Digital Communications Transmission

by Michael Prior-Jones
based on the course taught by Alister Burr.

Introduction

This course aims to cover some of the techniques used in the transmission of digital signals over cables or radio links. The first part deals with *baseband transmission* (i.e. transmitting a signal at its original frequency, usually over a cable), whilst the second part deals with *bandpass transmission*, which is using modulation to transmit a signal within a specified range of frequencies.

Baseband transmission

Firstly, some assumptions that we're making:

- we're trying to transmit a serial binary datastream.
- we're transmitting at baseband; i.e. without changing the frequency of the signal.

This sort of technique is almost exclusively used over cable systems, such as in telephone systems and computer networks.

Bit Error Ratio

With a digital transmission, any noise or distortion in the system may result in *bit errors*, where a bit is corrupted in transit and is interpreted wrongly by the receiver. Communications engineers talk a lot about *bit error ratio* (BER for short) which is defined as follows:

> *BER is the average ratio of the number of bits received in error to the total number of bits received.*

BER is also equal to the probability of receiving a bit in error. In digital TV, the maximum error rate tolerable before the picture breaks up is 2×10^{-4}. This means that for every 10000 bits received, 2 are in error: i.e. the probability of a bit error is 1 in 5000. Whilst this might seem quite long odds (you wouldn't back it at a racecourse!), the data rate is typically 25 Mb/s, so in one second you'll receive 25 million bits, of which an average of 5000 will be wrong. Error correction techniques are used to reconstruct the picture in the presence of errors, but I'm not going to talk about them now!

Signal power

Because communications is about transmitting energy, people tend to talk about signals in terms of *power*. Think back to GCSE physics (that's a long way!) and you should remember that power is the rate of change of energy; or the energy expended in unit time. If you're building some sort of radio transmitter, whether it's a 1W mobile phone or a 1MW TV transmitter, then you'll want to be making the best use of the power you have available. For this reason, people analyse the *power spectra* of signals, looking for power being transmitted that's not contributing to the overall performance of the system. More on this later.
Just to remind you, here's how to find signal power from signal voltage:

$$P(t) = \frac{v^2(t)}{Z}$$

Your signal voltage is *v(t)* and *Z* is your system impedance. Since impedance tends to be constant throughout a system (unless it's really weird!), we tend to normalise it out, so that our normalised power is just $v^2(t)$, the square of the voltage. You can always put it back later.

Signal Spectra

Cast your mind back to Fourier theory, and you'll remember that any signal can be decomposed into the sum of a series of sine waves. Plotting the amplitudes of these component waves against their frequency gives a signal spectrum. As is customary in communications, we tend to calculate the power of each component at a specific frequency and plot that instead. This is called *power spectral density*, and has units of Watts per Hertz.

Distortionless transmission

Ignoring noise and interference for the moment, let's consider the influence of our channel on a signal. At the most fundamental level, we need to compare the frequency response of our channel with the spectrum of our signal. If the signal spectrum remains within the flat region of the channel's frequency response, then the signal won't distort. Otherwise, the channel will distort the signal. Low-pass channels tend to integrate a square signal, distorting the edges of the pulses. High-pass channels differentiate, distorting the tops of the pulses. Depending on its severity, channel distortion may affect the reception of the signal.
If you're experiencing channel distortion, you have two basic ways to get rid of it:

