University of Colorado - Boulder

ASEN 6519: Advanced State Estimation

Final Project Report

Maximum-Likelihood Estimation of Signal-to-Noise Ratio for Binary Data in AWGN

Author:
Fernando Palafox[a]

[a]SID: 107070363

Professor:
Nisar Ahmed

A maximum-likelihood estimator of signal-to-noise (SNR) is derived for a baseband signal comprised of a binary code scaled by an unknown amplitude and additive white Gaussian noise with an unknown noise power spectral density. This estimator does not require the use of hardware components not already found within a typical GPS receiver, and instead leverages the output of the correlators used within the tracking phase of the signal processing chain. The performance of the estimator is then analyzed by computing its bias and then testing it with a numerical simulation. This simulation generates a simplified GPS signal with a known signal power, adds noise with a known charateristics and then compares the estimated SNR to the true SNR. Finally, the work is contextualized with respect to another important signal strength metric (C/N_0) and future work is suggested.

Ann and H.J. Smead
Aerospace Engineering Sciences
UNIVERSITY OF COLORADO **BOULDER**

I. Application and Context

The Global Positioning System (GPS) is a satellite-based radionavigation system which can be used to provide positioning and timing information to any GPS receiver around the world. This system relies on signals transmitted by satellites orbiting Earth at approximately 20,200 kilometers. Although these signals are transmitted at 27 watts, path losses and interference mean they reach Earth with a power of only 10^{-16} watts – well below the thermal noise floor. Fortunately, GPS signals and receivers are designed to work under such conditions. However, the computation of accurate position solutions still relies on keeping track of satellite signal strengths and only selecting to use data from the strongest signals. The decision whether to use a signal or discard it is done by computing metrics such as signal-to-noise ratio (SNR) which provide the user with a quantitative measure of signal health. Motivated by a desire to understand how these computations are done, I set out to derive a maximum-likelihood estimator of signal-to-noise ratio by following the ideas of [1] and applying them in the context of a simulated simplified GPS signal.

GPS signals are composed of a carrier wave transmitted at 1,575.24 MHz on which data is modulated using binary phase-shift keying modulation and then used by any GPS receiver to compute a position solution. Additionally, each satellite's signal is also modulated by a unique binary code known as a C/A code which has high auto-correlation, low cross-correlation with other C/A codes, and spectrum spreading properties. This extra layer of modulation allows the entire constellation of GPS satellites (approximately 31) to transmit on the same frequency band without interference. Additionally, through the use of a correlator filter, it allows for the recovery of the received signals from below the thermal noise floor. A diagram of the modulation scheme for the signals broadcasted by each satellite can be seen in figure 1.

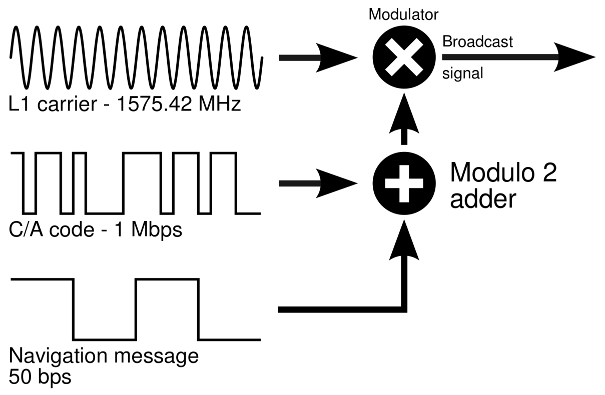

Fig. 1 GPS broadcast signal. Source: Wikipedia.

