7.10 Problems

7.1 Show that the matrices Q, S defined in Eqs. (7.3.13) and (7.3.30) satisfy the orthogonality property $Q^T S = 0$. Assuming that the weighting diagonal matrix W has positive diagonal entries, argue that the $N \times N$ matrix $A = [W^{-1/2}Q, W^{1/2}S]$ is non-singular. Using this fact, prove the projection matrix property (7.3.31). [*Hint:* work with $A(A^TA)^{-1}A^T$.]

7.2 Show that the Euler-Lagrange equation for the variational problem (7.6.1) is:

$$\dddot{x}(t) = \lambda^{-1} \sum_{n=0}^{N-1} \sum_{i=1}^{m_n} w_{ni}(y_{ni} - x(t_n))\,\delta(t - t_n)$$

and show that it is equivalent to

$$\dddot{x}(t) = \lambda^{-1} \sum_{n=0}^{N-1} \bar{w}_n(\bar{y}_n - x(t_n))\,\delta(t - t_n)$$

where $\bar{w}_n, \bar{y}_n$ were defined in (7.6.2). This is an alternative way to establish the equivalence of the variational problems (7.6.1) and (7.6.3).

7.3 First prove the following Fourier transform pair:

$$\exp(-b|t|) \;\longleftrightarrow\; \frac{2b}{b^2 + \omega^2}$$

where b is any complex number with $\mathrm{Re}(b) > 0$. Then, use it to prove that Eqs. (7.7.4) and (7.7.5) are a Fourier transform pair. Show the same for the pair in Eq. (7.7.8).

8

Whittaker-Henderson Smoothing

8.1 Whittaker-Henderson Smoothing Methods

Whittaker-Henderson smoothing is a discrete-time version of spline smoothing for equally spaced data. Some of the original papers by Bohlmann, Whittaker, Henderson and others,[†] and their applications to trend extraction in the actuarial sciences, physical sciences, and business and finance, are given in [405–438], including Hodrick-Prescott filters in finance [439–467], and recent realizations in terms of the ℓ_1 norm [468–478], as well as extensions to seasonal data [622–625,636,638]. The performance index was defined in Eq. (7.1.2),

$$\boxed{\mathcal{J} = \sum_{n=0}^{N-1} w_n \,|\, y_n - x_n \,|^2 + \lambda \sum_{n=s}^{N-1} |\, \nabla^s x_n \,|^2 = \min} \tag{8.1.1}$$

where $\nabla^s x_n$ represents the backward-difference operator $\nabla x_n = x_n - x_{n-1}$ applied s times. We encountered this operation in Sec. 4.2 on minimum-R_s Henderson filters. The corresponding difference filter and its impulse response are

$$\begin{aligned} D_s(z) &= (1 - z^{-1})^s \\ d_s(k) &= (-1)^k \binom{s}{k}, \quad 0 \le k \le s \end{aligned} \tag{8.1.2}$$

For example, we have for $s = 1, 2, 3$,

$$\mathbf{d}_1 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \quad \mathbf{d}_2 = \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix}, \quad \mathbf{d}_3 = \begin{bmatrix} 1 \\ -3 \\ 3 \\ -1 \end{bmatrix}$$

Because $D_s(z)$ is a highpass filter, the performance index attempts, in its second term, to minimize the spectral energy of x_n at the high frequency end, while attempting to interpolate the noisy observations with the first term. The result is a lowpass,

[†] Bohlmann considered the case $s = 1$, Whittaker and Henderson, $s = 3$, and Hodrick-Prescott, $s = 2$.

smoothing, operation. In fact, the filter $D_s(z)$ may be replaced by any other (causal) FIR highpass filter $D(z)$, or d_n in the time domain, with a similar result. Thus, a more general version of (8.1.1) would be:

$$\mathcal{J} = \sum_{n=0}^{N-1} w_n |y_n - x_n|^2 + \lambda \sum_{n=s}^{N-1} |d_n * x_n|^2 = \min \tag{8.1.3}$$

where $d_n * x_n$ denotes convolution and s is the filter order, that is, we assume that the impulse response is $d_n = [d_0, d_1, \ldots, d_s]$. The criteria (8.1.1) and (8.1.3) are examples of the method of *regularization*, which we discuss in more general terms in Sec. 8.6 and for ill-conditioned linear systems in particular in Sec. 15.10.

The summation limits of the second terms in Eqs. (8.1.1) and (8.1.3) restrict the convolutional operations to their steady-state range. For example, for a length-N causal input $\{x_0, x_1, \ldots, x_{N-1}\}$, the s-difference filter has the full convolutional output:

$$g_n = \nabla^s x_n = \sum_{k=\max(0,n-N+1)}^{\min(s,n)} d_s(k)\, x_{n-k}\,, \quad 0 \le n \le N-1+s \tag{8.1.4}$$

and the steady-state output (assuming $N > s$):

$$g_n = \nabla^s x_n = \sum_{k=0}^{s} d_s(k)\, x_{n-k}\,, \quad s \le n \le N-1 \tag{8.1.5}$$

Similarly, we have in the more general case,

$$\begin{aligned} g_n &= d_n * x_n = \sum_{k=\max(0,n-N+1)}^{\min(s,n)} d_k\, x_{n-k}\,, \quad 0 \le n \le N-1+s \\ g_n &= d_n * x_n = \sum_{k=0}^{s} d_k\, x_{n-k}\,, \quad s \le n \le N-1 \end{aligned} \tag{8.1.6}$$

In Sec. 4.2 we worked with the full convolutional form (8.1.4) and implemented it in a matrix form using the convolution matrix. We recall that the MATLAB functions `binom` and `diffmat` can be used to compute the impulse response $d_s(k)$ and the corresponding $(N+s)\times N$ full convolutional matrix D_s.

The filtering operation $g_n = \nabla^s x_n$, $0 \le n \le N-1+s$, can be expressed vectorially as $\mathbf{g} = D_s\mathbf{x}$, where $\mathbf{x}$ is the N-dimensional input vector $\mathbf{x} = [x_0, x_1, \ldots, x_{N-1}]^T$, and $\mathbf{g} = [g_0, g_1, \ldots, g_{N-1+s}]^T$, the $(N+s)$-dimensional output vector. Similarly, the operation $g_n = d_n * x_n$ can be expressed as $\mathbf{g} = D_{\text{full}}\mathbf{x}$, where the $(N+s)\times N$ full convolution matrix can be constructed using `convmat`—the sparse version of `convmtx`,

```
Dfull = convmat(d,N);      % sparse full convolution matrix
```

where $\mathbf{d} = [d_0, d_1, \ldots, d_s]^T$. The steady-state versions of the full convolution matrices are obtained by extracting their middle $N-s$ rows, and therefore, they have dimension $(N-s)\times N$. For example, we have for $N = 5$ and $s = 2$, with $\mathbf{d} = [d_0, d_1, d_2]^T$,

$$D_{\text{full}} = \left[\begin{array}{ccccc} d_0 & 0 & 0 & 0 & 0 \\ d_1 & d_0 & 0 & 0 & 0 \\ \hline d_2 & d_1 & d_0 & 0 & 0 \\ 0 & d_2 & d_1 & d_0 & 0 \\ 0 & 0 & d_2 & d_1 & d_0 \\ \hline 0 & 0 & 0 & d_2 & d_1 \\ 0 & 0 & 0 & 0 & d_2 \end{array}\right] \quad \Rightarrow \quad D = \left[\begin{array}{ccccc} d_2 & d_1 & d_0 & 0 & 0 \\ 0 & d_2 & d_1 & d_0 & 0 \\ 0 & 0 & d_2 & d_1 & d_0 \end{array}\right] = \left[\begin{array}{ccc} d_2 & 0 & 0 \\ d_1 & d_2 & 0 \\ d_0 & d_1 & d_2 \\ 0 & d_0 & d_1 \\ 0 & 0 & d_0 \end{array}\right]^T$$

The last expression shows that the steady matrix can also be viewed as the transposed of the convolution matrix of the reversed filter with $N-s$ columns. Thus, in MATLAB two possible ways of constructing D are:

```
1)  Dfull = convmat(d,N);   D = Dfull(s+1:N,:)
2)  D = convmat(flip(d),N-s)';
```
(8.1.7)

For the s-difference filter, we can use the equivalent (sparse) constructions:

```
1)  Dfull = diffmat(s,N);   D = Dfull(s+1:N,:)
2)  D = (-1)^s * diffmat(s,N-s)';
3)  D = diff(speye(N),s);
```
(8.1.8)

where the second method is valid because the reversed binomial filter is $(-1)^s$ times the unreversed one. The third method is the fastest [428], but does not generalize to an arbitrary filter $\mathbf{d}$. As an example, we have for $N = 7$, $s = 2$, and $\mathbf{d} = [1, -2, 1]^T$:

$$D = \left[\begin{array}{ccccccc} 1 & -2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & -2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & -2 & 1 \end{array}\right]$$

The corresponding steady-state output vector $\mathbf{g} = [g_s, g_{s+1}, \ldots, g_{N-1}]^T$ is given by $\mathbf{g} = D\mathbf{x}$, with squared norm,

$$\sum_{n=s}^{N-1} [d_n * x_n]^2 = \sum_{n=s}^{N-1} g_n^2 = \mathbf{g}^T\mathbf{g} = \mathbf{x}^T(D^TD)\mathbf{x}$$

Therefore, the performance index (8.1.1) or (8.1.3) can be written compactly as:

$$\mathcal{J} = (\mathbf{y}-\mathbf{x})^T W(\mathbf{y}-\mathbf{x}) + \lambda \mathbf{x}^T(D^TD)\mathbf{x} = \min \tag{8.1.9}$$

where W is the diagonal matrix of the weights, $W = \text{diag}([w_0, w_1, \ldots, w_{N-1}])$. The optimum solution is obtained by setting the gradient with respect to $\mathbf{x}$ to zero,

$$\frac{\partial \mathcal{J}}{\partial \mathbf{x}} = -2W(\mathbf{y}-\mathbf{x}) + 2\lambda(D^TD)\mathbf{x} = 0 \quad \Rightarrow \quad (W + \lambda D^TD)\mathbf{x} = W\mathbf{y}$$

with solution, which may be regarded as the estimate of $\mathbf{x}$ in the signal model $\mathbf{y} = \mathbf{x} + \mathbf{v}$,

$$\boxed{\hat{\mathbf{x}} = (W + \lambda D^T D)^{-1} W \mathbf{y}} \tag{8.1.10}$$

The matrix D plays the same role as the matrix Q^T in the spline smoothing case, but for equally-spaced data. As was the case in Sec. 4.2, the matrix $D^T D$ is essentially equivalent to the $(2s)$-differencing operator ∇^{2s}, after ignoring the first s and last s rows. For example, we have for $N = 7$ and $s = 2$,

$$D^T D = \left[\begin{array}{rrrrrrr} 1 & -2 & 1 & 0 & 0 & 0 & 0 \\ -2 & 5 & -4 & 1 & 0 & 0 & 0 \\ \hline 1 & -4 & 6 & -4 & 1 & 0 & 0 \\ 0 & 1 & -4 & 6 & -4 & 1 & 0 \\ 0 & 0 & 1 & -4 & 6 & -4 & 1 \\ \hline 0 & 0 & 0 & 1 & -4 & 5 & -2 \\ 0 & 0 & 0 & 0 & 1 & -2 & 1 \end{array}\right]$$

where we recognize the expansion coefficients $(1-z^{-1})^4 = 1 - 4z^{-1} + 6z^{-2} - 4z^{-3} + z^{-4}$.

