Synchronization in FT8

Mike Hasselbeck, WB2FKO

mph@sportscliche.com

Abstract

Deconstruction of the FT8 open-source *FORTRAN* code provides insight on the synchronization algorithm. The heart of the synchronization scheme is a Costas Array, a specially designed square matrix that was invented in 1965 to improve the reliability of underwater sonar. An intuitive explanation of the Costas Array is given, followed by a detailed description of its implementation in the FT8 decoder.

Keywords: FT8, WSJT-X, Costas Array

Introduction. FT8 is a sub-mode of WSJT-X that has become extremely popular for working DX because it enables fast, efficient communication in marginal, weak signal conditions. Forward Error Correction makes this possible, but it requires that transmitting stations and receiver be synchronized to better than 20 ms in time and less than 1 Hz in frequency. Such precision is generally not possible with amateur radio equipment using external reference clocks, so the protocol must supply its own synchronization signal. The different WSJT-X modes accomplish this in a variety of ways, depending on the design requirements. FT8 uses a 7x7 Costas Array [1]. The FT8 decoder performs a coarse search for Costas synchronization symbols in the 15-second frequency waterfall. Synchronization signals that are above a defined energy threshold are identified as candidates for additional decoding. Candidates are adjusted to have initial time and frequency offset accuracy of about 40 ms and 3 Hz. The synch signal of each candidate is then used in a second calculation that performs a quasi-coherent cross-correlation to fine-tune the time and frequency alignment between one or more transmitting stations and the reference frame of the decoder.

Principle of the Costas Array. The following pictorial example can provide an intuitive understanding. An example of a non-Costas array is the diagonal 3x3 matrix M. All the matrix elements are zero except along the diagonal at positions $M_{1,1}, M_{2,2}$, and $M_{3,3}$. Render the matrix as a mask, with transparent holes at the three diagonal positions (Figure 1, left). The mask is the frame of reference. An identical signal array is configured with LEDs at the same positions as the mask holes (Figure 1, right). Each LED contributes 1 unit of signal. To get the maximum light transmission of 3 through the mask, it must be perfectly aligned with the LED array.

Figure 1: 3x3 diagonal mask (left) and LED array (right).

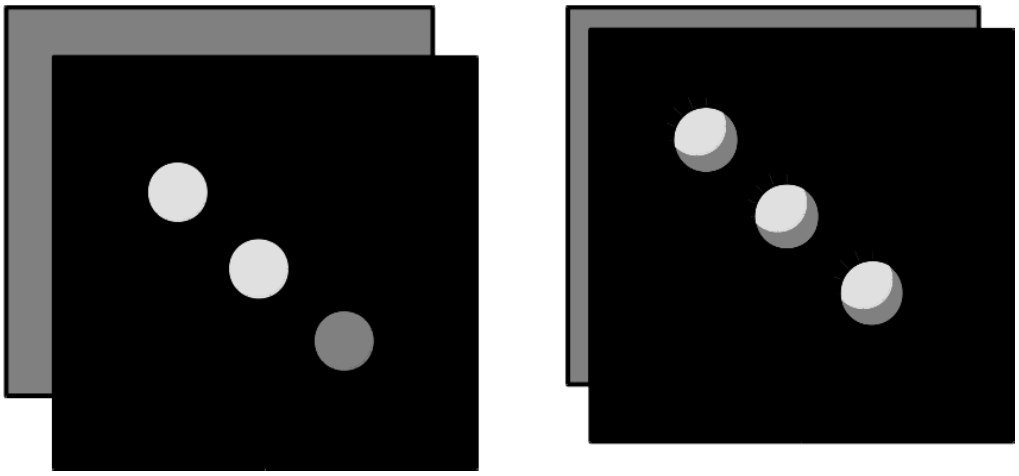

Figure 2: Misalignment of the mask by 1 array element on each axis (left); Transmission approaching maximum (right).

A systematic search of x and y is used to find the maximum brightness. Misalignment by an amount $\Delta x = -1$ and $\Delta y = +1$ is shown in Figure 2, left. Incremental, trial-and-error movement along x and y eventually locates the three holes (Figure 2, right), until the maximum shown in Figure 1 (right) is attained. The amount the mask must displaced to attain maximum brightness measures the x and y offsets of the signal from the reference.

Now assume one of the LEDs is off at either position $M_{1,1}$ or $M_{3,3}$. A maximum transmitted signal of 2 can be obtained at two different alignments: i) correct overlap or ii) a single element displacement in x and y as shown in Figure 2, left. Because the maximum achievable brightness is reduced from 3 to 2, we would know that one LED is off. Loss of one LED, however, introduces alignment ambiguity.

