The University of Kansas

Information and Telecommunication Technology Center

Technical Report

A Multipath Channel Estimation Algorithm Using a Kalman Filter

Rupul Safaya, Sam Shanmugan and
Joseph Evans

ITTC-FY2001-TR-13380-09

July 2000

Project Sponsor:
Information Technology Office
of the
Defense Advanced Research Projects Agency

Abstract:

Data based channel estimation methods offer low complexity and good performance and are thus quite widely used in communications systems today. But they are also wasteful of bandwidth since they use training sequences to estimate the channel.

This thesis presents a method of improving the channel estimate without increasing the length of the training sequence. This method uses the underlying channel model and the available data based estimate, to implement the channel estimation algorithm in the form of a Kalman filter. The Kalman filter based channel estimator leads to a significant gain in performance as compared to the data-only estimator. The Kalman filter also allows us to predict the state of the before the frame is actually received.

Table Of Contents:

Table Of Figures:

1 Introduction:

1.1 A brief background of wireless communications.

When land line based telephony systems were first introduced in the earlier part of this century, they allowed people to communicate almost instantaneously over large distances at a reasonable cost for the very first time. Indeed it was the advent of telephony and the advance in communications technology that has drastically influenced the way we live and our outlook on life.

If land based telephony ushered in the age of communications; then wireless communications is its legitimate successor. Though land based systems will go on providing backbone connectivity for a long time to come, it has become increasingly clear in the last decade or so, that for the last hop in the information chain (that links the user to his information source), the user prefers wireless access. The reason for this is simply convenience. Nobody wants his or her mobility restricted. Wireless access allows the user the freedom to be mobile. Apart from user satisfaction, there is a very legitimate justification of wireless connectivity from the service provider's point of view. There are many areas in the world that are still inaccessible to land line systems due to their remoteness or because of intervening inhospitable terrain. Wireless systems are a very practical alternate in such a scenario to replace or supplement the backbone landlines.

The last and perhaps most important factor in the drive towards mobile telephony is simply economics *[1]*. For the first 35 years since its first commercial deployment, wireless systems saw little market penetration due to the high cost and the technological challenges involved. But in the last fifteen years, cellular telephony alone (not including paging, amateur radio, terrestrial microwave radio systems) has been growing at rates similar to that of television and the automobile as seen in the figure below. So we see that wireless communications holds enough promise to be the technology that drives our lifestyle and indeed our culture into the next millenium.

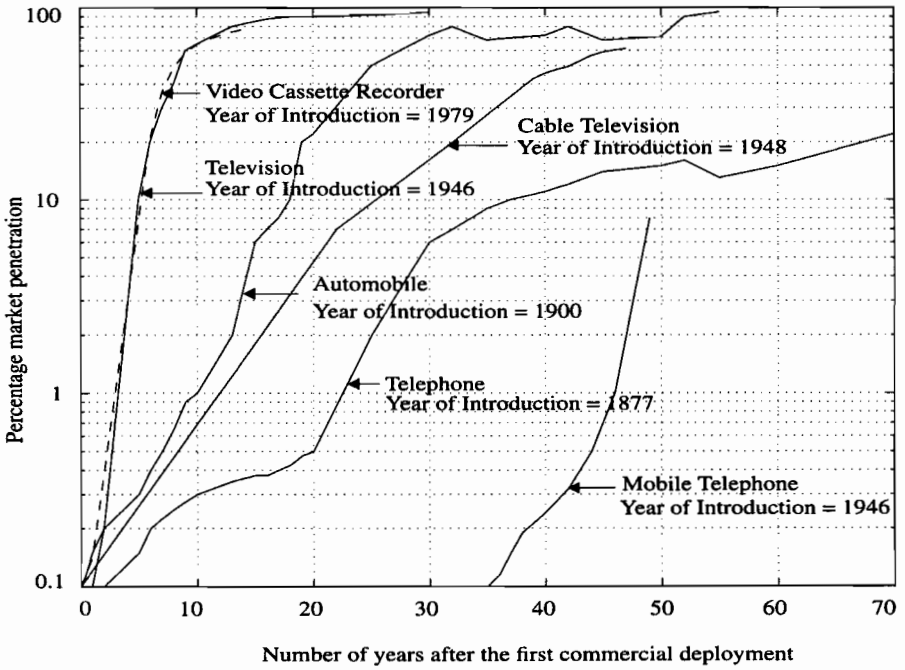

Figure 1-1: Mobile Telephony growth compared with other inventions this century

As with most things this promising, wireless communications opens a whole Pandora's box of complications. A radio signal transmitted from a base station to a mobile in a typical urban environment, exhibits variations in both received amplitude and phase/frequency. The change in amplitude us usually manifest in the form of sharp drops in signal level called *Fades*. Fades of 40dB or more below the mean signal level are common with successive minima occurring every few inches of the motion of the mobile. A vehicle traveling at 60 miles/hr can easily experience fades at the rate of 100Hz, thus distorting speech *[1]*.

Each radio wave received at the mobile has an associated Doppler shift that depends on the mobile velocity, the carrier frequency and the angle between the velocity vector and the wave propagation vector. This manifests itself as a random variance in the instantaneous frequency of the received signal, causing further distortion.

These obstacles may seem to defy any attempt at a systematic interpretation or analysis. However, starting from a model that takes into account the buildings and other structures in the vicinity of the mobile, that affect the signal, we can successfully predict many of the observed properties of the received signal by utilizing statistical techniques.

1.2 What is Channel Estimation?

Before we approach the problem of predicting and analyzing the observable properties of transmission, we must first define what we mean by a channel. In its most general sense, a *channel* can describe everything from the source to the sink of a radio signal *[3]*. This includes the physical medium (free space, fiber, waveguides etc.) between the transmitter and the receiver through which the signal propagates The word channel refers to this physical medium throughout this work. An essential feature of any physical medium is, that the transmitted signal is received at the receiver, corrupted in a variety of ways by frequency and phase distortion, inter symbol interference and thermal noise.

A *channel model* on the other hand can be thought of as a mathematical representation of the transfer characteristics of this physical medium. This model could be based on some known underlying physical phenomenon or it could be formed by fitting the best mathematical / statistical model on the observed channel behavior. Most channel models are formulated by observing the characteristics of the received signals for each specific environment. Different mathematical models that explain the received signal are then fit over the accumulated data. Usually the one that best explains the behavior of the received signal is used to model the given physical channel.

Channel estimation is simply defined as the process of characterizing the effect of the physical channel on the input sequence. If the channel is assumed to be linear, the channel estimate is simply the estimate of the impulse response of the system. It must be stressed once more that channel estimation is only a mathematical representation of what is truly happening. A "good" channel estimate is one where some sort of error minimization criteria is satisfied (e.g. MMSE).

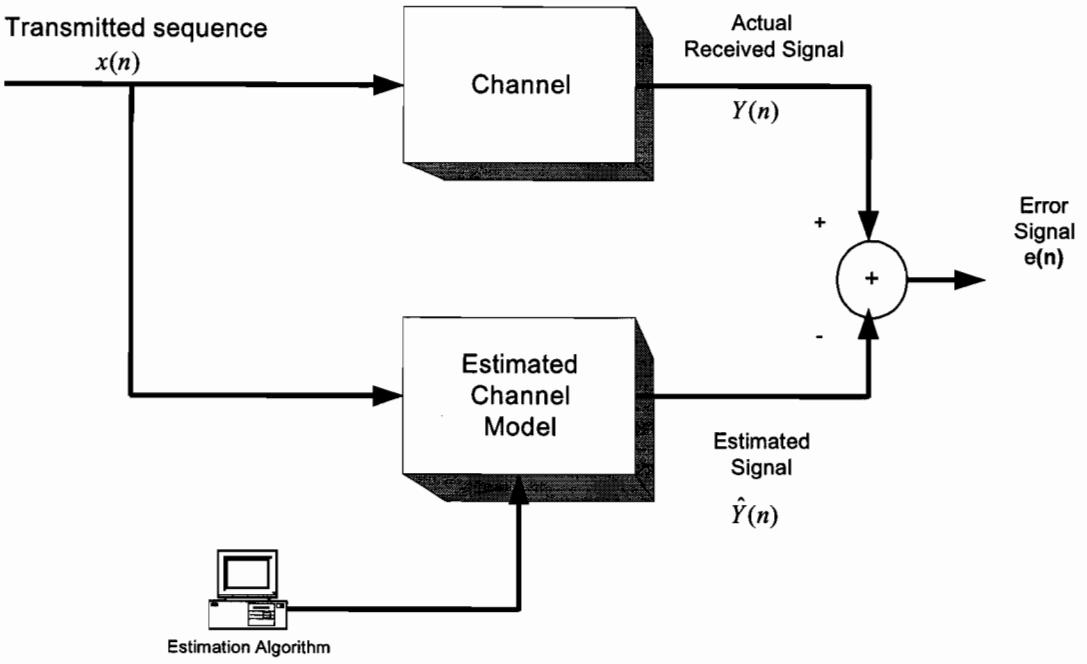

Figure 1-2: A general Channel Estimation Procedure.

In the figure above *e(n)* is the estimation error. The aim of most channel estimation algorithms is to minimizethe mean squared error (MMSE), $E[e^2(n)]$ while utilizing as little computational resources as possible in the estimation process.

1.3 Why Channel Estimation?

Channel estimation algorithms allow the receiver to approximate the impulse response of the channel and explain the behavior of the channel. This knowledge of the channel's behavior is wellutilized in modern radio communications. Adaptive channel equalizers utilize channel estimates to overcome the effects of inter symbol interference. Diversity techniques (for e.g. the IS-95 Rake receiver) utilize the channel estimate to implement a matched filter such that the receiver is optimally matched to the received signal instead of the transmitted one. Maximum likelihood detectors utilize channel estimates to minimize the error probability. One of the most important benefits of channel

estimation is that it allows the implementation of coherent demodulation. Coherent demodulation requires the knowledge the phase of the signal. This can be accomplished by using channel estimation techniques.

1.4 Training Sequences vs. Blind Methods.

Once a model has been established, its parameters need to be continuously updated (estimated) in order to minimize the error as the channel changes. If the receiver has a-priori knowledge of the information being sent over the channel, it can utilize this knowledge to obtain an accurate estimate of the impulse response of the channel. This method is simply called *Training sequence based Channel estimation*. It has the advantage of being used in any radio communications system quite easily. Even though this is the most popular method in use today, it still has its drawbacks. One of the obvious drawbacks is that it is wasteful of bandwidth. Precious bits in a frame that might have been otherwise used to transport information are stuffed with training sequences for channel estimation. This method also suffers due to the fact that most communication systems send information lumped frames. It is only after the receipt of the whole frame that the channel estimate can be extracted from the embedded training sequence. For fast fading channels this might not be adequate since the coherence time of the channel might be shorter than the frame time.

Blind methods on the other hand require no training sequences. They utilize certain underlying mathematical information about the kind of data being transmitted. These methods might be bandwidth efficient but still have their own drawbacks. They are notoriously slow to converge (more than 1000 symbols may be required for an FIR channel with 10 coefficients). Their other drawback is that these methods are extremely computationally intensive and hence are impractical to implement in real-time systems. They also do not have the portability of training sequence-based methods. One algorithm that works for a particular system may not work with another due to the fact they send different types of information over the channel.

1.5 Thesis Overview:

Consider a radio system in which training sequences are sent periodically over the channel to produce data based estimates. Sending training sequences over the channel instead of information is a waste (in terms of throughput) of the available bandwidth. This thesis presents a method of improving on these data based estimates without increasing the bandwidth usage.

We assume that the receiver is moving at a constant velocity and the Omni directional antenna receives an infinite number of reflected waves, with a uniformly distributed angle of arrival. Under these assumptions the received signal will have a spectrum known as the Jakes spectrum *[2]*. This enables us to adopt the Jakes channel model for the time varying tap-gain functions. The Jakes channel model allows us to predict the state of the channel, independent of the data based estimate. Instead of having only a single snapshot estimate of the channel, in the case of the data-only estimator, we now have two independent estimates of the channel.

The Kalman estimator combines the two independent estimates of the channel into a LMMSE estimate. The current data estimate of the channel is combined with the predicted (from the model) estimate. This current estimate is then projected through the channel model to predict the next state of the channel. Both these steps are combined into a concise form by using the Kalman filter.

As a result of using the Jakes model in conjunction with the data estimates, the Kalman filter based channel estimation procedure drastically improves on the performance of the data-only estimator. We are also able to predict the next state of the channel with some accuracy before the availability of the data estimate. Lastly as a consequence of using the Kalman filter, we can also improve on previous estimates, as more data is available.

This thesis is organized into two main sections. The first section below provides a thorough theoretical foundation for the concepts to follow. It explains the different types of multipath models and in particular the derivation of the Jakes Radio Model. In the next section, the channel estimation procedure is developed. It is developed first, for a simple Gauss-Markov channel and finally for the Jakes channel.

2 Theoretical formulation:

In this chapter we will lay down the basic theoretical groundwork necessary for us to be able to model the channel.

2.1 The general Multipath Radio Channel Model:

Signal multipath occurs when the transmitted signal arrives at the receiver via multiple propagation paths. Each of these paths may have a separate phase, attenuation, delay and Doppler frequency associated with it. Due to the random phase shift associated with each received signal, they might add up destructively, resulting in a phenomenon called *Fading*.

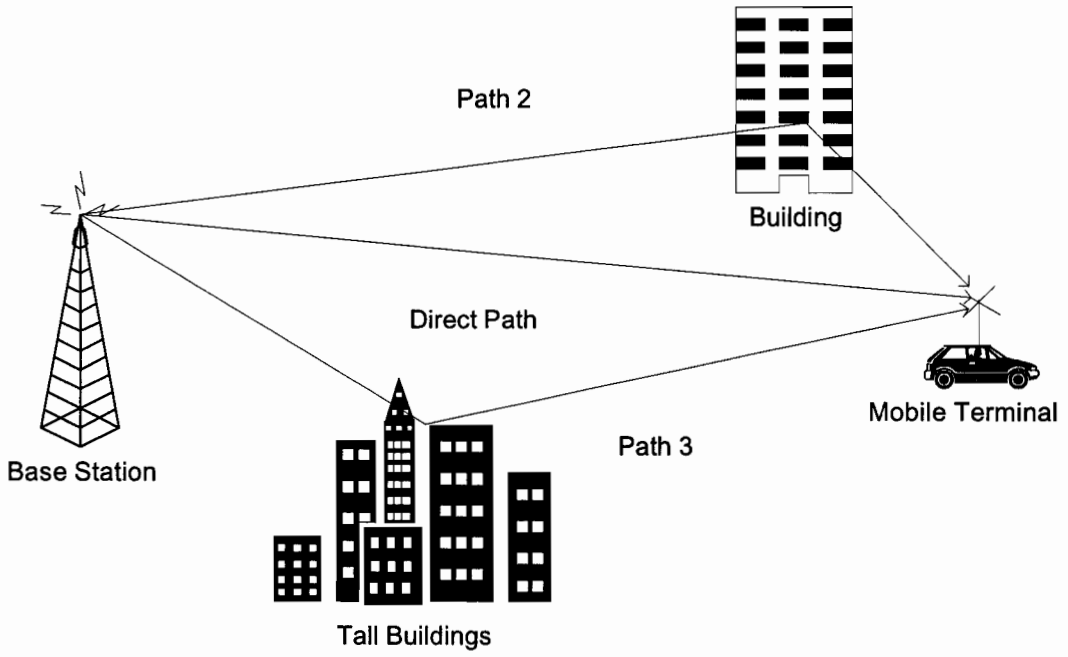

Figure 2-1: Illustration of a Multipath channel in an urban environment

Depending on the nature of the multiple paths received, there are two types of multipath channels *[3]*:

2.2 Discrete Multipath:

When the paths between the transmitter and the receiver are discrete, each with a different attenuation and delay, the channelis called a discrete multipath channel.

