Notes on
Stochastic Errors of Low Cost MEMS Inertial Units

Yigiter Yuksel
&
Huseyin Burak Kaygisiz

Contents

Part I

Introduction

This technical note discusses the stochastic errors of inertial units. Despite the fact that there are many texts on the sensor error modeling, almost all of these existing texts (if not all) lack any solid understanding. They simply replicate what has been written in the past for high-end inertial sensors without considering the peculiar characteristics of MEMS units. As a result, these existing texts are not useful for anything other than confusing junior navigation engineers, who apparently do not posses any critical thinking capability.

In this technical note, I will not describe the meaning of any error source. Instead, I will use [1, Annex-C] as a base for the error model definitions and describe what those models should mean for low cost INS designers. I will try to focus on the practical use of these error model parameters in the integrated navigation algorithms rather than any theoretical discussion. I assume that the readers of this technical note have already read Annex C of [1] and understood it thoroughly.

1 Why do we prefer Allan variance rather than standard ARMA modeling tools?

In electrical engineering curriculum, stochastic system identification is mainly developed around ARMA models which have a very strong theoretical background. There are also a good variety of tools ranging from normal equations to adaptive filters (even H_∞ filters) to deal with them. Therefore, one may argue that we should stick to ARMA models rather than focusing on two point difference power called Allan variance which is not even mentioned in any canonical book of electrical engineering (including signal processing).

As a matter of fact, AR(MA) modeling methods may at first seem very promising to everyone. The output of the Allan variance method is nothing but a simple curve made up of pseudo linear segments. One has to convert these curves into meaningful Markovian models before being able to use them in the Kalman filters. On the other hand, the end result of the ARMA tools is the model itself. The estimated ARMA coefficients can be directly inserted into the Kalman filters. That is why, one may consider ARMA modeling method a better alternative to Allan variance methods.

However, in reality this is not the case at all. ARMA modeling methods have a very serious problem which render them practically useless for inertial sensors. These methods assume that the entire state of the process (the state of all errors) is completely observable. However, the sensor outputs in a real sensor (i.e. the only state observable):

1. is certainly buried into high power additive white noise,
2. is usually sum of independent stochastic processes.

Under these conditions ARMA modeling cannot be used. There are some ways to remove the effect of additive white noise on the estimated ARMA coefficients. However, there is no known way of determining the coefficients of individual processes from the sum of them. It is interesting to note that although there are significant amount of papers on the use of ARMA tools in the inertial sensor modeling, none of them has ever been able to solve (or even explicitly mention) these problems. Therefore, at least for now, you should simply ignore these ARMA methods and stick to Allan variance.

2 What do we model in the Allan variance method?

The output of calibrated inertial sensor which is stationary can be represented as:

$$y(t) = u(t) + e(t) + b(T) + N(a, \omega, T, t) \tag{1}$$

where $y(t)$ is the nominally calibrated sensor output, $u(t)$ is the real kinematic value that the sensor is supposed to measure (i.e. real acceleration (a) or rotation rate (ω)) ,T is the temperature, $b(T)$ is the bias (which is probably dependent on the temperature), and $N(a, \omega, T, t)$ is the error induced by all the environmental factors (N is usually not a linear function).

The Allan variance can only be used to model the time varying component $e(t)$. This means that:

1. You must be sure that the only changing component during Allan variance tests is $e(t)$. If the temperature (T) changes during tests, $N(a, \omega, T, t)$ and $b(T)$ also changes which completely spoils the long term results. This is usually the main reason why you observe an inconsistent rising edge at the end of the Allan curves.
2. $N(a, \omega, T, t)$ and $b(T)$ must be modeled based on the repeated calibration tests. (Repeated calibration tests will not be discussed in this document at all).
3. Allan variance does not say anything about the bias component $b(T)$ which is usually represented as random constants. If someone asks you how you determine the initial variance of your bias, you must never answer him saying that you computed it from an Allan variance test.[1]
4. As it is difficult to keep the temperature constant, it is usually best to model $b(T)$ first using some other ad-hoc method and then compensate its effect before computing the Allan variance. This point is extremely crucial to obtain consistent Allan variance results for low cost units.

[1]Flicker noise floor can be used to determine initial uncertainty of bias as explained in Section 4. However, this is not possible for low end MEMS units.

5. You must always collect temperature data together with the inertial data during the Allan variance tests. If you do not have a previously computed temperature compensation function, at least fit a linear curve ($y(T) = aT + b$) to the collected data and subtract it from the sensor outputs before computing the Allan variance.

Part II
Sensor error components

It is generally assumed that $e(t)$ in equation (1) is the sum of the following components:

$$e(t) = ARW(t) + F(t) + Q(t) + S(t)$$

where

1. $ARW(t)$ is the angle random walk error
2. $F(t)$ is the flicker noise error
3. $Q(t)$ is the quantization error
4. $S(t)$ is the sinusoidal error

In the following section, these error components will be elaborated separately. The use of 1st order Markov processes in the form of $\dot{x} = ax + \omega$ will be discussed under the topic of approximating the flicker noise for the Kalman filters in Section 4.3.

3 Angle/Velocity Random Walk

Angle/velocity random walk (A/VRW) is the additive white noise component on the sensor outputs. It is interesting to note there are still people out there who claim to be an INS expert and yet do not even know this very basic fact. For instance, here is a comment from an IEEE reviewer for my paper:

> "The noise v has been explained in different way. In line 47, v is a white noise whereas v random walk in line 48. What is v?"

Although this particular reviewer has no idea what an angle random walk is, he accepted to review a paper on pure INS. Unfortunately, anyone who knows how to integrate a signal considers himself an INS expert nowadays.

A/VRW components are defined based on the (constant) value of their flat spectral levels. The low cost MEMS sensors almost always provide digital outputs. Therefore, it might be a little bit confusing to relate the specified power spectral levels of a white noise to the observed variance of actual (digital and discrete) sensor outputs. The following numerical examples illustrates how you can use the specification sheet values to compute the sample output variances of inertial sensors.

Example 1. ADXRS646 is a MEMS gyroscope with an analog rate output. The ARW value of ADXRS646 is specified as $0.01\,^\circ/\mathrm{sec}/\sqrt{\mathrm{Hz}}$ in [2]. Assuming that we sample this gyroscope with a perfect ADC at 25 Hz, what will be the standard deviation of the sampled data when the gyroscope is perfectly stationary?

If ADXRS646 were a rate integrating gyroscope then the answer would be:

$$\sigma_{ARW} = 0.01 \times \sqrt{25} = 0.05^\circ/\mathrm{sec}$$

However, ADXRS646 is a rate gyroscope rather than a rate-integrating type. Therefore, we cannot use the sampling frequency as the noise bandwidth parameter at all (farewell to good old days). As a matter of fact, the sampling frequency has nothing to do with the additive noise power for the rate gyroscopes. Therefore, the information given in this example is not sufficient for us to compute the noise power.

Example 2. Every rate sensor has some bandwidth adjustment option. The specification sheet of ADXRS646 [2] says that the sensor bandwidth must be set with a capacitor connected to its output pin. The last stage of ADXRS646 is an active filter with an op-amp. Therefore, the capacitor works like simple 1st order RC circuit with the internal 180kΩresistor. (See [2, Figure 1].) Let us assume that we connect a 0.009μF capacitor to the output pin. If we sample the sensor outputs again at 25Hz (with a perfect ADC), what would the standard deviation of the sampled data be?

In this configuration the bandwidth of the filter is equal to:

$$B = \frac{1}{2\pi CR} \approx 100\mathrm{Hz}$$

Therefore,

$$\sigma_{ARW} = 0.01 \times \sqrt{100 * 1.57} = 0.125^\circ/\mathrm{sec}$$

As you can see, the result is independent of the sampling period. The $1.57 = \frac{\pi}{2}$ coefficient inside the square root comes from the bandwidth definition of the 1st order RC circuits. If you are not comfortable with it, you can simply use 2 instead of 1.57.

$$\sigma_{ARW} = 0.01 \times \sqrt{100 * 2} = 0.14/\mathrm{sec} \tag{2}$$

Example 3. Suppose, the maximum frequency content of the motion that the ADXRS646 is supposed to measure is less than 25Hz. In this case, the 25Hz bandwidth can be considered as a suitable selection for the sensor. However, just to be safe we may prefer a 100Hz bandwidth. Or, perhaps the hardware designer may arbitrarily choose 100Hz even without asking us. Therefore we may not have another choice other than 100Hz.

Suppose we are designing a navigation system for humans who do not need to learn their position more often than 10Hz. In this case you probably prefer to run your INS at 10Hz. (Why would you want to run the INS faster and waste your processing power?) In this case, what should be your sampling rate?

The bandwidth of the motion is 25Hz. Therefore, you must choose at least 50Hz (2 times the bandwidth). The power of the additive white noise at 50Hz will be equal 0.14^2 as shown in equation (2). As you want to run your INS at 10Hz, you must compute the average of all 5 consecutive samples and use this average in your INS equations.[2] In this case, the power of the additive white noise on each sample that runs the INS will be:

$$\sigma_{ARW}^2 = 0.14^2/5 \approx 0.004(^\circ/\text{sec})^2 \tag{3}$$

However, under the specified conditions 50Hz sampling rate is not a suitable choice. Because, although you can theoretically sample your motion without any problem at 50Hz, you will be using you sensor in a non-efficient way. To see why this is indeed the case, let us repeat the above calculations with a 100Hz sampling rate.

At 100Hz, the power of additive noise on each sensor output will again be 0.14^2 as shown in equation (2). However, in this case, we will be using the average of 10 consecutive samples to run the INS. Therefore, the power of the white noise on each sample to run the INS will be:

$$\sigma_{ARW}^2 = 0.14^2/10 \approx 0.002(^\circ/\text{sec})^2 \tag{4}$$

As you can see, this time we reduced the effective white noise power to the half. Without paying a single cent, we increased the accuracy of our navigation system by simply increasing the sample rate.

Example 4. Suppose we are going to use this IMU configuration (100Hz bandwidth with 100Hz sampling rate) in an aided navigation system with a Kalman filter. What should be the Q matrix value of the <u>continuous time</u> Kalman filter corresponding to the discrete white sequence power in equation (4).

Q matrix value of the continuous time Kalman filter must be equal to the power spectral level of the white noise whose average (i.e. $\frac{1}{T}\int_0^T \omega(\mathrm{t})\mathrm{d}t$) has the same power computed in equation (4). Therefore,

[2]You must note that when you have 5 samples, you basically have 2 different options to run the INS. You may either run the entire INS equations 5 times, or you may take the average of these 5 samples and run the INS with it only once. In general, because of the non-commutativity of the rotations, this 2nd option is theoretically not correct. However, to keep the argument simple, I will simply ignore this point. Besides, the errors of low cost MEMS units are much bigger than the algorithmic error caused by the non-commutativity effects at 10Hz. Therefore, using the simple averages to reduce the INS execution rate can indeed be considered as a realistic approximation for noisy low cost sensors.

it must be:

$$\begin{aligned}\frac{Q(ARW)}{\text{INSPeriod}} &= \sigma^2_{ARW}\\ Q(ARW) &= \sigma^2_{ARW} \times \text{INSPeriod} = 0.002 \times \frac{1}{10}\left(\frac{\circ}{\sqrt{\text{sec}}}\right)^2\\ &= \left(0.014\frac{\circ}{\sqrt{\text{sec}}}\right)^2 \end{aligned} \tag{5}$$

Example 5. It seems computing the average of sensor outputs is a useful thing. Then, why don't we run our INS at 1Hz and compute the average of consecutive 100 samples in the above configuration?

First of all, if you intent to update your INS at 1Hz using simple averages of sensor outputs you will definitely start seeing the effect of insufficient discretization rate due to non-commutativity of the rotations. INS is not a simple integrator. The integral of rotation rate is never equal to your orientation change. If you really want to reduce the computation load of your INS you should implement a 2 speed INS structure.

However, for the time being let us assume that discretization is not an issue. Can we still improve anything by using 1 second averages? The answer is obviously no. The power of additive white noise on 1 sec averages will be:

$$\sigma^2_{ARW} = 0.14^2/100 \approx 0.0002(^\circ/\text{sec})^2$$

However, the Q matrix value will remain the same as shown below:

$$\begin{aligned}Q(ARW) &= 0.0002 \times 1\left(\frac{\circ}{\text{sec}\sqrt{\text{Hz}}}\right)^2\\ &= \left(0.014\frac{\circ}{\sqrt{\text{sec}}}\right)^2\end{aligned}$$

Therefore, for the error covariance point of view, it does not make any improvement as expected.