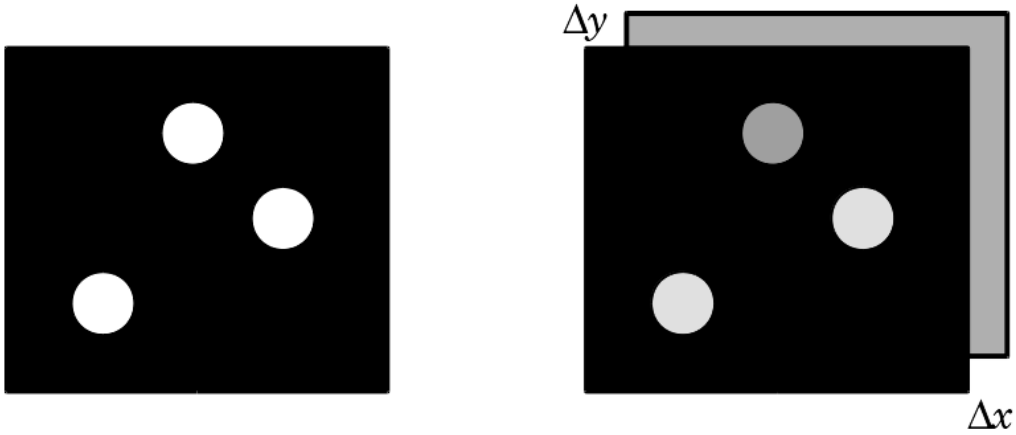

Figure 3: 3x3 Costas Array (left). Position offsets Δx and Δy between signal and reference frames are unambiguously located even with one LED off (right).

The Costas Array shown in Figure 3 (left) solves this problem. Holes and LEDs are located at array positions $M_{1,2}, M_{2,3}$, and $M_{3,1}$. If any one of the three LEDs is dark, the two remaining LEDs can be used to align the mask with no positioning ambiguity (Figure 3, right). Ambiguity returns if two or more LEDs

are off.

The 3x3 Costas Array assures alignment accuracy even when any one of the LEDs is compromised. Dark LEDs represent missing or corrupt data. This situation often occurs on noisy channels such as encountered in sonar, radar, and weak-signal communications. Maintaining critical time and frequency synchronization between transmitted and reflected signals in submarine sonar motivated research in this area by J.P. Costas [2].

In the above example, the Costas Array is used for two-axis (x, y) positional accuracy. In communication applications, the array dimensions are changed to time and frequency.

Costas Array in FT8. The square matrix representing the Costas Array is implemented with discrete time steps (columns) and frequency steps (rows). In FT8, there are 7 sequential time steps and 7 non-sequential frequency steps. The row values are the integers 3,1,4,0,6,5,2 and the time steps run from 1 to 7 as depicted in Figure 4. This is just one of the 200 possible 7x7 Costas Arrays [3]. The unique footprint of the Costas Array allows unambiguous alignment of the data stream at the receiving station. The decoder will look for the expected time sequence of symbols in the received message and attempt to establish the needed temporal and frequency synchronization between the two stations.

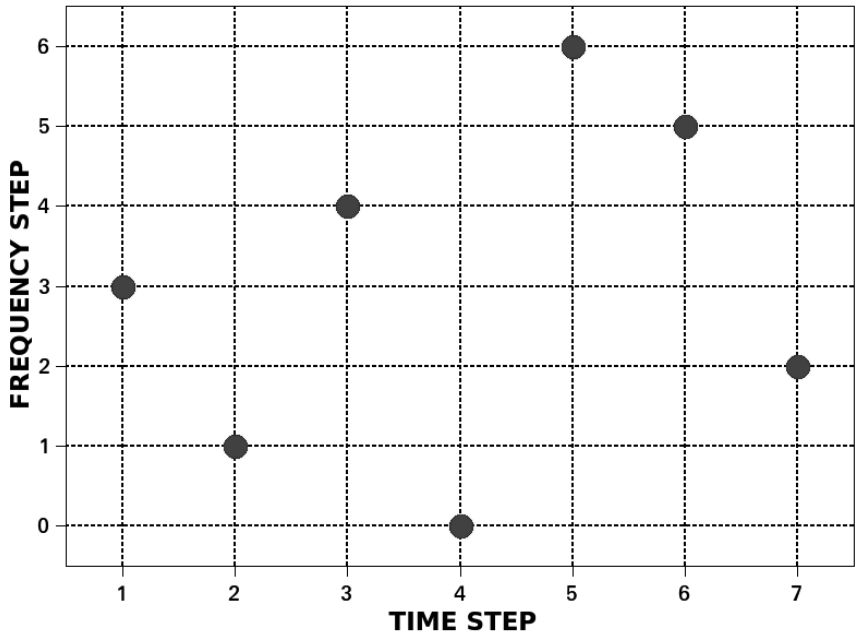

Figure 4: 7x7 Costas Array used in FT8 v2. The time and frequency steps are 160 ms and 6.25 Hz, respectively.

The 7x7 Costas Array is rendered by 7 of the 8 available FT8 tones, occupying 7 time steps. The sequence of tones is 3,1,4,0,6,5,2. Each tone is one symbol of duration 160 ms and 7 symbols require 1.12 seconds. To compensate for drift and fading that may occur over the 12.64 second duration of an FT8 transmission and to accommodate time synchronization offsets in the range: $-2 \leq \Delta t \leq +3$ seconds, the same Costas array is inserted at the beginning, middle, and end of each message. This means that 26.6% of each FT8 message is allocated to synchronization.

