**Name:** _____

**Instructions:**

- Submit your solutions on standard letter-size paper (8 1/2 by 11 inch).
- Hand in the completed exam by 6 P.M. on Friday, December 18, 2020 to Dr. Kassas in his office.
- Collaboration with your classmates is allowed through 6 P.M. on Thursday, December 17, 2020. No collaboration beyond this time is allowed with any other person besides Dr. Kassas. He is willing to talk about problems if he is available.
- At 6 P.M. on Thursday, December 17, 2019, an updated `mystery_data_file.bin` will be uploaded and a slightly updated version of the project will be posted, which includes an additional question involving the computation of the navigation solution. You need to submit your solution for the updated `mystery_data_file` and project.
- The solution you submit must be yours– copying/swapping solutions will be treated as cheating. If you collaborate with your classmates, you need to state the name(s) of your collaborator(s).
- You may use non-human outside sources (e.g., books). If you use such sources, then list them.
- Submit *all* your source code, design details, and plots.

**Changes to previously posted project:**

- Use the data in

  `https://overmars.eng.uci.edu/GNSS_Course/mystery_data_file2.bin`

  and track the signal corresponding to PRN 32 instead of PRN 29. Track only 60 seconds of the data.

- The front-end with which `mystery_data_file2.bin` was taken is the same NI USRP-2954R and the data format is exactly the same as that for `mystery_data_file.bin` data.

- Hand the following:

  (i) A copy of your top-level code and of your principal sub-functions, including `updatePll`, `updateDll` and your correlation function.

  (ii) Your estimate of $j_0$, the index of the first sample after the start time of the first complete C/A code interval.

  (iii) Time history plots over the full 60-second interval of the: (a) code error (in chips), (b) phase error (in degrees), (c) $\hat{f}_{D,k} = v_k/2\pi$, the estimated Doppler frequency (in Hz) (here, $v_k$ is the output of your carrier tracking loop filter (in rad/sec)), (d) estimated $C/N_0$ (in dB-Hz) as derived from your estimate of $\mathbb{E}[|\tilde{S}_k|^2]$, (e) tracked in-phase and quadrature components, (f) code phase estimate (in meters), and (g) a plot on the complex plane of each value of $\tilde{S}_k$ over the data set *after* your carrier tracking loop achieves lock. Generate the plot like this in MATLAB:

  ```
  plot(ItildeVec(iiPostLock:end),QtildeVec(iiPostLock:end), '.');
  ```

  Here, `iiPostLock` is a sample index that occurs soon after the PLL acquires phase lock on the signal. Explain what you see in this plot and how it relates to carrier phase tracking.

- Next, track only PRNs 10, 11, 14, and 31 besides 32, which you previously tracked. The following table gives the start time of C/A codes for PRNs 10, 11, 14, 31, and 32 at about 54 seconds into the data in terms of the first sample within a particular code. The last two digits of this start time have been blanked out. Determine these two digits from your tracking measurements. Then convert the start times to seconds, offset these by exactly 442681.598 seconds to bring them near to GPS time, and compute pseudoranges for all 5 PRNs.

| PRN | tRxRaw (samples) | tSv (seconds) |
|---|---|---|
| 10 | 1350006XX | 442735.598 |
| 11 | 1350020XX | 442735.601 |
| 14 | 1350001XX | 442735.604 |
| 31 | 1350014XX | 442735.605 |
| 32 | 1350013XX | 442735.601 |

- Finally, recall the model we developed in the lecture for pseudorange

$$\rho = \Delta r + c.\left[\delta t_R - \delta t_{SV}\right] + w_\rho.$$

We can augment this model to include the effects of ionospheric and tropospheric delays as

$$\rho = \Delta r + c.[\delta t_{\mathrm{R}} - \delta t_{\mathrm{SV}}] + c.[\delta t_{\mathrm{Iono}} - \delta t_{\mathrm{Tropo}}] + w_\rho,$$

where $\delta t_{\mathrm{Iono}}$ and $\delta t_{\mathrm{Tropo}}$ are expressed in seconds.

Given this model, one can construct a nonlinear least squares problem to solve for the state $\boldsymbol{x} = [x_r, y_r, z_r, \delta t_r]^\mathsf{T}$, as described in the final lecture. However, in order to solve this problem, one needs to know the position of the transmitting SV at the instant of transmission, the SVs clock offset $\delta t_{\mathrm{SV}}$ at the transmitting time instant, and $\delta t_{\mathrm{Iono}}$ and $\delta t_{\mathrm{Tropo}}$. The data in the table below are valid for the pseudoranges you calculated in the previous problem. The SV position `rSvECEF` is given in the meters in ECEF reference frame as $[x_s, y_s, z_s]$; the rest of the quantities are given in seconds.

