Steven J. Franke, K9AN and Joseph H. Taylor, K1JT

Address: 3314 E Anthony Dr., Urbana, IL 61802; **s-franke@illiniois.edu;** 272 Hartley Ave, Princeton, NJ 08540; **k1jt@arrl.org**

Open Source Soft-Decision Decoder for the JT65 (63,12) Reed-Solomon Code

Under-the-hood description of the JT65 decoding procedure, including a wholly new algorithm for its powerful error-correcting code.

1 — Background and Motivation

The JT65 protocol has revolutionized amateur-radio weak-signal communication by enabling operators with small or compromise antennas and relatively low-power transmitters to communicate over propagation paths not usable with traditional technologies. The protocol was developed in 2003 for Earth-Moon-Earth (EME, or "moonbounce") communication, where the scattered return signals are always weak.[1] It was soon found that JT65 also enables worldwide communication on the HF bands with low power, modest antennas, and efficient spectral usage. Thousands of amateurs now use JT65 on a regular basis, making contacts on all bands from 160 meters through microwaves.

JT65 uses timed transmitting and receiving sequences one minute long. Messages are short and structured so as to streamline minimal exchanges between two amateur operators over potentially difficult radio paths. Most messages contain two callsigns and a grid locator, signal report, acknowledgment, or sign-off; one of the tokens CQ, QRZ, or DE may be substituted for the first callsign. Alternatively, a message may contain up to 13 characters of arbitrary text. All messages are efficiently compressed into exactly 72 bits of digital information. It should be obvious that the JT65 protocol is intended for the basic purpose of completing legitimate, documented two-way contacts, but not for extended conversations. Full details of the message structure and encoding procedure were presented in an earlier publication.[1] For a concise description of the overall process of transmitting and receiving a JT65 message, see the accompanying sidebar *JT65 Message Processing*.

A major reason for the success and popularity of JT65 is its use of a strong error-correction code. Before transmission, each 72-bit message is divided into 12 six-bit *symbols* and augmented with 51 additional symbols of error-correcting information. These 51 *parity symbols* are computed according to information-theory rules that maximize the probability of correctly decoding the message, even if many symbols are received incorrectly. The JT65 code is properly described as a short block-length, low-rate Reed-Solomon code based on a 64-symbol *alphabet*. Characters in this alphabet are mapped onto 64 different frequencies for transmission.

Reed Solomon codes are widely used to ensure reliability in data transmission and storage. In hardware implementations, decoding is generally accomplished with a procedure such as the Berlekamp-Massey (BM) algorithm, based on *hard decisions* for each of the symbol values received. *Soft decisions* are potentially more powerful, however. For each received JT65 symbol we can estimate not only the value most likely to be correct, but also the second, third, etc., most likely. Most importantly, we can also estimate the probability that each of those possible values is the correct one. Decoders that make use of such information are called *soft-decision decoders*.

Until now, nearly all programs implementing JT65 have used the patented Kötter-Vardy (KV) algebraic soft-decision decoder, licensed to and implemented by K1JT as a closed-source executable for use only in amateur radio applications.[2] Since 2001 the KV decoder has been considered the best known soft-decision decoder for Reed Solomon codes.

We describe here a new open-source alternative called the Franke-Taylor (FT, or K9AN-K1JT) soft-decision decoding algorithm. It is conceptually simple, built on top of the BM hard-decision decoder, and in this application it performs even better than the KV decoder. The FT algorithm is implemented in the popular programs *WSJT*, *MAP65*, and *WSJT-X*, widely used for amateur weak-signal communication using JT65 and other specialized digital protocols. These programs are open-source, freely available, and licensed under the GNU General Public License.[3]

The JT65 protocol specifies transmissions

[1]Notes appear on page 17

that start one second into a UTC minute and last for 46.8 seconds. Receiving software therefore has more than ten seconds to decode a message before the start of the next minute, when the operator will send a reply. With today's personal computers, this relatively long time encourages experimentation with decoders of high computational complexity. With time to spare, the FT algorithm lowers the decoding threshold on a typical fading channel by many decibels over the hard-decision BM decoder, and by a meaningful amount over the KV decoder. In addition to its excellent performance, the new algorithm has other desirable properties, not least of which is its conceptual simplicity. Decoding performance and computational complexity scale in a convenient way, providing steadily increasing soft-decision decoding gain as a tunable parameter is increased over more than five orders of magnitude. Appreciable gain is available from our decoder even on very simple (and relatively slow) computers. On the other hand, because the algorithm benefits from a large number of independent decoding trials, further performance gains should be achievable through parallelization on high-performance computers.

The remainder of this paper is organized as follows. Section 2 presents a brief overview of the nature of Reed Solomon codes and their error-correcting capabilities. Section 3 provides statistical motivation for the FT algorithm, and Section 4 describes the algorithm in full detail. Material in these two sections is important because it documents our approach and underlines its fundamental technical contributions. These sections are heavier in formal mathematics than common in *QEX*; for this reason, some readers may choose to skip or skim them and proceed more quickly to the results. Most readers will benefit by reviewing the original paper on the JT65 protocol.[1] A procedure for *hinted decoding* — determining which one, if any, of a list of likely messages matches the one that was received — is outlined in Section 5. Finally, in Section 6 we present performance measurements of the FT and hinted decoding algorithms and make explicit comparisons to the BM and KV decoders familiar to users of older versions of *WSJT*, *MAP65* and *WSJT-X*. Section 7 summarizes some on-the-air experiences with the new decoder. Refer to the sidebar *Glossary of Specialized Terms* for brief definitions of some potentially unfamiliar language.

2 — JT65 Messages and Reed Solomon Codes

The JT65 message frame consists of a short, compressed 72-bit message encoded for transmission with a Reed-Solomon code. Reed-Solomon codes are *block codes* characterized by n, the length of their *codewords*; k, the number of message symbols conveyed by the codeword; and the transmission alphabet, or number of possible values for each symbol in a codeword. The codeword length and the number of message symbols are specified with the notation (n, k). JT65 uses a (63,12) Reed-Solomon code with an alphabet of 64 possible values for each symbol. Each of the 12 message symbols represents $\log_2 64 = 6$ message bits. The source-encoded message conveyed by a 63-symbol JT65 frame thus consists of 72 information bits. The JT65 code is *systematic*, which means that the 12 message symbols are embedded in the codeword without modification and another 51 parity symbols derived from the message symbols are added to form a codeword of 63 symbols.

In coding theory the concept of *Hamming distance* is used as a measure of disagreement between different codewords, or between a received word and a codeword. Hamming distance is the number of code symbols that differ in two words being compared. Reed-Solomon codes have guaranteed minimum Hamming distance d, where

$$d = n - k + 1. \qquad (1)$$

With n = 63 and k = 12 the minimum Hamming distance of the JT65 code is d = 52. With 72 information bits in each message, JT65 can transmit any one of $2^{72} \approx 4.7 \times 10^{21}$ possible messages. The codeword for any message differs from every other codeword in at least 52 of the 63 symbol positions.