Figure 5 (left) is a time plot of the imaginary part of the 7x7 Costas Array sampled at 12000 S/sec. This signal modulates the audio carrier frequency f_c selected by the FT8 operator for transmission. In version 2.1 of WSJT-X, Frequency Shift Keying (FSK) has been replaced with Gaussian FSK (GFSK). Transitions

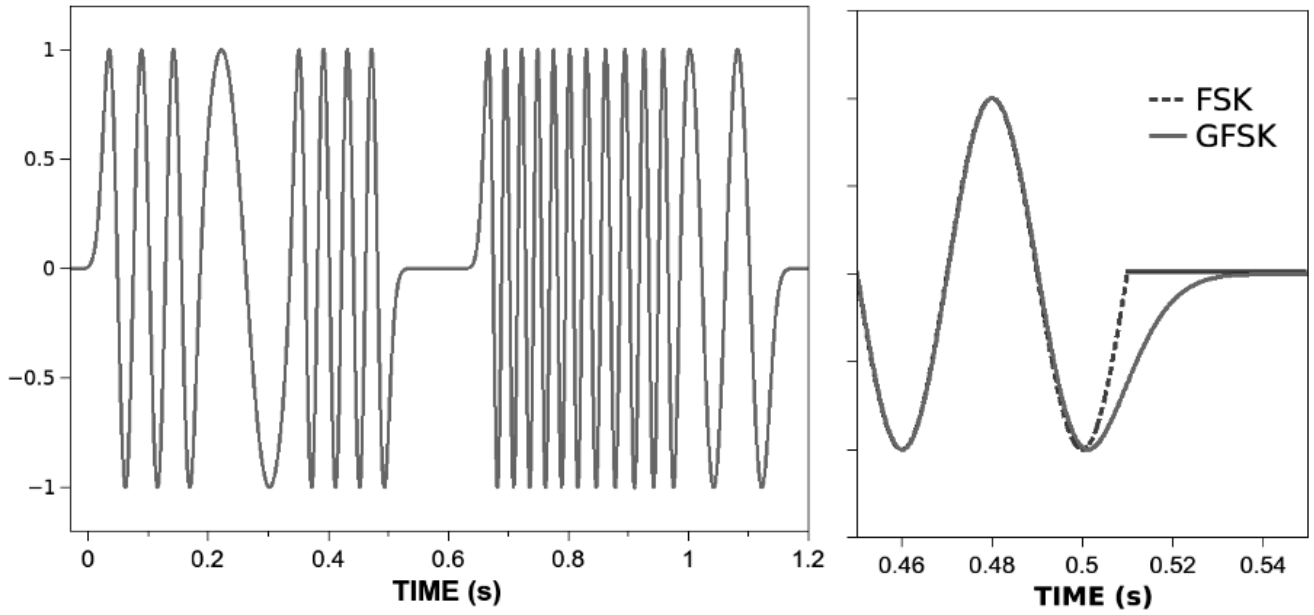

Figure 5: Left: The 7x7 Costas Array rendered as a sequence of GFSK symbols at 6.25 Hz multiples. The audio carrier frequency f_c has been set to zero for clarity. Right: Comparison of FSK and GFSK at the transition between Tone 4 and Tone 0.

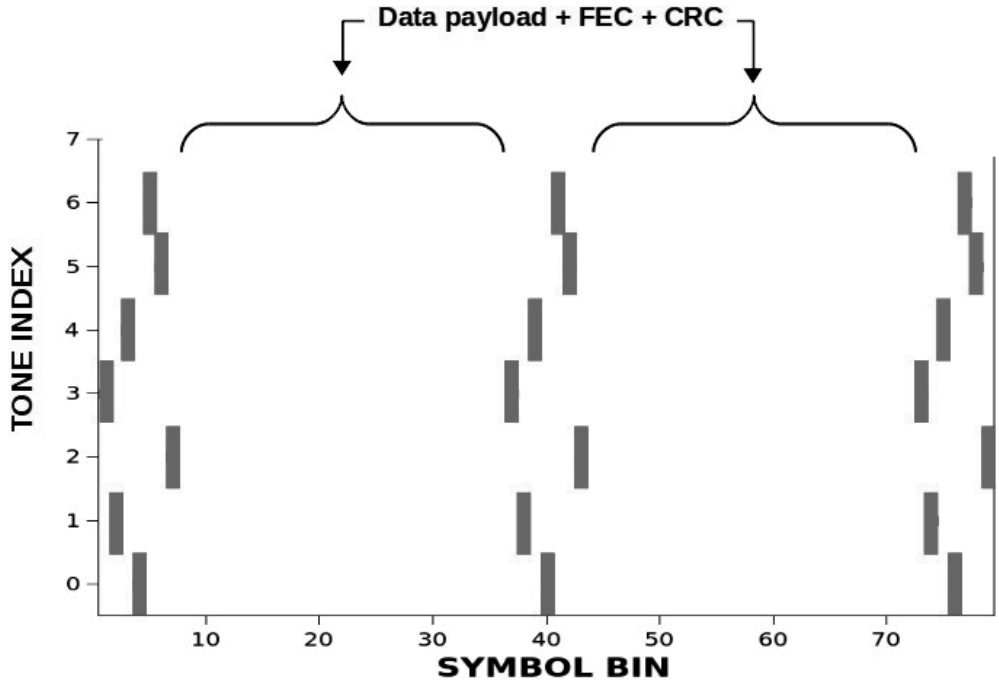

Figure 6: Arrangement of the 7x7 Costas Array in the transmitted FT8 signal. The synch tones occupy 21 of the 79 symbol bins. The information bits, Forward Error Correction, and Cyclical Redundancy Check are placed in the remaining 58 bins.

between the tones (symbols) are not as abrupt (Figure 5, right), resulting in a significant reduction of signal bandwidth [4].

