The improvements are more marked at low values of γ because the lower network loads allow the faded packets that

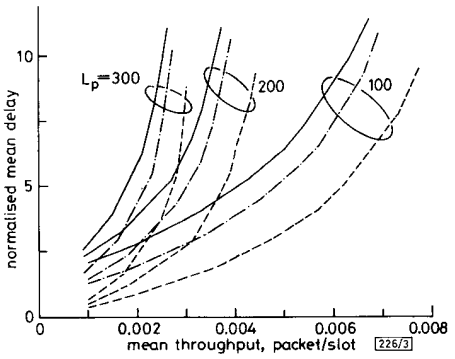

Fig. 3 *Throughput-delay performance with shorter mean backoff length*
——— fading, common backoff
—·—· fading packets nonpersistent, shorter mean backoff
– – – – nonfading

retransmit earlier greater probabilities of contending successfully for the channel. Clearly, this will not be the case at higher values of γ (network loads), and because faded packets that retransmit unsuccessfully back off according to the original algorithm, performance tends towards that of the common backoff algorithm, as Fig. 3 illustrates.

One-persistent backoff: Intuitively, the backoff delay of packets that fail by channel errors can be further optimised if the channel is sensed persistently after aborting transmission for the end of channel errors after which retransmission is initiated. For a fading channel, all users must thus be capable of end-of-fade detection, which in practice may be implemented through some form of receiver feedback. Fig. 4 compares the numerical throughput-delay performance with that of the common backoff algorithm.[2] Note that as with the short

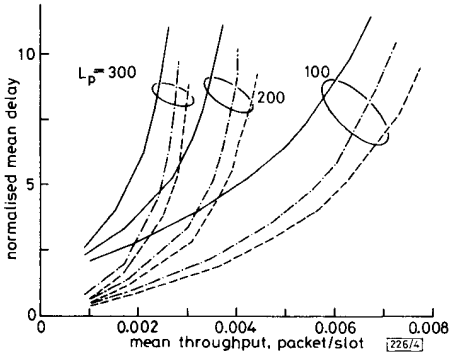

Fig. 4 *Throughput-delay performance with one-persistent backoff*
——— fading, common backoff
—·—· faded packets persistent backoff
– – – – nonfading

mean backoff length algorithm above, performance improvements are more significant at low values of γ and less considerable as γ increases. The reasons are similar to those discussed earlier for the first modified backoff algorithm. However, note that this second modification generally results in better network performance than the first. This can be attributed to the greater optimisation of the backoff delay as is intuitively expected.

K. C. CHUA
K. M. LYE
Department of Electrical Engineering
National University of Singapore
10 Kent Ridge Crescent
Singapore 0511, Singapore

18th February 1991

References

1 CHUA, K. C., and LYE, K. M.: 'Performance of multichannel CSMA/CD network with channels susceptible to random noise', *Electron. Lett.*, 1991, **27**, pp. 52–53
2 CHUA, K. C., and LYE, K. M.: 'Performance of multichannel CSMA/CD network with time-varying channels'. Tech. Rep. No. 1-2-91, Commun. Lab., Dept. Elect. Eng., National Univ. of Singapore, 1991
3 HO, K. K. Y., RAO, R. R., and WOLF, J. K.: 'Random-access systems with a time-varying channel', *IEEE Trans.*, 1990, **COM-38**, pp. 1293–1297
4 LYE, K. M., and HU, T. H.: 'Markov model for the digital Rayleigh fading channel'. Tech. Rep. No. 1-10-90, Commun. Lab., Dept. Elect. Eng., National Univ. of Singapore, 1990

EFFICIENT APPROXIMATION OF CONTINUOUS WAVELET TRANSFORMS

Indexing terms: Transforms, Algorithms

An efficient algorithm is developed for computing the continuous wavelet transform or wideband ambiguity function on a grid whose samples are spaced uniformly in time but placed arbitrarily in scale. The method is based on the chirp z transform and requires the same order of computation as constant-bandwidth analysis techniques, such as the short-time Fourier transform and narrowband ambiguity function.

