QAM CARRIER TRACKING FOR SOFTWARE DEFINED RADIO

Mr James D. Schreuder (Schreuder Engineering, Sydney, NSW, Australia;
james.schreuder@schreuder.com.au)

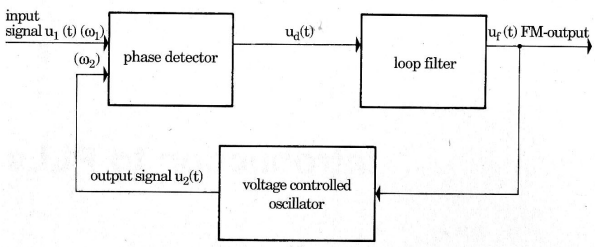
ABSTRACT

This paper investigates Quadrature Amplitude Modulation (QAM) carrier tracking techniques for Software Defined Radio implementations. These techniques were developed during a recent investigation project into the baseband signal processing requirements of a draft TIA Public Safety Radio standard proposal *Scalable Adaptive Modulation* (SAM) [1]. The proposal specified use of frequency multiplexed QAM channels incorporating Pilot Symbol Assisted Modulation (PSAM) [2]. This proposal has since been further updated and now forms the basis of the current P34 standard.

The investigation was performed using MATLAB based simulation code to implement baseband radio synchronization algorithms. The algorithms included: 4/16/64 QAM carrier and symbol clock tracking, digital AGC, and channel gain estimation techniques. This paper will focus on explaining how a reliable QAM carrier tracking technique can be developed from the basic building blocks of Phase Locked Loops (PLLs) implemented in software as investigated for SAM.

1. INTRODUCTION

A commonly used method of tracking RF signal phase is the Phase Locked Loop (PLL). In an analog circuit implementation, a PLL is comprised of an incoming sinusoidal signal which is multiplied by the sinusoidal output of a Voltage Controlled Oscillator (VCO). The VCO input sets the phase offset of the sinusoidal output and is controlled by the DC output of the loop filter. An example block diagram of a PLL circuit (from [3]) is shown below:



The loop filter is designed such that its DC output level is adjusted proportionately to the phase offset between the input sinusoid and the VCO output sinusoid. Assuming the feedback loop is stable, the input to the VCO will adjust the phase of the output sinusoid until its phase offset matches that of the input sinusoid.

2. IMPLEMENTING A DIGITAL PLL IN SOFTWARE

In software, a PLL can be implemented using a software loop to process each signal sample. This implementation replaces the analog loop filter with a digital version and the VCO with a synthesized VCO constructed from a simple sinusoid function (e.g. $\cos(2\pi f_c t[i] + \theta)$, where t[i] is an array of time values and θ a phase value).

However the continuous adjustment capability of the analog loop filter and VCO must be replicated in a software loop such that after the input of each sample, the (delayed) output level of the digital loop filter can be used to adapt the digitally synthesized VCO phase such that it equals the phase offset of the input signal (i.e. phase lock). This adaptation should occur with each loop iteration and minimize the difference between the phase offset of the input signal and digital VCO output.

This is an example of an optimization problem that can be solved using Adaptive Parameter Estimation [4]. This method uses an iterative algorithm to estimate an unknown parameter value x by descending (or ascending) the gradient of a performance function, denoted $J(x)$. The derivative of the function, $dJ(x)/dx$, is formulated to have a zero value when the error between x and the current estimate of $x[k]$ is minimized. This process can be defined by the equation:

$$x[k+1] = x[k] - \mu \frac{dJ(x)}{dx}$$

Where $x[k]$ is an estimate of x at time k, $x[k+1]$ is the estimate of x at time $k+1$, μ is a small positive number referred to as the algorithm step size.