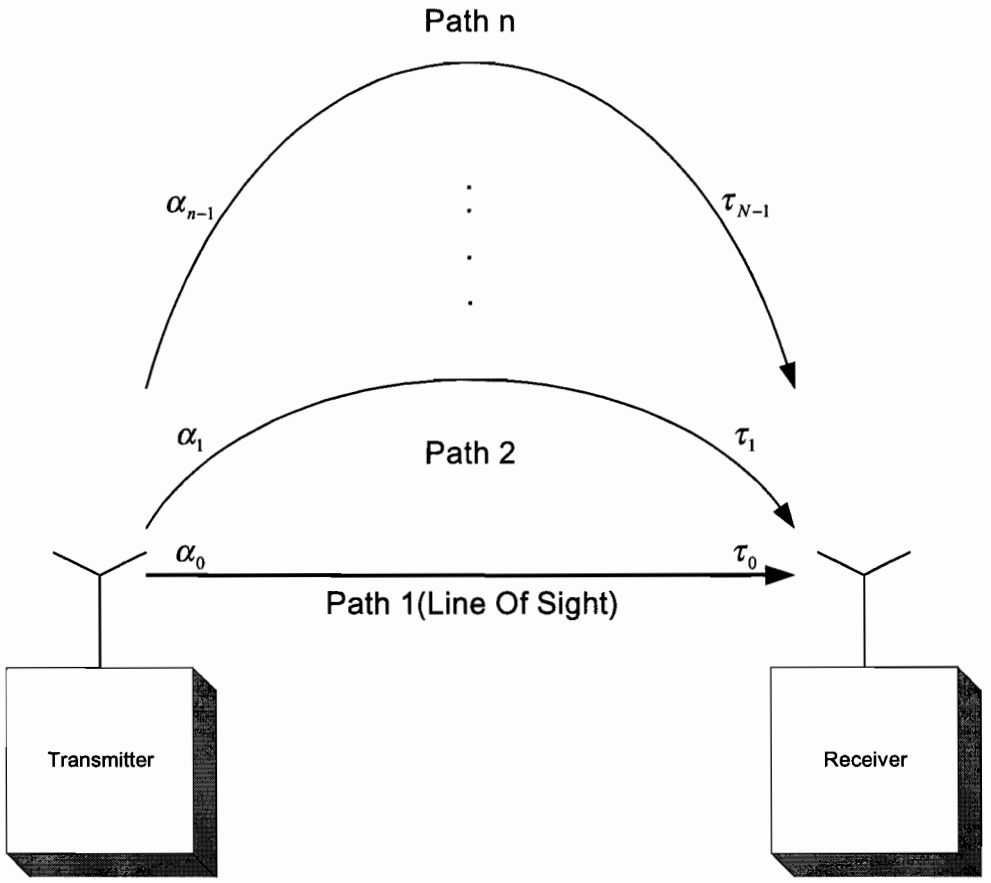

Figure 2-2: The Discrete Multipath Radio Channel

As shown in the figure above, the discrete multipath channel can bemodeled as follows:

$$y(t) = \sum_{i=1}^{N} \alpha_i s(t - \tau_i(t)) \qquad (2\text{-}1)$$

Where

N is the number of rays impinging on the receiver,

$s(t)$ is the bandpass input signal,

α_i is the path attenuation,

τ_i is the path delay.

It can be seen that a natural representation of the discrete multipath channel is a tapped delay line with time varying coefficients and possibly time varying tap spacing.

If we express *s(t)* as:

$$s(t) = \text{Real part of}\left\{\tilde{s}(t)e^{j2\pi f_C t}\right\} \tag{2-2}$$

We can express the complex channel output as:

$$\tilde{y}(t) = \sum_{i=1}^{N} \tilde{\alpha}_i \tilde{s}(t-\tau_i(t)) \tag{2-3}$$

where:

f_C : the frequency of the carrier.

$$\tilde{\alpha}_i = \alpha_i e^{j2\pi f_C t} \tag{2-4}$$

Thus we see that we can describe the time varying discrete multipath channel by a time varying complex impulse response:

$$\tilde{h}(\tau;t) = \sum_{i=1}^{N} \tilde{\alpha}_i \delta(t-\tau_i(t)) \tag{2-5}$$

Where $\tilde{\alpha}_i$ is the time varying complex attenuation of each path.

We can already see that for a fixed number of paths, *(N)* and path delays τ_i, if we were to specify the properties of the complex attenuation $\tilde{\alpha}_i$, for each path, we would be able to characterize the time varying channel.

2.3 Diffuse Multipath:

In some channels it is more appropriate to view the received signal as consisting of a continuum of multipath components*[3]*. This is called a diffuse multipath channel. The received signal is given by:

$$\tilde{y}(t) = \int_{-\infty}^{\infty} \tilde{a}(t;t)\tilde{s}(t-t)dt \tag{2-6}$$

where $\tilde{a}(\tau,t)$ is the attenuation at delay τ and time t.

The Low pass, time variant, impulse response is:

$$\tilde{h}(\tau;t) = \alpha(\tau;t)e^{j2\pi f_C \tau} \tag{2-7}$$

If the signal is band-limited, then the time varying diffuse multipath channel can be represented by a tap-delayed line with time varying coefficients and fixed tap spacing as shown in the figure below *[3]*.

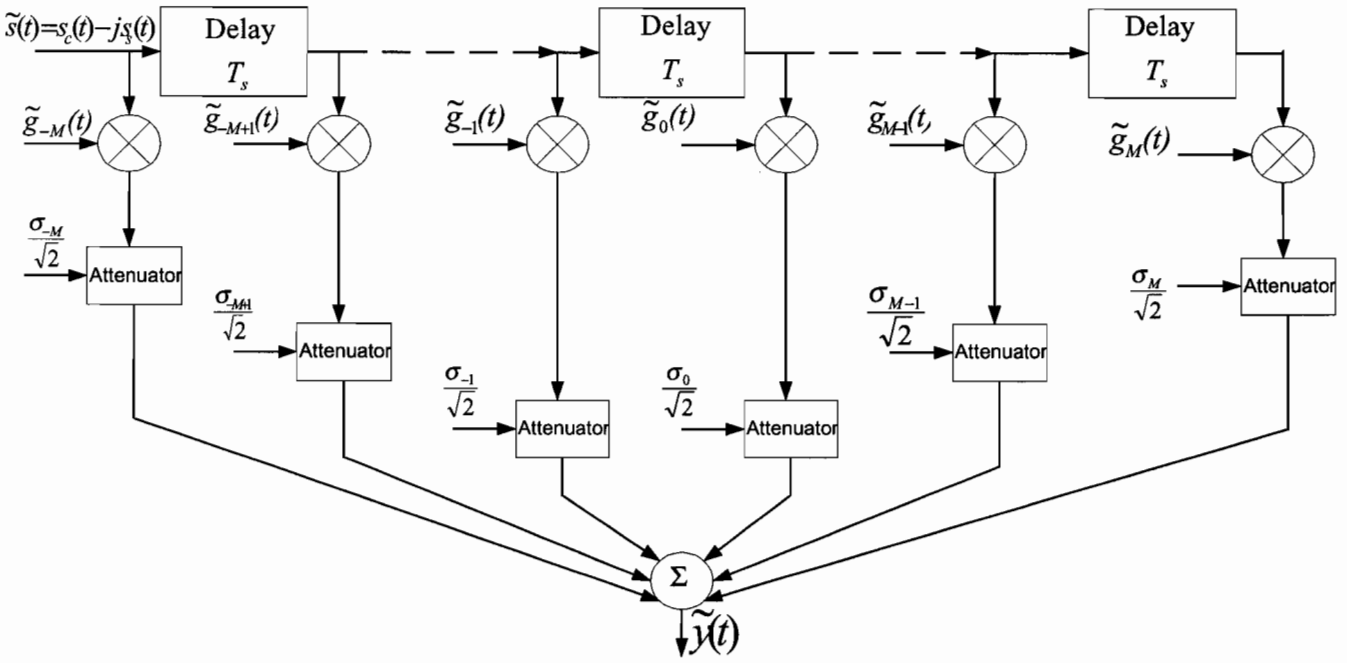

Figure 2-3: The Tap-Delay line radio model

In the figure above, T_S is the symbol time and the taps are symbol spaced.

The output signal can be written as:

$$\tilde{y}(t) = \sum_m \tilde{g}_m(t)\tilde{s}(t-\frac{m}{W}) \qquad (2\text{-}8)$$

where:

$$\tilde{g}_m(t) = \frac{1}{W}\tilde{h}(\frac{m}{W};t) \qquad (2\text{-}9)$$

$\tilde{g}_m(t)$: is the sampled (in the τ domain) complex low-pass equivalent impulse response.

W: is the Bandwidth of the Bandpass Signal. It has a low pass equivalent

Bandlimited to $\frac{W}{2}$

Using WSSUS(Wide Sense Stationary Uncorrelated Scattering) assumptions, the *Delay Cross Power Spectral density* is defined as:

$$R_C(\tau;\Delta t) = \frac{1}{2} E\left[\tilde{h}^*(\tau,t)\tilde{h}(\tau,t+\Delta t)\right] \qquad (2\text{-}10)$$

When $\Delta t = 0$

$$R_C(\tau;\Delta t) = R_C(\tau) \qquad (2\text{-}11)$$

This describes the average power as a function of path delay. The function $R_C(\tau)$ is known as *Multipath Intensity Profile* or *Delay Power Spectrum.* The range of τ for which $R_C(\tau)$ is essentially non zero is called the *Multipath Delay Spread.*

Another important characterization of the channel is the *Scattering function.* It describes the relationship between the power spectrum and the path delays. It is represented by:

$$S(\tau;v) = F[R_C(\tau;\Delta t)] = \int_{-\infty}^{\infty} R_C(\tau;\Delta t) e^{-j2\pi v\Delta t} d(\Delta t) \qquad (2\text{-}12)$$

where *F[]* is the Fourier transform operator. For fixed τ, the scattering function describes the power spectral density in the frequency variable v referred to as the *Doppler Frequency.*

Returning to the tap delay line model, if we assume that the number of scatterers in each path is infinitely large, as a consequence of the central limit theorem, we will see later, that we can model $\tilde{h}(\tau;t)$ as a complex gaussian process. If this process is assumed to be zero mean then the envelope, $\left|\tilde{h}(\tau;t)\right|$ is Rayleigh distributed and the channel is said to be a *Rayleigh Fading channel.* If part of the signal reaches directly while the rest reaches the receiver through a continuum of paths, then $\tilde{h}(\tau;t)$ can be modeled as a non-zero mean gaussian process. The channel is called a *Ricean Fading Channel.* So, to represent the impulse response of a multipath radio channel, we have to choose our tap gains as sampled versions of a complex gaussian process.

With this information, the tapped delay model takes the following form:

- Number of taps are $T_M W+1$. (2-13)

  Where T_M is the delay spread and W is the information bandwidth.

- Tap spacing is: $1/W$. (2-14)

  This is the resolution of the multipath channel model.

- Tap gain functions $\tilde{g}(t)$: A discrete time complex Gaussian processes with the variance of each component given by:

$$\sigma_m^2 = \frac{1}{W^2} R_C(\frac{m}{W}) \quad (2\text{-}15)$$

- PSD of the $\tilde{g}(t)$:

$$S_{gg} = \frac{1}{W^2} S(\frac{m}{W}; v) \quad (2\text{-}16)$$

As in the discrete case, once the number of taps and tap spacing has been specified, it only remains to track (or estimate) the time varying complex tap gain functions. If we can track the tap gain functions, we are then able to track the time varying physical channel itself as represented by the tap delay model in Figure 2-3.

We can characterize the tap gain functions by their power spectrum and relative variance. The channel power delay profile ($R_C(\tau)$) provides us the relative variance for the tap gain functions. In the next section we study the Jakes Radio Model which provides us the required power spectrum for the complex gaussian, tap gain processes.

2.4 Jakes Radio Model:

In order to produce a second independent estimate of the channel, we need an underlying model of the channel tap-gain process. If we assume that the receiver is moving at a constant velocity and its Omni-direction antenna receives an infinite number of scattered waves at uniformly distributed angles of arrival, then the received power spectrum has the shape of the Jakes spectrum. Under these assumptions we choose the Jakes model to represent the tap gain processes. A more detailed explanation of the observed spectrum at the receiver is presented below.

One easily observable property of the signal transmitted over a mobile channel, is the variation in its amplitude as the mobile is moved. Recordings made over the frequency range from 50 to 11.2 GHz have shown the envelope of the mobile radio signal to be Rayleigh distributed when measured over distances of a few tens of wavelengths where the mean signal remains constant. This suggests the assumption that any point *[2]* the received field is made up of horizontally travelling plane waves with random amplitudes and angles of arrivals for different locations. The phases are uniformly distributed on $[0, \pi]$ and the phases and amplitudes are assumed to be independent.

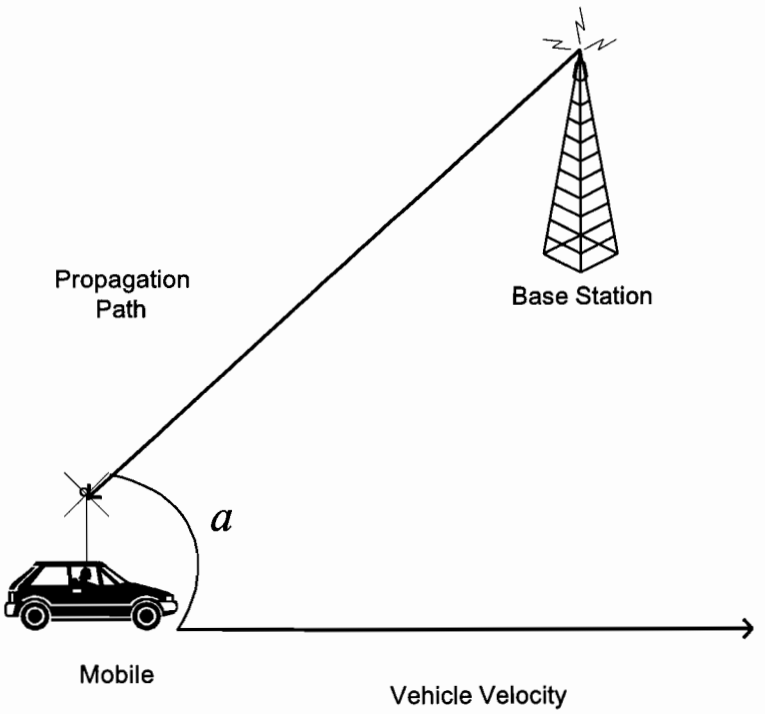

Figure 2-4: Doppler Shift as a function of velocity and angle

A diagram of this simple model is shown above in**Error! Reference source not found.** with plane waves from stationary scatterers (or directly from the receiver) incident on the mobile travelling in the xdirection with a velocity v.

The Doppler Shift f_d in every wave is given by:

$$f_d = \frac{v}{\lambda}\cos(\alpha) \tag{2-17}$$

where v: Vehicle velocity,

λ : Wavelength of the carrier.

The electric field component at the receiver is given by:

$$E = E_0 \sum_{n=1}^{N} C_n \cos(\omega_c t + \theta_n) \tag{2-18}$$

where $\theta_n = 2\pi f_d t + \phi_n$ (2-19)

ϕ_n : Random phase Uniformly distributed on $[0, \pi]$,

ω_c : The carrier frequency,

$E_0 C_n$: the real amplitude of the nth wave in the field.