The entire architecture of a GPS receiver is too complicated to be outlined in this investigation. Fortunately, a maximum-likelihood estimator (MLE) for SNR can still be derived by having a high-level understanding of the flow of signals within the tracking module of a GPS receiver. Within this module, the received signal is mixed with a replica of the carrier wave in order to bring down the signal into baseband - effectively turning the signal into a stream of 1's and 0's plus noise. The resulting signal $y(t)$ (which will be fully defined in the next section) is correlated with a replica of the code $x_i(t)$ using equation 11, which provides a quantitative measure of whether the signal broadcasted by satellite i is present in $y(t)$ or not. Moreover, the correlation results can be used to extract the navigation message from the sampled signal. For example, if a recorded signal $y(t)$ contains broadcasts from satellites 1 and 2, but not 3, correlating $y(t)$ with $x_1(t)$ and $x_2(t)$ will result in a large positive number. On the other hand, correlation of $y(t)$ with $x_3(t)$ will output a number close to zero. Note that in the tracking module, it is assumed that the job of determining which broadcasts are present within a received signal has already been done*, and that correlation is conducted not to determine the presence of a satellite, but to quantify the strength of the received signal. For this investigation, the output of the correlators will be used to determine the SNR of the received signals as it leverages quantities that are already being calculated, and does not require any additional hardware within a receiver.

$$\phi = \frac{1}{T_w} \int_0^{T_w} y(t)x(t)\,dt \tag{1}$$

$$T_w \triangleq \text{duration of code [s]}$$

*In a GPS receiver, a module preceding the tracker goes through a process known as acquisition, in which it iteratively correlates the received signal with a number of C/A codes to determine what satellites are present in the signal. This information is then relayed to the tracking module which will then only attempt to correlate the signals for those satellites. This might seem like the same task is being done twice, that is, running a correlator with codes that are known to be present in the received signal. The reason this happens is that the tracking module does not just run correlation once for detection purposes, it does so continuously through the entire data set in an effort to estimate both changes in the received code phase and Doppler in the carrier due to receiver or satellite movement. Changes in correlation strength care used to drive a control loop which is where the tracking module gets is name, as a control loop makes sure that the generate replicas are always synchronized with the incoming signal. Although important, understanding these details are is not required to understand the derivation of the estimator

II. Problem Formulation

The received signal, after being mixed down to baseband will be denoted by

$$y(t) = Ax_i(t) + n(t) \tag{2}$$

or in discrete form

$$y(k) = Ax_i(k) + n(k) \tag{3}$$

where x_i represents the C/A code corresponding to the ith satellite, and is normalized such that

$$\frac{1}{T_w}\int_0^{T_w} x_i(t)\,dt = 1 \tag{4}$$

A is an unknown amplitude factor such that $S = A^2$ is the received signal power. $n(t)$ is assumed to be additive white Gaussian noise with unknown power spectral density N_0. Communication in the presence of white Gaussian noise is often parameterized by the following SNR ratio

$$\xi \triangleq \frac{ST}{N_0} = \frac{A^2T}{N_0} = \frac{\text{(received signal energy)/(bit)}}{\text{(noise power)/(unit bandwidth)}} \tag{5}$$

where T is the time per bit. This is because the numerator represents the parameters which may be varied by the transmitter, and the denominator represents properties of the environment within which the signal is being transmitted [2]. Then, given only a received signal $y(t)$ computation of ξ requires an estimate of the unknown parameters A and N_0.

III. Objectives

The objectives for this investigation are as follows:

Level 1: Derivation of MLE estimator for SNR
Level 2: Analysis of estimator bias and variance
Level 3: Estimator performance using simulated signals

IV. Methods

Level 1: Derivation of MLE estimator

We begin with an analysis of the output of the correlator. Plugging the description of the continuous received signal 2 into the correlation equation 11 results in

$$\phi_i = \frac{1}{T_w}\int_0^{T_w} (Ax_i(t) + n(t))x_i(t)\,dt$$

Expanding into two integrals and simplifying using the assumed code normalization in equation 4

$$\phi_i = A + \frac{1}{T_w}\int_0^{T_w} n(t)x_i(t)\,dt$$

The second term is essentially the sum of completely independent random variables distributed according to a zero-mean Gaussian distribution with variance N_0. Which means that by the central limit theorem it can be re-written as

$$\phi_i = A + n_\phi, \quad n_\phi \sim N(0, {}^{N_0}\!/_{T_w})$$

Therefore, the probability of a single correlator output is given by

$$P(\phi) = \sqrt{\frac{T_w}{2\pi N_0}}\exp\left(-\frac{T_w}{2N_0}(\phi - A)^2\right)$$