Example 6. In the specification sheet of ADXRS646 [2], it is indicated that the angle random walk parameter (i.e. the value of the Q) is equal to $0.01\,^\circ/\text{sec}/\sqrt{\text{Hz}}$. (Remember that this is also the value that we used in Example 1). However, now we ended up with the value of $0.014\,^\circ/\text{sec}/\sqrt{\text{Hz}}$. What is the reason for this increase? Did we do anything wrong?

This increase in the Q value is normal. Note that, we are sampling the 100Hz bandlimited signal at 100Hz. However, as you may guess we had better be sampling it around 200Hz to be compatible with the Nyquist theorem. It is this incompatibility that causes this slight increase. To avoid this problem you may either increase the sampling frequency under the same bandwidth or lower sensor bandwidth keeping the sampling frequency same.

You must note that $0.01\,^\circ/\mathrm{sec}/\sqrt{\mathrm{Hz}}$ is our ultimate limit. Our objective is to reach that limit by adjusting the filter bandwidth and the sampling frequency in a proportional manner. If we use a very big bandwidth, we have to use a very big sampling rate to reach the limit which probably wastes microprocessor power. On the other hand, if we choose a very small bandwidth, then we may actually start filtering the motion itself rather than the noise. Therefore, we are trying to find a balance in between to reach $0.01\,^\circ/\mathrm{sec}/\sqrt{\mathrm{Hz}}$ limit.[3]

Example 7 and 8 will make this point clearer.

Example 7. As shown in Example3, as we increase the sampling rate, we can improve the white noise performance. Then, why don't we set the sampling rate to 1KHz and get a fantastic sensor out of a crappy MEMS unit?

We cannot do so, because the sensor outputs are band limited. We are not sampling a white noise but rather we sample the noise after it is filtered by a low pass filter. In the above examples the filter bandwidth was set to 100Hz. When you sample this band limited noise at 1KHz, you can no longer assume that noise on the consecutive samples are independent. Therefore, the average formula that we use in equation (4) can no longer be used to compute the noise power. As a matter of fact, although equation (3) is quite a correct value, equation (4) should only be considered as an approximation (100Hz samples of 100Hz band limited noise cannot not be considered independent theoretically).

As mentioned in the previous example, $0.01\frac{^\circ}{\mathrm{sec}\sqrt{\mathrm{Hz}}}\times \mathrm{SamplingFrequncy}$ is our ultimate limit. We cannot exceed this limit by increasing the sampling rate regardless of the filter bandwidth.

This final observation brings us another important question. What should be the minimum sampling rate that we can use in order to reach this limit given a filter bandwidth? One way to answer this question is to compute it with a pencil on a paper (the way of a decent engineer). However, I feel quite old to do so. Therefore, I prefer to use try and see method like all the other ignorant engineers of the world. The next example will demonstrate how we can use Allan variance curves to determine a proper sampling rate for a given bandwidth with try-and-see method.

Example 8. In this example I am going to use ADIS16407 [3]to demonstrate the effect of sampling rate and sensor bandwidth. In contrast to ADXRS646, ADIS16407 is a digital IMU which only outputs discretized (and digitized) samples. Like all other digital IMUs, its front end consist of an ADC and a digital filter. The ADC parameters are fixed. It uses an anti-aliasing filter with a bandwidth of 330Hz and samples this filtered signal at 819.2Hz. The user cannot change these values. Therefore, the maximum bandwidth of ADIS16407 was fixed to 330Hz. On the other hand, the user can change the bandwidth (or completely cancels it) of the second stage digital filter. ADIS16407 uses a Bartlett window with a user specified tap size for digital filtering.

[3]These conflicting requirements can be easily solved in hardware level if the sensor manufacturers could be smart enough to use analog integrators (or at least discrete counterpart of analog integrators). As a matter of fact, Intersense has already done so in NavChip. Analog Devices's iMems series has a digital integrator in it. However, the wast majority of the sensor manufacturers still rely on the low pass output filters. I guess they expect navigation engineers to think harder in order to cover their own stupidity.

The Analog Devices describes the white noise properties of ADIS16407 using the following the 3 parameters in its specification sheet [3]:

1. Rate noise density: $0.044^\circ/sec/\sqrt{Hz}$
2. Output noise: $0.8^\circ/deg$
3. Angle random walk: $1.9^\circ/\sqrt{Hr}$

In fact, these 3 parameters are closely related and can be computed from one another. However, it is still good to see all these 3 parameters in a specification sheet. Let me first review the meaning of these parameters before actually start dealing with the readout rate for a given bandwidth. [4]

"Rate noise density" is the counterpart of "Angle random walk" parameter of analog sensors. Remember that the angle random walk of ADXRS646 is $0.01\,^\circ/\mathrm{sec}/\sqrt{\mathrm{Hz}}$ [3]. Therefore, by comparing the rate noise density of ADIS16407 (which is 0.044) with the angle random walk of ADXRS646 we can conclude that ADXRS646 is a better gyroscope.[5]

As described above, ADIS16407 uses an ADC to digitize its signal. "Output noise" parameter is the power of white noise on each sampled data. In fact, you may compute this value by yourself:

$$\sigma^2 = 0.044 \cdot \sqrt{330 \times 1.57} = 1.01^\circ/sec \tag{6}$$

where 0.044 is the "rate noise density" and 330 is the bandwidth of the ADC. Note that the specification sheet value of "output noise" is $0.8^\circ/deg$ rather than $1.01^\circ/sec$. The reason for this difference is that equation (6) assumes a 1st order filter whereas ADIS16407 uses a second order filter internally. Therefore, you should definitely use $0.8^\circ/sec$ as the output noise.

$0.8^\circ/sec$ corresponds to the RMS value of the noise power on each sample at 819.2 Hz (the internal sampling rate). Therefore, the angle random walk parameter corresponding to this RMS value can be computed as follows:

$$\begin{aligned} ARW = 0.8 \times \frac{1}{\sqrt{819.2}} &= 0.028\frac{^\circ}{\sqrt{\mathrm{sec}}} \\ &= 1.68\frac{^\circ}{\sqrt{\mathrm{hr}}} \end{aligned} \tag{7}$$

Note that the specification sheet value of ARW is $1.9^\circ/\sqrt{Hr}$. Therefore, there is a difference between the computed value and the specification sheet value. However, this difference is not significant. Besides, in

[4]As far as I know, the terminology they use in their specification sheet does not depend on any published standard. Analog Devices seems to make-up their own terminology for their sensors. Although it is a little bit confusing, it is not hard to guess what each term means in reality.

[5]You should note that the per axis price of ADIS16407 is more expensive than ADXRS646. This is one of the reason why you should prefer analog sensors over digital ones whenever you can. The price that you pay for the digital sensors is for their front-ends which we do not need at all.

equation (7) we assume that each sample is independent, whereas this is not true exactly as the bandwidth of the first stage filter is 330Hz.

If you want to take advantage of full performance of this sensor, you must read its output as 819.2Hz. However, this rate is too much for most microprocessors to handle. Thus, you must use the internal digital filters of the sensor to filter the signal so that you can read the outputs at a slower rate.

Suppose that the maximum frequency content of the motion that this sensor is assumed to expose is limited to 20Hz. Therefore, we can safely adjust the bandwidth of the internal digital filter to 30Hz by setting number of Bartlett taps to 31 (which actually corresponds to $f_{-3db} = 31.71Hz$).

After this lengthy introduction here comes the question: (under the specified conditions) which sampling rate (or more correctly readout rate) should you use in order not to increase apparent ARW value on discrete output samples?

To determine a proper sampling rate, all you need to do is to compute the Allan variance of the sensor outputs with a relatively big sampling period. Such an Allan variance result is shown in figure 1. The sensor data for this figure is collected at 204.9Hz.[6] The blue curve represents the Allan variance of the data when the internal digital filter was set to 31Hz. As you can see from this curve the initial part of the Allan variance contains a correlated noise segment. In fact such initial curvatures are quite common in Allan variance of most MEMS sensors. They appear when the sampling rate is much bigger than the internal filter bandwidth. Also note that the peak value of this initial segment occurs around $\tau = 0.03\text{sec} \approx \frac{1}{30}$. Hence, this example also shows a quick and dirty way of determining the approximate internal bandwidth of the inertial sensors when it is not specified in the specification sheets.

The red curve in figure 1 represents the Allan variance of the same gyroscope without any additional digital filtering (however, note that it is still low-pass filtered by the ADC's anti aliasing filter whose bandwidth is fixed to 330Hz). By comparing the red and blue curves one can easily conclude that output filtering does not affect the average rate outputs of the sensors for averaging durations greater than 0.1seconds. In other words, if we run an INS at 5Hz with both filtered and and unfiltered gyroscope data the performance will be very similar. This clearly shows that, in contrast to the general belief, the primary purpose of the inertial sensors' output filters are not the white noise suppression. Those output filters are there so that we can reduce the sampling rate without affecting the performance.

Let us now see what happens when we reduce the sampling rate by explicitly discarding some samples. For this purpose, I simply downsampled both of the the signals with 4 (i.e. $GyroX = GyroX(1:4:end)$) and then recomputed the Allan variance. The comparison of these Allan variances of both the original and down-sampled signals are presented in figure 2. As you can see from this figure, the Allan variance of the "filtered and downsampled signal" (dashed-blue curve in figure 2) is exactly the

[6]The sensor data was in fact collected at 819.6Hz internally. However, as the USB interface that I used for data recording is not capable of transferring the data at that rate, I used the internal averaging and decimation filters of the ADIS16407 to downsample the data to 204.9Hz before they were transmitted to the computer. As the averaging filters have no affect on the Allan variance results, the results that I obtained at this rate is exactly (both theoretically and practically) equivalent to sampling the data at 819.6Hz. This is one of the reasons why internal averaging filters (i.e. digital counterpart of analog integrators) are so useful for us.

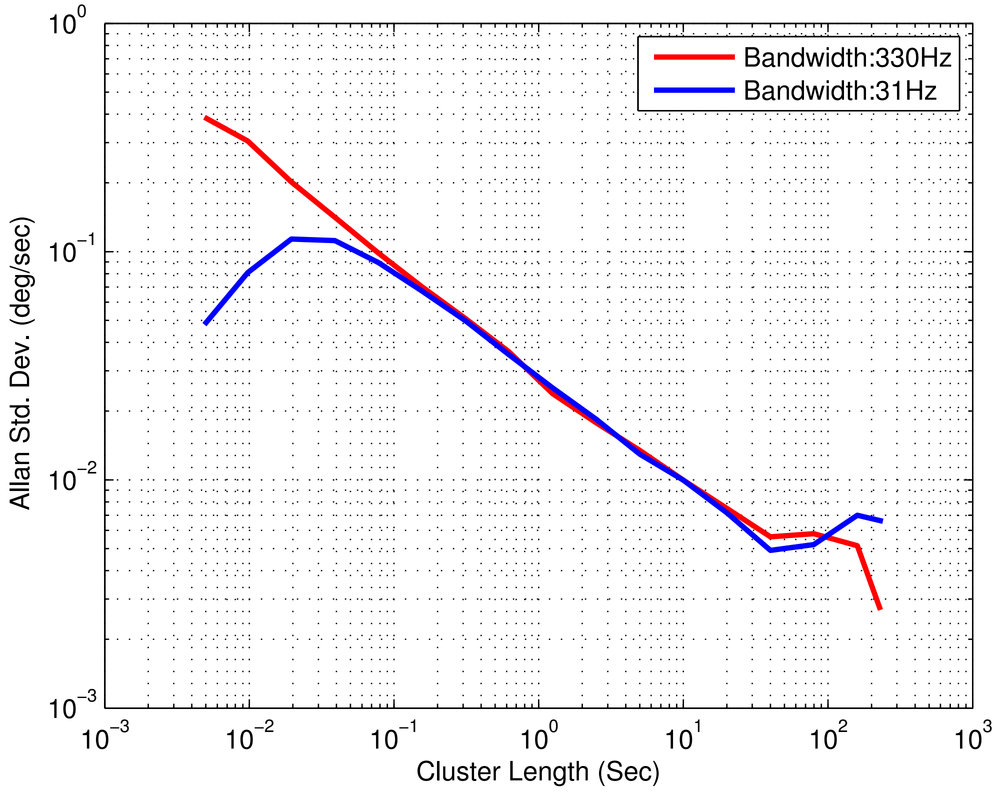


Figure 1: Allan standard deviation curves for ADIS16407 x-gyroscope. The blue curve represents the data when the internal digital filter was set to 31Hz. The red curve is for data which is not digitally filtered.

