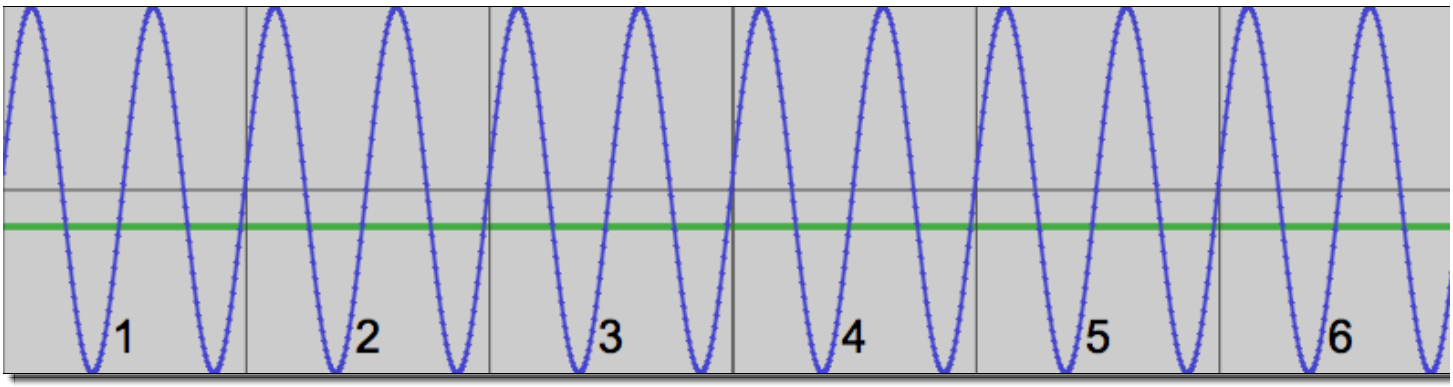
Glissando Modulation

Brian McMillin

Frequency Modulation

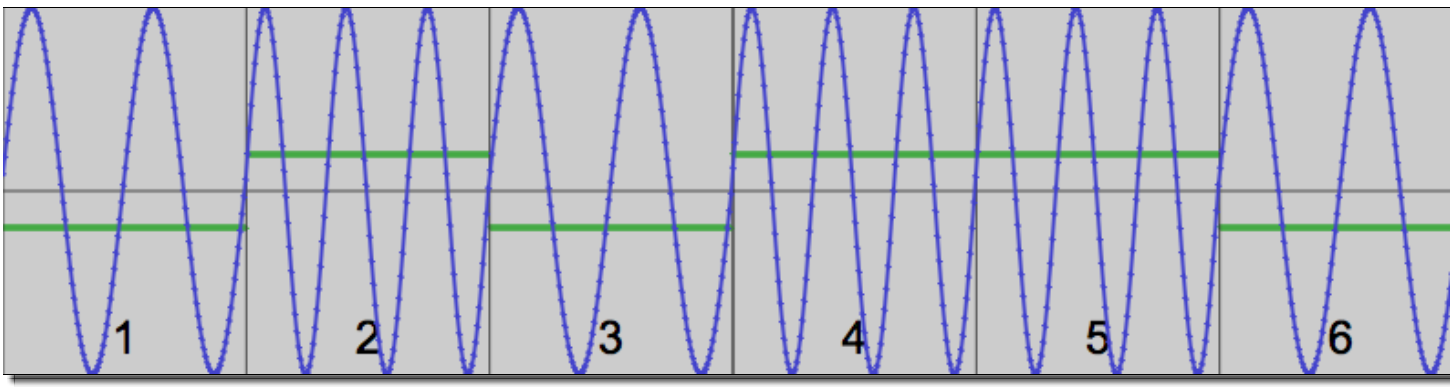
Consider the following graph depicting an unmodulated carrier. The carrier is the blue sine wave and the modulation is the green trace.

For this discussion the symbol rate is constant and we show 6 symbols across the graph.



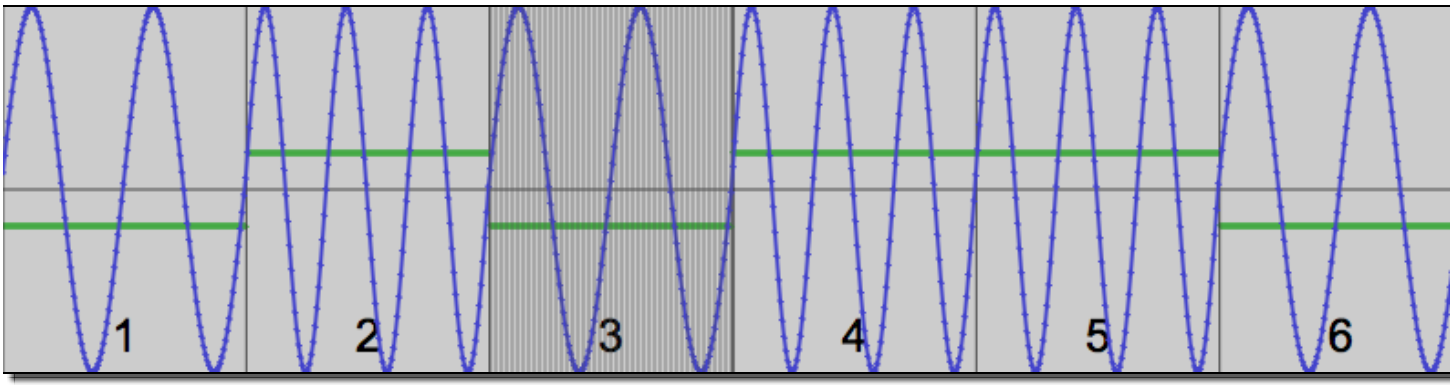
Now consider the frequency modulated carrier given a data stream of 010110.

