Introduction to GPS M-Code Signals for Onboarding of Navy Personnel

Jordan Hanson
Assistant Professor of Physics and Computer Science
Department of Physics and Astronomy, Whittier College
Summer Research Fellow with the Office of Naval Research

August 1, 2022

Outline

Outline

Introduction to GPS M-Code Signals - Six Easy Pieces

1. Radio transmission equation and signal strengths
2. Signals: amplitude versus time
3. More on binary signals
4. Power spectral densities
5. Mixing signals, carrier frequencies
6. Auto-correlation functions

Synthesis: Putting the pieces together

Radio transmissions and signal strengths

1. Problem statement
2. Derivation of Friis transmission equation
3. Practical examples (interactive)
4. Application to GPS signals

Radio transmissions and signal strengths

Problem Statement

Given how far away a radio transmitter is, and the transmitter and receiver antenna characteristics, how do we predict the received signal strength?

Variables we need to understand:

- Distance between radio TX and RX: R
- *Gain* of radio TX and RX, G_t and G_r
- *Wavelength* of radio signal: λ
- Transmitted and received power: P_t and P_r

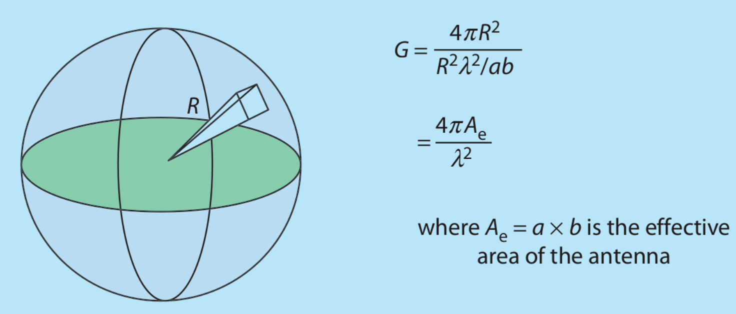


Figure 1: Adapted from Chapter 8 of *Introduction to Airborne Radar*, by Stimson, Griffiths, Baker, and Adamy. SciTech Publishing (2014).

How to translate the idea in the diagram into a formula for the gain:

$$s = r\theta \tag{1}$$

$$\theta_1 \approx \lambda/a \tag{2}$$

$$\theta_2 \approx \lambda/b \tag{3}$$

$$s_1 \approx R\lambda/a \tag{4}$$

$$s_2 \approx R\lambda/b \tag{5}$$

$$G = (4\pi R^2)/(s_1 s_2) = (4\pi R^2 ab)/(R^2\lambda^2) \tag{6}$$

$$A = ab \tag{7}$$

$$G = (4\pi A)/(\lambda^2) \tag{8}$$

Antenna Gain

The **gain** of a radio antenna with aperture efficiency ϵ, effective area A, radiating at a wavelength λ is

$$G = \frac{4\pi A}{\lambda^2}\epsilon \tag{9}$$

Note that the wavelength λ and frequency f are related by the *speed of light:* $f = c/\lambda$.

For radio waves in the atmosphere and space, $c \approx 0.3$ m/ns, or 0.3 km/μs.

Radio transmissions and signal strengths

Gain is typically quoted in a unit called a **dBi**: a decibel relative to isotropic sources. The decibel is a type of logarithmic unit widely used in RF fields.

$$G_{dBi} = 10 \log_{10}(G) \tag{10}$$

Interactive Question: Suppose we have a radio transmitter equipped with an antenna that operates at a wavelength of 10 cm, and an effective area of 40 cm by 40 cm. What is the gain in dBi, if the efficiency at this wavelength is 70%?

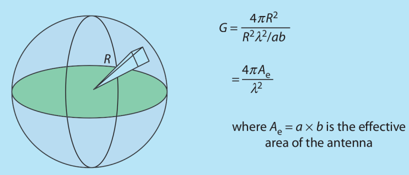
- A: 10 dBi
- B: 13 dBi
- C: 21.5 dBi
- D: 23 dBi

Radio transmissions and signal strengths

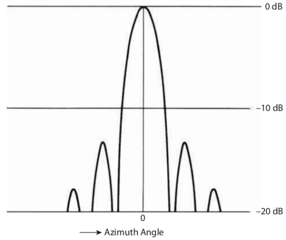
Now we can use the concept of *gain* to understand received power by a receiving antenna (RX) from a transmitting antenna (TX).

Variables we need to understand:

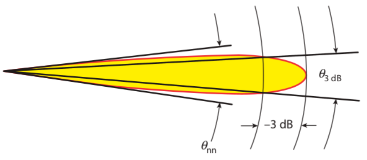
- Distance between radio TX and RX: R
- *Gain* of radio TX and RX, G_t and G_r
- *Wavelength* of radio signal: λ
- Transmitted and received power: P_t and P_r
- Power density: $P_t/(4\pi R^2)$



(a) The concept of *gain*.



(b) The concept of *radiation pattern* (dB) vs. angle.



(c) The concept of *radiation pattern*, visualized in 2D.

Figure 8-1. This three-dimensional plot shows the strength of the radiation from a pencil beam antenna.

(d) The concept of *radiation pattern*, visualized in 2D.

Radio transmissions and signal strengths

Now we can use the concept of *gain* to understand received power by a receiving antenna (RX) from a transmitting antenna (TX).

Suppose we have a TX transmitting P_t, received by RX with P_r. The transmitted power density will be $p = P_t/(4\pi R^2)$, augmented by G_t (Eq. 11). When p arrives at the RX, it will be collected over the A of the RX (Eq. 12). Changing variables from A to G_r (Eq. 13), we arrive at the Friis transmission formula (Eq. 14):

$$p = P_t/(4\pi R^2)G_t \tag{11}$$

$$P_r = P_t/(4\pi R^2)G_t A \tag{12}$$

$$A = \frac{G_r \lambda^2}{4\pi} \tag{13}$$

$$\frac{P_r}{P_t} = \frac{G_t G_r \lambda^2}{(4\pi R)^2} \tag{14}$$

Radio transmissions and signal strengths

Radio transmission signal strength

According to the Friis transmission formula, the ratio of received to transmitted radio power is

$$\frac{P_r}{P_t} = \frac{G_t G_r \lambda^2}{(4\pi R)^2} \tag{15}$$

Each gain factor includes non-ideal behavior due aperture efficiencies.

Interactive questions: (1) What is the RX power in dB? (2) What gain is necessary to achieve specified RX power?

Interactive question (1). For the system in the prior interactive question, imagine that the TX and RX are similar antennas separated by $R = 900$ m. We have $\lambda = 0.1$ m, $G_t = 13$ dBi, and $G_r = 13$ dBi. What is P_r/P_t in dB? This is known as *path loss.*

- A: -25 dB
- B: -75 dB
- C: -125 dB
- D: -175 dB

Interactive question (2). For the system in the prior interactive question, imagine that the design needs to change such that the overall path loss is now -60 dB. We cannot change R or λ, so all we can do is boost the gain. What should the new gain be, in dB, for the antennas?

- A: 5 dB
- B: 10 dB
- C: 20 dB
- D: 60 dB

Radio transmissions and signal strengths

Application to GPS signals: large path losses. In Fig. 3, three reference TX send signals to one RX. Knowing the signal strength of each signal constrains (via a system of equations) the location of the RX. The trouble is the distances represented by L1, L2, and L3.

