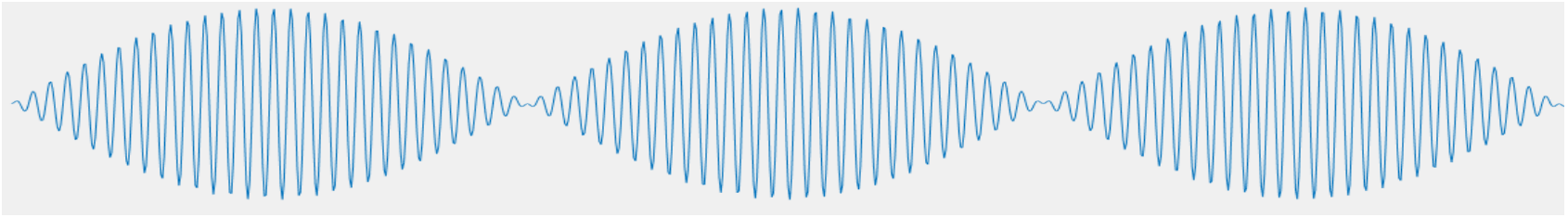
5G Air Interface – Part I
from NRZ to OFDM

Modulation

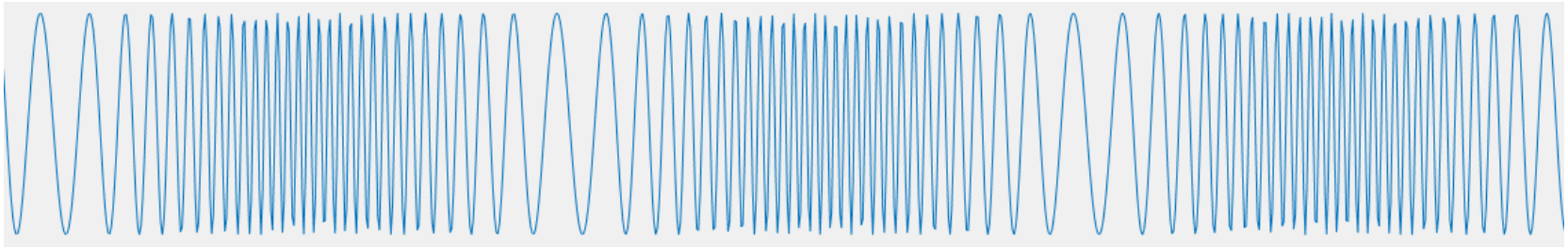
In order to understand how the air interface transports information
we need to understand the concept of modulation

Modulation is the variation of parameters of a (carrier) signal
in order to embed information in the (modulated) signal

For example, **A**mplitude **M**odulation is a kind of analog modulation
where we vary the amplitude of a sinusoid



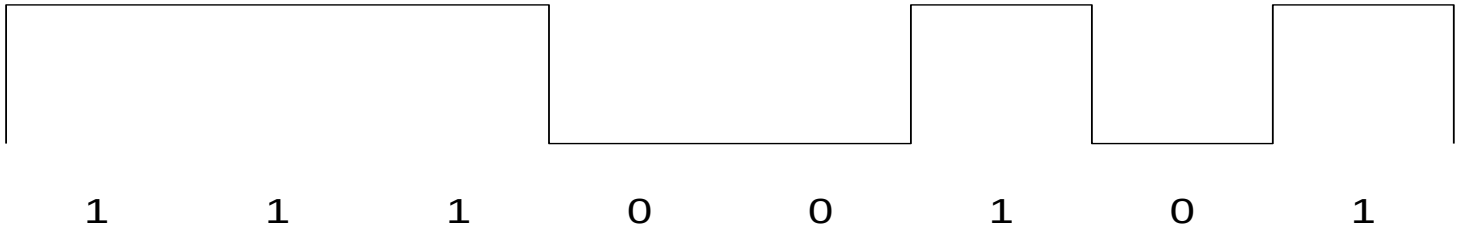
While in **F**requency **M**odulation we vary the frequency of a sinusoid



Digital modulation

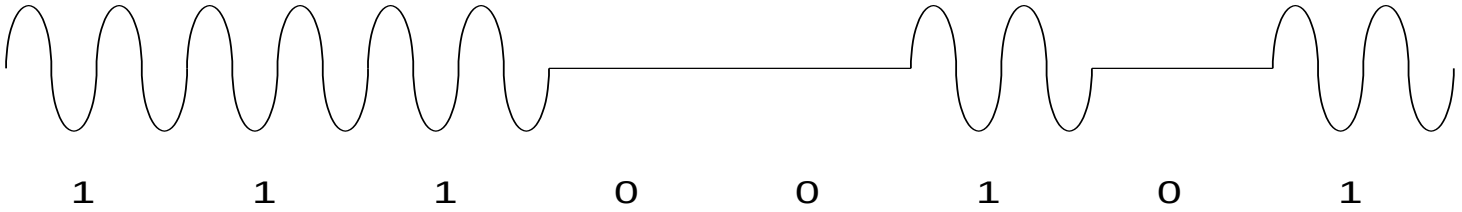
Here we are interested in digital modulation
that is modulating an analog signal to carry bits (not analog signals)

What most people envision is directly amplitude modulating DC with **0** and **1**



A technique called **N**on **R**eturn to **Z**ero
which may be used for short runs of copper wire

Slightly more advanced is to amplitude modulate a sinusoid with 0 or 1



Called **O**n **O**ff **K**eying
and still used for fiber optic communications (where DSP is expensive)

OFDM

The LTE air interface uses **O**rthogonal **F**requency **D**omain **M**odulation
which is the modulation type used by almost all modern systems, including:

- ADSL and VDSL (where it is called **D**iscrete **M**ulti**t**one **T**ransmission)
- G.hn home networking over twisted pair
- WiFi 802.11g (54 Mbps) and 802.11n (up to 600 Mbps)
- **D**igital **V**ideo **B**roadcasting – **T**errestial (digital TV)

OFDM is now preferred over other modulation techniques (including QAM)
due to its *efficiency* and *flexibility*

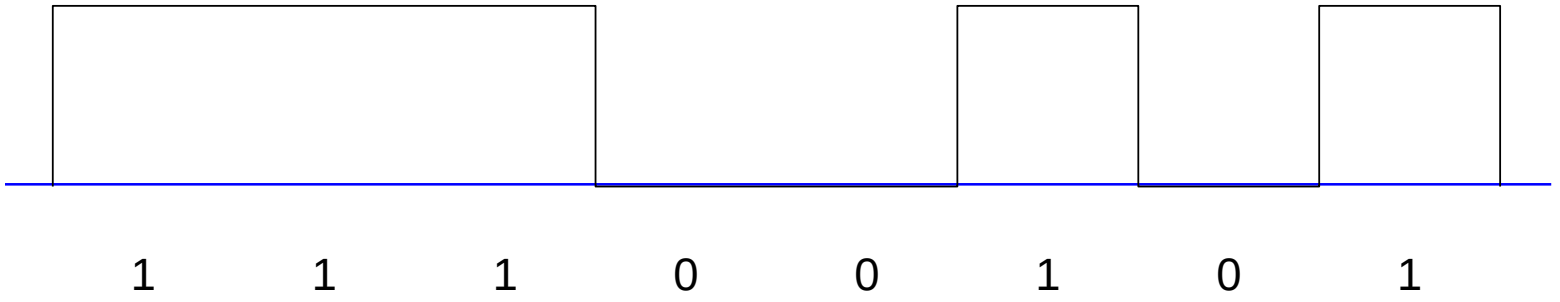
OFDM was invented and patented by Chang (Bell Labs) in 1966 (US3,488,445)
extended by Salzberg (Bell Labs) in 1967
made practical by Weinstein and Ebert (Bell Labs) in 1971
with further improvements by Cioffi (Stanford/Amati) in 1991

OFDM is related to FDM - the mux method used in the analog PSTN
and can be used to construct an efficient wideband modulation scheme
that can be naturally extended to a multiple access scheme

NRZ and PAM

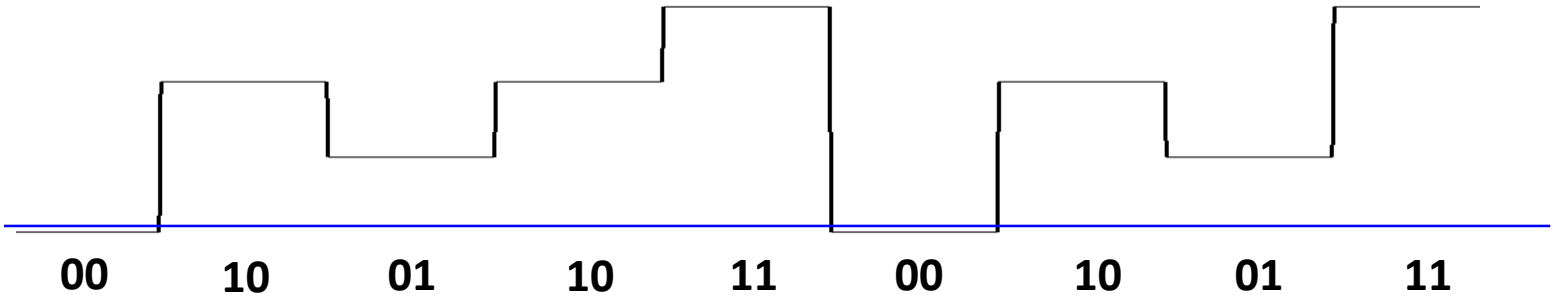
Let's start at the beginning to introduce the basic concepts!