We note from Equation (2-17) that the Doppler shift is bounded by $f_m = v/\lambda$, which in general will be much less than the carrier frequency. The figure below shows the measured spectrogram at a moving receiver *[2]*. We can clearly see that the Doppler shift is bounded.

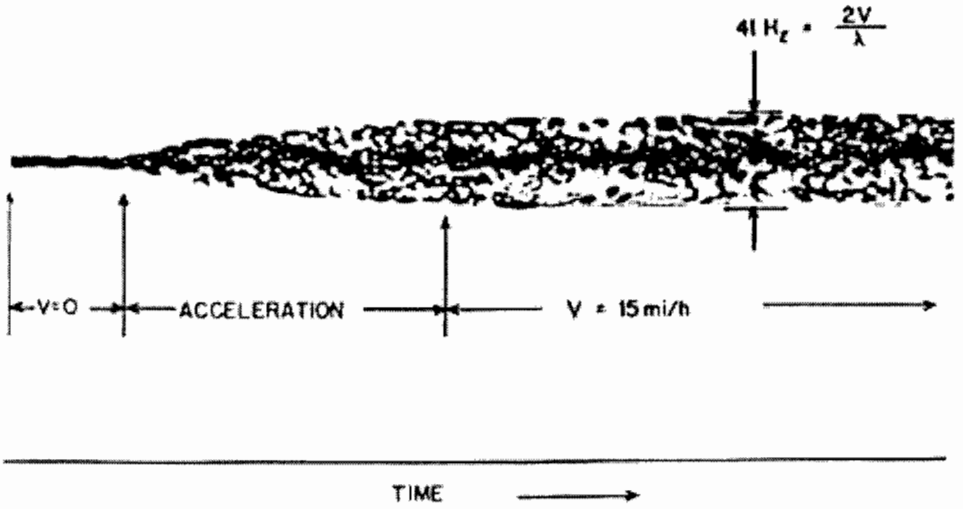

Figure 2-5: Measured Frequency Spectrogram of the RF Signal at 910 MHz

Since the Doppler spectrum at the receiver is bounded by $f_m = v/\lambda$, the electric field component at the receiver may then be described as a narrowband random process. As a consequence of the central limit theorem, i.e. as $N \rightarrow \infty$ (as the number of scatterers approaches infinity), *the received electric field is approximately a gaussian random process*. We can now apply all our knowledge of Gaussian random processes to model the received signal.

RF Spectra of the Electric field at the receiver.

Assuming that the electric field at the receiver may be represented by the sum of N incident waves received at angles uniformly distributed on $[0,2\pi]$. As $N \rightarrow \infty$, we would expect the incident power included in an angle between α and $\alpha+d\alpha$ would approach a continuous distribution. Denote $p(\alpha)d\alpha$ as a fraction of the incoming power contained within $d\alpha$. Assume that the antenna gain of the Omni-directional antenna is $G(\alpha)$.

Factoring in the antenna gain at the receiver, the differential variation of received power with angle is:

$$bG(\alpha)p(\alpha)d\alpha\,.^{1} \tag{2-20}$$

The maximum Doppler frequencyis given by:

$$f_m = \frac{v}{\lambda} \tag{2-21}$$

Where

v: Vehicle velocity

λ: Wavelength of the carrier.

Note the relationship of frequency with angle:

$$f(\alpha) = f_m \cos(\alpha) + f_C \tag{2-22}$$

Where

f_C : The Carrier frequency.

Since $f(\alpha) = f(-\alpha)$, the differential variation of power with frequency may be expressed as follows:

$$S(f)|df| = b[p(\alpha)G(\alpha) + p(-\alpha)G(-\alpha)]\,|d\alpha| \tag{2-23}$$

But

$$|df| = f_m|-\sin(\alpha)d\alpha| = \sqrt{f_m^2 - (f - f_C)^2}\,|d\alpha| \tag{2-24}$$

thus

$$S(f) = \frac{b}{\sqrt{f_m^2 - (f - f_C)^2}}[p(\alpha)G(\alpha) + p(-\alpha)G(-\alpha)] \tag{2-25}$$

and $S(f) = 0$ if $|f - f_C| > f_m$.

The angle of arrival is given by:

$$\alpha = \cos^{-1}\left(\frac{f - f_C}{f_m}\right) \tag{2-26}$$

---

[1] b is the average power that would be received by an isotropic antenna, i.e. G(α)=1

The electrical field in the z-direction is sensed with a vertical whip antenna with $G(\alpha) = 1.5$. Then the spectrum is:

$$S(f) = \frac{1.5b}{\sqrt{f_m^2 - (f - f_C)^2}}[p(\alpha) + p(-\alpha)] \qquad (2\text{-}27)$$

The simplest assumption for the distribution of the power with arrival angle α, is a uniform distribution:

$$p(\alpha) = \frac{1}{2\pi} \qquad -\pi \le \alpha \le \pi \qquad (2\text{-}28)$$

The normalized power spectrum is then given by:

$$S'(f) = \frac{S(f)}{\left(\frac{3b}{\omega_m}\right)} = \left[1 - \left(\frac{f - f_C}{f_m}\right)^2\right]^{-1/2} \qquad (2\text{-}29)$$

Its form is as follows:

Title:
/.automount/devnull/filez/users/rupul/research/data_kalman/figs/J_spec.eps
Creator:
MATLAB, The Mathworks, Inc.
Preview:
This EPS picture was not saved
with a preview included in it.
Comment:
This EPS picture will print to a
PostScript printer, but not to
other types of printers.

Figure 2-6: Jakes Spectrum at Baseband.

The Jakes power spectrum can be synthesized by using a shaping filter driven by gaussian white noise. The impulse response of the Jakes shaping filter is given by:

$$h_j(t) = 2^{1/4}\Gamma(\frac{3}{4}) f_m (2\pi f_m t)^{-1/4} J_{1/4}(2\pi f_m t) \qquad (2\text{-}30)$$

where:

Γ : is the Gamma function,

f_m : is the Doppler Bandwidth of the channel,

$J_{1/4}$: is the fractional Bessel function.

Given the power spectrum as described in Equation (2-29), the channel autocorrelation as derived by Jakes *[2]*,[4]is:

$$a(\tau) = F^{-1}(S(f)) = 2\pi J_0(2\pi f_m \tau) \qquad (2\text{-}31)$$

where

J_0 is the *0th* order Bessel function

This autocorrelation is shown in the figure below:

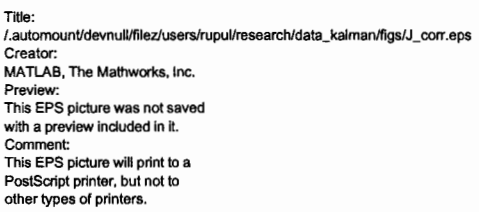

Figure 2-7: Jakes Autocorrelation.

Comparing Figure 2-5and Figure 2-6, we can see that the Jakes spectrum is a representation of the true spectrogram at the receiver. Assuming an infinite number of scatterers and a uniform angle of arrival, the Jakes model is then a good model for the radio channel. Thus the Jakes spectrum is assigned to the tap-gain processes.

3 The Channel Estimation algorithm:

3.1 Introduction:

The channel estimation procedure considers a Radio communication system in which training sequences are sent periodically to form data based estimates of the channel. The channel is assumed to be invariant over the time span of the training sequence being sent over the channel. The performance of the data estimator is proportional to the length of the training sequence sent (if channel noise remains constant). Any desired improvement in performance is gained by increasing the length of the training sequence. This, in general is undesirable since any such increase would result in more waste of channel bandwidth, which is better utlized for sending data rather than training information. The channel estimation algorithm presented in this section improves on the data based estimate without decreasing channel throughput.

The channel estimation procedure utilizes the underlying channel model in conjunction with the available data based estimate. As explained previously, under the assumptions of uniform angle of arrival and infinite scatterers, we adopt the Jakes model for the channel. The Jakes model assigns an autocorrelation and powerspectral density to the time varying tap gain processes of the channel. The Jakes model is represented as an auto-regressive process where the current value of the process is a weighted sum of previous values plus plant noise. The weights for the AR Jakesmodel can be calculated, by solving the Yule-Walker set of equations (derived from the autocorrelation of the process) or from a shaping filter (derived from the spectrum). This AR Jakes model enables us to predict the tap gain process independently of the data based estimator.

Once the Jakes model has been represented in AR form, along with the data based estimator we have two independent estimates of the channel. We first express the Jakes AR model in the form of a system equation. By viewing the data estimate as a noisy observation of the true tap-gain process, we can represent the data estimator as an observation equation. The system equation and the observation equation jointly form a *State-Space representation* of the dynamics of the tap-gain process. This state-space representation of the overall process is used to formulate the parameters of a Kalman

filter. The channel estimation algorithm in the form of a Kalman filter produces LMMSE (Linear Minimized Mean Squared Error) estimates of the tap-gain process. Since it uses both the data estimator and the Jakes channel model, we will show through simulations that it performs better than the data-only estimator.

First we use the Kalman filter based channel estimation algorithm to track a simple Gauss-Markov channel. Then we simulate for a single-ray channel as described by a Jakes model. Finally we simulate for the multipath channel driven by the Jakes model. In each of these cases results are presented showing the improvement on the data-only estimate.

3.2 Modeling the channel tap gain as an Auto-regressive process.

In order to develop a model for the tap-gain process, we must first explain how a complex gaussian random process can be represented by a general auto-regressive (AR) model.

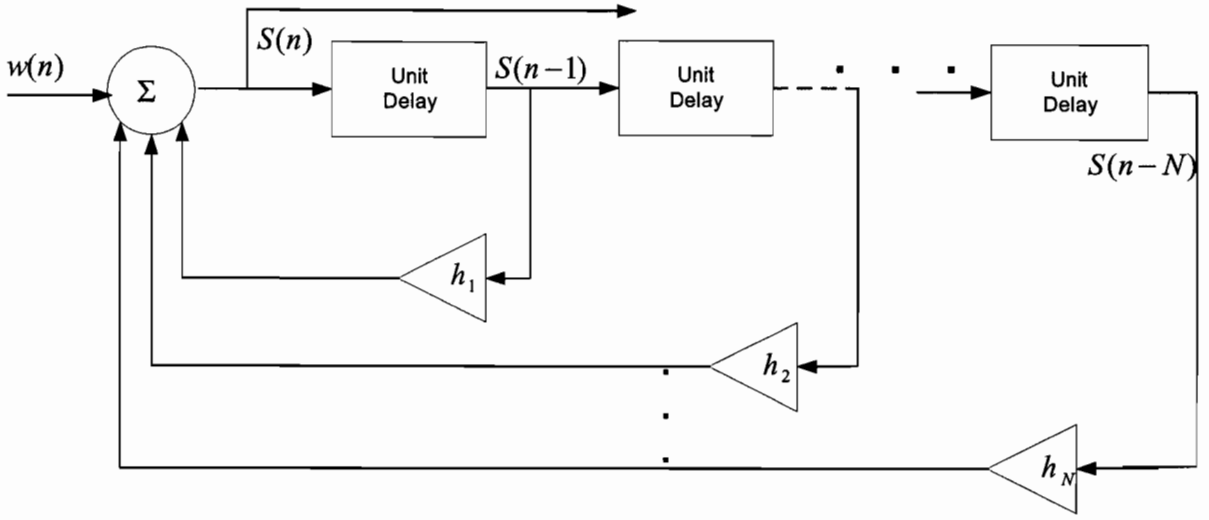

Figure 3-1: The General Auto-regressive process model.

Any stationary random process can be represented as an infinite tap AR process *[8]*. Of course an infinite tap AR process model is impractical. It is truncated to an *N*-tap form. We will use the truncated AR process model to represent the underlying model driving the tap-gain process.

An AR process has the form in the figure above and is represented by a diference equation of the form:

$$S(n) = \sum_{i=1}^{N} \phi_i S(n-i) + w(n) \qquad (3\text{-}1)$$

Where

S(n): is a complex gaussian process,

ϕ_i are the parameters of the model.

N: is the number of delays in the autoregressive model.

w(n): A sequence of identically distributed zero-mean Complex Gaussian random variables.

In other words:

$$E\{w(n)\}=0 \tag{3-2}$$

$$E\{w(n)w(j)\}=\begin{Bmatrix} s_n^2 & for\ n=j \\ 0 & n \neq j \end{Bmatrix} \tag{3-3}$$

$$f_{w(n)}(\lambda)=\frac{1}{\sqrt{2\pi\sigma_n^2}}e^{\left\{\frac{-\lambda^2}{2\sigma_n^2}\right\}} \tag{3-4}$$

The sequence $w(n)$ is called white Gaussian noise because its spectrum is broad and uniform over an infinite frequency range. Thus the AR process is simply another name for a linear difference equation model, driven by white gaussian noise.

This Nth order difference equation can easily be reduced to a *state model* in the vector form:

$$\bar{S}(n)=F\bar{S}(n-1)+\bar{W}(n) \tag{3-5}$$

where $\bar{S}$ and $\bar{W}(n)$ are column vectors of size $(N\times 1)$ and F is an $(N\times N)$ matrix

We now find the mean, variance and autocorrelation of the autoregressive process *[5]*:

Mean:

$$\mu_S = E[S(n)] = E\left[\sum_{i=1}^{N}\phi_i S(n-i)+w(n)\right]=0 \tag{3-6}$$

Variance:

$$\begin{aligned}\sigma_S^2 &= E\{S(n)S(n)\} \\ &= E\left\{S(n)\sum_{i=1}^{N}\phi_i S(n-i)+w(n)\right\} \\ &= \sum_{i=1}^{N}\phi_i R_{SS}(i)+\sigma_n^2\end{aligned} \tag{3-7}$$

Autocorrelation:

$$\begin{aligned}R_{SS}(m) &= E\{S(n-m)S(n)\} \\ &= E\left\{\left[\sum_{i=1}^{N}\phi_i S(n-i)+w(n)\right]S(n-m)\right\} \\ &= \sum_{i=1}^{N}\phi_i R_{SS}(m-i)\end{aligned} \tag{3-8}$$

The autocorrelation coefficient is:

$$r_{SS}(m) \equiv \frac{R_{SS}(m)}{s_X^2} \tag{3-9}$$

$$= \sum_{i=1}^{N} \phi_i r_{SS}(m-i), \qquad m \geq 1 \tag{3-10}$$

Equation (3-10) is an Nth order difference equation that can be solved for the desired AR coefficients *[5]*. Expressing the difference equation in the form of a matrix:

$$\begin{bmatrix} 1 & r_{SS}(1) & r_{SS}(2) & r_{SS}(3) & \cdots & r_{SS}(N-1) \\ r_{SS}(1) & 1 & & & \cdots & r_{SS}(N-2) \\ . & & & & & . \\ . & & & & & . \\ r_{SS}(N-1) & & & & \cdots & 1 \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \\ . \\ . \\ \phi_N \end{bmatrix} = \begin{bmatrix} r_{SS}(1) \\ r_{SS}(2) \\ . \\ . \\ r_{SS}(N) \end{bmatrix} \tag{3-11}$$

This matrix equation is known as the Yule– Walker equation. Its matrix form is:

$$\bar{R}\phi = \bar{r}_{SS} \tag{3-12}$$

Since R is invertible, we can obtain:

$$\phi = \bar{R}^{-1}\bar{r}_{SS} \tag{3-13}$$

This is the matrix of AR coefficients that models the complex gaussian process. If we are given the autocorrelation of the process, using the Yule-Walker equations, we can calculate the appropriate AR coefficients. This will allow us to express the underlying process model in a state-space form allowing us to formulate a Kalman filter to track/estimate the process.

3.3 Data Based Channel Estimator formulation:

The data based estimator uses training sequences sent over the channel to estimate the impulse response of the channel. We now describe how channel estimation is performed using the correlation method.