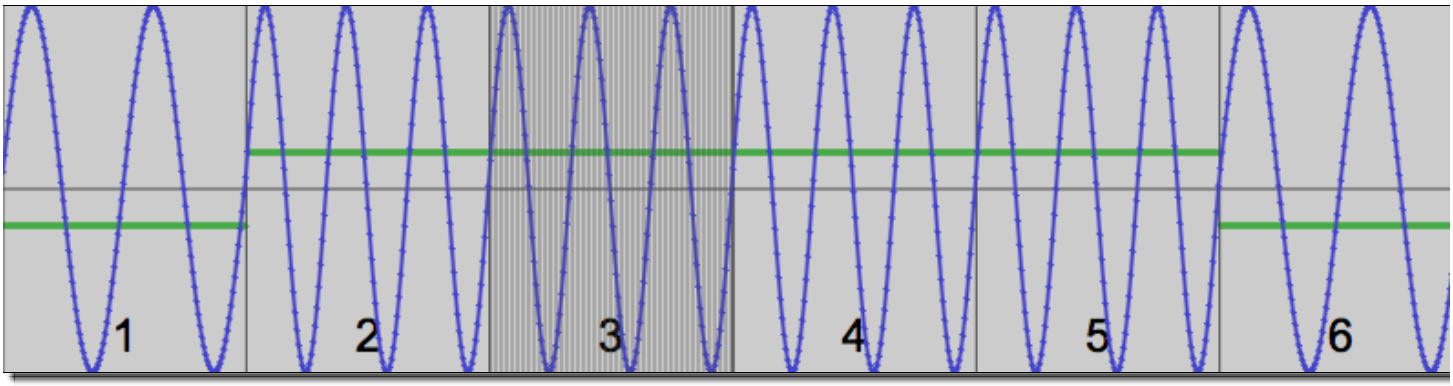
For clarity a '0' symbol is two cycles of the carrier and a '1' symbol is three cycles in the symbol time.



This can be considered a rudimentary implementation because all of the information is encoded in the frequency component. The phase of the carrier is completely ignored.

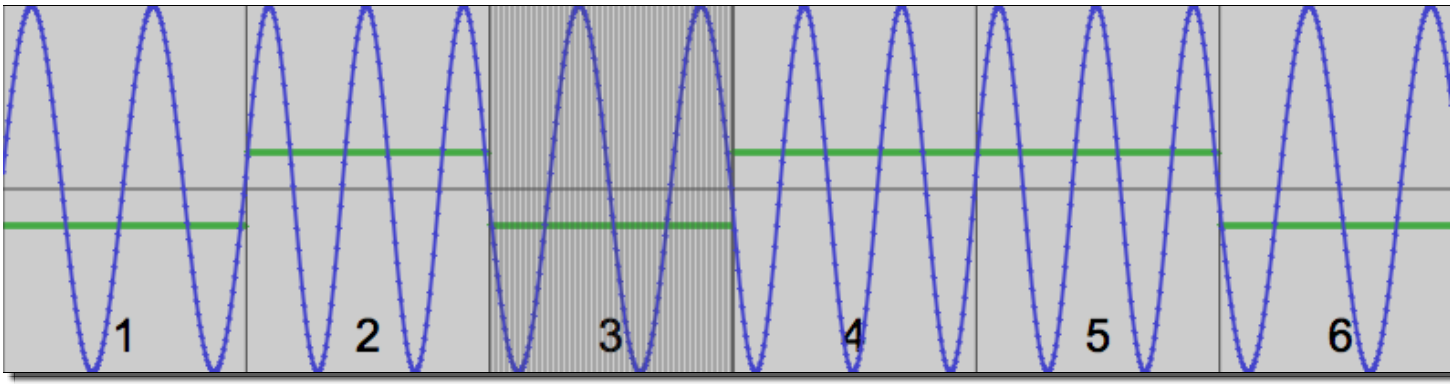
Now let us consider an error situation in which a single bit is obscured, perhaps by noise.

