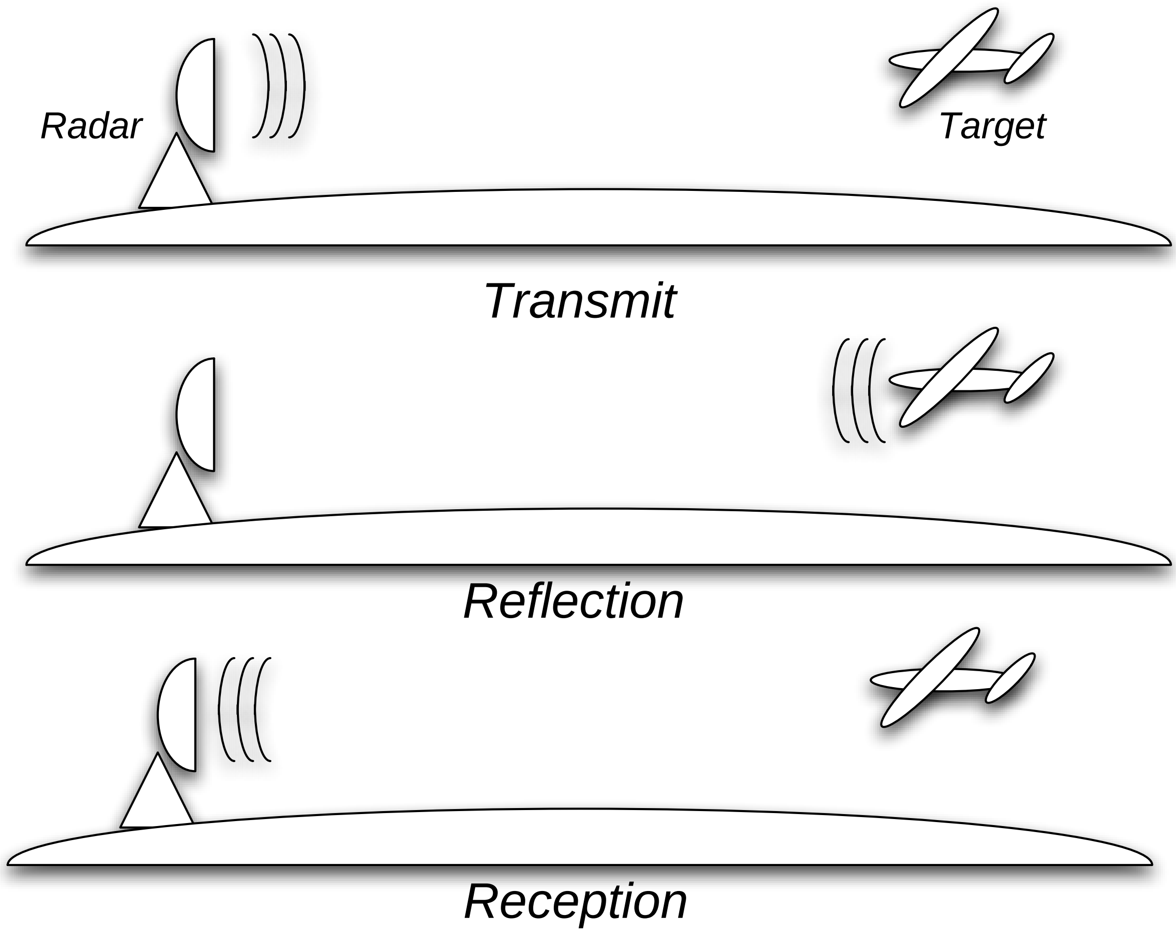
Stanford University
Autumn 2020-2021

Introduction to Analog and Digital Communications

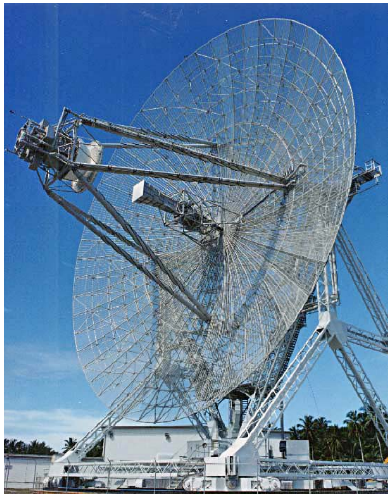
Lecture 19: Radar and CDMA

December 1, 2021

Basic Radar Idea



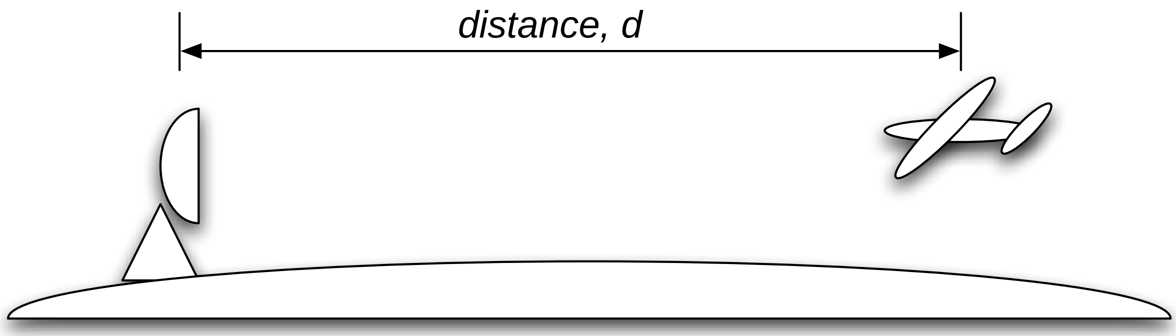
Radar Antennas

http://wikipedia.org/Radar.html

Radar Parameters

The target is a distance d from the antenna



The pulse travels a total distance of $2d$ at the speed of light c, which is 3×10^8 m/s, or 300 m/μs.

The echo returns after a delay of

$$t_d = \frac{2d}{c}$$

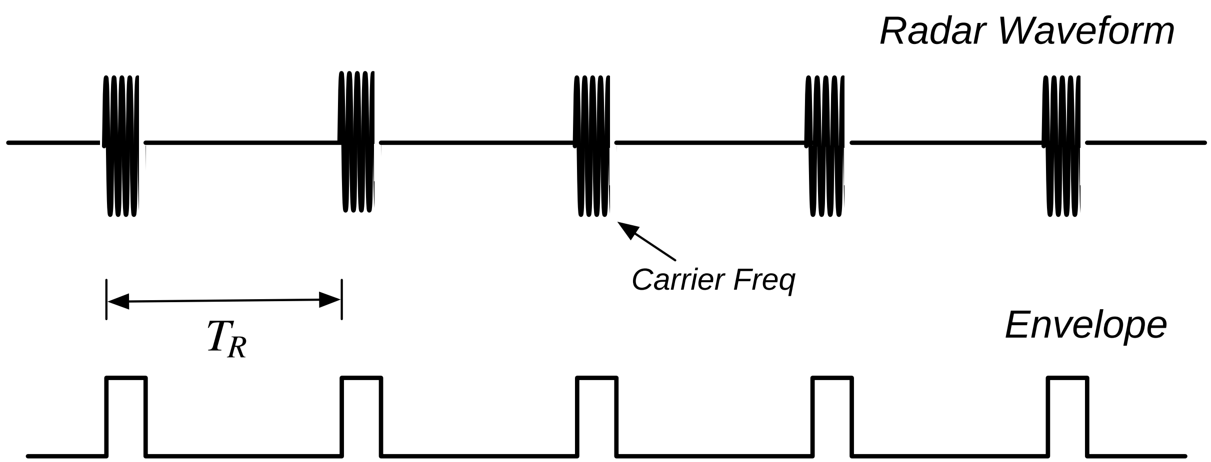
A delay of 1 μs corresponds to a distance of

$$d = ct_d/2 = (300 \text{ m}/\mu\text{s})(1\mu\text{s})/2 = 150 \text{ m}$$

There is also a frequency shift (Doppler shift) that we'll ignore for now, but will come back to shortly.

Simple Radar Waveform

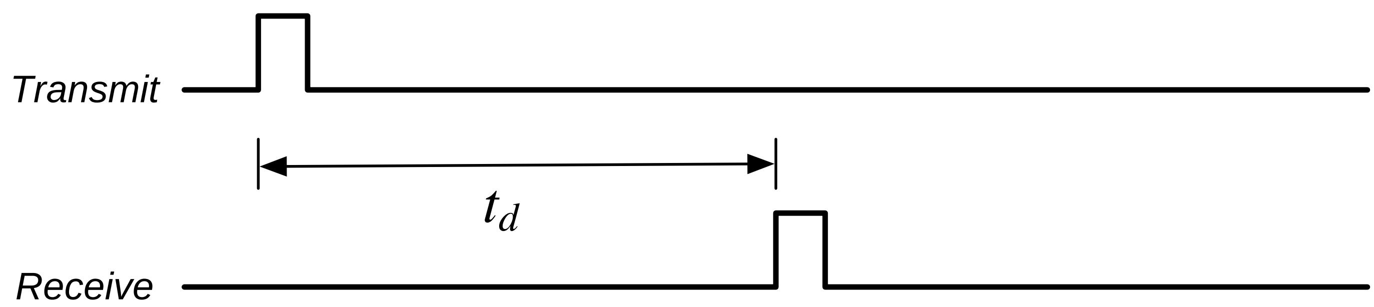
A simple radar waveform is an envelope multiplied by a carrier.



We only need to consider the envelope of the signal to understand basic radar signal processing.