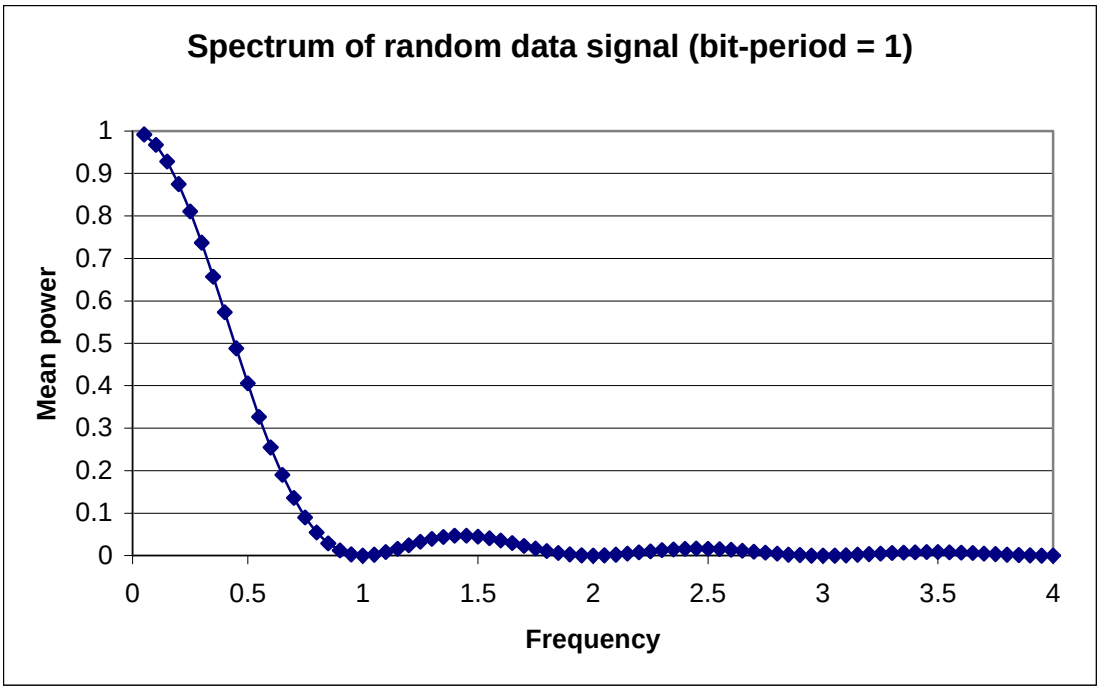
- change the channel frequency response to a more suitable one: in cable technology, this means buying a more expensive cable or using equalisers to compensate the frequency response. In radio systems, this is usually impossible!
- change the signal spectrum to a more suitable one for the channel, using modulation or line coding.

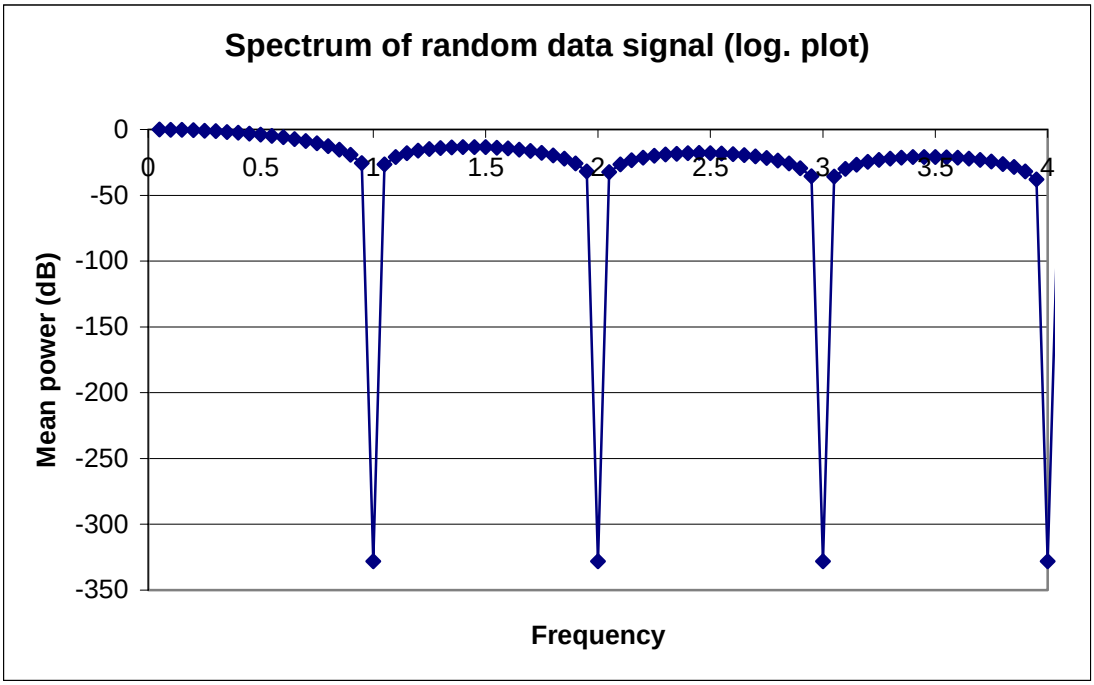
The spectrum of a random data signal

It's useful to know the spectrum of a "typical" data signal, so that we can see if it will fit into the channel bandwidth. There's a standard for this, and it's based around the random bitstream.