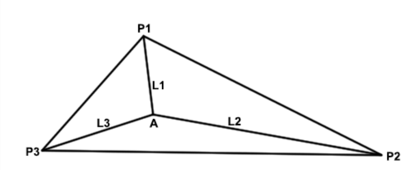


Figure 3: Trilateration with several GPS satellite TX and one RX.

A typical distance from a GPS RX to a satellite is 20,000 km.

Interactive question (3). What is the path loss of a TX/RX system (according to the Friis transmission equation) if $R = 20,000$ km? Assume $\lambda = 1$ m and $G = 10$ dB.

- A: -148 dB
- B: -158 dB
- C: -168 dB
- D: -204 dB

Satellite signals introduce path losses that must be counteracted if the signal is to be used for navigation.

Signals: amplitude versus time

Signals: amplitude versus time

1. Basic anatomy of a signal - part 1: units of time and amplitude
2. Example 1 with GNU Octave: plotting signals
3. **Special topic: sinusoids and complex signals**
4. Square pulses and binary sequences
5. **Special topic: counting in binary**
6. Sampling theorem
7. Example 2 with GNU Octave: aliasing, part 1

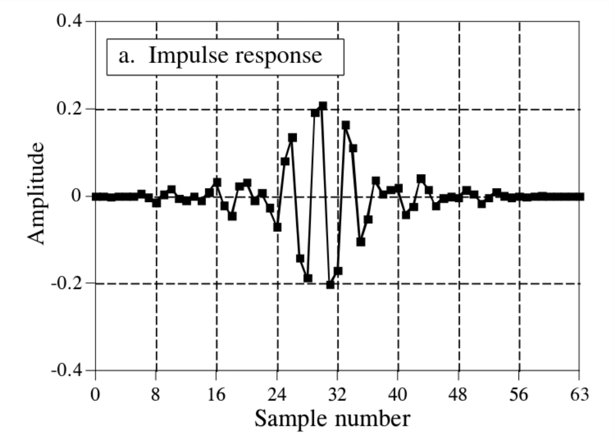


Figure 4: An example of a sampled, digitized signal representing the basic signal produced by an RF circuit when that circuit receives a pulse.

Signals: amplitude versus time

Time-domain signal properties:

1. Units of the y-axis: (A) amplitude from radio receivers as a voltage. For example, μV. (B) digitized samples, as in [0 : 2048], with the numbers corresponding to specific voltages.
2. Units of the x-axis: (A) specific times. For example, μs. (B) sampled times, such as [0 : 63], with samples related by Δt.

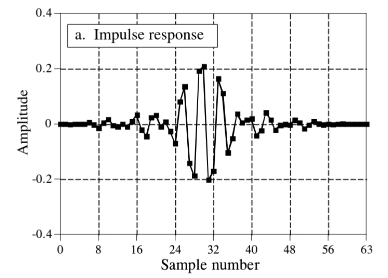


Figure 5: Amplitude versus sample number.

Signals: amplitude versus time

GNU Octave code example: “amplitude_time.m”

```
t_samples = 0:255;
sampling_frequency = 200.0; %Hz
delta_t = 1/sampling_frequency; %seconds
t = t_samples*delta_t;
frequency = 2.0; %Hz
signal = cos(2.0*pi*frequency.*t);
plot(t,signal,'linewidth',3,'color','black');
axis([0 max(t) -2 2]);
xlabel('Time (s)');
ylabel('Amplitude (V)');
set(gca(),'fontname','Calibri','fontsize',20);
grid on;
print('-dpdf','amp_time_2.pdf');
```

Signals: amplitude versus time

GNU Octave code example: “amplitude_time.m”

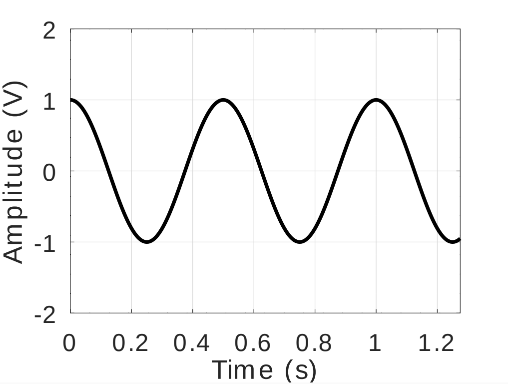


Figure 6: The figure created by the GNU Octave example.

Signals: amplitude versus time

GNU Octave exercise: Make the following changes, separately, to understand the impact on the signal:

```
t_samples = 0:511;
signal = 2*cos(2.0*pi*frequency.*t);
frequency = 4.0; %Hz
```

Other important quantities:

- *Power*: square each amplitude sample, and sum each value.
- *Average value*: sum the amplitude samples and divide by N.

Signals: amplitude versus time

```
P = sum(signal.^2); %Power
average_value = mean(signal); %Average
```

Other important quantities:

- *Power*: square each amplitude sample, and sum each value. What happens to the power when you calculate it with $f = 2$ Hz or $f = 4$ Hz?
- *Average value*: sum the amplitude samples and divide by N. What should be the average of a sinusoid, when the signal includes a whole number of periods?

Complex Signals: A tool for understanding signals

Let $j = \sqrt{-1}$, f be the frequency of a sinusoidal signal, and t be the time. Euler's theorem states that

$$\exp(2\pi jft) = \cos(2\pi ft) + j\sin(2\pi ft) \tag{16}$$

The *real* part of the signal is $\cos(2\pi ft)$, and the *imaginary* part is $\sin(2\pi ft)$.

Also note, exponentials and complex numbers have some useful properties:

- $\exp(x)\exp(y) = \exp(x+y)$
- $(x+jy)+(a+jb) = (x+a)+j(y+b)$

Signals: amplitude versus time

Treating real signals as complex

$$A\cos(2\pi ft) = \Re\{A\exp(2\pi jft)\} \tag{17}$$

By "taking the real part," we recover the cosine. "Taking the imaginary part" of a complex exponential gives the sine portion:

$$A\sin(2\pi ft) = \Im\{A\exp(2\pi jft)\} \tag{18}$$

Treating signals as exponentials allows us to multiply them easily, to understand product signal behavior.

Interactive Question (4). What happens when you multiply two signals with frequencies f_1 and f_2? (Start complex, then take the real part at the end).

Interactive Question (5). What happens when you multiply two signals with the same frequency f, but with different phases?

$$s_1(t) = a\cos(2\pi ft + \phi_1) \tag{19}$$

$$s_2(t) = a\cos(2\pi ft + \phi_2) \tag{20}$$

$$T(t) = s_1(t)s_2(t) \tag{21}$$

(Treat them as complex, then take the real part at the end).

Signals: amplitude versus time

Final GNU Octave exercise: Make the following change to the amplitude versus time code to understand the impact on the signal:

```
t_samples = 0:255;
sampling_frequency = 10.0; %Hz
delta_t = 1/sampling_frequency; %seconds
t = t_samples*delta_t;
frequency = 2.0; %Hz
```

How does this distort the signal?

- *Frequency greater than sampling frequency divided by 2*
- *Frequency less than sampling frequency divided by 2*

More on binary signals

More on binary signals

1. Binary numbers: digital signals from analog signals
2. Practical examples (interactive)
3. Chips and spreading symbols, chip rate, and data rate
4. Example 3 with GNU Octave: plotting a pseudo-random code sequence
5. Binary offset carriers

First of all, what is an *analog* circuit?