By combining this data with your pseudorange measurements, you have all the quantities you need to solve for the state vector $\boldsymbol{x}$. Perform this solution and determine the location of the receiver that recorded the `mystery_data_file2.bin`. Express your location $[x_r, y_r, z_r]$ in meters in the ECEF reference frame. You may convert this to latitude, longitude, and altitude if you are curious about the location. Express $\delta t_r$ in seconds to six decimal places.

```
Data for PRN 10
rSvECEF = 9121305.7077300   -23601900.3375000   7954520.8506900
dtIono =    2.273761e-08
dtTropo =    1.622817e-08
dtSV = -0.000100688843623000

Data for PRN 11
rSvECEF = -22226444.3115000   -7263521.9296100   12021436.2202000
dtIono =    1.945468e-08
dtTropo =    1.321500e-08
dtSV = -0.000664972769672000

Data for PRN 14
rSvECEF = -7443729.8892700   -15525794.7538000   20526967.7486000
dtIono =    1.439808e-08
dtTropo =    9.258317e-09
dtSV = -0.000036453122726400

Data for PRN 31
rSvECEF = -6485206.0043400   -24828766.7942000   6135430.7428900
dtIono =    1.545366e-08
dtTropo =    1.003285e-08
dtSV = 0.000246467440852000

Data for PRN 32
rSvECEF = 4189914.7336700   -14908405.1919000   21590991.0269000
dtIono =    1.854755e-08
dtTropo =    1.245384e-08
dtSV = -0.000125803551001000
```

**Previously posted project:**
Develop a GPS software-defined receiver (SDR) in MATLAB (or the language of your choice, e.g., LabVIEW) that acquires and tracks a single GPS $L_1$ C/A signal. The tracking component of your code will be a mechanization of the "full system" block diagram introduced in the lecture. Apply your code to the `mystery_data_file.bin` in

`http://overmars.engr.uci.edu/GNSS_Course/mystery_data_file.bin`

which includes about 70 seconds of digitized baseband data from a National Instruments (NI) universal software radio peripheral (USRP)-2954R. The NI USRP-2954R is a low-side-mixing front-end and produces exactly 2500 samples each millisecond of receiver time. The `mystery_data_file.bin` is a binary file of a stream of I and Q samples formatted as follows:

$$I(1), Q(1), I(2), Q(2), \ldots,$$

where each $I$ and $Q$ sample is a 16-bit integer ('`int16`'). Signals corresponding to PRNs 5, 13, 15, 20, 21, 25, 26, and 29 are present in the data.

Track the GPS signal corresponding to PRN 29 over the complete data set. PRN 29's signal is strong: its $C/N_0$ approximately equals 48 dB-Hz. Make the subaccumulation interval $T_{\text{sub}}$ equal to the C/A code interval (nominally 1 ms). Set $M = 1$ so that the accumulation interval is equal to the subaccumulation interval. Because you will be tracking a strong signal, there will be no need for long coherent integration, and with a code-length accumulation interval, you will not have to worry about data bit transitions occurring during the interval.

Set $t_{\text{eml}} = 0.8$ chips and set the bandwidth of the carrier tracking loop to 10 Hz and the bandwidth of the (carrier-aided) code tracking loop to 0.2 Hz. Use a 3rd-order carrier phase tracking loop and a 1st-order carrier-aided code phase tracking loop. Empirically determine a value for $\sigma_{IQ}^2$ by using your acquisition routine to calculate and average hundreds of null-hypothesis $\tilde{S}_k$ values (e.g., corresponding to a PRN that is not present in the data).

Here is a step-by-step procedure for your top-level code:

(a) Acquire PRN 29 using your FFT-based acquisition function (or perform brute-force acquisition). The outputs of the acquisition function should be (1) an estimate $\hat{t}_{s,k=0}$ of the start time of the 0th code interval [you can approximate this as the time of the data sample that immediately follows the beginning of the 0th C/A code ($\tau_{j_0}$)], and (2) the approximate Doppler frequency $f_{D,0}$ over the 0th code interval.

(b) Initialize the beat carrier phase estimate $\hat{\theta}(\tau_{j_0})$ to some arbitrary value (e.g., 0).

(c) Initialize a moving-window average of $|\tilde{S}_k|^2 = \tilde{I}_{p,k}^2 + \tilde{Q}_{p,k}^2$ with the peak $\tilde{I}_{p,0}^2 + \tilde{Q}_{p,0}^2$ value obtained during acquisition. This average can be converted to $C/N_0$ and is useful for calculating the normalization constant $C$ required in the dot-product code phase detector.

(d) Call the `configureLoopFilter` function to set up the phase tracking loop filter.

(e) Figure out how to set the initial state $\boldsymbol{x}_{k=0}$ of the phase tracking loop loop filter so that for an initial error signal $e_0$ equal to 0, the loop would output $v_0 = 2\pi f_{D,0}$, where $f_{D,0}$ is the Doppler estimate (in Hz) produced by the acquisition routine. "Priming the loop filter" like this is necessary for all 2nd- and 3rd-order carrier tracking loops because they have state (memory).