NRZ encodes 0 and 1 using 2 possible signal values (e.g., 0 or 1)



NRZ is only applicable to media that can pass DC (base-band)

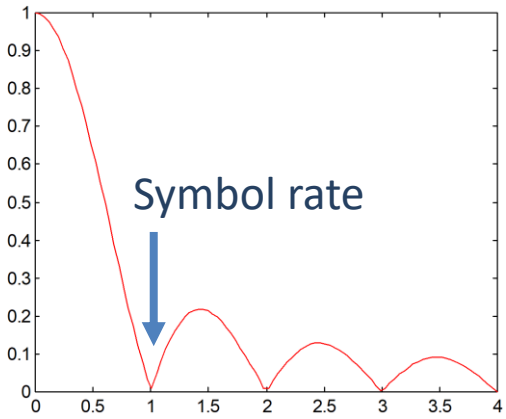
NRZ can be extended to multi-level **P**ulse **A**mplitude **M**odulation where each level represents a ***symbol*** (AKA *baud* after Jean-Maurice-Émile Baudot)



The number of bits per symbol is $\log_2$(number of possible symbol values)

NRZ/PAM spectrum

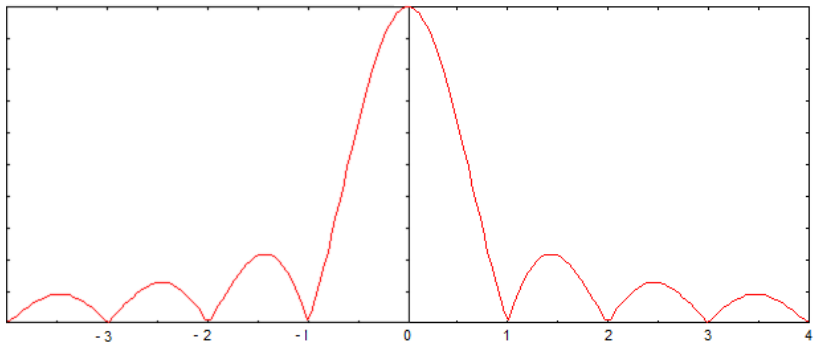
The spectrum of NRZ or PAM is (from the Wiener-Khinchin theorem)



The first zero is at the symbol rate

independent of the number of bits per symbol!

If we show the negative frequencies (not very interesting for real signals) we get



InterSymbol Interference

One problem impacting all modulation techniques is ISI

ISI occurs when one information-bearing symbol interferes with another

There are two main causes of ISI

1. limited bandwidth
   multiplying by a window in the *frequency domain*
   is equivalent to a *convolution* in the *time domain*
   which means that each symbol spreads into following symbol(s)

2. multipath
   when a signal propagates in space over multiple paths of different length
   a delayed symbol may be received during a subsequent symbol
   this results in a convolution
   with coefficients nonzero for physical delay differences

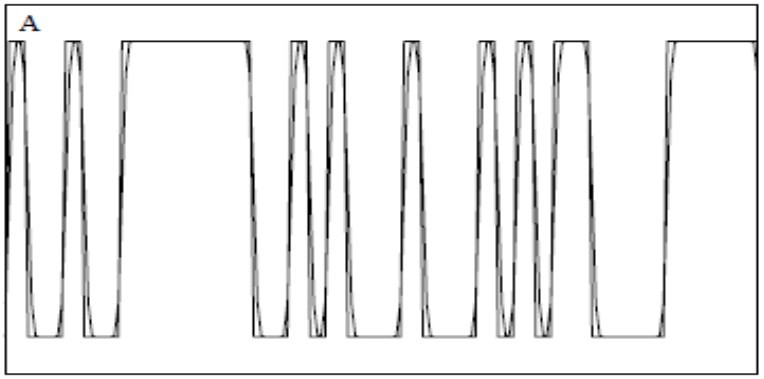
ISI increases with symbol rate
since a given delay overlaps more symbols at high rates

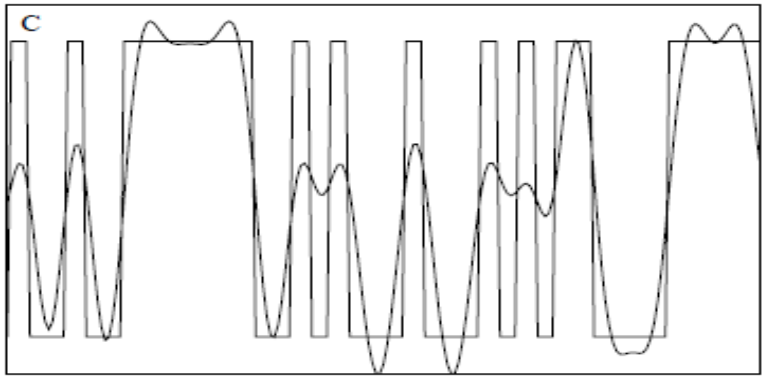
Limited Bandwidth ISI

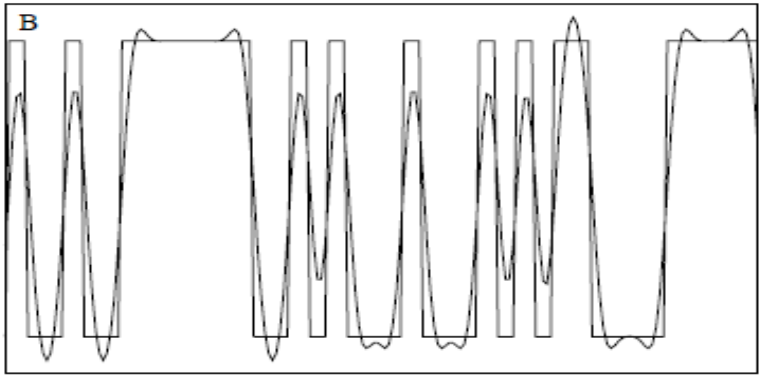
What happens if we try to squeeze an NRZ signal into a channel with insufficient bandwidth?

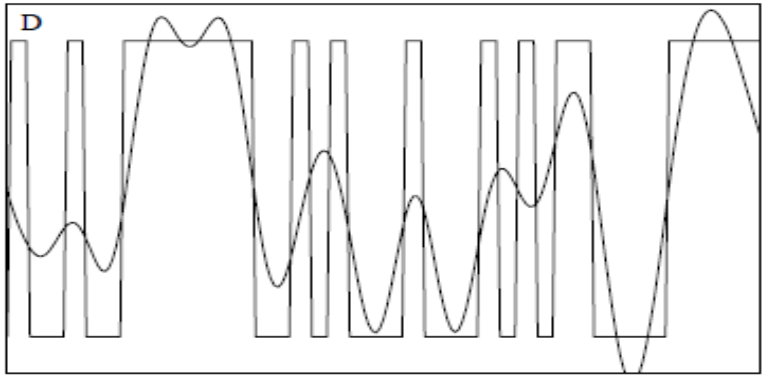
The signal exiting the channel simply can't vary fast enough resulting in ISI

sufficient BW to keep up with bit changes









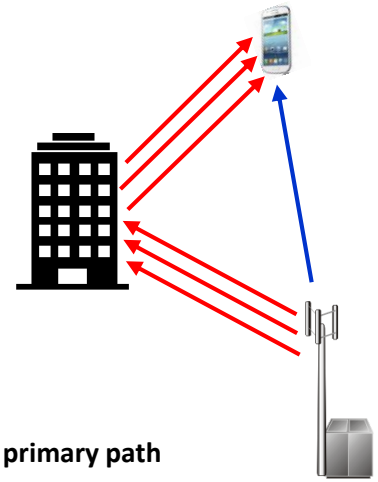
insufficient BW to keep up with bit changes

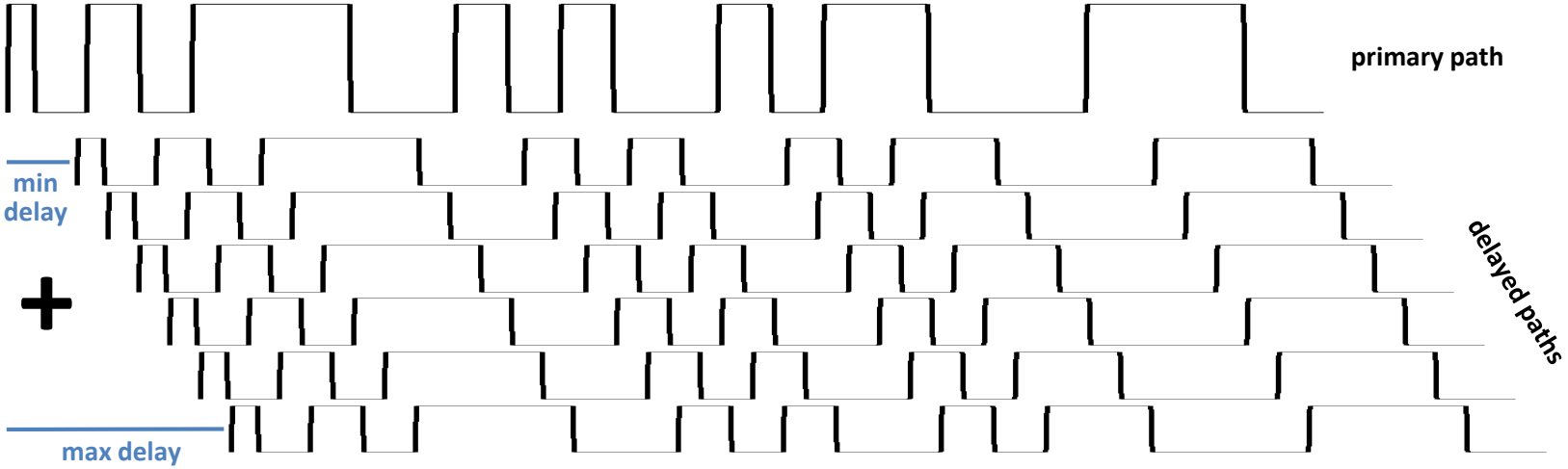
Multipath ISI

What happens when a signal is received over a primary (shortest) path and also over delayed paths (e.g. reflections off buildings) ?