And the likelihood function for set of M correlator outputs (all for the same satellite i) is

$$\mathcal{L}(\phi_{1:M}) = \left(\frac{T_w}{2\pi N_0}\right)^{\frac{M}{2}} \exp\left(-\frac{T_w}{2N_0}\sum_{j=1}^{M}(\phi_j - A)^2\right)$$

The log-likelihood function, after some simplifications is

$$\log\mathcal{L} = \frac{M}{2}\log\left(\frac{T_w}{2\pi}\right) - \frac{M}{2}\log(N_0) - \frac{T_w}{2N_0}\left(\sum_{j=1}^{M}\phi_j^2 - 2A\sum_{j=1}^{M}\phi_j + MA^2\right)$$

Which can be further simplified by noting that assuming constant A and noise properties, a large enough M means noise will cancel out when summing correlator outputs for the same satellite. Therefore,

$$\sum_{j=1}^{M}\phi_j = \sum_{j=1}^{M}(A + n_\phi) \approx MA$$

And the final log-likelihood function is given by

$$\log\mathcal{L} = \frac{M}{2}\log\left(\frac{T_w}{2\pi}\right) - \frac{M}{2}\log(N_0) - \frac{T_w}{2N_0}\left(\sum_{j=1}^{M}\phi_j^2 - MA^2\right)$$

This distribution is of the form admitting sufficient statistics, therefore, $\log\mathcal{L}$ will have a single unique maximum which depends on A and N_0 [1]. This maximum can be found by computing the partial derivatives with respect to A and N_0 and setting them equal to zero. Then, the following estimators can be recovered

$$\hat{A} = \frac{1}{M}\sum_{j=1}^{M}\phi_j \tag{6}$$

$$\hat{N}_0 = \frac{T_w}{M}\left(\sum_{j=1}^{M}\phi_j^2 - MA^2\right) \tag{7}$$

From which an estimate of ξ can be obtained in the form of

$$\hat{\xi} \triangleq \frac{T\hat{A}^2}{\hat{N}_0} \tag{8}$$

The attentive reader should note that something is awry here. This paper claims to derive a maximum-likelihood estimator of SNR. However, so far the only thing I've derived are two maximum-likelihood estimators for A and N_0, and then put together to form an estimator for SNR. Unfortunately, with the information presented so far, one can only say that the resulting $\hat{\xi}$ is an estimator, but not that it is an MLE (and therefore enjoy the properties of an MLE). Fortunately, someone has thought of this before [3] and shown that indeed, the ratio of two MLE's is actually the MLE of the ratio (phew). Therefore, we can confidently say that an MLE of ξ has been derived.

Level 2: Estimator Performance

It is known that maximum likelihood estimators, if they are statistically sufficient, guarantee minimum variance (Cramer-Rao Lower Bound). However, no guarantees can be made about biases. $\hat{A}$ can be shown to be unbiased

$$\mathbf{E}(\hat{A}) = \frac{1}{M}\sum_{j=1}^{M}\mathbf{E}(\phi_j) = \frac{1}{M}\sum_{j=1}^{M}A = A$$

On the other hand, computing the expectation of $\hat{N}_0$ results in

$$\mathbf{E}(\hat{N}_0) = \frac{T_w}{M}\left(\sum_{j=1}^{M}\mathbf{E}(\phi_j^2) - M\mathbf{E}(\hat{A}^2)\right) = \frac{T_w}{M}\left(M\left(A^2 + \frac{N_0}{T_w}\right) - MA^2 - \frac{N_0}{T_w}\right) = \frac{M-1}{M}N_0$$

Scaling the estimator, results in the following minimum-variance unbiased estimator for N_0

$$\hat{N}_0 = \frac{T_w}{M-1}\left(\sum_{j=1}^{M}\phi_j^2 - MA^2\right) \tag{9}$$

From which a new SNR estimator can be built

$$\hat{\xi} = \frac{M-1}{N_0}\left(\frac{A^2}{\sum\phi^2 - MA^2}\right) \tag{10}$$

Note that the even though the component estimators are now unbiased, the overall estimator may still have a bias.