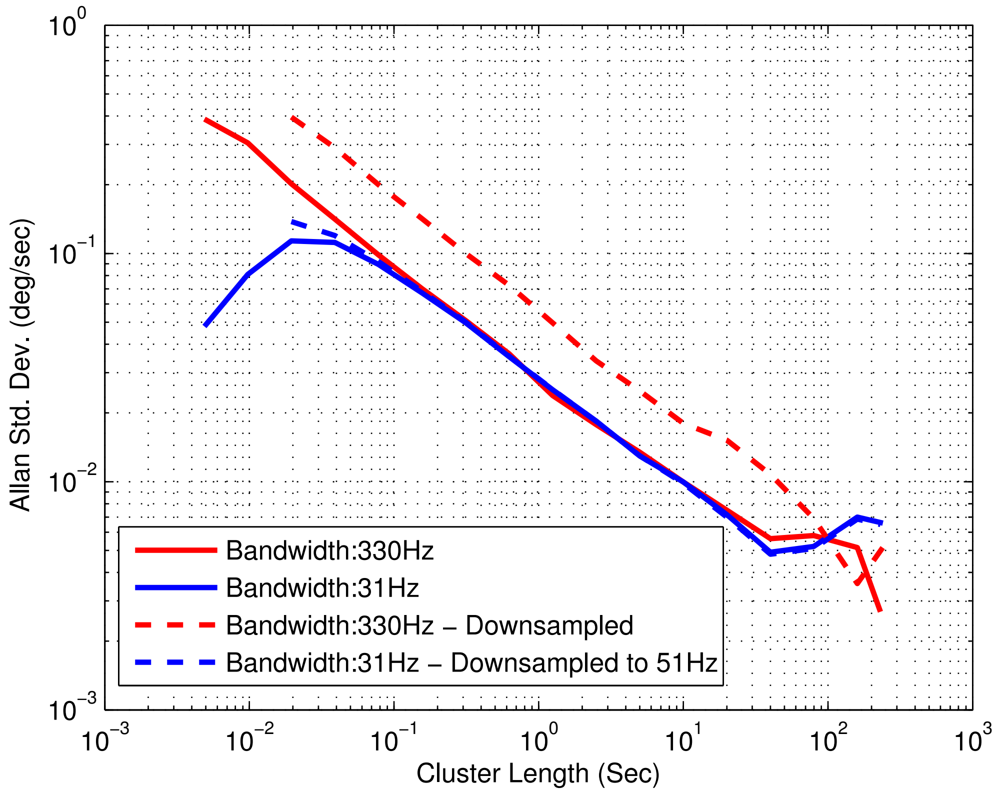


Figure 2: Comparison of Allan variances for down-sampled signals. The dashed curve represent the down-sampled signals and the solid curve represents the original Allan variances.

same as the original signals. On the other hand, the Allan variance of the "unfiltered but downsampled signal" (red-dashed curve in figure 2) is 4 times worse than the original data as expected (note that we downsampled the data with 4. Therefore, the Allan standard deviation is twice the original data.).

This simple example clearly illustrates that if we filter the data at 31Hz, we can simply sample its output at around 50-60Hz without causing any performance degradation. This is generally true, though not always. You can always try and see it by yourself. As long as you sample the signal with a rate approximately equal to the twice of the sensor bandwidth, you will be using your inertial sensors at its full capacity in terms of its additive white noise performance.[7]

The above discussion also shows that we do not have to filter the sensor outputs as long as the sensors can compute the rate averages by themselves. This is because we need only these averages to run our

[7] On the other hand, you must choose a sampling rate much higher than the twice of the motion bandwidth. The twice the bandwidth rule is only valid for the additive white noise performance, not for the effective signal reconstruction.

INS rather than the sampled data as explained in Example 3. The digital sensors of Analog Devices have these averaging/decimation filters in the last stage of their front-ends. Therefore, you can use Analog Devices' sensors without filtering the data at all by simply using these averaging/decimation filters. This is a quite a useful property especially when you need to use these sensors in high-dynamic environment with a low cost microprocessor which does not provide sufficient data throughput.

Example 9. Assume that:

1. We have completely noiseless 3 axis-accelerometer and use ADIS14607 as a 3-axis gyroscope.

2. ADIS14607 does not have any error source other than additive white noise.

3. We configured ADIS14607 to 31Hz bandwidth and record its output at 50Hz.

4. The system was placed on a perfectly horizontal and stationary platform.

Under these assumptions, what will be your position error after 1 minute of INS operation?

First, we need to compute the ARW parameter (PSD level of the white noise which is supposed to drive the continuous form of the error propagation equations). Figure 2 (blue dashed line) represents the Allan variance of the specified configuration. From the value of Allan variance curve at $\tau = 1$, we can see that the ARW is equal to $0.0282^\circ/\sqrt{\text{sec}}$. It should be noted that this value is also equal to the value computed in equation 7 as expected.

Under the specified assumptions, the simplified error propagation model for a single horizontal axis can be written as follows:

$$\begin{bmatrix} \delta\dot{p} \\ \delta\dot{v} \\ \delta\dot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & g \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \delta p \\ \delta v \\ \delta\phi \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \omega(t) \end{bmatrix} \tag{8}$$

where $\delta p, \delta v$ and $\delta\phi$ is x-position, x-velocity and x-tilt angle error respectively. [8] g represents the gravity. $\omega(t)$ is the additive white noise whose spectral level ($Q = \left(0.0282\frac{\circ}{\sqrt{sec}}\right)^2$) is called as angle random walk parameter. The solution of this differential equation (Eq. 8) is as follows:

$$\begin{bmatrix} \delta p(t) \\ \delta v(t) \\ \delta\phi(t) \end{bmatrix} = \begin{bmatrix} 1 & t & \frac{gt^2}{2} \\ 0 & 1 & gt \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \delta p(0) \\ \delta v(0) \\ \delta\phi(0) \end{bmatrix} + \begin{bmatrix} \frac{gt^2}{2} \\ gt \\ 1 \end{bmatrix} \omega(t)$$

[8]You should note that you cannot call $\delta\phi$ as roll or pitch error without knowing the true heading angle. The definition of roll and pitch angle errors are strictly coupled with the heading angle. For instance, x-axis tilt error will be equal to pitch error if and only if the system is aligned with local geodetic axis. If the system is 90° rotated on the horizontal platform than the x-tilt angle error will be equal to roll error. This is one of the peculiarity of large heading angle errors.

As we assume that the initial errors are all zero, we can compute the variance of the state vector as follows:

$$E\left\{\begin{bmatrix}\delta p(t)\\ \delta v(t)\\ \delta\phi(t)\end{bmatrix}\begin{bmatrix}\delta p(t)\\ \delta v(t)\\ \delta\phi(t)\end{bmatrix}^T\right\}=\int_0^t\int_0^t\left(\begin{bmatrix}\frac{g(t-\tau)^2}{2}\\ g(t-\tau)\\ 1\end{bmatrix}\underbrace{E\{\omega(\tau)\omega(\lambda)\}}_{Q\delta(\tau-\lambda)}\begin{bmatrix}\frac{g(t-\lambda)^2}{2}\\ g(t-\lambda)\\ 1\end{bmatrix}^2 \mathrm{d}\tau\mathrm{d}\lambda\right)$$
$$=Q\begin{bmatrix}\frac{g^2t^5}{20} & \frac{g^2t^4}{6} & \frac{gt^3}{6}\\ \times & \frac{g^2t^3}{3} & \frac{gt^2}{2}\\ \times & \times & t\end{bmatrix}$$

Therefore, after 1 minute, the variance of the total position error will be:

$$E\left\{\delta p_{Total}^2\right\}=2E\left\{\delta p^2\right\}=2\cdot\left(\frac{0.0282\pi}{180}\right)^2\cdot\frac{9.81^2\cdot 60^5}{20}$$
$$\approx 1812\mathrm{m}^2$$

As a result, the standard deviation of total position error after 1 minute will be equal to 42m under the specified conditions.

4 Flicker Noise

This noise is called as bias instability in [1]. However, several other factors also cause bias to vary in low cost MEMS units. Therefore, in my opinion, bias instability should not be used to define this specific form of noise for MEMS units as this leads to wrong understanding of the nature of bias variations.

A noise component whose power spectra is proportional to f^{-1} is called as flicker noise. This noise manifests itself as a horizontal (constant with 0-slope) segment in the Allan variance curves (see figure 3). Because of this 0-slope Allan variance property of the flicker noise, most navigation engineers perceive flicker noise in a substantially wrong manner. In this section, I briefly summarize some key properties of the flicker noise to correct the general misconceptions about it.

Let me start this section with some simple questions that you may encounter in many places.

Example 10. Let us assume that we apply a 2-position calibration test for the gyroscope whose Allan variance is shown in figure 3. Furthermore, also suppose that the rotation rate during the calibration is $150^\circ/sec$. What is the variance of the computed bias and scale factors if the calibration data was collected for 200 seconds at each position.

- Wrong Answer:

  Let $y_1$and $y_2$represents the 200 seconds averages. According to the figure 3, the standard deviation of

$y_1$and y_2 is 0.006$^\circ/sec$ (the value of the red-dashed line). Hence, using a white noise analogy:

$$\sigma_{bias} = std(\frac{y_1+y_2}{2}) = \sqrt{\frac{0.006^2+0.006^2}{4}} = 4.2^{-3}$$

$$\sigma_{scale} = std(\frac{y_1-y_2}{2\times 150}) = \frac{1}{2\times 150}\sqrt{0.006^2+0.006^2} = 2.8^{-5}$$

- Correct Answer:

$$\sigma_{bias} = \infty$$

$$\sigma_{scale} = \frac{1}{2\times 150}\sqrt{2\times \sigma^2_{Allan}(\tau = 100)} = 2.8^{-5}$$

Example 11. What happens to the bias/scale factor variances if we increase the averaging durations or collect data more than once at each position (e.g. collect data twice for +150$^\circ/sec$ and 5 times for -150$^\circ/sec$)?

- Wrong Answer: The variances are reduced.
- Wrong Answer: The variances remain the same.
- Correct Answer: The variances increase.

Example 12. The gyroscope bias estimate is just the average of the sensor outputs when the gyroscope is placed horizontally pointing to the east. Suppose, I just place my gyroscope properly and start computing the average for the whole day. Furthermore, suppose also that the sensor does not contain any noise component other than the ARW and flicker noises. Under these conditions, how does this average change during the day?

- Wrong Answer: Eventually the average value converges to a value which we call as the real bias.
- Wrong Answer: The Allan variance of flicker noise is constant. Therefore, the average value fluctuates between (bias$\pm$0.006) during the day.
- Correct Answer: The average value fluctuates with an increasing variance. Therefore, the more we wait, the more diverse values we obtain as bias.

If you can answer the above questions correctly, you may skip this section. Otherwise, you had better keep reading.

The flicker noise is defined based on the shape of its power spectral density. On the other hand, power spectral density is defined for only wide sense stationary signals in the standard texts. Therefore, one may

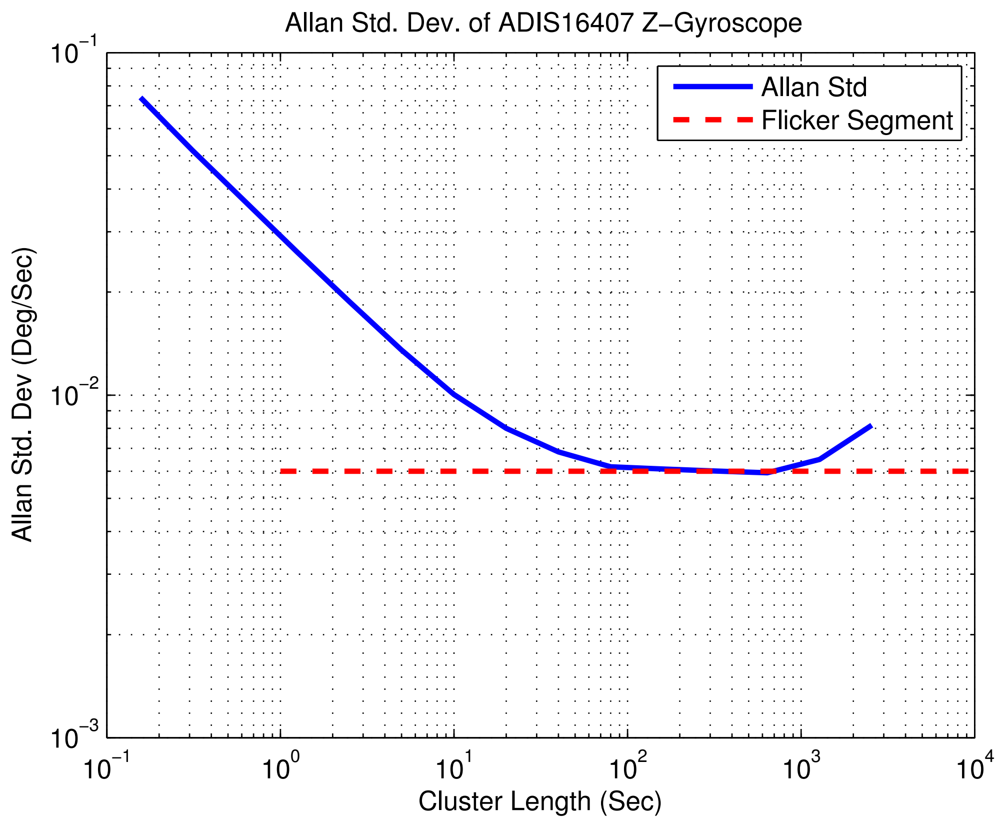


Figure 3: Allan standard deviation of ADIS16407 z-gyroscope. The red-dashed curve represent the flicker noise.

wrongly think that the flicker noise is a stationary signal. However, as it will be clear later in this section that it is a non-stationary signal whose spectra is defined only for a limiting sense (limit of a certain averaging operation). It is this non-stationary characteristics of flicker noise that makes the definition of "bias" theoretically not possible.