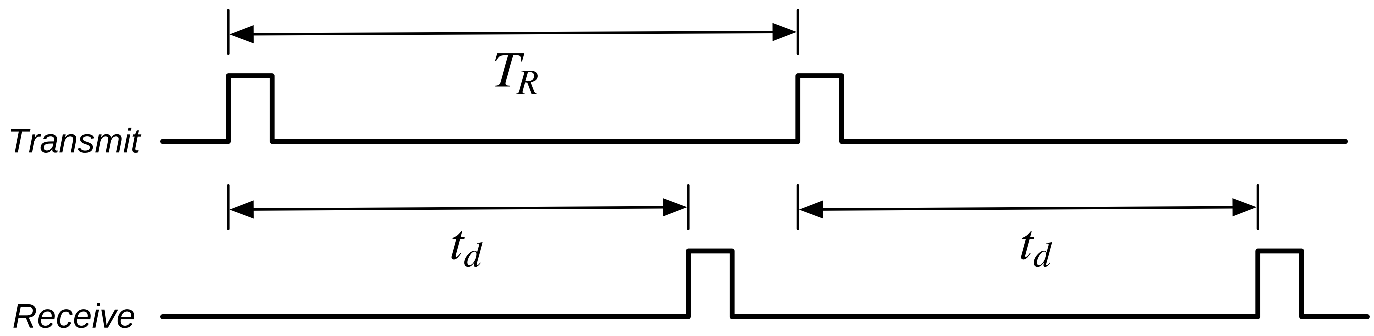
Range Ambiguity

In a simple experiment, we transmit a pulse, and wait for it to come back



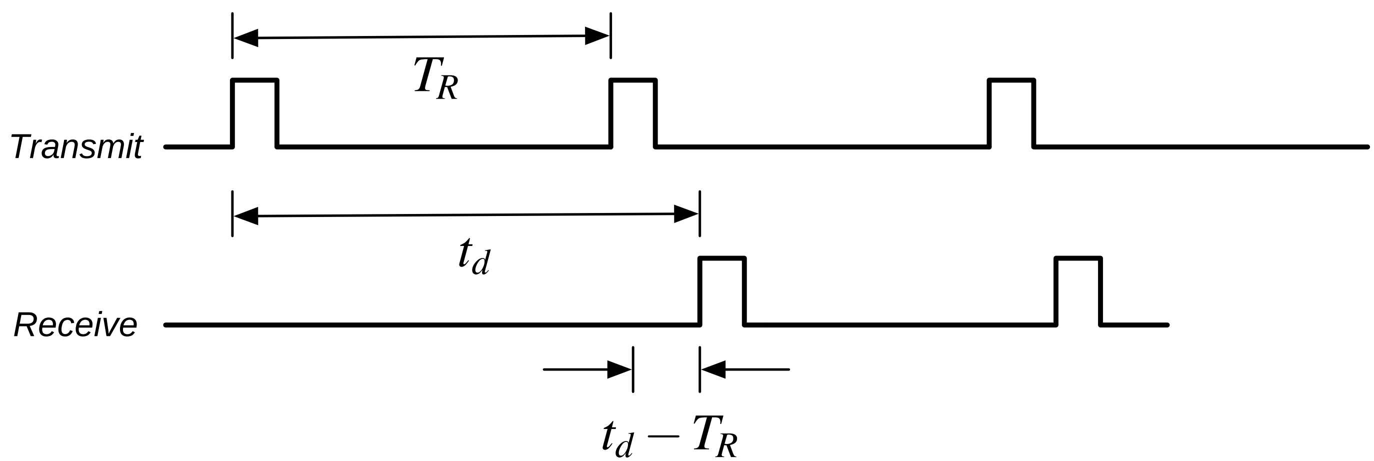
This tells us that there is a target at a distance $d = c\ t_d/2$.

Practically we want to repeat the measurement as rapidly as possible, to track the target



We transmit a new pulse after we receive the previous return.

If we don't wait long enough ...



We think there is a target at $c\ (t_d - T_R)/2$, which is much closer!

The targets at $c\ (t_d - T_R)/2$ and $c\ t_d/2$ are *ambiguous*, we can't tell them apart.

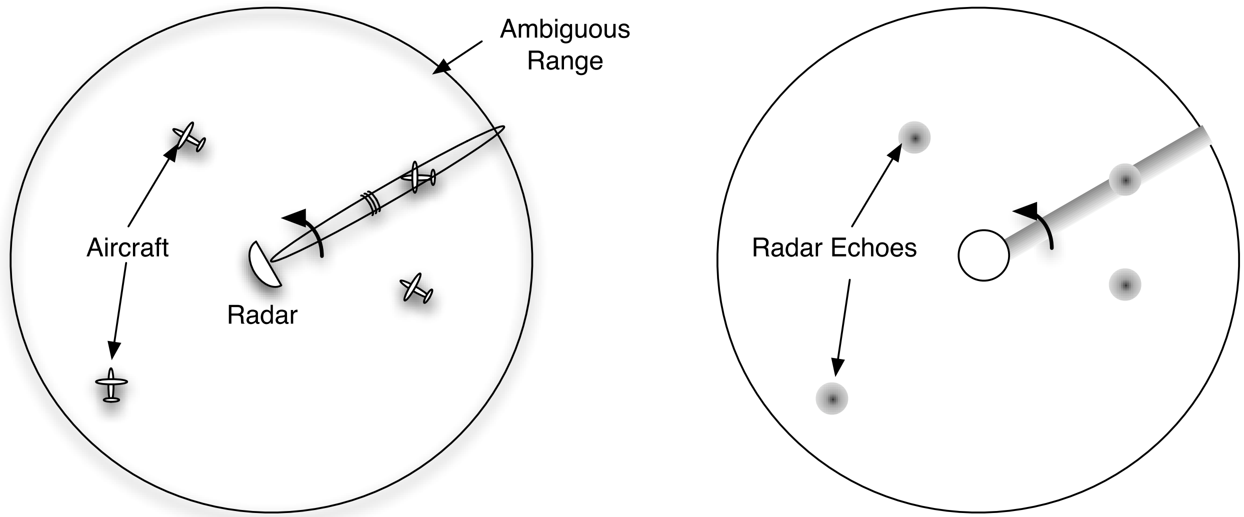
To be unambiguous, we need the roundtrip time $2d/c$ to be less than T_R. A given T_R has a maximum usable range.

Ballistic Missile Early Warning Systems (BMEWS) Thule, Greenland.

`https://en.wikipedia.org/wiki/Ballistic_Missile_Early_Warning_System`

Radar Display

Air traffic control radar gives a top-down 2D view of the airspace



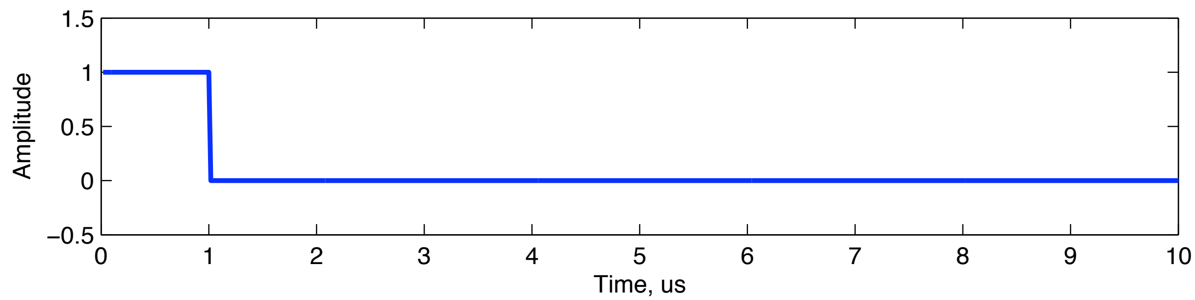
Air Traffic Control Radar

PPI Radar Display

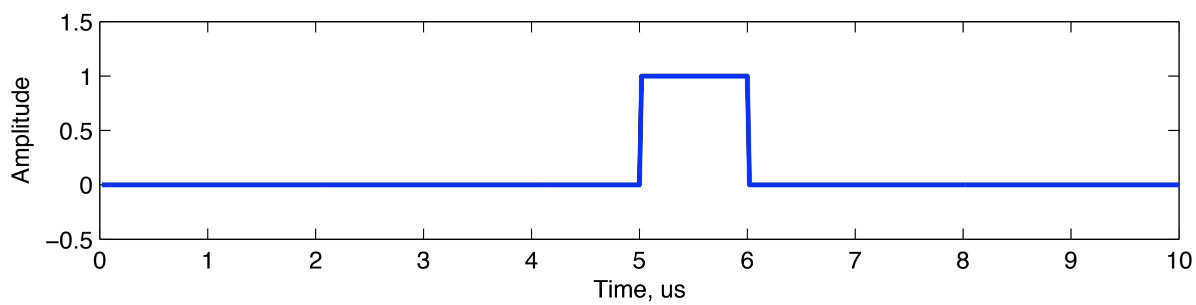
- Radar sends a pulse out, and listens for echoes
- The received signal is written on the display at the same angle
- The radar rotates, and sends out another pulse

Pulse Detection

How do we detect when the pulse has returned? It seems simple if the SNR is high. If we transmit a pulse

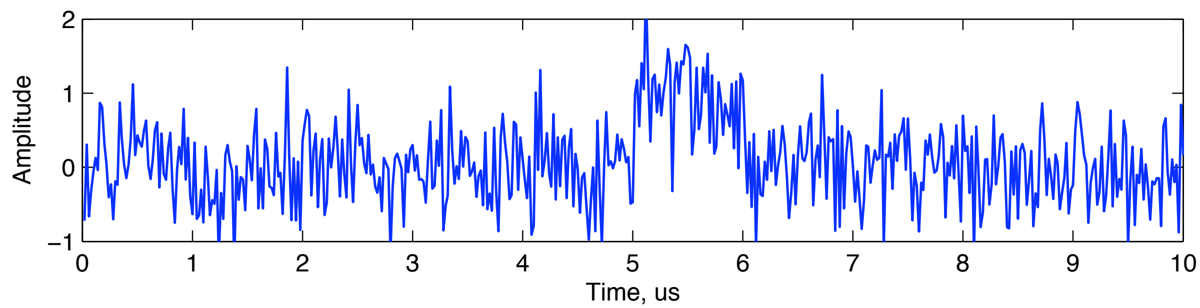


We get a delayed version of the pulses at the receiver



We could measure the time of arrival of the leading edge of the pulse, for example.