The $N \times N$ matrix $(W + \lambda D^T D)$ is sparse and banded with bandwidth $2s + 1$, and therefore, MATLAB solves Eq. (8.1.10) very efficiently by default (as long as it is implemented by the backslash operator). The function `whsm` implements Eq. (8.1.10):

```
x = whsm(y,lambda,s,w);      % Whittaker-Henderson smoothing
```

where method (2) is used internally to compute D, and `w` is the vector of weights, which defaults to unity. The function `whgen` is the generalized version that uses an arbitrary highpass filter $\mathbf{d}$, whose steady convolution matrix D is also computed by method (2):

```
x = whgen(y,lambda,d,w);     % generalized Whittaker-Henderson smoothing
```

Denoting the "hat" filtering matrix $H = (W + \lambda D^T D)^{-1} W$, and defining the error $\mathbf{e} = \mathbf{y} - \hat{\mathbf{x}} = (I - H)\mathbf{y}$, we may define a generalized cross-validation criterion for determining the smoothing parameter λ, which is analogous to that of Eq. (7.5.5):

$$\mathrm{GCV}(\lambda) = \frac{\mathbf{e}^T W \mathbf{e}}{\left[\mathrm{tr}(I - H)\right]^2} = \min \tag{8.1.11}$$

The function `whgcv` calculates it at any vector of λ's and finds the corresponding optimum:

```
[gcv,lopt] = whgcv(y,la,s,w);     % Whittaker-Henderson GCV evaluation
```

The GCV criterion should be used with some caution because it suffers from the same problem, as in the spline case, of typically underestimating the proper value of λ.

The Whittaker-Henderson method was compared to the local polynomial and minimum roughness filters in Examples 3.9.2 and 4.2.1. Some additional examples are discussed below.

Example 8.1.1: *NIST ENSO data.* We apply the Whittaker-Henderson (WH) smoothing method to the ENSO data which are another benchmark example in the NIST Statistical Reference Dataset Archives, and original reference [1240]. The data file `ENSO.dat` is available online from the NIST web sites:

`http://www.itl.nist.gov/div898/strd/nls/nls_main.shtml`
`http://www.itl.nist.gov/div898/strd/nls/data/enso.shtml`

The data represent the monthly averaged atmospheric pressure differences between Easter Island and Darwin, Australia. In the nonlinear NIST fit, the data are fitted to three sinusoids of unknown amplitudes and frequencies, except that one of the sinusoids is kept at the annual frequency. There are three significant cycles at 12, 26, and 44 months.

The upper-left graph of Fig. 8.1.1 compares the NIST fit with the Whittaker-Henderson method. We used $s = 3$ and smoothing parameter $\lambda_{\mathrm{opt}} = 6.6$. which was determined by the GCV function `whgcv`.

The lower-left graph shows the corresponding periodogram spectra plotted versus period in units of months/cycle. The three dominant peaks are evident. In the spectrum graphs, the digital frequency is $\omega = 2\pi f$ rads/month, with f measured in cycles/month, and with the corresponding period $p = 1/f$ measured in months/cycle.

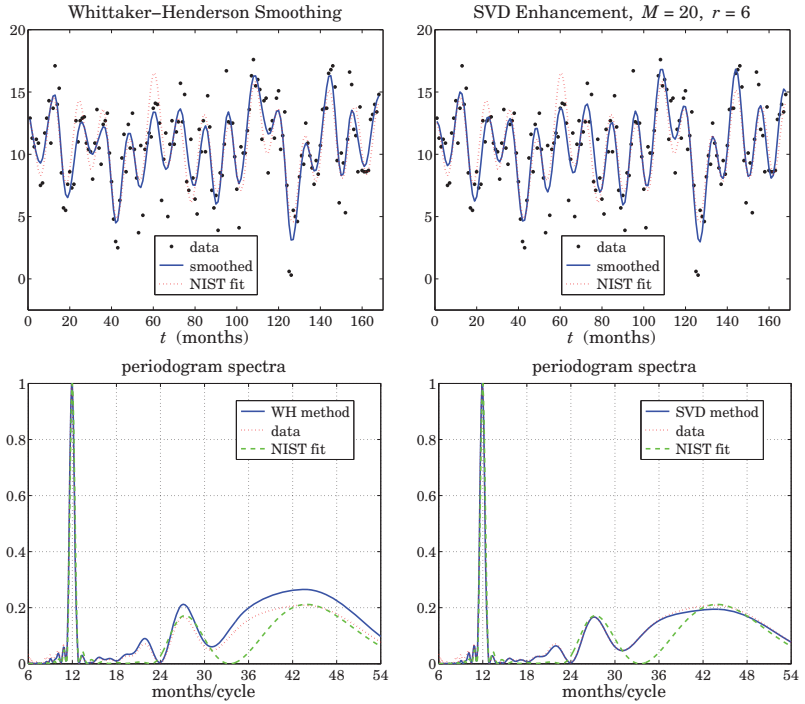


Fig. 8.1.1 Smoothed ENSO signal and spectra.

The time-domain WH signal agrees fairly well with the NIST fit. We note that in the places where the two disagree, the WH fit appears to be a better representation of the noisy data.

The upper-right graph shows the application of the SVD enhancement method, which typically works well for sinusoids in noise. The embedding dimension was $M = 20$ and the assumed rank $r = 6$ (three real sinusoids are equivalent to six complex ones.) The lower-right graph shows the corresponding spectral peaks. The following MATLAB code illustrates the generation of the four graphs:

```
Y = loadfile('ENSO.dat');                          % data file in OSP toolbox
y = Y(:,1); t = Y(:,2);                            % extract data signal

b1 = 1.0510749193E+01; b2 = 3.0762128085E+00; b3 = 5.3280138227E-01;
b4 = 4.4311088700E+01; b5 =-1.6231428586E+00; b6 = 5.2554493756E-01;
b7 = 2.6887614440E+01; b8 = 2.1232288488E-01; b9 = 1.4966870418E+00;

yf = b1 + b2*cos(2*pi*t/12) + b3*sin(2*pi*t/12) + b5*cos(2*pi*t/b4) ...
    + b6*sin(2*pi*t/b4) + b8*cos(2*pi*t/b7) + b9*sin(2*pi*t/b7);

s=3; la = linspace(2,10,100);                      % search range for λ
[gcv,lopt]=whgcv(y,la,s);                          % λopt = 6.6

yw = whsm(y,lopt,s);                               % WH smoothing method
M=20; r=6; ye = svdenh(y,M,r);                     % SVD enhancement method

figure; plot(t,y,'.', t,yw,'-', t,yf,':');         % upper-left graph
figure; plot(t,y,'.', t,ye,'-', t,yf,':');         % upper-right graph

p = linspace(6,54, 481); w = 2*pi./p;              % period in months/cycle

Sy = abs(freqz(zmean(y), 1, w)).^2;    Sy = Sy/max(Sy);        % spectra
Sf = abs(freqz(zmean(yf), 1, w)).^2;   Sf = Sf/max(Sf);
Sw = abs(freqz(zmean(yw), 1, w)).^2;   Sw = Sw/max(Sw);
Se = abs(freqz(zmean(ye), 1, w)).^2;   Se = Se/max(Se);

figure; plot(p,Sw, p,Sy,':', p,Sf,'--');           % lower-left graph
figure; plot(p,Se, p,Sy,':', p,Sf,'--');           % lower-right graph
```

The b_i parameters and the signal `yf` represent the NIST fit. Anticipating the three relevant peaks, the spectra were computed only over the period range $6 \le p \le 54$ months. The function `zmean` removes the mean of the signal so that the spectrum is not masked by the DC component. □

8.2 Regularization Filters

Most of the results of the spline smoothing case carry over to the discrete case. For example, we may obtain an equivalent digital filter by taking the signals to be double-sided and infinite. Using the Parseval identity, the performance index (8.1.3) becomes:

$$\begin{aligned}\mathcal{J} &= \sum_{n=-\infty}^{\infty} |y_n - x_n|^2 + \lambda \sum_{n=-\infty}^{\infty} |d_n * x_n|^2 = \\ &= \int_{-\pi}^{\pi} |\,Y(\omega) - X(\omega)\,|^2 \frac{d\omega}{2\pi} + \lambda \int_{-\pi}^{\pi} |\,D(\omega)X(\omega)\,|^2 \frac{d\omega}{2\pi} = \min\end{aligned} \tag{8.2.1}$$

where $D(\omega)$ is the frequency response[†] of the filter d_n, and we assumed unity weights, $w_n = 1$. The vanishing of the functional derivative of $\mathcal{J}$ with respect to $X^*(\omega)$,

$$\frac{\delta \mathcal{J}}{\delta X^*(\omega)} = X(\omega) - Y(\omega) + \lambda |D(\omega)|^2 X(\omega) = 0 \tag{8.2.2}$$

gives the effective equivalent smoothing filter $H(\omega) = X(\omega)/Y(\omega)$:

$$H(\omega) = \frac{1}{1 + \lambda |D(\omega)|^2} \tag{8.2.3}$$

The corresponding z-domain transfer function is obtained by noting that for real-valued d_n, we have $|D(\omega)|^2 = D(z)D(z^{-1})$, where $z = e^{j\omega}$, so that,

$$H(z) = \frac{1}{1 + \lambda D(z) D(z^{-1})} \tag{8.2.4}$$

Such "recursive regularization filters" have been considered in [440]. In particular, for the s-difference filter $D_s(z) = (1 - z^{-1})^s$, we have:

$$\boxed{H(z) = \frac{1}{1 + \lambda (1 - z^{-1})^s (1 - z)^s}} \quad \text{(Whittaker-Henderson filter)} \tag{8.2.5}$$

Similarly, Eq. (8.2.2) can be written in the z-domain and converted back to the time domain. Noting that $D_s(z)D_s(z^{-1}) = (1-z^{-1})^s(1-z)^s = (-1)^s z^s (1-z^{-1})^{2s} = (-1)^s z^s D_{2s}(z)$, we have:

$$X(z) - Y(z) + \lambda (-1)^s z^s (1 - z^{-1})^{2s} X(z) = 0\,, \qquad \text{or,}$$

$$(-1)^s z^s D_{2s}(z) X(z) = \lambda^{-1} \big(Y(z) - X(z)\big),$$

resulting in the time-domain $(2s)$-difference equation:

$$(-1)^s \nabla^{2s} x_{n+s} = \lambda^{-1} (y_n - x_n) \tag{8.2.6}$$

In the early years, some ingenious methods were developed for solving this type of equation [407–416]. Noting that $|D_s(\omega)| = |1 - e^{-j\omega}|^s = 2^s \left|\sin(\omega/2)\right|^s$, we obtain the frequency response of (8.2.5):

$$H(\omega) = \frac{1}{1 + \lambda \left| 2 \sin \dfrac{\omega}{2} \right|^{2s}} \tag{8.2.7}$$

The complementary highpass filter $H_c(z) = 1 - H(z)$ extracts the error residual component from the observations y_n, that is, $e_n = y_n - x_n$, or in the z-domain, $E(z) = Y(z) - X(z) = Y(z) - H(z)Y(z) = H_c(z)Y(z)$. Its transfer function and frequency response are given by:

$$H_c(z) = \frac{\lambda (1 - z^{-1})^s (1 - z)^s}{1 + \lambda (1 - z^{-1})^s (1 - z)^s}\,, \quad H_c(\omega) = \frac{\lambda \left| 2 \sin \dfrac{\omega}{2} \right|^{2s}}{1 + \lambda \left| 2 \sin \dfrac{\omega}{2} \right|^{2s}} \tag{8.2.8}$$

Fig. 8.2.1 shows a plot of $H(\omega)$ and $H_c(\omega)$ for $s = 1, 2, 3$ and the two values of the smoothing parameter $\lambda = 5$ and $\lambda = 50$. Increasing λ narrows the response of the lowpass filter and widens the response of the highpass one.