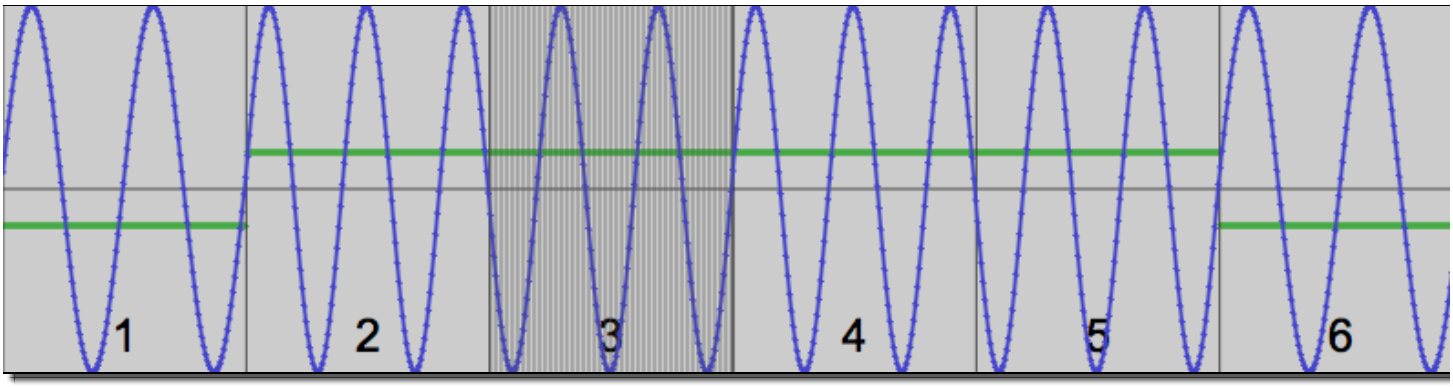




Our naive choice of a 2:3 ratio means that changes in a single bit do not affect the frequency or phase of any other parts of the signal. Recovering the lost bit relies on some form of Error Correcting Code.

Now consider an improved choice of frequency shift. In this case, we will use 2:2.5



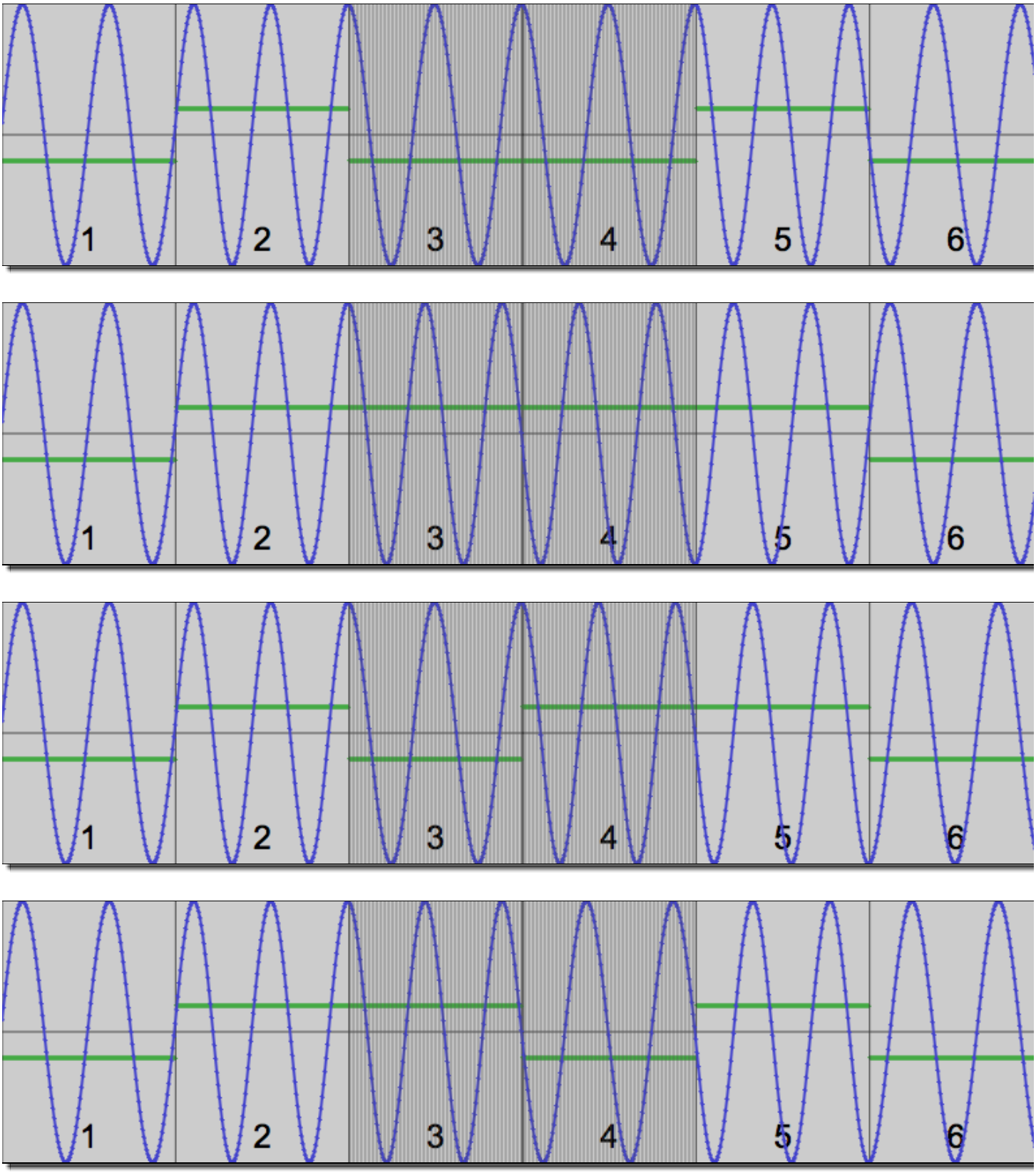


Now our single-bit drop-out can be recovered by noting the obvious 180° phase shift in the next symbol.