Introduction: Signal representations based on the wavelet transform and the closely related wideband ambiguity function appear promising in a number of applications.[1] Whereas very efficient algorithms exist for computing expansions of a signal onto a discrete orthogonal wavelet basis, the sampling density in both time and scale of such a representation is insufficient for a number of applications. Examples include analysing Doppler-shifted sonar signals, which requires fine resolution over a small range in the scale parameter, and generating a constant-Q time-frequency representation.

We propose an efficient algorithm for computing the continuous wavelet transform (CWT) on a grid of samples in time and scale, with equally spaced time samples and arbitrary sampling of the scale parameter. Complete freedom in the scale parameter makes the technique flexible enough to support the needs of most applications. The algorithm employs the chirp z-transform method[2] to achieve a computational complexity of $O[N_a L \log L]$ to compute $N_a L$ transform samples, where N_a is the number of arbitrary scale samples, and L is the effective number of samples of the signal and the normalised wavelet. The method implicitly timescales the normalised wavelet by an inverse scaling in the frequency domain and efficiently computes a constant-scale slice of the distribution using fast convolution via the FFT. This algorithm has the same order of complexity as the short-time Fourier transform and other time-frequency representations.

Continuous wavelet transform: The CWT of a signal $s(t)$ is the two-dimensional time-scale representation given by

$$W(t, a) = \frac{1}{\sqrt{(|a|)}} \int_{-\infty}^{\infty} s(u) h^*\left(\frac{u-t}{a}\right) du \quad (1)$$

where $h(t)$ is known as the normalised analysing wavelet and a is the scale parameter. Equivalently, the CWT can be expressed in terms of the Fourier transforms of the signal and wavelet

$$W(t, a) = \frac{\sqrt{(|a|)}}{2\pi} \int_{-\infty}^{\infty} S(\Omega) H^*(a\Omega) e^{j\Omega t} d\Omega \quad (2)$$

Typically, the wavelet is chosen to be a bandpass function, hence each constant-scale slice of the CWT can be interpreted

as the output of a bandpass filter $H^*(a\Omega)$ driven by the signal to be analysed. Changing the scale parameter varies the centre frequency and bandwidth of the filter in a constant-Q fashion, providing better time resolution at high frequencies.

In principle, the CWT can be computed by first finding the Fourier transforms of the signal and the normalised wavelet. The inverse Fourier transform of the product of $S(\Omega)$ and the scaled wavelet $H^*(a\Omega)$ yields one constant-scale slice of the transform.

In practice, we can proceed similarly, but must deal with sampled versions of the signal and wavelet. Assuming that $s(t)$ and $h(t)$ have been sampled at or above the Nyquist rate f_s, denote the sampled signal and wavelet by $s_d(n) = s(nT)$, $h_d(n) = h(nT)$, and denote their discrete-time Fourier transforms by $S_d(\lambda)$, $H_d(\lambda)$. To satisfy the Nyquist criterion, the sampled signal and wavelet must be bandlimited and, therefore, in general, cannot be represented by a finite number of samples. We assume that enough samples are available so that the discrete signals contain essentially all of the energy in $s(t)$ and $h(t)$. Let the number of samples of the signal and wavelet by N_s and N_h, respectively. The scale parameter is discretised to a set of values a_i, $i = 1, \ldots, N_a$; the value of each a_i is arbitrary.

Discretising eqn. 2 on a grid whose samples are uniformly spaced in time and arbitrarily spaced in scale yields

$$W(nT, a_i) \approx W_d(n, a_i) = c\sqrt{(|a_i|)} \times \sum_{k=-(M/2)}^{(M/2)-1} S_d\left(\frac{2\pi}{M}k\right) H_d^*\left(\frac{2\pi}{M}a_i k\right) e^{j(2\pi/M)kn} \quad (3)$$

where c is a constant and $M \geq N_s + N_h \max_i |a_i| - 1$ is chosen to implement the linear convolution indicated in eqn. 1 rather than a circular convolution. (Usually, M is chosen to be a power of 2.) Given $S_d[(2\pi/M)k]$ and $H_d^*[(2\pi/M)a_i k]$, eqn. 3 can be efficiently computed, because it takes the form of an M-point inverse discrete Fourier transform, which can be implemented using an FFT. Likewise, $S_d[(2\pi/M)k]$ is simply the M-point FFT of the zero-padded signal.