[†] Here, ω is the digital frequency in units of radians per sample.

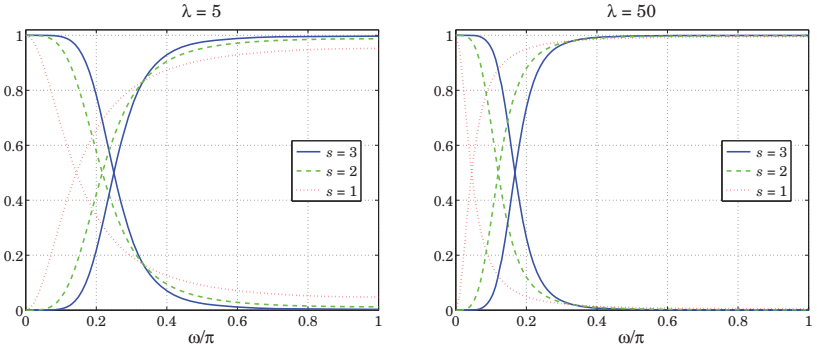


Fig. 8.2.1 Frequency responses of Whittaker-Henderson filters.

8.3 Hodrick-Prescott Filters

In macroeconomic applications such as extracting business cycles from GDP data, the standard signal model $y_n = x_n + v_n$ is interpreted to consist of a long-term trend represented by x_n and a shorter-term cyclical component v_n. The filters $H(z)$ and $H_c(z)$ extract the trend and cyclical components, respectively.

The use of Whittaker-Henderson smoothing with $s = 2$ has been advocated by Hodrick and Prescott [439] and has become standard in such applications. The Whittaker-Henderson filters are referred to as *Hodrick-Prescott filters* and there is a very large literature on the subject and on the use of other types of bandpass filters for extracting business cycles, a subset of which is [439–465].

As is the case in typical filter design, the filter parameter λ can be fixed by specifying a desired value for the filter's cutoff frequency ω_c corresponding to some standardized value of the gain. For the lowpass filter we have for general s, the condition:

$$\frac{1}{1+\lambda\left|2\sin\dfrac{\omega_c}{2}\right|^{2s}} = G_c \tag{8.3.1}$$

where G_c is desired value of the gain. The 3-dB cutoff frequency ω_c corresponds to $G_c = 1/\sqrt{2}$. In macroeconomic applications, the 6-dB frequency is often used, corresponding to the choice $G_c = 1/2$. For the highpass case, measuring the gain G_c relative to that at the Nyquist frequency $\omega = \pi$, we have the condition:

$$\frac{\lambda\left|2\sin\dfrac{\omega_c}{2}\right|^{2s}}{1+\lambda\left|2\sin\dfrac{\omega_c}{2}\right|^{2s}} = G_c\,\frac{2^{2s}\lambda}{1+2^{2s}\lambda} \tag{8.3.2}$$

Typically, *business cycles* are defined [446] as having frequency components with periods between 6 and 32 quarters (1.5 to 8 years). A bandpass filter with a passband $[\omega_1, \omega_2] = [2\pi/32,\ 2\pi/6]$ radians/quarter would extract such cycles.

The Hodrick-Prescott highpass filter $H_c(\omega)$ must therefore have a cutoff frequency of about $\omega_c = \omega_1 = 2\pi/32$. Hodrick-Prescott advocate the use of $\lambda = 1600$ for quarterly data. Interestingly, the values of $\lambda = 1600$ and $\omega_c = 2\pi/32$ rads/quarter, correspond to almost a 3-dB gain. Indeed, the gain calculated from Eq. (8.3.2) with $s = 2$ turns out to be $G_c = 0.702667 \equiv -3.065$ dB.

Using the same ω_c and G_c, but different values of s requires adjusting the value of λ. For example, solving Eq. (8.3.2) for λ with $s = 1, 2, 3$ gives:

$$\lambda = 1600\ (s=2),\quad \lambda = 60.654\ (s=1),\quad \lambda = 41640\ (s=3) \tag{8.3.3}$$

Similarly, the value of λ must be adjusted if the sampling frequency is changed. For example, the same cutoff frequency expressed in different units is:

$$\omega_c = \frac{2\pi}{32}\,\frac{\text{radians}}{\text{quarter}} = \frac{2\pi}{8}\,\frac{\text{radians}}{\text{year}} = \frac{2\pi}{96}\,\frac{\text{radians}}{\text{month}} \tag{8.3.4}$$

The above value $G_c = 0.702667$ used in (8.3.2) with $s = 2$ then gives the following values of λ for quarterly, yearly, and monthly sampled data:

$$\lambda = 1600\ \text{(quarterly)},\quad \lambda = 6.677\ \text{(yearly)},\quad \lambda = 128878\ \text{(monthly)} \tag{8.3.5}$$

Similarly, using the slightly more exact value $G_c = 1/\sqrt{2}$ and $s = 2$ gives:

$$\lambda = 1634.5\ \text{(quarterly)},\quad \lambda = 6.822\ \text{(yearly)},\quad \lambda = 131659\ \text{(monthly)} \tag{8.3.6}$$

There is not much agreement as to the values of λ to be used for annual and monthly data. Two other sets of values are as follows, with the first being used by the European Central Bank and the second recommended by [456,463],

$$\begin{aligned} &\lambda = 1600/4^2 = 100\ \text{(yearly)}, && \lambda = 1600\times 3^2 = 14400\ \text{(monthly)}\\ &\lambda = 1600/4^4 = 6.25\ \text{(yearly)}, && \lambda = 1600\times 3^4 = 129600\ \text{(monthly)} \end{aligned} \tag{8.3.7}$$

The latter choice is essentially the same as that of Eq. (8.3.5) based on the criterion (8.3.2). Indeed, for small ω_c, we may make the approximation $2\sin(\omega_c/2) \approx \omega_c$. Since $2^{2s}\lambda$ is typically much larger than unity, the right-hand side of Eq. (8.3.2) can be replaced by G_c, resulting in the following approximate solution, which turns out to be valid up to about $\omega_c \le 0.3\pi$,

$$\frac{\lambda\omega_c^{2s}}{1+\lambda\omega_c^{2s}} = G_c \quad\Rightarrow\quad \lambda = \frac{G_c}{(1-G_c)\,\omega_c^{2s}} \tag{8.3.8}$$

If in this formula, we adjust G_c to get $\lambda = 1600$ at $\omega_c = 2\pi/32$, we find $G_c = 0.70398 \equiv -3.049$ dB, which in turn generates the second set of values in Eq. (8.3.7).

Example 8.3.1: *US GDP for investment.* A protypical example is the application of the Hodrick-Prescott filter to the US GDP. Fig. 8.3.1 shows the real gross domestic product in chained (2000) dollars from private domestic investment, seasonally adjusted at annual rates. The data can be retrieved (as Table 1.1.6) from the BEA web sites:

```
http://www.bea.gov/
http://www.bea.gov/national/nipaweb/Index.asp
```

The signal to be smoothed is the log of the GDP, that is, $y = \log_{10}(\text{GDP})$, and the ordinate units are such that $y = 12$ corresponds to GDP $= 10^{12}$, or, one trillion dollars. The data are quarterly and span the years 1947–2008.

The upper-left graph shows the raw data and the WH-smoothed signal computed with $s = 2$ and $\lambda = 1600$, which as we mentioned above correspond to an approximate 3-dB cutoff frequency of 32 quarters. The upper-right graph shows the residual cyclical component. Its deviations above or below zero indicate the business cycles.

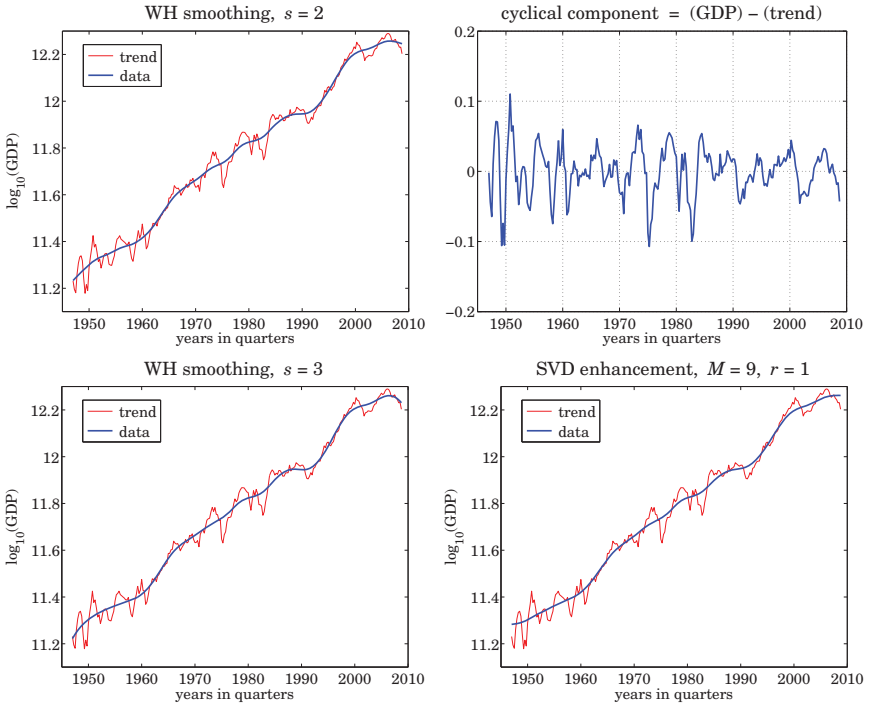


Fig. 8.3.1 U.S. quarterly GDP in private investment, 1947–2008.

For comparison, the left-bottom graph shows the WH-smoothed signal with $s = 3$ and $\lambda = 41640$ adjusted to match the same 3-dB cutoff frequency as the $s = 2$ case, see Eq. (8.3.3). The lower-right graph shows the smoothed trend from the SVD enhancement method applied with embedding dimension $M = 9$ and rank $r = 1$. The following MATLAB code illustrates the generation of the four graphs:

```
Y = loadfile('USGDP_Inv.dat');          % data file in OSP toolbox
y = log10(Y(:,2) * 1e9);                % Y was in billions
t = taxis(y,4,1947);                    % t-axis in quarters since 1947

s = 2; la = 1600; yt = whsm(y,la,s);    % WH smoothing with s = 2
figure; plot(t,yt,'-', t,y,'--');       % upper-left graph
```



```
yc = y-yt;                              % cyclical component
figure; plot(t,yc, '-');                % upper-right graph

s = 3; la = 41640.16; yt = whsm(y,la,s);  % WH smoothing with s = 3
figure; plot(t,yt,'-', t,y,'--');       % bottom-left graph

M=9; r=1; ye = svdenh(y,M,r);           % SVD enhancement method
figure; plot(t,ye,'-', t,y,'--');       % bottom-right graph
```

Except near the end-points, the smoothed trend for $s = 3$ is virtually indistinguishable from the $s = 2$ case, and therefore, it would lead to the same prediction of business cycles. The SVD trend is also very comparable. □