A training sequence of length M, known to the receiver is sent over the channel. It is assumed that the *channel does not change over the span of the data* sent

Its vector form is:

$$\bar{x} = [x_0 x_1 \ldots\ldots\ldots\ldots x_{M-1}]^T \qquad (3\text{-}14)$$

where $^{'T'}$ is the transpose operator.

This bit sequence has been mapped to unit energy symbols as follows:

$$\begin{Bmatrix} Bit & \rightarrow Symbol \\ 0 & \rightarrow +1 \\ 1 & \rightarrow -1 \end{Bmatrix} \qquad (3\text{-}15)$$

to simulate a BPSK modulation.

Let the channel impulse response at the snapshot when the training sequence is sent over the channel be:

$$\bar{h} = [\tilde{h}_0 \tilde{h}_1 \tilde{h}_2 \ldots\ldots \tilde{h}_{L-1}]^T \qquad (3\text{-}16)$$

where L is the channel impulse response length or the number of processesto be tracked.

The signal received will then be the convolution sum of the signal sent and the impulse response received in the presence of channel noise:

$$\bar{y} = \bar{x} * \bar{h} + n_c \qquad (3\text{-}17)$$

The convolution sum is given by:

$$y(n) = \sum_{m=0}^{L-1} h(m)x(n-m) + n_c \qquad (3\text{-}18)$$

This can be written in matrix form as follows:

$$\bar{y} = \begin{bmatrix} x_0 & 0 & . & 0 \\ x_1 & x_0 & . & . \\ . & x_1 & . & 0 \\ . & . & . & x_0 \\ x_{M-1} & . & . & x_1 \\ 0 & x_{M-1} & . & . \\ . & 0 & . & . \\ 0 & 0 & . & x_{M-1} \end{bmatrix} \begin{bmatrix} h_0 \\ h_1 \\ . \\ . \\ h_{L-1} \end{bmatrix} + \begin{bmatrix} n_{c_0} \\ n_{c_1} \\ . \\ . \\ . \\ n_{c_{L+M-21}} \end{bmatrix} \qquad (3\text{-}19)$$

Or in its vector form:

$$\bar{Y} = \bar{X}\,\bar{h} + \bar{n}_c \qquad (3\text{-}20)$$

where $\bar{Y}$ is an $(L+N-2\times 1)$ received signal vector.

$\bar{X}$ is an $(L+N-1\times L)$ Toeplitz matrix containing delayed versions of the training sequence sent.

$\bar{h}$ is the channel impulse response.

$\bar{n}_c$ is the zero mean additive white gaussian channel noise of variance σ_c^2.

By anchoring $E_b = 1$, the SNR of the channel is then given by:

$$\frac{E_b}{N_0} = \frac{1}{2\sigma_c^2} \qquad (3\text{-}21)$$

Following the general linear regression method given in*[5]*, the estimate of the channel is given by:

$$\hat{\bar{h}} = (\bar{X}^T\bar{X})^{-1}(\bar{X}^T\bar{Y}) \qquad (3\text{-}22)$$

We now calculate the error of the data based estimator. Since the received signal is given by

$$\bar{Y} = \bar{X}\,\bar{h} + \bar{n}_c \qquad (3\text{-}23)$$

its performance will vary depending on the current channel noise. Using the actual noisy and distorted received signal, we can see the error in estimation is:

$$\begin{aligned} \hat{\bar{h}} &= (\bar{X}^T\bar{X})^{-1}(\bar{X}^T(\bar{X}\,\bar{h} + \bar{n}_c)) \\ &= (\bar{X}^T\bar{X})^{-1}(\bar{X}^T\bar{X})\bar{h} + (\bar{X}^T\bar{X})^{-1}(\bar{X}^T\bar{n}_c) \\ &= \bar{h} + (\bar{X}^T\bar{X})^{-1}(\bar{X}^T\bar{n}_c) \end{aligned} \qquad (3\text{-}24)$$

Thus the error is:

$$\begin{aligned}\tilde{\bar{h}} &= \hat{\bar{h}} - \bar{h} \\ &= (\bar{X}^T \bar{X})^{-1}(\bar{X}^T \bar{n}_c)\end{aligned} \tag{3-25}$$

Properties of the data estimator:

Taking expextations on both sides of the estimation error, we get:

$$\begin{aligned}E\left[\tilde{\bar{h}}\right] &= E\left[(\bar{X}^T \bar{X})^{-1}(\bar{X}^T \bar{n}_c)\right] \\ &= (\bar{X}^T \bar{X})^{-1}(\bar{X}^T E[\bar{n}_c])\end{aligned} \tag{3-26}$$

Since the channel noise is zero mean, we get:

$$E\left[\tilde{\bar{h}}\right] = 0 \tag{3-27}$$

Thus we see that the estimator is unbiased.

The error covariance is defined as follows *[7]*:

$$\begin{aligned}P_D &= E\left[\tilde{\bar{h}}\left(\tilde{\bar{h}}\right)^H\right] \\ &= E\left\{\left[(\bar{X}^T \bar{X})^{-1}(\bar{X}^T \bar{n}_c)\right]\left[(\bar{X}^T \bar{X})^{-1}(\bar{X}^T \bar{n}_c)\right]^H\right\}\end{aligned} \tag{3-28}$$

[2]

Simplifying, we get:

$$\begin{aligned}P_D &= E\left\{\left[(\bar{X}^T \bar{X})^{-1}(\bar{X}^T \bar{n}_c)\right]\left[(\bar{X}^T \bar{n}_c)^H ((\bar{X}^T \bar{X})^{-1})^H\right]\right\} \\ &= E\left\{(\bar{X}^T \bar{X})^{-1}(\bar{X}^T \bar{n}_c)(\bar{n}_c{}^H \bar{X})(\bar{X}^T \bar{X})^{-1}\right\} \\ &= (\bar{X}^T \bar{X})^{-1}\bar{X}^T E\left(\bar{n}_c \bar{n}_c{}^H\right)\bar{X}(\bar{X}^T \bar{X})^{-1} \\ &= (\bar{X}^T \bar{X})^{-1}\bar{X}^T\left\{\sigma_c^2 I\right\}\bar{X}(\bar{X}^T \bar{X})^{-1} \\ &= \sigma_c^2\left[(\bar{X}^T \bar{X})^{-1}\left(\bar{X}^T \bar{X}\right)(\bar{X}^T \bar{X})^{-1}\right] \\ &= \sigma_c^2 (\bar{X}^T \bar{X})^{-1}\end{aligned} \tag{3-29}$$

We need to explain the significance of $(\bar{X}^T \bar{X})^{-1}$ before going any further.

---

[2] H is the hermetian transpose of a matrix. It is defined as the complex conjugate of the standard transpose.

$$(\overline{X}^T\overline{X}) = \begin{bmatrix} x_0 & x_1 & . & . & x_{M-1} & 0 & . & 0 \\ 0 & x_0 & x_1 & . & . & x_{M-1} & 0 & 0 \\ . & . & . & . & . & . & . & . \\ 0 & . & 0 & x_0 & x_1 & . & . & x_{M-1} \end{bmatrix} \begin{bmatrix} x_0 & 0 & . & 0 \\ x_1 & x_0 & . & . \\ . & x_1 & . & 0 \\ . & . & . & x_0 \\ x_{M-1} & . & . & x_1 \\ 0 & x_{M-1} & . & . \\ . & 0 & . & . \\ 0 & 0 & . & x_{M-1} \end{bmatrix}$$

$$= \begin{bmatrix} \sum_{i=0}^{M-1} x_i^2 & \sum_{i=0}^{M-1} x_i x_{i-1} & . & \sum_{i=0}^{M-1} x_i x_{i-L+1} \\ . & . & . & . \\ . & . & . & . \\ \sum_{i=0}^{M-1} x_i x_{i-L+1} & . & \sum_{i=0}^{M-1} x_i x_{i-1} & \sum_{i=0}^{M-1} x_i^2 \end{bmatrix} \qquad (3\text{-}30)$$

Since $x_i = \pm 1$, $\sum_{i=0}^{M-1} x_i^2 = M$

$$(\overline{X}^T\overline{X}) = \begin{bmatrix} M & \sum_{i=0}^{M-1} x_i x_{i-1} & . & \sum_{i=0}^{M-1} x_i x_{i-L+1} \\ . & . & . & . \\ . & . & . & . \\ \sum_{i=0}^{M-1} x_i x_{i-L+1} & . & \sum_{i=0}^{M-1} x_i x_{i-1} & M \end{bmatrix}$$

$$= M \begin{bmatrix} 1 & \frac{1}{M}\sum_{i=0}^{M-1} x_i x_{i-1} & . & \frac{1}{M}\sum_{i=0}^{M-1} x_i x_{i-L+1} \\ . & . & . & . \\ . & . & . & . \\ \frac{1}{M}\sum_{i=0}^{M-1} x_i x_{i-L+1} & . & \frac{1}{M}\sum_{i=0}^{M-1} x_i x_{i-1} & 1 \end{bmatrix} \qquad (3\text{-}31)$$

Since $$r_{xx}(k) = \frac{1}{R_{xx}(0)} \sum_{i=0}^{M-1} x_i x_{i-k} = \frac{1}{M} \sum_{i=0}^{M-1} x_i x_{i-k} \qquad (3\text{-}32)$$

Thus $(\overline{X}^T\overline{X})$ is nothing but an $(L \times L)$ matrix containing delayed versions of the training sequence autocorrelation.

$$(\bar{X}^T \bar{X}) = M \begin{bmatrix} 1 & r_{xx}(1) & . & r_{xx}(L-1) \\ . & . & . & . \\ . & . & . & . \\ r_{xx}(L-1) & . & r_{xx}(1) & 1 \end{bmatrix} \quad (3\text{-}33)$$

Thus

$$(\bar{X}^T \bar{X})^{-1} = \frac{1}{M} \begin{bmatrix} 1 & r_{xx}(1) & . & r_{xx}(L-1) \\ . & . & . & . \\ . & . & . & . \\ r_{xx}(L-1) & . & r_{xx}(1) & 1 \end{bmatrix}^{-1} \quad (3\text{-}34)$$

where $r_{xx}(\tau)$ is the normalized training sequence autocorrelation.

Thus the error covariance is given by:

$$P_D = \frac{\sigma_c^2}{M} \begin{bmatrix} 1 & r_{xx}(1) & . & r_{xx}(L-1) \\ . & . & . & . \\ . & . & . & . \\ r_{xx}(L-1) & . & r_{xx}(1) & 1 \end{bmatrix}^{-1} \quad (3\text{-}35)$$

For the case of an ideal auto correlation, the equation above simplifies to:

$$P_D = \frac{\sigma_c^2}{M}[I] \quad (3\text{-}36)$$

where $[I]$ is the $(L \times L)$ identity matrix.

It is interesting to note that for a single process estimate $(L = 1)$, the error covariance is given by:

$$P_D = \frac{\sigma_c^2}{M} \quad (3\text{-}37)$$

The result above shows the inverse relationship between the length of the training sequence and the covariance of the data estimate. As expected, it also shows that the data estimate worsens as noise in the channel increases.

3.4 Tracking a single tap-gain process.

Consider a point-to-point radio link. Training sequences are sent over this channel to produce data based estimate every frame. The channel is assumed to be invariant for the frame time. This is only a snapshot estimate of the channel for the time that the frame is sent. We intend to track this channel with the help of the underlying channel model. By projecting the estimates through the channel model and combining the predicted estimate with the data estimate, we intend to improve on the data-only estimate of the channel.

We will track two different channels. The first one is a simple Gauss-Markov (GM), channel represented as a first order AR process. The second channel is represented by the Jakes model in the form of a higher order AR process. In both the cases we develop a Kalman filter to track the tap-gain process. For the GM case this will be a scalar Kalman filter. For the Jakes model case we will develop a vector Kalman filter to track the process.

3.4.1 Tracking a Gauss-Markov tap gain process.

A Gauss-Markov process is described by an exponential autocorrelationfunction. Its form is shown in the figure below.

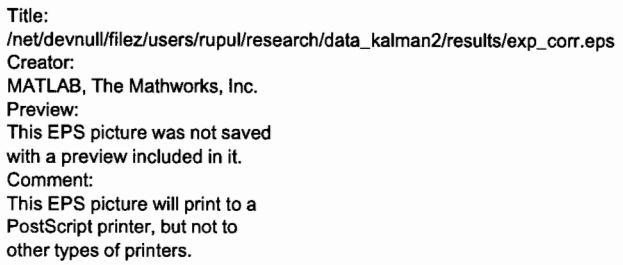

Figure 3-2: Autocorrelation for a Gauss-Markov process.

3.4.1.1 Kalman filter derivation.

In order to track this process with a Kalman filter, we will need to represent the AR model and the data estimation in a State-Space form. This requires the autoregressive model to be expressed as a system equation and the data estimate as an observation equation.

System Model:

As a consequence of the exponential autocorrelation, a Gauss-Markov process can be described as a first order auto-regressive process. In other words, the current state of the process is dependent upon the previous state alone. For a first order process the system model is simply:

$$S(n) = \phi_1 S(n-1) + w(n) \qquad (3\text{-}38)$$

Where

$S(n)$: is the complex gaussian tap-gain process.

ϕ_1 is the parameter of the AR model assumed.

$w(n)$: A sequence of identicallydistributed zero-mean Complex Gaussian random variables with variance σ_w^2.

Observation Model:

The data based estimator produces 'estimates' of the process. It can therefore can be looked upon as a noisy version of the process.

The observation model can be written as:

$$X(n) = S(n) + v(n) \qquad (3\text{-}39)$$

where: $S(n)$: The process at time n.

$X(n)$: The data based estimate of S(n)

$v(n)$: Error of the data based estimate.

$s_v^2 = \frac{s_c^2}{M}$ from equation (3-22) (3-40)

M : Number of training sequence bits in each frame.

Kalman filter for the first order AR process

Given the state space formulation for the process to be tracked, we can use the standard scalar Kalman filter equations to track the process. The scalar Kalman filter is defined by the following equations*[7],[10],[5],[6]*:

- The initial conditions are:

$$\hat{S}(0)=E\{S(n)\}=0 \tag{3-41}$$

$$P(1)\geq\{\sigma_w^2 and \sigma_v^2\} \tag{3-42}$$

- The Kalman gain is given by:

$$k(n)=\frac{P(n)}{P(n)+\sigma_v^2(n)} \tag{3-43}$$

- The current estimate of the process, after receiving the data estimate is given by:

$$\hat{S}_{curr}(n)=\hat{S}(n)+k(n)[X(n)-\hat{S}(n)] \tag{3-44}$$

- The predicted estimate of the process is given by:

$$\hat{S}(n+1)=\phi_1\{\hat{S}_{curr}(n)\} \tag{3-45}$$

- The current error covariance is given by:

$$P_{curr}(n)=[1-k(n)]P(n) \tag{3-46}$$

- The prediction error covariance (MSE in this case) is given by:

$$P(n+1)=\phi_1^2\{P_{curr}(n)\}+\sigma_w^2 \tag{3-47}$$

These equations define the scalar Kalman filter. Its actual implementation as an algorithm is explained next.

3.4.1.2 Simulation and Results:

In this section the scalar Kalman filter is used to track the Gauss-Markov process. The Kalman algorithm is explained in more detail and its prformance is analyzed.