A received word containing some *errors* (incorrect symbols) can be decoded into the correct codeword using a deterministic, algebraic algorithm provided that no more than t symbols were received incorrectly, where

$$t = \left\lfloor \frac{n-k}{2} \right\rfloor . \qquad (2)$$

For the JT65 code t = 25, so it is always possible to decode a received word having 25 or fewer symbol errors. Any one of several well-known algebraic algorithms, such as the BM algorithm, can carry out this hard-decision decoding. Two steps are necessarily involved in this process. We must (*1*) determine which symbols were received incorrectly, and (*2*) find the correct value of the incorrect symbols. If we somehow know that certain symbols are incorrect, that information can be used to reduce the work involved in step (*1*) and allow step (*2*) to correct more than t errors. In the unlikely event that the location of every error is known, and if no correct symbols are accidentally labeled as errors, the BM algorithm can correct up to $d - 1 = n - k$ errors.

The FT algorithm creates lists of symbols suspected of being incorrect and sends them to the BM decoder. Symbols flagged in this way are called *erasures*. With perfect erasure information up to $n - k = 51$ incorrect symbols can be corrected for the JT65 code. Imperfect erasure information means that some erased symbols may be correct, and some other symbols in error. If s symbols are erased and the remaining $n - s$ symbols contain e errors, the BM algorithm can find the correct codeword as long as

$$s + 2e \le d - 1 . \qquad (3)$$

If $s = 0$, the decoder is said to be an *errors-only* decoder. If $0 < s \le d - 1$, the decoder is called an *errors-and-erasures* decoder. The possibility of doing errors-and-erasures decoding lies at the heart of the FT algorithm. On that foundation we have built a capability for using soft information on the reliability of individual symbol values, thereby producing a soft-decision decoder.

3 — Statistical Framework

The FT algorithm uses the estimated quality of received symbols to generate lists of symbols considered likely to be in error, thus enabling decoding of received words with more than 25 errors. Algorithms of this type are generally called *reliability-based* or *probabilistic* decoding methods.[4] Such algorithms involve some amount of educated guessing about which received symbols are in error or, alternatively, about which received symbols are correct. The guesses are informed by quality metrics associated with the received symbols. To illustrate why it is absolutely essential to use such soft-symbol information in these algorithms it helps to consider what would happen if we tried to use completely random guesses, ignoring any available soft-symbol information.

As a specific example, consider a received JT65 word with 23 correct symbols and 40 errors. We do not know which symbols are in error. Suppose that the decoder randomly selects $s = 40$ symbols for erasure, leaving 23 unerased symbols. According to Eq. (3), the BM decoder can successfully decode this word as long as e, the number of errors present in the 23 unerased symbols, is 5 or less. The number of errors captured in the set of 40 erased symbols must therefore be at least 35.

The probability of selecting some particular number of incorrect symbols in a randomly selected subset of received symbols is governed by the hypergeometric probability distribution. Let us define N as the number of symbols from which erasures will be selected, X as the number of incorrect symbols in the set of N symbols, and x as the number of errors in the symbols actually erased. In an ensemble of many received words X and x will be random variables, but for this example we will assume that X is known and that only x is random. The conditional probability mass function for x with stated values of N, X, and s may be written as

$$P(x=\varepsilon \mid N,X,s)=\frac{\binom{X}{\varepsilon}\binom{N-X}{s-\varepsilon}}{\binom{N}{s}} \qquad (4)$$

where $\binom{n}{k}=\frac{n!}{k!(n-k)!}$ is the binomial coefficient. The binomial coefficient can be calculated using the function **nchoosek(n,k)** in the numerical computing language *GNU Octave*, or with one of many free online calculators. The hypergeometric probability mass function defined in Eq. (4) is available in *GNU Octave* as function **hygepdf(x,N,X,s)**. The cumulative probability that at least ε errors are captured in a subset of s erased symbols selected from a group of N symbols containing X errors is

$$P(x\ge\varepsilon|N,X,s)=\sum_{j=\varepsilon}^{s}P(x=j\mid N,X,s)\,. \qquad (5)$$

Example 1:

Suppose a received word contains $X = 40$ incorrect symbols. In an attempt to decode using an errors-and-erasures decoder, $s = 40$ symbols are randomly selected for erasure from the full set of $N = n = 63$ symbols. The probability that $x = 35$ of the erased symbols are actually incorrect is then

$$P(x=35)=\frac{\binom{40}{35}\binom{63-40}{40-35}}{\binom{63}{40}}\simeq 2.4\times10^{-7}\quad.$$

Similarly, the probability that $x = 36$ of the erased symbols are incorrect is $P(x=36)\simeq 8.6\times10^{-9}$. Since the probability of erasing 36 errors is so much smaller than that for erasing 35 errors, we may safely conclude that the probability of randomly choosing an erasure vector that can decode the received word is approximately $P(x=35)\simeq 2.4\times10^{-7}$. The odds of producing a valid codeword on the first try are very poor, about 1 in 4 million.

Example 2:

How might we best choose the number of symbols to erase, in order to maximize the probability of successful decoding? By exhaustive search over all possible values up to $s = 51$, it turns out that for $X = 40$ the best strategy is to erase $s = 45$ symbols. According to Eq. (3), with $s = 45$ and $d = 52$ then e must be 3 or less. Decoding will be assured if the set of erased symbols contains at least $40-3=37$ errors. With $N = 63$, $X = 40$, and $s = 45$, the probability of successful decode in a single try is $P(x\ge37)\simeq1.9\times10^{-6}$. This probability is about 8 times higher than the probability of success when only 40 symbols were erased. Nevertheless, the odds of successfully decoding on the first try are still only about 1 in 500,000.

Example 3:

Examples 1 and 2 show that a random strategy for selecting symbols to erase is unlikely to be successful unless we are prepared to wait a long time for an answer. So let's modify the strategy to tip the odds in our favor. Let the received word contain $X = 40$ incorrect symbols, as before, but suppose we know that 10 received symbols are significantly more reliable than the other 53. We might therefore protect the 10 most reliable symbols and select erasures from the smaller set of $N = 53$ less reliable ones. If $s = 45$ symbols are chosen randomly for erasure in this way, it is still necessary for the erased symbols to include at least 37 errors, as in Example 2. However, the probabilities are now much more favorable: with $N = 53$, $X = 40$, and $s = 45$, Eq. (5) yields $P(x\ge37)\simeq0.016$. Even better odds are obtained by choosing $s = 47$, which requires $x\ge38$. With $N = 53$, $X = 40$, and $s = 47$, $P(x\ge38)\simeq0.027$. The odds for producing a codeword on the first try are now about 1 in 38. A few hundred independently randomized tries would be enough to all-but-guarantee production of a valid codeword by the BM decoder.