In the PLL's case, $J(x)$ needs to be maximized when the phases of the input signal and synthesized VCO output are

equal which provides the highest DC output value from the digital loop filter. $J(x)$ is defined by [4] as:

$$J_{PLL}(\phi) = LPF\{r[kT_s].\cos(2\pi.f_0 kT_s + \phi[k])\}$$

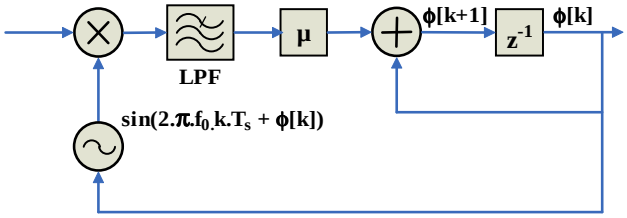
Assuming a small μ, the derivative of $J_{PLL}(\phi)$ with respect to ϕ at time k can be approximated as:

$$\frac{dJ_{PLL}(\phi)}{d\phi} = LPF\{-r[kT_s]\sin(2\pi f_0 kT_s + \phi[k])\}$$

The corresponding adaptive update equation is then:

$$\phi[k+1] = \phi[k] - \mu LPF\{r[k.T_s]\sin(2\pi f_0 kT_s + \phi[k])\}$$

The figure below shows the digital PLL containing a digital VCO, a signal multiplication step, a digital loop filter and an adaptive element update:

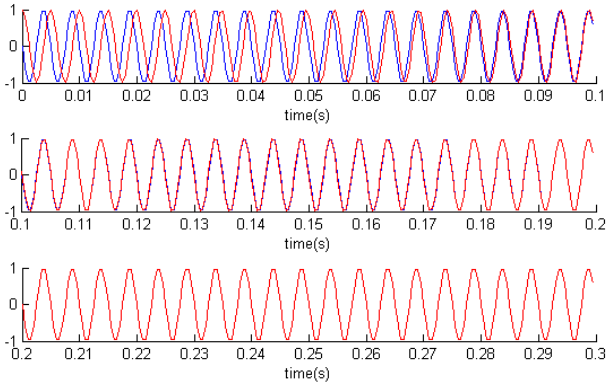


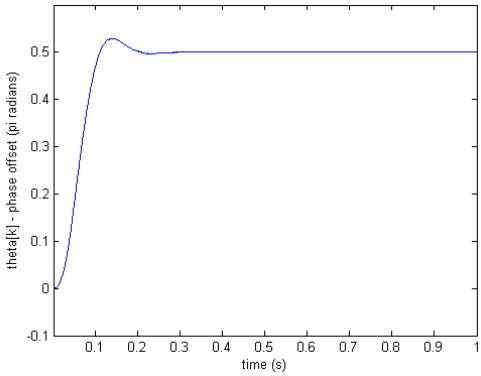
The MATLAB code used to implement this PLL is as follows:

```
Ts=1/2000; time=1; t=0:Ts:time; % time vector
f0=200; phoff=pi/2;             % carrier freq. and phase
fc=200;                         % assumed freq. at receiver
rp=cos(2*pi*fc*t+phoff);        % simplified received signal
carrier = rp;

fl=100; ff=[0 .01 .02 1]; fa=[1 1 0 0];
h=firpm(fl,ff,fa);              % LPF design
mu=.01;                         % algorithm stepsize

theta=zeros(1,length(t));
theta(1)=0;                      % initialize vector for estimates
z=zeros(1,fl+1);                % initialize buffer for LPF
for k=1:length(t)-1              % z contains past fl+1 inputs
  VCO(k) = sin(2*pi*f0*t(k)+theta(k));
  z=[z(2:fl+1), rp(k)*VCO(k)];
  update=fliplr(h)*z';          % new output of LPF
  theta(k+1)=theta(k)-mu*update; % algorithm update
end
```

The plots below show the software PLL synchronizing with input 200Hz cosine signal with phase offset of π/2 radians. The VCO initially has a zero phase offset. In the first plot, the input signal is shown in red and the VCO signal in blue. Iteration of the loop adjusts the phase of the VCO to match the phase of the input signal. At approximately 0.2 seconds, the two signals are aligned and the PLL is phase locked.

The second plot shows the VCO phase offset as it adapts to the input signal's phase offset. The plot confirms that phase lock is achieved at approximately 0.2 seconds and the phase locked value is the expected value of π/2 radians. The plot also shows that the algorithm initially overshoots the correct phase estimate and peaks at 0.52 radians at approximately 0.14 seconds. This overshoot can be minimized by reducing the step size, μ. However a trade-off exists in that smaller μ values increase the number of loop iterations required.

