ASE 389P-7 Problem Set 6

Posting Date: November 7, 2024

You need not hand in anything. Instead, be prepared to answer any of these problems—or similar problems—on an upcoming take-home exam. You may discuss your solutions with classmates up until the time that the exam becomes available, but do not swap work (including code).

Readings

All background reading material is found on Canvas. You'll find an analysis of the best estimators for beat carrier phase θ and Doppler f_D in [1]. Reference [2] develops the probability distribution for a ratio of two correlated nomal random variables, which will be useful for Problem 1 below. You'll find details on the best possible estimation of code phase t_s in the theory presented in [3]. Gupta's 1975 paper is a good reference on PLLs [4].

Problems

1. In lecture we considered maximum-likelihood (ML) estimation of the carrier phase error $\Delta\theta(\tau_{j_k})$ from a single complex accumulation

$$S_k = \rho_k \exp[i\Delta\theta(\tau_{j_k})] + n_k = I_k + iQ_k$$

where $\rho_k = N_k \bar{A}_k/(2\sigma_{IQ})$ and $n_k = n_{Ik} + in_{Qk}$, with $n_{Ik}, n_{Qk} \sim \mathcal{N}(0,1)$ being zero-mean independent and white Gaussian sequences. For convenience, we simplified the notation as

$$S = \rho \exp[i\theta] + n = I + iQ$$

and found the ML estimate of θ to be

$$\hat{\theta}_{\mathrm{ML}} = \arctan(Q/I)$$

(a) Find the theoretical probability distribution of $\hat{\theta}_{\mathrm{ML}}$. Hints: First find the distribution of $W = Q/I$. This was solved by D. V. Hinkley back in 1969 in the paper "On the ratio of two correlated normal random variables," posted on Canvas [see Eqs. (1) and (2) in the paper]. Call $p_W(w)$ the probability distribution function for W. Note that in our case the correlation coefficient between Q and I is zero. Then recognize that $\hat{\theta}_{\mathrm{ML}} = \arctan(W)$ and use the standard technique for finding the distribution of a function of a random variable. You may express the distribution of $\hat{\theta}_{\mathrm{ML}}$ in terms of $p_W(w)$. Assume $\theta \in [-\pi/2, \pi/2]$.

(b) For $\theta = 0$, simulate 1 million pairs of I and Q with $\rho = 4$ and estimate θ by

$$\hat{\theta}_{\mathrm{ML}} = \arctan(Q/I)$$

Compare a histogram of the one million values of $\hat{\theta}_{\mathrm{ML}}$ against the theoretical distribution (plot them both on the same figure).

(c) The Cramer-Rao Lower Bound (CRLB) is the minimum variance that any estimate of the unknown parameter θ could attain: $\sigma_\theta^2 = \mathrm{var}(\hat{\theta}) \geq \mathrm{CRLB}$. One finds the CRLB by first finding the Fisher Information Matrix J (which is just a scalar in this case):

$$J = E[H_\theta^2], \quad \text{where} \quad H_\theta = \frac{d}{d\theta} \log p(\boldsymbol{z}|\theta)$$

Here, log refers to the natural logarithm. The likelihood function for this problem, $p(\boldsymbol{z}|\theta)$, was introduced in lecture. Show that CRLB $= 1/\rho^2$, as claimed in lecture. Does variance of your one million simulated points from part (b) obey this bound?

2. Consider an extension to Problem 1 in which the kth complex accumulation is

$$S_k = \rho \exp[i(2\pi f k T_a + \theta)] + n_k$$

Suppose we don't know ρ, f, or θ and we wish to estimate these from the sequence $S_0, S_1, ..., S_{N-1}$ by maximum likelihood estimation. Let $\boldsymbol{\alpha} = [f, \rho, \theta]^T$ hold the unknown parameters.

(a) Find the 3-by-3 Fisher Information Matrix corresponding to $\boldsymbol{\alpha}$. Hint: You'll find a general solution (where k may not start at 0) in Rife's 1974 paper "Single-tone parameter estimation from discrete-time observations," posted on Canvas.

(b) Find the ML estimator for $\boldsymbol{\alpha}$. Hint: this is also found in the 1974 Rife paper.

(c) Generate a simulated time history S_k, $k = 0, 1, ..., N-1$, for $N = 100$ and some $\boldsymbol{\alpha}$ that you choose. Apply your ML estimator to this sequence to estimate $\boldsymbol{\alpha}$. Be prepared to estimate $\boldsymbol{\alpha}$ for a "mystery sequence" in an upcoming exam.