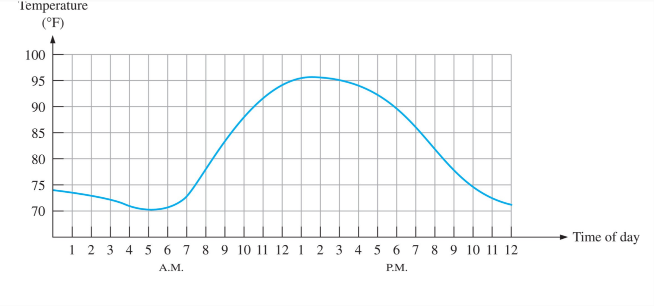


Figure 7: An example of an analog signal from a temperature sensor, converted from voltage.

More on binary signals

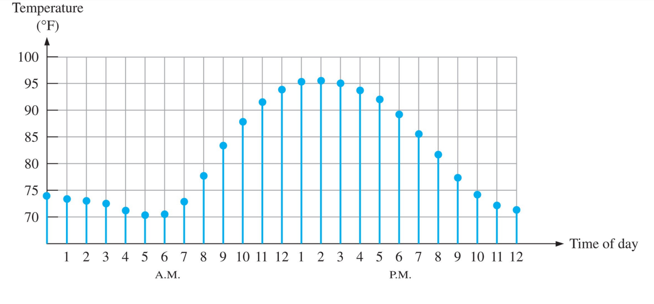


Figure 8: An example of that same signal, digitized and sampled.

Digital data forms the basis of computation:

- Noise issues, lossless transmission
- Constructed from *digits* … 1 and 0

More on binary signals

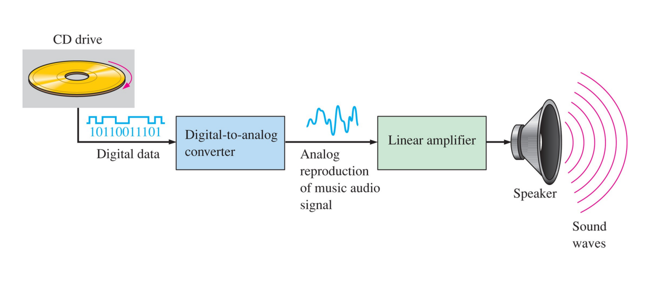


Figure 9: An example of a digital signal converted from binary to analog voltage signal.

How do we build up digital data from analog signals?

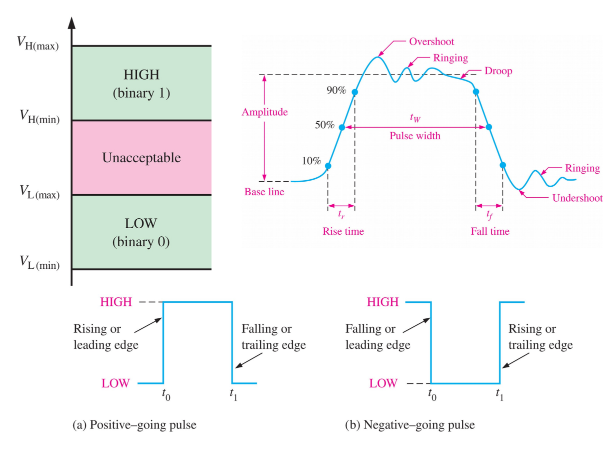


Figure 10: Logical "1" and "0."

More on binary signals

Terminology for digital signals:

1. **Frequency**, f and **period**, T: Signals per second, time between signals ($f = 1/T$).
2. **Pulse width**, t_{W}: time duration a pulse is HIGH.
3. **Duty cycle**: $t_{\mathrm{W}}/T \times 100\%$

More on binary signals

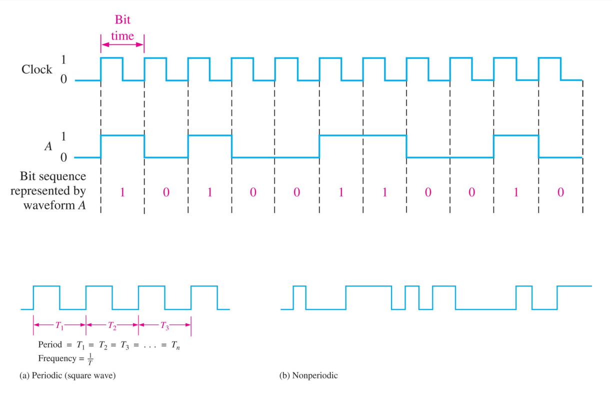


Figure 11: A clock signal is an example of a digital bitstream: alternating 1 and 0. It has a period and a frequency. Data can be *periodic* or *non-periodic*. (Professor: do some examples here).

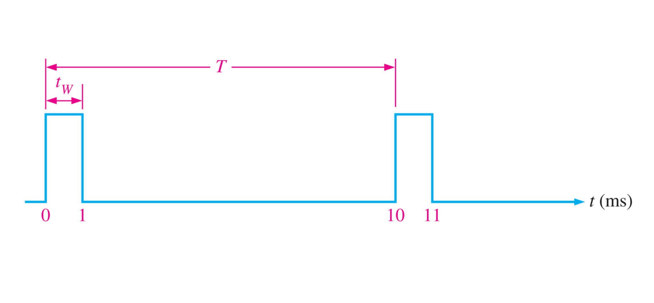


Figure 12: A periodic pulse demonstrating the concept of duty cycle. (Professor: do an example here and vary duty cycle).

What is the duty cycle in Fig. 12? What is the frequency?

More on binary signals

A script to produce a binary signal (install the “signal” package):

```
pkg load signal
t_samples = 0:255;
sampling_frequency = 200.0; %Hz
delta_t = 1/sampling_frequency; %seconds
t = t_samples*delta_t;
frequency = 2.0; %Hz
signal = square(2.0*pi*frequency.*t);
plot(t,signal,'linewidth',3,'color','black');
axis([0 max(t) -2 2]);
xlabel('Time (s)');
ylabel('Amplitude (V)');
set(gca(),'fontname','Calibri','fontsize',20);
grid on;
print('-dpdf','amp_time_3.pdf');
```

More on binary signals

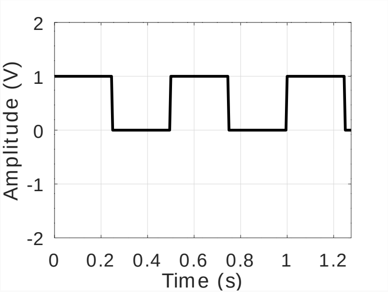


Figure 13: A periodic pulse demonstrating the concept of duty cycle. (Professor: do an example here and vary duty cycle). What is the frequency, period, and duty cycle of the signal?

More on binary signals

Make the following change to the code:

```
...
frequency = 2.0; %Hz
signal = square(2.0*pi*frequency.*t,10);
plot(t,signal,'linewidth',3,'color','black');
...
print('-dpdf','amp_time_4.pdf');
```

Interactive Question (6). If a binary signal repeats with period 1 μs, but has a pulse with of 0.2μs, what is the duty cycle?

