The filter bank approach

Consider $N$ samples of $y(t)$. The power of the filter output is

$$E\left\{ h^*_\omega y_L(t) \right\}^2 = h^*_\omega E \{ y_L(t) y_L^*(t) \} h_\omega = h^*_\omega R_y h_\omega,$$

where $h_\omega$ is the $L$-tap bandpass filter centered at frequency $\omega$, and

$$y_L(t) = \left[ y(t) \ldots y(t+L-1) \right]^T,$$

for $t = 0, \ldots, M - 1$, where $M = N - L + 1$. Generally, the data covariance matrix, $R_y$, is unknown. For now, assume that $y(t)$ is white, i.e., $R_y = \sigma_y^2 I$. Under this assumption, we design the bandpass filter such that the power of the filter output is minimized,

$$h_\omega = \arg \min_{h_\omega} h^*_\omega R_y h_\omega = \arg \min_{h_\omega} h^*_\omega h_\omega,$$

while enforcing that the filter have unit gain for frequency $\omega$, i.e.,

$$H(\omega) = \sum_{k=0}^{L-1} h_k e^{-i\omega k} = 1.$$
A simple approach

Thus,
\[ h_\omega = \arg \min_{h_\omega} h_\omega^* h_\omega \quad \text{subj. to} \quad h_\omega^* a_L(\omega) = \sum_{k=0}^{L-1} h_k e^{-i \omega k} = 1, \]

where
\[ a_L(\omega) = \begin{bmatrix} 1 & e^{i \omega} & \cdots & e^{i (L-1) \omega} \end{bmatrix}^T, \]

implying that the filter is found as
\[ h_\omega = a_L(\omega)/L \]

yielding
\[ \hat{\phi}(\omega) = h_\omega^* R_\omega h_\omega = a_L^*(\omega) R_\omega a_L(\omega)/L^2 = \hat{\phi}_M(\omega) \]

This is the periodogram estimate.

The classical Capon algorithm

Another way to design the filter is to minimize the output power, while enforcing a unit gain for the frequency of interest, i.e.,
\[ h_\omega = \arg \min_{h_\omega} h_\omega^* R_\omega h_\omega \quad \text{subject to} \quad h_\omega^* a_\omega = 1 \]

The solution is found as
\[ h_\omega = \frac{R_\omega a_L(\omega)}{a_L^*(\omega) R_\omega a_L(\omega)}, \]

yielding the estimated output power,
\[ \hat{\phi}_{\text{PSC}}(\omega) = h_\omega^* R_\omega h_\omega = \frac{1}{a_L^*(\omega) R_\omega a_L(\omega)} \]

termed the classical or Power Spectrum Capon (PSC) estimate. The designed filter will place nulls at the locations of energy other than \( \omega \).

Implementation

To form an efficient implementation, we make use of the Gohberg-Semencul formula stating that the inverse of a \( n \times n \) Toeplitz matrix, \( R_\omega \), can be computed as
\[ \sigma_{n-1}^2 R_{\omega}^{-1} = \begin{bmatrix} 1 & 0 & \cdots & 0 & a_1^* & \cdots & a_n^* \\ a_1 & 1 & \cdots & 0 & a_1^* & \cdots & a_n^* \\ \vdots & \vdots & \ddots & \vdots & \vdots & \cdots & \vdots \\ a_{n-1} & \cdots & a_1 & 1 & 0 & a_1^* & \cdots \\ 0 & a_1^* & \cdots & a_{n-1} & 0 & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \cdots & \vdots \\ 0 & \cdots & a_1^* & \cdots & a_n^* & 0 & \cdots \end{bmatrix}, \]

where \( \sigma^2 \) is the prediction error and with \( a_k \) being AR coefficients.