3. In lecture, we discussed the precision limits of estimating the code start time, t_s. This problem will walk through two batch techniques for estimating t_s and will prompt you to compare the the precision of each estimate against the CRLB.

Recall the simplified baseband models for the spreading code $x(j)$ and the local replica $l(j)$:

$$\begin{aligned} x(j) &= AC(\tau_j - t_s) + n_j \\ l(j) &= C(\tau_j - \hat{t}_s) \end{aligned}$$

Recall that $T = \tau_{j+1} - \tau_j$ is the sampling interval. Write a MATLAB script to do the following:

(a) Generate 5 ms worth of $x(j)$ and $l(j)$ for code ID (PRN) 1 with sampling rate $f_s = (40e6)/7\,\mathrm{Hz}$ and chip period $T_c = 1/(1.023\mathrm{e}6)\,\mathrm{s}$. Generate n_j as complex Gaussian noise with unit variance; i.e., let $n_j = a_j + ib_j$, where the quadrature noise components are zero-mean, white, independent, and have variance $\sigma_n^2 = \frac{1}{2}$. Make the true delay $t_s = 1\mathrm{e}{-3}\,\mathrm{s}$ and the estimate $\hat{t}_s = t_s + \Delta t_s$, with $\Delta t_s = 0.25\mathrm{e}{-6}\,\mathrm{s}$ being the estimation error. Use a carrier-to-noise ratio of $\frac{C}{N_0} = 45$ dB-Hz and $\sigma_n^2 = \frac{1}{2}$, and relate these to the amplitude by $A = \sqrt{4\sigma_n^2 T \frac{C}{N_0}}$ (with $\frac{C}{N_0}$ expressed in linear units).

(b) Compute $\tilde{R}_{xl}(m) = \frac{1}{N_k} \sum_{j=j_k}^{j_k+N_k-1} x(j) l(j-m)$ for $m \in \{-10, -9, \ldots 9, 10\}$, $j_k = 1024$, and N_k equal to the number of samples in 4 code periods.

(c) Nearest-Sample Estimator: Find

$$m_{\text{ns}} = \underset{m}{\text{argmax}}\, |\tilde{R}_{xl}(m)|$$

Compute $\Delta t_{\text{ns}} = m_{\text{ns}}T$ and record the error $\Delta t_{\text{ns}} - \Delta t_s$.

(d) Least-Squares Estimator: Solve for the least-squares best fit of the correlation function as follows. Let

$$R_c(m, \Delta t) = 1 - |mT - m_{\text{ns}}T - \Delta t| / T_c$$

and normalize $\tilde{R}_{xl}(m)$ as

$$\check{R}_{xl}(m) = \frac{|\tilde{R}_{xl}(m)|}{\max_m |\tilde{R}_{xl}(m)|}$$

Compute the mean-squared error (MSE) between $\check{R}_{xl}(m)$ and $R_c(m, \Delta t)$ as

$$\text{MSE} = \frac{1}{21} \sum_{m=-10}^{10} |\check{R}_{xl}(m) - R_c(m, \Delta t)|^2$$

Find the value Δt that minimizes the MSE by searching over 1024 values of Δt equally spaced between $[-T, T]$. Call the minimizing value Δt_{r}. Take $\Delta t_{\text{ls}} = m_{\text{ns}}T + \Delta t_{\text{r}}$ as your least-squares estimate of Δt_s. Record the error $\Delta t_{\text{ls}} - \Delta t_s$.

(e) Repeat (a) through (d) 1000 times with a unique instance of the sequence $\{n_j\}$ each time. From the resulting ensemble of errors, compute the mean squared error for the sample-level estimator and the least-squares estimator. Call these values $\hat{\sigma}^2_{\text{ns}}$ and $\hat{\sigma}^2_{\text{ls}}$. Multiply the square-root of these values by the speed of light (in m/s) to find the root-mean-squared error (RMSE) in meters. What are the RMSE values you computed? Is the difference between these two RMSE values significant?

(f) We can use the Cramer-Rao Lower Bound (CRLB) to see how efficient our estimators are. Recall from lecture that the CRLB for time estimation with our baseband signal model is

$$\text{CRLB} = \frac{1}{2\beta^2_{\text{ms}} T_a \frac{C}{N_0}} \leq \sigma^2_{\Delta t}$$

The mean-squared bandwidth for the GPS L1 spreading code from the third left to the third right null is approximately $\beta^2_{\text{ms}} = 1.275 \times 10^{13}\,(\text{rad/s})^2$. Plug in the C/N_0 and $T_a = N_k T$ used in your simulation and compute the CRLB. As before, multiply the square-root of this value by the speed of light to compute an RMSE distance. What value do you obtain? How close do the two estimators come to this value?