(f) Pass the input data $x(j)$ to a function that performs correlation over one subaccumulation interval to produce early, prompt, and late subaccumulations $\tilde{S}_{e,k}, \tilde{S}_{p,k}, \tilde{S}_{l,k}$. The function should mechanize the correlation "recipe" introduced in Problem 1 of Homework Assignment #5. Other inputs to the function will include: $f_{IF}$, $\hat{t}_{s,k}$, $f_{D,k}$, $\hat{\theta}(\tau_{j_k})$, $t_{\text{eml}}$, and the target PRN. The function should call a sub-function that generates early, prompt, and late versions of the oversampled C/A code corresponding to the target PRN.

(g) Update the moving-window average of $|\tilde{S}_k|^2 = \tilde{I}_k^2 + \tilde{Q}_k^2$ using the prompt accumulation.

(h) Route the correlation outputs $\tilde{S}_{e,k}, \tilde{S}_{p,k}$, and $\tilde{S}_{l,k}$ to `updatePll` and then to `updateDll`.

(i) Use the outputs of `updatePll` and `updateDll` to obtain $\hat{\theta}(\tau_{j_{k+1}})$ and $\hat{t}_{s,k+1}$. Set $f_{D,k} = v_k/2\pi$ Hz, where $v_k$ is the output of the PLL's loop filter.

(j) If all data have been exhausted, then quit. If not, set $k \leftarrow (k+1)$ and go to step (f).

Now, to generate a navigation solution, recall that in the lecture we defined the pseudorange measurement in terms of the recipe by which it is computed within a GNSS receiver:

$$\rho(t_r) = c \cdot [t_r - t_s(t(t_r) - \delta t_{\text{TOF}})] \,.$$

In this expression, $c$ is the speed of light, $t_r$ is the receiver clock time at receipt of the alignment feature, $t(t_r) - \delta t_{\text{TOF}}$ is the true time at the instant the alignment feature was transmitted by the satellite, $t_s$ is the satellite clock time at that time instant, and $\delta t_{\text{TOF}}$ is the signal time-of-flight. Time values are typically expressed in seconds and $c$ in meters per second, leaving $\rho$ in units of meters. The alignment feature is usually taken to be the beginning of a spreading code sequence (e.g., the beginning of a 1-ms GPS $L_1$ C/A code), or the beginning of some chip within the spreading code.

Apply your SDR for PRNs 5, 15, 18, 20, 21, and 29. Each sample in the data set arrives with a uniform spacing of $T_s = (2.5 \times 10^6)^{-1} = 400$ ns according to the receiver clock. In fact, the sample train can be thought of as *defining* the receiver clock: the arrival of each sample is the "tick" of the receiver clock.

If you have coded your tracking loops correctly, you will find that the spreading code phase for PRN 29 is aligned such that a particular C/A code happens to begin just after sample 138843337 (assume that the first sample is considered to be sample 0). Let's call the start of this C/A code our "alignment feature." As it turns out, this alignment feature was transmitted from the PRN 29 satellite at the following GPS time: 1915 weeks, 419242.767 seconds. GPS time is counted in weeks and seconds of week since 00:00:00 on January 6, 1980 (a Sunday). This is "zero hour" for GPS. This is a convenient time base—weeks and seconds are much easier to work with than years, months, days, hours, minutes, and seconds.

In the table below, you will find start times of C/A codes for the PRNs you were asked to track. These start times occur about 55 seconds into the data. The start times are given in terms of *elapsed* receiver samples (the first sample corresponds to 0 elapsed samples, the second to 1 elapsed sample, etc.). They are all within 1 ms of each other according to the receiver clock. You will also find the time of transmission of the beginning of the corresponding C/A codes, according to the satellite clock. Only the seconds of week are given. Notice that these "time stamps" applied by the satellite are given in integer milliseconds. This is because, according to each satellite's clock, each

C/A code starts exactly 1 ms after the last one, with the first C/A code of each week starting at exactly 00:00:00 on Sunday morning.

| PRN | t_r (samples) | t_s (seconds) |
|---|---|---|
| 5 | 138842507.3477026820182800 | 419242.758 |
| 15 | 138842886.9786712825298309 | 419242.758 |
| 18 | 138844812.2175282239913940 | 419242.759 |
| 20 | 138844629.4773647189140320 | 419242.767 |
| 21 | 138843158.2532410919666290 | 419242.760 |
| 29 | 138843337.5224231183528900 | 419242.767 |

From the data in this table, calculate pseudorange measurements for each of the 6 PRNs. You will need to convert the receiver time stamps from samples to seconds and apply a constant offset to bring the receiver time stamps close to GPS time (expressed here in seconds of week). Your pseudorange values may be different from others in the class because the offset you choose is arbitrary.