The composite signal displays ISI

Note that this ISI may change rapidly for mobile users





Combatting ISI

How can we combat ISI ?

There are basically 5 strategies, each with challenges/disadvantages

1. leave large *guard times* between symbols
   - inefficient since it wastes time

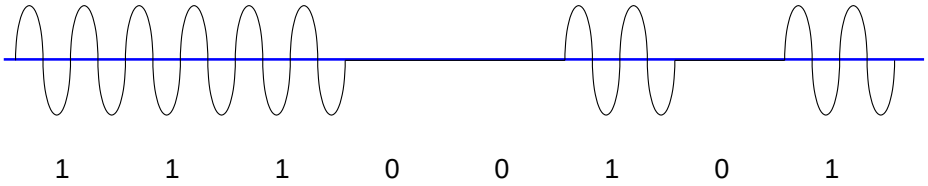
2. using an *equalizer* at the receiver (in time or frequency domain)
   - apply an inverse filter to compensate for the channel response
   - equalizers generally need training phases
   - linear equalizers suffer from noise amplification at near-zeros
   - decision feedback equalizers need to *close* a large loop
3. Tomlinson-Harashima *precoding*
   - THP applies the inverse filter before transmission
   - but requires a feedback channel to learn the channel response

4. using very low symbol rates (*OFDM* exploits this!)
   - (similar to 1.) may eliminate ISI but *seems* to strongly constrain data-rate
5. trellis encoding and using the Viterbi algorithm to decode
   - requires encoding transmitted symbols and complex decoding

OOK and ASK

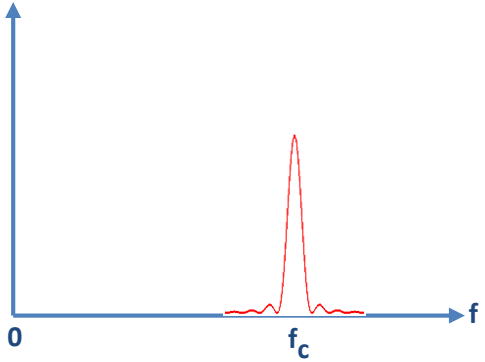
NRZ is limited to channels that pass DC, and thus not relevant to wireless

On-**O**ff **K**eying is formed by AM modulating a carrier of frequency f_c with NRZ
(alternatively you can think of this as frequency-shifting NRZ)



OOK can be extended to multi-amplitude **A**mplitude **S**hift **K**eying

The spectrum of OOK or ASK is (from frequency shifting NRZ's spectrum)



with spread between zeros of twice the symbol rate

The extra bandwidth can be utilized by more efficient modulation

Limitations on data rate

We saw that since the spectral width equals the symbol rate
(we can make do with the primary spectral lobe)
the symbol rate must not exceed the physical bandwidth

But the spectrum of PAM is independent of the number of bits per symbol
so why can't we use arbitrarily many bits per symbol ?

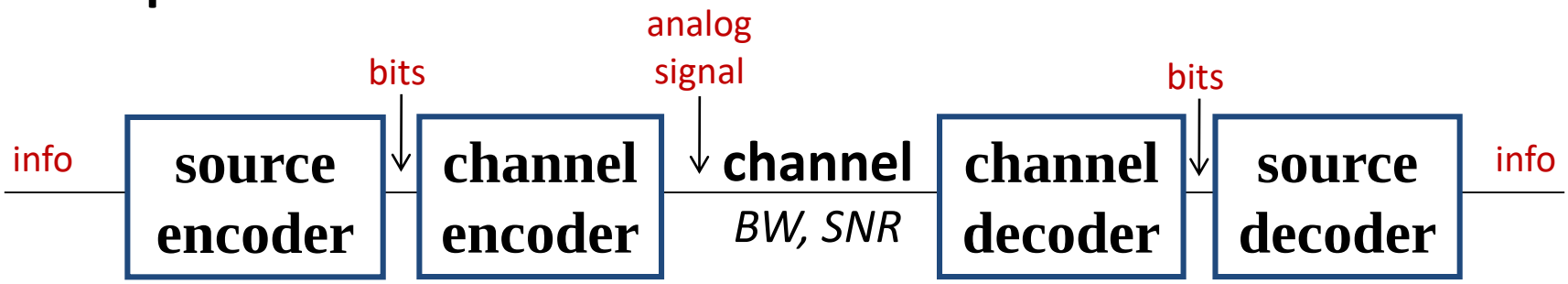
The problem here is noise –
if the symbol values are closer than the noise level
then we won't distinguish between them

To summarize:
if there is no limit on the signal's physical bandwidth or
if there is no noise added to the signal between transmitter and receiver
then there is no limitation on the number of bits/sec we can transfer

But, any physical channel with finite bandwidth and non-zero noise level
has a maximum capacity (Shannon's *channel capacity theorem*)

Shannon's Theorems

1. **Separation Theorem**



2. **Source Encoding Theorem**

Information can be quantified (in bits)

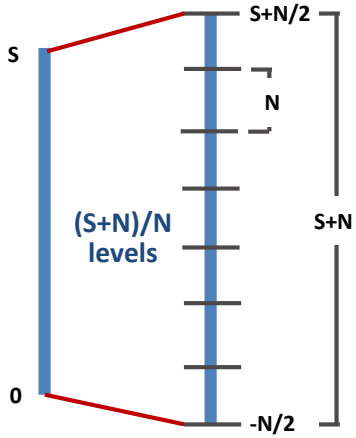
3. **Channel Capacity Theorem**

$C = BW \log_2 (SNR + 1)$

Proof of Shannon's capacity theorem

A simplistic justification of the capacity theorem is as follows:

- assume (*wolg*) that we transmit some signal with values between **0** and **S**
- assume that the channel adds DC-less noise with uniform distribution
  we'll call the peak-to-peak noise N
  so that the noise values are between –N/2 and +N/2
- the receiver always sees values between –N/2 and S+N/2
  so the receiver's dynamic range is S+N
- to maximize the information per symbol w/o overlap
  we space the signal levels by N
- so there can be (S+N)/N = S/N + 1 = SNR+1 different symbols
- hence each symbol contains $\log_2(\text{SNR} + 1)$ bits
- but there can be BW symbols per second



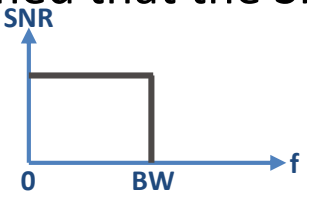
Hence, the maximum information rate $C = \text{BW} \log_2(\text{SNR} + 1)$ bit/s

The maximum spectral efficiency is $C/\text{BW} = \log_2(\text{SNR} + 1)$ bit/s/Hz

Capacity for frequency-dependent SNR

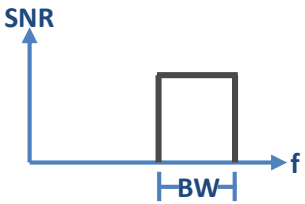
The capacity theorem assumed that the SNR was

- constant from DC to BW
- zero over BW



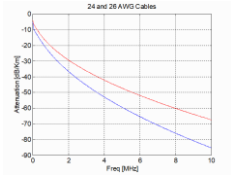
or more generally

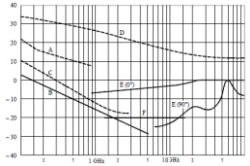
- constant in some passband
- zero outside the passband



Which will not be the case if either

- the signal attenuation (including multipath cancellation)
- the noise (including interference)





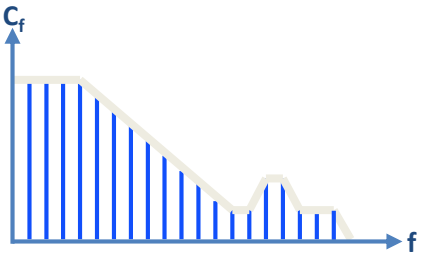
or both, vary with frequency

The extension of the theorem is simple

- divide the passband into channels of bandwidth Δf centered at f
- the capacity theorem states that for the channel $C_f \approx \log_2(SNR(f) + 1)\,\Delta f$
- the composite capacity is $\sum_f \log_2(SNR(f) + 1)\,\Delta f$

The theorem becomes exact when we send $\Delta f \rightarrow 0$

and then $C = \int \log_2(SNR(f) + 1)\,df$

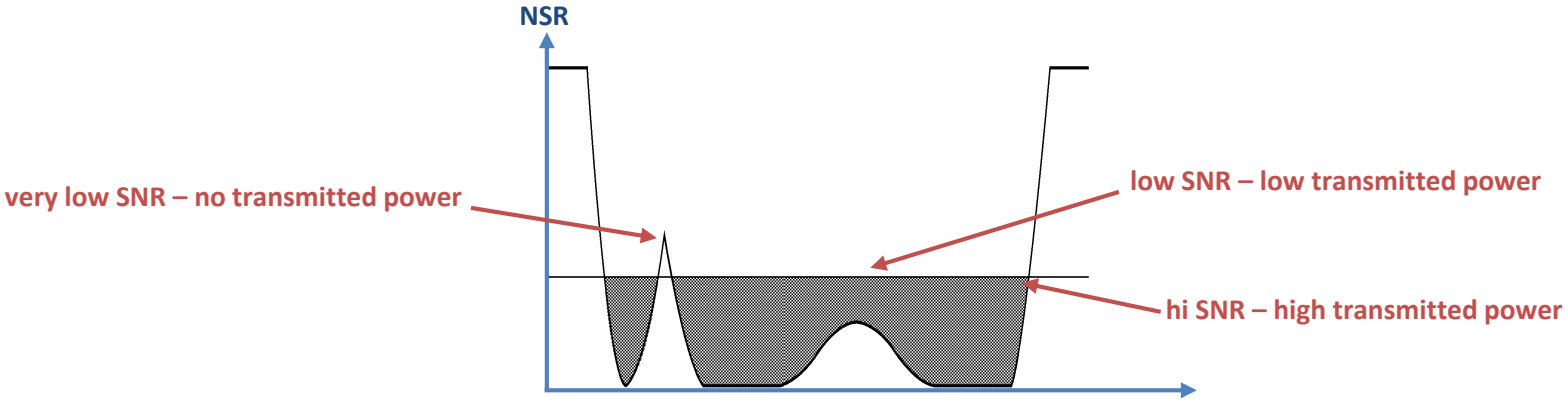


Water pouring theorem

The *converse* to Shannon's channel capacity theorem
is Gallager's water pouring theorem