4 — The Franke-Taylor Decoding Algorithm

Example 3 shows how statistical information about symbol quality should make it possible to decode received frames having a large number of errors. In practice the number of errors in the received word is unknown, so our algorithm simply assigns a high erasure probability to low-quality symbols and relatively low probability to high-quality symbols. As illustrated by Example 3, a good choice of erasure probabilities can increase the chance of producing a codeword by many orders of magnitude. Once erasure probabilities have been assigned to each of the 63 received symbols, the FT algorithm uses a random number generator to decide whether or not to erase each symbol, according to its assigned erasure probability. The list of erased symbols is then submitted to the BM decoder, which produces either a codeword or a flag indicating failure to decode.

The process of selecting the list of symbols to erase and calling the BM decoder comprises one cycle of the FT algorithm. The next cycle proceeds with a new selection of erased symbols. At this stage we must treat any codeword obtained by errors-and-erasures decoding as no more than a *candidate*. Our next task is to find a metric that can reliably select one of many proffered candidates as the codeword that was actually transmitted.

The FT algorithm uses quality indices made available by a noncoherent 64-FSK demodulator (see the sidebar *JT65 Message Processing*). The demodulator computes binned power spectra for each signaling interval; the result is a two-dimensional array $S(i,j)$, where the frequency index i assumes values 0 to 63 and the symbol index j has values 1 to 63. The most likely value for each symbol is taken as the frequency bin with largest signal-plus-noise power over all values of i. The fractions of total power in the two bins containing the largest and second-largest powers, denoted respectively by p_1 and p_2, are computed for each symbol and passed from demodulator to decoder as soft-symbol information. The FT decoder derives two metrics from p_1 and p_2, namely p_1-rank (the rank {1,2, …,63} of the symbol's fractional power $p_{1,j}$ in a sorted list of p_1 values) and the ratio p_2/p_1. High ranking symbols have larger signal-to-noise ratio than those with lower rank. When p_2/p_1 is close to 1, the most likely symbol value is only slightly more reliable than the second most likely one.

We use 3-bit quantization of the metrics p_1-rank and p_2/p_1 to index the entries in an 8×8 table of symbol error probabilities. The probabilities were derived empirically from a large data set of received words that were successfully decoded. The table provides an estimate of the *a priori* probability of symbol error based on the metrics p_1-rank and p_2/p_1. This table is a key element of the algorithm, as it determines which symbols are effectively protected from erasure. The *a priori* symbol error probabilities are close to 1 for low-quality symbols and close to 0 for high-quality symbols. Recall from Examples 2 and 3 that candidate codewords are produced with higher probability when the number of erased symbols is larger than the number of incorrect symbols. Correspondingly, the FT algorithm works best when the probability of erasing a symbol is somewhat larger than the probability that the symbol is incorrect. For the JT65 code we found empirically that good decoding performance is obtained when the symbol erasure probability is about 1.3 times the symbol error probability.

The FT algorithm tries successively to decode the received word using independent educated guesses to select symbols for erasure. For each iteration a stochastic erasure vector is generated based on the symbol erasure probabilities. The erasure vector is sent to the BM decoder along with the full set of 63 hard-decision symbol values. When the BM decoder finds a candidate codeword it is assigned a quality metric d_s, the *soft distance* between the received word and the codeword:

$$d_s = \sum_{j=1}^{n} \alpha_j \left(1 + p_{1,j}\right). \tag{6}$$

Here $\alpha_j = 0$ if received symbol j is the same as the corresponding symbol in the codeword, $\alpha_j = 1$ if the received symbol and codeword symbol are different, and $p_{1,j}$ is the fractional power associated with received symbol j. Think of the soft distance as made up of two terms: the first is the Hamming distance between the received word and the codeword, and the second ensures that if two candidate codewords have the same Hamming distance from the received word, a smaller soft distance will be assigned to the one where differences occur in symbols of lower estimated reliability.

In practice we find that d_s can reliably identify the correct codeword if the signal-to-noise ratio for individual symbols is greater than about 4 in linear power units. We also find that significantly weaker signals can be decoded by using soft-symbol information beyond that contained in p_1 and p_2. To this end we define an additional metric u, the average signal-plus-noise power in all received symbols according to a candidate codeword's symbol values:

$$u = \frac{1}{n} \sum_{j=1}^{n} S\left(c_j, j\right). \tag{7}$$

Here the c_j's are the symbol values for the candidate codeword being tested.

The correct JT65 codeword produces a value for u equal to the average of n = 63 bins containing both signal and noise power. Incorrect codewords have at most $k - 1 = 11$ such bins and at least $n - k + 1 = 52$ bins containing noise only. Thus, if the spectral array $S(i, j)$ has been normalized so that the average value of the noise-only bins is unity, u for the correct codeword has expectation value (average over many random realizations) given by

$$\bar{u}_c = 1 + y, \tag{8}$$

where y is the signal-to-noise ratio in linear power units. If we assume Gaussian statistics and a large number of trials, the standard deviation of measured values of u is

$$\sigma_c = \left(\frac{1 + 2y}{n}\right)^{1/2}. \tag{9}$$

In contrast, the expected value and standard deviation of the u-metric for an incorrect codeword (randomly selected from a population of all "worst case" codewords, *i.e.*, those with k - 1 symbols identical to corresponding ones in the correct word) are given by

$$\bar{u}_i = 1 + \left(\frac{k-1}{n}\right) y, \tag{10}$$

$$\sigma_i = \frac{1}{n}\left[n + 2y\left(k - 1\right)\right]^{1/2}, \tag{11}$$

where the subscript i is an abbreviation for "incorrect".

If u is evaluated for a large number of distinct candidate codewords, one of which is correct, we should expect the largest value u_1 to be drawn from a population with statistics described by $\bar{u}_c$ and σ_c. If no tested codeword is correct, u_1 is likely to come from the $\left(\bar{u}_i, \sigma_i\right)$ population and to be several standard deviations above the mean. In either case the second-largest value, u_2, will likely come from the $\left(\bar{u}_i, \sigma_i\right)$ population, again several standard deviations above the mean. If the signal-to-noise ratio y is too small for decoding to be possible or the correct codeword is never presented as a candidate, the ratio $r = u_2/u_1$ will likely be close to 1. On the other hand, correctly identified codewords will produce u_1 significantly larger than u_2 and thus smaller values of r. We therefore apply a ratio threshold test, say $r < R_1$, to identify codewords with high probability of being correct. As described in Section 6, we use simulations to set an empirical acceptance threshold R_1 that maximizes the probability of correct decodes while ensuring a low rate of false positives.