GFSK modulation is identical for the 21 synchronization tones and 58 encoded message symbols. Legacy FT8 (in versions 1.9 and earlier) uses a 7x7 Costas Array that is the reverse sequence of the current version, ie. 2,5,6,0,4,1,3. This was changed in version 2 so that the new decoder could potentially recognize signals from earlier versions of the program. The location of synch tones, however, remains the same (Figure 6).

There is additional bandwidth required for each symbol (tone) to accommodate the baud rate, ie. 6.25 Hz. Tone 0 occupies frequency $f_0 = f_c \pm 3.125$ Hz. Tone 7 is $f_7 = f_c + 43.75 \pm 3.125$ Hz. The total FT8 signal resides in the frequency span: $f_0 = f_c - 3.125$ to $f_7 = f_c + 43.75 + 3.125$ Hz, which defines the 50 Hz modulation bandwidth.

Decoding. Signals may appear on a multitude of carrier frequencies anywhere in the receiver audio passband with an unknown time offset relative to the receiver clock. Figure 7 illustrates two trains of FT8 synchronization symbols. The audio carrier frequencies are located at Tone 0; the time offset between the two signals is slightly greater than one symbol bin.

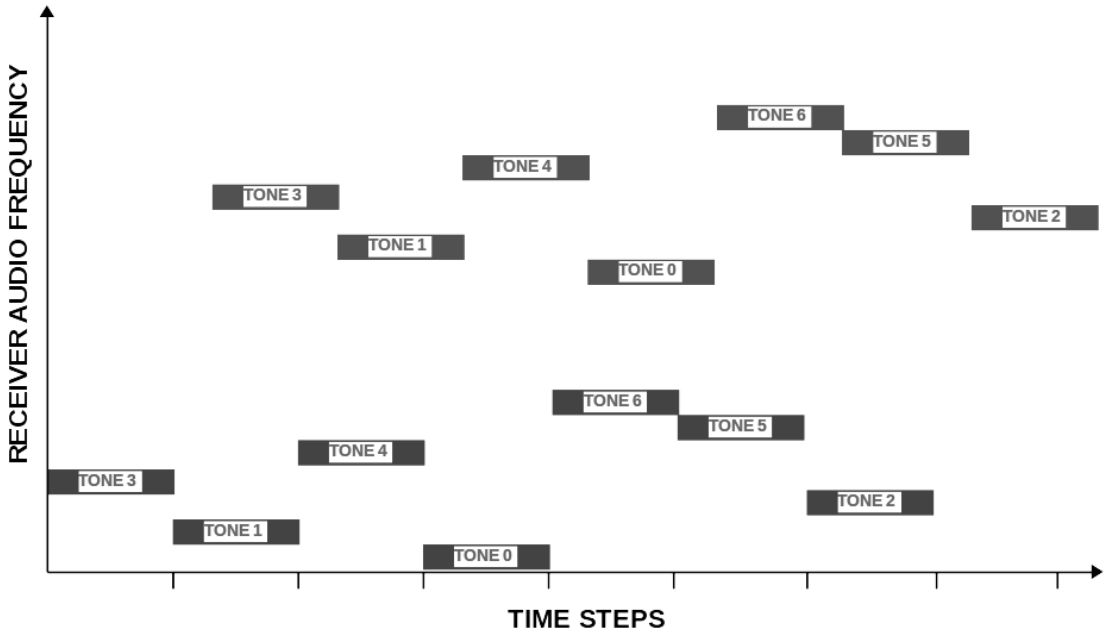

Figure 7: Costas Array symbol sequence for two FT8 signals in the receiver passband. The signals are offset in time by slightly greater than one symbol bin (0.16 s/symbol). Eight of the 79 symbol bins are shown.

The first task of the decoder is to establish time synchronization with any incoming data. This is accomplished in a series of operations depicted by the block diagram in Figure 8. Each block is described in more detail below.

The received FT8 signal is sampled at 12000 S/sec for 15 seconds, generating 180,000 16-bit audio samples. This corresponds to 180k x 16 = 2.88 Mb of data. Decoding does not commence until all the data has been acquired.

The decoder begins by searching the data for synchronization signals. This is handled by the *FORTRAN* subroutine sync8.f90 in the FT8 source code. The audio energy spectrum is calculated at sequential, partially overlapping time windows. The time increment is 1/4 of the duration of a single FT8 symbol,