The spectrum of the optimum (capacity-achieving) modulated signal
is given by the *water pouring* algorithm

- fill a pitcher with a volume of water
  representing the total power to be transmitted
- pour the water over the reciprocal SNR(f) graph (Noise to Signal Ratio)



This power spectrum can be achieved by

- spectral shaping
- explicit power loading

To FEC or not to FEC

Shannon's separation theorem implies
that it is suboptimal to use *data-layer* **E**rror **C**orrection **C**ode schemes

This is because an ECC always adds extra non-information bits
reducing the information bit rate
signal layer ECC methods (e.g., TCM) don't contradict the separation theorem

Yet, all mobile systems are full of error detection/correction codes:

- CRC (Cyclic Redundancy Code) at MAC layer before passed to physical layer
- repetition codes for very small data blocks
- TBCC (Tail-Biting Convolutional Codes) or polar codes for control channels
- turbo codes or LDPC (Low Density Parity Code) for data channels

This is because in contrast with the assumptions of the capacity theorem
the SNR is often not stationary due to *noise events* and *fading*
resulting in errors in the recovered information

ECC techniques

There are basically three techniques to combat errors:

- ARQ – **A**utomatic **R**epeat re**q**uest
  - use an error *detection* algorithm (e.g., parity check, CRC) to detect errors
  - whenever error detected request retransmission of information
- FEC – **F**orward **E**rror **C**orrection
  - use an error *correction* algorithm (e.g., repetition, LDPC, Turbo codes) to correct errors in reception without retransmission

Note that it's easier to *detect* errors than to *correct* them

For example: RS(k+2t+1,k) can detect 2t byte errors but only correct t of them

- HARQ – Hybrid ARQ (in 4/5G we use **S**top **A**nd **W**ait HARQ)
  - use FEC to correct as errors if you can
  - use ARQ when detect errors that could not be corrected
  - retransmission may provide *additional* FEC to enable correction

HARQ variations

HARQ has many variations !

chase combining (Type I HARQ) adds both FEC and CRC to every message
adding significant overhead even if channel quality is good

Type II HARQ sends message + error detecting parity bits (low overhead)
and sends FEC only if requested

soft combining - receiver doesn't discard messages with errors
but rather combines all information received

incremental redundancy - retransmissions are not identical
but rather add new FEC information

Stop **A**nd **W**ait - sender waits for ACK before sending next message
thus increasing latency (alternative is to buffer messages like in TCP)

Multiple SAW processes – use multiple stop and wait processes (8 in 4G)
when 1 SAW process is waiting for its ACK others can send messages
(number of SAW processes should be matched to RTT)

FEC terminology

A (n,k) *block* FEC encoder inputs k bits (or more generally symbols)
and outputs n>k bits (or symbols)

It is called *systematic* if the message contains the original k message bits
and r = (n-k) redundancy bits

The code's *rate* is k/n, i.e., k useful bits for every n bits sent/received

A code's **H**amming **D**istance is the distance between two blocks
the idea is to obtain the highest HD possible

Examples

- *Hamming code* of block length $n=2^r$ adds r+1 bits to the message
  it is thus a $(2^r-1, 2^r-r-1)$ code with Hamming Distance =3
  and can detect any 2 bit errors and correct any 1 error
- *Reed Solomon* codes work on bytes (not bits)
  RS(n,k) has HD=n-k=2t+1, detects all 2t byte errors, corrects any t bytes
  rate is k/n, e.g., RS(204,188) has rate 188/204 = 92%
- *convolutional* (not block) codes that insert a redundant bit every k bits
  have rate 1/2, 2/3, 3/4, 4/5, etc.

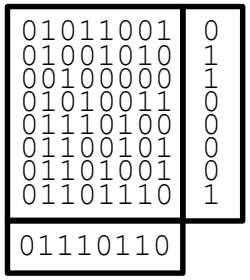
How does ECC work?

The simplest FEC is a repetition code – simply send the message N times
such coding has very low rate
and is reserved for the most critical information

The simplest error detection scheme is block parity
which can detect any single bit error

Stronger error detection often uses **C**yclic **R**edundancy **C**hecks
Ethernet's 32-bit CRC can detect all 3-bit errors for frames up to 11455 bytes

Parity bits can be also be used for error correction



Low **D**ensity **P**arity **C**heck codes (Gallager 1960)
use sparsely connected parity checks

LDPC, *Turbo* codes, and *polar* codes
can all achieve optimal performance, with different trade-offs

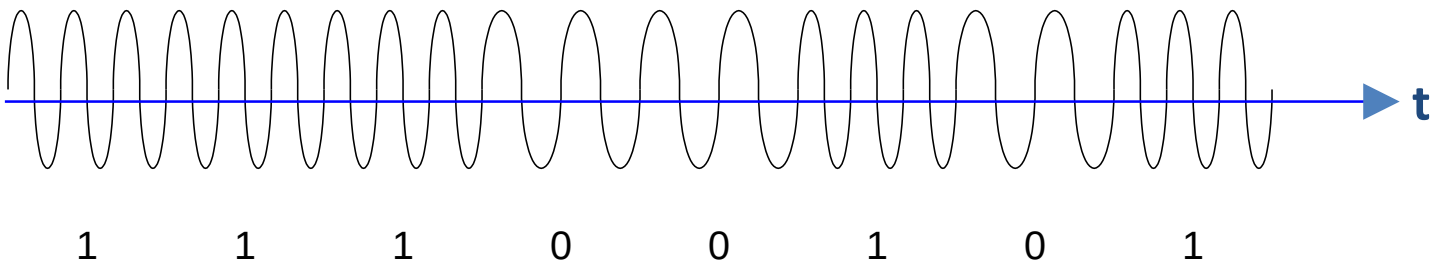
- LDPC (used in 5G data channels) can achieve high throughput at high rates
- polar codes (used in 5G control channels) are best for small blocks

FSK

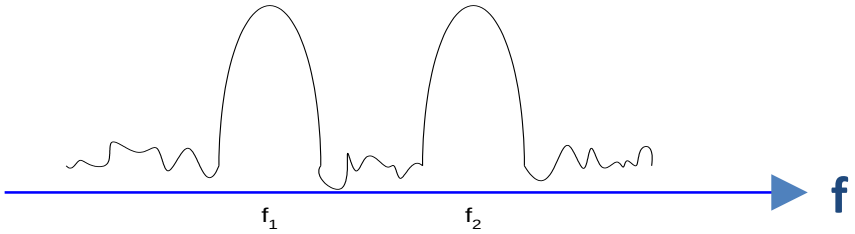
What can we do about noise?

We can gain resilience by using *frequency diversity*

i.e., using two independent OOKs with basically the same information



This is called **F**requency **S**hift **K**eying

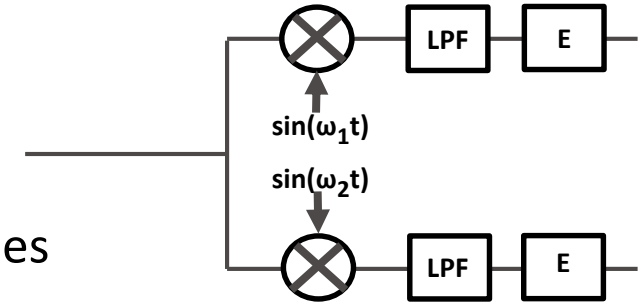


We can extend this to multiple tones (mFSK) with N bits/tone (symbol)

The problem with FSK

Note that we are exploiting the fact
that sinusoids of different frequencies are *orthogonal*

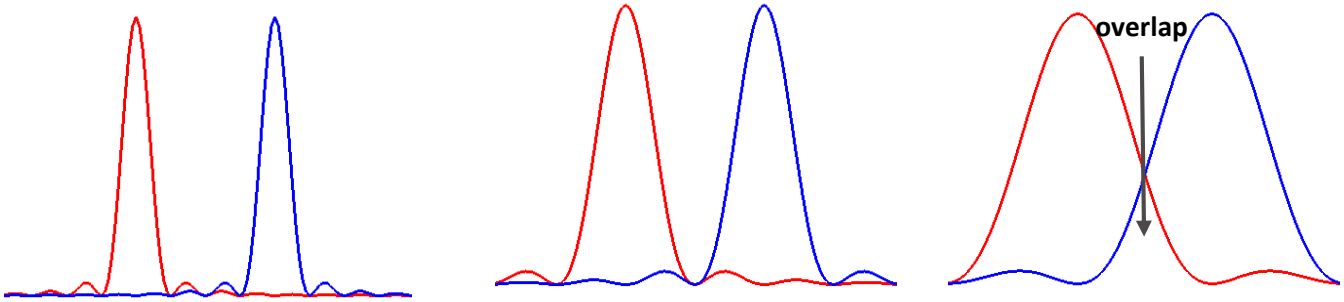
The optional receiver
mixes down to zero
filters
and compares energies



So, why can't we use many frequencies to bypass Shannon?