Level 3: Truth-model Testing

In order to verify whether this estimator works as expected, I ran it through a script which generated noisy data with known characteristics, computed SNR using the previously derived estimators and then compared it to the true SNR. First I generated a real 1023 bit C/A code with a bit rate of 1.023 MHz (same as what a satellite would transmit) and sampled it at 2MHz, a typical sampling frequency for a GPS receiver. This resulted in a vector $x(k)$ containing the sampled C/A code and a code duration of T_w = 1ms. This vector was then scaled by a factor A such that the equivalent received signal power without noise was -75 dBW. Noise samples $n(k)$ were generated using a standard Gaussian distribution. Then, they were scaled by noise factor which was computed by starting with a typical noise floor of -135 dBm/Hz, converting into the equivalent noise power spectral density of $N_0 = 3.1623 \times 10^{-17}$ W/Hz, and then computing noise power with the following formula

$$P_n = 2BN_0 \ \ [W]$$

where B is the sampling bandwidth. The sampled code and generated noise were then added together to form samples

$$y(k) = x(k) + \sqrt{(P_n)}n(k)$$

which correspond to the received signal after it is mixed down to baseband (and without any navigation data bits). $y(k)$ was then fed into equation 10 to generate an estimate of SNR. For the computation of any correlation operations, the following formula for discrete correlation was used:

$$\phi = \frac{1}{T_w}\sum_{k=1}^{N_s} y(k)x(k)\Delta t \tag{11}$$

$$N_s \triangleq \text{samples per code}$$
$$\Delta t \triangleq \text{sampling interval [s]}$$

V. Results

Simulation results were promising and outlined in the following table

	Units	Truth	MLE	Error
A	[V] or [A]	1.7783×10^{-4}	1.7766×10^{-4}	1.6430×10^{-7}
N_0	[W/Hz]	3.1623×10^{-17}	3.5362×10^{-17}	3.7391×10^{-18}
ξ	[dB]	29.9012	29.4079	0.4934

The results look very good, with a final SNR very close to the true value. However, literature on the subject suggested that these estimators might not be as reliable for all possible combinations of A and N_0. Therefore, I decided to run estimates for a fixed N_0 but a varying A and took a look at the corresponding SNR and erorrs for every case. The results are seen in figures 2 and 3 and show that the estimators indeed fail to accurately compute SNR as A goes above a value of 0.2×10^{-3}. This was very interesting result and points to the fact that even if certain properties can be proved for set of estimators (such as zero bias), using them to compute new value may yield yield an estimator which does not share the same properties as its components.