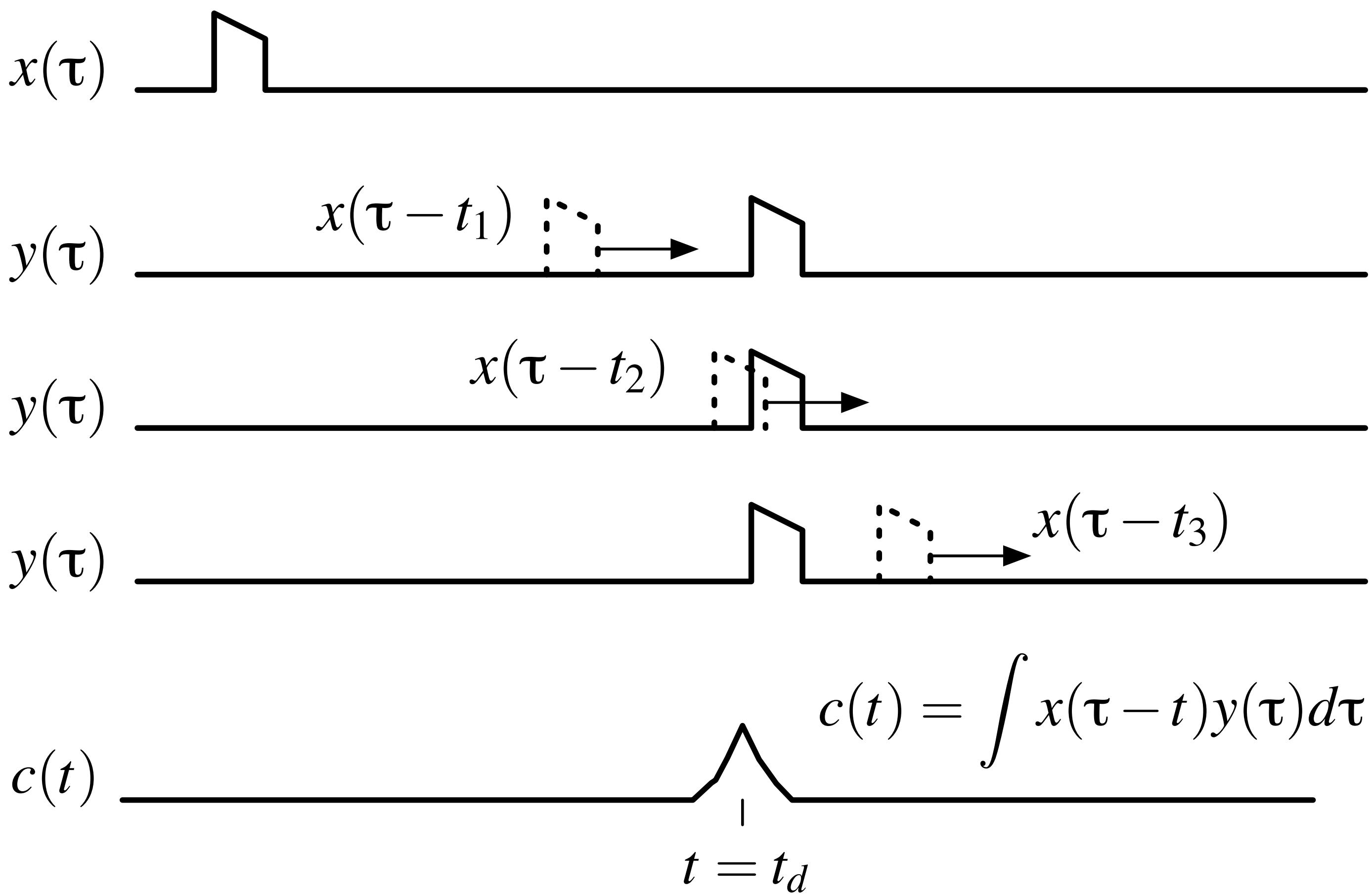
In practice, the returned signal is much smaller than the transmitted signal, and has been corrupted by noise,



Here it is much harder to say where the reflected signal is!

Matched Filters, and Correlation Receivers

Solution is to search for signals that look like the transmitted signal.



If $x(t)$ is the transmit signal, and $y(t)$ is the returned signal, then

$$c(t) = \int_{-\infty}^{\infty} x(\tau - t)y(\tau)d\tau$$

This is the cross-correlation of x and y, which is written as

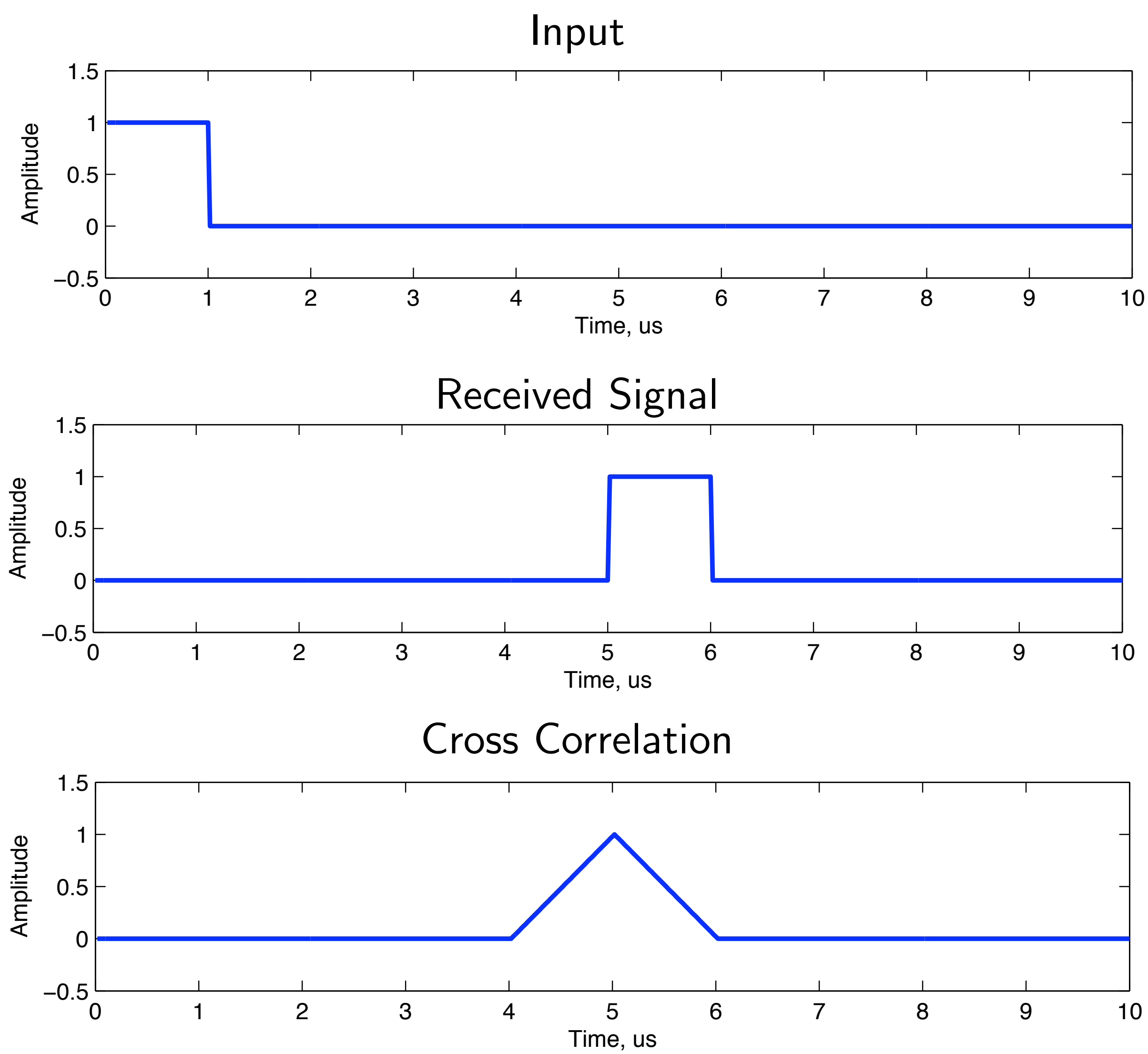
$$(x \star y)(t) = \int_{-\infty}^{\infty} x(\tau - t)y(\tau)d\tau = \int_{-\infty}^{\infty} x(\tau)y(\tau + t)d\tau$$

To look for a reflected signal at time t,

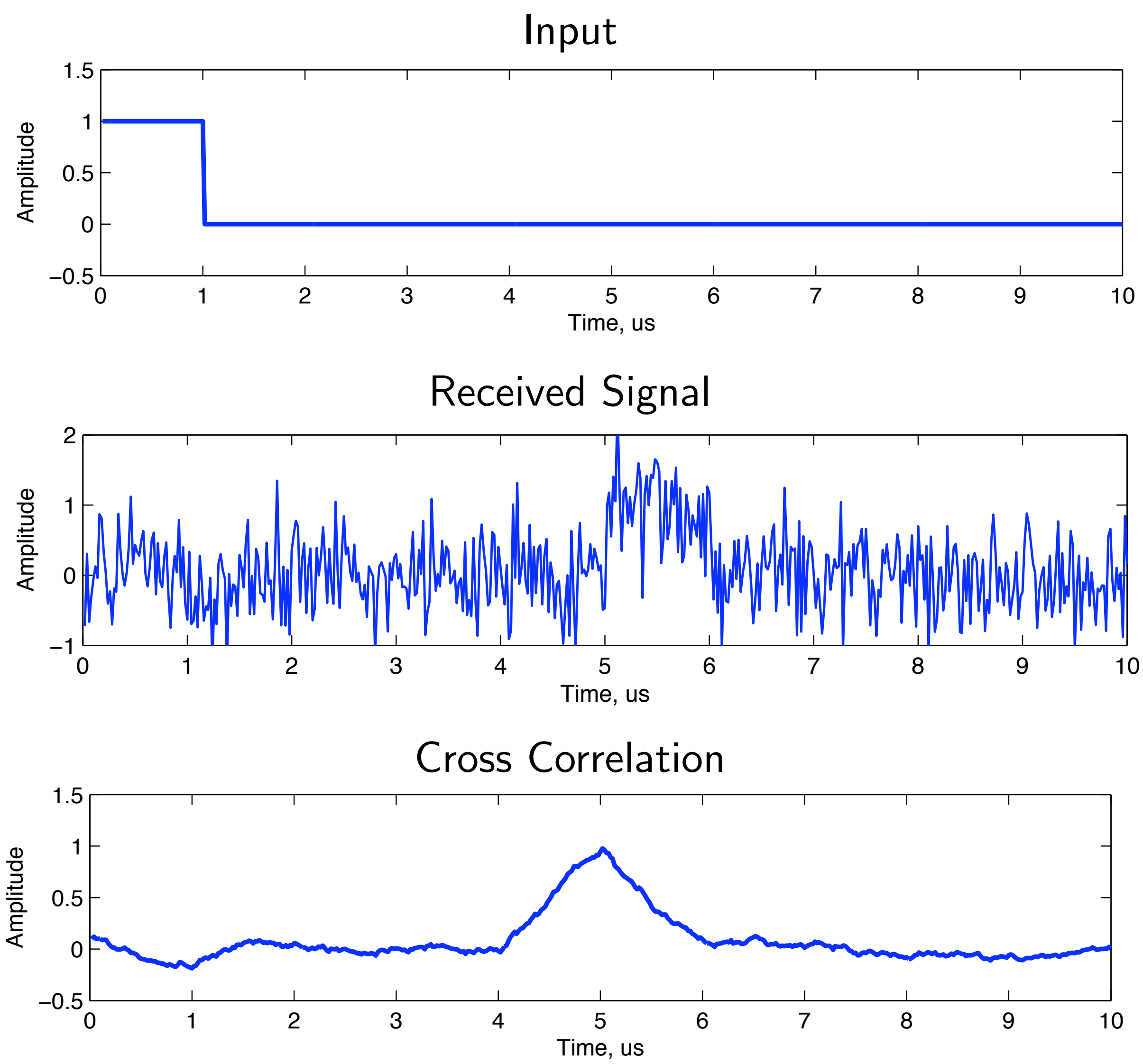
- Shift the transmitted signal to time t,
- Multiply point by point with the received signal,
- Then integrate.

This is just like convolution, except you don't reverse one of the signals.