Because sinusoids of different frequencies are only orthogonal
when we integrate over all time (from -∞ to + ∞)

If each bit only lasts a short time
the uncertainty theorem says that its frequency is uncertain
so we can't be sure which tone was transmitted!



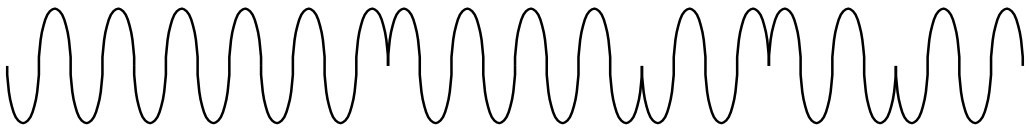
PSK

But we can differentiate between sinusoids of the same frequency - but different phases after a single cycle !

$$\int \sin(\omega t)\cos(\omega t)\,dt = 0$$

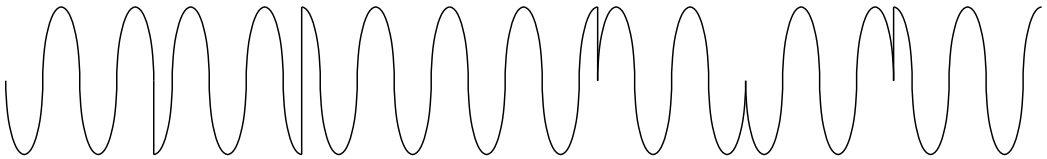
This is the basis of **P**hase **S**hift **K**eying

BPSK has 1 bit / symbol



1 1 1 0 0 1 0 1

QPSK has 2 bits / symbol

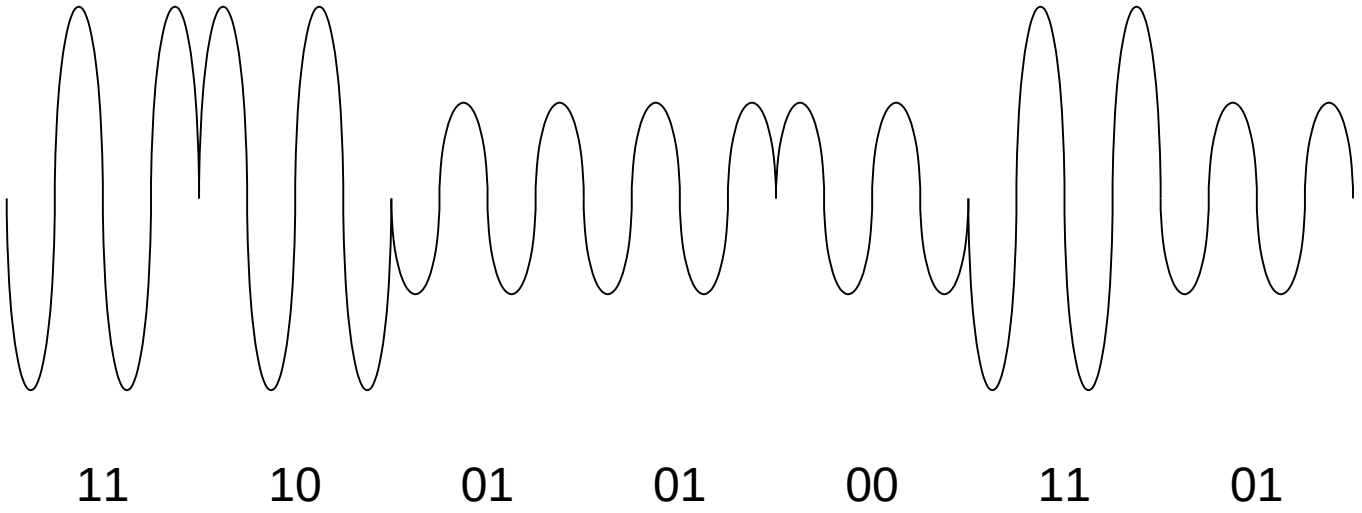


11 10 01 01 00 11 01

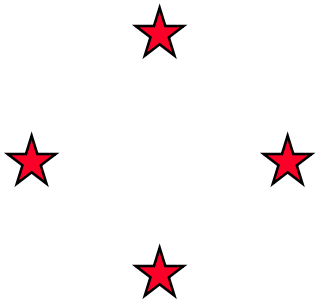
QAM

Finally, we can combine PSK and ASK (but not FSK)
to get **Q**uadrature **A**mplitude **M**odulation

2-QAM has 4 different symbols (+sin, -sin, +3sin, -3sin)
and thus 2 bits/symbol

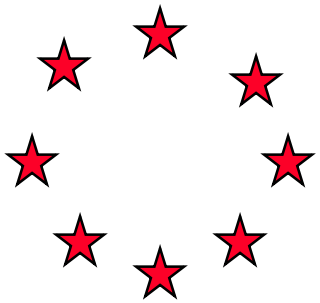


Some *constellations*



QPSK

2 bits/symbol



8PSK

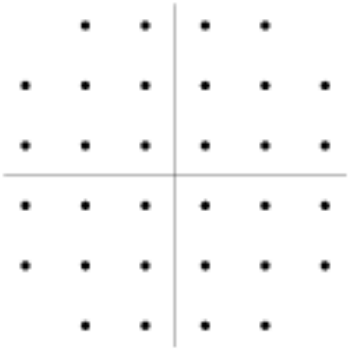
3 bits/symbol

16QAM

4 bits/symbol

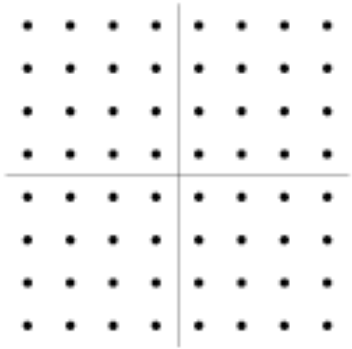
V.29

4 bits/symbol



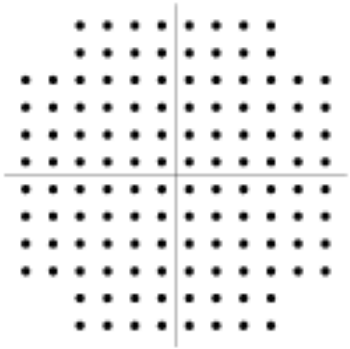
V.32 9600 bps

5 bits/symbol



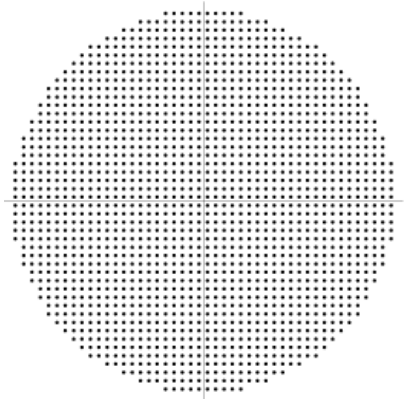
V.32 12000 bps

6 bits/symbol



V.32 14400 bps

7 bits/symbol



V.34 33600 bps

Spectral efficiency of simple modulations

We defined spectral efficiency as the bit/s per Hz

We can now compare the modulation techniques we have seen so far

modulation	bit/symbol	BW/symbol rate	spectral efficiency
NRZ	1	1	1
BPSK	1	2	0.5
QPSK	2	2	1
8PSK	3	2	1.5
16QAM	4	2	2
64QAM	6	2	3
256QAM	8	2	4

Note that we are using *raw* (DSB) efficiencies

we will see that the values for PSK and QAM can be improved by a factor of 2 !

FDM

QAM (including PSK as a special case) is a very efficient modulation
    approaching the Shannon capacity for simple bandpass channels

But the spectrum of a QAM signal is inflexible
    like all discretely keyed modulations of a single carrier
        it is a sinc centered at f_c with first zeros at $f_c \pm$ symbol-rate

The spectrum can be shaped using a Tomlinson precoder
    but this requires a feedback channel and precomputing the filter

So, QAM does not lend itself to water-pouring

The trick is to use Frequency Domain Multiplexing

We saw FDM as a technique to mux different information sources

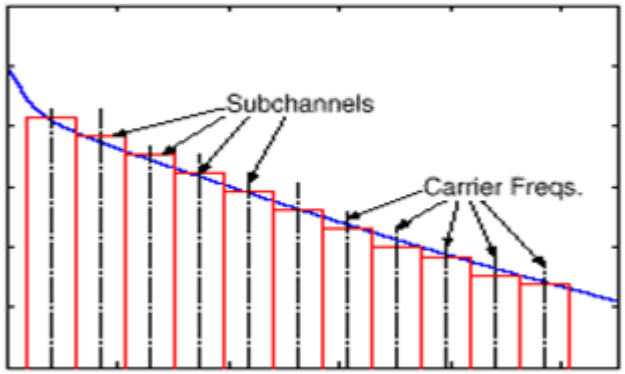
Here we divide a single information stream into blocks of bits
    and mux them together using distinct carrier frequencies (sub-carriers)

Each sub-carrier signal can
- have its own power level
- use its own modulation technique (PSK, QPSK, 16QAM, 64QAM, ...)

thus directly implementing water pouring

FDM combats ISI but creates ICI

FDM uses many subchannels, each with low bandwidth but low data rate



Since the data rate is low, there is essentially no ISI

And since each subchannel is localized in frequency
we can perform equalization in the frequency domain (FEQ)
i.e., simply multiply each frequency and shift its phase