As with all decoding algorithms that generate a list of possible codewords, a stopping criterion is necessary. FT accepts a codeword unconditionally if the Hamming distance X and soft distance d_s obey specified criteria $X < X_0$ and $d_s < D_0$. Secondary acceptance criteria $d_s < D_1$ and $r < R_1$ are used to validate additional codewords that fail the first test. A timeout is used to limit execution time if no acceptable codeword is found in a reasonable number of trials, T. Today's personal computers are fast enough that T can be set as large as 10^5, or even higher. Pseudo-code for the FT algorithm is presented in an accompanying box, **Algorithm 1**.

Inspiration for the FT decoding algorithm came from a number of sources.[4,5,6] After developing this algorithm, we became aware that our approach is conceptually similar to a stochastic, erasures-only list decoding algorithm described in another reference.[7] That algorithm is applied to higher-rate Reed-Solomon codes on a symmetric channel using binary phase-shift keying (BPSK). Our 64-ary input channel with 64-FSK modulation required us to develop unique methods for assigning erasure probabilities and for defining acceptance criteria to select the best codeword from the list of tested candidates.

5 — Hinted Decoding

The FT algorithm is completely general. With equal sensitivity it can recover any one of the $2^{72} \approx 4.7 \times 10^{21}$ different messages that can be transmitted with the JT65 protocol. In some circumstances it's easy to imagine a *much* smaller list of messages (say, a few thousand messages or less) that would be among the most likely ones to be received. One such favorable situation exists when making short Amateur Radio contacts that exchange minimal information including callsigns, signal reports, perhaps Maidenhead locators, and acknowledgments. On the EME path or a VHF or UHF band with limited geographical coverage, the most common received messages frequently originate from callsigns that have been decoded before. Saving a list of previously decoded callsigns and associated locators makes it easy to generate a list of hypothetical messages and their corresponding codewords at very little computational expense. The resulting candidate codewords can be tested in almost the same way as those generated by the probabilistic method described in Section 4. We call this approach "hinted decoding;" it is sometimes referred to as the *Deep Search* algorithm. In certain limited situations it can provide enhanced sensitivity for the principal task of any decoder, namely to determine precisely what message was sent.

For hinted decoding we again invoke a ratio threshold test, but in this case we use it to answer a more limited question. Over the full list of messages considered likely, we want to know whether a suitable metric can distinguish with confidence between the one correct codeword and all others in the generated list — or, alternatively, to determine that the correct codeword is *not* contained in the list. We again find that the most effective metric involves a comparison of u_1 and u_2, the largest and second-largest values of total signal-plus-noise power among all the tested codewords. The criterion for comparison is chosen empirically to maximize the number of correct decodes while ensuring that false decodes are rare. Because tested candidate codewords are drawn from a list typically no longer than a few thousand entries, rather

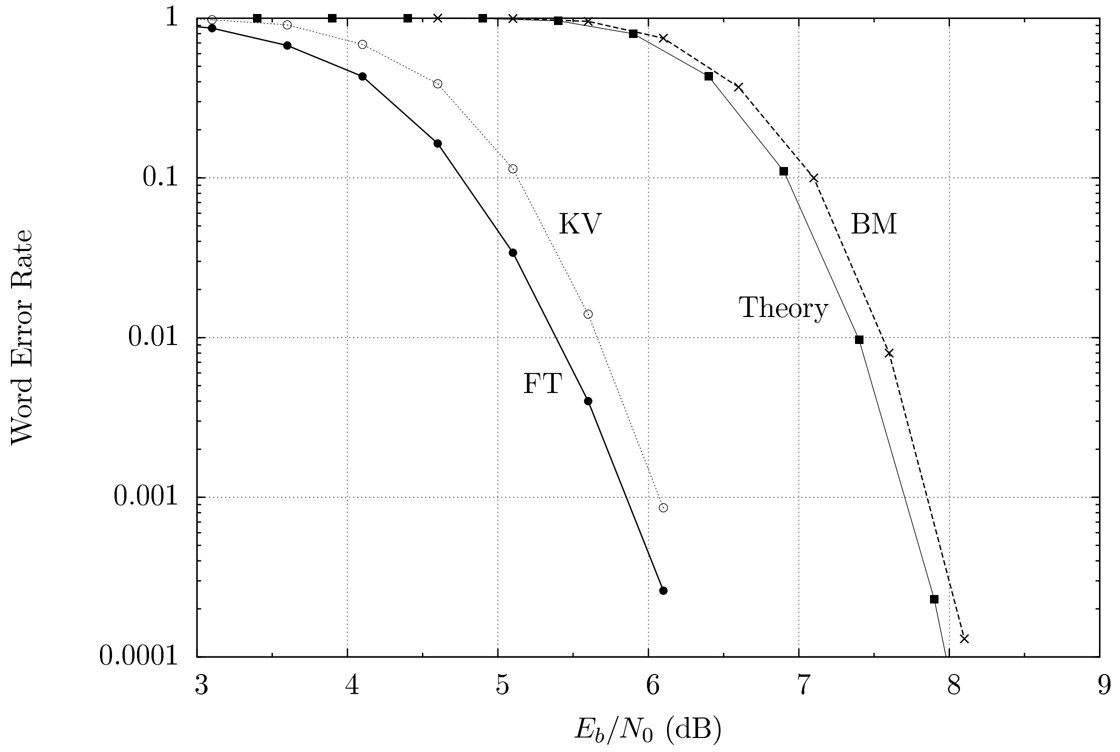


Figure 1 — Word error rates as a function of E_b / N_0, the signal-to-noise ratio per information bit. The curve labeled 'Theory' shows a theoretical prediction for the hard-decision BM decoder. Remaining curves represent simulation results on an AWGN channel for the BM, KV, and FT decoders. The KV algorithm was executed with complexity coefficient $\lambda = 15$, the most aggressive setting historically used in the *WSJT* programs. The FT algorithm used timeout setting $T = 10^5$.