- A: 10 percent
- B: 20 percent
- C: 200 percent
- D: 2 percent

Interactive Question (7). Draw in your notes a binary signal representing the following bit sequence: 11001101. Bonus: How could we make our code represent this signal?

```
t_samples = 0:255;
sampling_frequency = 200.0; %Hz
delta_t = 1/sampling_frequency; %seconds
t = t_samples*delta_t;
period = length(t_samples)/8;
seq = [1 1 0 0 1 1 0 1];
signal = []
for i=seq
        signal = [signal ones(1,period)*i];
endfor
plot(t,signal,'linewidth',3,'color','black');
axis([0 max(t) -2 2]);
xlabel('Time (s)');
ylabel('Amplitude (V)');
set(gca(),'fontname','Calibri','fontsize',20);
grid on;
print('-dpdf','amp_time_4.pdf');
```

More on binary signals

Encoding our signals. What if we don't want someone to intercept our data and use it? Consider the following signal:

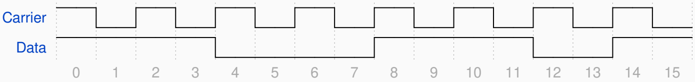


Figure 14: (Top) A binary carrier signal, and (Bottom) a binary sequence representing some data: 11001101, at the same rate as the clock signal.

Notice: The pulse width of the "Carrier" signal in Fig. 14 is one half the minimum pulse width of the "Data" signal. For one period of the carrier, we have one data bit. The fastest *data rate* is equal to the carrier frequency.

More on binary signals

Encoding our signals. What if we don't want someone to intercept our data and use it? Consider the following signal:

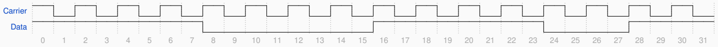


Figure 15: (Top) A binary carrier signal, and (Bottom) a binary sequence representing the same data as Fig. 14, 11001101, at one half the rate as in Fig. 14.

Notice: The pulse width of the "Carrier" signal in Fig. 14 is one fourth the minimum pulse width of the "Data" signal. For two periods of the carrier, we have one data bit. The fastest *data rate* is equal half of the carrier frequency.

More on binary signals

Now multiply the carrier and the data signals. The result is a *pseudo-random code*:

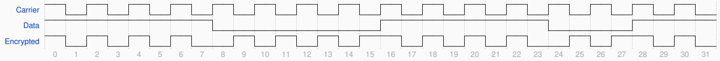


Figure 16: (Top) A binary carrier signal, (Middle) The data stream. (Bottom) the product of the carrier and the data stream.

Notice: The encrypted data stream now appears as random data at the carrier frequency. If the data stream is already encrypted through binary techniques, the signal at the bottom of Fig. 16 would be very difficult to intercept and use.

More on binary signals

The spreading symbol: By taking the data bit and multiplying it by 1010, the data is said to be *spread* by a *spreading symbol* 1010.

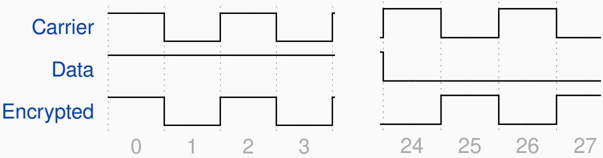


Figure 17: (Left) The first data bit. (Right) The second to last data bit.

Interactive Question (8). Suppose we have a binary carrier with a period of 0.1 μs, and a data stream of 1011 0010 to transmit. (a) Write out the encrypted data sequence by multiplying the carrier with the data stream. Assume the data rate is half the frequency of the carrier. (b) How long (in time) is the encrypted data?

Some vocabulary:

1. **Chip**: the unit of the data in the encrypted result.
2. **Symbol**: the pieces of data made from multiple chips.
3. **Chip rate**: the rate of the encrypted signal.
4. **Symbol rate**: the rate of the data stream.
5. **Spreading factor**: the ratio of the chip rate to the data rate.

Power spectral densities

1. Basic anatomy of a signal - part 2: units of frequency and power
2. Units: dB, and dB/Hz
3. Practical examples (interactive)
4. The Fourier transform and FFT algorithm
5. Example with GNU Octave: FFT of a sinusoid
6. Example with GNU Octave: FFT and noise
7. Example with GNU Octave: FFT and mixing

Units of frequency and power: learn to flip numbers.

Period	Frequency
1 ms	1 kHz
1 μs	1 MHz
1 ns	1 GHz
1 ps	1 THz

Table 1: (Left) Period (Right) Frequency

Interactive Question (9). What is the period of a 90 MHz sine wave carrier?

- A: 11 μs
- B: 11 ns
- C: 11 ms
- D: 11 ps

Units of frequency and power: learn to flip numbers.

Period	Frequency
1 ms	1 kHz
1 μs	1 MHz
1 ns	1 GHz
1 ps	1 THz

Table 2: (Left) Period (Right) Frequency

Interactive Question (10). What is the period of a 5 GHz sine wave carrier?

- A: 0.2 μs
- B: 0.2 ms
- C: 0.2 ps
- D: 0.2 ns

Units of frequency and power: learn to flip numbers.

Period	Frequency
1 ms	1 kHz
1 μs	1 MHz
1 ns	1 GHz
1 ps	1 THz

Table 3: (Left) Period (Right) Frequency

Interactive Question (11). What is the period of a 2 MHz sine wave carrier?

- A: 0.5 μs
- B: 0.5 ms
- C: 0.5 ps
- D: 0.5 ns

Units of frequency and power: convert power to dB.

Power (Watts)	Power (dB)
1 W	0
1 mW	-30
1 μW	-60
1 nW	-90

Table 4: (Left) Period (Right) Frequency

Interactive Question (12). What is 0.5 Watts expressed in dB?

- A: -3 dB
- B: -1 dB
- C: -3 dB
- D: -6 dB

Units of frequency and power: convert power to dB.

Power (Watts)	Power (dB)
1 W	0
1 mW	-30
1 μW	-60
1 nW	-90

Table 5: (Left) Period (Right) Frequency

Interactive Question (13). What is 0.5 milliwatts expressed in dB?

- A: -3 dB
- B: -30 dB
- C: -33 dB
- D: -60 dB

Voltage and Power

In a circuit, the relationship between voltage V, power P, and resistance R, is

$$P = \frac{V^2}{R} \tag{22}$$

In GPS receivers, or radio receivers in general, the actual resistance at the transmission frequency is not usually important. What we must remember is that $P \propto V^2$.

Voltage and Decibels

Given the relationship between power and voltage:

$$P_{dB} = 20 \log_{10}(V/V_{\mathrm{ref}}) \tag{23}$$

Units of amplitude and power: convert power to dB.

Interactive Question (14). If we're observing a signal at -10 dB, and the *amplitude* (voltage) of the signal doubles, what's the new power in dB?

- A: -4 dB
- B: -6 dB
- C: -10 dB
- D: 0 dB

Interactive Question (15). If we're observing a signal at -60 dB, and the *amplitude* (voltage) of the signal increases by a factor of 10, what's the new power in dB?