8.4 Poles and Impulse Response

Because of the invariance under the substitution $z \to z^{-1}$, the $2s$ poles of the filter $H(z)$ of Eq. (8.2.5) come in two groups with s poles inside the unit circle and their reciprocals outside. It follows that $H(z)$ can be expressed in the factored form:

$$H(z) = \frac{1}{1+\lambda(1-z^{-1})^s(1-z)^s} = \prod_{k=1}^{s}\left[\frac{(1-z_k)^2}{(1-z_k z^{-1})(1-z_k z)}\right] \tag{8.4.1}$$

where $z_k, k = 1, 2, \ldots, s$, denote the s poles inside the unit circle. The numerator factors $(1-z_k)^2$ ensure that the right-hand side has unity gain at DC ($z = 1$), as does the left-hand side. The stable impulse response is double-sided and can be obtained by performing an inverse z-transform with the unit circle as the inversion contour [12]:

$$h_n = \oint_{\text{u.c.}} H(z) z^n \frac{dz}{2\pi j z}, \quad -\infty < n < \infty \tag{8.4.2}$$

Inserting the factored form (8.4.1) into (8.4.2), we find

$$h_n = \sum_{k=1}^{s} A_k z_k^{|n|}, \quad -\infty < n < \infty \tag{8.4.3}$$

where the coefficients A_k are given by

$$A_k = \left(\frac{1-z_k}{1+z_k}\right)\prod_{\substack{i=1\\ i\neq k}}^{s}\left[\frac{(1-z_i)^2}{(1-z_i z_k^{-1})(1-z_i z_k)}\right], \quad k = 1, 2, \ldots, s \tag{8.4.4}$$

The poles can be obtained in the form $z_k = e^{j\omega_k}$, where ω_k are the complex frequencies of the denominator, that is, the frequencies that are solutions of the equation:

$$1 + \lambda\left[2\sin\frac{\omega}{2}\right]^{2s} = 0 \tag{8.4.5}$$

where we note that even though ω is complex, we still have $(1-z^{-1})(1-z) = 4\sin^2(\omega/2)$ for $z = e^{j\omega}$. The solution of Eq. (8.4.5) is straightforward. The s frequencies ω_k that lead to poles z_k that are inside the unit circle can be parametrized as follows:

$$\left[2\sin\frac{\omega}{2}\right]^{2s} = \frac{-1}{\lambda} = \frac{e^{j\pi(2k-1)}}{\lambda} \quad \Rightarrow \quad \sin\frac{\omega_k}{2} = \frac{e^{j\pi(2k-1)/(2s)}}{2\lambda^{1/(2s)}}, \quad k = 1, 2, \ldots, s$$

Thus, the desired set of poles are:

$$z_k = e^{j\omega_k}\,,\quad \omega_k = 2\arcsin\left[\frac{e^{j\theta_k}}{2\lambda^{1/(2s)}}\right],\quad \theta_k = \frac{\pi(2k-1)}{2s}\,,\quad k = 1,2,\ldots,s \tag{8.4.6}$$

If s is even, then the z_k (and the coefficients A_k) come in conjugate pairs. If s is odd, then the zero at $k = (s+1)/2$ is real and the rest come in conjugate pairs. In either case, h_n given by (8.4.3) is real-valued and decays exponentially from either side of the time axis. The MATLAB function `whimp` calculates h_n at any vector of ns and also produces the poles and residues z_k, A_k, $k = 1,2,\ldots,s$,

```
[h,z,A] = whimp(lambda,s,n);      % Whittaker-Henderson impulse response and poles
```

Fig. 8.4.1 shows the impulse responses for $s = 1,2,3$ with λs chosen as in (8.3.3) so that the (complementary) filters have the same 3-dB cutoff frequency. The impulse response of the complementary filter is $h_c(n) = \delta(n) - h(n)$. Therefore, the three responses will have roughly the same time width.

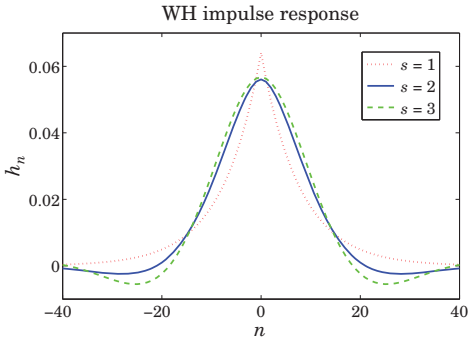


Fig. 8.4.1 Impulse responses of Whittaker-Henderson filters.

8.5 Wiener Filter Interpretation

Finally, we note that, as in the spline smoothing case, the equivalent filter $H(z)$ can be regarded as the optimum unrealizable Wiener filter for estimating x_n from the observations y_n in the signal model:

$$y_n = x_n + v_n\,,\quad \nabla^s x_n = w_n \tag{8.5.1}$$

where v_n, w_n are mutually uncorrelated white noise signals with variances σ_v^2, σ_w^2. Indeed, the transformed signals

$$\bar{x}_n = \nabla^s x_n = w_n\,,\quad \bar{y}_n = \nabla^s y_n = w_n + \nabla^s v_n$$

are stationary and it follows from the results of [643–649] that the optimum Wiener filter will be in this case:

$$H(z) = \frac{S_{\bar{x}\bar{y}}(z)}{S_{\bar{y}\bar{y}}(z)} = \frac{S_{ww}(z)}{S_{ww}(z) + D_s(z)D_s(z^{-1})S_{vv}(z)} = \frac{\sigma_w^2}{\sigma_w^2 + \sigma_v^2 D_s(z)D_s(z^{-1})} \tag{8.5.2}$$

Thus, we may identify $\lambda = \sigma_v^2/\sigma_w^2$. Using such a model, the optimum estimation filter based on a finite, length-N, set of observations y_n can be implemented efficiently as a Kalman filter smoother requiring $O(N)$ operations. On the other hand, the solution of the matrix equation (8.1.10) is just as efficient due to the sparse and banded nature of the linear system.

8.6 Regularization and Kernel Machines

Regularization was initially invented as a method for solving ill-posed, inconsistent, overdetermined, and ill-conditioned inverse problems. Recently it has been applied also to support-vector machines and kernel methods for machine learning. There is a huge literature on the subject, a small subset of which is [480–519]. Here, we present a short discussion with particular emphasis on deconvolution and kernel regression methods.

Both spline and Whittaker-Henderson smoothing are examples of regularization. The performance index Eq. (8.1.3) can be generalized further to cover the case of deconvolution, or inverse filtering,

$$\mathcal{J} = \sum_n |y_n - f_n * x_n|^2 + \lambda \sum_n |d_n * x_n|^2 = \min \tag{8.6.1}$$

where f_n and d_n are FIR filters. This attempts to solve $y_n = f_n * x_n$ for x_n by deconvolving the effect of f_n. We may write (8.6.1) in a compact matrix form using the convolution matrices F, D of the two filters:

$$\mathcal{J} = \|\mathbf{y} - F\mathbf{x}\|^2 + \lambda\|D\mathbf{x}\|^2 = (\mathbf{y} - F\mathbf{x})^T(\mathbf{y} - F\mathbf{x}) + \lambda\mathbf{x}^T(D^T D)\mathbf{x} = \min \tag{8.6.2}$$

The solution is obtained from the gradient,

$$\frac{\partial \mathcal{J}}{\partial \mathbf{x}} = -2F^T(\mathbf{y} - F\mathbf{x}) + 2\lambda D^T D\mathbf{x} = 0\,,\quad \text{or,}$$

$$\hat{\mathbf{x}} = (F^T F + \lambda D^T D)^{-1} F^T \mathbf{y} \tag{8.6.3}$$

The problem (8.6.2) is of course much more general than inverse filtering. The method is known as *Tikhonov regularization* and as *ridge regression*. The linear system $\mathbf{y} = F\mathbf{x}$ may in general be overdetermined, or underdetermined, or rank defective. We discuss such cases in Chap. 15. To simplify the discussion, we assume here that the linear system is either square and invertible or overdetermined but F having full rank. For $\lambda = 0$, we obtain $\hat{\mathbf{x}} = (F^T F)^{-1} F^T \mathbf{y}$, which is recognized as the unique pseudoinverse least-squares solution. In the square case, we have $\hat{\mathbf{x}} = F^{-1}\mathbf{y}$. We are envisioning a signal model of the form $\mathbf{y} = F\mathbf{x} + \mathbf{v}$ and the objective is to determine an estimate of $\mathbf{x}$. We have then,

$$\hat{\mathbf{x}} = F^{-1}\mathbf{y} = F^{-1}(F\mathbf{x} + \mathbf{v}) = \mathbf{x} + F^{-1}\mathbf{v} \tag{8.6.4}$$

A potential problem with this estimate is that if F is ill-conditioned with a large condition number—a common occurrence in practice—the resulting inverse-filtered noise component $\mathbf{u} = F^{-1}\mathbf{v}$ may be magnified to such an extent that it will mask the desired term $\mathbf{x}$, rendering the estimate $\hat{\mathbf{x}}$ useless. The same can happen in the overdetermined case. The presence of the regularization term helps in this regard by providing a more well-conditioned inverse. For the deconvolution problem, one typically selects D to be the unit matrix, $D = I$, leading to the solution,

$$\hat{\mathbf{x}} = (F^T F + \lambda I)^{-1} F^T \mathbf{y} \tag{8.6.5}$$

To see how regularization improves the condition number, let $\lambda_{\max}, \lambda_{\min}$ be the maximum and minimum eigenvalues of $F^T F$. Then, the condition numbers of $F^T F$ and $F^T F + \lambda I$ are $\lambda_{\max}/\lambda_{\min}$ and $(\lambda_{\max} + \lambda)/(\lambda_{\min} + \lambda)$. A highly ill-conditioned problem would have $\lambda_{\min} \ll \lambda_{\max}$. It is straightforward to verify that the larger the λ, the more the condition number of the regularized matrix is reduced:

$$\lambda \gg 1 \quad \Rightarrow \quad \frac{\lambda_{\max} + \lambda}{\lambda_{\min} + \lambda} \ll \frac{\lambda_{\max}}{\lambda_{\min}}$$

For example, if $\lambda_{\min} = 10^{-3}$ and $\lambda_{\max} = 10^3$, we have $\lambda_{\max}/\lambda_{\min} = 10^6$, but the regularized version $(\lambda_{\max} + \lambda)/(\lambda_{\min} + \lambda)$ takes approximately the values 11, 2, 1.1, for $\lambda = 10^2, 10^3, 10^4$, respectively.

Regularization is not without problems. For noisy data the basic tradeoff is that improving the condition number by increasing λ causes more distortion and smoothing of the desired signal component $\mathbf{x}$. As usual, choosing the proper value of λ is more of an art than science and requires some trial-and-error experimentation. The method of cross-validation can also be applied [368] as a guide.

Other choices for D, for example differencing matrices, are used in applications such as edge-preserving deblurring of images. We discuss deconvolution and inverse filter design further in Sections 12.14 and 15.11.

Next, we consider briefly the connection of regularization to machine learning and reproducing kernel Hilbert spaces. We recall that the objective of the spline smoothing performance index,

$$\mathcal{J} = \sum_{n=0}^{N-1} [y_n - f(t_n)]^2 + \lambda \int_{t_a}^{t_b} [\ddot{f}(t)]^2 dt = \min \tag{8.6.6}$$

was to "learn" the unknown function $f(t)$ from a finite subset of N noisy observations $y_n = f(t_n) + v_n$, $n = 0, 1, \ldots, N-1$. The concept can be generalized from functions of time to multivariable functions of some independent variable, say $\mathbf{x}$, such as three-dimensional space. The observed data samples are of the form $y_n = f(\mathbf{x}_n) + v_n$, and the objective is to learn the unknown function $f(\mathbf{x})$. The performance index is replaced by,

$$\mathcal{J} = \sum_{n=0}^{N-1} [y_n - f(\mathbf{x}_n)]^2 + \lambda \|f\|^2 = \min \tag{8.6.7}$$

where $\|f\|$ is an appropriate norm that depends on the approach one takes to the minimization problem. One possible and very successful approach is to use a *neural network* to model the unknown function $f(\mathbf{x})$. In this case the regularization norm depends on the parameters of the neural network and its assumed structure (typically a single-hidden layer is sufficient.)