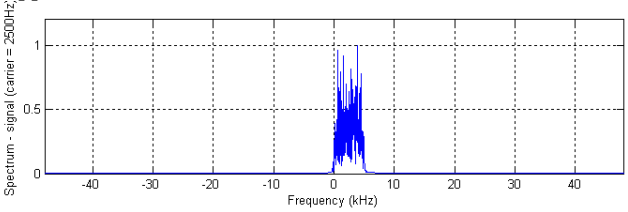




3. QAM PHASE RECOVERY USING PLLS

A common method used to track a digital radio carrier is to extract a sinusoidal carrier signal from the received signal and then track the signal's carrier phase using a PLL. This method is recommended for digital modulations such as Pulse Amplitude Modulation (PAM) where a narrowband filter can extract a replica of the carrier signal due to its large frequency component magnitude.

However, QAM modulation represents a suppressed carrier modulation scheme. The QAM spectrum has no distinguishable carrier frequency component for a PLL to track. For example the plot below shows the spectrum of a 4-QAM (QPSK) signal as used by SAM (note that the message signal has been Raised Cosine filtered) with a suppressed carrier at 2500Hz.



Despite this, [5] describes a method for QAM phase recovery by tracking the carrier of the signal, pre-manipulated with a 4th-power operation, followed by a narrowband filter centered on the quadrupled carrier frequency component. The performance function suitable for a 4th power PLL is then determined to be:

$$J_{4PLL} = \frac{1}{4}\cos(4(\theta - \phi))$$

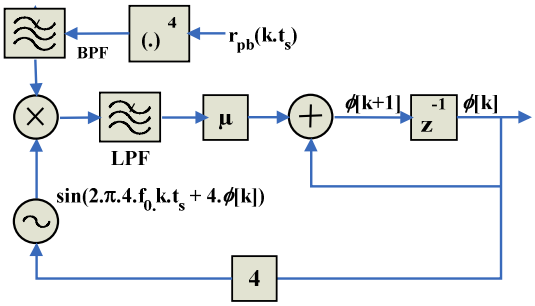
where θ is the phase of the received signal and ϕ is the phase of the VCO in the PLL. This will be maximized when $\theta = \phi$. Using a gradient descent strategy, the update equation is then defined as:

$$\phi[k+1] = \phi[k] + \mu \frac{dJ_{4PLL}}{d\phi}$$

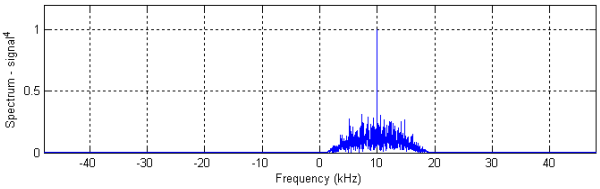
giving an update equation of:

$$\phi[k+1] = \phi[k] + \mu \sin(4(\theta[k] - \phi[k]))$$

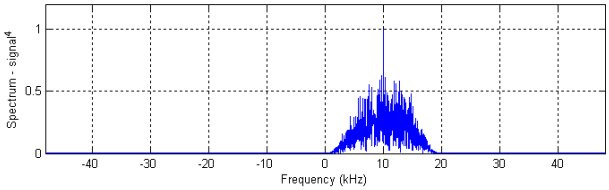
A suitable PLL design is shown in the figure below:

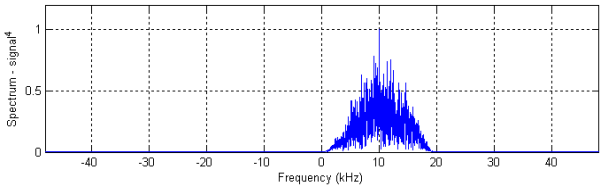


The plot below shows the spectrum for a fourth power 4-QAM signal. A strong frequency component can be seen to exist at 4 times the carrier frequency (4 * 2500Hz = 10,000Hz).



Unfortunately the technique is not as successful at extracting higher order M-QAM based carrier signals. The plots below show the fourth-power signal spectrum for 16-QAM and 64-QAM. The magnitude of the carrier component is shown to diminish relative to the message frequency components as the QAM order increases.





Both [6] and [7] state that the method works for 4-QAM, but [6] states that it only works "passably" well for 16-QAM due to the difficulty in removing the non carrier frequency components introduced. And [7] states that 16-QAM constellation "pattern jitter" will be apparent but that the "outer points dominate" such that the "fourth-power signal has a component at $4f_c$ that is usable if a very narrow band PLL is used to refine it". However, a narrow band filter will require a high filter order that will lead to increased DSP computation cost and should be considered in any SDR design.