- A: -30 dB
- B: -40 dB
- C: -60 dB
- D: -10 dB

Power spectral densities

Given the structure of a time-domain signal, there are ways to convert it to a corresponding structure in the frequency-domain. One way is called the *Fourier Transform:*

Conversion from Time to Frequency Domain

The conversion between a signal in the *time-domain* $v(t)$ and a signal in the *frequency-domain* $\widetilde{v}(f)$ is called the **Fourier Transform**:

$$\widetilde{v}(f) = \int_{-\infty}^{\infty} e^{-2\pi jft} v(t) dt = \int_{-\infty}^{\infty} \cos(2\pi ft) V(t) dt - j \int_{-\infty}^{\infty} \sin(2\pi ft) V(t) dt \tag{24}$$

Computing the Fourier transform is like multiplying the signal by sines and cosines and then integrating with respect to time. The output is complex.

The *magnitude squared* of the Fourier transform of a time-domain signal is proportional to the power at a given frequency:

Power from Fourier Transform: 1

$$|\widetilde{v}(f)|^2 = \left| \int_{-\infty}^{\infty} e^{-2\pi jft} V(t) dt \right|^2 \tag{25}$$

Example: An AC circuit delivers current through a resistor with resistance R. The instantaneous power P is $P = V^2(t)/R$. The average power over a time T, with $R = 1\Omega$, is

$$P = \frac{1}{T} \int_{-T/2}^{T/2} |v(t)|^2 dt \tag{26}$$

Power spectral densities

According to Parseval's Theorem ($R = 1\Omega$):

Power from Fourier Transform: 2

$$P = \lim_{T\to\infty} \frac{1}{T} \int_{-\infty}^{\infty} |v(t)|^2 dt = \lim_{T\to\infty} \frac{1}{T} \int_{-\infty}^{\infty} |\widetilde{v}(f)|^2 df \qquad (27)$$

Example: suppose we calculate the mass of an object with 1D density $\rho(x)$ by integrating:

$$m = \int_a^b \rho(x) dx \qquad (28)$$

The *integrand* is the density, whereas the *result* on the left-hand side is the *mass*.

Power spectral densities

This implies that the power density, or power spectral density (PSD), is the integrand in Eq. 27.

Power from Fourier Transform: 3

$$PSD(f) = \lim_{T\to\infty} \frac{1}{T}|\tilde{v}(f)|^2 \tag{29}$$

and

$$|\tilde{v}(f)|^2 = \left|\int_{-\infty}^{\infty} e^{-2\pi jft} V(t)dt\right|^2 \tag{30}$$

Notice that Eq. 30 has units of V^2s^2, or $(V/Hz)^2$. This implies that the units of Eq. 29 has units of V^2/Hz. When integrated with respect to frequency, the result would have units of V^2 (power with $R = 1\Omega$).

When we receive radio transmissions, the signal (after several steps of filtering, mixing, and processing) may be digitized. The discrete form of Eq. 29 is given below.

Power from Fourier Transform: 4

$$PSD(f_n) = \frac{(\Delta t)^2}{T} \left| \sum_{n=-N/2}^{N/2} v_n e^{-2\pi j f n \Delta T} \right|^2 = \frac{(\Delta t)^2}{T} \left| \widetilde{v}_n \right|^2 \qquad (31)$$

The units of Eq. 31 remain V^2/Hz. Note that (a) $t_n = n\Delta t$, (b) $T = N\Delta t$, and (c) $f_n = n/T = n/(N\Delta t)$.

We may also write the PSD in terms of N instead of T.

Power from Fourier Transform: 5

$$PSD(f_n) = \frac{(\Delta t)^2}{T} \left|\widetilde{v}_n\right|^2 = \frac{\Delta t}{N} \left|\widetilde{v}_n\right|^2 \quad (32)$$

The units of Eq. 31 remain V^2/Hz. Note that (a) $t_n = n\Delta t$, (b) $T = N\Delta t$, and (c) $f_n = n/T = n/(N\Delta t)$.

The Fast Fourier Transform: In computer code, and some examples of RF hardware, an algorithm called the fast Fourier transform (FFT) is implemented. In GNU Octave, it is called simply `fft()`.

```
fs = 1000.0;
dt = 1/fs;
f = 3.0;
t = 0:dt:(2*pi);
v_n = 2.0*sin(2.0*pi*f.*t)+1.0;
vf_n = fft(v_n)
```

GNU Octave Example: The code below can be found in `amplitude_frequency.m`.

```
f_s = 200.0; %Hz
dt = 1/f_s; %Seconds
t = 0:dt:1.0; %Seconds
N = length(t); %Integer
f = 20.0; %Hz
A = 2.0; %Volts
s = A*cos(2.0*pi*f.*t); %Volts
S = fft(s); %Fast Fourier Transform (FFT)
S = conj(S).*S; %FFT squared
S = 2*S(1:end/2); %Take the positive frequencies
S = dt/N*S;
S = 10*log10(S);
f = linspace(0,f_s/2,N/2);
plot(f,S,'-o','linewidth',1,'color','black','markersize',4);
axis([0 f_s/2 -100 20]);
xlabel('Frequency (Hz)');
ylabel('V^2/Hz');
set(gca(),'fontname','Calibri','fontsize',20);
grid on;
print('-dpdf','amp_freq_1.pdf');
```

GNU Octave Example: The code below can be found in `amplitude_frequency.m`.

```
f_s = 200.0; %Hz
dt = 1/f_s; %Seconds
t = 0:dt:1.0; %Seconds
N = length(t); %Integer
f = 20.0; %Hz
A = 2.0; %Volts
s = A*cos(2.0*pi*f.*t); %Volts
```

This portion of the code defines a sampling frequency of 200 Hz, and creates time samples accordingly. Then, a cosine function with frequency 20 Hz and amplitude 2V is evaluated at those times.

GNU Octave Example: The code below can be found in `amplitude_frequency.m`.

```
S = fft(s); %Fast Fourier Transform (FFT)
S = conj(S).*S; %FFT squared
S = 2*S(1:end/2); %Take the positive frequencies
```

This portion of the code computes the FFT of the cosine samples. Next, the absolute value squared is found, and we take half of the array (times 2). This is because (for real signals) the amplitudes of the FFTs at negative frequencies are the same as the positive ones.

GNU Octave Example: The code below can be found in `amplitude_frequency.m`.

```
S = dt/N*S;
S = 10*log10(S);
f = linspace(0,f_s/2,N/2);
plot(f,S,'-o','linewidth',1,'color','black','markersize',4);
axis([0 f_s/2 -100 20]);
xlabel('Frequency (Hz)');
ylabel('V^2/Hz');
set(gca(),'fontname','Calibri','fontsize',20);
grid on;
print('-dpdf','amp_freq_1.pdf');
```

To convert to PSD, we divide by N, and multiply by Δt. Next, we create an array of the right frequencies (ranging from 0 to $f_{max} = f_s/2$). This follows from the sampling theorem one of the prior pieces. Finally, we convert to decibels and plot the signal.

Finally, we convert to decibels and plot the signal.

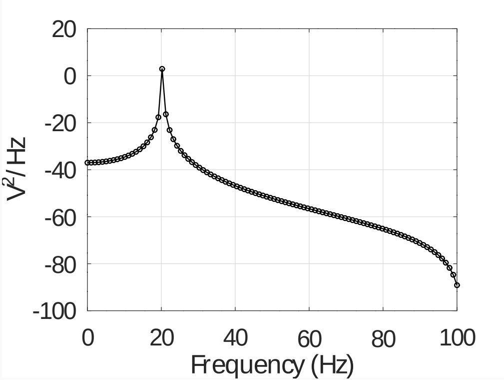


Figure 18: The power spectral density of a 20 Hz cosine wave sampled at 200 Hz.