The computation of the scaled spectrum, $H_d[(2\pi/M)a_i k]$, is more difficult, and lies at the heart of the fast algorithm. $H_d[(2\pi/M)a_i k]$, $-M/2 \leq k \leq M/2 - 1$, consists of samples of the discrete-time Fourier transform of the wavelet, $H_d(\lambda)$, in the interval $|\lambda| \leq a_i \pi$. The intersample spacing is $(2\pi/M)a_i$. When a_i is a rational number, these samples could be computed via an FFT, but, in general, the cost would be prohibitive. For example, computation of M samples with $a_i = 0.99$ requires a $100M$-point FFT. Furthermore, if a_i is not rational, then FFT-based computation is impossible. We now show that the samples $H_d[(2\pi/M)a_i k]$ can be efficiently computed using the chirp z-transform algorithm. Using the definition of the discrete Fourier transform and setting $p = k + M/2$, we have

$$H_d\left(\frac{2\pi}{M}a_i k\right) = H_d\left(\frac{2\pi}{M}a_i p - \pi a_i\right) = \sum_{n=0}^{M-1} h_d(n) e^{-j(2\pi/M)a_i pn} e^{j\pi a_i n}$$

$$p = 0, \ldots, M-1 \quad (4)$$

which is recognised as the chirp z-transform of the sequence $h_d(n)$ with parameters $W = e^{-j(2\pi/M)a_i}$ and $A = e^{-j\pi a_i}$. The cost of computing the chirp z transform is only a few times greater than an M-point FFT. Further details regarding the implementation of the transform are given in Reference 2. Note that the case $a_i < 1$ requires samples of $H_d(\lambda)$ outside the interval $|\lambda| \leq \pi$. Because these samples correspond to analogue frequencies above the Nyquist frequency and the wavelet is assumed bandlimited, we set $H_d[(2\pi/M)a_i k] = 0$ for $|k| > M/2a_i$.

Now that we have an efficient means for computing samples of the discrete-time Fourier transform of the wavelet, it is straightforward to formulate a fast algorithm for computing samples of the CWT. First, we compute the M-point FFT of the signal $s_d(n)$. Then, for each desired constant-scale slice, compute $H_d[(2\pi/M)a_i k]$ from $h_d(n)$ using the chirp z-transform algorithm, and take the inverse FFT indicated in eqn. 3.

Computational complexity: For each scale sample a_i, the fast algorithm requires a chirp z transform to compute the scaled wavelet spectrum and an inverse FFT to compute one constant-scale slice of the wavelet transform. The inverse FFT is of length M. The chirp z transform requires a forward and inverse Fourier transform of length $L \geq N_h + M - 1$, along with $N_h + M$ pre- and postmultiplications. Including the computation of the spectral product $S_d[(2\pi/M)k]H_d^*[(2\pi/M)a_i k]$, assuming a complex-valued signal and wavelet and a computational cost of $5N \log_2 N$ floating point operations for a basic radix-2 FFT, the total number of floating point operations to compute the sampled continuous wavelet transform is

$$N_a(10L \log_2 L + 5M \log_2 M + 6N_h + 6L) + 5M \log_2 M \quad (5)$$

The above formula assumes precomputation of the weighting functions required by the chirp z transform; without precomputation, the computational burden is perhaps twice as great.

The $O[N_a L \log_2 L]$ complexity of this algorithm compares favourably with the $O[N_a N_s N_h \max_i |a_i|]$ complexity of an implementation using direct convolution, and is of the same order as fixed-scale representations such as the short-time Fourier transform, although the multiple zero-padded FFTs in the chirp z transform may make the fast continuous wavelet transform a few times more costly in some cases.