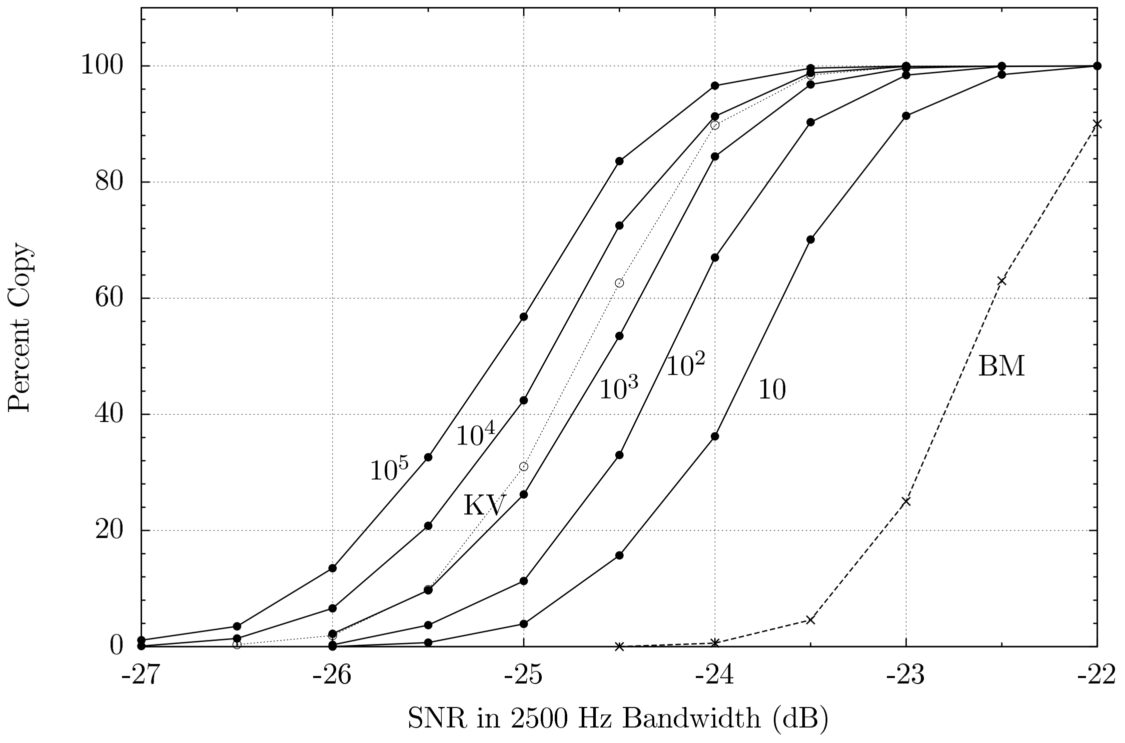


Figure 2 — Percent of JT65 messages copied as a function of SNR_{2500}, assuming additive white Gaussian noise and no fading. Numbers adjacent to curves specify values of timeout parameter for the FT decoder. Open circles and dotted line show results for the KV decoder with complexity coefficient $\lambda = 15$. Results for the BM algorithm are plotted with crosses and dashed line.

than 2^{72}, the limit can be more relaxed than that used in the FT algorithm. Thus, for the limited subset of messages suggested by previous experience to be likely, hinted decodes can be obtained at lower signal levels than required for the full universe of 2^{72} possible messages. Pseudo-code for the hinted-decoding algorithm is presented as **Algorithm 2**.

6 — Decoder Performance Evaluation

Comparisons of decoding performance are usually presented in the professional literature as plots of word error rate versus E_b / N_0, the ratio of the energy collected per information bit to the one-sided noise power spectral density. For weak-signal Amateur Radio work, performance is more usefully presented as the probability of successfully decoding a received word plotted against SNR_{2500}, the signal-to-noise ratio in a 2500 Hz reference bandwidth. The relationship between E_b / N_0 and SNR_{2500} is described in Appendix A. Examples of both types of plot are included in the following discussion, where we describe simulations carried out to compare performance of the FT algorithm and hinted decoding with other algorithms and with theoretical expectations. We have also used simulations to establish suitable default values for the acceptance parameters X_0, D_0, D_1, R_1, and R_2.

6.1 — Simulated results on the AWGN channel

Results of simulations using the BM, KV, and FT decoding algorithms on the JT65 code are presented in terms of word error rate versus E_b / N_0 in Figure 1. For these tests we generated at least 1000 signals at each signal-to-noise ratio, assuming the additive white Gaussian noise (AWGN) channel, and we processed the data using each algorithm. For word error rates less than 0.1 it was necessary to process 10,000 or even 100,000 simulated signals in order to capture enough errors to make the measurements statistically meaningful. As a test of the fidelity of our numerical simulations, Figure 1 also shows results calculated from theoretical probability distributions for comparison with the BM results. The simulated BM results agree with theory to within about 0.1 dB. The differences are caused by small errors in the estimates of time and frequency offset of the received signal in the simulated data. Such "sync losses" are not accounted for in the idealized theoretical results.

As expected, on the AWGN channel the soft-decision algorithms FT and KV are about 2 dB better than the hard-decision BM algorithm. In addition, FT has an edge over KV that increases from about 0.2 dB at higher SNRs to nearly 0.5 dB at low SNR. With timeout parameter $T = 10^5$ execution time for FT is longer than that for the KV algorithm, but still small enough to be fully practical on today's home computers.

Error-free transmission is important in commercial applications, so plots like Figure 1 are often extended downward to error rates of 10^{-6} or even less. The circumstances for minimal Amateur Radio contacts are very different, however. Decoding failure rates of order 0.1 or higher may be perfectly acceptable: they simply require repeat transmissions. In such circumstances the essential information is more usefully presented in a plot showing the percentage of transmissions copied correctly as a function of signal-to-noise ratio. Figure 2 shows the FT and KV results from Figure 1 in this format, along with additional FT results for $T = 10^4$, 10^3, 10^2 and 10. It's easy to see that the FT decoder produces more decodes than KV when T is greater than about 3000. As already noted in connection with Figure 1, FT with $T = 10^5$ has approximately 0.5 dB gain over KV at low SNR. It also provides very significant gains over the hard-decision BM decoder, even when limited to very small T.

Parameter T in the FT algorithm is the maximum number of symbol-erasure trials allowed for a particular attempt at decoding a received word. Most successful decodes take only a small fraction of the maximum allowed number of trials. Figure 3 shows the number of stochastic erasure trials required to find the correct codeword plotted as a function of X, the number of hard-decision errors in the received word. This test run used 1000 simulated transmissions at $SNR_{2500} = -24$ dB, just slightly above the decoding threshold, with timeout parameter $T = 10^5$. No points are shown for $X \le 25$ because all such words are successfully decoded by a single run of the errors-only BM algorithm. Figure 3 shows that the FT algorithm decodes received words with as many as $X = 43$ symbol errors. It also shows that the average number of trials increases with the number of errors in the received word. The variability of decoding time also increases dramatically with the number of errors in the received word. These results provide insight into the mean and variance of execution time for the FT algorithm, since execution time is roughly proportional to the number of required erasure trials.

6.2 — Simulated results for Rayleigh fading and hinted decoding

Figure 4 presents the results of simulations for signal-to-noise ratios ranging from -18 to -30 dB, again using 1000 simulated signals for each plotted point. We include three curves for each decoding algorithm: one for the AWGN channel and no fading, and two more for simulated Doppler spreads of 0.2 and 1.0 Hz. These simulated Doppler spreads are comparable to those encountered on HF ionospheric paths and also for EME at VHF and the lower UHF bands. For comparison we note that the JT65 symbol rate is about 2.7 Hz. It is interesting to note that while Rayleigh fading severely degrades the success rate of the BM decoder, the penalties are much smaller with both FT and Deep Search (DS) decoding. Simulated Doppler spreads of 0.2 Hz actually increased the FT decoding rate slightly at SNRs close to the decoding threshold, presumably because with the low-rate JT65 code, signal peaks provide the information needed for good copy.