The reproducing kernel approach that we discuss here is to assume that $f(\mathbf{x})$ can be represented as a linear combination of a finite or infinite set of nonlinear basis functions $\phi_i(\mathbf{x})$, $i = 1, 2, \ldots, M$, where for now we will assume that M is finite,

$$f(\mathbf{x}) = \sum_{i=1}^{M} \phi_i(\mathbf{x}) c_i = \boldsymbol{\phi}^T(\mathbf{x})\mathbf{c}, \qquad \boldsymbol{\phi}(\mathbf{x}) = \begin{bmatrix} \phi_1(\mathbf{x}) \\ \phi_2(\mathbf{x}) \\ \vdots \\ \phi_M(\mathbf{x}) \end{bmatrix}, \quad \mathbf{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_M \end{bmatrix} \tag{8.6.8}$$

This is analogous to the approach of Chap. 3 where $f(t)$ was modeled as a polynomial in t and expanded in the monomial basis functions $s_i(t) = t^i$. Here, we define the regularization norm in terms of the coefficients c_i as follows:

$$\|f\|^2 = \sum_{i=1}^{M} \frac{c_i^2}{\lambda_i} = \mathbf{c}^T \Lambda^{-1} \mathbf{c} \tag{8.6.9}$$

where λ_i is a set of some given positive coefficients, and $\Lambda = \text{diag}([\lambda_1, \lambda_2, \ldots, \lambda_M])$. The function values at the N observation points are $f(\mathbf{x}_n) = \boldsymbol{\phi}^T(\mathbf{x}_n)\mathbf{c}$, and can be arranged into an N-dimensional column vector:

$$\mathbf{f} = \begin{bmatrix} f(\mathbf{x}_0) \\ \vdots \\ f(\mathbf{x}_n) \\ \vdots \\ f(\mathbf{x}_{N-1}) \end{bmatrix} = \Phi\mathbf{c}, \qquad \Phi = \begin{bmatrix} \boldsymbol{\phi}^T(\mathbf{x}_0) \\ \vdots \\ \boldsymbol{\phi}^T(\mathbf{x}_n) \\ \vdots \\ \boldsymbol{\phi}^T(\mathbf{x}_{N-1}) \end{bmatrix} \tag{8.6.10}$$

where Φ has dimension $N \times M$. Thus, the performance index can be written compactly,

$$\boxed{\mathcal{J} = (\mathbf{y} - \Phi\mathbf{c})^T (\mathbf{y} - \Phi\mathbf{c}) + \lambda \mathbf{c}^T \Lambda^{-1} \mathbf{c} = \min} \tag{8.6.11}$$

The solution for the optimum coefficients $\mathbf{c}$ is obtained by setting the gradient to zero:

$$\frac{\partial \mathcal{J}}{\partial \mathbf{c}} = -2\Phi^T(\mathbf{y} - \Phi\mathbf{c}) + 2\lambda \Lambda^{-1}\mathbf{c} = 0 \quad \Rightarrow \quad (\lambda\Lambda^{-1} + \Phi^T\Phi)\mathbf{c} = \Phi^T\mathbf{y} \tag{8.6.12}$$

$$\mathbf{c} = (\lambda\Lambda^{-1} + \Phi^T\Phi)^{-1}\Phi^T\mathbf{y} \tag{8.6.13}$$

Using the matrix-inversion lemma, we have:

$$(\lambda\Lambda^{-1} + \Phi^T\Phi)^{-1} = \frac{1}{\lambda}[\Lambda - \Lambda\Phi^T(\lambda I + \Phi\Lambda\Phi^T)^{-1}\Phi\Lambda] \tag{8.6.14}$$

from which it follows that:

$$(\lambda\Lambda^{-1} + \Phi^T\Phi)^{-1}\Phi^T = \Lambda\Phi^T(\lambda I + \Phi\Lambda\Phi^T)^{-1} \tag{8.6.15}$$

where I is the $N\times N$ identity matrix. Thus, the optimal coefficients are given by

$$\mathbf{c} = \Lambda\Phi^T(\lambda I + \Phi\Lambda\Phi^T)^{-1}\mathbf{y} \tag{8.6.16}$$

The observation vector $\mathbf{f} = \Phi\mathbf{c}$ and estimated function value $f(\mathbf{x}) = \boldsymbol{\phi}^T(\mathbf{x})\mathbf{c}$ are then,

$$\boxed{\begin{aligned}&\mathbf{f} = \Phi\Lambda\Phi^T(\lambda I + \Phi\Lambda\Phi^T)^{-1}\mathbf{y}\\ &f(\mathbf{x}) = \boldsymbol{\phi}^T(\mathbf{x})\Lambda\Phi^T(\lambda I + \Phi\Lambda\Phi^T)^{-1}\mathbf{y}\end{aligned}} \tag{8.6.17}$$

The appearance of the bilinear products of the basis functions suggests that we define the *kernel* function:

$$\boxed{K(\mathbf{x},\mathbf{x}') = \boldsymbol{\phi}^T(\mathbf{x})\Lambda\boldsymbol{\phi}(\mathbf{x}') = \sum_{i=1}^{M}\lambda_i\phi_i(\mathbf{x})\,\phi_i(\mathbf{x}')} \tag{8.6.18}$$

Let us also define the $N\times N$ symmetric positive-definite kernel matrix K and N-dimensional coefficient vector $\mathbf{a} = [a_0, a_1, \ldots, a_{N-1}]^T$ by

$$\boxed{\begin{aligned}&K = \Phi\Lambda\Phi^T\\ &\mathbf{a} = (\lambda I + K)^{-1}\mathbf{y}\end{aligned}} \tag{8.6.19}$$

so that $\mathbf{c} = \Lambda\Phi^T\mathbf{a}$ and $\mathbf{f} = \Phi\mathbf{c} = \Phi\Lambda\Phi^T\mathbf{a} = K\mathbf{a}$ and $f(\mathbf{x}) = \boldsymbol{\phi}^T(\mathbf{x})\Lambda\Phi^T\mathbf{a}$. The matrix elements of K can be expressed in terms of the kernel function:

$$K_{nm} = \big(\Phi\Lambda\Phi^T\big)_{nm} = \boldsymbol{\phi}^T(\mathbf{x}_n)\Lambda\boldsymbol{\phi}(\mathbf{x}_m) = K(\mathbf{x}_n,\mathbf{x}_m) \tag{8.6.20}$$

for $n, m = 0, 1, \ldots, N-1$. Similarly, we have for $f(\mathbf{x})$,

$$\begin{aligned}f(\mathbf{x}) &= \boldsymbol{\phi}^T(\mathbf{x})\Lambda\Phi^T\mathbf{a} = \boldsymbol{\phi}^T(\mathbf{x})\Lambda\big[\boldsymbol{\phi}(x_0),\ldots,\boldsymbol{\phi}(\mathbf{x}_n),\ldots,\boldsymbol{\phi}(x_{N-1})\big]\mathbf{a}\\ &= \big[K(\mathbf{x},\mathbf{x}_1),\ldots,K(\mathbf{x},\mathbf{x}_n),\ldots,K(\mathbf{x},\mathbf{x}_{N-1})\big]\mathbf{a} = \sum_{n=0}^{N-1}K(\mathbf{x},\mathbf{x}_n)\,a_n\end{aligned} \tag{8.6.21}$$

Thus, we may express (8.6.17) directly in terms of the kernel function and the coefficient vector $\mathbf{a}$,

$$\boxed{\mathbf{f} = K\mathbf{a}\,,\quad f(\mathbf{x}) = \sum_{n=0}^{N-1}K(\mathbf{x},\mathbf{x}_n)\,a_n} \tag{8.6.22}$$

Moreover, since $\mathbf{c} = \Lambda\Phi^T\mathbf{a}$, the norm $\|f\|^2$ can also be expressed in terms of K and the vector $\mathbf{a}$ as follows, $\|f\|^2 = \mathbf{c}^T\Lambda^{-1}\mathbf{c} = (\mathbf{a}^T\Phi\Lambda)\Lambda^{-1}(\Lambda\Phi^T\mathbf{a}) = \mathbf{a}^T(\Phi\Lambda\Phi^T)\mathbf{a}$, or,

$$\|f\|^2 = \mathbf{a}^T K\mathbf{a} \tag{8.6.23}$$

Thus, the knowledge of the kernel function $K(\mathbf{x},\mathbf{x}')$—rather than the knowledge of the possibly infinite set of basis functions $\phi_i(\mathbf{x})$—is sufficient to formulate and solve

the regularization problem. Indeed, an equivalent optimization problem to (8.6.11) is the following, with the performance index to be minimized with respect to $\mathbf{a}$:

$$\boxed{\mathcal{J} = (\mathbf{y} - K\mathbf{a})^T(\mathbf{y} - K\mathbf{a}) + \lambda\mathbf{a}^T K\mathbf{a} = \min} \tag{8.6.24}$$

The vanishing of gradient with respect to $\mathbf{a}$ leads to the same solution as (8.6.19),

$$\frac{\partial\mathcal{J}}{\partial\mathbf{a}} = -2K^T(\mathbf{y} - K\mathbf{a}) + 2\lambda K\mathbf{a} = 0 \quad\Rightarrow\quad (\lambda K + K^T K)\mathbf{a} = K^T\mathbf{y} \quad\Rightarrow\quad \mathbf{a} = (\lambda I + K)^{-1}\mathbf{y}$$

where we used the symmetry property $K^T = K$ and assumed that K was invertible.

The linear vector space of functions of the form $f(\mathbf{x}) = \boldsymbol{\phi}^T(\mathbf{x})\mathbf{c}$, spanned by the set of basis functions $\{\phi_i(\mathbf{x}),\ i = 1, 2, \ldots, M\}$, can be turned into an inner-product space (a Hilbert space if $M = \infty$) by endowing it with the inner product induced by the norm (8.6.9). That is, for any two functions $f_1(\mathbf{x}) = \boldsymbol{\phi}^T(\mathbf{x})\mathbf{c}_1$ and $f_2(\mathbf{x}) = \boldsymbol{\phi}^T(\mathbf{x})\mathbf{c}_2$, we define the inner product:

$$\langle f_1, f_2\rangle = \mathbf{c}_1^T\Lambda^{-1}\mathbf{c}_2 \tag{8.6.25}$$

The resulting vector space, say $\mathcal{H}$, is referred to as a *reproducing kernel Hilbert space*. By writing the kernel function in the form, $K(\mathbf{x},\mathbf{x}') = \boldsymbol{\phi}^T(\mathbf{x})\Lambda\boldsymbol{\phi}(\mathbf{x}') \equiv \boldsymbol{\phi}^T(\mathbf{x})\mathbf{c}(\mathbf{x}')$, with $\mathbf{c}(\mathbf{x}') = \Lambda\boldsymbol{\phi}(\mathbf{x}')$, we see that, as a function of $\mathbf{x}$ for each fixed $\mathbf{x}'$, it lies in the space $\mathcal{H}$, and satisfies the two reproducing-kernel properties:

$$f(\mathbf{x}') = \big\langle f(\cdot), K(\cdot,\mathbf{x}')\big\rangle\,,\qquad K(\mathbf{x},\mathbf{x}') = \big\langle K(\cdot,\mathbf{x}), K(\cdot,\mathbf{x}')\big\rangle \tag{8.6.26}$$