For the noiseless case before, we get



For the high-noise case,



Cross correlation properties

- Better defined peak, easier to identify time
- Broader peak, twice the duration of the transmit pulse
- Good suppression of noise, optimal under reasonable assumptions

If the received signal at time t_d is really $y(\tau) = x(\tau - t_d)$, then the cross correlation at t_d is

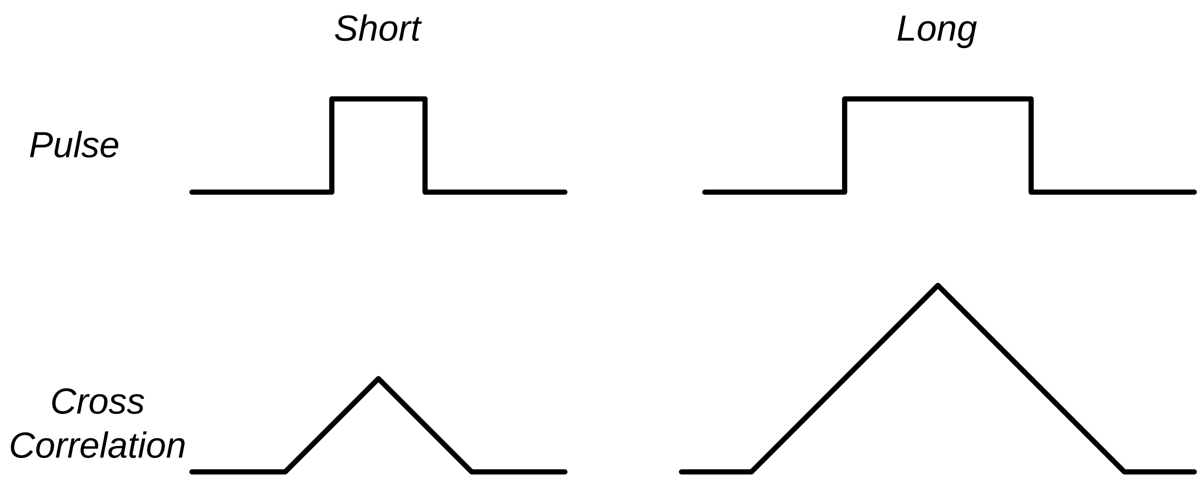
$$c(t_d) = \int x(\tau - t_d)y(\tau)d\tau = \int x(\tau - t_d)x(\tau - t_d)d\tau = \int x^2(\tau)d\tau = E_x$$

which is the energy in the transmitted pulse.

Transmit Pulses

We would like the transmitted pulse to have a large E_x, so that we can detect signals far away.

Peak RF amplitude is limited, so we have to make the pulses longer to increase E_x.



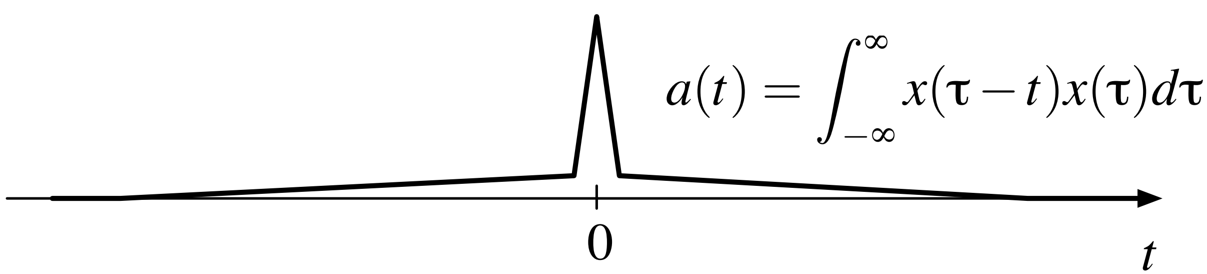
Doubling the pulse length doubles E_x, but makes it harder to accurately identify the return time.

Pulse Compression

Can we have the large E_x of a long pulse, with the temporal resolution of a short pulses?

Remarkably, yes!

What we want are pulses whose cross-correlation with themselves are small, except right at the origin.

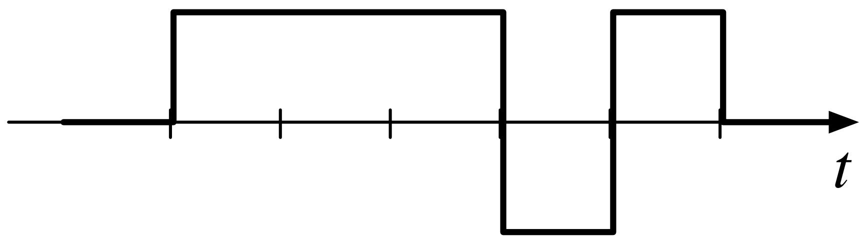


This is the *autocorrelation* of $x(t)$.

Finding pulses with these properties has been a major research effort in radar.

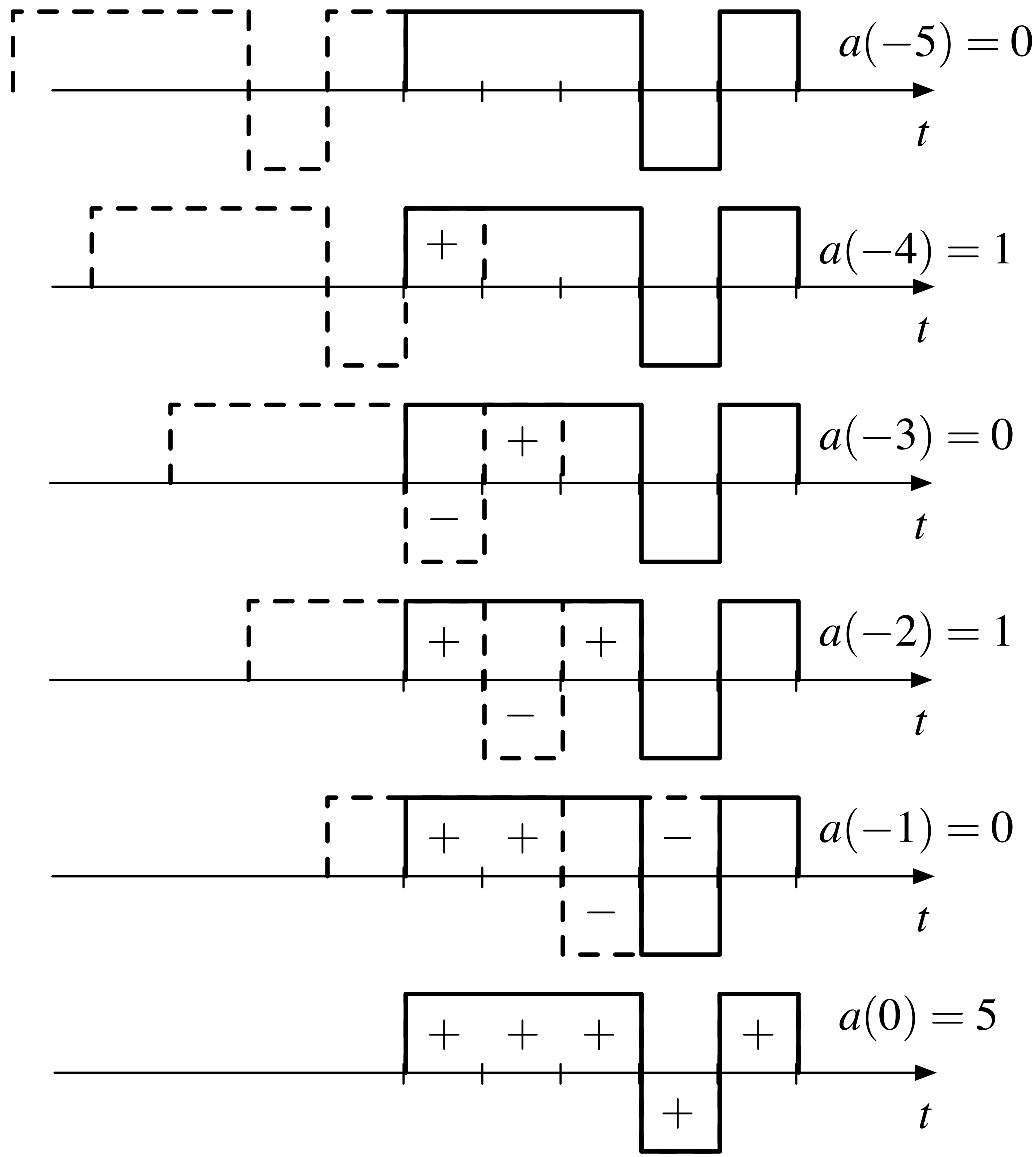
One example are pulses based on *Barker codes*.

A length 5 Barker code is



It is described by the amplitudes of the subpulses, in this case $(+1, +1, +1, -1, +1)$.

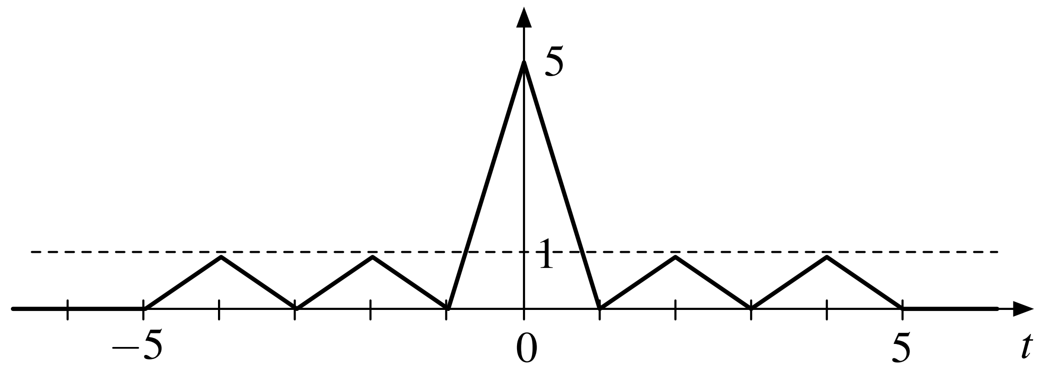
The remarkable thing about Barker codes, is that the autocorrelation on an N sample code is either ± 1 or 0 everywhere, except at $t = 0$, when it is N!



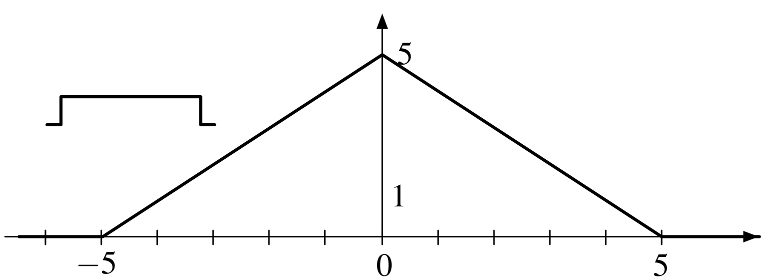
$a(-5) = 0$
t
$a(-4) = 1$
t
$a(-3) = 0$
t
$a(-2) = 1$
t
$a(-1) = 0$
t
$a(0) = 5$
t

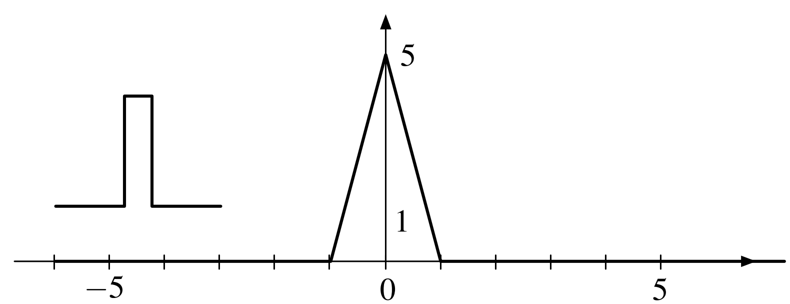
The positive time samples can be found similarly.