Imagine generating a whole lot of totally random binary data and then transmitting it with logic high for a '1' and logic low for a '0'. This is known as binary non-return-to-zero (NRZ) and is about the simplest means of transmission available. This gives a succession of random pulses. To get the spectrum of this random signal, we look at its *mean power spectrum*, which you can calculate by taking the Fourier transform of a single data pulse and then squaring it. There's some slightly bizarre maths involved in finding this, but I'm not going to go into it now.

The nulls in the spectrum are at multiples of the bit-rate. In the plots below, the bit-period is 1 second, so the bit-rate is 1Hz. The portions between the nulls are called *lobes*. The largest one is called the *main lobe* and contains most of the energy in the signal. It's quite common to filter off the other lobes (*sidelobes*) and just transmit the main one, although this (obviously) introduces a certain amount of distortion in the time domain. This makes the original square wave look more sinusoidal.


Spectrum of random data signal (bit-period = 1)
Mean power
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.5
1
1.5
2
2.5
3
3.5
4
Frequency


Spectrum of random data signal (log. plot)
Mean power (dB)
0
-50
-100
-150
-200
-250
-300
-350
0
0.5
1
1.5
2
2.5
3
3.5
4
Frequency

Pulse shapes and line codes

The most basic digital line coding system is that used in logic circuits- logic '0' is represented by 0V and logic '1' by a higher voltage (often 5V). This is called *binary non-return-to-zero* (NRZ) and is only used for communicating over short distances at low frequencies, if at all. It has a number of drawbacks:

- the signal contains some d.c. which isn't actually carrying any signal, just wasting power.
- the signal spectrum is inefficient- it occupies a wide bandwidth with its many sidelobes.
- the receiver may experience sychronisation problems- a long run of '0's or '1's may cause the receiver to lose the accurate timing it needs to interpret the data properly.

There are all sorts of methods for eliminating these problems, although most of them are compromises: for instance, a very robust system is likely to use more bandwidth and have a more complex receiver design than a simple system. It all depends what your priorities are. Optical fibre, for instance is an almost-ideal transmission medium. There's little noise and loss and a very large bandwidth. Short-wave radio, however, is full of noise and interference and bandwidths are relatively narrow. Choosing your design is a matter of weighing up all the strengths and weaknesses of each system against the nature of the problem you're trying to solve.

Changing the pulse shape:

In NRZ we used a rectangular pulse to represent a logic '1'. One simple option is to change this to have a shorter pulse followed by a gap representing '1'. This is called *return-to-zero* signalling (RZ) and has twice the main lobe bandwidth of simple NRZ, but it is easier to decode and slightly more noise-tolerant.
You can use any pulse shape you like- and we'll come back to this later. To find the spectrum, just take the square of the Fourier transform of the pulse shape.

High-frequency distortion:

If your channel has a poor high-frequency response (copper cable, for example) then the transitions between the pulses will be affected: instead of a clean, sharp transition, you'll get an exponential curve which can blur the distinction between two pulses. This effect is known as *intersymbol interference*.[1] To avoid this, you can filter the signal to restrict it to the flat region of the frequency response. This sounds fine, except that it makes the pulses ring (oscillate) into subsequent pulses!
However, your receiver's going to sample your signal at defined intervals, so as long as the ringing from previous pulses has died away by the time you take your sample, then you'll get an accurate result. This technique was described by the ubiquitous Mr Nyquist, and is thus called the *Nyquist Criterion*. I've just described it in terms of time, but we'll come on to seeing how it affects the frequency domain shortly.