As a matter of fact, the non-integrable nature of f^{-1} spectra also implies that the spectra itself can only be defined for a limiting sense too. Similar to the spectra of a random walk process, the integral of the flicker noise spectra over a finite band near DC (0-frequency) is infinite. That is why, its autocorrelation does not converge to 0 as $\tau \to \infty$. In other words, the value of the process at $t = \infty$ is correlated with its value at $t = -\infty$. Furthermore, unlike the random walk (but similar to the white noise), $\int_{f_c}^{\infty} S_x(f)df$ is also infinite for any cut-off frequency. Therefore, the discrete time equivalent of a flicker noise cannot be defined without considering the effect of some virtual cut-off frequencies for the sampling operation.

4.1 Some identities regarding the flicker noise

Despite its non-integrable feature, we can still derive some useful properties for the flicker noise from its very simple spectral definition. In this section, two of such properties of the flicker noise are presented.

Let the spectra of flicker noise be represented as follows:

$$s_x(f) = \frac{h}{2|f|}, \quad -\infty < f < +\infty$$

Furthermore, let us also assume that the transfer function of the sampler (i.e. ADC) is as follows:

$$h_s(t) = \frac{1}{T_s} rect(t, T_s)$$

$$H_s(\omega) = e^{-\frac{j\omega T}{2}} sinc(\frac{\omega T}{2})$$

where,

$$rect(t,T) = \begin{cases} 1 & 0 < t < T \\ 0 & else \end{cases}$$

4.1.1 Power of variation between 2 samples

Let y_1 and y_2 be 2 samples of flicker noise ($x(t)$) separated by T seconds. In this case, the difference of these samples can be defined as:

$$\begin{aligned} y(t) &= y_1(t) - y_2(t) \\ &= x(t) * \underbrace{(h_s(t) - h_s(t-T)}_{h(t)} \end{aligned}$$

where $x(t)$ is he flicker noise and "$*$" represents the convolution operation. Thus, the spectra of $y(t)$ can be computed as follows:

$$\begin{aligned} S_y(f) &= H(f)S_x(f)H^*(f) \\ &= 2h\frac{sin^2(\pi f T_s)}{(\pi f T_s)^2}\frac{sin(\pi f T)}{f} \end{aligned} \tag{9}$$

Equation 9 is absolutely integrable and thus it is a spectra of a stable and stationary process. Therefore, the power of $y(t)$ is equal to the integral of equation (9):

$$\sigma_y^2 = \int_{-\infty}^{+\infty} s_y(f)\mathrm{d}f$$

Computing the indefinite integral of eq.9 is something that exceeds my basic level (and ashamedly poor) calculus knowledge. However, note that the definition that I used as the sample difference is in fact equal to what is defined as "2 sample difference variance with dead time (T)" in general texts on the frequency stability such as [4]. In the appendix of [4], it is shown that (though not derived) the integral of equation (9) is equal to:

$$\begin{aligned} r &= \frac{T}{T_s} \\ \sigma_y^2 &= h[-2r^2 ln(r) + (r+1)^2 ln(r+1) + (r-1)^2 ln(r-1)] \end{aligned} \tag{10}$$

Note that when $T = T_s$ the power of the difference of the samples is equal to the (twice) of the Allan variance. Therefore, by setting r to 1 in equation (10) we can also compute the Allan variance:

$$AllanVar(T) = h \cdot 2ln(2) \tag{11}$$

As seen in equation (11), the Allan variance value is independent of time T (or T_s). Therefore, the Allan variance is a constant (0-slope) segment as indicated at the beginning of this section.

For all practical purposes equation (11) can be approximated as follows:

$$\sigma_y^2(T) \approxeq h \cdot ln\left(\left(\frac{T}{T_s}\right)^2\right) + 4h \cdot ln(2) \tag{12}$$

Equation 12 is important as it reveals the following observations:

1. As $T \rightarrow \infty$, the variance between 2 samples (σ_y^2) also goes to infinity. This proves that, the variance of the flicker noise is time variant and divergent.

2. Thus, we cannot determine the variance of bias as indicated in the example 10. Each sample of sensor output contains both bias and flicker noise which theoretically has infinite variance. The average of two random variables one of which has infinite variance must be infinite as indicated in the correct answer of example 10.

3. Equation 12 also contains the term T_s which is kind of an artificial parameter introduced to define a proper sampling operation. Because of the non-integrable characteristics of the flicker noise, it is not possible to derive the noise powers without such parameters. However, trying to determine an accurate T_s value for a real sensor is not practical either. Therefore, equation (12) should be used only in a relative sense.

4.1.2 Power of flicker noise average

In the previous section, we only analyzed how the sensor output changes in time. However, as explained in the example 12, the bias estimate is computed as the average of the sensor outputs in a given interval. The diverging noise power does not mean that the power of its average also diverges to infinity. Therefore, examples 11 and 12 has yet to be answered.

In order to avoid infinite sample variance problem, we define the average as the output of the following filter (see also figure 4):

$$h_{av}(t) = \frac{1}{T_{av}} rect(t, T_{av}) - \frac{1}{T_s} rect(t - T_{av}, T_s)$$

The output spectra of this filter when the input is a flicker noise can be computed as follows:

$$\begin{aligned} y_{av}(t) &= h_{av}(t) * x(t) \\ S_{y_{ac}}(\omega) &= |H_{av}(\omega)|^2 \underbrace{S_x(\omega)}_{\frac{h\pi}{\omega}} \\ &= \frac{h\pi}{\omega}\left[sinc^2\left(\frac{\omega T_{av}}{2}\right) + sinc^2\left(\frac{\omega T_s}{2}\right) - 2sinc\left(\frac{\omega T_{av}}{2}\right) sinc\left(\frac{\omega T_s}{2}\right) cos\left(\frac{\omega(T_{av}+T_s)}{2}\right)\right] \end{aligned} \tag{13}$$

Equation 13 is absolutely integrable. Therefore, $y_{av}(t)$ is stable and its variance is equal to the integral of equation 13. Computing the indefinite integral of equation (13) is something that again far exceeds my

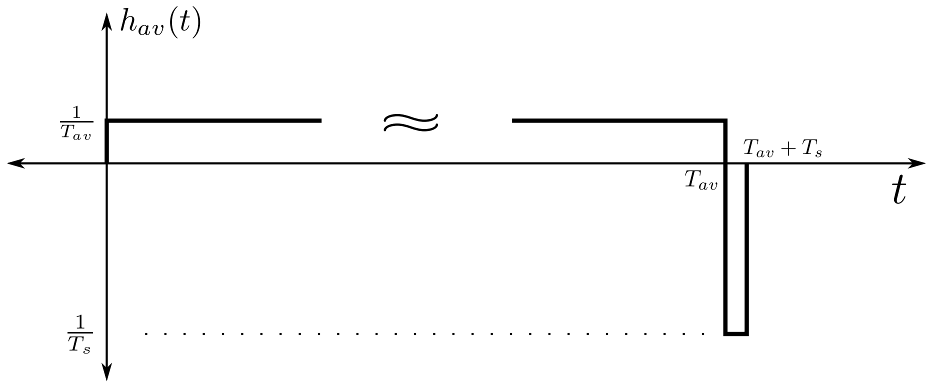


Figure 4: Averaging filter. The 2nd rectangular window is used to represent the initial sample to remove from the average in order to avoid infinite power problem.

calculus knowledge. However, it can be shown by simulations that the integral value of equation (13) can be approximated for $T_{av} >> T_s$ as follows:

$$\begin{aligned}\sigma^2_{y_{av}} &= \frac{1}{2\pi}\int_{-\infty}^{+\infty} S_{y_{av}}(\omega)\mathrm{d}\omega \\ &\approx h\cdot ln\left(\frac{T_{av}}{T_s}\right)+b \qquad T_{av} >> T_s \end{aligned} \tag{14}$$

where b is a constant which is <u>not</u> closely related with the Allan variance value. It should be noted that the rate of change of variance in equation (14) is half of the equation (12). This reduction comes from the fact that equation (14) represents the variance of the average whereas equation (12) represents the variance of a sample.

Using equation (14), I can now answer the example 11 and 12 in a more convincing way:

1. According to equation (14) as the averaging time increases, the variance of the average value also increases without a bound. This proves that the averaging operation does not converge to a fixed number. As a result, as you increase the averaging duration for the sensor calibration, the variation of the so-called "bias" estimate will also increase as indicated in the example 12.

2. From the "bias" estimation point of view, computing the averages at different positions is nothing but a fancy way of increasing the total averaging duration. (Note that you do not need to perform 2 position calibration in order to estimate the bias. Just making sure that the input is zero is sufficient to compute it.) Therefore, as the number of positions (or the number of repetitions for the same calibration procedure) is increased, the variation of the "bias" estimates will also increase as indicated in example 11.

Now that you know what happens as the averaging duration increases, you should read once more the existing literature about the sensor calibration. You will be amazed by the number of ignorant engineers who claim in their “peer reviewed” papers that they obtain better calibration results by using more than 6 positions. It is indeed true that using more than 6 positions is useful (and in fact is necessary) for most of the low cost MEMS units. However, the reason for this is not to improve the calibration results but to be able to estimate more number of calibration parameters which cannot be observed with a 6-position test. Unfortunately, those engineers (and the referees of their papers) are not even aware of this fact. Yet, I am sure that all of them identifies themselves as INS experts without any hesitation.

4.2 Simulating the flicker noise

In order to perform Monte Carlo tests on the “brand new” navigation algorithms we have to simulate the flicker noise of the inertial sensors (as well as the other noise components). Based on the previous discussion, you may have anticipated that simulating the flicker noise must be one of the hardest things in the Monte Carlo tests. However, in contrary to this, it is in fact quite simple. In his interesting paper [5] Keshner sketches a very easy method for simulating the $\frac{1}{f}$ noise in the frequency region of interest. According to [5], an approximate $\frac{1}{f}$ spectra can be generated by cascading 1 pole/zero pair for each decade of the frequency of interest. If you think this approximation in terms of the bode plots of pole/zero pairs (whose zeros are bigger than poles), it makes perfect sense. In the following paragraphs, I will present one such example to clarify his method.

Example 13. Generate a signal whose Allan standard deviation is equal to 0.06 between $\tau = 1$ and $\tau = 100$.

In order to generate it we have to generate a signal whose spectra is approximately equal to $\frac{0.06}{\sqrt{2ln(2)}}\frac{1}{2f}$ between $f = 0.01$ and $f = 1$. The code sample presented in the Listing 1 computes the pole and zero locations of the transfer function whose spectra is approximately proportional to $\frac{1}{f}$ for $f_{min} << f << f_{max}$. As can be seen from this code listing, we first divide the specified frequency into $4 \times$ nstage equal logarithmic steps. The parameter “nstage” represents the number of cascaded 1st order zero/pole stage that we would like to use for the frequency region of interest. As indicated in [5], only 1 pole/zero pair is sufficient for each decade. If a better approximation is needed, more number of stages can be used. However, in this case the approximation around the frequency limits deviates more. (Therefore, only 1 stage must be used when $\frac{f_{max}}{f_{min}} \approx 10$.)