The result is



This has the energy of a long rectangular pulse, with the temporal resolution of a high-amplitude short rectangular pulse.





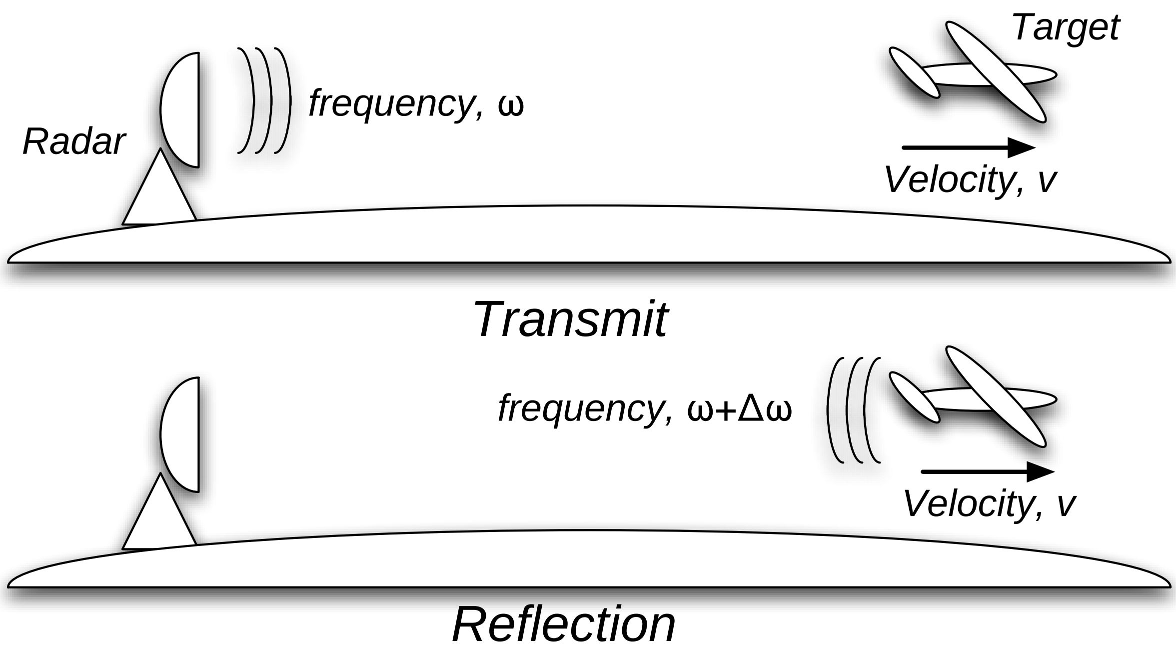
This is called pulse compression, because the autocorrellation is effectively compressed compared to the transmit pulse.

Surprisingly, Barker codes are known for only a few N!

N	Code
2	$(+1, -1)$
3	$(+1, +1, -1)$
4	$(+1, -1, +1, +1)$
5	$(+1, +1, +1, -1, +1)$
7	$(+1, +1, +1, -1, -1, +1, -1)$
11	$(+1, +1, +1, -1, -1, -1, +1, -1, -1, +1, -1)$
13	$(+1, +1, +1, +1, +1, -1, -1, +1, +1, -1, +1, -1, +1)$

Besides radar and pulse compression, Barker codes are also used in communication for spread spectrum modulation.

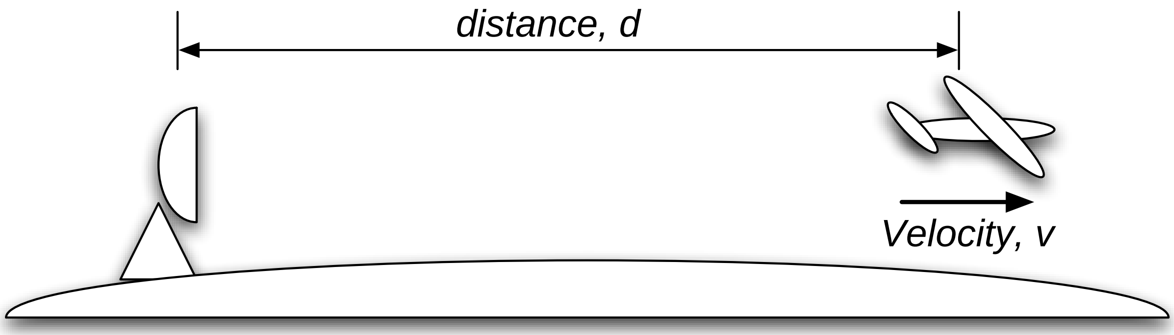
Basic Doppler Radar Idea



The target is moving.

The reflected RF pulse is shifted in frequency.

Doppler Shift



The target is a distance d from the antenna, moving at a velocity v.

The phase of the received signal is the total distance divided by the wavelength λ.

$$\phi = -2\pi \frac{2d}{\lambda}$$

The negative sign is due to the fact that the received signal is *delayed* in time.

The frequency of the signal is

$$\omega_d = \frac{d}{dt}\phi = \frac{d}{dt}\left(-2\pi\frac{2d}{\lambda}\right) = -2\pi\frac{2v}{\lambda}$$

The wavelength $\lambda = c/(\omega/2\pi) = 2\pi c/\omega_c$, where c is the speed of light, so

$$\omega_d = -2\pi\frac{2v}{2\pi c/\omega_c} = -\frac{2v}{c}\omega_c$$

or, dividing both sides by 2π, and using $f = \omega/2\pi$,

$$f_d = -\frac{2v}{c}f_c$$

For an airplane traveling 300 m/s (just below the speed of sound, 330 m/s), and a radar frequency of $f_c = 1$GHz, this gives

$$f_d = -\frac{(2)(3 \times 10^2 \text{ m/s})}{3 \times 10^8 \text{ m/s}}(1 \times 10^9 \text{ Hz}) = -2 \times 10^3 \text{ Hz} = -2 \text{ kHz}.$$

You also see a similar frequency shift in communications.

The shift is half of the radar doppler shift, since there is only the one-way path.

This shift is noticeable when you are decoding packets. The frequency will drift depending on the changing distance between you and the airplane.

It is even more of an issue with low earth orbit satellites, due to their very high speeds.

Pulsed Doppler

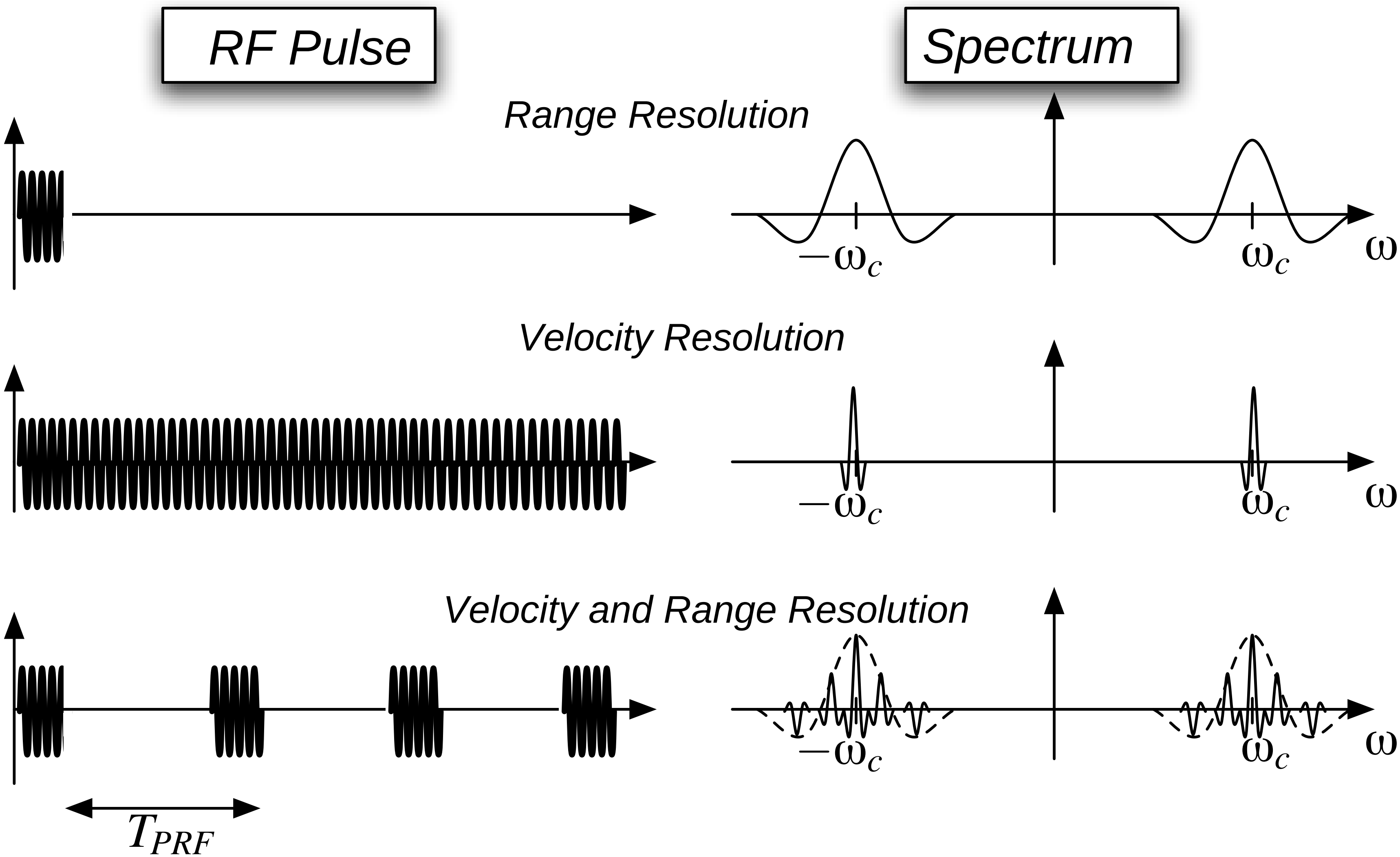
A single, short pulse gives good range resolution, but poor frequency (velocity) resolution.