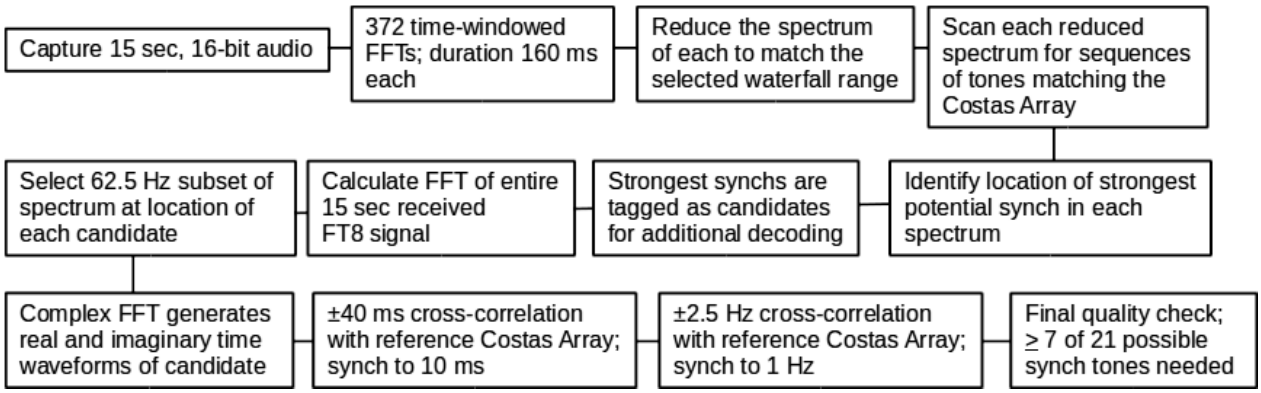

Figure 8: Block diagram of the FT8 synchronization process.

i.e. 40 ms. 372 individual spectra are obtained, with the last 120 ms of the 15 second data capture ignored. Spectra are generated by performing a sequence of 372 time-partitioned Fast Fourier Transforms (FFT_i where i=1,2,...372) with data collected from 160 ms time windows, but interleaved by 1/4 symbol. For example, FFT_1 is evaluated for the interval 0–160 ms, FFT_2 is 40–200 ms, FFT_3 is 80–240 ms, FFT_4 is 120–280 ms up to FFT_{372} for the final 160 ms time segment. This is illustrated for the first $\sim$ 1 second of the received signal in Figure 9.

Each 160 ms sample interval is zero-padded to produce a 320 ms input signal for the FFT calculation; the FFT_i spectra span a frequency range from 3.125 Hz to 6 kHz. The operator sets the FT8 waterfall frequency range considerably smaller than this, eg. 200–2500 Hz, so only the calculated FFT range matching the waterfall is used.

The decoder next attempts to obtain time synchronization between the receiving station and a possible transmitting station at an audio baseline frequency f_c. There can be many received stations in the audio waterfall, each with a different f_c and time synch offset. The data at f_c is scanned for the correct synch tones employing 125 different start times t_0 in the range $t_0 = 0.5 \pm 2.5$ seconds, where the +0.5 second offset accounts for the delayed start of transmit. The time search increment is also 40 ms.

Starting at the lowest frequency that was set in the waterfall (eg. f_c = 200 Hz) the decoder sums the

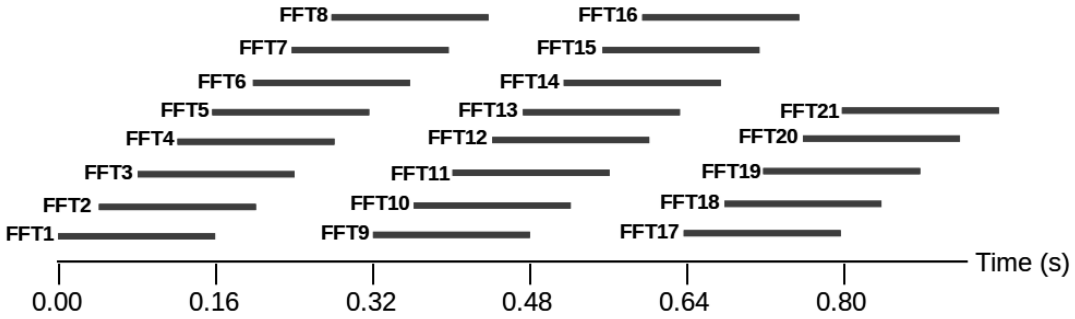

Figure 9: Searching for the synch signal at the start of the captured FT8 waveform. FFTs are calculated in the time intervals depicted by the horizontal bars. If the transmitter and receiver are perfectly aligned, the 7 elements of the first Costas array will be in the intervals FFT1, FFT5, FFT9, FFT13, FFT17, FFT21, and FFT25 (not shown). The decoder scans 125 different time offsets looking for the synch tones. The vertical axis in this plot is meaningless.

spectral energy density at the expected locations of the 7 Costas synch signals. As an example, when t_0 = 0 the energies at the following 7 frequency locations are summed: f_c + 18.75 Hz in FFT_1, f_c + 6.25 Hz in FFT_5, f_c + 25 Hz in FFT_9, f_c in FFT_{13}, f_c + 37.5 Hz in FFT_{17}, f_c + 31.25 Hz in FFT_{21}, and f_c + 12.5 Hz in FFT_{25}. These are slices of the time-partitioned energy spectrum corresponding to the Costas array sequence [3,1,4,0,6,5,2]. This synch sum t_a is normalized to the spectral content in all of the lowest 7 frequency bins for each of the 7 time intervals. The normalization sum t_{0a} is obtained by adding frequency bins 0–6 in FFT_1, FFT_5, FFT_9, FFT_{13}, FFT_{17}, FFT_{21}, and FFT_{25}. There are 7 components in the sum t_a and 49 components in t_{0a}.