Eliminating the abmiguity in a corrupted signal by using redundant encoding in both frequency and phase reduces the load on the Error Correction mechanism and improves overall noise immunity.

Obviously, an actual implementation would use smaller frequency shifts and would be designed to assist with more than single-bit dropouts.

To aid visualization along these lines, consider the two-symbol dropout depicted below using 2:2.25 shift.



Note that the four possible symbol combinations in the obscured area result in three unique phase angles at the beginning of the next symbol. We cannot

distinguish 0-1 from 1-0, but we can clearly identify 1-1 and 0-0.

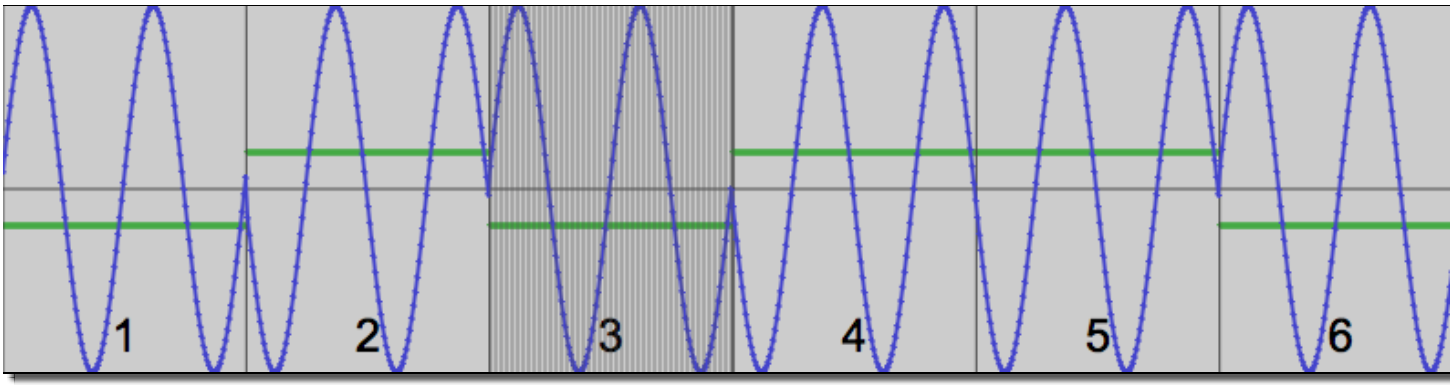
The required frequency and phase information are naturally obtained from the Fourier Transform of the received signal.

Phase-Shift Keying

Phase-Shift keying is another commonly used modulation scheme. The information symbols are encoded in the phase of the carrier, which is a constant frequency. It will be noted that the same problem discussed above is present in standard PSK implementations.

There is no redundancy in the encoding, and the frequency component of the FFT is not used.

The following graph shows a representation of Binary Phase Shift Keying in which the carrier phase shifts by 180°.



Self-Clocking and Run-Length-Limited Coding

All of the modulation techniques discussed above have assumed that the symbol boundaries (time on the horizontal axes) were well-known. In the real world this will never be the case.

A number of methods of establishing symbol synchronization have been used.

Manchester encoding simply uses two bit-times per symbol with, for example,

0-1 meaning '1' and 1-0 meaning '0'. This ensures a well-defined state change for every symbol at the expense of bandwidth.

Serial ASCII is encoded with a fixed '0' start bit and fixed '1' stop bit for every 8 data bits. This ensures a worst-case of both a 0 and 1 bit out of every ten bits on the transmission medium.

In all cases, the bit transitions are used to establish the boundaries of each symbol.

The present discussion has centered on binary systems - i.e., systems in which a symbol can take on only two states. Advanced modulation schemes usually allow a single symbol to fall into more states, usually a power of two, to represent more than one bit.

Allowing more states provides the opportunity to ensure a self-clocking mechanism with a much lower overhead (bandwidth penalty) than can be achieved with these binary symbols.

The proposed modulation technique, to be discussed shortly, will encode N bits per symbol using 2^N+1 unique states. This implies that the number of unique states will be odd, and that the clocking overhead drops radically as the number of bits-per-symbol increases.