A single, long pulse gives good frequency (velocity) resolution, but poor range resolution.

How do we get both, to localized the velocity in a particular vessel, or heart valve?

Many different solutions.

A common solution in radar is *Pulsed Doppler*, shown on the next page.



Pulsed Doppler waveform

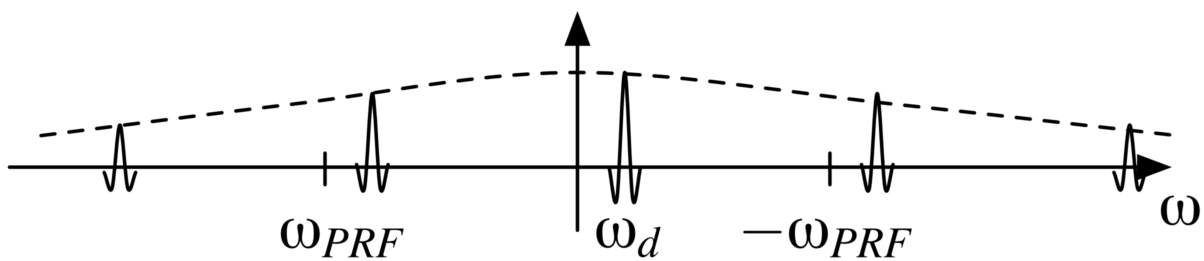
A short pulse is repeated many times, at an spacing T_{PRF}.

The round-trip time must be less than this

$$t_d = \frac{2d}{c} < T_{PRF}$$

Otherwise a closer voxel will also be returning signal at the same time, and the signal is *ambiguous*.

After quadrature demodulation, the baseband spectrum looks like



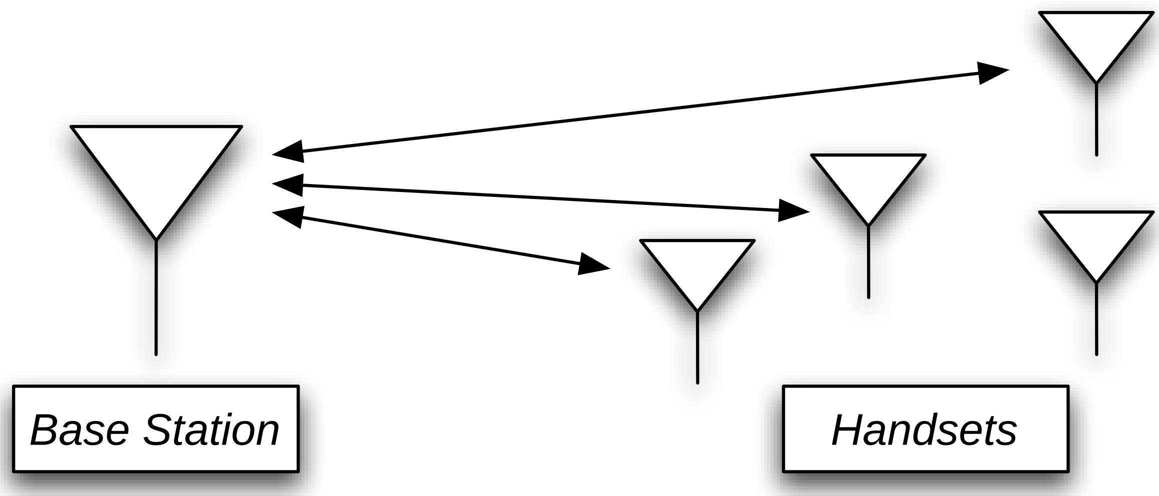
The spacing between the sublobes is $f_{PRF} = 1/T_{PRF}$, and this must be greater than the Doppler shift

$$f_d = \frac{2v}{c} f_c < f_{PRF}$$

or the velocity is ambiguous. This is the same as sampling.

Often, f_{PRF} is not high enough due to the round-trip time required to reach a specific depth, d, and aliasing occurs.

Cell Telephone



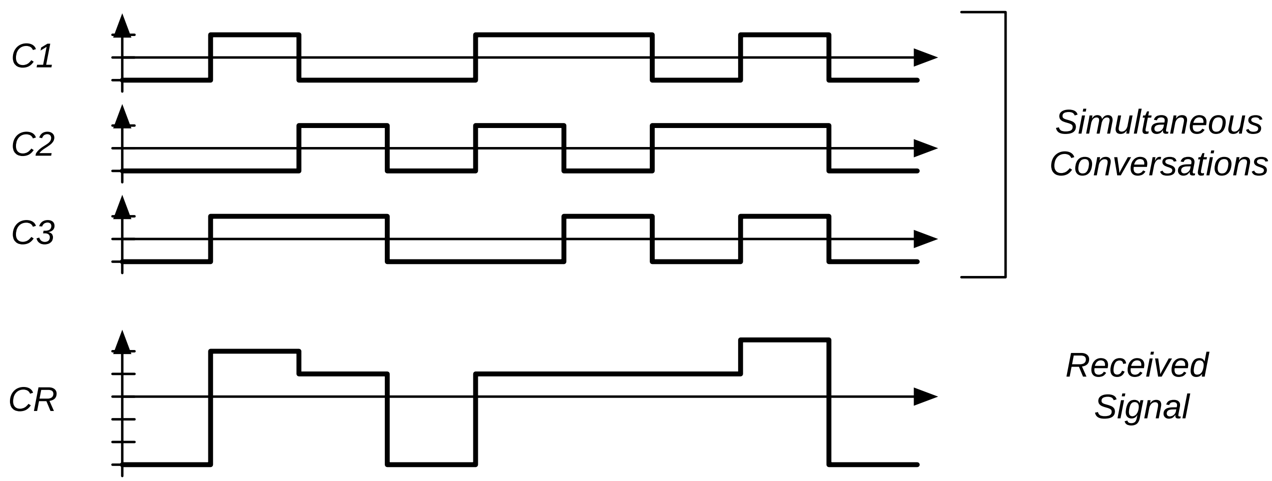
Base station talks to all of the handsets in the cell.

Simple solution, each link has a frequency. However, there are a limited number of frequencies.

We would like to have multiple connections for each frequency. How do we do this?

Sharing a Channel

If all of the handsets try to use the same frequency, the signals superimpose, and we can't sort them out.



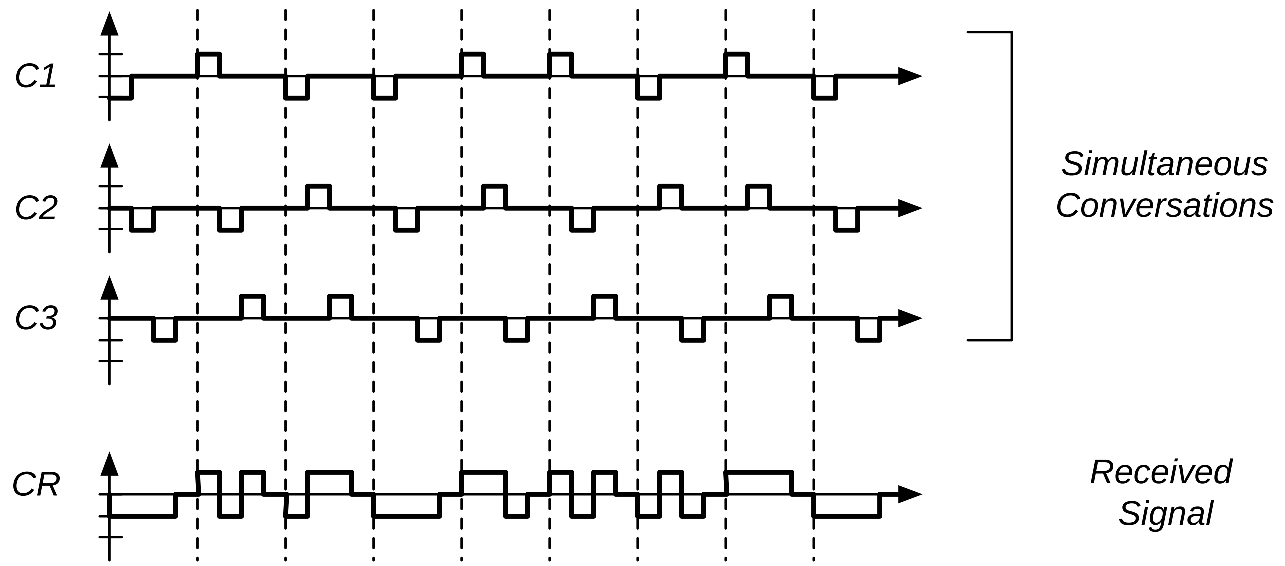
Two methods (out of many) for sharing the channel are

- Time division multiplexing (TDMA)
- Code division multiplexing (CDMA)

Time Division Multiplexing

Each transmit bit is divided into several subintervals.

Each handset only talks during its allotted times.



Here there are four time slots. C1 only transmits during the first time slot, C2 the second time slot, etc.

There are a limited number of time slots, limiting the number of simultaneous users. We could add one more user, but the second would fail.

The received signal bandwidth is increased by a factor of four in this case (the spectrum is spread by that factor).

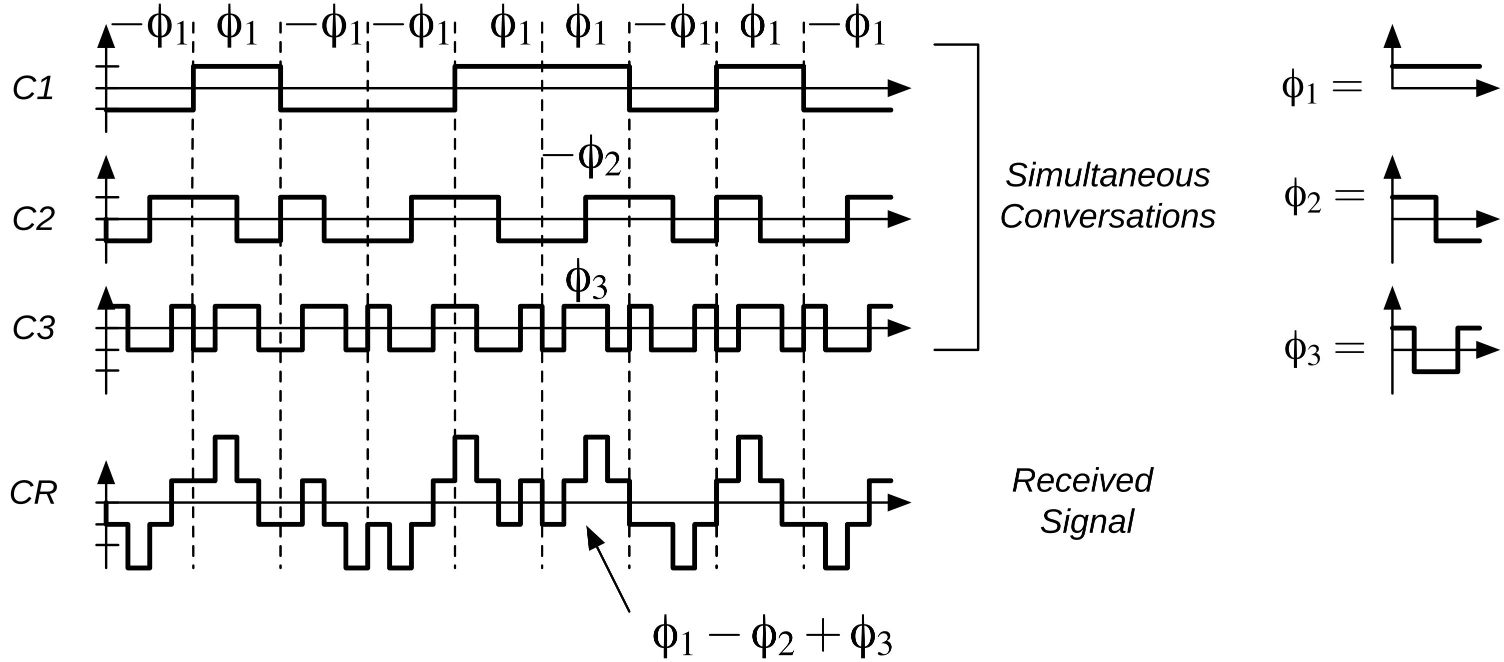
Higher data rates can be obtained by allocating one user several time slots.