Remarks:
(i) The Musicus algorithm couples PSC with the AR spectral estimate, and the PSC can be seen as an average of \( \hat{\phi}_M^k(\omega) \) of order \( k = 0, \ldots, n \).
(ii) The Musicus implementation can, for \( P \) frequency points, be done in \( \mathcal{O}(n^2 + P \log P) \) operations.
(iii) The choice of algorithm used to compute the AR coefficients will significantly affect the spectral estimates.
The amplitude spectrum Capon algorithm

Recalling the notion of the amplitude spectrum, we model the data as
\[ y(t) = \alpha_\omega e^{i\omega t} + \epsilon(t), \quad t = 0, \ldots, N - 1, \]
where \( \alpha_\omega \) is the (complex-valued) amplitude of a sinusoid at \( \omega \), and \( \epsilon(t) \) an additive colored noise containing all the remaining signal. In vector form,
\[
y_L(t) = \begin{bmatrix} y(t) & \ldots & y(t + L - 1) \end{bmatrix}^T = \alpha_\omega \begin{bmatrix} e^{i\omega t} & \ldots & e^{i\omega(t+L-1)} \end{bmatrix}^T + \epsilon_L(t),
\]
where \( \epsilon_L(t) \) is a vector formed from the noise sequence.

Wishing to minimize the contribution from the noise sequence, we design the filter such that

\[ h_\omega = \arg \min_{h_\omega} h_\omega^* Q_\omega h_\omega \quad \text{subject to} \quad h_\omega^* a_L(\omega) = 1, \]

where \( Q_\omega = E[\epsilon_L(t)\epsilon_L^*(t)] \). Note that
\[
R_\omega = E[y_L(t)y_L^*(t)] = E \left\{ |\alpha_\omega a_L(\omega)e^{i\omega t} + \epsilon_L(t)|^2 \right\} = |\alpha_\omega|^2 a_L(\omega)a_L^*(\omega) + Q_\omega,
\]
as we assume the noise to be independent from the signal. Thus,
\[
Q_\omega = R_\omega - |\alpha_\omega|^2 a_L(\omega)a_L^*(\omega)
\]
is a reasonable estimate of \( Q_\omega \). But,
\[
h_\omega^* Q_\omega h_\omega = h_\omega^* \left[ R_\omega - |\alpha_\omega|^2 a_L(\omega)a_L^*(\omega) \right] = h_\omega^* R_\omega h_\omega - |\alpha_\omega|^2
\]
yielding the classical Capon filter!

The APES algorithm

Nothing prevents us from finding another estimate of \( Q_\omega \). Consider
\[
y_L(t) = \alpha_\omega a_L(\omega)e^{i\omega t} + \epsilon_L(t) = z_L(\omega)e^{i\omega t} + \epsilon_L(t),
\]
which yields the unstructured least-squares estimate
\[
z_L(\omega) = \frac{1}{M} \sum_{t=0}^{M-1} y_L(t)e^{-i\omega t} \equiv Y_\omega
\]
Thus, we can estimate the noise covariance matrix as
\[
Q_\omega = R_\omega - Y_\omega Y_\omega^*
\]
In this case,
\[
\min_{h_\omega} h_\omega^* Q_\omega h_\omega \neq \min_{h_\omega} h_\omega^* R_\omega h_\omega
\]
The APES algorithm

We obtain the minimizing filter as

\[ h_\omega = \hat{Q}_L(\omega)^{-1} \frac{\hat{a}_L(\omega)}{\hat{a}_L(\omega) Q_L^* \hat{a}_L(\omega)} \]

yielding the Amplitude and Phase Estimation (APES) spectral estimate as

\[ \hat{\phi}_{\text{APES}}(\omega) = \frac{\hat{a}_L(\omega)^* Q^{-1}_L Y_\omega}{\hat{a}_L(\omega)^* Q_L^* \hat{a}_L(\omega)} \]

Using the matrix inversion lemma, one can rewrite \( \hat{\phi}_{\text{APES}}(\omega) \) as

\[ \hat{\phi}_{\text{APES}}(\omega) = \left( \mu^* \mu - \mu^* \nu \nu^* \mu + |\mu^* \nu|^2 \right)^{1/2} \]

where \( \mu = \hat{R}^{-1/2}_y a_L(\omega) \) and \( \nu = \hat{R}^{1/2}_x Y_\omega \).

Remarks

(i) One can compute both ASC and APES very efficiently.