In order to compute the parameters of the system who has the aforementioned spectral properties, I called the function in the Listing 1 with the following parameters:

$$[\mathrm{k,p,z}] = \mathrm{flickgen}\left(\frac{0.06^2}{2ln(2)}, \quad 0.001 \cdot 2\pi, \quad 10 \cdot 2\pi, \quad 4 \right) \tag{15}$$

As can be seen from the arguments, I specified the limits a decade wider than the original ones in order to minimize the distortion effects around the actual limits. Furthermore, I also used $\log(\frac{10}{0.001}) = 4$ stages.

Listing 1: Matlab code to compute the pole/zero locations of the system whose spectra is approximately equal to $\frac{1}{f}$ between ω_{min} and ω_{max} radians.

```
function [gain, pls, zrs]=flickgen(h, wmin, wmax, nstage)
  wstep=(wmax/wmin)^(1/(4*nstage));
  wmid=sqrt(wmax*wmin);
  pls=zeros(nstage,1);
  zrs=zeros(nstage,1);
  magmid=1;
  for i=1:nstage
    pls(i)=-(wmin*wstep^(4*i-3));
    zrs(i)=-(wmin*wstep^(4*i-1));
    magmid=magmid*(wmid^2+zrs(i)^2)/(wmid^2+pls(i)^2);
  end
  gain=sqrt(h*pi/wmid/magmid);
```

The return values of this function correspond to the following system model:

$$H(s) = 0.0114\frac{(s-0.0353)(s-0.3533)(s-3.5333)(s-35.3329)}{(s-0.0112)(s-0.1117)(s-1.1173)(s-11.1733)} \tag{16}$$

The power spectral density of equation (16) is shown in figure 5. As can be seen from this figure, the spectra of the system closely approximates the $\frac{1}{f}$ curve between $0.01 < f < 1$Hz. A better approximation could have been obtained if more than 4 stages had been used. The Allan standard deviation of the same system is presented in figure 6 which has a relatively straight segment between $1 < \tau < 100$ seconds. The initial descending segment of the Allan curve represents the effect of generated white noise. As the number of poles and zeros in equation (16) is the same, the system has a direct input-output coupling. Therefore, the output of the system contains not only the flicker noise but also a white noise component. If this part is not desired for any reason, then the last zero in equation (16) should be removed. To obtain a better approximation in the Allan variance domain, the frequency limits used in equation (15) should be increased.

4.3 Modeling the flicker noise in the navigation Kalman filters

Although the flicker noise can be successfully simulated using the method described in the previous section, we use another approach to model the flicker noise in the Kalman filters. The previous method is based on a higher order system model excited by a single white noise. Instead of using such higher order models, we prefer to approximate flicker noise as a sum of 1st order Markovian models in the Kalman filters just for the practical reasons. In the following paragraphs, I will first present the justification of this method. Then, I will show the application of this approach on a real sensor example.

First, it should be noted the spectra of the flicker noise can be represented as the weighted sum of the

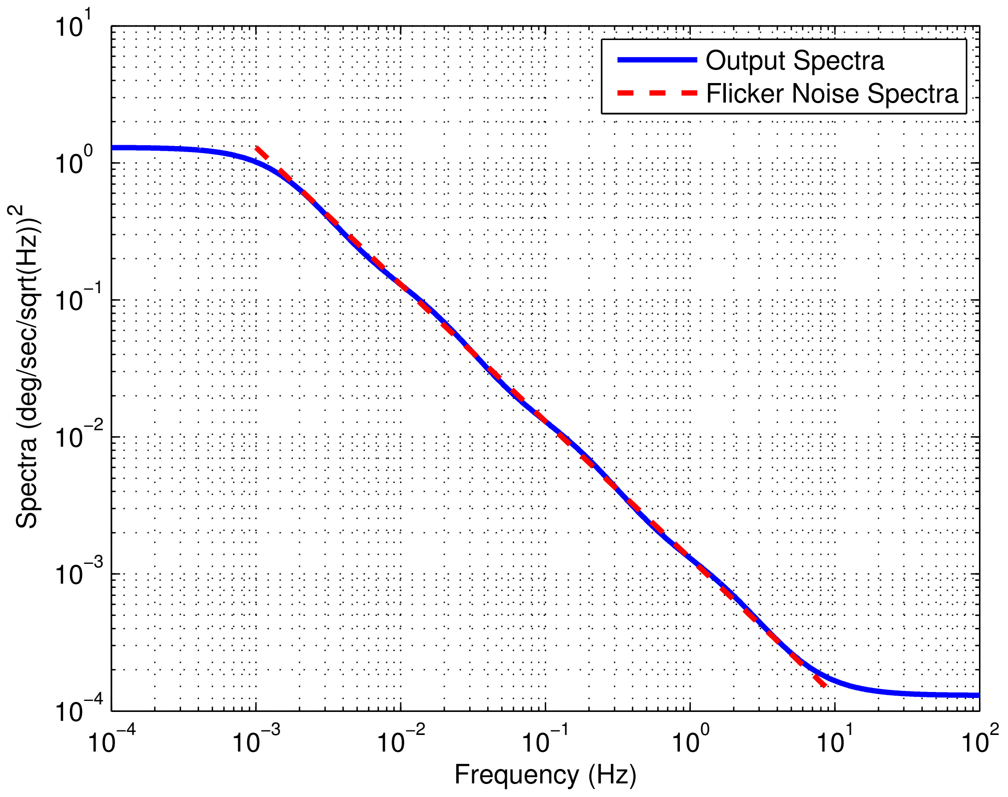


Figure 5: The output spectra of the system defined in equation (16). The red-dashed line represent the theoretical spectra of a flicker noise.

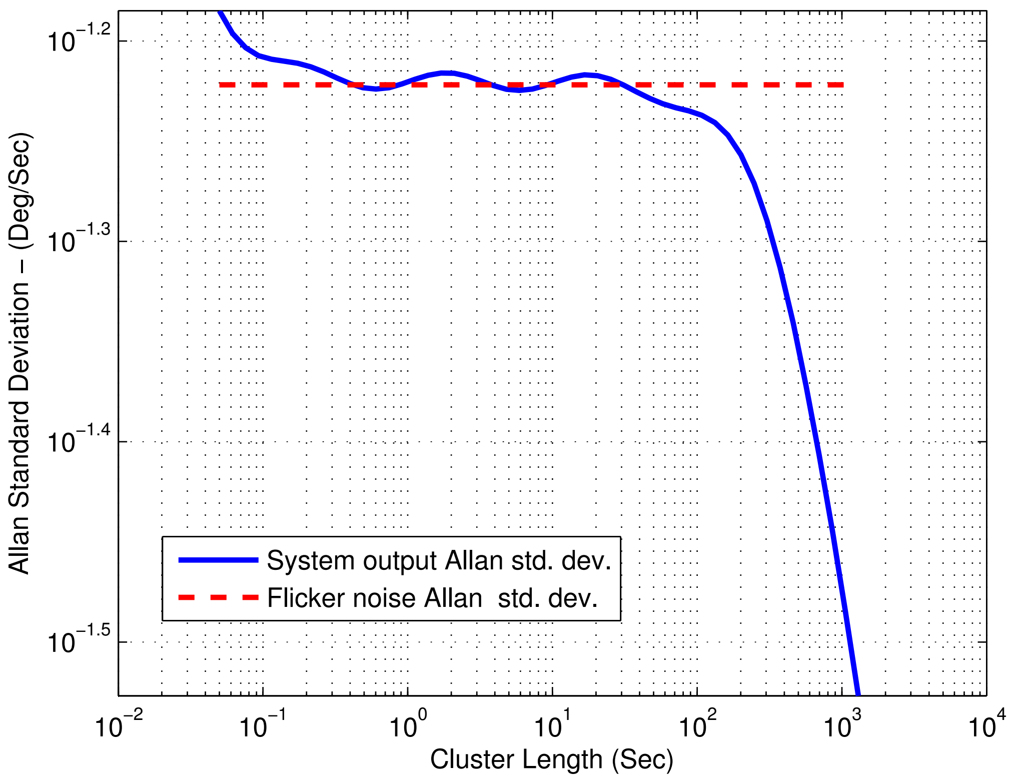


Figure 6: Comparison of Allan standard deviations of the specified system in equation (16) and a flicker noise with $h = \frac{0.06^2}{2ln2}$.

infinitely many exponentially correlated processes as shown below:

$$\begin{aligned}\int_0^\infty \frac{2h}{\omega^2+\alpha^2}\mathrm{d}\alpha &= \frac{2h}{\omega}tan^{-1}\left(\frac{\alpha}{\omega}\right)|_0^\infty \\ &= \frac{h}{2f}\end{aligned} \tag{17}$$

The integral in equation (17) can be approximated as a Riemann sum. However, this causes each term to have different power which is something that we do not prefer. Therefore, we first apply the following change of variable in equation (17):

$$10^\beta = \alpha \Rightarrow \frac{h}{2f} = \int_{-\infty}^{\infty} \frac{2h10^\beta ln(10)}{\omega^2+10^{2\beta}}\mathrm{d}\beta$$

We can now approximate the above integral as an infinite sum:

$$\frac{h}{2f} = \lim_{\Delta\beta\to 0}\sum_{k=-\infty}^{\infty}\frac{2h10^{k\Delta\beta}ln(10)\Delta\beta}{\omega^2+10^{2k\Delta\beta}}$$

Each component in the summation term corresponds to the spectra of the following process:

$$\dot{x(t)} = -10^{k\Delta\beta}x(t) + \sqrt{2h10^{k\Delta\beta}ln(10)\Delta\beta}n(t)] \tag{18}$$

where $n(t)$ is a unity power white noise. The time constant of this process is equal to $10^{-k\Delta\beta}$ and its steady state power is:

$$\sigma_x^2 = hln(10)\Delta\beta$$

As can be seen from this final expression, each process in the summation term has exactly the same power. This proves that we can express a flicker noise as the sum of infinitely many 1st order Markov processes with the same steady state power.

From the discussions of the previous section, we know that the spectra of the flicker noise can be roughly approximated by cascading 1st order systems which are separated by a decade in the frequency domain. If we set $\Delta\beta = 1$ in equation (18), we can obtain a similar separation. Under this condition, the peak value of the Allan standard deviation of each model in equation (18) will be equal to:

$$\begin{aligned}\sigma_{Allan}(\tau_{\text{peak}}) &= 0.437\sqrt{2}\sqrt{ln10}\sqrt{h} \\ &= 0.9742\sqrt{h}\end{aligned} \tag{19}$$

At the same time, the Allan standard deviation of the flicker noise with a spectra $\frac{h}{2f}$ is equal to $\sqrt{h2ln2} = 1.174h$. This value is roughly equal to equation (19). This verifies that approximating a flicker noise as a sum of the 1st order Markov processes with the following properties is a reasonable approach:

1. Each processes is approximately separated by a decade
2. The peak Allan variance value of each process is approximately equal to the Allan variance of actual flicker noise.

Example 14. The Allan standard deviation curve of z-gyroscope of ADIS16407 is presented in figure 7 as the blue solid line. Determine an approximate model to be used in the Kalman filter for the flicker noise of this gyroscope.