These follow from the definition (8.6.25). Indeed, given $f(\mathbf{x}) = \boldsymbol{\phi}^T(\mathbf{x})\mathbf{c}$, we have,

$$\big\langle f(\cdot), K(\cdot,\mathbf{x}')\big\rangle = \mathbf{c}^T\Lambda^{-1}\mathbf{c}(\mathbf{x}') = \mathbf{c}^T\Lambda^{-1}\Lambda\,\boldsymbol{\phi}(\mathbf{x}') = \mathbf{c}^T\boldsymbol{\phi}(\mathbf{x}') = f(\mathbf{x}')$$

$$\big\langle K(\cdot,\mathbf{x}), K(\cdot,\mathbf{x}')\big\rangle = \mathbf{c}(\mathbf{x})^T\Lambda^{-1}\mathbf{c}(\mathbf{x}') = \boldsymbol{\phi}^T(\mathbf{x})\Lambda\Lambda^{-1}\Lambda\,\boldsymbol{\phi}(\mathbf{x}') = \boldsymbol{\phi}^T(\mathbf{x})\Lambda\,\boldsymbol{\phi}(\mathbf{x}') = K(\mathbf{x},\mathbf{x}')$$

One can re-normalize the basis functions by defining $\bar{\phi}_i(\mathbf{x}) = \lambda_i^{-1/2}\phi_i(\mathbf{x})$, or, vectorially $\bar{\boldsymbol{\phi}}(\mathbf{x}) = \Lambda^{-1/2}\boldsymbol{\phi}(\mathbf{x})$, which imply the renormalized basis matrix $\bar{\Phi} = \Phi\Lambda^{1/2}$ and coefficient vector $\bar{\mathbf{c}} = \Lambda^{-1/2}\mathbf{c}$. We obtain then the alternative expressions:

$$\begin{aligned}&\bar{\mathbf{c}} = \bar{\Phi}^T\mathbf{a}\\ &\mathbf{f} = \Phi\mathbf{c} = \bar{\Phi}\bar{\mathbf{c}}\\ &K = \Phi\Lambda\Phi^T = \bar{\Phi}\bar{\Phi}^T\\ &\|f\|^2 = \mathbf{c}^T\Lambda^{-1}\mathbf{c} = \bar{\mathbf{c}}^T\bar{\mathbf{c}}\end{aligned} \tag{8.6.27}$$

and kernel function,

$$K(\mathbf{x},\mathbf{x}') = \boldsymbol{\phi}^T(\mathbf{x})\Lambda\boldsymbol{\phi}(\mathbf{x}') = \bar{\boldsymbol{\phi}}^T(\mathbf{x})\bar{\boldsymbol{\phi}}(\mathbf{x}') \tag{8.6.28}$$

Eq. (8.6.28) expresses the kernel function as the dot product of two vectors and is known as the *kernel trick*. Given a kernel function $K(\mathbf{x},\mathbf{x}')$ that satisfies certain positive-definiteness conditions, the existence of basis functions satisfying Eq. (8.6.28) is guaranteed by Mercer's theorem [512]. The remarkable property of the kernel regularization approach is Eq. (8.6.22), which is known as the *representer theorem* [512],

$$f(\mathbf{x}) = \sum_{n=0}^{N-1}K(\mathbf{x},\mathbf{x}_n)\,a_n \tag{8.6.29}$$

It states that even though the original least-squares problem (8.6.11) was formulated in a possibly infinite-dimensional Hilbert space, the resulting solution is represented by a finite number of terms in Eq. (8.6.29). This property is more general than the above case and it applies to a performance index of the form:

$$\mathcal{J} = L(\mathbf{y} - \Phi\mathbf{c}) + \lambda \mathbf{c}^T \Lambda^{-1} \mathbf{c} = \min \tag{8.6.30}$$

where $L(\mathbf{z})$ is an arbitrary (convex, increasing, and differentiable) scalar function that replaces the quadratic norm $L(\mathbf{z}) = \mathbf{z}^T\mathbf{z}$. Indeed, the vanishing of the gradient gives,

$$\frac{\partial \mathcal{J}}{\partial \mathbf{c}} = -\Phi^T \frac{\partial L}{\partial \mathbf{y}} + 2\lambda \Lambda^{-1}\mathbf{c} = 0 \quad \Rightarrow \quad \mathbf{c} = \Lambda \Phi^T \frac{1}{2\lambda} \frac{\partial L}{\partial \mathbf{y}}$$

which implies for $\mathbf{f} = \Phi\mathbf{c}$ and $f(\mathbf{x}) = \boldsymbol{\phi}^T(\mathbf{x})\mathbf{c}$,

$$\mathbf{f} = K\mathbf{a}\,, \quad f(\mathbf{x}) = \sum_{n=0}^{N-1} K(\mathbf{x}, \mathbf{x}_n)\, a_n\,, \quad \text{with} \quad \mathbf{a} = \frac{1}{2\lambda} \frac{\partial L(\mathbf{y} - K\mathbf{a})}{\partial \mathbf{y}} \tag{8.6.31}$$

where the last equation is a nonlinear equation for the N-vector $\mathbf{a}$. Of course, in the quadratic-norm case, $L(\mathbf{z}) = \mathbf{z}^T\mathbf{z}$, we obtain the equivalent of (8.6.19),

$$\mathbf{a} = \frac{1}{\lambda}(\mathbf{y} - K\mathbf{a})$$

Kernels and the above representation property are used widely in machine learning applications, such as support vector machines [499]. Some typical kernels that satisfy the representation property (8.6.28) are polynomial and gaussian of the type:

$$\begin{aligned} K(\mathbf{x}, \mathbf{x}') &= (c + \mathbf{x}\cdot\mathbf{x}')^p \\ K(\mathbf{x}, \mathbf{x}') &= \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right) \end{aligned} \tag{8.6.32}$$

By mapping nonlinear problems into linear ones, kernel methods offer a new paradigm for solving many of the classical problems of estimation and classification, including the "kernelization" of methods such as principal component analysis [513], canonical correlation analysis, array processing [516], and adaptive filtering [519]. Some accessible overviews of kernel methods with emphasis on regularization are [504,510,515]. For more details, the reader may consult the references [480–519].

8.7 Sparse Whittaker-Henderson Methods

Several variations of the Whittaker-Henderson method have been proposed in the literature that use different norms for the two terms of Eq. (8.1.1), such as the following criterion based on the ℓ_q and the ℓ_p norms, and using unity weights w_n for simplicity,

$$\mathcal{J}_{qp} = \sum_{n=0}^{N-1} |y_n - x_n|^q + \lambda \sum_{n=s}^{N-1} |\nabla^s x_n|^p = \min \tag{8.7.1}$$

Such criteria are capable of handling outliers in the data more effectively. Eq. (8.7.1) can be written vectorially with the help of the s-differencing matrix D of Eq. (8.1.8),

$$\mathcal{J}_{qp} = \|\mathbf{y} - \mathbf{x}\|_q^q + \lambda \|D\mathbf{x}\|_p^p = \min \tag{8.7.2}$$

where $\|\mathbf{x}\|_p$ denotes the ℓ_p norm of the vector $\mathbf{x} = [x_0, x_1, \cdots, x_{N-1}]^T$,

$$\|\mathbf{x}\|_p = \left[\sum_{n=0}^{N-1} |x_n|^p\right]^{\frac{1}{p}} \quad \Rightarrow \quad \|\mathbf{x}\|_p^p = \sum_{n=0}^{N-1} |x_n|^p$$

For $p = \infty$, we have instead,

$$\|\mathbf{x}\|_\infty = \max_{0\le n\le N-1} |x_n|$$

For $p = 0$, we define $\|\mathbf{x}\|_0$ as the *cardinality* of the vector $\mathbf{x}$, that is, the number of *nonzero* elements of $\mathbf{x}$. We note that $\|\mathbf{x}\|_p$ is a proper norm only for $p \ge 1$, however, the cases $0 \le p < 1$ have also been considered.

The case $\mathcal{J}_{11}$ was studied in [422,426] and formulated as a linear programming problem, the case $\mathcal{J}_{pp}$, including the ℓ_∞ norm case, $p = \infty$, was studied in [424], and the more general case, $\mathcal{J}_{qp}$, in [425]. More recently, the case $\mathcal{J}_{21}$, called ℓ_1 *trend filtering*, has been considered in [468] and has received a lot of attention [469–478].

Generally, the cases $\mathcal{J}_{2p}$ are examples of so-called ℓ_p-regularized least-squares problems, which have been studied very extensively in inverse problems, with renewed interest in sparse modeling, statistical learning, compressive sensing applications—a small and very incomplete set of references on regularization and sparse regularization methods is [479–590]. We discuss such regularization issues in more detail in Chap. 15. Next, we concentrate on the original $\mathcal{J}_{22}$ criterion, and the $\mathcal{J}_{21}$ and $\mathcal{J}_{20}$ criteria,

$$\begin{aligned} \mathcal{J}_{22} &= \|\mathbf{y} - \mathbf{x}\|_2^2 + \lambda \|D\mathbf{x}\|_2^2 = \min \\ \mathcal{J}_{21} &= \|\mathbf{y} - \mathbf{x}\|_2^2 + \lambda \|D\mathbf{x}\|_1 = \min \\ \mathcal{J}_{20} &= \|\mathbf{y} - \mathbf{x}\|_2^2 + \lambda \|D\mathbf{x}\|_0 = \min \end{aligned} \tag{8.7.3}$$

The $\mathcal{J}_{21}$ and $\mathcal{J}_{20}$ criteria tend to promote the *sparsity* of the regularizing term $D\mathbf{x}$, that is, $D\mathbf{x}$ will be a *sparse* vector consisting mostly of zeros with a few nonzero entries. Since $D\mathbf{x}$ represents the s-differenced signal, $\nabla^s x_n$, its piecewise vanishing implies that the trend x_n will be a piecewise polynomial of order $s-1$, with the polynomial pieces joining continuously at few break (or, kink) points where $\nabla^s x_n$ is nonzero.

This is similar to the spline smoothing case, except here the locations of the break points are determined dynamically by the solution of the optimization problem, whereas in the spline case they are at prescribed locations.

For differencing order $s = 2$, used in Hodrick-Prescott and ℓ_1-trend-filtering cases, the trend signal x_n will be a piecewise linear function of n, with a sparse number of slope changes. The case $s = 3$, used originally by Whittaker and Henderson, would correspond to piecewise parabolic segments in n. The case $s = 1$, corresponding to the original Bohlmann choice, results in a piecewise constant trend signal x_n. This case is known also as *total variation* minimization method and has been applied widely in image processing.

The $\mathcal{J}_{21}$ problem can be implemented easily in MATLAB with the CVX package.[†] The $\mathcal{J}_{20}$ problem, which produces the sparsest solution, can be solved by an *iterative reweighted ℓ_1-regularized* method [468], or alternatively, by an *iterative reweighted least-squares* method, and can also be used to solve the $\mathcal{J}_{21}$ and the $\mathcal{J}_{2p}$ problems.