Make the following changes to the code:

```
s = A*cos(2.0*pi*f.*t); %Volts
s = s + A*cos(4.0*pi*f.*t); % Volts
```

How will the new graph of the PSD differ from Fig. 18?.

The following figure can be generated using `amplitude_frequency_2.m`

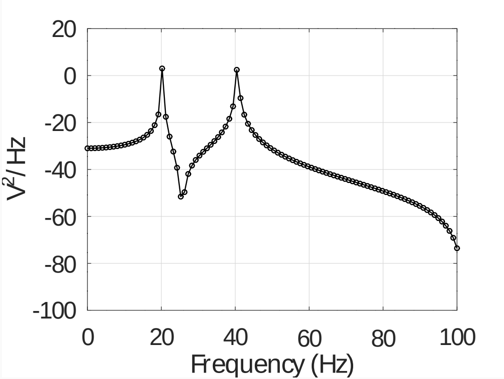


Figure 19: The PSD of 20 Hz and 40 Hz cosine waves sampled at 200 Hz.

Make the following changes to the code:

```
s = A*cos(2.0*pi*f.*t); %Volts
s = s + A*cos(4.0*pi*f.*t); % Volts
s = s + randn(size(t))*A/8; % Volts
```

Random noise. In addition to signal, radar and GPS systems always detect random noise in addtion. This is a broad topic, because there are many sources of noise in RF systems. The above line simply adds random “Gaussian” noise to the samples of the signal before creating the PSD.

(Use `amplitude_frequency_3.m`).

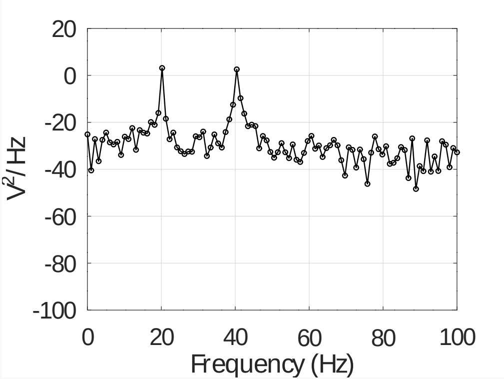


Figure 20: The PSD of 20 Hz and 40 Hz cosine waves (plus Gaussian noise) sampled at 200 Hz. There is equal signal power at both frequencies, and the noise power is distributed at all frequencies.

Finally, make the following changes to the code:

```
s = A*cos(2.0*pi*f.*t); %Volts
s = s.*cos(4.0*pi*f.*t)*A; % Volts
s = s + randn(size(t))*A/8; % Volts
```

Mixing signals. Where *should* the peaks be, on top of the noise? (The cosines are being multiplied).

(Use `amplitude_frequency_4.m`).

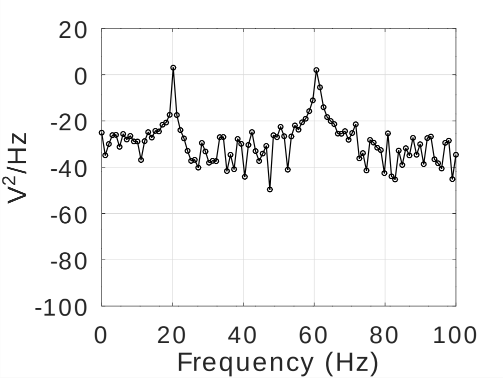


Figure 21: The PSD of 20 Hz and 40 Hz cosine waves (plus Gaussian noise), that have been *mixed*, sampled at 200 Hz.

Power spectral densities

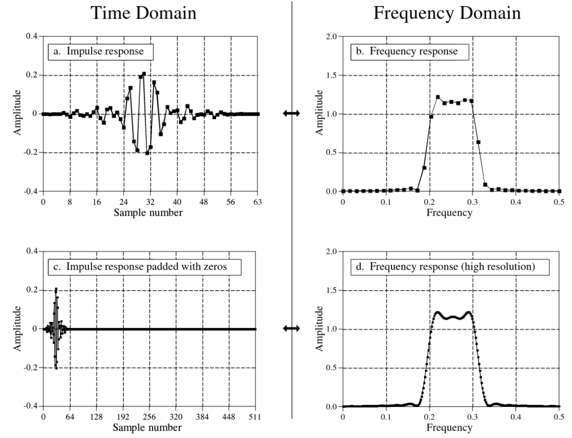


Figure 22: Sampling for *longer* gives better frequency resolution.

Power spectral densities

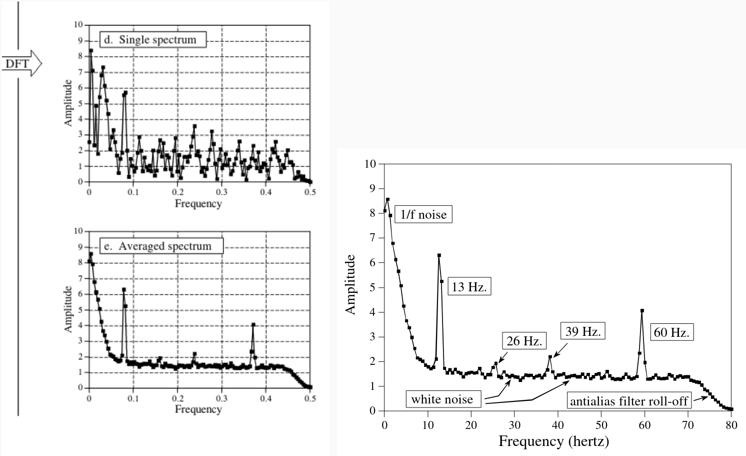


Figure 23: Averaging many PSDs reduces the effect of noise, and enhances the signal to noise ratio.

Mixing signals, carrier frequencies

1. Mathematics of mixing signals:
    - Review of complex signals
    - Interactive questions about mixing
2. Block diagram of mixing
3. Examples with GNU Octave: moving a signal to a carrier frequency
4. Examples with GNU Octave: moving a spectrum by mixing with a carrier

Review: mulitplication of complex sinusoids.

$$s_1(t) = \Re\left\{ae^{2\pi j f_1 t}\right\} \tag{33}$$

$$s_2(t) = \Re\left\{ae^{2\pi j f_2 t}\right\} \tag{34}$$

$$s_1(t) \times s_2(t) = s_3(t) + s_4(t) \tag{35}$$

$$s_3(t) = \frac{a^2}{2}\cos(2\pi(f_1 + f_2)t) \tag{36}$$

$$s_4(t) = \frac{a^2}{2}\cos(2\pi(f_1 - f_2)t) \tag{37}$$

Professor: provide proof.

Mixing signals, carrier frequencies

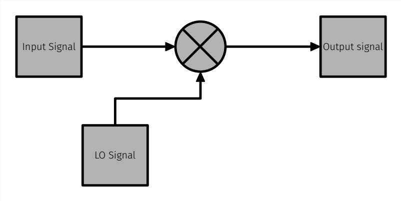
Interactive Question (16): Suppose we have a radio receiver with a tunable LO (local oscillator). We tune it to 1360 kHz, and multiply the LO signal with an incoming AM signal at 930 kHz to provide an output. At which frequency will the output have the highest PSD (power spectral density)?