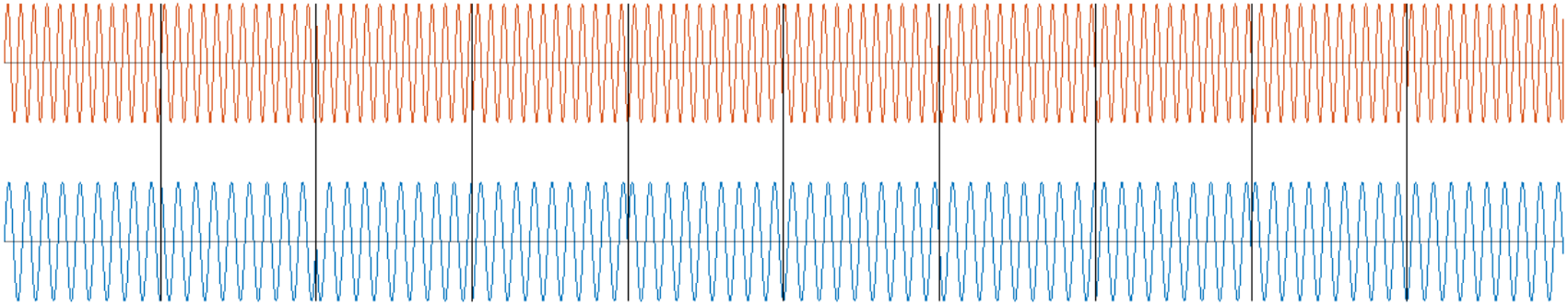
But, we need to space the sub-carriers far enough apart
to avoid **I**nter**C**hannel **I**nterference

This squanders bandwidth, distancing us from attaining the Shannon capacity
(similar to what we saw with FSK)

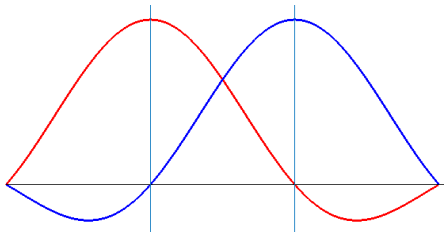
OFDM

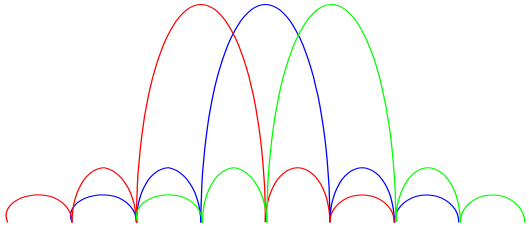
The solution is called Orthogonal FDM (OFDM)

- all sub-channels use the same symbol rate (even if different modulations)
- sub-carriers are spaced at precisely the symbol rate
- the sub-carriers are the precisely orthogonal
  and hence do not interfere with each other



- ICI is eliminated with no guard frequencies needed
- simple implementation based on the FFT



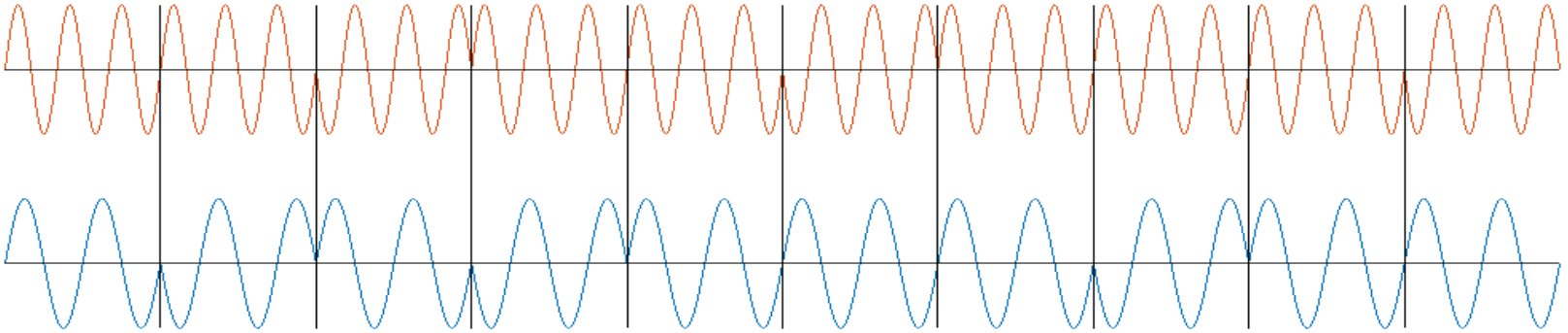


Why are the channels orthogonal?

Let's look at the baseband signal

where all sub-carriers are multiples of the symbol rate

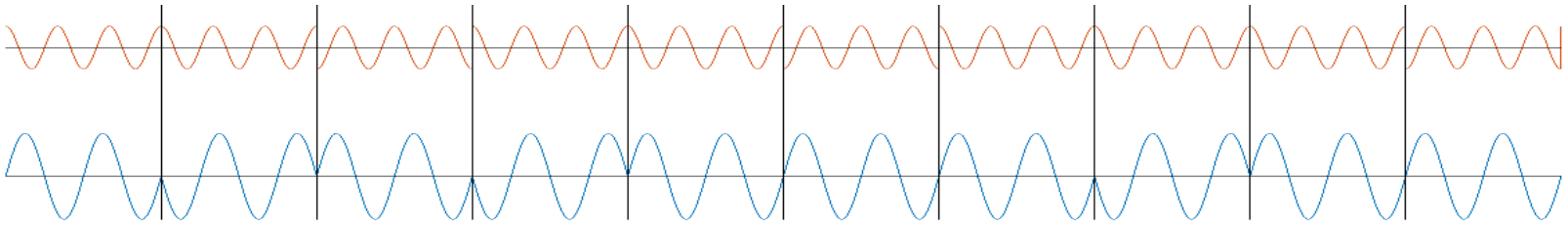
so that there are an integral number of sinusoid cycles in a symbol



Any two such signals are precisely orthogonal

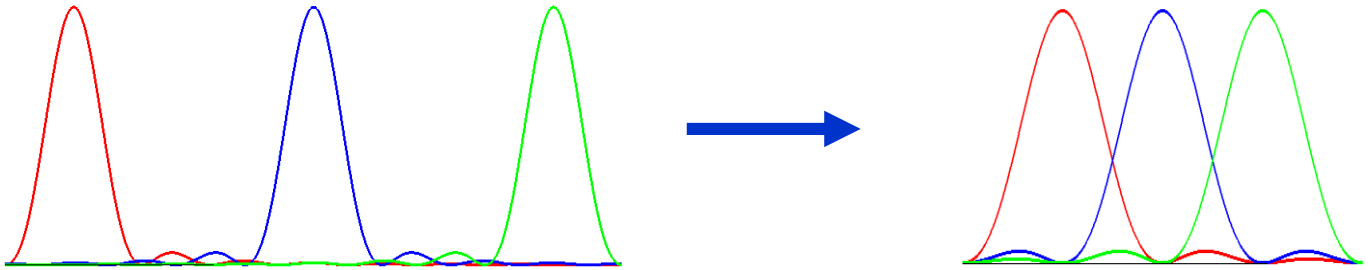
$$\int_0^{\frac{2\pi}{\omega}N} e^{in_1\omega t}\, e^{-in_2\omega t}\, dt = 0$$ (the only requirement is for a whole number of cycles of sin **Δω** t !)

and the same is true if we arbitrarily phase shift or amplify the signals



Spectral efficiency

OFDM provides the minimum possible sub-carrier spacing
and hence eliminates the need for *guard bands*

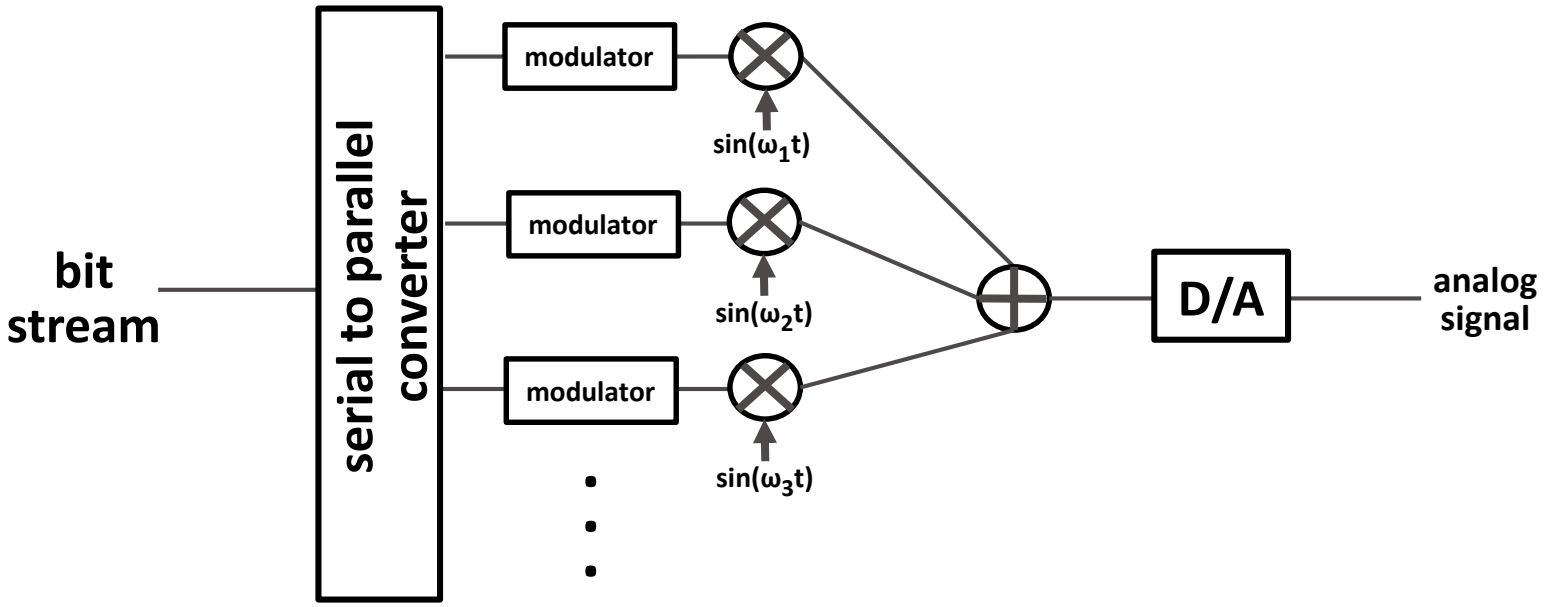


Now each sub-channel effectively occupies only symbol-rate of bandwidth
So the spectral efficiencies improve by a factor of 2 !