(g) Suppose we are asked to design the spreading code for a new GNSS. An engineer proposes a signal with power spread uniformly across a bandwidth equal to f_s. The mean-squared bandwidth of this signal is $\beta^2_{\text{ms}} = (2\pi f_s)^2/12 \quad (\text{rad/s})^2$. Use this new value to compute the CRLB and RMSE distance. How does this signal's CRLB compare to that of the regular GPS L1 spreading code?

4. In lecture three loop filters were introduced; these lead to first-, second-, and third-order closed-loop transfer functions for code and carrier phase tracking loops:

- $D(s) = K$ [leads to first-order closed-loop transfer function $H(s)$]
- $D(s) = \frac{K(s+a)}{s}$ [leads to second-order closed-loop transfer function $H(s)$]
- $D(s) = \frac{K(s^2+as+b)}{s^2}$ [leads to third-order closed-loop transfer function $H(s)$]

The form of the loop filters is optimal for the step, ramp, and parabolic input, respectively, in that they minimize the integral square phase error under a bandwidth constraint.

Experiment with these loop filters as follows:

(a) Derive the closed-loop transfer function $H(s)$ for each loop, assuming the loop is composed of the loop filter $D(s)$ and a voltage-controlled-oscillator modeled as a perfect integrator $1/s$, as described in lecture.

(b) For each $H(s)$, develop an LTI model in Matlab. You may use `tf` or a related function for this. Set the parameters as discussed in lecture for a loop noise bandwidth $B_n = 10$ Hz.

(c) For each of the LTI models, calculate the response to a step, ramp, and parabolic input using the Matlab `lsim` function or equivalent.

(d) Using the final value theorem, calculate the steady-state error e_{ss} for each $H(s)$ and for each input considered in lecture: step, ramp, and parabola. Compare your calculations against the plots from your corresponding "Matlab experiments."

(e) Generate a frequency response for each of the three $H(s)$ by invoking Matlab's `bode` function.

5. Discretize the continuous-time LTI system models from Problem 4 by invoking the Matlab c2d function. Assuming a zero-order hold on the inputs as your discretization method, discretize $D(s)$ and $\text{NCO}(s) = \frac{1}{s}$ independently, then combine these with the transfer function

$$A[z] = \frac{z+1}{2z},$$

which models the phase averaging that occurs from k to $k+1$, to obtain the closed-loop discretized transfer function $H[z]$ discussed in lecture. Consider the following set of discretization intervals T, in milliseconds: $1, 10, 20, 40$. For each $H[z]$ and for each T in this set, plot the frequency response, the step response, and the unit ramp response (response to a $R(s) = 1/s^2$ input).

Given your results, what would be a good "rule of thumb" for the maximum value of the product $B_n T$ for which the behavior of the continuous-time system is approximately preserved, where B_n is the target loop noise bandwidth used in the design of the original continuous-time loop filters $D(s)$.

Calculate the *actual* loop noise bandwidth for each of the discretized loops. The actual loop noise bandwidth tends to diverge from B_n as $B_n T$ increases. Following is some code that performs this calculation on `Hz`, a Matlab model of the discrete-time closed-loop LTI system.

```
% Form the closed-loop system model
Hz = feedback(sysOpenLoop,1);

% Calculate Bn_act
walias = pi/Ta;
```

```
wvec = [0:10000]'*(walias/10000);
[magvec,phsvec] = bode(Hz,wvec);
magvec = magvec(:);
Bn_act = sum(magvec.^2)*mean(diff(wvec))/(2*pi*(magvec(1,1)^2));
```

Explain what this code does and why it accurately estimates the actual loop noise bandwidth.

Experiment with the `foh` and `tustin` discretization methods for $D(s)$ and $\mathrm{NCO}(s)$, as opposed to the `zoh` method. Explain in your own words what these do. For a given discretization interval T do they provide better frequency and step responses? Do they provide an actual bandwidth for $H[z]$ that is closer to B_n, the target bandwidth?

Do you suppose that a loop filter $D[z]$ designed entirely in the z-domain could have an arbitrarily high B_nT and remain stable? Could B_nT be higher than the maximum suggested by your rule of thumb (which applies to a system converted from continuous time to discrete time via zero-order hold)?