- A: 130 kHz
- B: 2290 kHz
- C: 430 kHz
- D: 930 kHz

Interactive Question (17): Let the incoming radio signal frequency be f. Suppose we always put our tunable LO frequency at $f - 430$ kHz. Suppose we then mix the LO and incoming signal, and *filter* the output with a low-pass filter that takes away everything above f. Which of the following is true of the output signal?

- A: It will be filtered and therefore attenuated.
- B: It will always be at the frequency f.
- C: It will be amplified and at the frequency f.
- D: It will be at 430 kHz.

Mixing moves signals in the PSD. In an example from the prior piece (power spectral densities), we see the movement of signals in frequency-space.



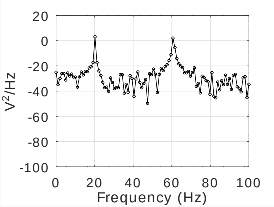


Figure 24: (Left) The basic circuit diagram for mixing. (Right) Example PSD after mixing a 40 Hz carrier with a 20 Hz signal.

What happens when $f_{carrier} \gg f_{signal}$?

Mixing signals, carrier frequencies

GNU Octave example: See the script `amplitude_frequency_5.m` in the code folder. The key elements are below:

```
f_s = 250.0 * 1000.0; %Hz (250 kHz)
dt = 1/f_s; %Seconds
T = 0.1; %Seconds
t = 0:dt:T; %Seconds
N = length(t); %Integer
f_1 = 2.0 * 1000.0; %Hz (2 kHz)
f_2 = 4.0 * 1000.0; %Hz (4 kHz)
f_3 = 8.0 * 1000.0; %Hz (8 kHz)
f_carrier = 100.0 * 1000.0; %Hz (100 kHz)
A = 2.0; %Volts
s = A*cos(2.0*pi*f_1.*t); %Volts
s = s+A*cos(2.0*pi*f_2.*t); %Volts
s = s+A*cos(2.0*pi*f_3.*t); %Volts
s = s.*cos(2.0*pi*f_carrier.*t)*A; %Volts
s = s + randn(size(t))*A/8; % Volts
```

Mixing signals, carrier frequencies

GNU Octave example: See the script `amplitude_frequency_5.m` in the code folder. The key elements are below:

```
...
%The usual PSD routine
...
subplot(1,2,1)
plot(f,S,'-o','linewidth',1,'color','black','markersize',4);
axis([0 f_s/2/1000.0 -100 20]);
xlabel('Frequency (kHz)');
ylabel('V^2/Hz');
set(gca(),'fontname','Calibri','fontsize',14);
grid on;
subplot(1,2,2)
plot(f,S,'-o','linewidth',1,'color','black','markersize',4);
axis([90.0 110.0 -100 20]);
xlabel('Frequency (kHz)');
ylabel('V^2/Hz');
set(gca(),'fontname','Calibri','fontsize',14);
grid on;
```

Mixing signals, carrier frequencies

Results of `amplitude_frequency_5.m`:

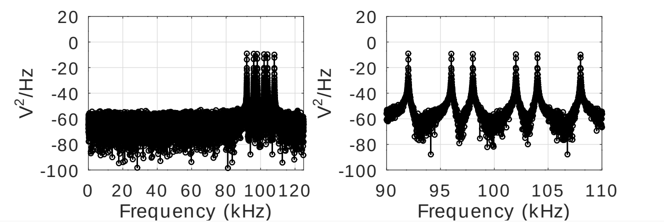


Figure 25: (Left) The entire PSD. (Right) The PSD, but just the [90-110] kHz bandwidth.

Interactive Question (18): In Fig. 25, what frequencies correspond to the peaks?

- A: 92, 96, and 98 kHz
- B: 102, 104, and 108 kHz
- C: 2, 4, and 8 kHz
- D: 92, 96, 98, 102, 104, and 108 kHz

What happens to the spectrum of a signal comprised of many frequencies? Results of `amplitude_frequency_6.m`:

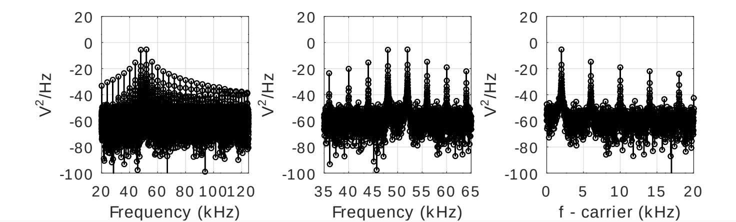


Figure 26: (Left) The PSD from 20 kHz to critical frequency. (Middle) [35.0-65.0] kHz. (Right) The PSD vs. frequency minus the carrier frequency.

Mixing signals, carrier frequencies

GNU Octave example: See the script `amplitude_frequency_6.m` in the code folder. The key elements are below:

```
f_1 = 2.0 * 1000.0; %Hz (2 kHz)
f_carrier = 50.0 * 1000.0; %Hz (100 kHz)
A = 2.0; %Volts
s = A*square(2.0*pi*f_1.*t); %Volts
s = s.*cos(2.0*pi*f_carrier.*t)*A; %Volts
s = s + randn(size(t))*A/8; % Volts
...
f = f/1000.0; %Convert to kHz
figure(1,'position',[0,0,700,200]);
subplot(1,3,1)
plot(f,S,'-o','linewidth',1,'color','black','markersize',4);
...
plot(f,S,'-o','linewidth',1,'color','black','markersize',4);
axis([35.0 65.0 -100 20]);
...
plot(f-f_carrier/1000.0,S,'-o','linewidth',1,'color','black','markersi
axis([0.0 20.0 -100 20]);
```

Auto-correlation functions

Auto-correlation functions (ACFs)

1. **Mixing a signal with itself**: definition of the discrete ACF
2. **Interactive questions**
3. Examples with GNU Octave: ACF of a square pulse
4. The Wiener-Khinchin Theorem (frequency-domain computation of ACF)
5. **The addition of noise**: statistical properties of noise ACFs

Auto-correlation functions (ACFs)

Recall that signal mixing corresponds to multiplying two signals at a point in time.

Definition of ACF

The auto-correlation function (ACF) for a real, sampled, digitized signal s_n for N times t_n and time “lag” τ is

$$R(k) = \sum_{n=-N}^{N} s_n s_{n-k} \tag{38}$$

where $t_k = \tau = k\Delta t$, and $t_n = n\Delta t$.

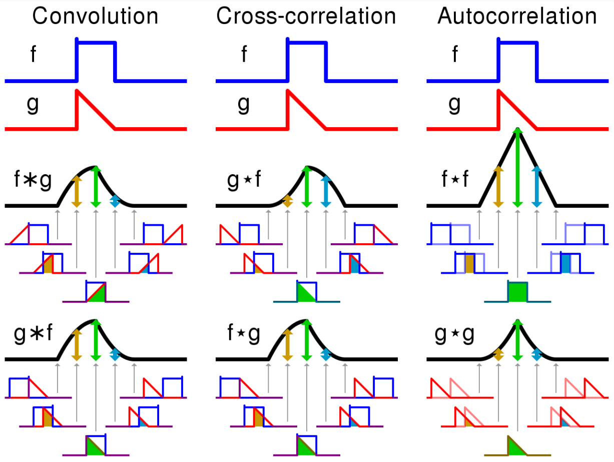


Figure 27: (Left column) Convolution. (Middle column) Cross-correlation, for two signals *f* and *g*. (Right column) Cross-correlation for *f* and *f* or *g* and *g* (ACF).