(ii) In general, the ASC estimator has better resolution than the PSC and APES estimators, whereas the APES estimator has a better amplitude estimate.

(iii) The CAPES approach combines these benefits, estimating the frequencies using ASC, and the amplitudes using APES.

(iv) The estimators benefit from using \( \hat{R}_f b \) instead of \( \hat{R}_f \).

(v) It is often convenient to rewrite the APES criteria as

\[ \min_{h_\omega, \nu_\omega} \frac{1}{M} \sum_{t=0}^{M-1} |h_\omega^* Y(L(t) - \alpha_\omega e^{-i \omega t})|^2 \]

Displacement structures

From an implementation perspective, it is interesting to note that several of the filterbank algorithms depend on the form \( L_a, a, \) where \( a \) is a Fourier vector, and

\[ R_c^{-1} = L_c^* L_c \]

It is therefore of interest to efficiently evaluate \( L_c \). This can be done by exploiting the displacement structure of \( R_c \).

The displacement rank is defined as the rank, \( r \), of the matrix \( \nabla R_c \), where

\[ \nabla R_c = R_c - Z R_c Z^* = G J G^* \]

where \( Z \) is a lower triangular \( n \times n \) matrix, \( G \) is an \( n \times r \) matrix, and \( J \) is a signature matrix. Here, \( r \) provides a measure of the structure in \( R_c \).
Displacement structures

For a Toeplitz matrix, rank(\(\nabla R_c\)) = 2,

\[ \nabla R_c = R_c - ZR_cZ^* \]

where \(Z\) is a lower shift matrix. Furthermore, it can be shown that \(R_c^{-1}\) has the same displacement rank as \(R_c\).

Using the generalized Schur recursion, \(L_c\) can be computed in \(O(rt^2)\) operations. Similarly, one may form a computationally efficient time-updating of \(L_c\).


The IAA algorithm

The IAA algorithm

The IAA algorithm

Let \(p_k(i)\) denote the estimate of \(p_k\) at iteration \(i\), and \(R(i)\) the matrix formed from \(p_k(i)\). Then the iterative adaptive approach (IAA) updates the estimates as

\[ p_k(i+1) = \frac{|a|^2 R^{-1}(i)\bar{y}y^T}{a_k|a|^2 R^{-1}(i)a_k} \]

for \(k = 1, \ldots, K + N\)

with the initial estimates being set to the identity matrix. Typically, \(m = 10-15\) iterations are sufficient.

Note that the IAA estimate is computationally cumbersome, but that one may form computationally efficient algorithms by exploring the strong structure of the estimate. In particular,

\[ g = R^{-1}(i)\bar{y} \]

can be computed via a Levinson-style algorithm. The denominator can be formed using Golberg-Semencul representations and the FFT. The complexity of the resulting implementation is about

\[ C_{IAA} = m[N^2 + 12\phi(2N) + 3\phi(K)] \]

where \(\phi(M)\) denotes the complexity of computing an FFT of length \(M\).

The IAA algorithm

Let \( p_k(i) \) denote the estimate of \( p_k \) at iteration \( i \), and \( R(i) \) the matrix formed from \( p_k(i) \). Then the iterative adaptive approach (IAA) updates the estimates as

\[
    p_k(i + 1) = \frac{\|a_k R^{-1}(i) y\|^2}{\|a_k R^{-1}(i) a_k\|} \quad \text{for} \quad k = 1, \ldots, K + N
\]

with the initial estimates being set to the identity matrix. Typically, \( m = 10 - 15 \) iterations are sufficient.

Note that the IAA estimate is computationally cumbersome, but that one may form computationally efficient algorithms by exploring the strong structure of the estimate. In particular, \( g = R^{-1}(i) y \) can be computed via a Levinson-style algorithm. The denominator can be formed using Gohberg-Semencul representations and the FFT. The complexity of the resulting implementation is about

\[
    C_{I A A} = m[N^2 + 12\phi(2N) + 3\phi(K)]
\]

where \( \phi(M) \) denotes the complexity of computing an FFT of length \( M \).