[1] *Symbol* is the term used for a particular state of a line code or modulation scheme. NRZ has two symbols- +V and 0V, and is described as being *one bit per symbol*. Other schemes (mostly RF schemes) have more than one bit per symbol.

It actually boils down to saying that you can transmit a datastream at a bitrate R_b in a bandwidth of $R_b/2$, although this is incredibly difficult in practice!
In the frequency domain, the spectrum needs to have *vestigial symmetry* about the frequency $R_b/2$, which is a form of odd symmetry (think back to A-level maths).
A filter which is used to ensure that this condition is met is called a *Nyqist filter* and the pulses that it generates are called *Nyquist pulses*.

Raised-cosine Nyquist filtering:

In order to fulfil the Nyquist criterion, we usually use a filter to create pulses with smooth edges. Mathematically, this is done with a *raised cosine function* which is just a cos wave centred on 50% of the pulse amplitude. The width of the roll-off region is defined by the *roll-off factor* which is usually expressed as a percentage. The signal bandwidth in this case becomes:

$$W = r_b \frac{1+\beta}{2}$$

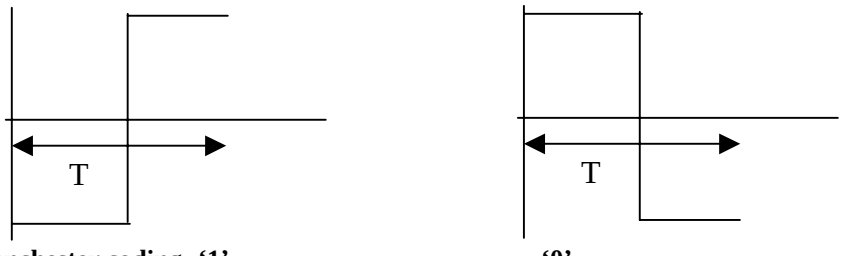
where β is the roll-off factor, varying from 0 to 1.
Adjusting the roll-off allows you to control the signal bandwidth- 100% roll-off gives almost no ringing, but makes the bandwidth equal to the data rate. It's possible to use lower values of roll-off, but this increases the risk of intersymbol interference.

Low frequency distortion:

If, conversely to what we've described above, you have a channel with poor low frequency response, the signals begin to droop during the symbol periods. It's worst when you get a long run of identical symbols (usually '1's) because at the end of the run, the last few '1's may be interpreted as '0's.
These problems can be reduced by:

- eliminating any d.c. coefficient in the signal (this also improves transmitter efficiency)
- coding to ensure that long runs at the same voltage level are avoided.

One of the most common methods for doing this is *Manchester coding*, which is used on the original 10Mb/s Ethernet. Manchester pulses look like this:



Manchester coding- '1' **'0'**

Manchester coding has no d.c. term- it averages out to zero volts, and the longest run is 2 symbols. However, the theoretical minimum bandwidth is twice that of NRZ. The theoretical redundancy is 50%.
Because of the frequent transitions in Manchester coding, it's very easy to keep the receiver's clock in synchronisation with the transmission. However, some coding

schemes actually carry a clock component, and the effect of this is to produce impulses in the frequency response at multiples of the clock frequency. Unipolar RZ is one such scheme. How do you tell ? Well, invent a random data sequence with equal numbers of '1's and '0's and draw it out in your chosen coding scheme. Then draw a bipolar clock waveform onto it with a period equal to the shortest period in your data waveform. Mark the half-cycles with '+' if the clock and data are in phase and with a '-' if they're not. Then add up the numbers of '+' and '-'; if they're equal, there's no clock component present.
Having a clock component as part of your signal wastes transmission power (and possibly bandwidth as well), although it makes *clock recovery* at the receiver much easier!

An alternative to Manchester coding for signalling on optical fibres is Alternate Mark Inversion, which is simple NRZ but with alternate '1's at alternate polarities: +1, -1, +1, -1 and so on. This has a theoretical 38% redundancy and fits into the same bandwidth as NRZ, but has a very small amount of low frequency content.