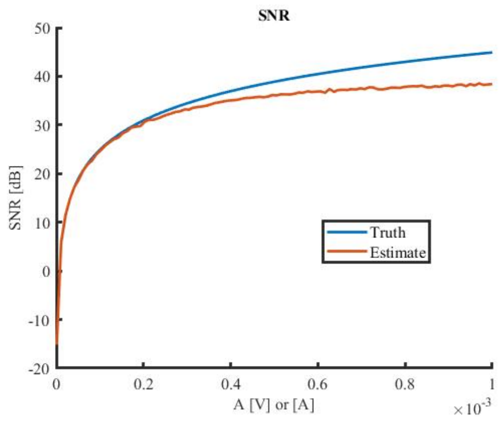

Fig. 2 Divergence in SNR estimate with increasing A

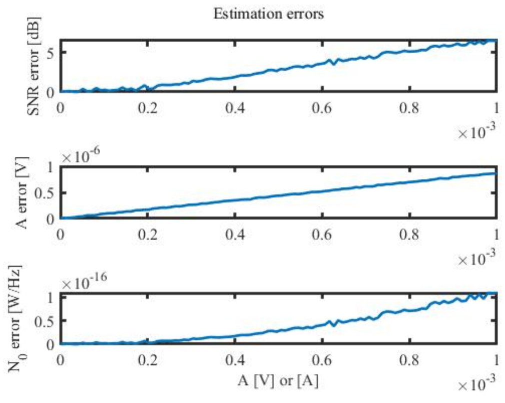

Fig. 3 Increasing estimation errors with increasing A

VI. Discussion & Conclusions

The derived maximum-likelihood estimators forms a basis for computing a measure of the signal-to-noise ratio of a binary code in additive white Gaussian noise. However, much work remains to be done. For example, perfect correlation was assumed throughout the entire derivation when in reality, the control loop in the tracking module of a GPS receiver is not perfect, and code phase errors are bound to be introduced into the correlators. Furthermore, additional sources of errors commonly encountered by GPS receivers, such as multi-path, might mean render invalid the assumption of independence between correlator outputs. Additionally, testing could be improved by adjusting the values of A and N_0 to better represent a real-life situation. Although I did plenty of research to come up with a simulation that I think is representative of a real signal, my lack of technical proficiency in communication theory means that I had trouble intepreting the significance of the results. For example, we can tell that the estimator does not have great performance as A increases, but how significant is this drop-off in estimation accuracy? Perhaps the only region that really matters in a real life scenario is the one below 0.2×10^{-3}V, where the estimator is still pretty good. It would also be useful to test the algorithm with real GPS data. This might yield useful information if the resulting SNR estimates are compared to those given by other proven estimators such as a Signal-to-Noise Variance (SNV) estimator or a Moment Method (MM) estimator.

Finally, it is important to point out a fundamental characteristic of the estimation of signal strength in GPS (and in spread-spectrum signals in general). Recall that a GPS signal is composed of a carrier wave modulated by a C/A code and navigation data bits, and then note how the MLE estimator derived in this investigation only makes reference to the strength of the C/A code after the carrier wave has been mixed out. However, computing the full relationship between signal power and noise power in GPS involves an extra step not discussed in the paper: the quantification of the strength of the received carrier wave. This is why, for GPS signals a common measure of the relationship between the received signal strength and the noise power is *carrier*-to-noise ratio expressed as C/N_0 which compares the power of the original unmodulated carrier wave before being spread by a C/A code during transmission by a satellites [4]. Although details on how C/N_0 is computed are beyond the scope of this paper, it is sufficient to say that it is a very useful metric because it is independent of the receiver bandwidth and processing stage. On the other hand, intepreting SNR requires specifying these parameters and as well as the entire signal processing chain. Perhaps an interesting future investigation would be the derivation of a maximum likelihood estimator for C/N_0.

In any case, this investigation was an immensely useful exercise in cohesively combining my knowledge of GPD and estimation theory in order to deepen my understanding of a subject I was interested in. The multi-disciplinary nature of the investigation meant that I had to spend many hours getting up to speed with concepts within subjects such as communication theory and electrical engineering that I did not have much experience with. Additionally, it was also an exercises in patience as I kept running into dead-ends trying to gain a deeper understanding of concepts mentioned within the literature. In many cases, I would be slowed down by unproven or un-cited statements that were presumably obvious to the writers, but certainly not to me. Although this was very annoying, it forced me to practice the skill of reading between the lines in an effort to understand what the authors were trying to communicate beyond what was written on any one paper or presentation. I also had to deal with the reconciliation of statements from different sources which initially seemed contradicting, but actually were not once I completely understood notation, assumptions and definitions. In the end, being able to put something together that I feel like I have a solid understanding of after many hours of combing through obtuse literature was a very rewarding endeavour and as I head towards a PhD, I am excited for what is to come.