4. QAM PHASE RECOVERY USING DECISION-DIRECTED CARRIER TRACKING

The final QAM carrier tracking method investigated takes a different approach to carrier tracking. The method generates a phase error signal by exploiting the phase and amplitude difference between each received symbol value and the nearest ideal QAM constellation symbol value. This method is called Decision-Directed Carrier Tracking (DDCT) because the decisions (the choice of the nearest allowable symbol in the constellation) direct the adaptation of the receiver's carrier phase to match the transmitter's phase by minimizing the mean-square of an error function. This error function (proposed in [8]) is defined as the phase difference between each received baseband symbol $r_{bb}[kT_s]$ and its nearest ideal constellation point c_{iq} on the complex plane:

$$e[kT_s] = c_{iq} - r_{bb}[kT_s]$$

Therefore the performance function to be optimized in this algorithm is the mean squared error (MSE) of the baseband symbol demodulation error:

$$J_{MSE} = E\{|e[kT_s]|^2\}$$

The adaptive update algorithm can then be defined as:

$$\phi[k+1] = \phi[k] + \frac{\mu \partial J_{MSE}}{\partial \phi}$$

The analysis in [8] does not directly derive the expression for J_{MSE}. Instead the following relationship between the demodulated symbol phase α_{bb} and nearest ideal constellation point α_c is determined:

$$\sin(\alpha_{bb} - \alpha_c) = \frac{\Im m\left[\overline{e[kT_s]} \cdot r_{bb}[kT_s]\right]}{|c_{iq}||r_{bb}|}$$

This has the same sign as the phase error between the ideal constellation point c_{iq} and the received demodulated baseband symbol $r_{bb}[kT_s]$, provided the phase error is not too

large. Since sin(ϕ) is approximately equal to ϕ for small ϕ, this term is also nearly a linear function of the error for small phase ϕ values.

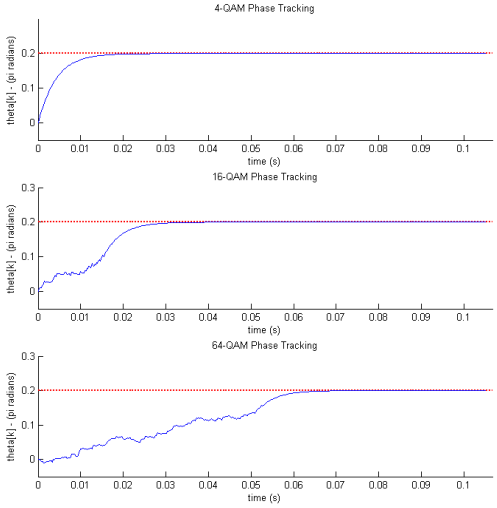
This phase error signal can be used in a phase-locked loop to iteratively adjust ϕ so that the demodulated baseband symbol values are aligned in phase angle with the ideal constellation points. Changing ϕ by some angle has the effect of rotating the baseband symbol values by the negative of the phase angle. This error signal derivation gives the final adaptive update equation as:

$$\phi[k+1] = \phi[k] + \mu \frac{\Im m\left[\overline{e[kT_s]} r_{bb}[kT_s]\right]}{\left|c_{iq}\right| \left|r_{bb}\right|}$$

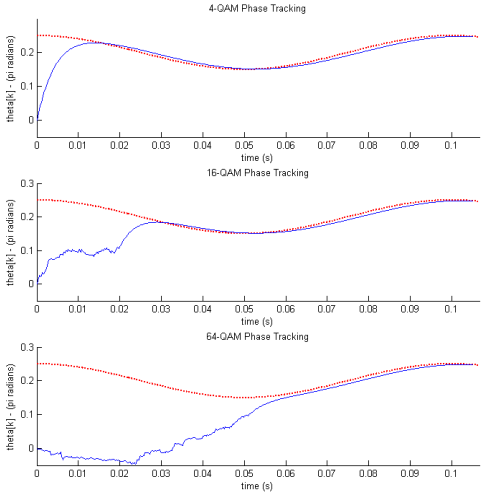
The MATLAB code used to implement this PLL is as follows:

```
CARRIER = 1000;
k = 1; mu = 0.1; M = 16; Ts = 1 / 4800;
phaseNow = 0; phaseEst = phaseNow; phaseInc = 2*pi*CARRIER * Ts;

for s = pbSymbols(1:end) % An array of passband QAM symbols
  % Demodulate the passband symbol and store in array
  bbSymbols[k] = s .* exp(-j * phaseNow);
  % Find the nearest QAM constellation point to symbol s
  decisionSymbol = qamMatch(s, M);

  % Calculate the phase error
  decisionError = decisionSymbol - s;
  % Calculate the new phase estimate
  theta[k] = phaseEst;
  phaseEst = phaseEst + mu * (imag(conj(decisionError)*s)
                              / (abs(decisionSymbol)*abs(s)));
  % Calculate the next demodulation phase value
  phaseNow = phaseNow + phaseInc + phaseEst;
  k = k + 1;
end
```