Fig. 1 displays a contour plot of a continuous wavelet transform, computed using the fast algorithm, of a signal consisting of three Gaussian components centred at different times and frequencies, with the wavelet equal to the rightmost component. Fig. 2 contains a contour plot of a short-time Fourier transform computed using a baseband equivalent of the wavelet as a window function. The variation in time and frequency resolution with frequency is clear in the wavelet transform in Fig. 1; the choice between a wavelet transform

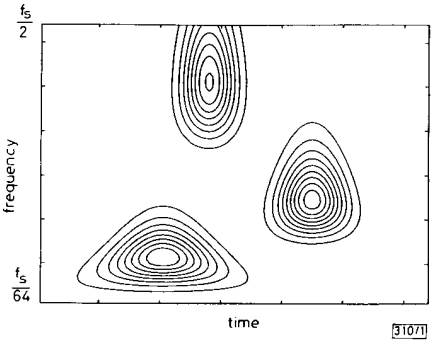

Fig. 1 *Wavelet transform magnitude of three-component Gaussian signal computed with bandpass Gaussian wavelet centred for $f_0 = f_s/6$*
Scale axis is linear in frequency $f = f_0/a$

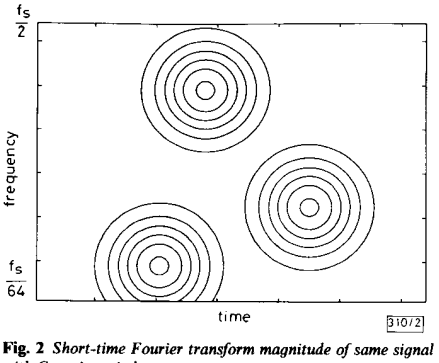

Fig. 2 *Short-time Fourier transform magnitude of same signal computed with Gaussian window*

and a short-time Fourier transform depends on the character of the particular class of signals to be analysed.

Acknowledgement: This work was supported by National Science Foundation, grant number MIP-9012747, and the Joint Services Electrical Program, grant number N00014-90-J-1270.

D. L. JONES

Coordinated Science Laboratory
University of Illinois
1101 W. Springfield Ave.
Urbana, IL 61801, USA

4th March 1991

R. G. BARANIUK

Electrical and Computer Engineering
252 Engineering Research Laboratory
University of Illinois
Urbana, IL 61801, USA

References

1 FLANDRIN, P.: 'Wavelets and related time-scale transforms'. *Advanced Signal Processing Algorithms, Architectures, and Implementations*, Luk, F. T. (Ed.), Proc. SPIE 1348, 1900, pp. 2–13
2 RABINER, L. R., SCHAFER, R. W., and RADER, C. M.: 'The chirp z-transform algorithm', *IEEE Trans.* 1969, **AU-17**, (2), pp. 86–92

CHAOS IN PHASE-LOCKED LOOPS

Indexing terms: Chaos, Phase-locked loops

The chaotic behaviour of phase-locked loops (PLL) in tracking a received signal whose frequency is varying linearly with time is studied. This system is described by a third-order nonlinear autonomous equation. The Lyapunov exponents and dimension of the system are calculated to confirm the chaotic phenomenon. Numerical results indicate that the parameter ranges where the chaos could occur are realistic in the typical designs.

Introduction: Phase-locked loops (PLL) have been widely used in the last three decades, e.g. in space-vehicle-to-earth data links, AM and FM demodulations, FSK decoders, motor speed controls, and frequency synthesisers. A basic development was presented by Viterbi.[1] Because there is a sinusoidal nonlinearity in the PLL, the phenomenon of chaos is believed to exist in such a system.[2,3] The limitation of the second-order PLL is that the loop will not lock if the received signal frequency exceeds an upper bound; that is, if the input frequency is bigger than an upper bound, lock on is impossible on the basis of the singular-point behaviour. We have considered the chaotic behaviour of third-order PLL via perturbation analysis for phase-modulated signals previously.[4] The objective of this Letter is to provide a numerical study of the third-order PLL when the frequency of the received signal is varying linearly with time. As expected, once the circuit parameters are within a certain range, a chaotic phenomenon can be observed. Moreover, the Lyapunov exponents and Lyapunov dimension of the system are also calculated to confirm the results.

