define

$$\begin{aligned} y[n] &= \Gamma(x[n]) \\ &= (x[n] + x[n-1] + x[n-2])/3 \end{aligned}$$

If $x[n] = \cos(n\pi/2)$,

$$\cdots, 1, 0, -1, 0, 1, 0, -1, 0, 1, \cdots$$

for instance, then $\Gamma(x[n]) =$

$$\cdots, 0, 1/3, 0, -1/3, 0, 1/3, 0, -1/3, 0, \cdots$$

It is easy to see that this satisfies the linearity and shift-invariance properties, so this defines an filter.

Impulse Response of a Filter

A nice result about filters: Let $\delta[n]$ be

$$\cdots, 0, 0, 1, 0, 0, \cdots$$

(1 at time $n = 0$) – called a *unit impulse* input. $\Gamma(\delta[n])$ is called the *impulse response* of the filter.

If we know $h[n] = \Gamma(\delta[n])$, then we also know the output of the filter on any other signal $x[n]$. Reason: We can write

$$x[n] = \sum_{k=-\infty}^{\infty} x[k]\delta[n-k]$$

Hence by linearity

$$\Gamma(x[n]) = \sum_{k=-\infty}^{\infty} x[k]\Gamma(\delta[n-k]).$$

Then by shift-invariance

$$\Gamma(x[n]) = \sum_{k=-\infty}^{\infty} x[k]h[n-k].$$

The Transfer Function

Package $x[n]$ by its "z-transform", the series

$$X(z) = \sum_{n=-\infty}^{\infty} x[n]z^{-n}$$

Do same for the impulse response of a filter $h[n]$ and the output $y[n]$ for input $x[n]$:

$$H(z) = \sum_{n=-\infty}^{\infty} h[n]z^{-n}, \ Y(z) = \sum_{n=-\infty}^{\infty} y[n]z^{-n}$$

Coefficient of z^{-n} in the product $H(z)X(z)$ is:

$$\sum_{k=-\infty}^{\infty} h[k]x[n-k],$$

same as $y[n]$ (!)

That is: $Y(z) = H(z)X(z)$. $H(z)$ is called the *transfer function*.

Filter Design

Note: $H(e^{i\omega})$ is the (discrete) Fourier transform of h.

Goal is to design filters to perform various operations on signals, by designing $H(z)$.

Typical problems: design a *low-pass* or *high pass* filters to meet certain specifications.

- Low-pass filters: remove high-frequency components of signals – typically *smooth out* or *blur* signals (can be used to remove high-frequency "noise")

- High-pass filters: remove low-frequency components – typically focus on fine details, rapid changes (can be used to detect features)

Selesnick-Burrus Design Paradigm

reference: Selesnick and Burrus, *IEEE Trans. Circuits and Systems II*, v. 45(1), January 1998.

Given the low-pass filter $H(z)$, consider

- *magnitude response:* $F(\omega) = |H(e^{i\omega})|^2$
- *group delay:* $G(\omega) = \frac{d}{d\omega}\Big(\text{arg}(H(e^{i\omega}))\Big)$

We would like to, simultaneously,

1. make the passband *flat*,
2. make the stopband *flat*, and
3. make the group delay $G(\omega)$ *flat* across the passband.

Specifications

Provided $N \geq K + L + M + 1$, can find $H(z)$ satisfying:

- $h[0] = 1$ and $F^{(2i)}(0) = 0$, for $i = 1, \ldots M$. (*flatness in passband*).

- $z = -1$ a zero of $H(z)$ of order K (*flatness in stopband* $\omega = \pi$),

- $G^{(2j)}(0) = 0$, for $j = 1, \ldots, L$ (*flat group delay*)

(F and G are even functions, so odd-order derivatives at 0 automatically zero.)

The System of Equations

- Computing derivatives of F and G explicitly give *polynomial equations* in the filter coefficients $h[n]$, or in their *moments*

$$m[k] = \sum_{n=0}^{N-1} n^k h[n], k \geq 0 \tag{1}$$

- $h[n]$ and $m[k]$ are related by the linear equations (1), so can use either set of variables.

- Moments are somewhat more convenient.

- Need to understand the solutions of a rather large, quite messy system of polynomial equations in $m[k]$.