This calculation is repeated at the expected location of the remaining two synch signals. For the present example with t_0 = 0, these are time positions 5.76 seconds (Costas tones expected in FFT_{144}, FFT_{148}, FFT_{152}, FFT_{156}, FFT_{160}, FFT_{164}, FFT_{168}) and 11.52 seconds (Costas tones expected in FFT_{288}, FFT_{232}, FFT_{236}, FFT_{240}, FFT_{244}, FFT_{248}, FFT_{252}). This produces synch sums t_b and t_c and normalization sums t_{0b} and t_{0c}, respectively. The decoder then generates sums $t = t_a + t_b + t_c$ and $t_N = (t_{0a} + t_{0b} + t_{0c} - t_a - t_b - t_c)/6$. t_N represents the averaged energy content of the 6 nominally empty frequency bins at each of the 7 time positions. The final normalized synch signal is $S_{abc} = t/t_N$.

For each f_c, the decode algorithm looks for the maximum total synch signal while scanning 125 time increments in the range $-2 \le t_0 \le 3$ seconds. For start times $t_0 < 0$, some or all of the Costas synch signal t_a will not be available. To accommodate this, a second normalized synch signal S_{bc} is calculated as above, except t_a and t_{0a} are ignored. The larger value of either S_{abc} or S_{bc} is recorded as the total synch signal at f_c for each of the 125 start times. Normalization allows S_{abc} and S_{bc} to be directly compared despite their different data counts.

The frequency f_c is incremented by 3.125 Hz and the above time synch calculation repeated. The frequency iteration process continues until the highest frequency that was set in the waterfall is reached. For waterfall audio in the range f_c = 200–2500 Hz, there will be 737 x 125 = 92,125 synchronization calculations performed for each 15-second FT8 receive interval.

A 2-dimensional array holds the best (strongest) total synch signal for each f_c and the corresponding index of its time offset. This array is sorted from weakest to strongest signals using the *FORTRAN* subroutine in indexx.f90 [5]. A baseline synch signal is established at the approximate midway point of the sorted array range; all synch signals in the array are normalized to this value. This means only about half of the captured audio spectrum will have synch signals with a normalized value ≥ 1.

Starting with the strongest normalized synch signal, the decoder checks for values ≥ 1.5. If the strongest normalized synch signal is < 1.5, the decoder exits. Signals above the 1.5 threshold are tagged as *candidates* for further decoding. The audio baseline frequency f_c, time synch offset, and normalized synch signal of each candidate are recorded. As many as 200 candidate signals can be acquired. If there are two candidates within 4 Hz of each other, however, the weaker candidate is discarded.

At this point, synchronization has been established to 3.125 Hz and 40 ms for every candidate signal in the waterfall display. Greater accuracy is possible using coherent detection with the *FORTRAN* subroutine ft8b.f90.

The first operation in ft8b is to reacquire the spectrum of the complete 15-second FT8 signal. This is accomplished with a call to the subroutine ft8_downsample.f90, which performs a FFT. A frequency resolution of 0.0625 Hz is obtained by zero-padding the time signal by 1 second to make 192,000 total time steps at 12000 samples/second.

There is a carrier frequency f_c associated with each candidate. The decoder locates f_c in the spectrum and slices out the portion $f_c - 9.375$ Hz to $f_c + 53.125$ Hz representing 1000 sample points. f_c is subtracted

from this array subset to baseline the spectrum at $f_c = 0$ Hz. An inverse FFT is then performed on the frequency-shifted spectrum to generate a complex time domain signal (i.e. in-phase and quadrature, phase-shifted +90°) without the carrier. The 16-second complex temporal waveforms have 3200 points, giving 5 ms resolution. Each candidate signal is re-cast in complex form to enable tighter synchronization and eventual decoding.

The decoder enters subroutine sync8d.f90 and generates a complex reference waveform with $k = 7$ symbols $R_{k,m}$ representing the FT8 Costas Array. Because the correlation calculations make a heavy demand on the processor, the sampling rate is reduced by a factor 60 compared to the waveform shown in Figure 5. The Costas Array used in this subroutine has $m = 32$ samples/symbol separated by 5 ms. GFSK smoothing is not effective at the coarser resolution, so the reference waveforms are rendered with standard FSK (upper 2 traces in Fig. 10).