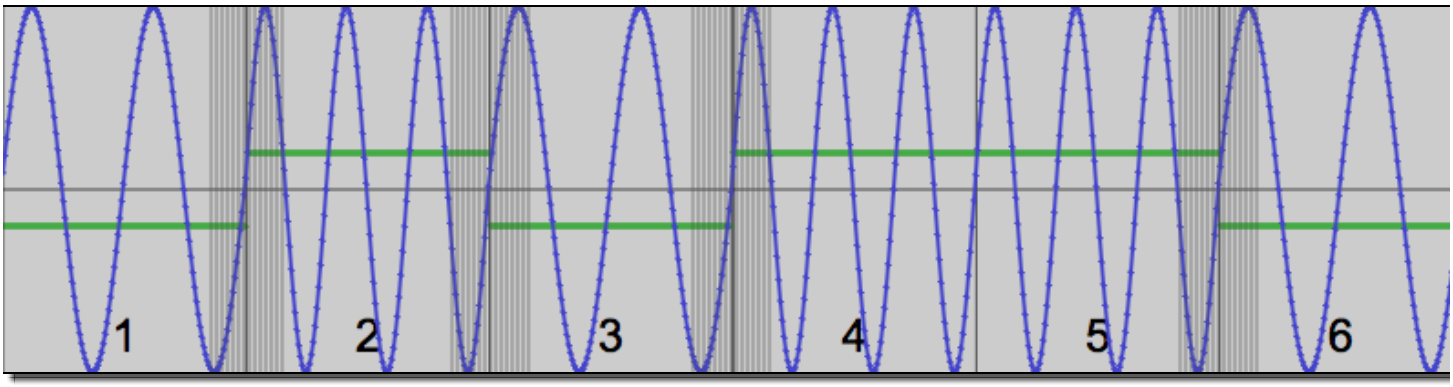
Simply stated, the sender replaces the second of two consecutive identical symbols with the 'extra' symbol. Thus, any message is guaranteed to never contain consecutive identical symbols. This ensures unambiguous boundaries for every symbol and allows the symbol clock to be continuously adjusted as needed. A tight lock on the symbol rate is especially important in the presence of inexpensive hardware, noise and Doppler shifts.

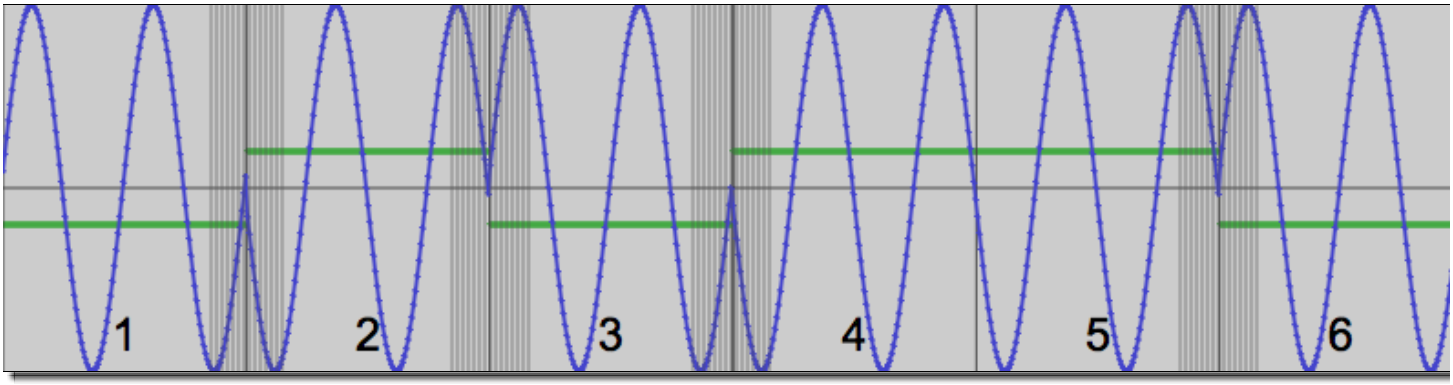
Obscured Transitions

The diagrams so far have presumed instantaneous transions from one state to the next at each symbol boundary. Real-world signals are never so clean, and a

large part of the bandwidth in a typical communication channel is wasted because of the need for the medium to stabilise in each new configuration.

The regions of ambiguity for Frequency Shift Keying and Phase Shift Keying are shown below.

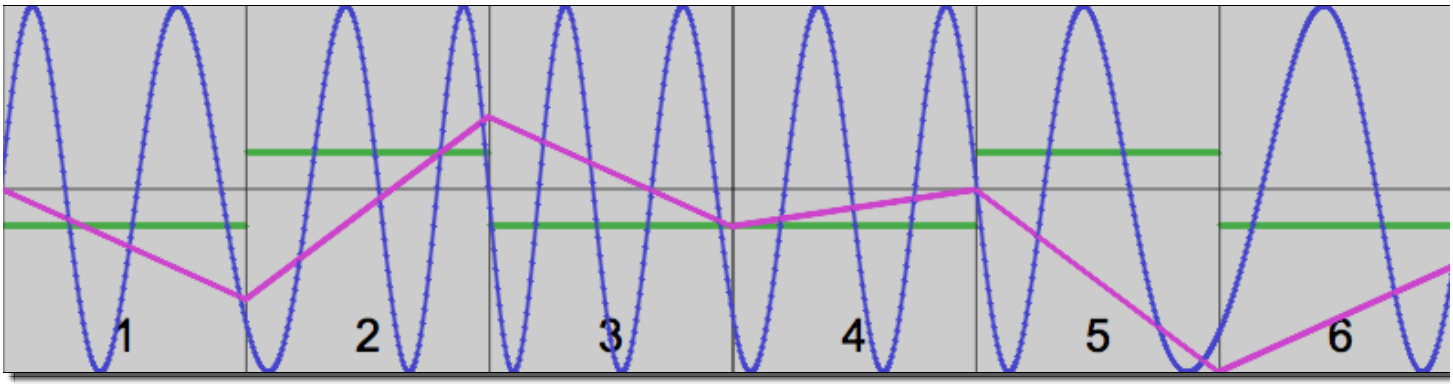




Glissando Modulation

Glissando Modulation is a technique that addresses many of the shortcomings of common modulation schemes. It replaces the constant levels and sharp transitions that traditionally identify symbols with smooth changes in frequency. Hence the term glissando.