6. Write a Matlab function that builds a discrete-time loop filter for a feedback tracking loop. The function should assume the loop is of the form of the linearized discrete-time Costas loop model that was discussed in lecture. The function should return a state-space representation of the loop filter $D[z]$ and should also calculate the actual loop bandwidth of the closed-loop discretized system by analyzing the frequency response of the closed-loop transfer function $H[z]$.

   Your function should adhere to the following interface:

```
function [Ad,Bd,Cd,Dd,Bn_act] = configureLoopFilter(Bn_target,Ta,loopOrder)
% configureLoopFilter : Configure a discrete-time loop filter for a feedback
%                       tracking loop.
%
%
% INPUTS
%
% Bn_target ----- Target loop noise bandwidth of the closed-loop system, in
%                 Hz.
%
% Ta ------------ Accumulation interval, in seconds.  This is also the loop
%                 update (discretization) interval.
%
% loopOrder ----- The order of the closed-loop system.  Possible choices
%                 are 1, 2, or 3.
%
%
% OUTPUTS
%
% Ad,Bd,Cd,Dd --- Discrete-time state-space model of the loop filter.
%
% Bn_act -------- The actual loop noise bandwidth (in Hz) of the closed-loop
%                 tracking loop as determined by taking into account the
%                 discretized loop filter, the implicit integration of the
%                 carrier phase estimate, and the length of the accumulation
```

```
%                  interval.
%
%+------------------------------------------------------------------------------+
% References:
%
%
%+==============================================================================+
```

Hint: Here is some example code for the second-order case:

```
% Use is made here of the intermediate variables 'K' and 'a' as in the
% 'Phase-Locked Loops' paper by Gupta, 1975.
% The following calculations assume zeta = 0.707.
% Definitions of intermediate variables in terms of omegaN, zeta, Bn:
%
%  zeta^2 = K/(4*a)
%  omegaN^2 = K*a
%  Bn = (K+a)/4
%
% Set up open loop from theta[k] to thetahat[k]
K = 8*Bn_target/3;
a = K/2;
omegaN = sqrt(K*a);
Ds = K*tf([1 a],[1 0]);
Dz  = c2d(Ds,Ta,'zoh');  % Conversion to discrete time
NCO = tf([Ta],[1 -1],Ta);% zoh-disretized NCO
PD = 1/2*tf([1 1],[1 0],Ta);
sysOpenLoop = PD*Dz*NCO;
[Aol,Bol,Col,Dol]=ssdata(ss(sysOpenLoop));

% Convert the loop filter to a discrete-time state-space model
[Ad,Bd,Cd,Dd]=ssdata(Dz);
```

7. In lecture, we modeled the output of the accumulation block in the Costas phase tracking loop as

$$S_k = \frac{N_k \bar{A}_k d_i}{2} \exp(j\Delta\theta_k) + n_k = I_k + jQ_k$$

with $n_k = n_{Ik} + jn_{Qk}$. From this, derive the following model for the conventional Costas phase detector $e_k = I_k Q_k$:

$$\begin{aligned} e_k =& \frac{N_k^2 \bar{A}_k^2}{8} \sin(2\Delta\theta_k) + n_{e,k} \\ E[n_{e,k}] =& 0 \\ E[n_{e,k} n_{e,l}] =& \left[\frac{N_k^2 \bar{A}_k^2}{4} \sigma_{IQ}^2 + \sigma_{IQ}^4 \right] \delta_{kl} \end{aligned}$$

8. Write a Matlab function that executes a single update to a phase tracking loop. Assume an arctangent phase detector of the form $e = \text{atan}(Q_p/I_p)$, where I_p and Q_p are the prompt in-phase and quadrature accumulations over the interval from t_{k-1} to t_k. [1] Here, we use the shorthand notation t_k to indicate the estimated start time of the kth accumulation; t_k is equivalent to $\hat{t}_{s,k}$ from your notes. The first sample that will participate in the accumulation beginning at t_k will be $x(\tau_{j_k})$, where j_k is the minimum value of j that respects the bound $\hat{t}_{s,k} \le \tau_j$.