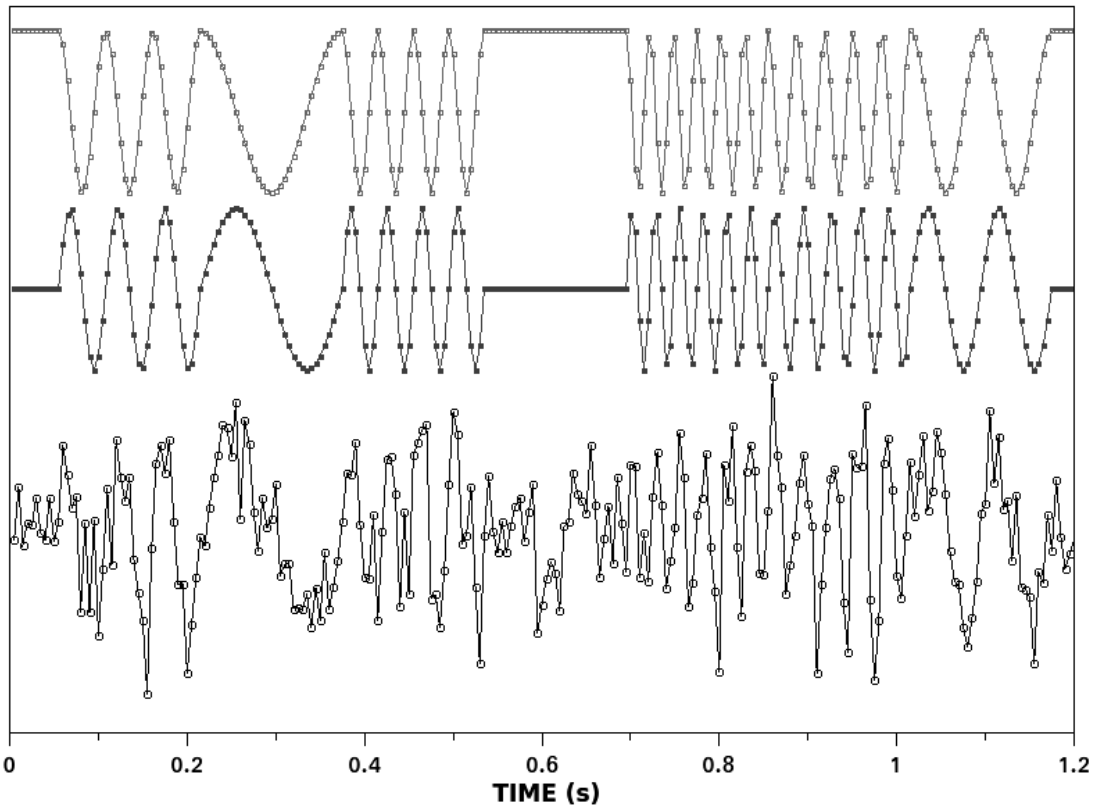

Figure 10: Real (top) and imaginary (middle) components of the reference complex Costas Array waveform with 32 points/symbol as generated by subroutine sync8d.f90. There is padding with Tone 0 on both sides for the cross-correlation calculations. The bottom waveform depicts the imaginary component of a simulated received signal with Gaussian amplitude white noise; signal-to-noise ratio is 2:1. Not shown is a similar noisy waveform for the real component. Waveforms are displaced vertically for clarity.

The candidate signal is trimmed to 3125 data points separated by 5 ms and contains up to three Costas Array synch waveforms $C_{j,k,m}$ located at the beginning (j=1), middle (j=2), and end (j=3). Each candidate has time synchronization already established to ± 40 ms, so it is only necessary to search within this 80 ms range with $n = 17$ time-step iterations of 5 ms/step. This is implemented with a cross-correlation between the received symbols C and reference symbols R in the Costas Array as follows. A complex product is calculated for each received symbol C at each Costas Array location j:

$$z_{j,k} = \sum_{m=1}^{32} C_{j,k,m} R^*_{k,m} \tag{1}$$

where each of the $m = 1{-}32$ elements (i.e. sample points) of a symbol C is multiplied with the corresponding element in the complex conjugate of the reference symbol R^*. The squared amplitude of each calculation in Equation (1) is summed to produce the aggregate synchronization signal:

$$T_n = \sum_{k=1}^{7} \left[|z_{1,k}|^2 + |z_{2,k}|^2 + |z_{3,k}|^2 \right] \tag{2}$$

T_n is evaluated at $n = 17$ different time alignments between C and R: $\Delta t = -40, -35, ...0... + 35, +40$ ms. At each time step n, $j \times k = 21$ individual correlations are performed, each with duration of one symbol, i.e. 160 ms. The decoder looks for the maximum value of T_n and sets this as the final time synchronization point.

Long duration correlations using the entire 7-symbol sequence of the Costas Array can, in principle, generate sharper peaks. This is not practical because signal coherence may be lost over the propagation path on a timescale approaching 1 second. Moreover, the carrier phase may be misaligned with respect to the reference Costas Array waveform anywhere in the range $0{-}2\pi$. The summed correlation power of individual symbols given by Equation (2) produces a useful peak without any knowledge of the carrier phase offset. Coherence is only required for the duration of each 160 ms symbol.

The operation described above can be illustrated by calculating cross-correlations of the reference Costas Array with simulated received signals. Representative waveforms are shown in Figure 10. The top and middle waveforms are the real and imaginary (quadrature) components, respectively, of the reference Costas Array, i.e. without noise. The bottom waveform shows the imaginary component with additive Gaussian white noise at a signal-to-noise ratio of 2:1 as might be encountered with a weak received signal [6].

Calculated cross-correlations for a single FT8 Costas Array [3,1,4,0,6,5,2] are shown in Figure 11, representing just one term on the right side of Equation (2). A noise-free sequence of 7 symbols establishes the synchronization point to better than 10 ms (solid curve). The three dashed curves show the effect of additive Gaussian white noise at a signal-to-noise ratio of 2:1. The correct synchronization point is still located with an accuracy of ± 10 ms. This illustration is for a single Costas Array sequence; there may be as many as two more additional signals z_j available on the received signal to produce a useful maximum in T.