The traditional concept of being "at a level" or "at a frequency" is replaced with "changing at this rate". The symbol value is encoded in the rate-of-change of the medium, not the absolute level.



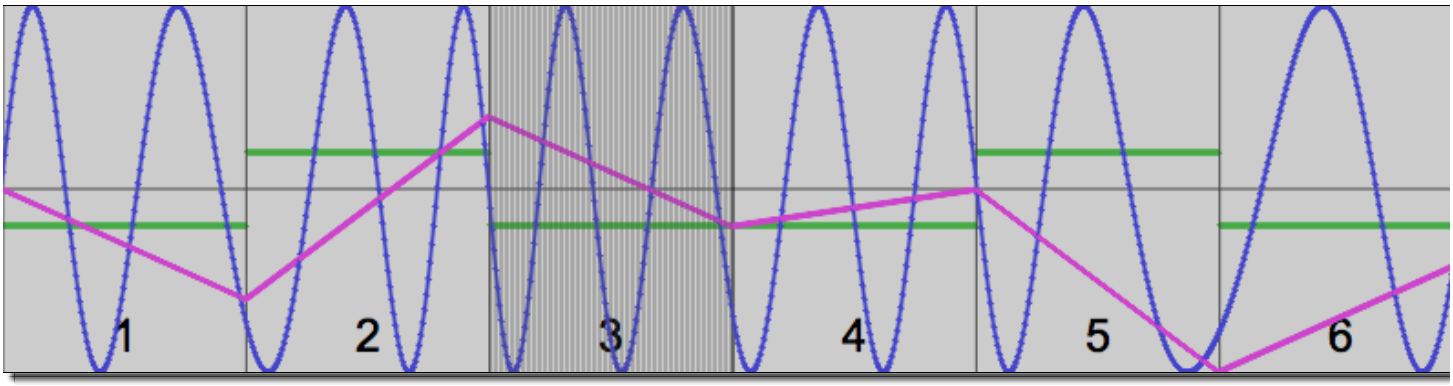
In this graph, the magenta line represents the instantaneous frequency. As a fundamental rule, frequency will be continuously changing in a phase-coherent manner. There will never be rapid transitions in any of the resulting waveforms. Symbol boundaries are ALWAYS indicated by a point of inflection in the frequency change.

In this (manufactured for clarity) example a symbol '0' is indicated by a slow change in frequency during the symbol interval. A symbol '1' is indicated by a rapid change in frequency.

Symbol number four in the example is a second '0' in a row. In order to enforce the self-clocking rule of "no consecutive identical symbols", we introduce an "extra" (third) symbol indicated by a very-slow change in frequency.