State space models of PLL: The block diagram of the PLL is shown in Fig. 1. The basic operation of the PLL is to feed back the voltage-controlled oscillator (VCO) frequency f_0 to the phase detector (PD) and lock on the input signal f_i. At this point, the loop is said to be synchronised, or phase-locked.

PLL can be modelled equivalently as shown in Fig. 2.[1] We see that the PD is replaced by a subtracter and sinusoidal nonlinearity. The VCO is replaced by an integrator. In Fig. 2, $\phi(t)$ is the phase difference between input and VCO.

The PLL operation is governed by the differential equations of phase error $\phi(t)$ and its time derivative. To track the

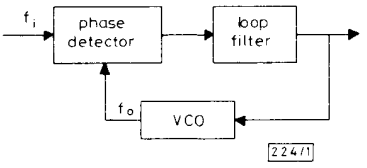

Fig. 1 *Phase-locked loop block diagram*

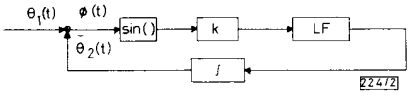

Fig. 2 *Model of PLL*

frequency-variable signals and eliminate the steady state error of the system, the filter transfer function is chosen as follows:[1]

$$H(s) = 1 + \frac{a}{s} + \frac{b}{s^2} \tag{1}$$

Thus, we have obtained a third-order nonlinear ordinary differential equation that contains a number of independent parameters.

$$\frac{d^3\phi(t)}{dt^3} + k\frac{d^2 \sin\phi(t)}{dt^2} + ka\frac{d \sin\phi(t)}{dt} + kb \sin\phi(t) = \frac{d^3\theta_1(t)}{dt^3} \tag{2}$$

If the received signal frequency is varying linearly with time, as is the case when a signal is transmitted from a constant frequency oscillator aboard a vehicle which has a constant radial acceleration relative to the receiver, then

$$\theta_1(t) = \tfrac{1}{2}Rt^2 + (\omega - \omega_0)t + \theta_0$$

where R is the constant rate of change of frequency, ω_0 the free-running angular frequency, ω the received constant frequency signal with an initial phase θ_0. Hence

$$\frac{d^3\phi(t)}{dt^3} + k\cos\phi\frac{d^2\phi(t)}{dt^2} + (ka\cos\phi)\frac{d\phi(t)}{dt} - k\sin\phi\left[\frac{d\phi(t)}{dt}\right]^2 + kb\sin\phi = 0 \tag{3}$$

Eqn. 3 is a nonlinear third-order autonomous equation. Let us define the state variables as follows:

$$x(t) \equiv \phi(t)$$
$$y(t) \equiv \dot{\phi}(t) + k\sin\phi$$
$$z(t) \equiv -\ddot{\phi}(t) - ka\sin\phi$$

then the state space is given by

$$\dot{x}(t) = y(t) - k\sin x(t)$$
$$\dot{y}(t) = -ka\sin x(t) - z(t)$$
$$\dot{z}(t) = kb\sin x(t) \tag{4}$$

In the following Section, the computer simulation of eqn. 4 and its Lyapunov exponents and dimension will be calculated.

Simulation results: Consider the PLL eqn. 4 with system parameters $k = 0{\cdot}01$, $a = 100$ and $b = 1$. Fig. 3 indicates a time series solution $x(t)$ against t. Figs. 4 and 5 are the observed data projected onto the (y, z) and (x, y) planes, respectively.

To confirm that the PLL is indeed chaotic, we compute three Lyapunov exponents for eqn. 4. The method of calcu-