modulation	bit/symbol	BW/symbol rate	spectral efficiency
BPSK	1	1	1
QPSK	2	1	2
8PSK	3	1	3
16QAM	4	1	4
64QAM	6	1	6
256QAM	8	1	8

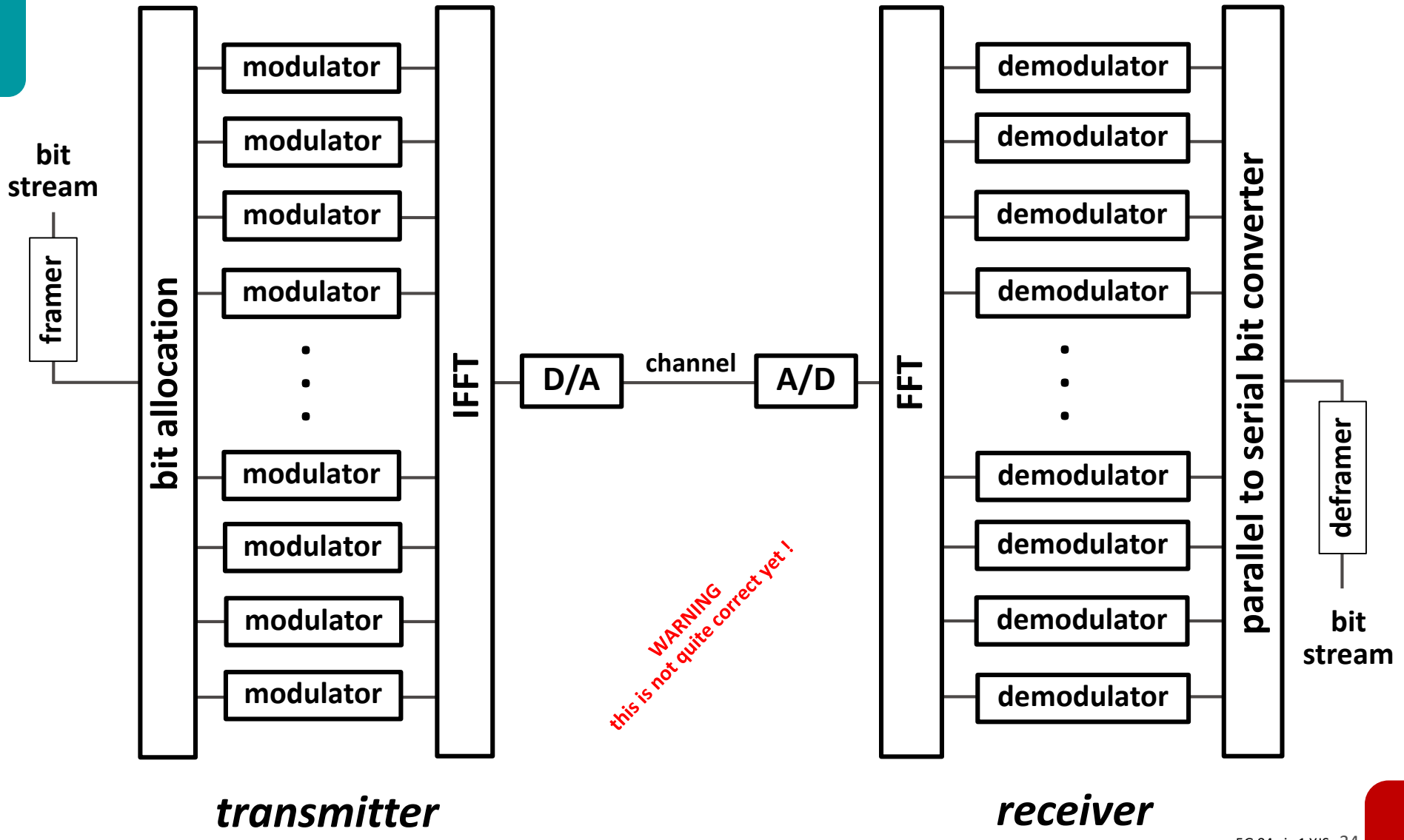
FFT

In order to transmit each sub-channel
we should modulate a baseband signal
up-mix each to the desired sub-carrier frequency (multiply by $e^{i\omega_c t}$)
and add the sub-channels together



The up-mixing can be performed in parallel for all sub-carriers
by the inverse Fourier transform (Zimmerman and Kirsch 1967)
and the FFT does it quickly (Weinstein and Ebert 1969)

OFDM modem paradigm



Cyclic prefix

What we previously derived *almost* works

For **analog digital** signal processing

linear cyclic convolution in the time domain **y = h * x**

is equivalent to multiplication in the frequency domain

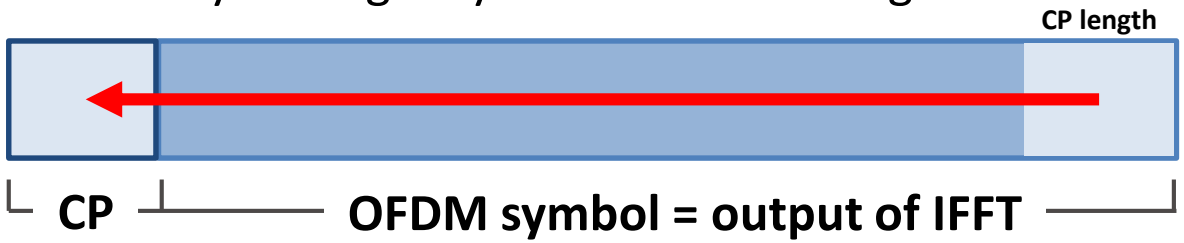
$$Y(\omega) = H(\omega)\, X(\omega)$$

$$Y_k = H_k\, X_k$$

So, the linear convolution in the analog channel

has to be converted into cyclic convolution for the digital channel

This is done by adding a **C**yclic **P**refix to the signal



which basically acts as a *guard interval* to eliminate ISI

The CO duration tends to be from 1/32 up to 1/4 of the symbol duration

and has to be long enough to include the maximum ISI spread

For example, if ISI is due to multipath

then the CP must be long enough to incorporate the longest delayed path

CP at work

OFDM splits the signal in the time domain into *frames*

In order for the signal processing of each frame to be independent
the CP ensures that delay spread is contained within the received frame

As a simple example, assume a frame of 8 samples
a 2-sample multipath $y_n = x_n + h_1 x_{n-1} + h_2 x_{n-2}$ and a CP of 2 samples

The OFDM frame before CP insertion is $x_0\, x_1\, x_2\, x_3\, x_4\, x_5\, x_6\, x_7$

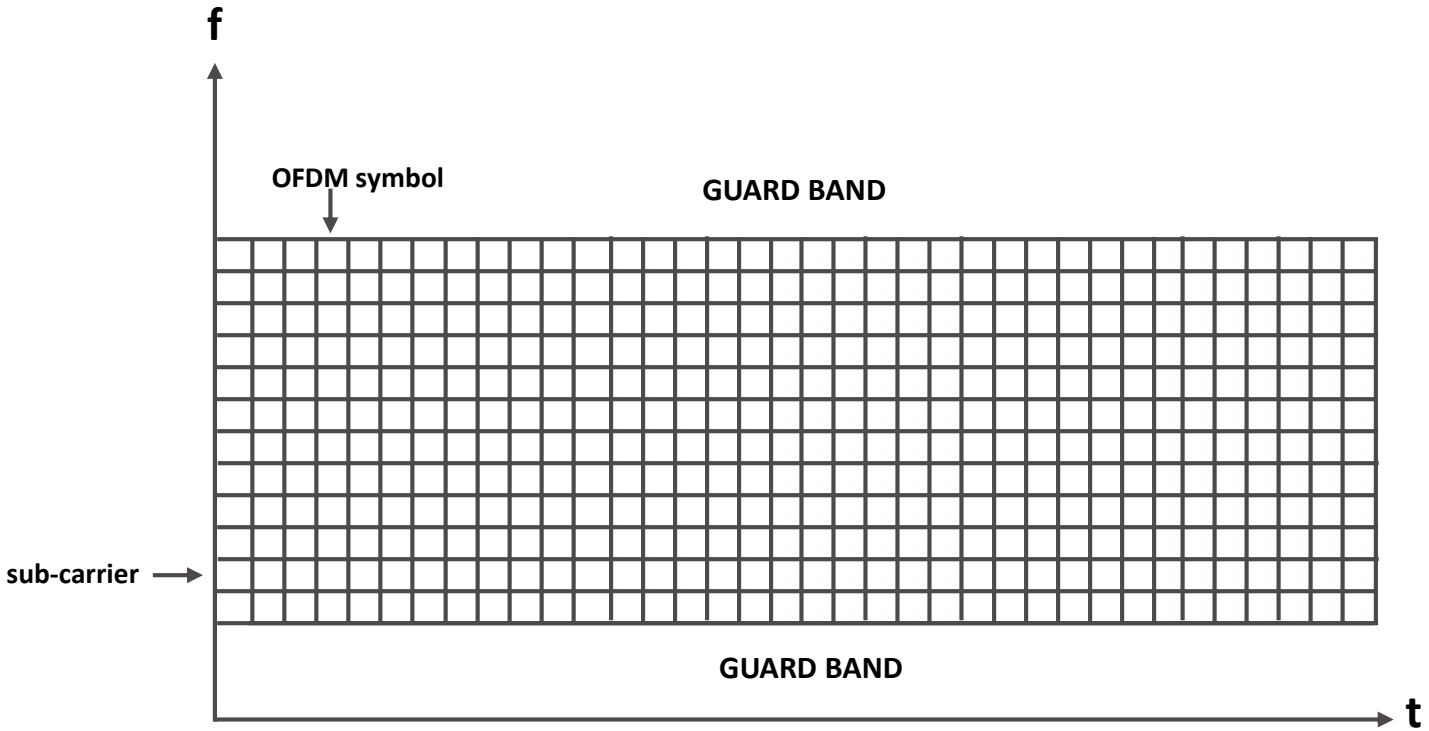
After CP insertion the transmitted OFDM frame is $x_6\, x_7\, x_0\, x_1\, x_2\, x_3\, x_4\, x_5\, x_6\, x_7$