You can transmit and receive on the same frequency by allocating different time slots to each. This is done in some of the Chinese cell phone systems.

Identifying Individual Channels

What we'd like is a "tag" that allows each channel to be selected, while ignoring all of the others.

If $\phi_1(t)$ is the tag, or code for one channel, we send a sequence of $(-\phi_1(t), +\phi_1(t), +\phi_1(t), \cdots)$ to represent the sequence $(-1, +1, +1, \cdots)$



The received signal at one interval contains these superimposed codes from each channel, weighted by channel gain, transmit power, etc

$$y(t + kT) = \sum_n D_{n,k}\phi_n(t)$$

$D_{n,k}$ is the information from the n^{th} channel at the k^{th} sample.

How do we choose the code waveforms to make it easy to find the signal of interest, and ignore all the others?

Orthogonal Codes

What we want is to choose the $\phi_n(t)$ so that we can extract $D_{n,k}$ (the information we are trying to receive) by a simple matched filter,

$$\begin{aligned}
\tilde{D}_{n,k} &= \frac{1}{T}\int_0^T \phi_n(t) y(t+kT) dt \\
&= \frac{1}{T}\int_0^T \phi_n(t) \sum_m D_{m,k}\phi_m(t) dt \\
&= \sum_m D_{m,k} \frac{1}{T}\int_0^T \phi_n(t)\phi_m(t) dt
\end{aligned}$$

If we choose

$$\frac{1}{T}\int_0^T \phi_n(t)\phi_m(t)dt = \begin{cases} 1 & \text{if n=m} \\ 0 & \text{otherwise} \end{cases}$$

Then we only get the term we get is the term we want,

$$\tilde{D}_{n,k} = D_{n,k}$$

and we ignore all of the other channels!

We want to choose $\phi_n(t)$ to be *orthogonal* signals.

There are many possible choices. For example

$$\phi_n(t) = \cos(n\omega_0 t)$$

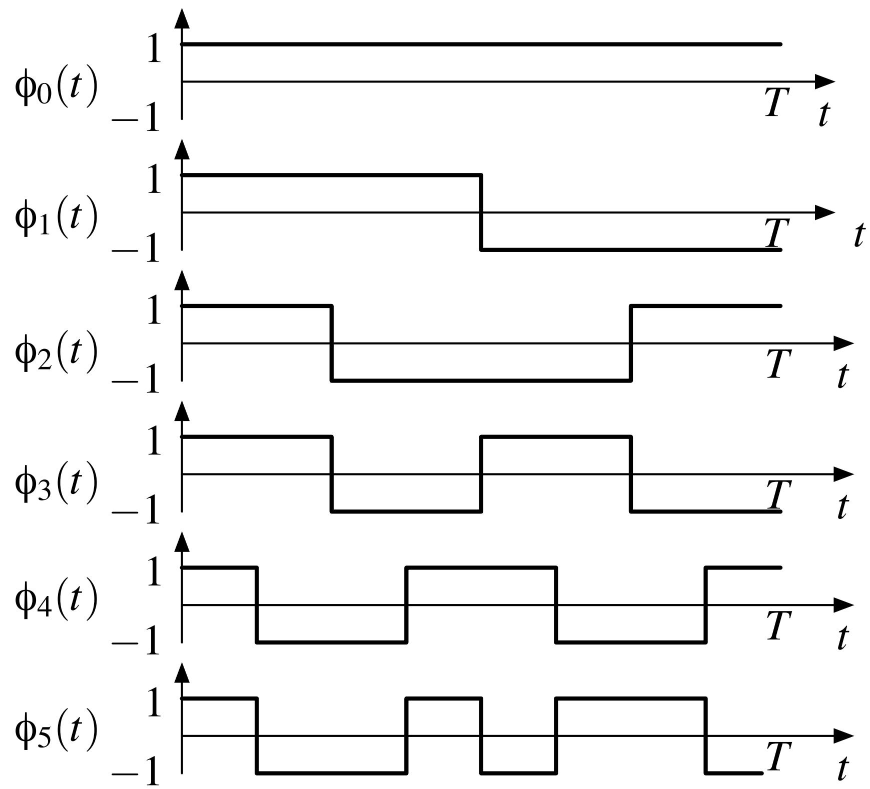
where $\omega_0 = \frac{2\pi}{T}$. The $D_{n,k}$ are then the coefficients of the cosine Fourier series of the signal. This effectively shifts each channel to a different frequency! This is similar to FSK, and often uses the same waveforms.

However, what we *really* want is something very easy to compute ...

Hadamard Waveforms

Hadamard waveforms are an orthogonal set made up of only ± 1's.

The first couple of Hadamard signals are (in sequency order):



The signs of the Hadamard signals can be generated (in a different order) by defining

$$H_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

and then defining

$$H_4 = \begin{pmatrix} H_2 & H_2 \\ H_2 & -H_2 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}$$

and similarly for $H_8, H_{16}, ...$

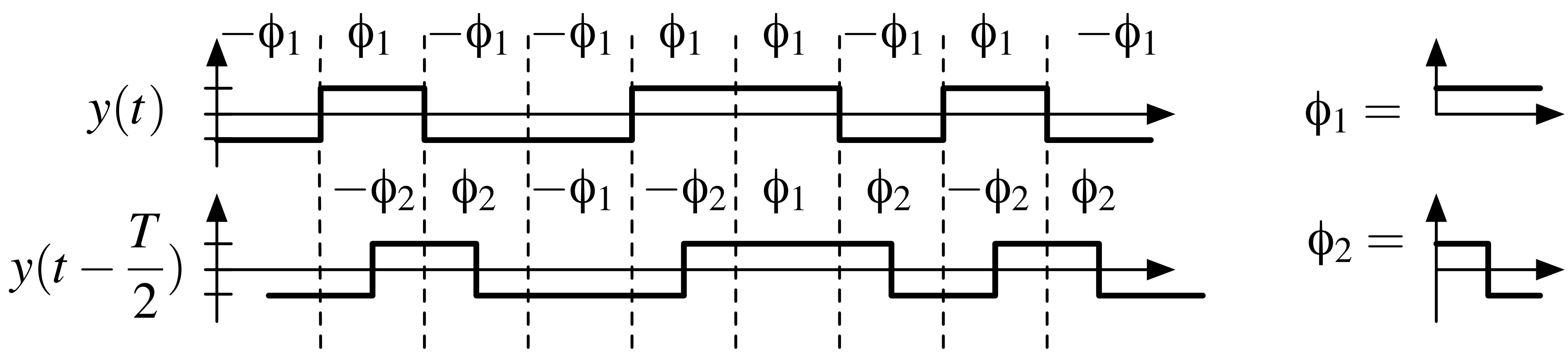
It is easy to show that the columns of H_N are orthogonal, and hence that the Hadamard functions with these signs are orthogonal (try this!).

We can tag each channel by assigning it a Hadamard waveform $\phi_n(t)$.

We can continue to add users by adding new codes (at some point we run out of bandwidth, though).

Synchronous Detection

Hadamard decoding fails if it is not synchronized

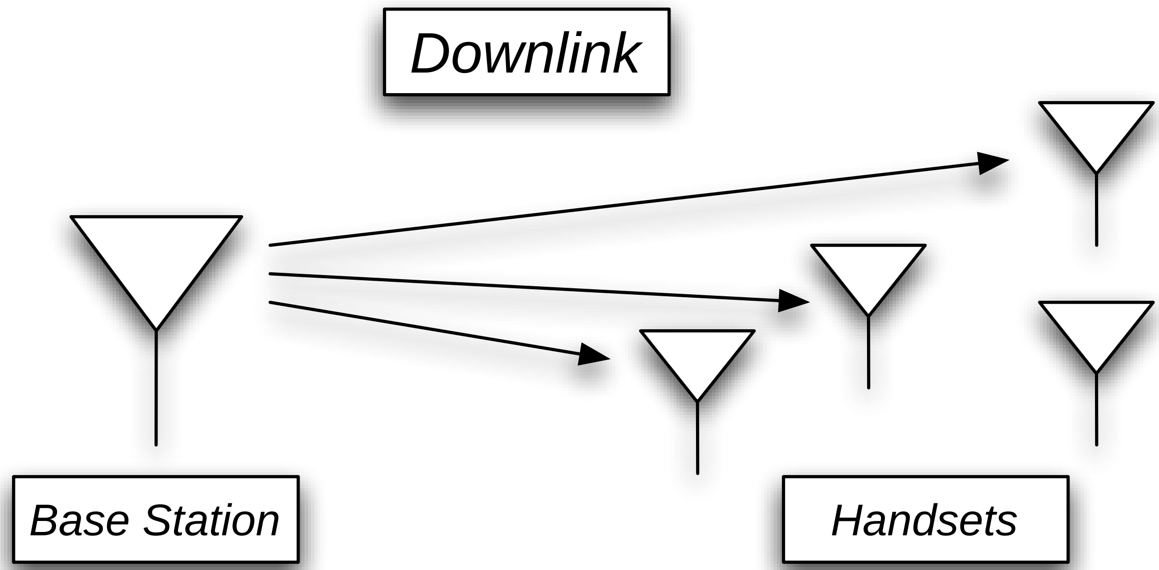


Without a delay, the signal decodes to channel 1.

With a delay, the decoding is completely different.