Simulation Parameters:

The parameters for the simulation are as follows:

- System equation: $S(n) = .9S(n-1) + w(n)$
- Observation Equation: $X(n) = S(n) + v(n)$
- The frame rate is : $R_F = 5\times10^4 \, Frames/\sec$. The simulation is run at the frame rate.
- $M = 8$: The length of the training sequence. The channel is assumed to be invariant for these M bits.
- The signal to noise ratio of the channel, for $E_B = 1$ is:

$$\frac{E_B}{N_O} = \frac{E_B}{2\sigma_c^2} = 6dB$$

Thus $\sigma_c^2 = .1256$

- The data estimator variance $s_v^2 = \frac{\sigma_c^2}{M} = 0.0157$ (3-48)
- The variance of the tap gain plant noise is $s_w^2 = 2\sigma_V^2 = 0.0314$ (3-49)

The first order Kalman filter is used to track the process using the above parameters. In order to simulate the performance of the Kalman filter we must first create a first order process.

Process Synthesis:

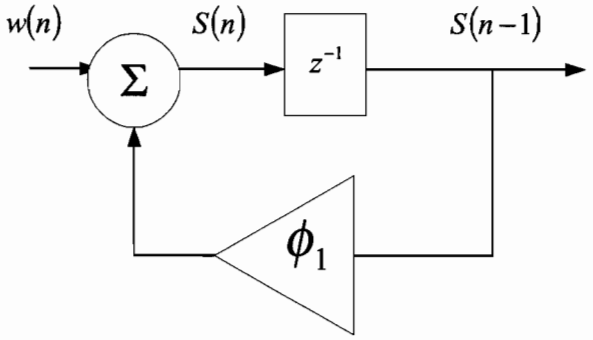

Figure 3-3: Synthesizing the Gauss-Markov process.

The figure above is a representation of the algorithm below for synthesizing a GM process:

Initializations: $S_{temp} = 0 ; S = [\ ];$

Start Iteration { Create complex white Gaussian noise 'w 'of variance σ_w^2

$$S_{temp} = \phi_1 \times S_{temp} + w$$

$$S = \begin{bmatrix} S \\ S_{temp} \end{bmatrix}$$

} End Iteration

In this simulation we will use the Kalman filter to track the process synthesized above.

Algorithm Implementation:

Previously, only the equations for the scalar Kalman filter were presented. The Kalman algorithm is described below in more detail.

The channel estimation algorithm is as follows:

{ Initiallize:

Time: $(n = 1)$

- $P(1) = 1$. Assume initial predicted error covariance.
- $\hat{S}(1) = 0$. Assume initial prediction.
- {Start Iteration:
    - Perform data based estimation, i.e. get $X(n)$.
    - Calculate Kalman gain:

$$k(n) = \frac{P(n)}{P(n)+1}$$

- Calculate current estimate:

$$\hat{S}_{curr}(n) = \hat{S}(n) + k(n)[X(n) - \hat{S}(n)]$$

- Calculate current error covariance:

$$P_{curr}(n) = [1 - k(n)]P(n)$$

- Predict ahead:

$$\hat{S}(n+1) = .9\{\hat{S}_{curr}(n)\}$$

- Predict the error covariance:

$$P(n+1) = (.9)^2\{[1 - k(n)]P(n)\} + 1$$

    - *Time:* $(n = n+1)$

End loop }

Results:

A first order process was created as described above and the Kalman algorithm was used, to track the process. The figure below shows the results of the simulation.

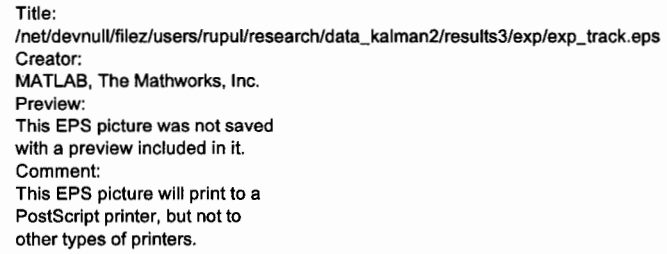

Figure 3-4: Channel estimation for a first order process.

MSE for the estimator:

The mean square error is defined as follows:

The current MSE is:

$$P_{curr} = E\left\{\left[S(n) - \hat{S}_{curr}(n)\right]\left[S(n) - \hat{S}_{curr}(n)\right]^H\right\} \tag{3-50}$$

The prediction MSE is:

$$P = E\left\{\left[S(n) - \hat{S}(n)\right]\left[S(n) - \hat{S}(n)\right]^H\right\} \tag{3-51}$$

Note the relationship between the current and the prediction error covariance:

$$P = \phi_1^{\,2}\{P_{curr}\} \tag{3-52}$$

It is important to remember that the Kalman gain and the error covariance equations are independent of the actual observations. The covariance equation alone is all that is required for characterizing the performance of the filter. This is done by solving for the steady state covariance for the given filter.

When the covariance matrix reaches steady state:

$$\begin{aligned} P(n+1) &= P(n) \\ &= \phi_1^{\,2}\{[1-k(n-1)]P(n-1)\}+\sigma_w^2 \end{aligned} \tag{3-53}$$

This equation is known as the *Ricatti* equation. The solution to this equation is the steady state error covariance P_{SS}. We can attempt to solve this analytically. But since the error covariance actually converges quite quickly to its steady state value, it can be easily calculated by iteration.

The steady state co-variances for this system are:

- The steady state current error covariance is:

$$P_{curr_{SS}} = .0113 \tag{3-54}$$

- The steady state prediction error covariance is:

$$P_{SS} = .0406 \tag{3-55}$$

The actual error covariances for the simulation were calculated by averaging the results over 100 Monte Carlo runs. The following are the results:

- The simulated current error covariance is:

$$P_{curr_{Sim}} = .0113 \tag{3-56}$$

- The simulated prediction error covariance is:

$$P_{Sim} = .0406 \tag{3-57}$$

We can see that there is an appreciable increase in the accuracy of the data based estimate. The percentage in crease is:

$$\frac{\sigma_V^2 - P_{curr_{SS}}}{\sigma_V^2} = 28\% \qquad (3\text{-}58)$$

By using the underlying model for the channel in conjunctin with the data based estimate, we have drastically improved upon the snapshot estimate available. Not only this, we can also predict the next state of the channel, with a *MSE of .0406*. The GM case served as a demonstration for the capabilities of this channel estimation algorithm. The Kalman filter was easily able to improve the data estimate for this simple channel model. We now turn our attention towards the higher order Jakes model.

3.4.2 Tracking a Jakes tap-gain process

Now consider a line of sight, single ray radio channel. The underlying channel model is no longer a simple GM process. Instead we use a higher order Jakes model for this channel. Similar to the previous section where we expressed the first order AR process in a state-space form and used the scalar Kalman filter to track the process, we will express the Jakes process in a state-space form too and track it with a Kalman filter.

The Jakes tap-gain process is stationary and can be represented as an auto-regressive process. In the GM case, the model turned out to be a simple first order AR model. For the Jakes process, we need to calculate the autoregressive model parameters. As we had stated earlier in section (3.2), we can derive this from the auto-correlation of the process by solving the Yule-walker equations. We can also derive the AR coefficients from the Jakes power spectrum by using the closed form expression of the Jakes channel-shaping filter.

3.4.2.1 Auto-Regressive Form for the Jakes model.

In section (2.4) the power spectrum of the Jakes process was given. The AR parameters for this process will be derived from the knowledge of this spectrum. The process can be synthesized by using a noise-shaping filter as given by equation (2-30). We will use this shaping filter to derive the AR form of the process. The closed form expression for a shaping filter that synthesizes the Jakes spectrum from Gaussian white noise is:

$$h_j(t) = 2^{1/4}\Gamma(\frac{3}{4}) f_m (2\pi f_m t)^{-1/4} J_{1/4}(2\pi f_m t) \qquad (3\text{-}59)$$

where:

Γ : is the Gamma function.

f_m : is the Doppler Bandwidth of the channel.

$J_{1/4}$: is the fractional Bessel function.

Its form is as follows:

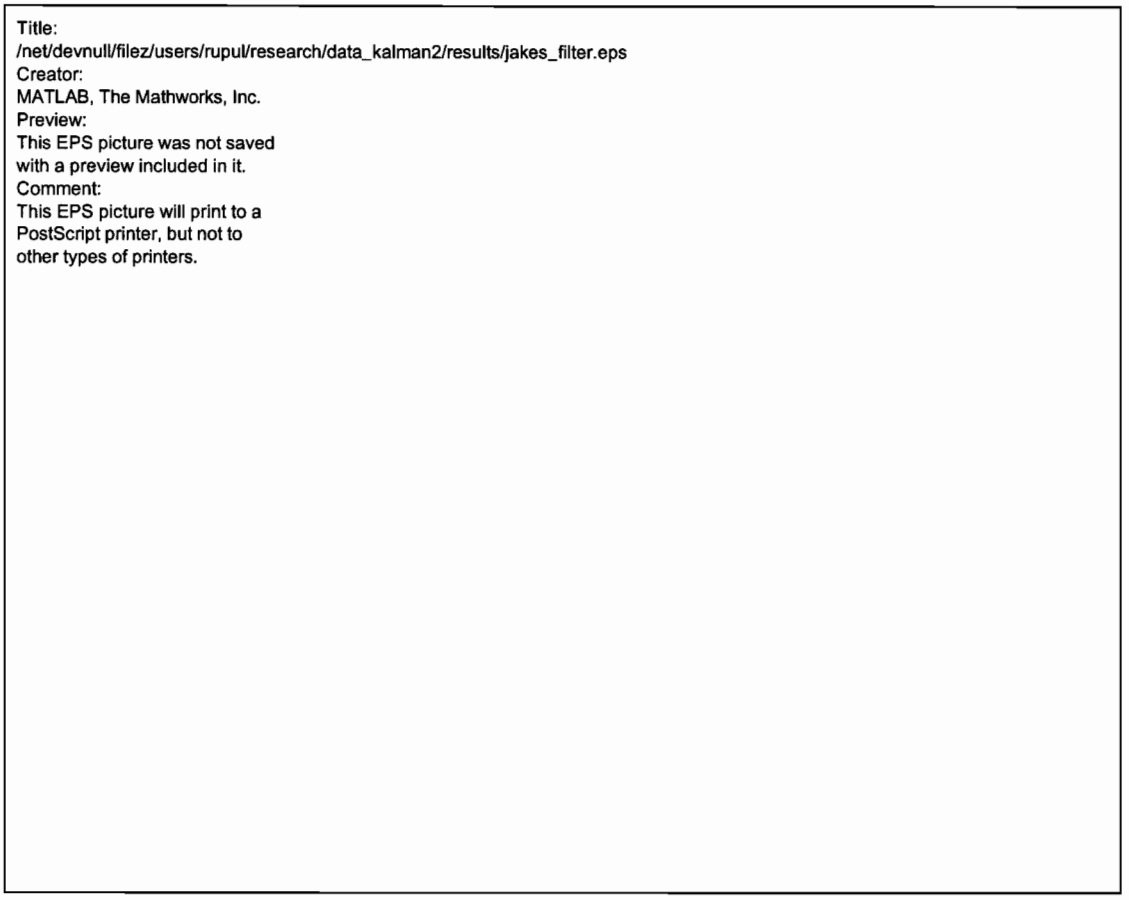

Figure 3-5: Impulse response of the Jakes channel-shaping filter.

This filter is used to synthesize the Jakes tap gain process. In order to derive the AR coefficients from this filter, we sample it. The sampling frequency is chosen to be 16 times the Doppler bandwidth. This is done to avoid any aliasing or frequency warping *[3]*.

$$h_j(n) = h_j\left(\frac{n}{F_S}\right) \quad n = 0.....M\text{-}1 \qquad (3\text{-}60)$$

Where M (in this case) is the number of taps in the FIR Jakes shaping filter.

$$F_S = 16 \times f_m \qquad (3\text{-}61)$$

If the input of this filter is gaussian white noise then the output of the shaping filter is simply the convolution sum:

$$S(n) = \sum_{m=0}^{M-1} h_j(m) w(n-m) \qquad (3\text{-}62)$$

Where $w(n)$ is gaussian white noise.

The convolution sum itself is nothing but a weighted moving average of the white noise inputs. A moving average model has the following form:

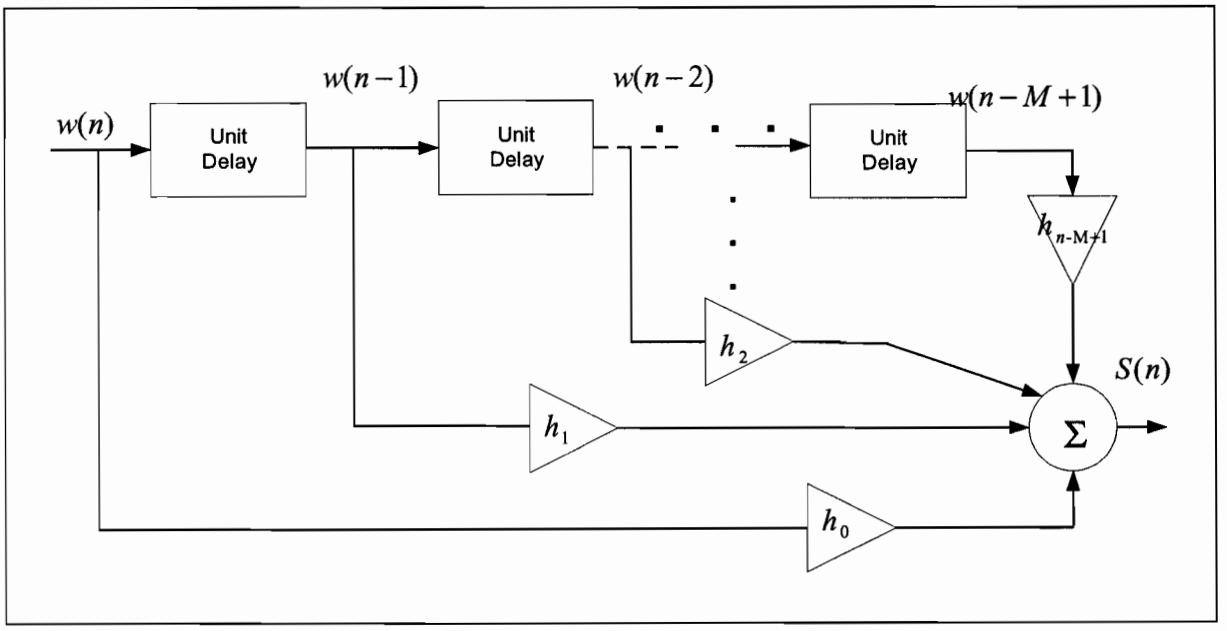

Figure 3-6: Convolution as a moving average.

We will convert the MA form of the Jakes process into its equivalent AR form. We will do this using the method described in*[8]*.