There are several variants of the iterative reweighted least-squares (IRLS) method, [520–528,532,553,557,560,565,566], but the basic idea is to replace the ℓ_p norm with a weighted ℓ_2 norm, which can be solved iteratively. Given any real number $0 \le p \le 2$, let $q = 2 - p$, and note that for any real number $x \neq 0$, we can write,

$$|x|^p = \frac{|x|^2}{|x|^q} \approx \frac{|x|^2}{|x|^q + \varepsilon}$$

where ε is a sufficiently small positive number needed to also allow the case $x = 0$. Similarly, we can write for the ℓ_p-norm of a vector $\mathbf{x} \in \mathbb{R}^N$,

$$\begin{aligned} \|\mathbf{x}\|_p^p &= \sum_{i=0}^{N-1} |x_i|^p \approx \sum_{i=0}^{N-1} \frac{|x_i|^2}{|x_i|^q + \varepsilon} = \mathbf{x}^T W(\mathbf{x})\mathbf{x} \\ W(\mathbf{x}) &= \operatorname{diag}\left[\frac{1}{|\mathbf{x}|^q + \varepsilon}\right] = \operatorname{diag}\left[\frac{1}{|x_0|^q + \varepsilon}, \frac{1}{|x_1|^q + \varepsilon}, \ldots, \frac{1}{|x_{N-1}|^q + \varepsilon}\right] \end{aligned} \tag{8.7.4}$$

Then, the ℓ_p-regularized problem $\mathcal{J}_{2p}$ can be written in the form,

$$\mathcal{J} = ||\mathbf{y} - \mathbf{x}||_2^2 + \lambda ||D\mathbf{x}||_p^p = ||\mathbf{y} - \mathbf{x}||_2^2 + \lambda\, \mathbf{x}^T D^T W(D\mathbf{x})\, D\mathbf{x} = \min \tag{8.7.5}$$

which leads to to the following iterative algorithm,

$$\boxed{\begin{aligned} &\text{for } k = 1, 2, \ldots, K, \text{ do:} \\ &\quad W_{k-1} = W(D\mathbf{x}^{(k-1)}) \\ &\quad \mathbf{x}^{(k)} = \arg\min_{\mathbf{x}} ||\mathbf{y} - \mathbf{x}||_2^2 + \lambda\, \mathbf{x}^T D^T W_{k-1} D\mathbf{x} \end{aligned}} \quad \text{(IRLS)} \tag{8.7.6}$$

with the algorithm initialized to the ordinary least-squares solution of criterion $\mathcal{J}_{22}$,

$$\mathbf{x}^{(0)} = (I + \lambda D^T D)^{-1}\mathbf{y}$$

The solution of the optimization problem in (8.7.6) at the kth step is:

$$\mathbf{x}^{(k)} = (I + \lambda D^T W_{k-1} D)^{-1}\mathbf{y}$$

Thus, the choices $p = 0$ and $p = 1$ should resemble the solutions of the ℓ_0 and ℓ_1 regularized problems.

Example 8.7.1: *Global Warming Trends.* This is a continuation of Example 3.9.2 in which we compared several smoothing methods. Fig. 8.7.1 compares the Whittaker-Henderson trends for the ℓ_2, ℓ_1, and ℓ_0 cases, with $s = 2$, as well as the corresponding regularizing differenced signals, $\nabla^s x_n$.

[†] `http://cvxr.com/cvx/`

The ℓ_1 case was computed with the CVX package. The corresponding IRLS implementation is not shown since it produces virtually indistinguishable graphs from CVX.

The ℓ_0 case was implemented with the IRLS method and produced slightly sparser differenced signals as can be observed in the graphs. The MATLAB code used to generate these graphs is summarized below.

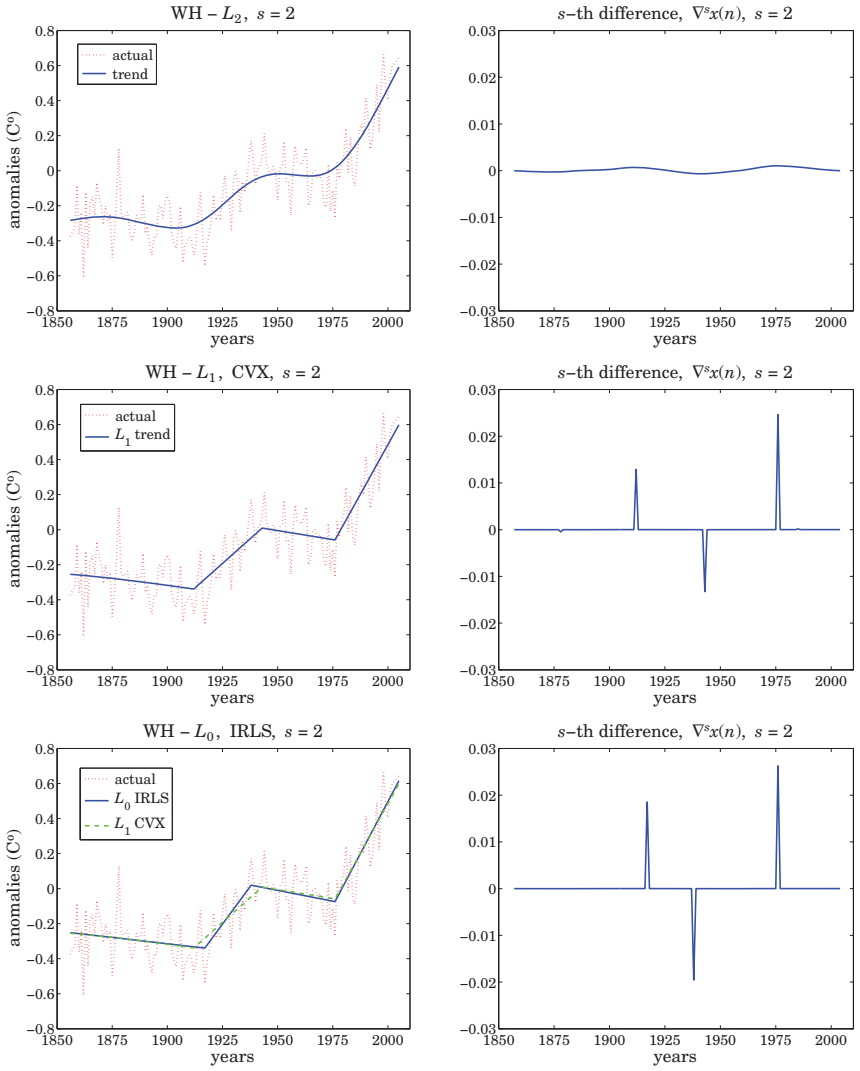


Fig. 8.7.1 Comparison of ℓ_2, ℓ_1, and ℓ_0 trends, and differenced signals for $s = 2$.

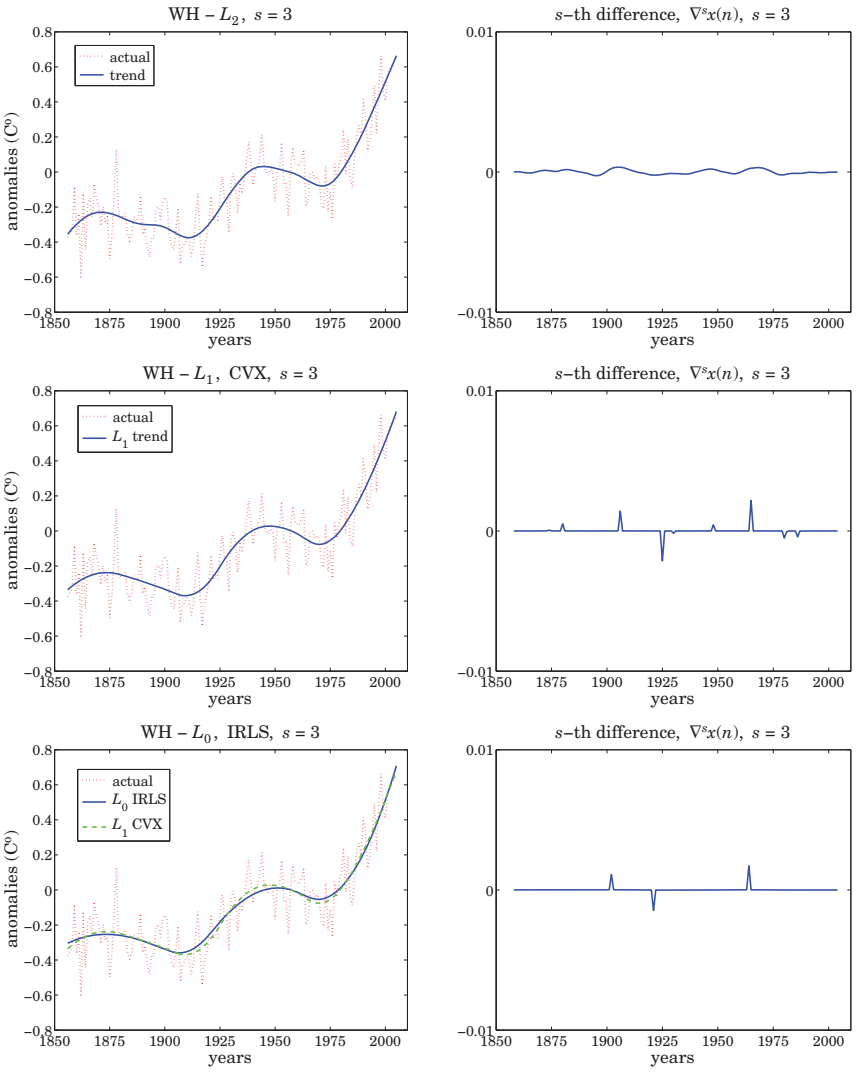


Fig. 8.7.2 Comparison of ℓ_2, ℓ_1, and ℓ_0 trends, and differenced signals for $s = 3$.

```
Y = loadfile('tavenh2v.dat');              % load temperature data file

n = Y(:,1); y = Y(:,14); N = length(y);    % extract dates and data

s = 2; Ds = diff(speye(N),s);              % (N-s)xN differencing matrix
ns = n(s:end-1);
```



```
la = 10000; x = whsm(y,la,s);              % Whittaker-Henderson with L2 norm

figure; plot(n,y,'r:', n,x,'b-');          % plot trend
figure; plot(ns, Ds*x,'b-');               % plot differenced trend

la = 10;                                   % Whittaker-Henderson with L1 norm
cvx_quiet(true);                           % CVX package, http:/cvxr.com/cvx/
cvx_begin
    variable x(N)
    minimize( sum_square(y-x) + la * norm(Ds*x,1) )
cvx_end

figure; plot(n,y,'r:', n,x,'b-');          % plot trend
figure; plot(ns, Ds*x,'b-');               % plot differenced trend

p = 0; q = 2 - p; epsilon = 1e-8;          % Whittaker-Henderson with L0 norm
I = speye(N); K = 10;                      % using K=10 IRLS iterations
la = 0.05;

x  = (I + la*Ds'*Ds) \ y;                  % initialize iteration

for k=1:K,                                 % IRLS iteration
   W = diag(1./(abs(Ds*x).^q + epsilon));
   xk = (I + la*Ds'*W*Ds) \ y;
   x = xk;
end

figure; plot(n,y,'r:', n,x,'b-');          % plot trend
figure; plot(ns, Ds*x,'b-');               % plot differenced trend
```

Fig. 8.7.2 compare the ℓ_2, ℓ_1, ℓ_0 cases for $s = 3$, which fits piecewise quadratic polynomials to the data. The ℓ_0 case is again the sparsest. (Color graphs online). □

8.8 Computer Project – US GDP Macroeconomic Data

In this project you will study the *Whittaker-Henderson smoothing method* formulated with the L_2 and L_1 norms, and apply it to the US GDP macroeconomic data. For a length-N signal, y_n, $0 \le n \le N-1$, the optimization criteria for determining a length-N smoothed signal x_n are,

$$\begin{aligned}(L_2): \quad & \mathcal{J} = \sum_{n=0}^{N-1} |\, y_n - x_n \,|^2 + \lambda \sum_{n=s}^{N-1} |\, \nabla^s x_n \,|^2 = \|\mathbf{y}-\mathbf{x}\|_2^2 + \lambda \, \|D_s \mathbf{x}\|_2^2 = \min \\ (L_1): \quad & \mathcal{J} = \sum_{n=0}^{N-1} |\, y_n - x_n \,|^2 + \lambda \sum_{n=s}^{N-1} |\, \nabla^s x_n \,| \;\; = \|\mathbf{y}-\mathbf{x}\|_2^2 + \lambda \, \|D_s \mathbf{x}\|_1 = \min \end{aligned} \tag{8.8.1}$$

where D_s is the $(N-s)\times N$ convolution matrix corresponding the s-difference operator ∇^s. It can be constructed in MATLAB by,

```
Ds = diff(eye(N),s);        % or, in sparse form, Ds = diff(speye(N),s);
```

The solution of problem (L_2) is straightforward:

$$\mathbf{x} = (I + \lambda D_s^T D_s)^{-1} \mathbf{y} \tag{8.8.2}$$

The solution of problem (L_1) can be obtained with the CVX package as follows:

```
cvx_begin
   variable x(N)
   minimize( sum_square(x-y) + lambda * norm(Ds*x,1) );
cvx_end
```

It can also be solved with the *iterative reweighted least-squares* (IRLS) algorithm, as discussed in Sec. 8.7.