(The SLIM algorithm

Let us now assume that \( \sigma_1 = \ldots = \sigma_N = \sigma \) and that the information vector

\[
    s = [s_1 \cdots s_K]^T
\]

is sparse, containing only a few non-zero elements. A typical way to enforce this sparsity is to form the estimate using a LASSO formulation as

\[
    \min_{\{a_k\}} \| y - \sum_{k=1}^{K} a_k s_k \|^2 \quad \text{subj. to} \quad |s_k| = \sum_{k=1}^{K} |s_k| < \eta.
\]

where \( \eta \) is a user defined threshold. This is a convex minimization problem.

An alternative is to use another sparsity inducing penalty function. On such alternative is to use the \( \ell_q \) norm

\[
    |s|^q = \sum_{k=1}^{K} |s_k|^q \quad \text{for} \quad 0 < q \leq 1,
\]

which has been found to yield improved estimates. However, the minimization is then no longer convex.
The SLIM algorithm

One way to include such a constraint is to form the maximum a posteriori (MAP) estimate

$$\max_{\mathbf{y}, \mathbf{s}, \eta} f(\mathbf{y} | \mathbf{s}, \eta) f(\mathbf{s})/f(\eta) = \max_{\mathbf{s}, \eta} \frac{1}{(\pi \eta)^{N/2}} e^{-\frac{1}{2} \eta \mathbf{y}^T \mathbf{y}} \prod_{n=1}^{d} e^{-\frac{1}{2} (|s_n|^q - 1)}$$

where $f(\mathbf{s})$ is a sparsity inducing prior (if $q = 1$, it becomes Laplacian), whereas $\eta$ is an improper prior, implying that it has equal probability in the range $[0, \infty)$, and

$$\mathbf{A} = [a_1 \ldots a_d]$$

Taking the negative logarithm yields

$$\min_{\mathbf{s}, \eta} d \log(\eta) + \frac{1}{\eta} \mathbf{y}^T \mathbf{y} - 2 \sum_{n=1}^{d} |s_n|^q - 2 \sum_{n=1}^{d} \mathbf{y}^T a_n$$

which can be solved via a cyclic minimization.

---

The SPICE algorithm

We again consider the semi-parametric model, with

$$\mathbf{y} = \sum_{\ell=1}^{d} \mathbf{a}(\omega_{\ell}) \mathbf{x}_\ell + \mathbf{e} = \begin{bmatrix} a_1 & \ldots & a_d \end{bmatrix} \mathbf{x}$$

Then,

$$\mathbf{R} = E(\mathbf{y} \mathbf{y}^*) = \mathbf{A} \mathbf{A}^*,$$

where (as for IAA)

$$\mathbf{P} = \begin{bmatrix} |s_1|^2 & 0 & \ldots & 0 \\ 0 & |s_2|^2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & |s_N|^2 \end{bmatrix} \overset{\text{diag}}{=} \begin{bmatrix} p_1 & \ldots & p_{K+N} \end{bmatrix}$$
The SPICE algorithm

The SParse Iterative Covariance-based Estimation (SPICE) method finds the estimated $P$ using the weighted covariance fitting criteria

$$\{p\} = \arg \min_p \left\| R^{-1/2} (R - yy^*) \right\|_F^2 = \arg \min_p y^* R^{-1} y + \sum_{k=1}^{N+d} w_k^2 p_k$$

where

$$w_k = \frac{\| a_k \|}{\| y \|}$$

This minimization is convex and can be solved using a second-order cone program (SOCP).