7 — On-the-air Experience

The JT65 protocol has proven remarkably versatile. Today the mode is used by thousands of amateurs around the world, communicating over terrestrial paths on the MF and HF bands and over terrestrial as well as EME paths from 50 MHz through 10 GHz. Three *submodes* are in use, together accommodating a wide range of Doppler spreads and potential instrumental instabilities. All three submodes transmit the 63 data symbols interspersed with 63 synchronization symbols at keying rate $11025/4096 = 2.69$ baud. Submode JT65A uses tone spacing equal to the symbol rate; its total occupied bandwidth is $66 \times 2.69 = 177.6$ Hz. Submodes B and C have tone spacings and occupied bandwidths 2 and 4 times larger. In practice JT65A is generally used at 50 MHz and below, JT65B on 144 through 432 MHz, and JT65C at 1296 MHz and above.

Figure 5 shows portions of the main window and spectrogram displays from program *WSJT-X*, illustrating replies to a CQ from K1JT on 144.118 MHz using submode JT65B on the EME path. Speckled vertical lines on the waterfall at 1494 and 1515 Hz are the synchronizing tones of signals from DL7UAE and SP6GWB. Other visible speckles (barely above the noise) up to about 1870 Hz are some of the data tones from these two stations. Two lines of decoded text show that the estimated average signal strengths were $SNR_{2500} = -23$ and -24 dB, respectively, just one or two dB above decoding threshold for the FT decoder. Note that the two signals overlap throughout more than 90% of their occupied bandwidths, yet both are decoded cleanly and without errors. Such behavior is typical of the JT65 protocol.

As another example, Figure 6 shows activity in submode JT65A during a single minute on the 20 m amateur band. At this time the band was crowded with overlapping signals. With care you can count at least 19

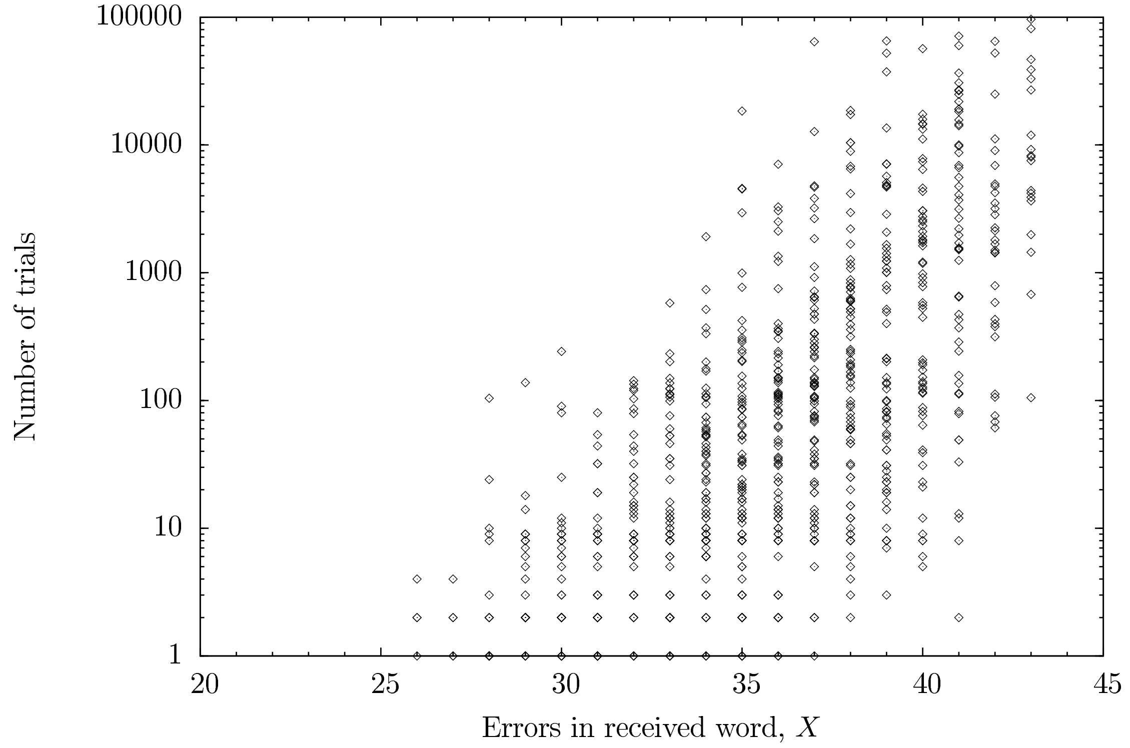


Figure 3 — Number of trials needed to decode a received word *vs.* Hamming distance *X* between received word and decoded codeword. We used 1000 simulated transmissions on an AWGN channel with no fading. The signal-to-noise ratio was $SNR_{2500} = -24$ dB, or $E_b / N_0 = 5.1$ dB.