As can be seen from figure 7-(A), the flicker noise of this sensor spans from $50 < \tau < 2000$. Therefore, 2 Markov processes are needed to approximate it. Based on visual inspection, I decided that the peak Allan standard deviation values of these 1st order Markov processes should be at:

$$\left\{ \begin{pmatrix} \tau_{\text{peak}} & \sigma(\tau_{\text{peak}}) \end{pmatrix}_i \right\} = \left\{ \begin{pmatrix} 75 & 0.00525 \end{pmatrix}, \begin{pmatrix} 1320 & 0.0063 \end{pmatrix} \right\}$$

From these peak value coordinates, the parameters of the 1st order Markov process in the form of $\dot{x(t)} = -\frac{1}{T_c}x(t) + q_c n(t)$ can be computed as follows[1]:

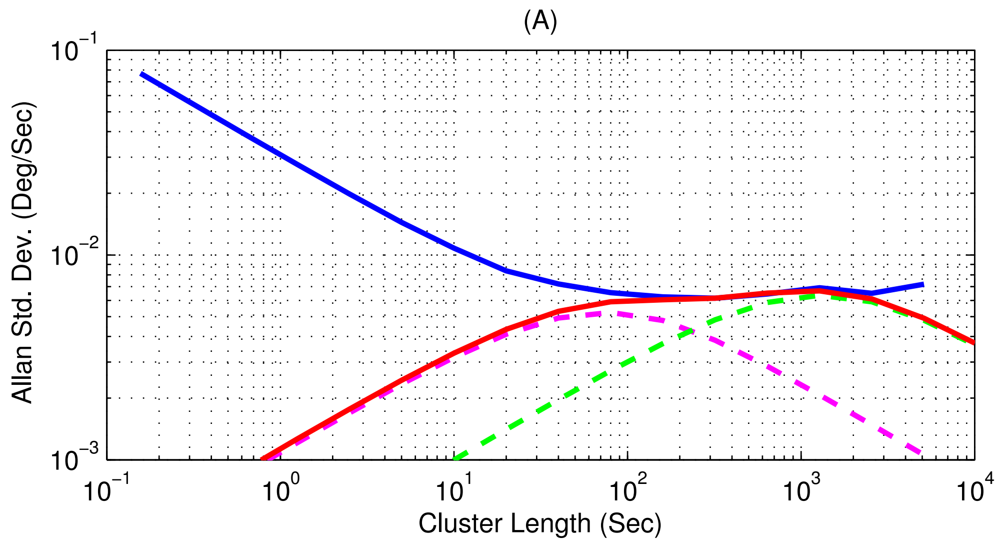
$$T_c = \tau_{\text{peak}}/1.89$$
$$q_c = \frac{\sigma(\tau_{\text{peak}})}{0.437\sqrt{T_c}}$$

In figure 7-(A), the theoretical Allan standard deviations of each these 1st order Markov processes are shown as magenta and green curves. The red curve in the same figure represents the sum of these two Allan standard deviations. The comparison of the red curve and the blue curve (which is the Allan standard deviation of the actual gyroscope output) shows a good match. Therefore, these two Markov processes can be used as the model of the flicker noise in the Kalman filter as long as the maximum operation period of the INS is limited to 20 minutes.

The initial part of the Markov process approximation in the figure 7-(A) (red curve) may seem disturbingly different from the actual Allan values (blue curve). However, when an angle random walk process is added to the model, we obtain the the red-dashed curve in the figure 7-(B). As can be seen from this theoretical Allan standard deviation of the complete model, the specified Markov processes are indeed sufficient to describe the flicker noise in the observed data period.

5 Quantization noise

All MEMS inertial sensors are rate measuring sensors. They physically sense acceleration and rotation rate. Most of the low cost sensors contain a digital front-end both to sample these rate signals in discrete time and



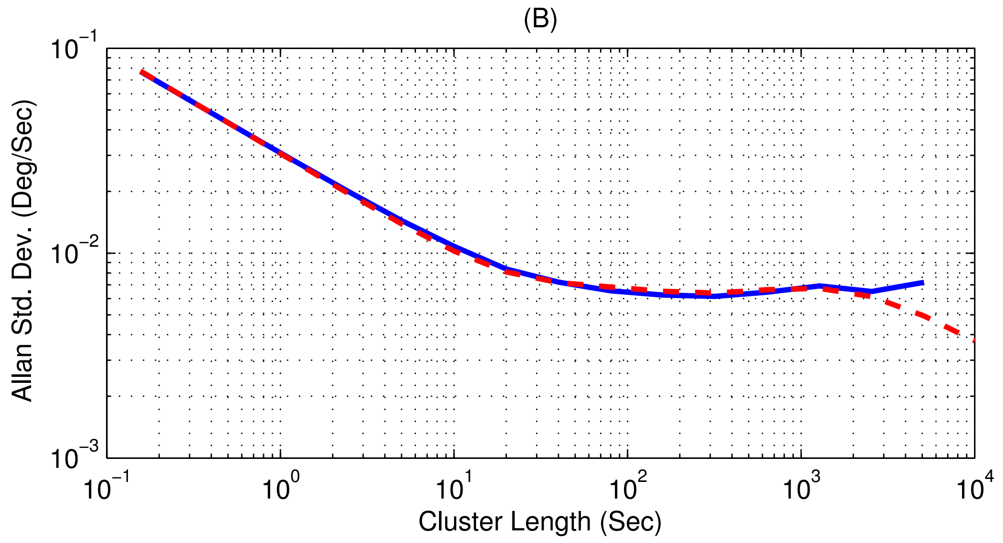


Figure 7: Allan standard deviation curve of ADIS16407 z-gyroscope. (A): Allan standard deviations of the 1st order Markov processes (green and magenta curves) together with the actual data (blue curve). (B): Comparison of actual Allan standard deviation (blue curve) with the fitted model (red curve).

to digitize them. Because only a limited number of bits are used in the digitization process, these low cost sensor outputs naturally corrupted with finite bit quantization effects.

In the documents related with the Allan variance such as [1], there is a section describing the Allan variance of the so called “Quantization” noise. Just because of its name, most people (including the ones who shamelessly call themselves as INS experts too) believes that the Quantization noise in those documents is the same as finite bit Quantization effects that we observe at the outputs of low cost MEMS units. However, in reality, they are two completely different quantization noise effects that has no relation at all with each other.

The quantization noise described in [1] is a kind of time quantization which appears at the output of rate integrating sensors (although there is no such terminology in the existing literature, I will use the term time-quantization to refer to this type of quantization in the rest of this document). The most distinctive feature of this type of quantization noise is that it does not accumulate. In other words, let us assume that $\{y_1, y_2 \ldots, y_n\}$ represents the discrete (and also digitized) outputs of a rate integrating sensor and each sample contains a digitization error δy_i with a maximum power of Δ. In this case, the maximum power of the quantization error on the sum of all outputs ($Y = \sum_{i=1}^{n} y_i$) will also be equal to Δ (i.e. it does not accumulate regardless of the summation).

On the other hand, this is not the case at all for the bit quantization error of MEMS units. For those units, the total quantization error power of the sum of the samples (i.e. Y) can be as much as $n\Delta$. If the bit quantization error is assumed to be white, than the power will be $\sqrt{n}\Delta$ which again grows unbounded as the number of samples increases.

As can be seen from this simple explanation, the difference between the bit-quantization of the rate sensors and the time-quantization of the rate-integrating sensors are striking. Even if the per sample quantization powers for the rate and rate-integrating sensors are the same, their effects on the computed navigation solution is completely different. As analyzed in detail in [6], the time-quantization errors do not have any significant impact even for the high precision INSs. Therefore, even if we observe a very dominant “-2”-segment in the Allan standard deviation curve, we can completely (and safely) ignore this type of quantization errors.

In contrast to time-quantization effects, handling the bit-quantization errors is much more difficult, if not possible at all. First of all, you must note that bit-quantization errors do not appear on the Allan variance curves by any meaningful and identifiable sense. At best (if we are lucky), we do not see their effects at all regardless of the actual magnitude of the bit-quantization. (Remember that you can always observe -2-slope segment curves for time-quantization effects if you can arbitrarily increase the sampling period.) This is because, Allan variance data are collected when the sensor is stationary. Therefore, the effect of the bit-quantization on the input signal will be a random constant and the Allan variance is invariant to random constant errors. However, the bit-quantization also affects the noise output. Therefore, what we can actually compute is the Allan variance of the quantized noise rather than the real noise. One may argue that as too many samples are used during the computation of the Allan variance values, the effect of quantization will disappear due to over-sampling. However, the experiments reveal that we cannot rely on this logic at all. One

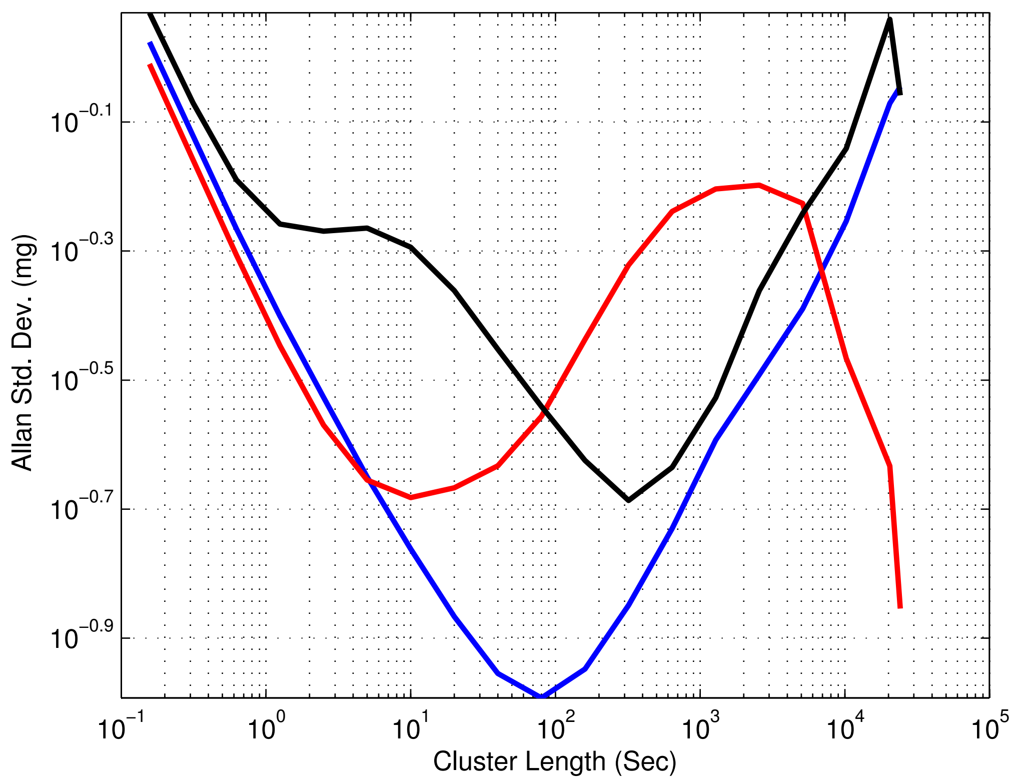


Figure 8: Allan standard deviation of ADIS16407 accelerometers. Each curve belongs to a different axis. The data for these curves are collected at 6.4Hz by setting the decimation(averaging) factor to 128 in the output registers.

such example is shown in figure 8. In this figure the Allan variance of the all 3 accelerometers of ADIS16407 is presented. According to its specification sheet [3], the maximum range of these accelerometers is ±18g and each acceleration sample is digitized with 14bit. Therefore, the bit resolution is 3.33mg (e.g. when $0 \leq a < 3.33$mg the output will be 0). As can be seen in figure 8, the Allan variance of the collected sensor data turns out to be anything but useful under such a big bit resolution.

Regardless of the huge inconsistencies between the curves in figure 8, it is still possible to get some values regarding the additive white noise from the initial "approximately" -1-segments. According to these figures the VRW values of the sensors are equal to $0.38\frac{mg}{\sqrt{Hz}}$which is somehow close to specification sheet value of $0.5\frac{mg}{\sqrt{Hz}}$. However, one must be extremely careful in using these VRW values in the navigation Kalman filters. As an example, suppose that we decided to run our INS at 5Hz. In this case we would expect the power of additive white noise on each accelerometer sample to be equal to $0.38 \times \sqrt{5} = 0.89mg$. However, this value would be totally wrong because we know that on each sample there is $3.33mg$ quantization error. Therefore, if the IMU is used under a dynamic environment, the power of additive white noise must be set to at least 3.33mg rather than 0.89mg. On the other hand, if the IMU is stationary (for instance during the initial leveling phase), a random constant with a initial standard deviation of 3.33mg must be explicitly considered during the covariance calculations.

As a matter of fact such huge quantization values as in the case of ADIS16407 accelerometers are quite problematic to theoretically deal with. Although rate sensors can be quite easily converted to rate integrating sensor with a suitable analog front end (all is needed an analog integrator stabilized with a digital feedback), the MEMS producers for some reason do not consider that alternative at all. They just implement the most useless front-end and expect their user to deal with all kind of problems caused by it. So far, the only company that I know of providing integrated rate output (i.e. angle and velocity increments) is Intersense. I have recently seen some examples of NavChip of Intersense and I was quite impressed with its performance. It seems the motion cube is a better sensor than all others in the same price range.

Finally, you should note that despite the huge bit-quantization error of the accelerometers, the dominant error source of ADIS16407 is still its gyroscopic errors. The effect of the accelerometer quantization errors on the final positional accuracy is much (and much) smaller than that of its gyroscope bias stability. Therefore, even if you would like to use that sensor in any 6DoF project, you do not have to bother yourself too much with the accelerometer quantization errors.