The second column of the OSP data file, `USGDP_Inv.dat`, represents the quarterly US GDP for private investment in billions of dollars. Read this column with the help of the function `loadfile` and then take its log:

```
Y = loadfile('USGDP_Inv.dat');
y = log10(Y(:,2) * 1e9);
```

These data represent a prototypical example for the application of Whittaker-Henderson filters, referred to in this context as Hodrick-Prescott filters.

a. Choose difference order $s = 2$ and regularization parameter $\lambda = 1600$. Solve the Whittaker-Henderson problem (L_2) and plot the solution $\mathbf{x}$ together with the actual data $\mathbf{y}$.

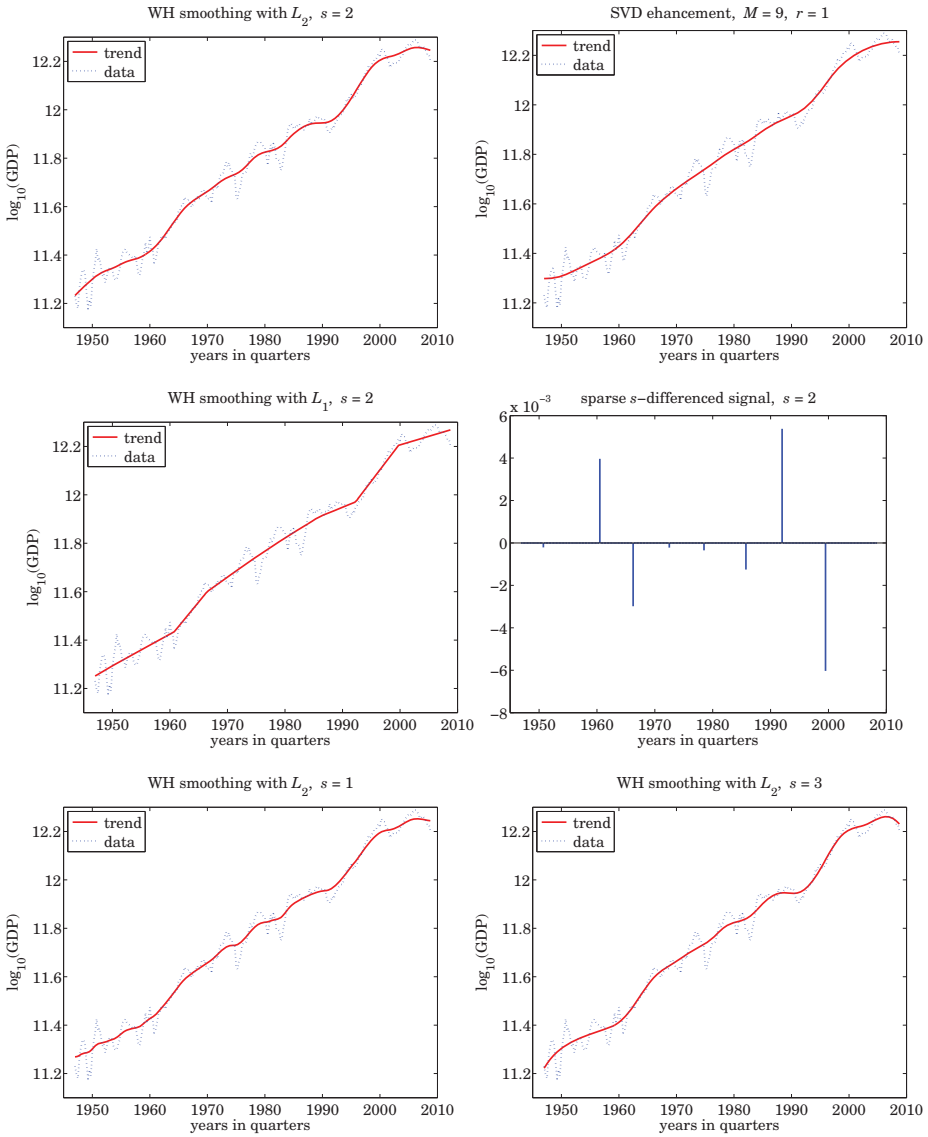
b. Calculate the SVD enhanced version of $\mathbf{y}$, by the following steps: (i) remove and save the mean from $\mathbf{y}$, (ii) form its forward/backward data matrix using an embedding order of $M = 9$, (iii) subject it to $K = 8$ SVD enhancement iterations using rank $r = 1$, (iv) extract the enhanced signal from the enhanced data matrix, and (v) add the mean that was removed. Plot the resulting enhanced signal together with $\mathbf{y}$. The MATLAB steps are summarized in Sec. 15.17 (use, `type=2`, for the F/B case).

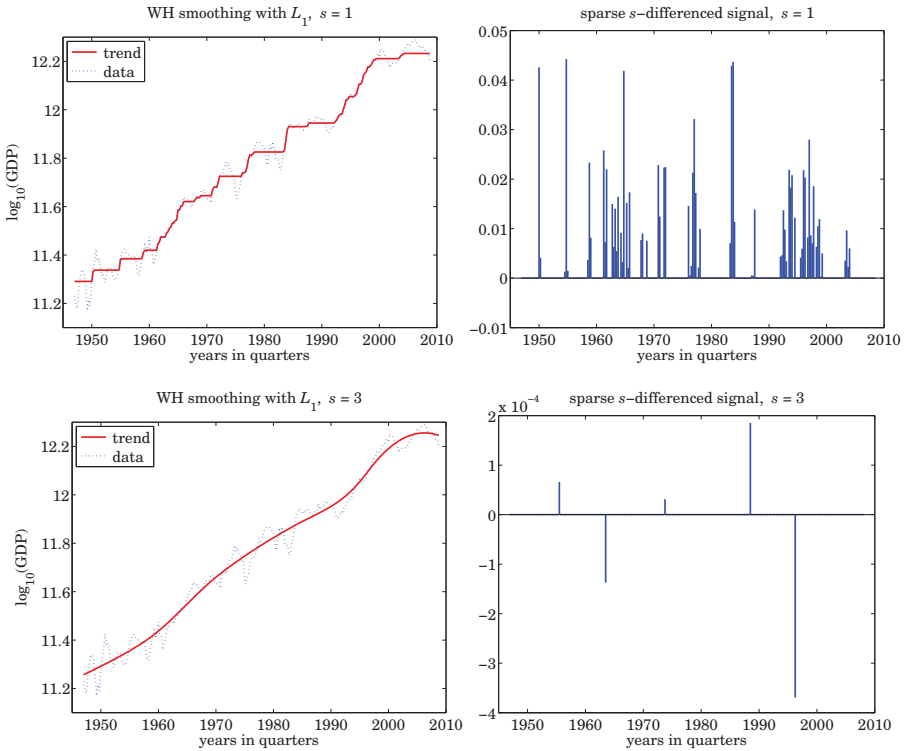
c. For the value $\lambda_1 = \lambda/480 = 1600/480$ and difference order $s = 2$, solve problem (L_1), and plot the solution $\mathbf{x}$ together with the actual data $\mathbf{y}$. Moreover, on a separate graph, plot the differenced signal $D_s\mathbf{x}$ using a stem plot and observe its sparseness, which means that $\mathbf{x}$ is piece-wise linear. The particular choice for λ_1 was made in order for the (L_2) and (L_1) problems to have comparable RMS errors.

d. Repeat parts (a) and (c) for $s = 1$ and regularization parameter $\lambda = 60.65$ for the (L_2) problem (justified in Sec. 8.3), and $\lambda_1 = 1$ for the (L_1) problem, chosen to achieve comparable RMS errors. Notice how the (L_1) problem results in a piece-wise constant fit. But $D_s\mathbf{x}$ is not as sparse because $s = 1$ is not really a good choice.

   The $s = 1$ case is an example of the so-called *total-variation minimization* method, used widely in image processing.

e. Repeat parts (a) and (c) for $s = 3$ and regularization parameter $\lambda = 41640.16$ for the (L_2) problem (justified in Sec. 8.3), and $\lambda_1 = \lambda/1000$ for the (L_1) problem, chosen to achieve comparable RMS errors. Here, the (L_1) problem will result in piece-wise quadratic polynomial fits. Some example graphs are shown below.





8.9 Problems

8.1 For the case $s = 1$, show that the Whittaker-Henderson filter has poles z_1, $1/z_1$, where

$$z_1 = e^{-\alpha}, \quad \alpha = 2\,\text{asinh}\left(\frac{1}{2\sqrt{\lambda}}\right)$$

For the case $s = 2$, show that the filter has poles $\{z_1, z_1^*, 1/z_1, 1/z_1^*\}$, where

$$z_1 = \left(\sqrt{1-ja^2} + jae^{j\pi/4}\right)^2 = \frac{1}{2}D^2\left(1-\frac{a}{D}\right)^2\left(1+j\frac{a}{D}\right)^2, \quad D = \sqrt{1+\sqrt{1+a^4}}, \quad a = \frac{1}{2\lambda^{1/4}}$$

Show that in both cases $|z_1| < 1$.

8.2 Determine explicit expressions in terms of λ for the quantities σ^2 and z_1 that appear in the factorization of the denominator of the Hodrick-Prescott filter:

$$1 + \lambda(1-z^{-1})^2(1-z)^2 = \sigma^2(1-z_1z^{-1})(1-z_1^*z^{-1})(1-z_1z)(1-z_1^*z)$$

What are the numerical values of σ^2, z_1 for $\lambda = 1600$? What are the values of the coefficients of the second-order filter $(1 - 2\,\text{Re}(z_1)z^{-1} + |z_1|^2z^{-2})$?

8.3 Consider the performance index (8.6.1) for a regularized deconvolution problem. Making the enough assumptions, show that the performance index can be written in terms of frequency responses as follows,

$$\begin{aligned}\mathcal{J} &= \sum_n |y_n - f_n * x_n|^2 + \lambda \sum_n |d_n * x_n|^2 \\ &= \int_{-\pi}^{\pi} \Big[\,|\,Y(\omega) - F(\omega)X(\omega)\,|^2 + \lambda\,|\,D(\omega)X(\omega)\,|^2\Big]\,\frac{d\omega}{2\pi} = \min\end{aligned}$$

Determine the optimum $X(\omega)$ that minimizes this index. Then, show that the corresponding optimum deconvolution filter $H(z) = X(z)/Y(z)$ is given by:

$$H(z) = \frac{F(z^{-1})}{F(z)F(z^{-1}) + \lambda D(z)D(z^{-1})}$$

What would be the stochastic state-space model for x_n, y_n that has this $H(z)$ as its optimum double-sided (unrealizable) Wiener filter for estimating x_n from y_n?