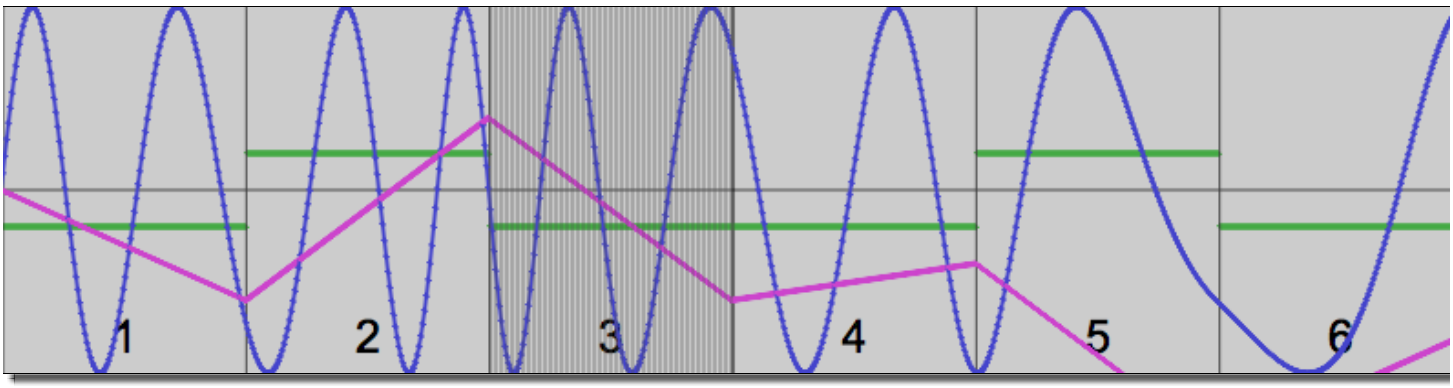
Data Recovery

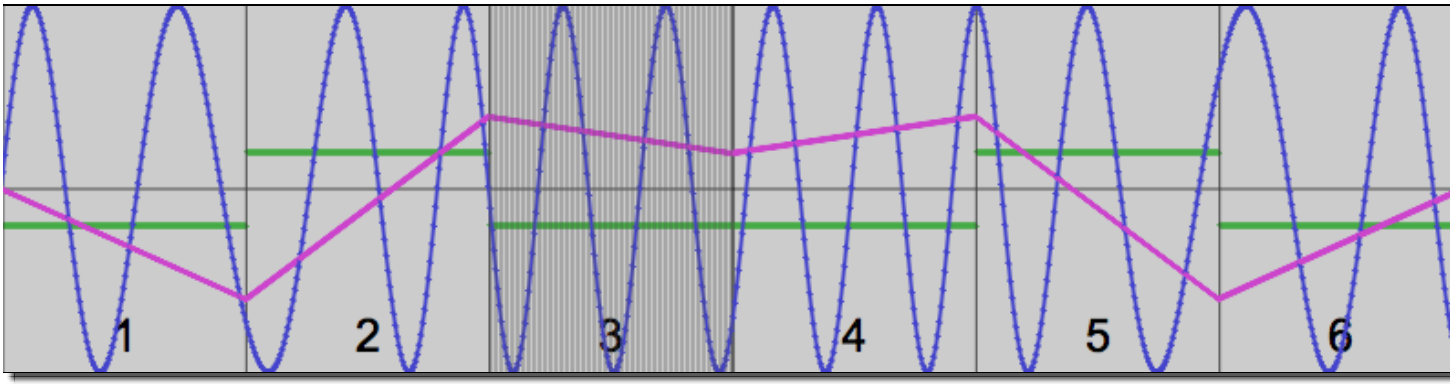
Now let us consider the problem of recovering a lost symbol in the data stream.



These two (incorrect) candidate sequences violate obvious rules and result in a

signal that does not match the subsequent parts of the signal that were correctly received.





Observe that there is only one possible symbol that could fit in the obscured area. Thus, it is possible to recover a symbol value using only the information encoding the surrounding symbols.

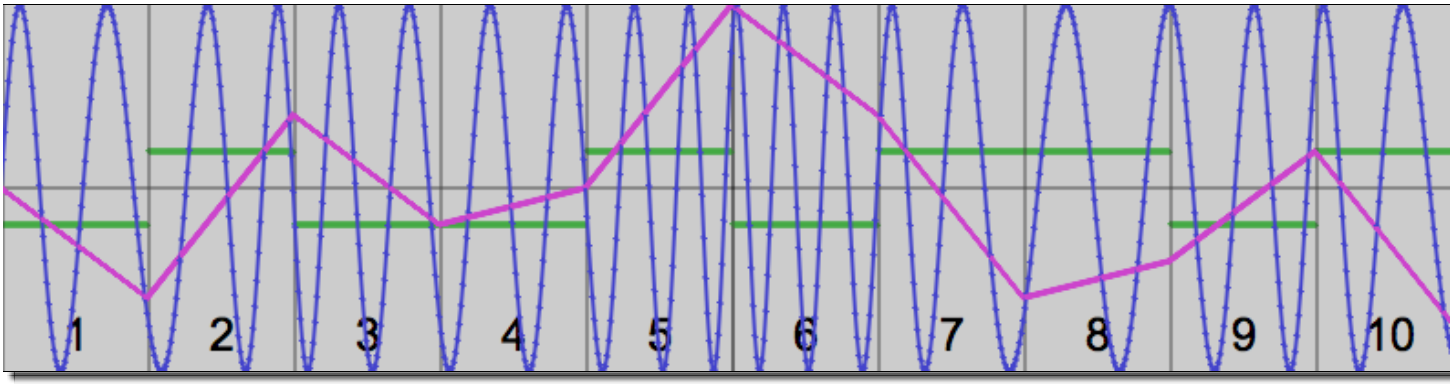
Details

The diagrams above are greatly simplified to emphasize the basic principles. A minimal implementation would use 2^N+1 different sweep rates (slopes), where N is the number of bits per symbol. These Glissandos could be either rising or falling in frequency, thus using a grand total of $2*(2^N+1)$ different sweep rates.

Each of the symbols could be encoded with either a rising or falling Glissando. The transmitter chooses the direction that moves toward the center frequency.

This rule provides additional hints that can be used by the receiver to lock onto an incoming signal. In particular, the frequent changes in direction ensure that

transmitted power will be concentrated near the center frequency. Initial phase and symbol lock may be expedited by looking for this center frequency. In addition, frequency offset and shift (Doppler) can be rapidly compensated for.



This example shows a properly encoded sample of ten binary symbols: 0100101101. There are three possible rising shifts and three possible falling shifts. The shift direction is always toward the center frequency. The second of consecutive identical symbols (4 and 8 in the illustration) are encoded as the smallest glissando. Symbols '0' and '1' are represented by medium and large shifts, respectively.

It is not strictly necessary to introduce the additional run-length-limiting "extra" symbol. Any sequence of identical symbols could be encoded by alternating the direction of the glissando and thus preserving the self-clocking feature at the symbol level. This alternation of encodings might, however take place with an offset some distance from the center frequency. A sophisticated decoder might incorrectly adjust the center frequency based on this erroneous assumption. Alternating in this way violates the "always move toward the center frequency" rule.

Further, it is believed that breaking up any alternating pairs of symbols will be generally beneficial to noise immunity.