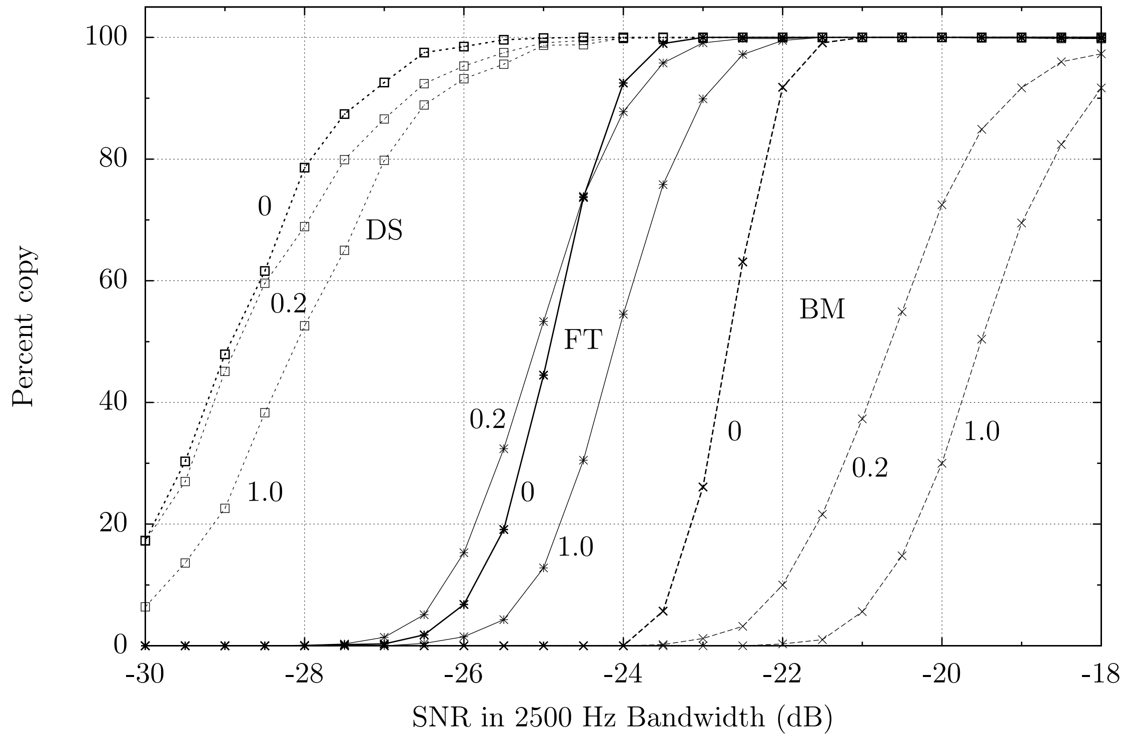


Figure 4 — Percentage of JT65 messages successfully decoded as a function of SNR_{2500}. Results are shown for the hard-decision Berlekamp-Massey (BM) and soft-decision Franke-Taylor (FT) decoding algorithms. Curves labeled 'DS' correspond to the hinted-decode (Deep Search) algorithm with a codeword list of length $L = 5850$. Numbers adjacent to the curves are simulated Doppler spreads in Hz. In the current version of *WSJT-X* the performance of the DS algorithm is limited by synchronization failures when *SNR* is less than about 28 dB.

distinct synchronizing tones (the speckled vertical lines in the Figure), and can see as many as four signals overlapping in some places. After signal processing to demodulate the signals and produce soft-symbol data for the FT decoder, program *WSJT-X* extracts and decodes 21 error-free messages from this recorded data segment. This result is achieved with a relatively small timeout parameter, $T = 1000$. For these results the decoder uses two successive sweeps over the spectrum. The strongest signals (12 in this example) are sequentially decoded and subtracted from the raw data after the first pass. Another 9 signals are decoded in the second pass. For comparison, the hard-decision BM decoder decodes only 12 messages from this recording (9 in the first pass and 3 more in a second).

Our implementation of the FT decoder, written in a combination of FORTRAN and C, is freely available as open-source code.[8] For the Berlekamp-Massey part of the algorithm we use routines written by Phil Karn, KA9Q, modified slightly so that the Reed-Solomon syndromes are computed only once in our most time-consuming loop (Steps 2 through 8, **Algorithm 1**).[9] The FT algorithm has become an integral part of programs *WSJT*, *MAP65*, and *WSJT-X*. Improvement in sensitivity over the Kötter-Vardy decoder is small, only a few tenths of a dB, but especially on the EME path such small advantages are sometimes very important. Perhaps even more essential, programs in the *WSJT* family are now entirely open source. We no longer need to use the patented KV algorithm or the specially licensed executable program `kvasd[.exe]`.

8 — Acknowledgments

We thank Charlie Suckling, G3WDG; Bill Somerville, G4WJS; Casey Smith, KD9DSW; Edson Pereira, PY2SDR; Leif Asbrink, SM5BSZ; Rex Moncur, VK7MO and Roger Rehr, W3SZ, for helpful comments on an earlier version of this paper.

A — Appendix: Signal to Noise Ratios

The signal to noise ratio in a bandwidth B, that is at least as large as the bandwidth occupied by the signal is:

$$SNR_B = \frac{P_s}{N_0 B} \tag{12}$$

where P_s is the average signal power (W), N_0 is one-sided noise power spectral density (W/Hz), and B is the bandwidth in Hz. In Amateur Radio applications, digital modes are often compared based on the SNR defined in a 2.5 kHz reference bandwidth, SNR_{2500}.

In the professional literature, decoder performance is characterized in terms of E_b / N_0, the ratio of the energy collected per information bit, E_b, to the one-sided noise power spectral density, N_0. Denote the duration of a channel symbol by τ_s (for JT65, $\tau_s = 0.3715$ s). JT65 signals have constant envelope, so the average signal power is related to the energy per symbol, E_s, by

$$P_s = E_s / \tau_s. \tag{13}$$

The total energy in a received JT65 message consisting of n = 63 channel symbols is $63E_s$. The energy collected for each of the 72 bits of information conveyed by the message is then

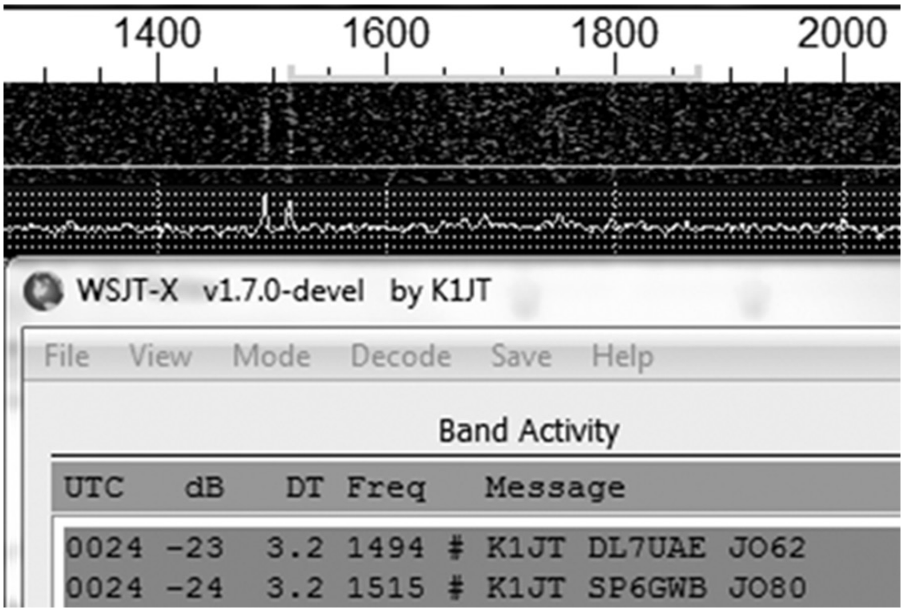
$$E_b = \frac{63 E_s}{72} = 0.875 E_s. \tag{14}$$

Using equations (12) – (14), SNR_{2500} can be written in terms of E_b / N_0:

$$SNR_{2500} = 1.23 \times 10^{-3} \frac{E_b}{N_0} \tag{15}$$

If all quantities are expressed in dB, then:

$$\begin{aligned} SNR_{2500} &= \left(E_b/N_0\right)_{\text{dB}} - 29.1\ \text{dB} \\ &= \left(E_s/N_0\right)_{\text{dB}} - 29.7\ \text{dB} \end{aligned} \tag{16}$$



```
UTC   dB   DT Freq   Message
0024 -23  3.2 1494 # K1JT DL7UAE JO62
0024 -24  3.2 1515 # K1JT SP6GWB JO80
```

Figure 5 — Examples of JT65B EME signals recorded at K1JT. Numbers above the spectrogram are audio frequencies in Hz, and the spectrogram's vertical span is one minute of time. The horizontal green bar on the frequency axis indicates the bandwidth occupied by the second decoded signal, a reply from SP6GWB. See text for additional details.