Optimum Reception & Channel Noise

Okay, so you've decided on a suitable transmission format for your channel. Now what? Well, whilst your transmission format may give perfectly distortionless transmission in the available bandwidth, you have to allow for the presence of noise… Noise comes in a wide variety of forms, but (for simplicity, and to avoid doing lots of awful probability maths) we'll stick to Additive White Gaussian Noise. This (if you're really interested) has a Gaussian (or Normal) distribution curve and a flat spectrum. It's simply described in terms of its *power spectral density*- the noise power in a 1Hz bandwidth.
When designing a receiver, it's usually sensible to put a filter at the input, to remove as much out-of-band noise as possible. But what response should the filter have ? The answer is (fairly obviously) that the filter's frequency response should be the same as the signal's spectrum. Such a filter is called a *matched filter* because it's matched to the signal (don't confuse this with transmission-line impedance matching…) and it delivers the best possible signal-to-noise ratio.
In fact, assuming a perfectly matched filter, the output SNR is as follows:

$$SNR = \left(\frac{2E}{N_0} \right)$$

where E is the energy in a single signalling pulse (ie signal power x symbol period) and N_0 is the input noise power spectral density.

Implementing such a filter, however, is very difficult. You can approximate with analogue filters, although these are rather difficult to design. Most people needing a matched filter take the brute force of a digital signal processing chip to it, and process the signal directly in the time domain. DSP's beyond the scope of this course, but it is jolly clever!

There is an interesting alternative to a matched filter- the *correlation detector*. This uses a multiplier and integrator to correlate the incoming signal against a reference signal (probably generated by some clock recovery circuit further down the line) and produce a clean output. The need for a reference signal tends to make this technique rather limited in application.

Sampling, Thresholds and Eye Diagrams

The conventional *decision detector* uses a sample-and-hold circuit to hold the signal level steady whilst a comparator compares it against a threshold to decide whether it's a '1' or a '0'. But when should you sample, and what should be the threshold ? The *eye diagram* can tell you the answer. You can see them on an oscilloscope by triggering the 'scope from the same clock signal as your datastream, and then looking at what comes out of your receiver filter. What you see is several waveforms all drawn on top of each other, giving gaps (*eyes*) between the paths of the waves. Where the eye is at its widest is the best place to sample, and the ideal threshold is midway between the average '1' level and the average '0' level. This does assume that '1's and '0's are equally probable and the noise on the channel is unaffected by the data.

Calculating Bit Error Ratios

It's possible to calculate a theoretical Bit Error Ratio (BER) based on knowing your coding scheme and the channel noise power.

Assuming that your threshold is halfway between '1' and '0', that you have a matched filter and that your noise is Gaussian, then:

$$BER = Q\left(\frac{V_1 - V_0}{2\sigma}\right)$$

where sigma is the noise distribution's standard deviation. Q is the (rather awkward) Q function, that doesn't integrate, and so has to be found numerically or graphically. Q is defined as follows:

$$Q(z) = \int_z^\infty \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) dx$$

You're always given a Q-function graph in an exam…

Finding sigma is slightly easier: sigma squared is equal to the mean square noise power at the output of your matched filter, so:
since our signal to noise ratio is (as earlier):

$$SNR = \left(\frac{V^2}{\sigma^2}\right) = \left(\frac{2E}{N_0}\right)$$

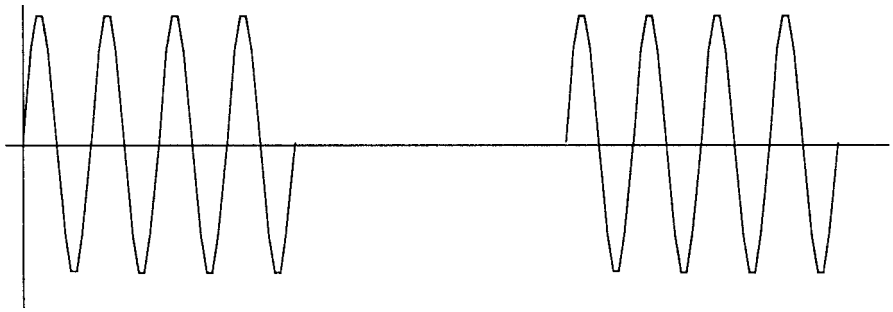
You can substitute into the BER equation according to the nature of your coding scheme. This depends on the values of V_1 and V_0 (for unipolar codes, V_0 =0, for polar codes it's $-V_1$). You also need to substitute E (the energy per pulse) for E_b (the average energy per symbol period).