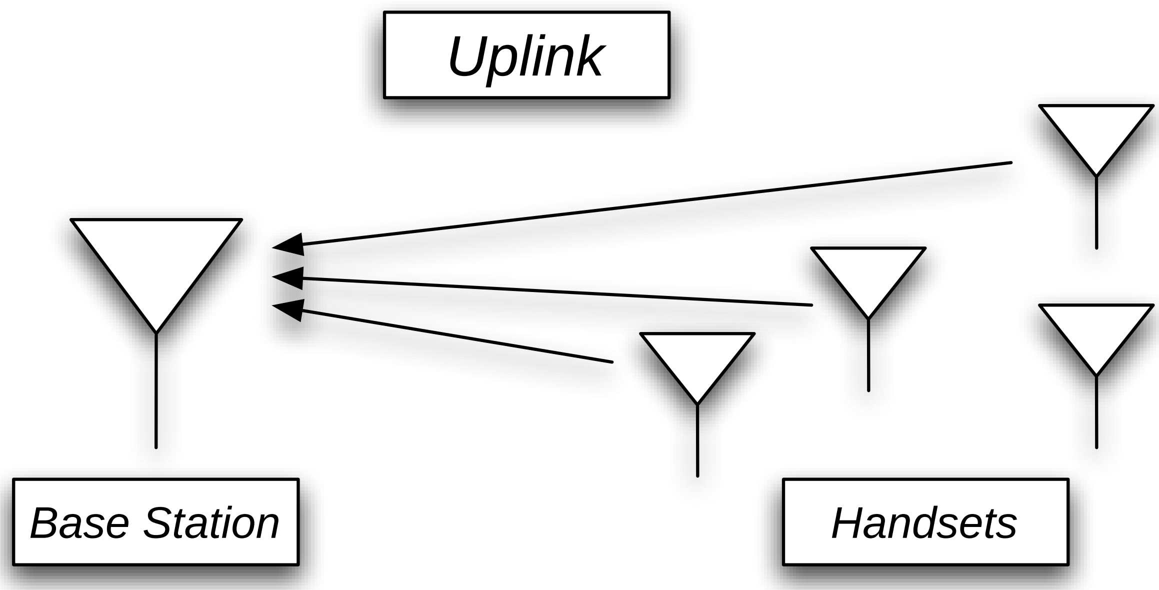
The Hadamard codes only work if all of the signals are synchronized.

Fine when the basestation is talking to the handsets (the downlink). The basestation can synchronize all the signals before transmitting.



Does not work well when the handsets are talking to the basestation (the uplink).

Each channel has it's own delay:



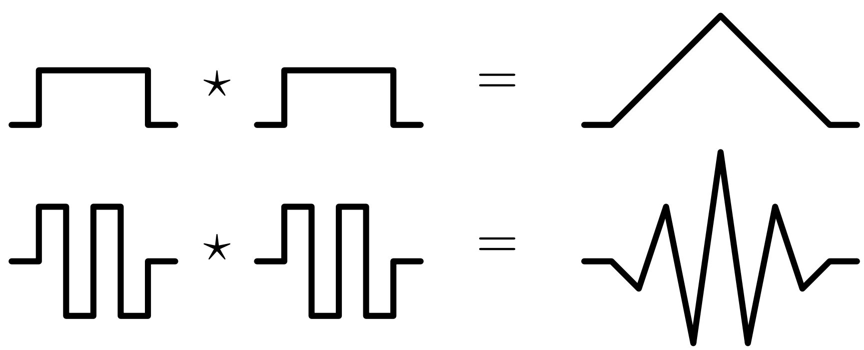
Even if we could synchronize to one channel, we would be decoding the other channels with delays, and these interfere with our channel.

Uplink Codes

Ideally, we would like a code that

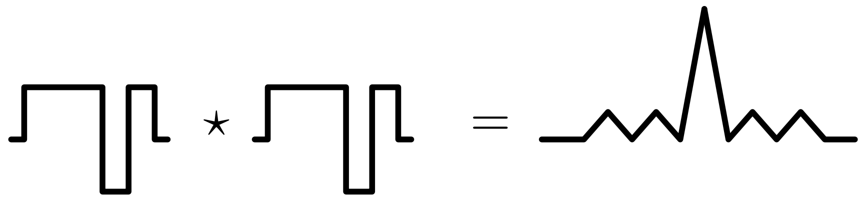
- Is orthogonal, so that we can isolate a particular channel
- Has a sharp autocorrelation, so that we can accurately identify the delay of a channel, and decode it properly.

Unfortunately, Hadamard codes have broad autocorrelations:



Not good for estimating delays!

We would like autocorrelations like the Barker codes,

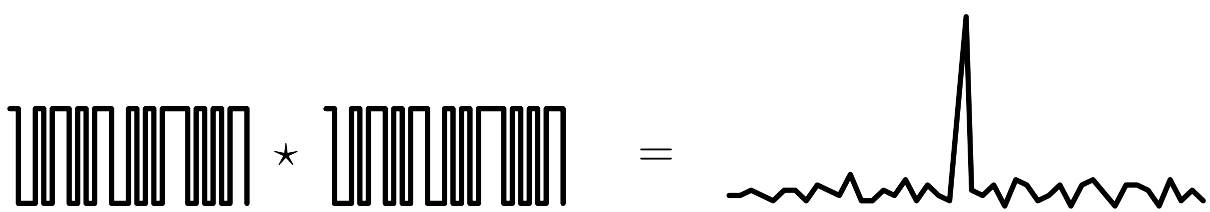


However, there are no orthogonal sets of signals with the autocorrelation properties of Barker codes.

Pseudo-Noise Codes

Pseudo-noise codes approximate the ideal sharp auto-correlation, orthogonal codes. These are deterministic codes that approximate the characteristics of a noise sequence.

They are ± 1 with probability of 1/2, and have the run lengths of -1's and 1's of a random sequence.



At zero shift, the product of the two is a contant "1", and the autocorrelation is "N". At other shifts, the product of each interval is just is likely to be $+1$ as -1, so the autocorrelation is small.

These are pseudo random noise (PRN) codes. One common example are Gold codes.

They are not completely orthogonal, although

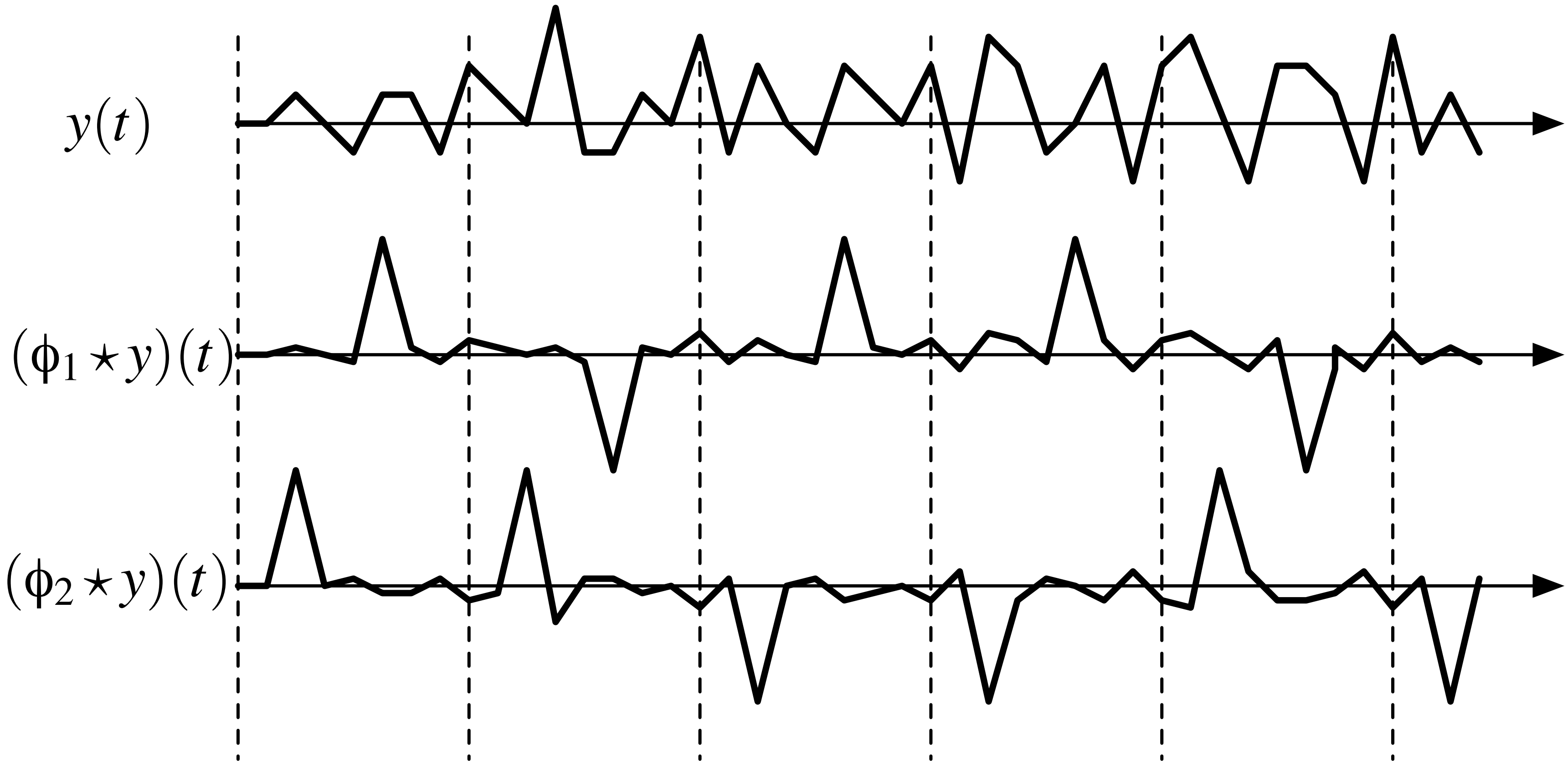
$$\frac{1}{T}\int_0^T \phi_n(t)\phi_k(t)dt$$

should be small (again, ± 1 equally likely, so the integral will be small).

Other channels appear as noise-like interference.

If we correlate the received signal with $\phi_n(t)$ we get a large signal for the n^{th} channel, with a peak at the delay for the n^{th} channel.

Adding users looks likes like an increased background noise level, which softly degrades performance.



Each channel properly decoded, with its own delay.

Other channel just adds noise.

Spread Spectrum Communications

Orthogonal codes allow users to share a channel.

Hadamard codes work well when the channels can be synchronized, such as when the basestation is talking to multiple handsets.

For unsynchronized channels, we want both orthogonality, and good autocorrelations. Psuedo-random codes are a good approximation.

We can continue to add users by handing out more codes, with a soft degradation of performance. Other users look like an increased noise level.

For GPS, each satellite has its own 1023-bit Gold code, and all transmit on the same frequency.

There are also other approaches to spread-spectrum communications, such as frequency hopping. This used for robustness (resistance to jamming) and security.