The following plots show the result of the above receiver PLL achieving carrier phase lock with a fixed transmitter carrier phase offset of 0.2π radians and frequency 1000Hz. Results for 4, 16, and 64-QAM signals are shown for the SAM symbol clock rate of 4800 baud. The red line shows the known phase offset introduced in the transmitter. The blue line shows the value of $\phi[k]$ as the PLL converges to the fixed transmitter phase. Each plot was generated using a μ value of 0.05.



In addition, the plots below show that the algorithm successfully tracks a varying relative carrier phase at the receiver:



5. QAM CARRIER FREQUENCY OFFSETS

Previously it was assumed that the carrier frequency used by the transmitter was exactly known by the receiver. In practice, RF and IF demodulators will normally exhibit small frequency tolerances leading to a frequency offset between transmitter and receiver. To understand the impact of this offset, a received passband signal r_{pb} can be written in terms of the original QAM symbol message stream m[k]:

$$r_{pb} = m[k] e^{j(2\pi f_t kT_s + \theta)}$$

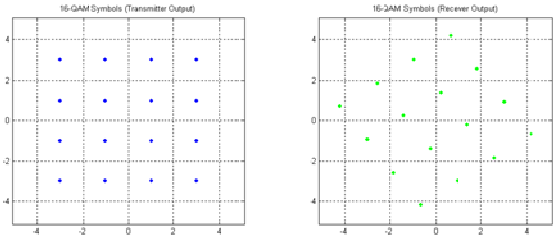
The transmitter carrier frequency is identified by f_t and phase offset θ. Following demodulation by the receiver, the received QAM symbol baseband values r_{bb} are then defined as:

$$\begin{aligned} r_{bb} &= m[k] e^{j(2\pi f_t kT_s + \theta)} \cdot e^{-j(2\pi f_r kT_s + \phi)} \\ &= m[k] e^{j[2\pi(f_t - f_r)kT_s + (\theta - \phi)]} \end{aligned}$$

where ϕ is the phase of the receiver. Now assuming that $f_r = f_t$, this equation then becomes:

$$r_{bb} = m[k] e^{j(\theta - \phi)}$$

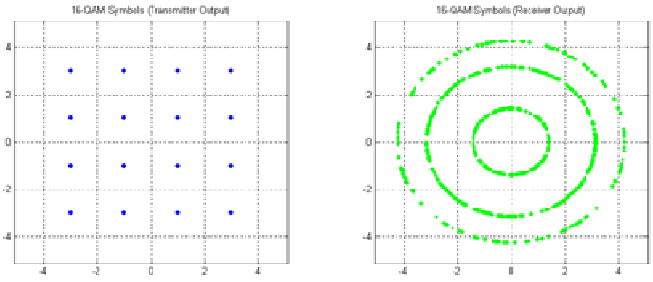
Therefore the demodulation step results in phase rotation of the demodulated signal equal to the phase difference between the transmitter phase θ and receiver VCO phase ϕ, (θ - ϕ). Demodulation of a 16-QAM signal with 0.2π radians phase offset then results in the following 16-QAM constellations:



The constellation plots show the receiver QAM symbols are rotated by 0.2π radians as predicted. Assuming that $f_r \neq f_t$ and that $(f_r - f_t)$ = 10Hz then the received baseband symbol stream can be defined as:

$$r_{bb}[k] = m[k]e^{j(2\pi \cdot 10 \cdot kT_s + 0.2\pi)}$$

The frequency offset will therefore introduce a phase rotation in the received QAM symbol constellation proportional to the size of the frequency offset (the symbol constellation is rotating with time). The plots below show the constellation results of demodulating a 16-QAM signal with 0.2π radians phase offset and 10Hz frequency offset:

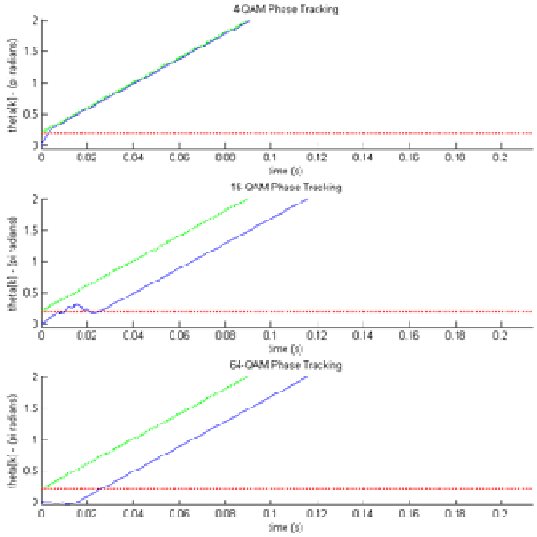


The received symbol constellation is spinning as predicted. And the phase rotation can be defined as:

$$\theta[k] = 2\pi(f_t - f_r)kT_s + (\theta - \phi)$$

This indicates that the input to the PLL VCO, $\phi[k]$, will exhibit a phase offset of $(\theta - \phi)$ and a gradient equal to the frequency difference $(f_r - f_t)$. This is confirmed by the following plots using the QAM based PLL with a frequency offset of 10 Hz and phase offset of 0.2π radians. The blue plot shows the $\phi[k]$ value generated by the PLL loop, the red plot the constant phase offset, and the green plot the predicted $\phi[k]$ value calculated from:

$$\phi[k] = 2\pi \cdot 10 \cdot kT_s + 0.2\pi$$



Interestingly, the 4-QAM plot converges exactly to the predicted plot, but the 16 and 64-QAM plots converge to the same gradient but with a different phase offset. This is due to the inherent "$\pi/2$ phase ambiguity" in the Decision-Directed algorithm. This ambiguity results from the fact that the receiver cannot distinguish between phase offsets that are multiples of $\pi/2$ due to the square symmetry of M-QAM constellations (a square constellation looks exactly the same when it is rotated by $\pi/2$ radians as it does rotated by π radians etc). It is recommended in [5] that this ambiguity can be resolved using one of the following methods:

- differentially encoding the message source so that the change in symbol value between each symbol is known
- letting a trained equalizer automatically add a rotational phase to achieve a match to the training symbols
- correlating the down-sampler output with a known/training signal

However, [9] also suggests "by insertion of known data symbols into the symbol stream". Since SAM was specified to use inserted Pilot and Synchronization symbols in the data stream (a PSAM scheme), the pilot and sync symbols were used to overcome the $\pi/2$ phase ambiguity in the project.

6. TRACKING FREQUENCY OFFSETS IN DECISION-DIRECTED QAM CARRIER TRACKING

When the current DDCT PLL is operating with a carrier frequency offset, the constellation ceases to rotate with time but remains fixed with an angular offset that is proportional to the frequency offset. However, since the frequency offset has been shown to be simply the gradient (derivative $\partial\phi/\partial Ts$) of the changing phase, the constellation angular offset can be corrected using an additional accumulator in the PLL (a 2nd-order loop).

In [8], the existing QAM PLL is extended to track the frequency offset. The QAM phase error signal that was proportional to sine of the angle between the demodulated symbol and the nearest ideal constellation point was derived to be:

$$\sin(\alpha_{bb} - \alpha_c) = \frac{\Im m\left[\overline{e[kT_s]} \cdot r_{bb}[kT_s]\right]}{\left|c_{iq}\right|\left|r_{bb}\right|}$$

Therefore, the phase error to be tracked on each PLL loop iteration is:

$$\Delta\phi = \frac{\Im m\left[\overline{e[kT_s]} \cdot r_{bb}[kT_s]\right]}{\left|c_{iq}\right|\left|r_{bb}\right|}$$