In order to convert the convolution equation to its equivalent AR form, we define the delay operator z^{-1} as:

$$\begin{aligned} z^{-1}S(n) &= S(n-1) \\ z^{-j}S(n) &= S(n-j) \end{aligned} \tag{3-63}$$

Using the delay operator, equation (3-63) can be written as:

$$S(n) = \left(\sum_{m=0}^{M-1} h_j(m)\left(z^{-1}\right)^m \right) w(n) \tag{3-64}$$

$$S(n) = \psi(z^{-1})w(n) \tag{3-65}$$

Where $$\psi(z^{-1}) = \sum_{m=0}^{M-1} h_j(m)\left(z^{-1}\right)^m \tag{3-66}$$

$\psi(z^{-1})$ can be regarded as polynomials in the powers of $\left(z^{-1}\right)$. This allows us to rewrite (3-65) as:

$$\frac{1}{\psi(z^{-1})} S(n) = w(n) \tag{3-67}$$

In order to further simplify the equation above, we need to simplify $\frac{1}{\psi(z^{-1})}$ first.

$$\frac{1}{\psi(z^{-1})} = \frac{1}{h_0 + h_1 z^{-1} + h_2 z^{-2} \ldots\ldots h_{M-1}\left(z^{-1}\right)^{M-1}} \tag{3-68}$$

This rational function can be expanded using partial functions as follows:

$$\frac{1}{\psi(z^{-1})} = \frac{r_1}{1 - p_1 z^{-1}} + \frac{r_2}{1 - p_2 z^{-2}} + \ldots\ldots\ldots \frac{r_{M-1}}{1 - p_{M-1}\left(z^{-1}\right)^{M-1}} \tag{3-69}$$

Since $$\begin{aligned} \frac{1}{1 - pz^{-1}} &= \sum_{i=0}^{\infty} \left(pz^{-1}\right)^i \\ &= \left[1 + pz^{-1} + \left(pz^{-1}\right)^2 + \left(pz^{-1}\right)^3 \ldots\ldots \infty\right] \end{aligned} \tag{3-70}$$

We can write equation (3-69) as:

$$\frac{1}{\psi(z^{-1})} = r_1\left[1 + p_1 z^{-1} + \left(p_1 z^{-1}\right)^2 + \left(p_1 z^{-1}\right)^3 .....\infty\right] +$$
$$r_2\left[1 + p_2 z^{-1} + \left(p_2 z^{-1}\right)^2 + \left(p_2 z^{-1}\right)^3 .....\infty\right] +$$
$$.$$
$$.$$
$$r_{M-1}\left[1 + p_{M-1} z^{-1} + \left(p_{M-1} z^{-1}\right)^2 + \left(p_{M-1} z^{-1}\right)^3 .....\infty\right] \quad (3\text{-}71)$$

Simplifying the equation above we get:

$$\frac{1}{\psi(z^{-1})} = \sum_{i=1}^{M-1} r_i + z^{-1}\sum_{i=1}^{M-1} r_i p_i + z^{-2}\sum_{i=1}^{M-1} r_i p_i^2 .....\infty \quad (3\text{-}72)$$

This is an infinite series sum. Truncating this to N+1 terms we get:

$$\frac{1}{\psi(z^{-1})} = \sum_{i=1}^{M-1} r_i + z^{-1}\sum_{i=1}^{M-1} r_i p_i + z^{-2}\sum_{i=1}^{M-1} r_i p_i^2 .....z^{-N}\sum_{i=1}^{M-1} r_i p_i^N \quad (3\text{-}73)$$

For conciseness, we denote the summation terms in the equation above as follows:

$$\frac{1}{\psi(z^{-1})} = \Pi_0 + z^{-1}\Pi_1 + z^{-2}\Pi_2 .....z^{-N}\Pi_N \quad (3\text{-}74)$$

Where
$$\Pi_n = \sum_{i=1}^{M-1} r_i p_i^n \quad (3\text{-}75)$$

Inserting equation (3-74) into (3-67) we get:

$$\left(\Pi_0 + z^{-1}\Pi_1 + z^{-2}\Pi_2 .....z^{-N}\Pi_N\right)S(n) = w(n) \quad (3\text{-}76)$$

Recall that z^{-N} is the delay operator. Simplifying the equation above, we get:

$$\Pi_0 S(n) = -\Pi_1 S(n-1) - \Pi_2 S(n-2) ..... - \Pi_N S(n-N) + w(n) \quad (3\text{-}77)$$

$$S(n) = \frac{1}{\Pi_0}\left\{-\Pi_1 S(n-1) - \Pi_2 S(n-2) ..... - \Pi_N S(n-N) + w(n)\right\} \quad (3\text{-}78)$$

Compare this equation to the auto-regressive system equation, restated here.

$$S(n) = \sum_{i=1}^{N} \phi_i S(n-i) + w(n) \quad (3\text{-}79)$$

Or $$S(n) = \phi_1 S(n-1) + \phi_2 S(n-2) + ...\phi_N S(n-N) + w(n) \quad (3\text{-}80)$$

We can clearly see the relationship between the moving average and auto regressive coefficients. By comparison, we get the following expression for the auto-regressive coefficients derived from the moving average model:

$$\phi_n = -\frac{\Pi_n}{\Pi_0} \quad (3\text{-}81)$$

We must note here that for a moving average system driven by unity variance gaussian noise is equivalent to an autoregressive process where the plant noise driving the autoregressive system is given by:

$$\sigma_w^2 = \left(\frac{1}{\Pi_0}\right)^2 \quad (3\text{-}82)$$

The figure below summarizes the two equivalent forms for representing a linear process.

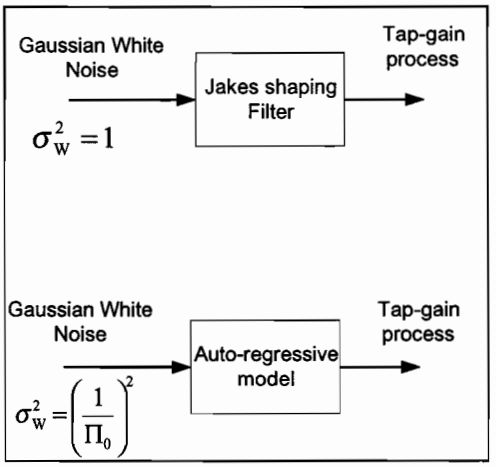

Figure 3-7: Two equivalent representations of the tap gain system.

This procedure is used to calculate the parameters for the autoregressive model from the Jakes shaping filter.

Model validation

The MA model of the FIR Jakes shaping filter is used to calculate the AR parameters. But a finite length MA model is represented by an infinite length AR model. An infinite length model is of course impractical and is truncated to a finite length. The

length of the AR model effects the spectrum of the process modeled. The true Jakes spectrum is modeled by an infinite order AR process. We will study how faithfully lower order AR models represent the Jakes model.

Consider the following parameters for the channel:

- $f_m = 50Hz$. This is the Doppler bandwidth of the channel.
- $F_S = 16 \times f_m$. This is the Jakes-shaping-filter sampling rate.
- $F_{S_{PSD}} = 5 \times 10^3 samples / Hz$. This is the Jakes spectrum sampling frequency.
- $N_{MA} = 64$. This is the FIR shaping filter length.
- N is the length of the truncated AR model.

We will compare the analytical autocorrelation and the power spectral density of the Jakes model with the calculated autocorrelation and spectrum for different length AR processes.

Once having calculated the parameters of the AR model, the power spectrum and the autocorrelation are calculated as follows:

- $$S_{SS}(f) = \frac{\sigma_W^2}{\left|1 - \sum_{i=1}^{N} \phi_i e^{-j2\pi fi}\right|^2} \qquad (3\text{-}83)$$

where σ_W^2 is the AR model plant noise.

- It is not easy to derive an analytical expression for the autocorrelation. So we estimate it from the process created using the AR models of different length. In each case a process 4096 samples long was created.

The figures below show how truncated AR processes of different length compare with the true Jakes spectrum.

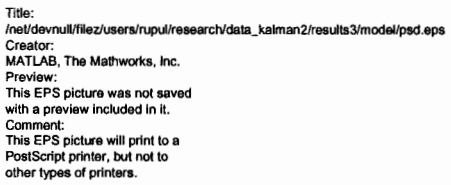

Figure 3-8: Jakes spectrum compared with truncated AR processes'spectrum.

The next figure shows the actual Jakes autocorrelation as compared to the those derived from truncated AR processes.

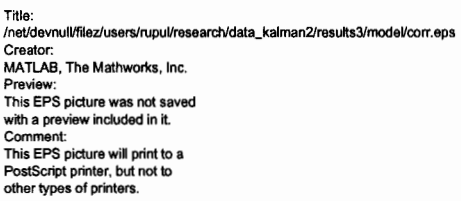

Figure 3-9: Jakes autocorrelation compared to those from truncated AR processes.

As expected, the longer AR model is a more faithful representation of the Jakes model. But this must of course be balanced against a longer startup time (in the channel estimation procedure since N taps have to be initialized) and a higher computational load.

3.4.2.2 Kalman filter derivation.

Once more, in order to track this process with a Kalman filter, we will need to represent the AR model and the data estimate in a State-Space form. In the previous section we tracked the process with a scalar Kalman filter. We will track the Jakes process with a vector Kalman filter.

System Model:

In section (3.4.2.1), we derived a method of calculating the AR form of the Jakes process from the shaping filter. Now, the autoregressive coefficients are used to form a concise system equation for the tap gain process.

The auto-regressive form of the tap gain process is as follows:

$$S(n) = \sum_{i=1}^{N} \phi_i S(n-i) + w(n) \qquad (3\text{-}84)$$

where $\phi_i = -\frac{?_i}{?_0}$ are the coefficients of the auto-regressive process derived from the Jakes spectrum.

N: The number of taps in the auto-regressive model.

The matrix form of this equation is:

$$\begin{bmatrix} S(n) \\ S(n\text{-}1) \\ . \\ . \\ S(n\text{-}N+1) \end{bmatrix} = \begin{bmatrix} \phi_1 & \phi_2 & . & . & \phi_N \\ 1 & . & . & 0 & 0 \\ 0 & 1 & . & 0 & 0 \\ 0 & . & . & . & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} S(n\text{-}1) \\ S(n\text{-}2) \\ . \\ . \\ S(n\text{-}N) \end{bmatrix} + \begin{bmatrix} w(n) \\ 0 \\ . \\ . \\ 0 \end{bmatrix} \qquad (3\text{-}85)$$

Expressed more concisely as vectors, the system equation defining the Jakes autoregressive process is:

$$\bar{S}(n) = A\bar{S}(n-1) + \bar{W}(n) \tag{3-86}$$

where:

$$A = \begin{bmatrix} \phi_1 & \phi_2 & . & . & \phi_N \\ 1 & . & . & 0 & 0 \\ 0 & 1 & . & 0 & 0 \\ 0 & . & . & . & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{3-87}$$

and the plant noise covariance matrix is given by:

$$Q = E\left[\bar{W}(n)\left(\bar{W}(n)\right)^H\right] = \begin{bmatrix} s_w^2 & 0 & 0...0 \\ 0 & 0..... & 0 \\ .0 & 0 & 0..... \ 0 \\ 0 & 0 & ..... \ 0 \end{bmatrix} \tag{3-88}$$

Observation Model:

The data based estimator produces 'estimates' of the process. It can therefore can be looked upon as a noisy version of the process. The data estimate at any time can be written as:

$$X(n) = S(n) + v(n) \tag{3-89}$$

where: *S(n):* The process at time n.

X(n): The data based estimate of S(n)

v(n): Error of the data based estimate.

$s_v^2 = \dfrac{\sigma_c^2}{M}$ from equation (3-22). (3-90)

M: Number of training sequence bits in each frame.

In order to be consistent with the matrix form of the system equation. we can write equation (3-89) as:

$$\begin{bmatrix} X(n) \\ X(n\text{-}1) \\ . \\ . \\ X(n\text{-}N+1) \end{bmatrix} = \begin{bmatrix} S(n) \\ S(n\text{-}1) \\ . \\ . \\ S(n\text{-}N+1) \end{bmatrix} + \begin{bmatrix} v(n) \\ v(n\text{-}1) \\ . \\ . \\ v(n\text{-}N+1) \end{bmatrix} \tag{3-91}$$

Or expressed concisely in the form of vectors the equation above becomes:

$$\bar{X}(n) = H \times \bar{S}(n) + \bar{v}(n) \tag{3-92}$$

where H is an $(N \times N)$ identity matrix (3-93)

and the observation noise covariance is givenby:

$$R = E\left[\bar{v}(n)\left(\bar{v}(n)\right)^{H}\right] = s_{v}^{2} \times \begin{bmatrix} 1 & 0 & 0...0 \\ 0 & 1..... & 0 \\ .0 & 0 & 1..... \ 0 \\ 0 & 0 & ..... \ 1 \end{bmatrix} \tag{3-94}$$

Kalman filter for the higher order AR process

Given the state space formulation above, for the Jakes tap gain processes, we can use the vector Kalman filter to track them. The vector Kalman filter equations are as follows *[6],[5].[10]*:

- The initial conditions are:

$$\hat{S}(0) = E\{S(n)\} = \text{zero matrix of length } (N \times L)^{3} \tag{3-95}$$

$$P(1) \geq \left\{\sigma_{w}^{2}[I] and \sigma_{v}^{2}[I]\right\} \tag{3-96}$$

- The Kalman gain is given by:

$$K(n) = P(n)H^{T}[HP(n)H^{T} + R]^{-1} \tag{3-97}$$

- The current estimate of the process, given the data estimate is given by:

---

[3] From now on all variables are vectors unless otherwise stated. The overbar representation for a vector is dropped for the sake of readability.

$$\hat{S}_{curr}(n)=\hat{S}(n)+K(n)[X(n)-H\hat{S}(n)] \quad (3\text{-}98)$$

- The predicted estimate of the process, is given by:

$$\hat{S}(n+1)=A\{\hat{S}_{curr}(n)\} \quad (3\text{-}99)$$

- The current error covariance is given by:

$$P_{curr}(n)=[I-K(n)H]P(n) \quad (3\text{-}100)$$

- The predicted error covariance is given by:

$$P(n+1)=A\{P_{curr}(n)\}A^T+Q \quad (3\text{-}101)$$

These equations define the vector Kalman filter. Its actual implementation as an algorithm is explained next.

3.4.2.3 Simulation and Results:

In this section the vector Kalman filter is used to track the Jakes process. The Kalman algorithm is explained in more detail and its performance is analyzed.

Simulation Parameters:

- System equation:

$$\bar{S}(n)=A\bar{S}(n-1)+\bar{W}(n) \quad (3\text{-}102)$$

Where

- $N=5$. This is the number of taps in the AR model.
- $$\phi=[.9086,-0.0590,-0.0548,-0.0486,-0.0409] \quad (3\text{-}103)$$
- $$\Pi_0=12.20 \quad (3\text{-}104)$$
- $$A=\begin{bmatrix} .9086 & -0.0590 & -0.0548 & -0.0486 & -0.0409 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \quad (3\text{-}105)$$

- Observation Equation:

$$\bar{X}(n) = H \times \bar{S}(n) + \bar{v}(n) \qquad (3\text{-}106)$$

Where H : is an $(N \times N)$ identity matrix (3-107)

- The Doppler bandwidth is $f_m = 500Hz$
- The frame rate is : $R_F = 5 \times 10^4$ *Frames/* sec . The simulation is run at the frame rate.
- $M = 8$: The length of the training sequence. The channel is assumed to be invariant for these M bits.
- The signal to noise ratio of the channel, for $E_B = 1$ is:

$$\frac{E_B}{N_O} = \frac{E_B}{2\sigma_c^2} = 6dB$$

Thus $\sigma_c^2 = .1256$

- The variance of the tap gain plant noise is $s_w^2 = 2\sigma_V^2 = 0.0314$ (3-108)
- $$Q = 0.0314 \times \begin{bmatrix} 1 & 0 & 0 & \ldots\ldots & 0 \\ 0 & \ldots\ldots & \ldots & \ldots & 0 \\ 0 & 0 & \ldots\ldots & . & 0 \\ 0 & 0 & \ldots\ldots & \ldots & 0 \end{bmatrix} \qquad (3\text{-}109)$$
- The data estimator variance $s_V^2 = \frac{\sigma_c^2}{M} = 0.0157$ (3-110)

$$R = .0157. \times \begin{bmatrix} 1 & 0 & 0 & \ldots & 0 \\ 0 & 1 & \ldots\ldots & . & 0 \\ 0 & 0 & 1 & \ldots\ldots & 0 \\ 0 & 0 & \ldots\ldots & \ldots & 1 \end{bmatrix} \qquad (3\text{-}111)$$

The vector Kalman filter defined above is to be used to track the $N = 5$ AR process. For simulation purposes we must first create this process before we can track it.