Bandpass transmission

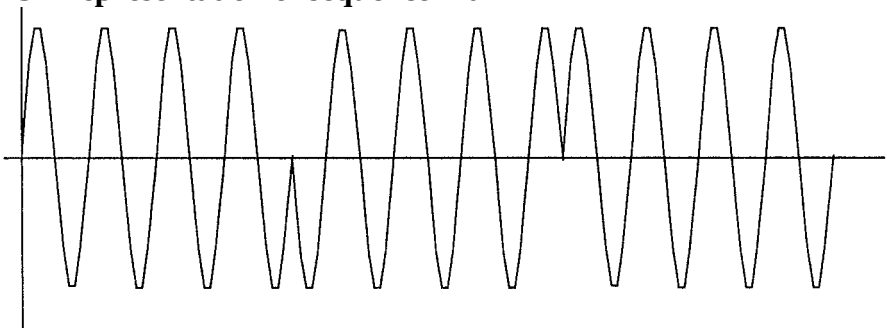
Bandpass transmission is (usually) concerned with radio, where you're trying to fit your signal a bandwidth specified by the regulatory authorities. Conventionally, one *modulates* the signal onto a *carrier* at a specific frequency. Since a bog-standard sine wave has three basic characteristics, amplitude, frequency and phase, you can vary these in sympathy with your signal to generate your modulated output.
Modulation schemes fall into two basic camps- *linear modulation* and *exponential modulation*. Linear schemes vary the amplitude and/or phase of the carrier, and don't affect the frequency- they have a mathematically linear relationship between signal, carrier and output. Exponential schemes have an exponential relationship (obviously!) and can vary the frequency of the output.

Linear modulation: ASK and BPSK

Amplitude modulation is the easiest to think of- just vary the amplitude of the carrier wave in sympathy with the amplitude of the signal. ASK (amplitude shift keying) is the result of multiplying a unipolar NRZ signal by the carrier- a '1' is represented by a burst of carrier and a '0' by 0v. This isn't a particularly fantastic modulation scheme as we'll see shortly. The alternative is BPSK (binary phase-shift keying), which is the result of multiplying a polar NRZ signal by the carrier- the difference between '1' and '0' is a 180 degree phase shift in the carrier.



ASK representation of sequence '101'



BPSK representation of sequence '101'

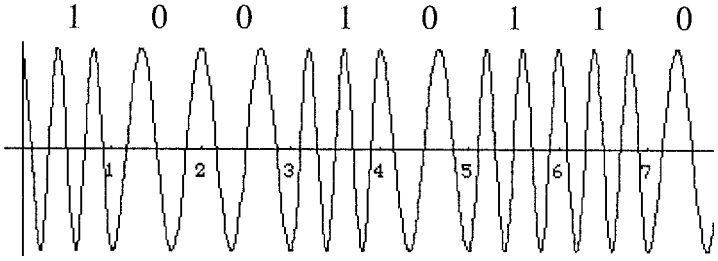
The spectra of these schemes are very similar- they look like the baseband spectra, only shifted up to the carrier frequency and with a mirror-image spectrum on the other side of the carrier.
ASK generates a carrier component which wastes power. BPSK has an identical spectrum, but no carrier component.

Exponential modulation: FSK and MSK

This is rather more mathematically complex, as it can't be modelled as a simple "carrier x data" equation.
Continuous phase Frequency Shift Keying is an example: we modulate the phase of the carrier by making the rate of change of carrier phase depend on the data: for a '1', the rate of change of phase is positive, and for a '0' it's negative.
This rather complex maths leads to a waveform with a high frequency for a '1' and a low frequency for a '0' *and with smooth transitions between them*. Simple FSK can be implemented linearly (early computers did this- remember the whining noise from your cassette player whilst it loaded *Manic Miner*?) by simply switching abruptly between two frequencies.
The modulation index of continuous phase FSK is defined as:

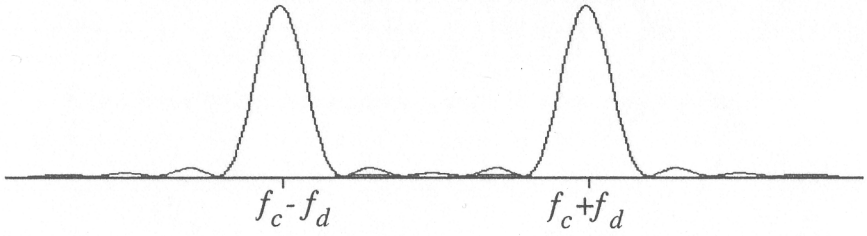
$$h = 2 f_d T$$

where f_d is the amount of frequency shift and T is the bit period.



Waveform of continuous-phase FSK

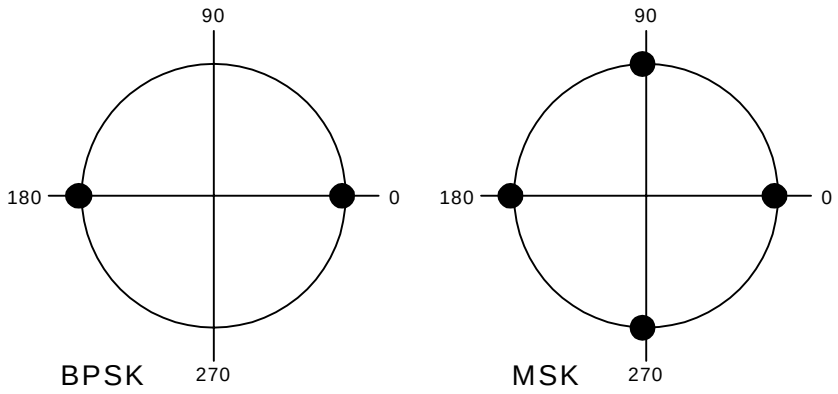
If h is much greater than 1, then you have wide-shift FSK (WS-FSK) where the spectrum consists of two sinc-pulse peaks at a distance f_d from the carrier.



Spectrum of WS-FSK

If you make h = 0.5, the result is Minimum Shift Keying (MSK) which doesn't vary frequency at all- it simply changes phase. When data ='1', then the carrier phase changes by +j and when it's '0' the phase changes by –j. This is most easily seen on a *constellation diagram*, which is simply a phasor diagram of all the possible states of the carrier. Remember that phasor diagrams show amplitude as distance from the origin, and phase as angle from the positive x-axis.

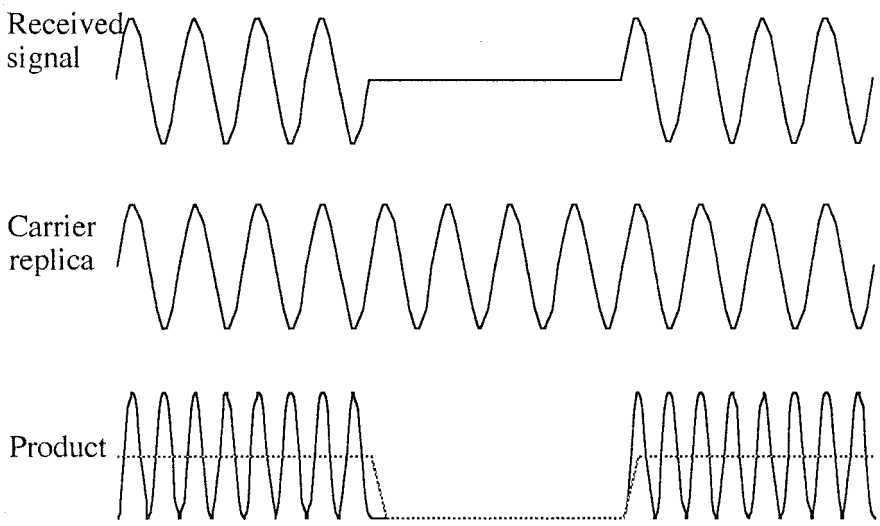
Here are a couple of constellation diagrams to give you the idea:



On the left you can see that BPSK has two states at equal amplitude but 180 degrees out of phase. On the right, MSK has four states at equal amplitude, each 90 degrees out of phase. MSK uses relative phase encoding- a '1' means change state 90 degrees anticlockwise, and '0' means change state 90 degrees clockwise.