6 Sinusoidal Errors

All MEMS units internally operates around a resonant frequency (especially the gyroscopes which have to vibrate/rotate a mass to measure the Coriolis force). Because of this, every MEMS sensor output are significantly contaminated by kind of pseudo-deterministic sinusoidal components. In most cases (at least for the sensors that are designed by decent engineers), these sinusoidal components are filtered out by the output stage filters. In fact, that is one of the reasons why every MEMS unit must always be used with some kind of low pass filter regardless of the sampling frequency.

However, there are sensors on the market whose low frequency outputs also contain significant amount of sinusoidal noise components. As the frequency of those components are in the range of the motion frequency it is impossible to filter them with an output stage filter. Therefore, they significantly affect the navigation accuracy.

In general, sinusoidal noise components manifest themselves as quite weird bulges on the Allan variance curves. The exact shape of the Allan variance of a deterministic sinusoidal error is presented in [1]. However, as both the leading and trailing edges of the Allan variance of the sinusoidal signals decays fast, you should not expect to see such a curve in your Allan variance figures of the real sensors. When the sensors are contaminated with a sinusoidal component, you can at most observe some very strange additional bulges on either -1 (ARW) or 0 (Flicker Noise) segments. As a matter of fact, whenever you observe some irregularity

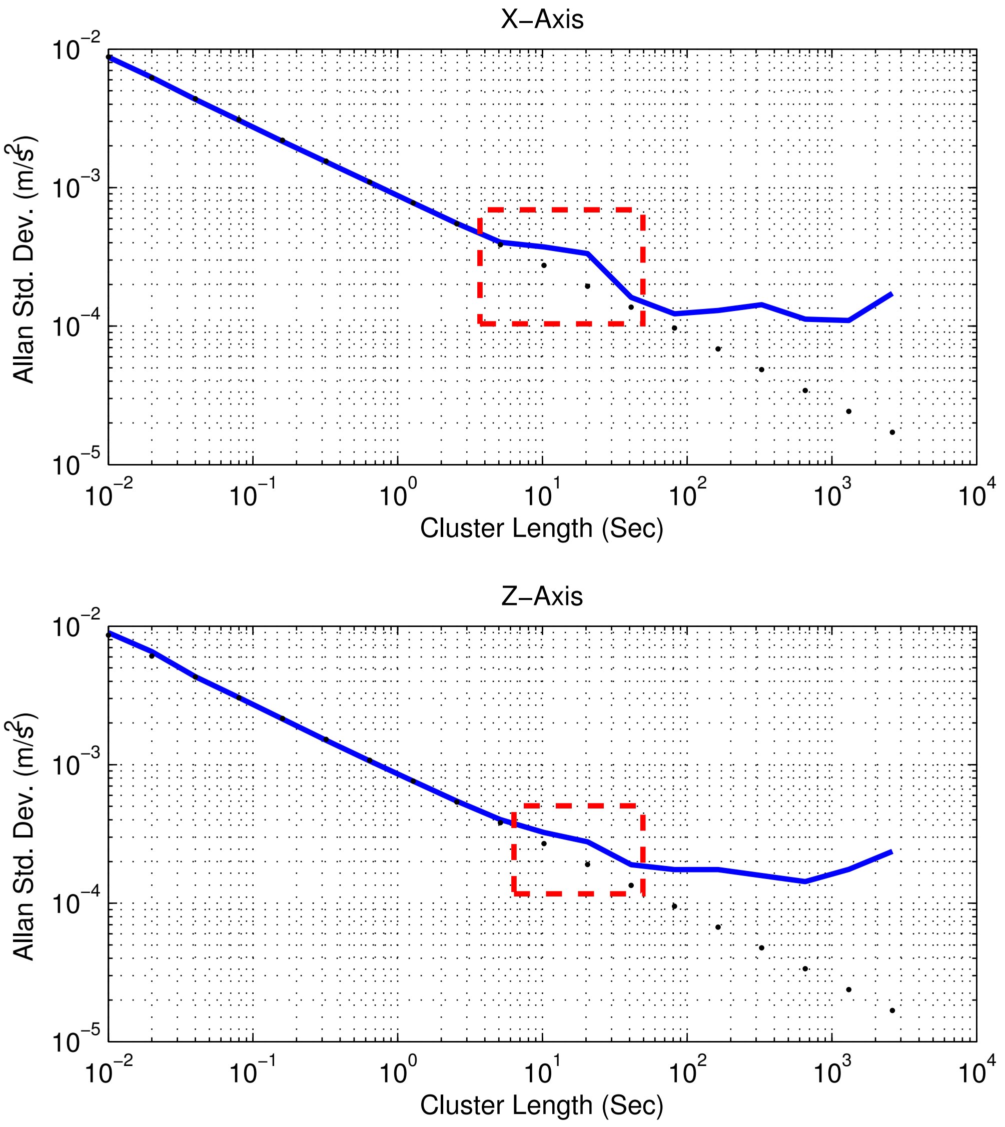


Figure 9: The Allan standard deviation of 3-axis LIS3L02AL ([7]) accelerometer. The blue solid lines are the Allan curves. The black dots are the VRW segments with a slope of -1. The red rectangles shows the locations where the sinusoidal components appear on the Allan standard deviation values for x and z axes.

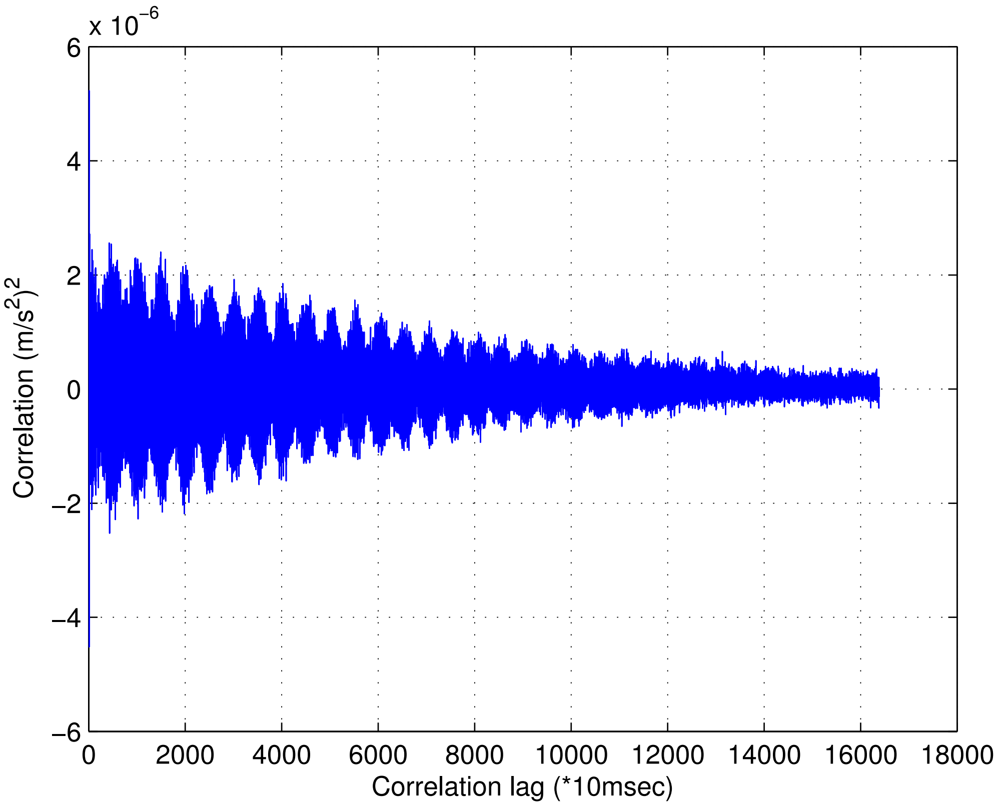


Figure 10: (Modified) Autocorrelation of LIS3L02AL z-axis accelerometer whose Allan standard deviation is presented in figure 9.

especially on the V/ARW segment, you must always suspect a possible sinusoidal noise component regardless of the shape of it. A clear example of such a situation is shown in figure 9 where the Allan standard deviations of an LIS3L02AL ([7]) accelerometer are presented.[9] The red rectangles in this figure indicates the region where we can observe the aforementioned irregularities. As the irregularity is only very slight in the z-axis of figure 9, one may suspect whether or not that axis contains a sinusoidal error as claimed. The (modified) autocorrelation of the same z-axis accelerometer data is presented in figure 10. As can be clearly seen in this (modified) autocorrelation figure, the output does certainly contain a damped sinusoidal error.

Unfortunately, sinusoidal errors cannot always be detected from the Allan variance curves. When the frequency of the sinusoidal error is close to the sampling frequency, the additive white noise usually hinders the effect of those components on the Allan variance values. An example of this situation is presented in

[9]This is a very old accelerometer from STM which is now obsolete. I have not used any STM accelerometer since then. Therefore, I do not know whether or not the STM's sensors still suffer from the same problem.

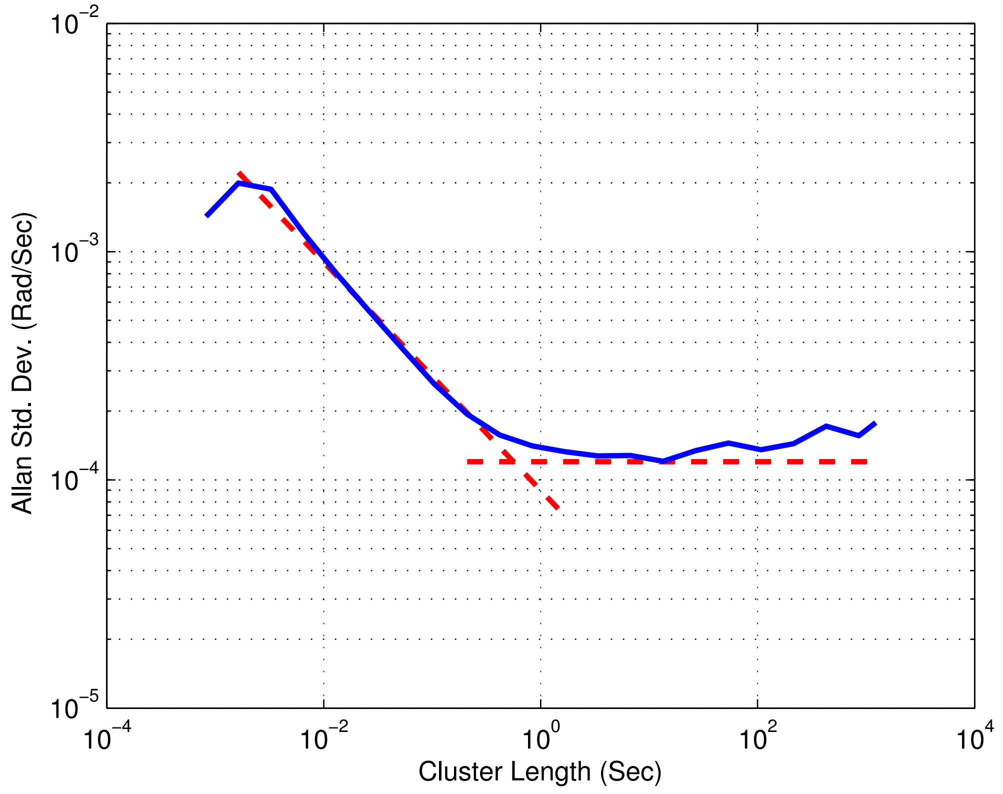


Figure 11: Allan variance of y-gyroscope of IMU3000 (blue solid curve). The red dashed segments represents the standard angle random walk and flicker noise segments.