Auto-correlation functions (ACFs)

Example: Why is the ACF of f (Fig. 28) a triangle?

- For lags that make one copy of f *too early*, there is no overlap and so the product of f and f is zero.
- For lags that make one copy of f *too late*, there is no overlap and so the product of f and f is zero.
- Non-zero contributions to the ACF for a given lag come from the overlap of f and f.

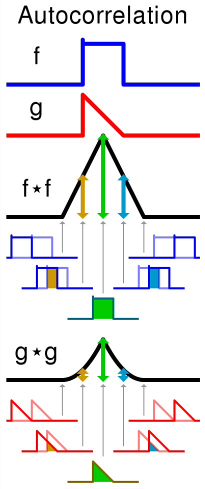


Figure 28: Cross-correlation for f and f or g and g (ACF).

Interactive Question (19):
Suppose we compute the ACF of a signal that is two bits wide with an amplitude of 1.0, defined like $0, 0, 1, 1, 0, 0$. What are the height and width of the ACF?

- A: Height: 1, Width: 1
- B: Height: 2, Width: 2
- C: Height: 3, Width: 2
- D: Height: 2, Width: 3

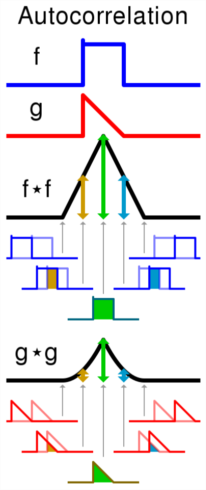


Figure 29: Cross-correlation for f and f or g and g (ACF).

Auto-correlation functions (ACFs)

Interactive Question (20): What is the ACF of the following signal: 0, 1, 0, 0, 1, 0?

- A: 00100200100
- B: 00100300100
- C: 00001210000
- D: 00000200000

Hint: as you slide one signal across the other, ask how many times there should be a non-zero overlap?

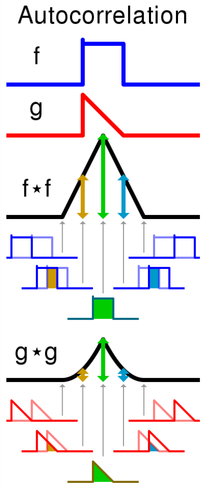


Figure 30: Cross-correlation for f and f or g and g (ACF).

Auto-correlation functions (ACFs)

GNU Octave example: Use `autocorrelation_1.m` to graph the ACF of a square pulse with known width.

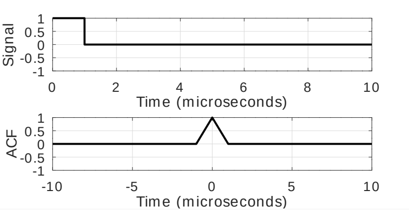


Figure 31: The results of running the octave example.

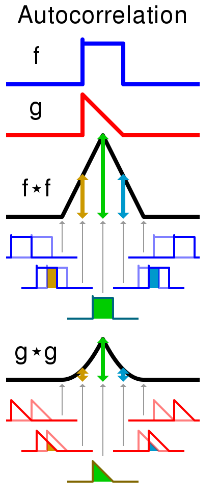


Figure 32: Cross-correlation for f and f or g and g (ACF).

Auto-correlation functions (ACFs)

```
...
T = 1.0; %pulse width, microseconds
fs = 1000.0; %Sampling rate, MHz
dt = 1/fs; %Delta t, microseconds
total_time = 10.0*T;
t = 0:dt:total_time;
s = zeros(size(t));
N = floor(T/dt);
s(1:N) =  ones(1,N);
subplot(2,1,1)
plot(t,s,'color','black','linewidth',3.0);
...
subplot(2,1,2);
acf = xcorr(s);
acf = acf/max(acf);
t_acf = [-fliplr(t) t(2:end)];
plot(t_acf,acf,'color','black','linewidth',3.0);
...
```

GNU Octave example: What happens when we add additional pulses? Use `autocorrelation_2.m` to obtain the results.

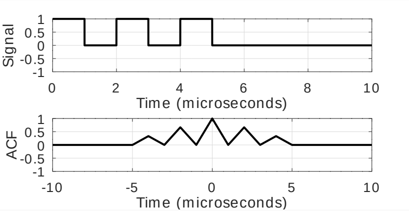


Figure 33: The results of running the octave example.

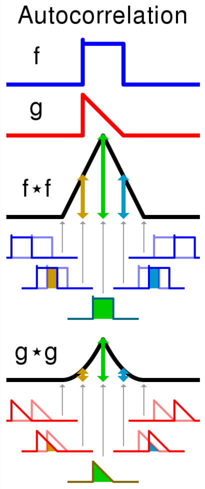


Figure 34: Cross-correlation for f and f or g and g (ACF).

Auto-correlation functions (ACFs)

```
...
T = 1.0; %pulse width, microseconds
fs = 1000.0; %Sampling rate, MHz
dt = 1/fs; %Delta t, microseconds
total_time = 10.0*T;
t = 0:dt:total_time;
s = zeros(size(t));
N = floor(T/dt);
s(1:N) = ones(1,N);
s(2*N:3*N-1) = ones(1,N);
s(4*N:5*N-1) = ones(1,N);
subplot(2,1,1)
plot(t,s,'color','black','linewidth',3.0);
...
subplot(2,1,2);
acf = xcorr(s);
acf = acf/max(acf);
t_acf = [-fliplr(t) t(2:end)];
plot(t_acf,acf,'color','black','linewidth',3.0);
...
```

The Wiener-Khinchin theorem is a way of relating the frequency-domain information to the ACF.

The Wiener-Khinchin theorem

The inverse Fourier transform of the power spectral density (PSD) of a continuous signal is the ACF, and the Fourier transform of the ACF is the PSD.

$$R(\tau) = \int_{-\infty}^{\infty} P(f) e^{2\pi j f \tau} df \tag{39}$$

$$P(f) = \int_{-\infty}^{\infty} R(\tau) e^{-2\pi j f \tau} d\tau \tag{40}$$

Example with Wiener-Khinchin theorem. Suppose $P(f) = A\delta(f - f_0)$, meaning that there is power concentrated at f_0, and no other frequency. Show that the ACF is a sinudoid. *Does this make sense physically?*

$$\delta(f - f_0) = \begin{cases} \infty & f = f_0 \\ 0 & f \neq f_0 \end{cases} \tag{41}$$

$$\int_{-\infty}^{\infty} \delta(f - f_0) df = 1 \tag{42}$$

$$\int_{-\infty}^{\infty} g(f)\delta(f - f_0) df = g(f_0) \tag{43}$$

GNU Octave example: What happens when we add noise? Use `autocorrelation_3.m` to obtain the results.

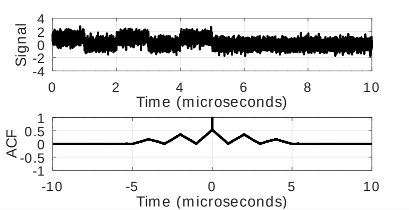


Figure 35: The results of running the octave example.

Notes on Fig. 35:

- Gaussian white noise was added.
- SNR of only 2.
- Cross-correlation (in general) suppresses noise
- Exception: a lag of zero, when the noise is equal to itself.