VII. References

[1] Kerr, Robert B. “On Signal and Noise Level Estimation in a Coherent PCM Channel.” IEEE Transactions on Aerospace and Electronic Systems, vol. AES-2, no. 4, July 1966, pp. 450–454, 10.1109/taes.1966.4501796. Accessed 5 May 2022.

[2] Viterbi, A. J. “On Coded Phase-Coherent Communications.” IRE Transactions on Space Electronics and Telemetry, vol. SET-7, no. 1, 1961, pp. 3–14, 10.1109/iret-set.1961.5008745. Accessed 5 May 2022

[3] Gagliardi, R., and C. Thomas. “PCM Data Reliability Monitoring through Estimation of Signal-To-Noise Ratio.” IEEE Transactions on Communications, vol. 16, no. 3, June 1968, pp. 479–486, 10.1109/tcom.1968.1089851. Accessed 5 May 2022.

[4] GNSS, Inside. “Measuring GNSS Signal Strength.” Inside GNSS - Global Navigation Satellite Systems Engineering, Policy, and Design, 2 Dec. 2010, insidegnss.com/measuring-gnss-signal-strength/. Accessed 5 May 2022.

VIII. Appendix

```
%% Housekeeping
clear all; close all; clc;

%% Parameters
settings.T_w = 1e-3; % Length of a single word [s]
settings.T_M = 1; % Length of colleceted data [s]
settings.T = 1.023e6; % chipping rate [chips/s]
settings.N_b = 1023; % Number of bits [bit]
settings.F_s = 2e6; % Sampling rate [Hz]

n_f = -135; % Standard outside noise [dBm/Hz]
p_r = -75; % Received power in dBw


%% Calculate noise spectral density and amplification factor A

N_f_dbhz = n_f - 30;
settings.N_0 = 10^(N_f_dbhz/10);

bandwidth = settings.F_s/2;
noise_power = 2*bandwidth*settings.N_0;
settings.A = sqrt(10^(p_r/10));

%% Load C/A code
code = generateCAcode(1);
code_2 = generateCAcode(2);

% Append code
settings.M = settings.T_M/settings.T_w;
code_app = repmat(code,1,settings.M);

%% Sample and add noise

% Sample code
settings.samplesPerReplica = round((settings.F_s/settings.T) * settings.N_b);
samplesTotal           = settings.M*settings.samplesPerReplica;

x_1_single = sampleCode(code,settings.samplesPerReplica,settings.F_s,settings.N_b,settings.T);
x_2_single = sampleCode(code_2,settings.samplesPerReplica,settings.F_s,settings.N_b,settings.T);

x        = sampleCode(code_app,samplesTotal,settings.F_s,settings.N_b,settings.T);

% Generate noisy samples
y = settings.A*x + sqrt(noise_power)*randn(1,samplesTotal);

%% Correlation

% Compute correlation
phi = correlation_fn(y,x_1_single,settings);
phi_2 = correlation_fn(y,x_2_single,settings);

% Plot correlation comparison
figure
hold on
plot(phi)
plot(phi_2)
hold off
```

```
legend('1','2')

%% SNR estimation using derived estimator

A_hat  = 1/settings.M*sum(phi);
N_0_hat = settings.T_w/(settings.M-1)*(sum(phi.^2) - settings.M*A_hat^2);
SNR_hat = (settings.M - 1)/settings.N_b*(A_hat^2)/(sum(phi.^2) - settings.M*A_hat^2);
SNR_hat_db = 10*log10(SNR_hat);

SNR_true = (1/settings.T)*settings.A^2/settings.N_0;
SNR_true_db = 10*log10(SNR_true);

SNR_error = abs(SNR_hat - SNR_true);
SNR_eror_db = abs(SNR_hat_db - SNR_true_db);

A_error = abs(A_hat - settings.A);
N_0_error = abs(N_0_hat - settings.N_0);

%% Test a variety of A values

test_ranges(linspace(1.0000e-06,1e-3,100),3.1623e-17,settings);
```