Demodulation technqiues

You can demodulate a signal in two ways- called *coherent* and *non-coherent* demodulation. Coherent demodulation is rather mathematical- you multiply the received signal by the original carrier and then filter to remove the high frequency components. This requires you to either transmit the carrier, or recover it from the signal by some other means. Such circuitry is called *carrier recovery* and is often complex. It's also beyond the scope of this course (thankfully!)
Coherent demodulation allows the receiver to compare the phase of the signal with that of the carrier- this is called *absolute phase* information.



Coherent demodulation, using a replica carrier.

Non-coherent demodulation is much simpler, as there's no carrier recovery to worry about. ASK can be demodulated by an *envelope detector*- just a diode followed by a low-pass filter. The time constant of the filter should be:

$$RC = \frac{T_{bit}}{1.5}$$

where T_{bit} is the bit-period.

BPSK can be demodulated by using a time-delay circuit to compare each bit-period with the previous one and thus recover the data. This is known as *differential phase shift keying (DPSK).*

The effects of noise

In bandpass transmission, we treat noise as a carrier at our frequency of interest, modulated with random amplitude and phase. There's lots of maths involved in seeing what happens when noise passes into your demodulator, but the fundamental result is this: if you have a *coherent* demodulator which is followed by a matched filter, the whole system behaves exactly as if the modulation wasn't there- as if you'd just transmitted the signal at baseband. The signal-to-noise ratio at the output of the matched filter is still:

$$SNR = \left(\frac{V^2}{\sigma^2}\right) = \left(\frac{2E}{N_0}\right)$$

In the same way, the BER calculation for BPSK is exactly the same as that for polar NRZ, and that for ASK is the same as that for unipolar NRZ.

	BPSK	ASK
Coherent	$BER = Q\left(\sqrt{\frac{2E_b}{N_0}}\right)$	$BER = Q\left(\sqrt{\frac{E_b}{N_0}}\right)$
Non-coherent	$BER = \frac{1}{2}e^{\left(-\frac{E_b}{N_0}\right)}$	$BER = \frac{1}{2}e^{\left(-\frac{E_b}{2N_0}\right)}$

Demodulating FSK works in much the same way: you demodulate WS-FSK coherently as two ASK signals and then combine the results. It has the same BER performance as ASK, in terms of average bit energy, but it requires only half the peak it energy. MSK can be demodulated by using two reference carriers at 90 degrees to each other (*in quadrature*), and this has the same performance as BPSK.

Multiple Access

The principles of multiple access are more thoroughly discussed in Dave Pearce's Networks course. However, I'll summarise the salient points:

Multiple access is about letting multiple users at various different locations share a common channel. This can be done in several ways, although only two are discussed here. In time-division multiple access (TDMA), the users take it in turns to transmit. Each user has control of the channel for a predetermined period, known as a *slot*. Users have to buffer their data until a slot becomes available for them to transmit in. On the other hand, it's possible to cut the channel up into a number of smaller-bandwidth channels (this is called frequency-division multiple access, FDMA) and you can then assign a particular frequency to each individual user (like stations on a radio dial). In theory, both FDMA and TDMA offer very similar performance, although FDMA can be prone to intermodulation problems due to non-linearities in the transmission amplifiers. This can be eliminated by not running the amplifiers at full power, but this wastes energy, which can be important in a battery-powered system. Both FDMA and TDMA have the same average power requirement, although TDMA requires a higher peak power as the number of users increases. FDMA is often combined with *cellular planning*, where frequencies are allocated in groups to transmitters, but such that adjacent transmitters are always on different frequencies (think back to Geography lessons- colouring in maps with different countries in diffe In practical communications systems, the cellular FDMA frequencies are themselves divided into time-slots using TDMA techniques (this is a *hybrid* system), so that a user gets exclusive use of a frequency for a defined period of time. This allows extremely large numbers of users relatively easily.