   Your function should perform the following three steps: (1) form the error e, (2) process e through a state-space realization of the loop filter $D[z]$, and (3) output the Doppler estimate v_k and the state of the loop filter. The function should adhere to the following interface:

```
function [xkp1,vk] = updatePll(s)
% updatePll : Perform a single update step of a phase tracking loop with an
%             arctangent phase detector.
%
%
% INPUTS
%
% s ------------- A structure with the following fields:
%
%    Ip --------- The in-phase prompt accumulation over the interval from
%                 tkm1 to tk.
%
%    Qp --------- The quadrature prompt accumulation over the interval from
%                 tkm1 to tk.
%
%    xk --------- The phase tracking loop filter's state at time tk.  The
%                 dimension of xk is N-1, where N is the order of the loop's
%                 closed-loop transfer function.
%
%  Ad,Bd,Cd,Dd -- The loop filter's state-space model.
%
% OUTPUTS
%
%
%    xkp1 -------- The loop filter's state at time tkp1. The dimension of xkp1
%                  is N-1, where N is the order of the loop's closed-loop
%                  transfer function.
%
%    vk ---------- The Doppler frequency shift that will be used to drive the
%                  receiver's carrier-tracking numerically controlled
%                  oscillator during the time interval from tk to tkp1, in
%                  rad/sec.
%
%+------------------------------------------------------------------------------+
% References:
%
```

[1] I_p and Q_p are equivalent to I_{k-1} and Q_{k-1} in the lecture notes. They are the accumulations that begin at $\hat{t}_s(\tau_{j_{k-1}}) \le \tau_{j_{k-1}}$, where $\tau_{j_{k-1}}$ is the time of the first sample in the accumulation. I_{k-1} and Q_{k-1} are called I_p and Q_p here to avoid the confusion that arises because the usual state-space loop state update $x_{k+1} = A_d x_k + B_d e_k$ would instead read $x_{k+1} = A_d x_k + B_d e_{k-1}$ under the definition of S_k from lecture. The confusion arises because the error e is actually an average phase error over the interval t_{k-1} to t_k and so it can be associated with either index $k-1$ or index k.

```
%
%+==============================================================================+
```

Write a top-level script to test your functions `configureLoopFilter` and `updatePll`. The top-level script should generate a fictitious phase time history, average this over sub-intervals of length T_a to generate a time history of prompt in phase and quadrature measurements I_p and Q_p, and send these to the `updatePll` function one-by-one for processing. Take the zero vector as the initial state x_0 of the loop filter. Even starting from this initial zero state, the tracking loop will be able to pull in and obtain phase lock as long as the initial time rate of change of the fictitious phase time history is small.

Experiment with your phase tracking loop as follows:

(a) Set $T_a = 10$ ms and the target B_n to 10 Hz.

(b) Generate a fictitious phase time history that you think will be difficult to track and see how the tracking loop performs.

(c) Modulate your phase time history with instantaneous 180-degree phase transitions, as would be caused by BPSK modulation. How does this affect the loop? Why?

(d) Add zero-mean Gaussian white noise to the I_p and Q_p measurements. Start small and increase the noise until you see cycle slipping.

(e) Figure out how to add a level of noise to the I_p and Q_p measurements that is commensurate with a given C/N_0 ratio. Then reduce the C/N_0 (increase the noise variance relative to the magnitude of the I_p and Q_p phasor) until you see cycle slipping. Identify this value of C/N_0 (expressed in dB-Hz) as the tracking threshold of your phase tracking loop.

Hint: Recall from lecture the expression

$$\frac{C}{N_0} = \frac{E[|S_k|^2] - 2\sigma_{IQ}^2}{2\sigma_{IQ}^2 T_a}$$

Note that the numerator of the right-hand side is simply the noise-free squared amplitude of the vector S_k. In your simulation, you can set this to any arbitrary value in your calculation of I_p and Q_p from the original fictitious phase time history. Then you can solve for σ_{IQ}^2 and generate noise with this variance. Add independent noise to the I_p and Q_p accumulations. Note that the C/N_0 value in the above equation is in units of Hz, not dB-Hz.

(f) How could you improve (decrease) the tracking threshold of your loop? Test your conjectures.

References

[1] D. Rife and R. Boorstyn, "Single tone parameter estimation from discrete-time observations," *IEEE Transactions on information theory*, vol. 20, no. 5, pp. 591–598, 1974.

[2] D. V. Hinkley, "On the ratio of two correlated normal random variables," *Biometrika*, vol. 56, no. 3, pp. 635–639, 1969.

[3] J. A. Nanzer, M. D. Sharp, and D. Richard Brown, "Bandpass signal design for passive time delay estimation," in *2016 50th Asilomar Conference on Signals, Systems and Computers*, Nov. 2016, pp. 1086–1091.

[4] S. Gupta, "Phase-locked loops," *Proc. IEEE*, vol. 63, no. 2, pp. 291–306, 1975.