The received OFDM signal is $y_6\, y_7\, y_0\, y_1\, y_2\, y_3\, y_4\, y_5\, y_6\, y_7$
where y_6 and y_7 contain ISI from the previous frame and are *discarded*
and

$$\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & h_2 & h_1 \\ h_1 & 1 & 0 & 0 & 0 & 0 & 0 & h_2 \\ h_2 & h_1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & h_2 & h_1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & h_2 & h_1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & h_2 & h_1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & h_2 & h_1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & h_2 & h_1 & 1 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{pmatrix}$$

All the information is present for equalization to recover the original $x_0 \cdots x_7$
and the matrix is *circulant* – which is solved in the frequency domain!

OFDM signal structure



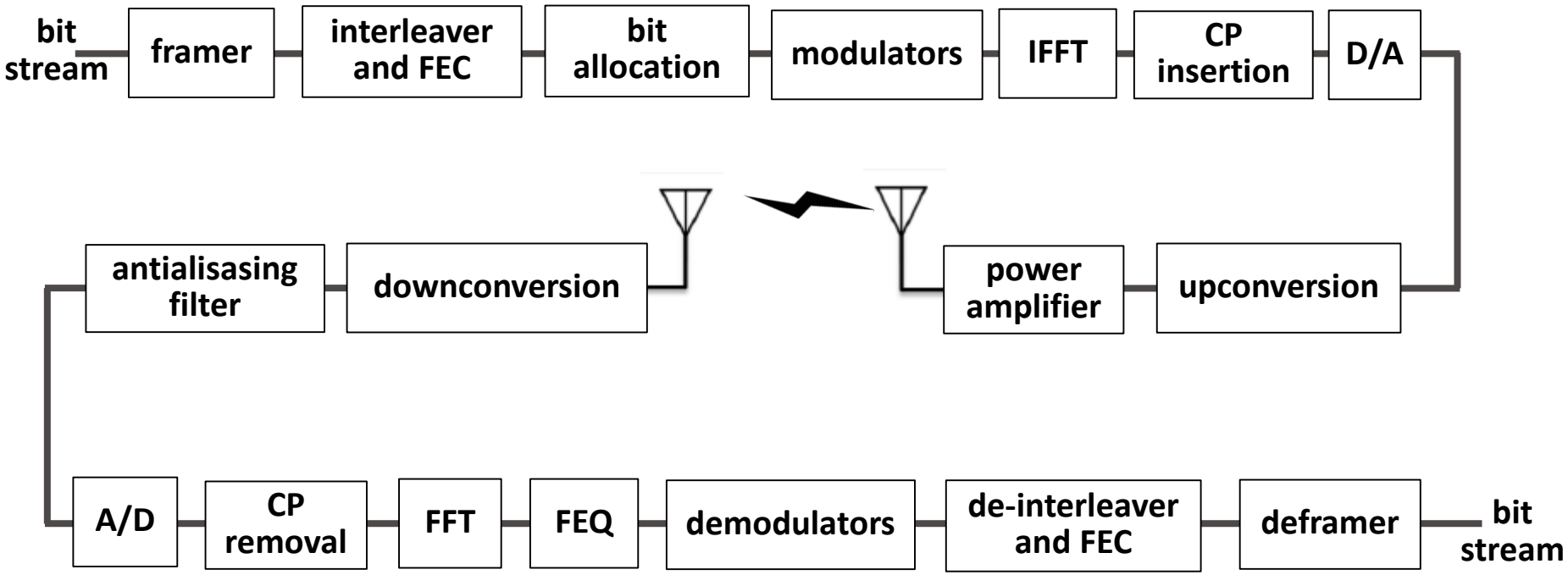
For LTE:

- **channel bandwidth ..., 5, 10, 15, or 20 MHz**
- **guard band overhead is 10%**
- **sub-carrier spacing = 15 kHz**
- **OFDM symbol duration = 1/15KHz = 66.67 µsec**
  - **short CP = 4.7 µsec so total duration = 71.367 µsec**
    - **1 *slot* = 7 symbols ≈ ½ msec***
  - **long CP = 16.7 µsec so total duration = 83.367 µsec**
    - **1 *slot* = 6 symbols ≈ ½ msec***

BW (MHz)	usable BW (MHz)	subchannels	FFT
5	4.5	300	512
10	9	600	1024
15	13.5	900	1536
20	18	1200	2048

* CP durations are *adjusted* so that the slot is precisely ½ msec

OFDM modem



Warning: this is *still* somewhat over-simplified!

PAR

The **P**eak to **A**verage **R**atio (AKA crest factor) of a signal
is the ratio between its maximum peak-to-peak to its root mean square

For example, for a single the p2p is 2 while the rms is √2, so PAR= √2

When adding N independent sine values (as is done in OFDM)
the PAR is multiplied by N (the Gaussian tail)

This creates a major problem for OFDM transmitter's power amplifier

Power amplifiers are only linear in a limited operating range
if the input signal value is too large, the output saturates
leading to distortion that breaks orthogonality between sub-carriers

This is especially problematic for the cellular uplink
where the UE amplifier can not be sophisticated/expensive

Many PAR reduction techniques have been developed
but it may be better to avoid using multi-carrier methods in the UL

SC-FDM

To drastically reduce the PAR we can use **S**ingle **C**arrier FDM
sometimes called linearly precoded OFDMA (LP-OFDMA)

Don't be confused, SC-FDM is a close relative to OFDM

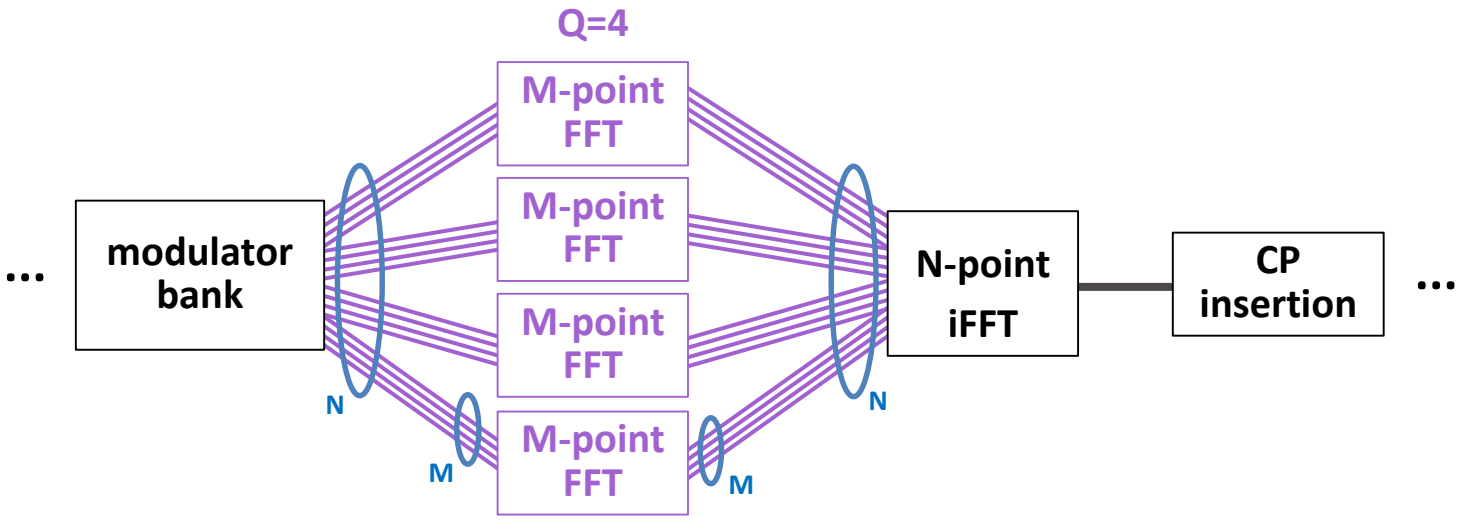
To transmit SC-FDM instead of OFDM
instead of inputting N QAM symbols into the iFFT
we group them into blocks of M (M = N/Q)
and perform a M-point FFT on each of these *before* the N-point iFFT

This results in Q orthogonal sub-carriers
if Q=1 (M=N) FFT and iFFT cancel out, resulting in a single carrier
if Q>1 the PAR is Q times that of a single carrier signal

So, why if we want a single carrier – why use OFDM at all?

- we can still perform the equalization in the frequency domain
- we can still flexibly allocate channels

SC-FDM transmitter detail



SC-FDM only