In order to track the additional phase change due to a carrier frequency offset, the following additional phase accumulation step is included in the PLL:

$$\psi[k+1] = \psi[k] + \mu_2 \Delta\phi[k]$$

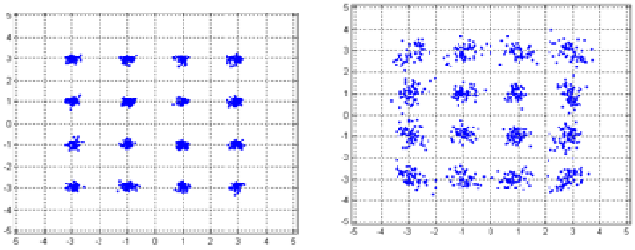
The final completed second-order adaptive update equation for DDCT is then:

$$\varphi[k+1] = \varphi[k] + 2\pi f_r T_s + \mu_1 \Delta\phi[k] + \psi[k]$$

The previous MATLAB code example can then be extended as follows:

```
CARRIER = 1000;
k = 1; mu = 0.1; M = 16; Ts = 1 / 4800;
phaseNow = 0; psi = 0; phi = 0; phaseInc = 2 * pi * CARRIER * Ts;

for s = pbSymbols(1:end) % An array of passband QAM symbols

  % Demodulate the passband symbol and store in array
  bbSymbols[k] = s .* exp(-j * phaseNow);

  % Find the nearest QAM constellation point to symbol s
  decisionSymbol = qamMatch(s, M);

  % Calculate the phase error
  decisionError = decisionSymbol - s;

  % Calculate the new phase estimate
  theta[k] = phi;
  phaseError = (imag(conj(decisionError)*s))
                / (abs(decisionSymbol)*abs(s));
  psi = psi + mu2 * phaseError;
  phi = mu1 * phaseError + psi;


  % Calculate the next demodulation phase value
  phaseNow = phaseNow + phaseInc + phi;
  k = k + 1;
end
```

The two plots below show the results of PLL code with a carrier frequency of 2700Hz carrier and frequency offset of 210Hz. The left plot shows the correctly demodulated constellation despite the large carrier frequency offset. The right plot shows the constellation of a demodulated 16-QAM signal with channel noise introduced (SNR = 20dB). The algorithm is effective even with noise introduced. This is because the Adaptive Parameter Estimation technique inherently provides a low-pass filtering (averaging) function on the PLL VCO output.



7. CONCLUSION

It has been shown that a simple PLL loop can be implemented in software using Adaptive Parameter Estimation. The estimation method requires the derivation of a performance function that the algorithm descends (or ascends) to lock onto the unknown phase of a received signal. This method can then be extended to track the phase and frequency of QAM signals using the appropriate performance function.

Decision Directed Carrier Tracking provides a reliable method for tracking carrier phase and frequency offsets for M-QAM symbols. It can be implemented with low computation cost as it only operates on the received passband M-QAM symbol values rather than on every signal sample. The method can also operate successfully in channel noise.

However the method has the disadvantages of the inherent $\pi/2$ phase ambiguity at the demodulator output and the requirement for the symbol clock timing to be known prior to demodulation. In the TIA SAM proposal, both disadvantages were resolved by the use of Synchronization and Pilot symbols inserted into the protocol's data stream.

8. REFERENCES

[1] TIA "Wideband Air Interface Scalable Adaptive Modulation (SAM) Physical Layer Specification", TIA-902-BAAB, 2003.

[2] J.M. TORRANCE, L. HANZO, "Comparative Study of Pilot Symbol Assisted Modem Schemes". Sixth International Conference on Radio Receivers and Associated Systems, pp 36 – 41, 26 – 27 September 1995.

[3] R. E. BEST, Phase-Locked Loops: Design, Simulation and Applications, McGraw-Hill. 2003

[4] R. JOHNSON, W. A. SETHARES, W. A., Telecommunication Breakdown: Concepts of Communication Transmitted by Software-Defined Radio, Pearson Prentice Hall, 2004.

[5] R. JOHNSON, A Digital Quadrature Amplitude Modulation (QAM) Radio, Pearson Prentice Hall, 2003

[6] J.B.ANDERSON, Digital Transmission Engineering, Prentice Hall, 1999.

[7] J. A. C. BINGHAM, The Theory and Practice of Modem Design, Wiley Press, 1988.

[8] S. A. TRETTER, Communication system design using DSP algorithms: with laboratory experiments for the TMS320C6701 and TMS320C6711, Kluwer Academic/Plenum Publishers, New York, 2002.

[9] L. E. FRANKS, Carrier and Bit Synchronization in Data Communication - A Tutorial Review. IEEE Transactions on Communications, COM-28, 1980