(Stoica, Babu & Li, TSP, Jan 2011)

Example

Estimating the coherence function

The magnitude square coherence (MSC) spectrum between two signals is formed as

$$\gamma_\omega = \frac{|\phi_{xy}(\omega)|^2}{\phi_{xx}(\omega) \phi_{yy}(\omega)}$$

where

$$\phi_{xy}(\omega) = \sum_{k=-\infty}^{\infty} r_{xy}(k) e^{-j\omega k},$$

for $k, p = 1, 2$, with

$$r_{xy}(l) = \mathbb{E}(x_k(n) x_p^*(n - l)).$$

To estimate the MSC, one thus needs to estimate the auto- and cross-spectra.
Estimating the coherence function

Forming the auto- and cross-spectral estimates as
\[ \phi_{xx,x}(\omega) = \frac{h^*_k R_{xx,k} h_k}{L}, \]
where
\[ h^*_C = \frac{R^{-1}_{xx,k} a_x}{a^*_x R_{xx,k} a_x}, \]
\[ h^*_A = \frac{Q^{-1}_{xx,k} a_x}{a^*_x Q_{xx,k} a_x}, \]
yielding
\[ \gamma^C = \frac{|a^*_x R^{-1}_{xx,k} R_{xx,k} a_x|^2}{|a^*_x R_{xx,k} a_x| |a^*_x R_{xx,k} a_x|^2}, \]
\[ \gamma^A = \frac{|a^*_x Q^{-1}_{xx,k} R_{xx,k} a_x|^2}{\prod_{k=1}^2 |a^*_x Q_{xx,k} a_x | |a^*_x Q_{xx,k} a_x|^2}, \]

Introducing
\[ R^{-1}_{xx,k} = L^* a_x \]
and
\[ \mu_k = L a_x \text{ and } \tilde{\mu}_k = \Psi^* \mu_k \]
allows the forms
\[ \gamma^C = \frac{[\mu^*_k L a_x R_{xx,k} a_x]^2}{|\mu_k|^2 |\mu_k|^2}, \]
\[ \gamma^A = \frac{[\tilde{\mu}^*_k L a_x R_{xx,k} a_x]^2}{|\mu_k|^2 |\mu_k|^2}, \]
which may be implemented efficiently.

(Jakobsson, Alty & Benesty, IEEE T SP, 2007)

Example

---

Example
Application: Nuclear magnetic resonance

The NMR signal can be well-modelled as a sum of damped sinusoids in noise,

\[ y(t) = \sum_{k=1}^{n} a_k e^{-\alpha_k t + i\omega_k t} + \epsilon(t), \]

where \( \alpha_k \) is some (positive) damping factor to be estimated. This can be done by forming a two-dimensional filterbank, over the unknown \( \alpha \) and \( \omega \), such that

\[ h_{\omega,\alpha}, y_L(t) = a e^{-\alpha t + i\omega t} + w(t). \]

This yields the APES minimization as

\[ \min_{h_{\omega,\alpha}, a} \sum_{t=1}^{M} |h_{\omega,\alpha} y_L(t) - a e^{-\alpha t + i\omega t}|^2 \]

The estimates are found as the peaks in the \( \alpha, \omega \) plane of

\[ \hat{a}(\alpha, \omega) = h_{\omega,\alpha}^* Y_{\omega,\alpha} = h_{\omega,\alpha}^* \left( e^{2\alpha - 2\alpha t^2} \right) \sum_{t=1}^{M} |y(t) e^{-\alpha t}| e^{-i\omega t} \]

where the ASC and APES filters are found as

\[ h_{\omega,\alpha}^{ASC} = \frac{R_{\omega,\alpha}^{-1} s_{\omega,\alpha}}{s_{\omega,\alpha} R_{\omega,\alpha} s_{\omega,\alpha}} \]

\[ h_{\omega,\alpha}^{APES} = \frac{Q_{\omega,\alpha}^{-1} s_{\omega,\alpha}}{s_{\omega,\alpha} Q_{\omega,\alpha} s_{\omega,\alpha}} \]

where the estimated \( R \) and \( Q \) are obtained similarly to before, and

\[ s_{\omega,\alpha} = \text{diag} \left( 1, e^{-\alpha}, \ldots, e^{-\alpha(L-1)} \right) \left[ \begin{array}{cccc} 1 & \epsilon \omega & \ldots & \epsilon \omega(L-1) \end{array} \right]^T \]

Current implementation cannot exploit the \( \alpha \) structure.

Example

One can also form semi-parametric sparsity-inducing estimators. This is a topic we are currently working on.

(Svard, Adalbjornsson, Jakobsson, 14)