Form of The Equations

- Flatness equations (quadratic in the $m[i]$):

  $$\binom{2i}{i}m[i]^2 + 2\sum_{\ell=0}^{i-1}\binom{2i}{\ell}(-1)^{i+\ell}m[\ell]m[2i-\ell]$$
  $$= 0,\ i = 1, \ldots, M.$$

  $$\sum_{\ell=0}^{j}(1-\frac{2\ell}{2j+1})\binom{2j+1}{\ell}(-1)^{\ell}m[\ell]m[2j+1-\ell]$$
  $$= 0,\ j = 1, \ldots, L.$$

- Normalization: $m[0] = 1$

- Zeros at $z = -1$: remainder of $H(z)$ on division by $(1+z)^K$ is zero – K linear equations on $m[i]$

- Additional equations: Can express $m[k]$ for $k \geq N$ in terms of $m[0], \ldots, m[N-1]$ by solving linear equations, and substitute.

To Summarize

Get $L + M + K = N - 1$ equations in $N - 1$ variables $m[1], \ldots, m[N-1]$, and we expect only finitely many solutions. (Mainly interested in real ones.)

Notation:

- $I_{K,L,M}$ is the ideal generated by all these,
- $J_{L,M}$ is the ideal generated by the quadratics (doesn't depend on K)

“Easy” and “Hard” Cases

Selesnick and Burris:

- (”Region I”) The *easier* cases are
$$\left\lfloor \frac{M-1}{2} \right\rfloor \le L \le M$$
All variables except $m[1]$ can be eliminated using linear algebra techniques.

- (”Region II”) The *harder* cases are
$$0 \le L \le \left\lfloor \frac{M-1}{2} \right\rfloor - 1$$

Example: if $L = 0$, there are 2^M solutions of the design equations in these cases(!) (One way to see this: the generators for $I_{K,0,M}$ are already “almost” a Gröbner basis for $I_{K,0,M}$ with respect to a *grevlex* order with $m[N-1] > m[N-2] > \cdots > m[1]$.)

Some Results

$$\#\text{sols} = \dim \mathbf{C}[m[1], \ldots, m[N-1]]/I_{K,L,M}$$

in "Region II"− doesn't depend on $K \geq 1$. $L =$ 0 column omitted: 2^M.

M/L	1	2	3	4	5
5	16				
6	26				
7	48	24			
8	78	38			
9	152	66	32		
10		112	50		
11			86	40	
12			142	62	
13				106	48

Cases $M \leq 7$ check with Selesnick-Burrus. $M \geq$ 8 new; blank space below Region II boundary means (?)

Methods Used

- For the cases M odd, $L = (M-3)/2$ at the "corners" of the Region II boundary, *surprisingly(?)*, a *lex* GB is not too hard to compute if the variables are ordered $m[N-1] > m[N-2] > \cdots > m[1]$. The dimension reported is then the degree of the univariate polynomial in $m[1]$, and this can be solved by numerical root-finding.

- For some cases with very small L, a *grevlex* GB and eigenvalue methods are effective.

- Best general method I have found is a "hybrid" approach

The "Best" Solution Method

- The quadratic flatness conditions are homogeneous if weight($m[i]$) $= i$, and equations have a very special form.

- In fact, the variety defined by the quadratics, $V(J_{L,M})$, is a special type of *rational scroll* – it has a parametrization via rational (monomial) curves with "directrix" a linear variety.

- It is always easy to find this parametrization (a *lex* GB of $J_{L,M}$ with $m[N-1] > m[N-2] > \cdots > m[1]$ is easy to compute and shows it clearly).

- Substituting from this parametrization reduces the number of variables considerably to $M - L$.

"Best" Solution Method, cont.

- The equations left have quite low degrees in all variables except $m[1]$.

- By *either* sparse resultants, or pairwise ordinary (Sylvester) resultants, can eliminate all the variables except $m[1]$ and get a univariate polynomial in the ideal $I_{K,L,M}$.

- In cases I have looked at this way, the actual computation is fast, but I don't have an algorithm for forming the pairwise ordinary (Sylvester) resultants (did it "by hand" trial and and error), and there are always extraneous factors in the result.

- Sparse resultants are relatively slower, and also have large extraneous factors.

Things That Make These Systems Hard

- High degrees of univariate polynomials

- For large M, L, the univariate polynomial has *gigantic* coefficients.

- A *BIG* numerical analysis problem. Need to use very many decimal digits in floating-point computation to get accurate roots.

- In fact this is the current "bottle-neck" in the cases I have worked out.

Conclusions

CAS's like Maple, etc. where the number of digits in floating point computation can be set are very convenient for this! (Arbitrary precision arithmetic systems would also be well-adapted, of course.)

These systems of equations are a real challenge to the "state of the art" in polynomial system solving. It's possible to "nibble around the edges" of Region II a bit, but solving them requires a lot of computer time. A "new idea" is probably needed for a general solution method.