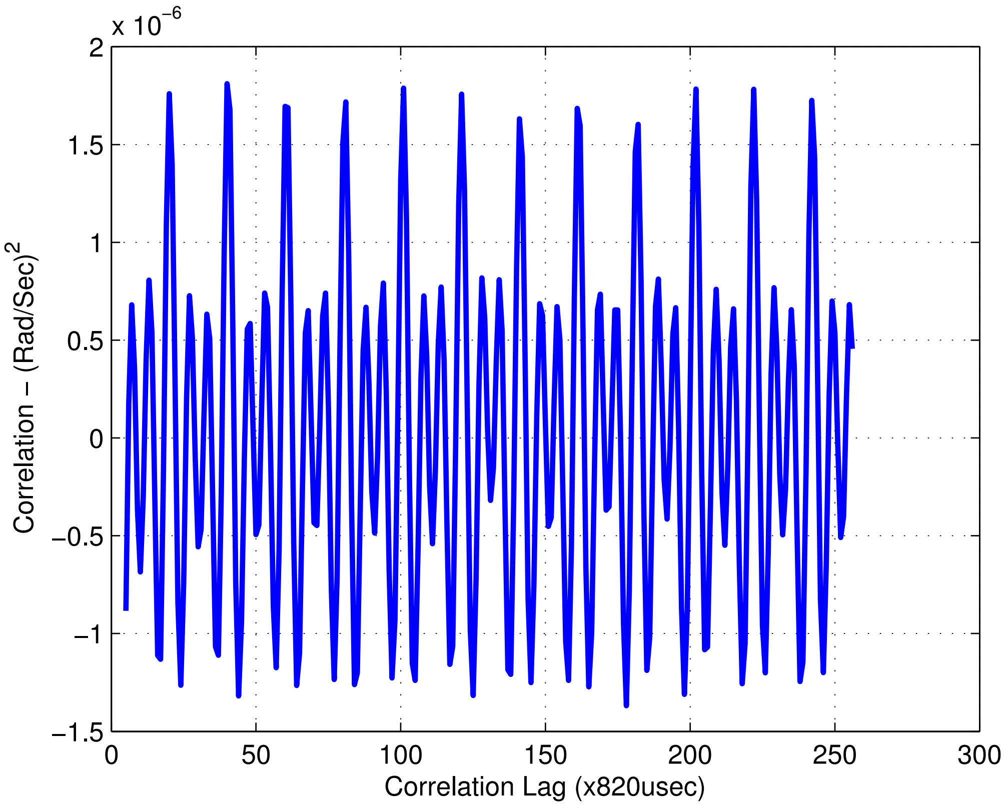


Figure 12: (Modified) Autocorrelation of the y-gyroscope of the IMU3000

figure 11. In this figure the Allan variance of IMU3000 ([8]) is shown together with the standard ARW and flicker noise segments. A close visual inspection does not reveal anything unusual about this curve. The initial curvature is a result of the output stage filters.[10][11] As described in Section 3, these curvatures appears when the sampling frequency is set higher than the bandwidth of the output filters. We can also observe a regular ARW and a flicker noise segments on this Allan curve. The curve itself does not seem to contain anything unanticipated.

On the other hand, you can see the (modified) autocorrelation of exactly the same data in figure 12.

[10]In fact, there are some irregularities in this initial segment. If it were only due to the output filter, we would expect the trailing edge to closely follow the -1 segment. As seen in figure 11, this is not the case. However, we cannot be sure because the donkeys working in Invensense (not Intersense) do not specify which kind of filter they use in their sensors. Therefore, it is hard to conclude whether this irregularity is due to the filtering or not.

[11]Invensense is the most ignorant and disgusting inertial MEMS producer that I have ever seen in my whole life. They do not hesitate to lie in their web site about their sensor performance just to be able to promote themselves. It is exactly because of these type of manufacturers that the inertial sensor industry fails to answer our (navigation engineers) demands. I witnessed a couple of people who trusted the Invensense, and started their designs with their sensors, and eventually failed in reaching satisfactory navigation performance even under significant external aiding.

As this figure clearly reveals, the output of this sensor is significantly corrupted by a powerful sinusoidal component. This example clearly proves that you must always check for sinusoidal errors regardless of whether or not you detect anything unusual in the Allan variance curves. Allan variance is a great tool for “power law” noises. However, when it comes to other type of noises (such as sinusoidal errors or correlated noises with a very big correlation time) it usually fails to produce satisfactory results. Therefore, you must always cross check your results with another modeling method and verify that your Allan variance based models are complete. For this purpose, I usually use the correlation based modeling method described in [9].

If you try to compute the autocorrelation of raw sensor data, you will not be able to see anything other than some pseudo triangular waveforms which do not resemble to any kind of Markovian process correlation. The reason for this is that the raw autocorrelation waveforms are dominated by the very slowly changing components. As the observation duration is smaller than the possible time constant of these slowly changing components, the autocorrelation of raw sensor outputs is almost arbitrary. To obtain some meaningful data from the autocorrelation, these slowly changing components must be stripped off. The autocorrelations presented in Figure 10 and 12 are computed after such a preprocessing (that is why, I referred them as modified autocorrelation). The details of this autocorrelation based inertial sensors analysis can be found in [9].

7 Rate random walk errors

Rate random walk error is the real random walk type error on rate outputs of the inertial sensors (i.e. rotation rate and acceleration). As the additive white noise components are called as “angle random walk” (Section 3), it is customary to specifically use the term “rate random walk” to denote this error component.

There is nothing special about this error which manifest itself as “+1 segment” in the Allan standard deviation curves. However, in most cases, you will not be able to observe such a perfect +1 segment. The Allan variance method is especially insufficient to model highly correlated noises such as random walk which theoretically has an infinite time constant. In order to be able to observe them on the Allan variance curves, we have to collect a very large data set. However during long data collection periods, the sensor outputs are inevitably affected by the ambient temperature which usually spoils the RRW segments on the Allan variance curves. That is the reason why in most cases you would end-up with segments whose slope substantially differs from +1.

If you can observe a consistent +1 curve, it is very easy to compute the disturbance noise power (i.e. the power of $\omega(t)$ in the random walk process $\dot{x(t)} = \omega(t)$) (See [1]). However, if you observe an arbitrary segment with a positive slope which is not equal to 1, then what should you do? In this case, you should select a value ($[\sigma_{\text{Allan}}(\bar{\tau}), \bar{\tau}]$) on this curve and then use ([1]- Eq.C7) to compute the disturbance power. Although this seems seems too arbitrary, it has a certain logic behind it. In this way, you are approximating a highly correlated noise (whose nature is not know) with a random walk process whose $\bar{\tau}$-seconds integral power is equal to $(\bar{\tau}\sigma_{\text{Allan}}(\tau))^2$.

In order to use a random walk model in a Kalman filter, we also have to specify its initial power (i.e. $E\{x^2(0)\}$). In general, one may argue that equation (12) can be used for this purpose. When T is set to the amount of time since the sensor "bias" is estimated, equation (12) gives a rough approximation of bias variation power. However, it must be noted that equation (12) can be assumed valid only if:

1. Ambient conditions does not affect the sensor outputs
2. The sensor was not go through any power cycle (i.e. it is not turned off/on since it is calibrated)

It is obvious that this 2nd condition cannot be met at all. If fact, this condition by itself has quite interesting outcomes. It is known that the flicker noise is ultimately related to the memory of excited electrons. During a power cycle, this memory changes, but it does not become completely independent to the past values at all. That is the reason why we measure similar sensor bias values even if the the sensor goes through a power cycle. However, as there is no theoretical or empirical expression for these power cycle effects, we had better not trust on equation (12) as the initial power of random walk.

The most realistic solution for the determination of initial random walk power is to perform repeated calibration tests. The observed variations on the bias estimates should then be used as $E\{x^2(0)\}$ in the Kalman filters. Although performing repeated calibration tests under a variety of different environmental conditions takes too much time (and not to mention incredibly boring), unfortunately there is no other solution.

In contrast to low cost MEMS units, the high-end inertial sensors usually do not contain any significant random walk error. Even if they contain a random walk error, it becomes significant only after a very long continuous operation. Therefore, we almost never use an additional random walk components in the navigation Kalman filters of high-end sensors. The affect of initial bias variations (due to the flicker noise) is accounted by slightly increasing the initial power of 1st order Markov processes which are used to model the short term flicker noise effects.

Although this approach works very well for high-end inertial sensors (I do not know any single example which use another approach for high-end systems), it usually does not work effectively for low cost MEMS units. As can be seen in figure 11, the flicker noise of low cost MEMS units starts dominate in a very short time (1 second in figure 11). The time constant of the 1st order Markov process used to approximate this flicker noise will also be very small for this gyroscope. Regardless of which value is used as the initial power of this approximation, the prediction step of the Kalman filter will reduce it to the steady state power in a very short time (also it will reduce the state value to zero). Therefore, if the initial bias uncertainty cannot be estimated in a very short time, the filter will not be able to consider it properly at all for the rest of the navigation duration.

That is why, in many cases, the most practical way of modeling sensor noise of low cost units is to use only a random walk process to model both the flicker and the real random walk errors. That is the reason why many people can claim to design an INS without knowing any single thing that has been described in this report. At the end of the day, the low cost MEMS units are so erroneous that it is usually impossible to assign some reasonable error models. The only thing that we can do is to use a random walk process and hope for the best. As every idiot can do that, every idiot can claim to design an INS without knowing anything about the sensor modeling.

8 Rate ramp

[1] describes a deterministic rate drift as the rate ramp errors. Just because [1] mentions a rate ramp, most inexperienced engineer tend to consider any segment with a slope greater than 1 as rate ramp. However, you should never do that. You must never see a rate ramp in the Allan variance of low cost units. If you see it for some reason, you must be sure that:

1. You properly compensate for the temperature as described in Section 2. (with a chance of %99, this is the reason for the rate ramp)
2. You have a stable power source.
3. Your sensor readout circuit is working properly.

After checking everything, if you are still sure there is a rate drift, then you must change your calibration model and recalibrate your unit with a time dependent calibration model.

Under no circumstances, you should try to assign a rate ramp model to any positive slope segment in the Allan variance of low cost MEMS units. If for some reason, it consistently appears on the Allan variance curves (and you are sure that you compensate for the temperature), you should consider throwing that sensor away and use some other inertial unit.

This does not mean that you should not have any positive slope segment in the Allan variance. When you see such a segment, you should model it as a Random walk after you are sure that there is no rate drift regardless of the slope of the segment.

Part III
Conclusions

Modeling inertial sensors are the first and the most difficult step in the INS designs. Despite its importance, it seems that not many people really understand what stochastic modeling is. There exist significant amount of study in the existing literature. However, they are all a primitive the restatement of the very basic facts that are already described in [1]. Standards such as [1] are specifically written for a high-end inertial sensor. Although the same (or at least very similar) terminology are used for the error models of both the high-end sensors and the low cost MEMS units, one should never blindly assume that these different classes of sensors have similar characteristics.

This technical report summarizes the major stochastic error models of low cost inertial sensors. I intentionally tried to avoid repeating anything that has been already described in [1]. Instead of a theoretical discussion on the error models, I focused on the interpretation of the existing error models for the low cost MEMS units. I believe several real sensor examples presented in this document will not only clear any confusion about the definition of the error models but also will teach new engineers how to use the error model parameters in any INS design task.

Stochastic error modeling is only one part of the low cost inertial sensor characterization. In general, the performance of high end inertial sensors can be completely characterized by these stochastic error models. However, low cost units have a huge repeatability problem. Stochastic error models are obtained from stationary data. But the error characteristics of low cost units abruptly changes under different environments. Even worse, limited previous laboratory tests had convinced me that the output characteristics may change in time without any apparent reason even the environmental conditions are kept constant. Therefore, the repeatability characteristics of low cost MEMS units must also be analyzed using vast amount of calibration tests. In this document, repeatability characteristics are briefly analyzed under the flicker noise concept. However, one should not rely on those analysis in low cost MEMS units.

References

[1] “Ieee standard specification format guide and test procedure for single-axis interferometric fiber optic gyros,” *IEEE Std 952-1997*, p. i, 1998.

[2] “Adxrs646: High stability, low noise vibration rejecting yaw rate gyroscope,” Sep 2011.

[3] Analog Devices, *ADIS1607: Ten Degrees of Freedom Inertial Sensor*, rev.c ed., October 2011.

[4] J. A. Barnes, A. R. Chi, L. S. Cutler, D. J. Healey, D. B. Leeson, T. E. McGunigal, J. A. Mullen, W. L. Smith, R. L. Sydnor, R. F. C. Vessot, and G. M. R. Winkler, “Characterization of frequency stability,” *Instrumentation and Measurement, IEEE Transactions on*, vol. IM-20, pp. 105 –120, may 1971.

[5] M. Keshner, “1/f noise,” *Proceedings of the IEEE*, vol. 70, pp. 212 – 218, march 1982.

[6] P. G. Savage, “Analytical modeling of sensor quantization in strapdown inertial navigation error equations,” *Journal of Guidance, Control, and Dynamics*, vol. 25, no. 5, pp. 833–842, 2002.

[7] STMicroelectronics, “Lis3l02al mems inertial sensor: 3-axis - +/-2g ultracompact linear accelerometer,” May 2006.

[8] Invensense, “Imu-3000 motion processing unit product specification rev 1.3,” 05 2011.

[9] "Error modeling and characterization of environmental effects for low cost inertial mems units," pp. 598 –612, may 2010.