Process Synthesis:

The Jakes process is synthesized using the shaping filter. As we have seen previously, a moving average system driven by unity variance gaussian noise is

equivalent to an autoregressive process where the plant noise driving the autoregressive system is given by:$\sigma_w^2 = (1/\Pi_0)^2$. This relationship can be reversed in our case to produce a process whose equivalent AR process is driven by unity variance noise. The figure below illustrates this idea:

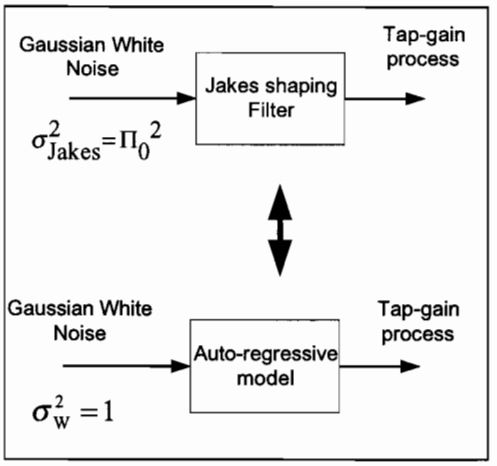

Figure 3-10: **Jakes Process synthesis.**

We use the Jakes shaping filter driven by gaussian noise of variance $\sigma^2_{Jakes} = (\Pi_0)^2$ to produce the tap gain process desired.

Algorithm Implementation:

The channel estimation algorithm utilizing the Vector Kalman filter is as follows:

{ Initiallize:

Time:$(n = 1)$

- $P(1) = (N \times N)$ identity. Assume initial predicted error covariance.
- $\hat{S}(1) = [0....0]^T$. Assume initial prediction is an $(N \times 1)$ zero vector.
- {Start Iteration:
    - Perform data based estimation, i.e. get $X(n)$.
    - Calculate Kalman gain:

$$K(n) = P(n)H^T[HP(n)H^T + R]^{-1}$$

- Calculate current estimate:

$$\hat{S}_{curr}(n)=\hat{S}(n)+K(n)[X(n)-H\hat{S}(n)]$$

- Calculate current error covariance:

$$P_{curr}(n)=[I-K(n)H]P(n)$$

- Predict ahead:

$$\hat{S}(n+1)=A\{\hat{S}_{curr}(n)\}$$

- Predict the error covariance:

$$P(n+1)=A\{P_{curr}(n)\}A^T+Q$$

- *Time:* $(n=n+1)$

End loop }

Results:

The Jakes process created above is tracked with a vector Kalman filter. The figure below shows the results of the simulation:

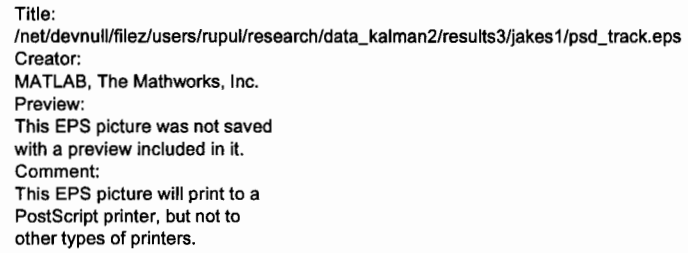

Figure 3-11: Channel estimation of a Jakes process.

Error Covariance

The error covariance matrices are defined as follows:

The current error covariance is:

$$P_{curr} = E\left\{\left[S(n) - \hat{S}_{curr}(n)\right]\left[S(n) - \hat{S}_{curr}(n)\right]^H\right\} \tag{3-112}$$

Recalling the state-space formulation, we can expand the previous equation as:

$$P_{curr} = E\left\{\begin{bmatrix}\tilde{S}(n)\\ \tilde{S}(n-1)\\ .\\ \tilde{S}(n-N+1)\end{bmatrix}\begin{bmatrix}\tilde{S}(n)\\ \tilde{S}(n-1)\\ .\\ \tilde{S}(n-N+1)\end{bmatrix}^{H}\right\} \tag{3-113}$$

We can then interpret the diagonal elements as follows:

$$P_{curr} = \begin{bmatrix} MSE_{(n/n)} ** \ldots\ldots * \\ * \quad MSE_{(n-1/n)} ** \ldots\ldots * \\ * \quad * \quad MSE_{(n-2/n)} ** \ldots\ldots * \\ . \\ .** \ldots\ldots\ldots\ldots\ldots\ldots\ldots MSE_{S(n-N+1/n)} \end{bmatrix} \tag{3-114}$$

where $MSE_{(n/n)}$: the MSE of the current state's estimate.

$MSE_{(n-1/n)}$: the MSE of the previous states estimate etc.

Similarly, the prediction error covariance is defined as:

$$P = E\left\{\left[S(n+1) - \hat{S}(n+1)\right]\left[S(n+1) - \hat{S}(n+1)\right]^{H}\right\} \tag{3-115}$$

$$P = E\left\{\begin{bmatrix}\tilde{S}(n+1)\\ \tilde{S}(n)\\ .\\ \tilde{S}(n-N+2)\end{bmatrix}\begin{bmatrix}\tilde{S}(n+1)\\ \tilde{S}(n)\\ .\\ \tilde{S}(n-N+2)\end{bmatrix}^{H}\right\} \tag{3-116}$$

We can then interpret the diagonal elements as follows:

$$P = \begin{bmatrix} MSE_{(n+1/n)} ** \ldots\ldots * \\ * \quad MSE_{(n/n)} ** \ldots\ldots * \\ * \quad * \quad MSE_{(n-1/n)} ** \ldots\ldots * \\ . \\ .** \ldots\ldots\ldots\ldots\ldots\ldots\ldots MSE_{S(n-N+2/n)} \end{bmatrix} \tag{3-117}$$

where $MSE_{(n+1/n)}$: the MSE of the prediction.

$MSE_{(n/n)}$: the MSE of the current state's estimate.

We thus see that we are performing prediction, filtering and smoothing all in one filter. We now present the diagonal elements of the steady state covariances:

- The diagonal elements for the steady state current error covariance are:

$$diag(P_{curr_{SS}}) = \begin{bmatrix} 0.0110 \\ 0.0058 \\ 0.0041 \\ 0.0033 \\ 0.0027 \end{bmatrix} \quad (3\text{-}118)$$

- The steady state prediction error covariance is:

$$diag(P_{SS}) = \begin{bmatrix} 0.0403 \\ 0.0110 \\ 0.0058 \\ 0.0041 \\ 0.0033 \end{bmatrix} \quad (3\text{-}119)$$

The actual error co-variances for the simulation were calculated by averaging the results over 100 Monte Carlo runs. The following are the results:

- The simulated current error covariance is:

$$diag(P_{curr_{Sim}}) = \begin{bmatrix} 0.0112 \\ 0.0101 \\ 0.0108 \\ 0.0115 \\ 0.0120 \end{bmatrix} \quad (3\text{-}120)$$

- The simulated prediction error covariance is:

$$diag(P_{Sim}) = \begin{bmatrix} 0.0382 \\ 0.0112 \\ 0.0101 \\ 0.0108 \\ 0.0115 \end{bmatrix} \quad (3\text{-}121)$$

We can see from the results above that if the data based estimate has a MSE of $s_v^2 = 0.0157$, and $diag(P_{Sim})(2,1) = [\ 0.0112\ ]$ then the Kalman filter has significantly improved the current estimate as compared to the data estimate.

The percentage increase is:

$$\frac{\sigma_V^2 - diag(P_{Sim})(2,1)}{\sigma_V^2} = 29\% \qquad (3\text{-}122)$$

In this simulation we have shown that with the help of the underlying Jakes channel model, we can improve the estimate of the channel by a significant amount. We no longer have to wait for the data estimate to arrive either, we can predict the state of the channel with a MSE of 0.0382 .

3.5 Tracking Multiple Jakes tap-gain processes.

We now consider a radio system in which the signal arrives at the receiver from multiple paths. Training sequences are sent regularly over the channel to estimate its impulse response. The data based estimator only gives us a snapshot of the channel for the time that the training sequence is sent. By utilizing the underlying channel model on each of these paths, we intend to improve on the data-only channel estimate without increasing the length of the training sequence.

In section (3.4.2), we had assumed a Jakes channel model for the single path channel. In the multipath case, on each path of the channel, we assume the same underlying Jakes channel model. The Jakes channel model is assumed under the assumptions that the mobile is moving with a constant velocity, there are an infinite number of scatterers on each path and that the angle of arrival for each of these paths is uniformly distributed. Previously, we had expressed the data estimator and the channel model in the state-space form and had derived the vector Kalman filter from that form. For the multipath case, we present a modification on the state space-form to derive the vector Kalman filter based multipath channel estimation algorithm.

3.5.1 Kalman filter derivation

The state-space representation requires us to express the channel model in the form of a system equation and the data based estimator in the form of an observation equation. We sill then use this form to derive the appropriate Kalman filter.

System model:

If we assume that the tap-gain processes are independent, but have the same Jakes spectrum, we will have multiple tap-gain processes but with the same underlying Jakes model. Their auto-regressive representations are:

$$
\begin{aligned}
S_1(n) &= \sum_{i=1}^{N} \phi_i S_1(n-i) + w_1(n) \\
S_2(n) &= \sum_{i=1}^{N} \phi_i S_2(n-i) + w_2(n) \\
&\cdot \\
&\cdot \\
S_L(n) &= \sum_{i=1}^{N} \phi_i S_L(n-i) + w_L(n)
\end{aligned}
\qquad (3\text{-}123)
$$

where: $S_l(n)$ is the l^{th} process to be tracked.

ϕ_i is the AR model parameters. These were calculated in section (3.4.2.1)and are the same for each process.

$w_l(n)$ is the plant noise driving the tap gain function. The relative variance is determined by the power delay profile of the channel.

L is the number of taps (or the number of processes to track) in the radio model.

The matrix form of equation (3-84) for L'processes (i.e. L radio channel model taps) is

$$
\begin{bmatrix} S_1(n) \ldots\ldots\ldots\ldots S_L(n) \\ S_1(n-1) \ldots\ldots\ldots S_L(n-1) \\ . \qquad\qquad . \\ . \\ S_1(n-N+1) \ldots S_L(n-N+1) \end{bmatrix} = \begin{bmatrix} \phi_1 \phi_2 \ldots\ldots\ldots \phi_N \\ 1 \;\; 0 \;\; 0 \ldots\ldots 0 \\ 0 \;\; 1 \ldots\ldots\ldots 0 \\ \ldots\ldots\ldots\ldots\ldots 0 \\ 0 \;\; 0 \ldots\ldots\ldots 1 \;\; 0 \end{bmatrix} \begin{bmatrix} S_1(n-1) \ldots\ldots\ldots S_L(n-1) \\ S_1(n-2) \ldots\ldots\ldots S_L(n-2) \\ . \qquad\qquad . \\ . \\ S_1(n-N) \ldots S_L(n-N) \end{bmatrix} + \begin{bmatrix} w_1(n) \ldots\ldots\ldots w_L(n) \\ 0 \ldots\ldots\ldots\ldots\ldots 0 \\ . \qquad\qquad . \\ . \\ 0 \ldots\ldots\ldots\ldots\ldots 0 \end{bmatrix} \qquad (3\text{-}124)
$$

Expressed in vector form, the system equation defining the multipath tap-gain processes is:

$$\bar{S}(n) = A\bar{S}(n-1) + \bar{W}(n) \tag{3-125}$$

where:

$$A = \begin{bmatrix} \phi_1 \phi_2 \ldots\ldots \phi_N \\ 1 \quad 0 \quad 0 \ldots\ldots 0 \\ 0 \quad 1 \ldots\ldots \quad 0 \\ \ldots\ldots\ldots\ldots \quad 0 \\ 0 \quad 0 \ldots\ldots 1 \quad 0 \end{bmatrix} \tag{3-126}$$

and the $(L \times L)$ plant noise covariance matrix is given by:

$$Q = E\left[\bar{W}(n)\left(\bar{W}(n)\right)^H\right] = \begin{bmatrix} \sum_{l=1}^{L} s_w^2(l) \ldots\ldots 0 \\ 0 \ldots\ldots\ldots\ldots 0 \\ 0 \ldots\ldots\ldots\ldots 0 \\ . \\ 0 \ldots\ldots\ldots\ldots 0 \end{bmatrix} \tag{3-127}$$

Observation Model:

The data based estimate for each process is independent of the other, so we are justified in the following data based estimate model:

$$\begin{aligned} X_1(n) &= S_1(n) + v_1(n) \\ X_2(n) &= S_2(n) + v_2(n) \\ &. \\ &. \\ X_L(n) &= S_L(n) + v_L(n) \end{aligned} \tag{3-128}$$

where: $S_l(n)$: The $l^{'th}$ process at time n.

$X_l(n)$: The $l^{'th}$ data based estimate of $S(n)$

$v_l(n)$: Error of the $l^{'th}$ data based estimate.

$s_v^2 = \frac{\sigma_c^2}{M}$ is assumed to be the same for all paths. The assumptions are explained below.

M: Number of training sequence bits in each frame.

We can write equation (3-89) as:

$$\begin{bmatrix} X_1(n) & . & . & X_L(n) \\ X_1(n-1) & . & . & X_L(n-1) \\ . & . & . & . \\ . & . & . & . \\ X_1(n-N+1) & . & . & X_L(n-N+1) \end{bmatrix} = \begin{bmatrix} S_1(n) & . & . & S_L(n) \\ S_1(n-1) & . & . & S_L(n-1) \\ . & . & . & . \\ . & . & . & . \\ S_1(n-N+1) & . & . & S_L(n-N+1) \end{bmatrix} + \begin{bmatrix} v_1(n) & . & . & v_L(n) \\ v_1(n-1) & . & . & v_L(n-1) \\ . & . & . & . \\ . & . & . & . \\ v_1(n-N+1) & . & . & v_L(n-N+1) \end{bmatrix} \quad (3\text{-}129)$$

Or expressed concisely in the form of vectors the equation above becomes:

$$\bar{X}(n) = H \times \bar{S}(n) + \bar{v}(n) \quad (3\text{-}130)$$

where:

$$H : \text{is an } (N \times N) \text{ identity matrix} \quad (3\text{-}131)$$

and the observation noise covariance is given by:

$$R = E[v(n)(v(n))^H] = (L\sigma_v^2) \times [I] \quad (3\text{-}132)$$

where $[I]$ is an $(N \times N)$ identity matrix.

We have assumed here that the autocorrelation of the training sequence is ideal. The MSE of the data estimation on all the paths will then be the same.

Kalman filter for multiple AR processes

Using the state space equations above, we derive a vector Kalman filter to track the tap-gain processes. The vector Kalman filter structure will be the same as for the single path case but the parameters of the filter are different.

- The initial conditions are:

$$\hat{S}(0)=E\{S(n)\}=\text{zero vector of length } (N\times 1) \quad [4] \qquad (3\text{-}133)$$

$$P(1) \geq \left\{\sigma_w^2[I] and \sigma_v^2[I]\right\} \qquad (3\text{-}134)$$

- The Kalman gain is given by:

$$K(n) = P(n)H^T[HP(n)H^T + R]^{-1} \qquad (3\text{-}135)$$

- The current estimate of the process, given the data estimate is given by:

$$\hat{S}_{curr}(n)=\hat{S}(n) + K(n)[X(n)-H\hat{S}(n)] \qquad (3\text{-}136)$$

- The predicted estimate of the process, is given by:

$$\hat{S}(n+1)=A\{\hat{S}_{curr}(n)\} \qquad (3\text{-}137)$$

- The current error covariance is given by:

$$P_{curr}(n) = [I - K(n)H]P(n) \qquad (3\text{-}138)$$

- The predicted error covariance is given by:

$$P(n+1) = A\{P_{curr}(n)\}A^T + Q \qquad (3\text{-}139)$$

These equations define the vector Kalman filter. Its actual implementation as an algorithm is explained next.