Fine frequency adjustment is performed with a cross-correlation using the same subroutine sync8.f90 and procedure outlined above. The frequency is scanned in $n = 11$ increments of $\Delta f = 0.5$ Hz/step over the range ± 2.5 Hz. This corresponds to an incremental phase scan of $\Delta\phi = -4.5°, -3.6°, ...0°... + 4.5°$ in an attempt to peak the cross-correlation T_n. This second cross-correlation brings frequency synchronization to within 1 Hz. The time and frequency corrections determined by the two cross-correlations are then applied to the corresponding candidate data set to establish the desired synchronization.

Before attempting to decode the message contained in the 58 information bins, the ft8b subroutine makes a final quality check of the optimized synchronization. A sequence of 79 time-partitioned FFTs is performed at the time slot of every tone in the entire candidate signal. Each 160 ms symbol has 32 sample points that are used by the FFT algorithm to calculate its complex spectrum. The first 8 points in the FFT data array correspond to the energy content in the 8 FT8 tones; only these 8 frequency bins are of interest. The FT8 tone with the largest magnitude is assigned to that bin. In this way, the 21 synchronization bins are assigned one of the FT8 tones. The information bins are evaluated with a separate procedure.

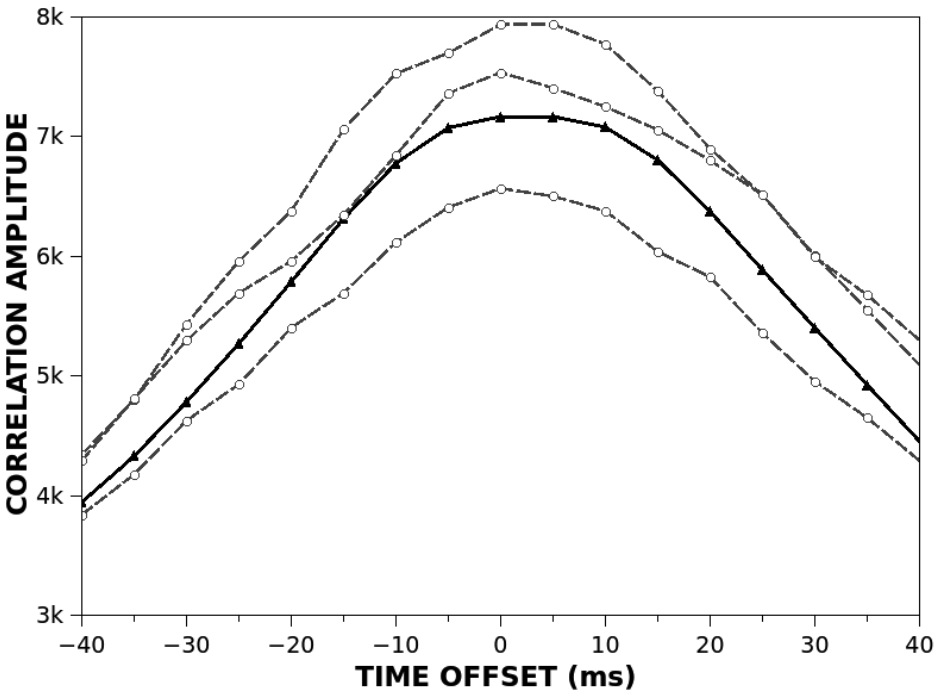

Figure 11: Simulated cross-correlations using a single FT8 Costas Array [3,1,4,0,6,5,2]. A noise-free, perfectly phase-aligned correlation is depicted with the solid black line. The dashed curves are for signals having carrier phase randomly aligned with the reference plus additive Gaussian white noise; signal-to-noise 2:1.

The next step is to compare the tones that have been assigned to bins 1–7, 37–43, and 73–79 with the tones expected for the Costas Array, i.e. the tone sequence 3,1,4,0,6,5,2 as shown in Figure 6. Three error-free sequences results in the maximum possible 21 matches. Because of the powerful Forward Error Correction present in the coded messages, the decoder can reliably proceed with as little as 7 matches. Fewer than 7 matches will cause the decoder to give up and evaluate the next candidate.

Acknowledgements. Helpful dialogue was provided by J. Frazier (KC5RUO), P. Karn (KA9Q), S. Franke (K9AN), and J. Taylor (K1JT).

Disclaimer: This is not official documentation for the WSJT-X open source project. It represents the author's best understanding of the FT8 synchronization scheme. There may be errors and misconceptions.

Updates to this document will be posted at: www.sportscliche.com/wb2fko/tech.html.

References

[1] J. Taylor, S. Franke, B. Somerville "Work the World with WSJT-X, Part 2: Codes, Modes, and Cooperative Software Development", QST (Nov. 2017)

[2] J.P. Costas, "A study of a class of detection waveforms having nearly ideal range-doppler ambiguity properties", Proc. IEEE **72**, 996 (1984).

[3] S.W. Golomb and H. Taylor "Construction and properties of Costas arrays", Proc. IEEE **72**, 1143 (1984).

[4] J. Taylor, S. Franke, B. Somerville "The FT4 Protocol for Digital Contesting", WSJT-X Developer Technical Note (April 22, 2019).

[5] B.P. Flannery, S. Teukolsky, W.T. Vetterling, W.H. Press, *Numerical Recipes in FORTRAN: The Art of Scientific Computing*, Cambridge:NY (1997).

[6] The simulation ignores leakage resulting from windowing in the inverse FFT.