The choice of encoding for the "extra" symbol as the minimum-size Glissando requires some review. First, it is presumed that the choice should be for either the minimum-Glissando or the maximum-Glissando. Other data-to-Glissando mappings should probably be assigned in some arbitrary fashion, since it is

presumed that properly structured data will look like a random symbol sequence. Second, given a truly random symbol sequence, the identical-consecutive-symbol situation will occur with a probability of 1/N, where N is the number of normal symbols. This means that the "extra" symbol will be used with a frequency of 1/N, while all of the other symbols will occur slightly less often, at a rate of $1/N - 1/(2N^2)$.

Now the question boils down to this: should the Glissando be minimum, thus tending to concentrate the power near the center frequency, or should the Glissando be maximum, thus tending to spread the signal over the full bandwidth. For the moment I choose the minimum Glissando to be the most likely.

Beat Frequencies

In a real-world implementation each symbol would be represented by a fair number of cycles of the carrier frequency. The number of cycles, and the number of samples for the digital fourier transforms, are kept reasonable by mixing the carrier with a local oscillator (or beat-frequency oscillator).

In a traditional setting, one places a premium on the stability of these oscillators. They are used to tune to the median carrier frequency, and to compensate for clock differences between the sender and receiver.

For recovering a Glissando Modulated signal, it is anticipated that a set of (digitally implemented) swept-frequency oscillators will be used. These controlled-sweep signals, when mixed with the received signal, will yield a pure (single frequency) beat if and only if the sweep rate matches the glissando of the received symbol. This correlation will be identified using traditional digital filters.

Further advances in data recovery will be possible by using this digital mixing and filtering process over spans of many symbols. A candidate symbol sequence is encoded by the receiver, mixed with the sampled received data, and examined for maximum correlation.

Combining knowledge of the possible symbol sequences with the frequency and phase data sampled from the sender enables robust recovery of the data. This improved data recovery takes place before any digital error correction, and thus enables greater noise immunity, higher data rates or longer transmission distances.

In addition, the removal of external, arbitrary fixed references for time and frequency allow the system to be tolerant of clock drift, poor synchronization and Doppler shift.

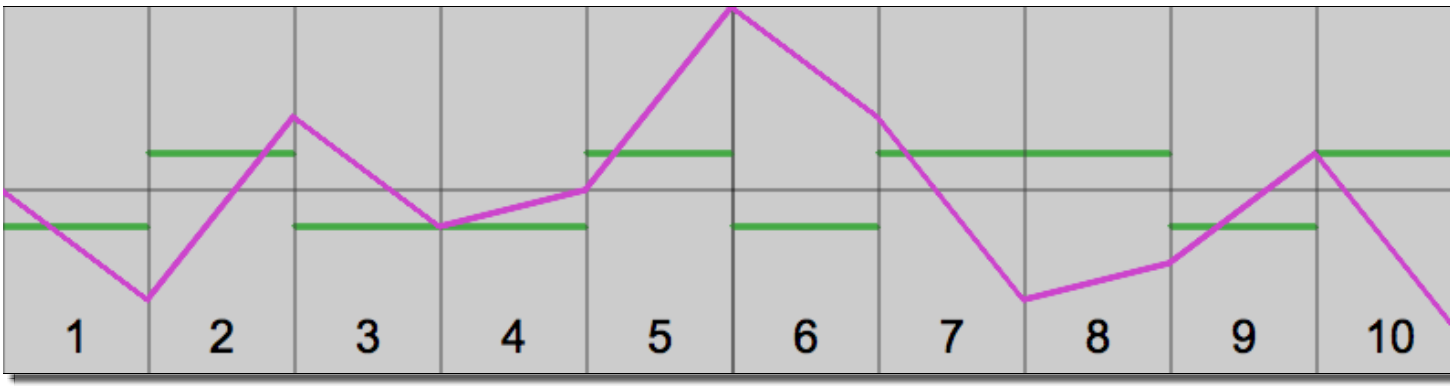
Spread Spectrum

The previous discussion has emphasized modulation and data recovery for narrow-band transmissions. Reducing the bandwidth and improving noise immunity have been the primary considerations. It is also possible to use Glissando Modulation as the basis of a robust spread-spectrum implementation. By making the frequency-changes increasingly large, the same amount of transmitted power can be spread across an entire band. Multiple simultaneous (competing) transmissions would be supported and could be readily differentiated by the receiver. A Code-Division Multiple Access technique would be appropriate, with the receiver looking for correlation with a known pseudo-random pattern. Glissando Modulation would pose essentially no conflict with traditional band utilization. Interference to and from outside sources would be minimal.

Wired Interconnects

Using Glissando coding to represent signal on a wired interface may have advantages in certain situations. Using the rate of change (dV/dt) of the signal for the encoding makes these signals immune to common-mode offsets. The self-clocking feature ensures that the transfer will be immune to clock skew. This is particularly important for high data rates in the case of cables with multiple conductors, or different lengths. The lack of extreme dV/dt edges means that the driver circuitry need not draw as much current on every transition. The driver current requirements will be much more stable over time

and will not introduce as much noise into adjacent circuitry and the environment. Crosstalk between adjacent conductors will be minimized.



Recovery of the data in this case may be a primarily analog process, with capacitive coupling to differentiate the signal slope and digitization of the resulting level. Simple logic could then recover the clock and data stream.

Brian McMillin
28 Apr 2016