---

[4] All variables are vectors unless otherwise stated. The overbar representation for a vector is dropped for the sake of readability.

3.5.2 Simulation and Results:

In this section the vector Kalman filter is used to track the Jakes processes. The parameters for the Kalman algorithm are explained in more detail and its performance is analyzed.

Simulation Parameters:

- System equation as defined in the previous section:

$$\bar{S}(n) = A\bar{S}(n-1) + \bar{W}(n) \tag{3-140}$$

Where

- $N=5$. This is the number of taps in the AR model.
- $L=3$: This is the number of tap-gain processes being tracked.
- $$\phi = [.9086, -0.0590, -0.0548, -0.0486, -0.0409] \tag{3-141}$$
- $$\Pi_0 = 12.20 \tag{3-142}$$
- $$A = \begin{bmatrix} .9086 & -0.0590 & -0.0548 & -0.0486 & -0.0409 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{3-143}$$
- Observation Equation:

$$\bar{X}(n) = H \times \bar{S}(n) + \bar{v}(n) \tag{3-144}$$

Where $$H : \text{is an } (N \times N) \text{ identity matrix} \tag{3-145}$$

- The Doppler bandwidth is $f_m = 500Hz$
- The frame rate is : $R_F = 5 \times 10^4 \, Frames/\sec$. The simulation is run at the frame rate
- $M = 8$: The length of the training sequence. The channel is assumed to be invariant for these M bits.
- The signal to noise ratio of the channel, for $E_B = 1$ is:

$$\frac{E_B}{N_O} = \frac{E_B}{2\sigma_c^2} = 6dB$$

Thus $\sigma_c^2 = .1256$

- The power delay profile $s_w^2(l) = 0.0314 \times [1, 0.9, 0.81]$ (3-146)
- The covariance of the plant noise is:

$$Q = \begin{bmatrix} 0.0851 & . & 0 \\ . & . & . \\ 0 & . & 0 \end{bmatrix} \quad (3\text{-}147)$$

- The data estimator variance $s_v^2 = \frac{\sigma_c^2}{M} = 0.0157$. Here the assumption is made that the training sequence has an ideal thumbtack autocorrelation.

$$R = 3 \times 0.0157. \times \begin{bmatrix} 1 & 0 & 0 & \ldots & 0 \\ 0 & 1 & \ldots\ldots & & 0 \\ .0 & 0 & 1 & \ldots\ldots & 0 \\ 0 & 0 & \ldots\ldots & & 1 \end{bmatrix} = \begin{bmatrix} .0471 & 0 & 0 & 0 & 0 \\ 0 & .0471 & 0 & 0 & 0 \\ 0 & 0 & .0471 & 0 & 0 \\ 0 & 0 & 0 & .0471 & 0 \\ 0 & 0 & 0 & 0 & .0471 \end{bmatrix}$$

(3-148)

Process Synthesis:

In the single process case, the process was synthesized from the Jakes filter. In the multiple tap case, we have to synthesize L processes with relative variances scaled by the power delay profile of the channel.

We are given that the power delay profile $s_w^2(l) = 0.0314 \times [1, 0.9, 0.81]$. The processes are easily created, by scaling the input noise of the Jakes filter by the power delay profile. Thus the input noise variance for the Jakes filter is given by:

$$\sigma_{Jakes}^2(l) = \Pi_o^2 \sigma_W^2(l) \quad (3\text{-}149)$$

Algorithm Implementation:

The channel estimation algorithm utilizing the Vector Kalman filter changes very slightly for the multipath case:

{ Initiallize:

Time:$(n=1)$

- $P(1)=(N\times N)$ identity. Assume initial predicted error covariance.
- $\hat{S}(1)=\begin{bmatrix}0.....0\\0.....0\\.\\0.....0\end{bmatrix}^T$ Assume initial prediction is an*(NxL)* zero matrix.
- {Start Iteration:
    - Perform data based estimation, i.e. get $X(n)$.
    - Calculate Kalman gain:

$$K(n)=P(n)H^T[HP(n)H^T+R]^{-1}$$

- Calculate current estimate:

$$\hat{S}_{curr}(n)=\hat{S}(n)+K(n)[X(n)-H\hat{S}(n)]$$

- Calculate current error covariance:

$$P_{curr}(n)=[I-K(n)H]P(n)$$

- Predict ahead:

$$\hat{S}(n+1)=A\{\hat{S}_{curr}(n)\}$$

- Predict the error covariance:

$$P(n+1)=A\{P_{curr}(n)\}A^T+Q$$

    - *Time:* $(n=n+1)$

End loop }

Results:

The multiple Jakes processes are tracked by a single Kalman filter. The figures below show the results of the simulation.

Title:
/net/devnull/filez/users/rupul/research/data_kalman2/results3/jakesm/track1.eps
Creator:
MATLAB, The Mathworks, Inc.
Preview:
This EPS picture was not saved
with a preview included in it.
Comment:
This EPS picture will print to a
PostScript printer, but not to
other types of printers.

Figure 3-12: Channel estimation of the first process

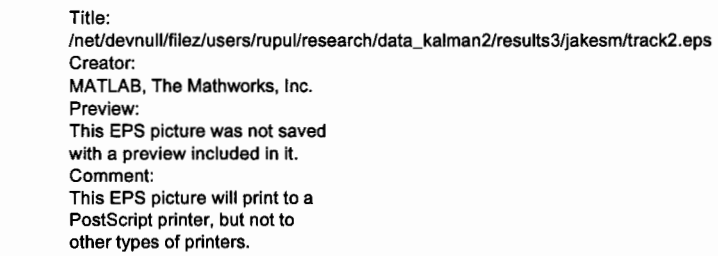

Figure 3-13: Channel estimation of the second process

Title:
/net/devnull/filez/users/rupul/research/data_kalman2/results3/jakesm/track3.eps
Creator:
MATLAB, The Mathworks, Inc.
Preview:
This EPS picture was not saved
with a preview included in it.
Comment:
This EPS picture will print to a
PostScript printer, but not to
other types of printers.

Figure 3-14: Channel estimation of the third process

Error Covariance:

The error covariance matrices are defined as fdlows:

The current error covariance is:

$$P_{\text{curr}} = E\left\{\left[S(n) - \hat{S}_{\text{curr}}(n)\right]\left[S(n) - \hat{S}_{\text{curr}}(n)\right]^{H}\right\} \tag{3-150}$$

Recalling the state-space formulation, we can expand the previous equation as:

$$P_{curr} = E\left\{\begin{bmatrix} S_1(n) & \ldots & S_L(n) \\ S_1(n\text{-}1) & \ldots & S_1(n\text{-}1) \\ . & & . \\ . & & \\ S_1(n\text{-}N+1) & \ldots & S_L(n\text{-}N+1) \end{bmatrix}\begin{bmatrix} S_1(n) & \ldots & S_L(n) \\ S_1(n\text{-}1) & \ldots & S_1(n\text{-}1) \\ . & & . \\ . & & \\ S_1(n\text{-}N+1) & \ldots & S_L(n\text{-}N+1) \end{bmatrix}^{H}\right\} \tag{3-151}$$

We can then interpret the diagonal elements as follows:

$$P_{curr} = \begin{bmatrix} \sum_l MSE_{(n/n)}(l) & * & * & \cdots\cdots * \\ * & \sum_l MSE_{(n-1/n)}(l) & * & \cdots\cdots * \\ * & * & \sum_l MSE_{(n-2/n)}(l) & \cdots\cdots * \\ . & & & \\ . * * \cdots\cdots & & & \sum_l MSE_{(n-N+1/n)}(l) \end{bmatrix} \quad (3\text{-}152)$$

where $\sum_l MSE_{(n/n)}(l)$: the sum of the MSE of the current state's estimate on all processes.

$\sum_l MSE_{(n-1/n)}(l)$: the MSE of the previous states estimate etc. on all processes.

Similarly, the prediction error covariance is defined as:

$$P = E\left\{\left[S(n+1) - \hat{S}(n+1)\right]\left[S(n+1) - \hat{S}(n+1)\right]^H\right\} \quad (3\text{-}153)$$

We can then interpret the diagonal elements as follows:

$$P = \begin{bmatrix} \sum_l MSE_{(n+1/n)}(l) & * & * & \cdots\cdots * \\ * & \sum_l MSE_{(n/n)}(l) & * & \cdots\cdots * \\ * & * & \sum_l MSE_{(n-1/n)}(l) & \cdots\cdots * \\ . & & & \\ . * * \cdots\cdots & & & \sum_l MSE_{(n-N+2/n)}(l) \end{bmatrix} \quad (3\text{-}154)$$

where $\sum_l MSE_{(n+1/n)}(l)$: the sum of the MSE of the predicted state estimate on all processes.

$\sum_l MSE_{(n/n)}(l)$: the MSE of the current states estimate etc. on all processes.

Finally the following are the steady state and simulated error covariances. The actual error covariances for the simulation were calculated by averaging the results over 100 Monte Carlo runs.

$diag\,(P_{Sim}(l))$	Process # (l)			$diag\ (P_{Sim})$	$diag\ (P_{SS})$	
	1	2	3			
	0.0384	0.0351	0.0321	0.1056	0.1111	
	0.0112	0.0109	0.0106	0.0327	0.0321	
	0.0098	0.0097	0.0097	0.0292	0.0169	
	0.0105	0.0105	0.0105	0.0315	0.0122	
	0.0112	0.0111	0.0112	0.0335	0.0096	
$diag\,(P_{Curr_{Sim}}(l))$				$diag\,(P_{Curr_{Sim}})$	$diag\,(P_{Curr_{SS}})$	
	0.0112	0.0109	0.0106	0.0327	0.0321	
	0.0098	0.0097	0.0097	0.0292	0.0169	
	0.0105	0.0105	0.0105	0.0315	0.0122	
	0.0112	0.0111	0.0112	0.0335	0.0096	
	0.0117	0.0117	0.0117	0.0351	0.008	

The first three columns show the process by process error covariance for the current and the next state. These MSE are then added up in the fourth column to produce the simulated error covariance (prediction) and the current error covariance. We can see that the simulated performance is close to the expected performance.

The goal of this channel estimation algorithm was to perform better than the data based estimate by using the underlying channel model. We have already proved that the Kalman filter based estimator does this for the single process case. We now analyze the results for the multiple tap case.

- The data based estimate has a MSE of $s_v^2 = 0.0157$ on each path.
- Comparing this to the individual path MSE (i.e. $diag(P_{curr_{sim}})(1,1)$ for each path), the improvement is given by $\frac{\sigma_V^2 - diag(P_{curr_{sim}})(1,1)}{\sigma_V^2}$.
- The improvement for each path is:
  - Path 1 improvement: $\frac{\sigma_V^2 - diag(P_{curr_{sim}})(1,1)}{\sigma_V^2} = \frac{.0157 - .0112}{.0157} \times 100 = 28.66\%$
  - Path 2 improvement: $\frac{.0157 - .0109}{.0157} \times 100 = 30.57\%$

- Path 3 improvement: $\frac{.0157 - .0106}{.0157} \times 100 = 32.48\%$

We can see that there is an almost *30%* gain in MSE performance compared to the data-only estimator. The Kalman filter based estimator has accomplished this by utilizing the underlying channel model without increasing the length of the training sequence sent.

4 Conclusions

The main contribution of this thesis is the development of a Kalman filter based channel estimation algorithm. We considered a multipath radio channel with a time varying impulse response. Training sequences are sent periodically to produce snapshot estimates of the channel. The performance of this data based estimator is dependent upon the length of the training sequence. Moreover we have estimates of the channel available only at the instants when the data estimate is available. The Kalman filter based channel estimator addressed both these issues.

We assumed a Jakes model for the channel. This, in conjunction with the data estimator allowed us to formulate the Kalman algorithm. As a consequence of using the underlying channel model, we can now predict the state of the channel (with lesser accuracy of course) without having to wait for the data estimate to arrive. The Kalman estimator improved upon the performance of the data estimator by almost thirty percent on each path. Since the only way (assuming channel noise is not under the operators control) to increase the accuracy of the data estimate is to increase the length of the training sequence, the Kalman estimator provides an efficient technique of improving the channel estimate without wasting anymore bandwidth. In radio systems where bandwidth is prohibitively expensive or there is just no more room on the frame for any more training sequence information, the Kalman estimator solution becomes even more attractive.

5 Future Work

The work presented in this thesis can be extended various ways including the following:

- *Use of multiple sampling rates[10] :* In this thesis it is assumed that data estimates are available at the end of every frame. The channel is assumed to be a constant for the duration of this frame. The Kalman filter based estimator provides current estimates after processing the data based estimate and hence at the end of each frame received.

  We can increase the usefulness of the method presented in this thesis by running the Kalman filter at a higher rate than the frame rate. In the intervals that no data estimate has arrived, we can perform only the time-update portion of the Kalman filter. When data is received, we perform the measurement update portion of the Kalman algorithm. Estimates can then be made available on as fine a division of the time line as we desire. The second advantage is that the data arrival times need not be uniform. For a-periodically available data we merely perform time updates until a data estimate is received.

- *Different Models on each Path:* In this thesis we have assumed that on each path, the underlying process model is the same. In case the model varies from path to path, the Kalman filter can still be used but with a few modifications. The system matrix will be as follows:

$$A = \begin{bmatrix} \phi_{11} & \phi_{21} & & & \phi_{N1} \\ \phi_{12} & \phi_{22} & \phi_{32} & . & \phi_{N2} \\ . & . & . & . & . \\ . & . & . & & . \\ \phi_{1L} & . & . & . & \phi_{NL} \end{bmatrix}$$

  with the system equation suitably modified.

- *Correlated paths:* In this work, we have assumed that the multipaths are not correlated. Correlated multipaths can still be tracked using this Kalman algorithm but further work needs to be done to modify the Kalman filter to track correlated paths. A

good starting point is reference [11] where a Cholesky decomposition is used to generate correlated multipath waves.

- *Actual Implementation:* This algorithm is very well suited for implementation on a live system. The discrete Kalman filter is well documented as a robust algorithm. It will be very interesting to compare the theoretical and actual performance of the algorithm.

6 References

1. T.S. Rappaport, *Wireless Communications Principles and Practice.*. Publisher: Prentice Hall press. 1996.
2. William C. Jakes, *Microwave mobile communications.*, Publisher: IEEE Press, 1993.
3. Michel C. Jeruchim, Philip Balaban, K. Sam Shanmugan,*Simulation Of Communication Systems.* Publisher: Plenum, 1992.
4. Simulation Of Communication Systems Class Notes. Author: K. Sam Shanmugan, University of Kansas.
5. K. Sam Shanmugan, Arthur M. Breipohl, *Random signals : detection, estimation, and data analysis.* Publisher: Wiley, 1988.
6. Mohinder S. Grewal and Angus P. Andrews, *Kalman filtering : theory and practice.* Publisher: Prentice-Hall, 1993.
7. James D. Hamilton., *Time series analysis.* Publisher: Princeton University Press, 1994.
8. George E.P. Box, Gwilym, M. Jenkins, Gregory C. Reinsel, *Time series analysis: forecasting and control.* Publisher: Prentice Hall, 1994.
9. C.A. Montemayor, Paul Fikkema, "Near Optimum Iterative Estimation of Dispersive Multipath Channels". *VTC'98.*
10. Frank L. Lewis, *Optimal estimation: with an introduction to stochastic control theory.* Publisher: Wiley, c1986.
11. T. Hattori, K. Hirade, "Generation Method Of Mutually Correlated Multipath Fading waves," *Electronics and Communications in Japan,* Vol. 59-B, pp.69-76, 1976.