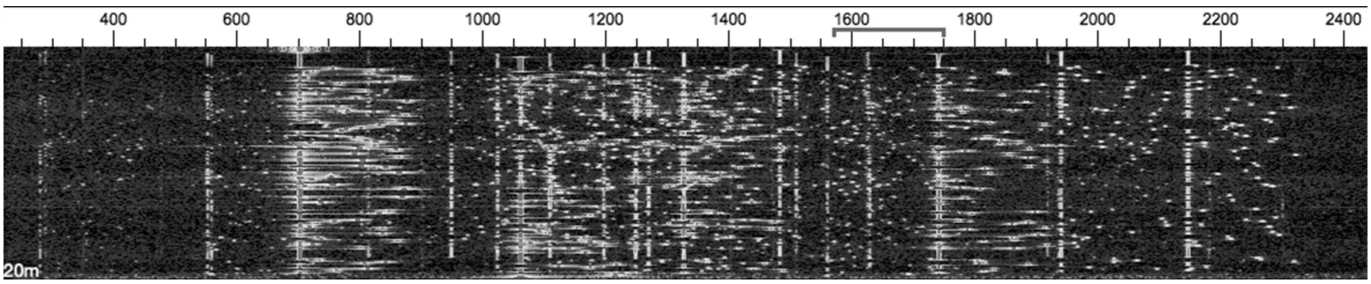


Figure 6 — Spectrogram from *WSJT-X* showing one minute of data collected under crowded band conditions on the 20 m band. Numbers on the scale are frequencies (in Hz) above 14.076 MHz.

JT65 Message Processing

1. User *A* enters or selects message consistent with formatting rules of JT65.
2. Transmitting software at *A*: compress message into 12 six-bit symbols, then add 51 six-bit parity symbols.
3. Intersperse 63 synchronizing symbols among the 63 information-carrying symbols.
4. Start transmission 1 s into a UTC minute. Transmit each symbol value at a distinct frequency.
5. Signal propagates from *A* to *B*, arriving much weaker and corrupted by noise, fading, and Doppler spread.
6. Receiving software at *B*: remove impulsive noise; detect synchronizing signal, measure its frequency and time offset.
7. Shift spectrum to put sync tone at zero frequency, correcting for any measured drift.
8. Compute binned power spectra $S(i, j)$ for all information symbols. (Index *i* runs over 64 possible symbol values, index *j* over 63 symbol numbers.)
9. Remove any possible spurs (signal appearing at same *i* for all *j*).
10. Apply **Algorithm 1**, the FT algorithm.
11. Optional: if FT decoding was unsuccessful apply **Algorithm 2**, hinted decoding.
12. Display decoded message for User *B*.

Algorithm 1

Pseudo-code for the FT algorithm.

```
1. For each received symbol, define the erasure probability as 1.3 times the a priori symbol-error probability determined from soft-symbol information $\{p_1 - \text{rank}, p_2/p_1\}$.
2. Make independent stochastic decisions about whether to erase each symbol by using the symbol's erasure probability, allowing a maximum of 51 erasures.
3. Attempt errors-and-erasures decoding using the BM algorithm and the set of erasures determined in step 2. If the BM decoder produces a candidate codeword, go to step 5.
4. If BM decoding was not successful, go to step 2.
5. Calculate the hard-decision Hamming distance $X$ between the candidate codeword and the received symbols, along with the corresponding soft distance $d_s$ and the quality metric $u$.
6. If $u$ is the largest one encountered so far, preserve any previous value of $u_1$ by setting $u_2 = u_1$. Then set $u_1 = u$, $d_1 = d_s$, $X_1 = X$, and save the codeword.
7. If $X_1 < X_0$ and $d_1 < D_0$, go to step 11.
8. If the number of trials is less than the timeout limit $T$, go to step 2.
9. If $d_1 < D_1$ and $r = u_2/u_1 < R_1$, go to step 11.
10. Otherwise, declare decoding failure and exit.
11. An acceptable codeword has been found. Declare a successful decode and return the saved codeword.
```

Algorithm 2

Pseudo-code for hinted decoding

```
1. Generate a list of $L$ codewords considered likely to be received. Set a pointer to the start of this list.
2. Fetch the next candidate codeword and calculate its metric $u$.
3. If $u$ is the largest metric encountered so far, preserve any previous value of $u_1$ by setting $u_2 = u_1$. Then set $u_1 = u$ and save the codeword.
4. If the number of tested codewords is less than $L$, go to step 2.
5. If $r = u_2 / u_1 < R_2$, go to step 7.
6. Otherwise, declare decoding failure and exit.
7. An acceptable codeword has been found. Declare a successful result and return the codeword and the value $q = 100(u_1 - bu_2)$ as a confidence indicator. (By default we use the value $b = 1.12$ for submode JT65A.)
```

Glossary of Specialized Terms

Alphabet	A sequence of possible symbol values used for signaling. JT65 uses a 64-character alphabet, values in the range 0 to 63.
Block code	An error-correcting code that treats data in blocks of fixed size.
Codeword	For the JT65 code, a vector of 63 symbol values each in the range 0 to 63.
Deterministic algorithm	A series of computational steps that for the same input always produces the same output.
Erasure	A received symbol may be "erased" when confidence in its value is so low that it is unlikely to provide useful information.
Hamming distance	The Hamming distance between two codewords, or between a received word and a codeword, is equal to the number of symbol positions in which they differ.
Hard decision	Received symbols are assigned definite values by the demodulator.
Received word	A vector of symbol values, possibly accompanied by soft information on individual reliabilities.
Soft decision	Received symbols are assigned tentative values (most probable, second most probable, etc.) and quality indicators.
Soft distance	The soft distance between a received word and a codeword is a measure of how greatly they differ, taking into account available soft information on symbol values.
Source encoding	Compression of a message to use a minimum number or bits. JT65 source-encodes all messages to 72 bits.
Stochastic algorithm	An algorithm involving chance or probability in determining the series of computational steps to be taken.
Symbol	The information carried in one signaling interval, usually an integral number of bits. JT65 uses 6-bit symbols.

Steve Franke, K9AN, holds an Amateur Extra class license. He was first licensed in 1971 and has previously held call signs WN9IIQ and WB9IIQ. An early and abiding fascination with radio science led to his current position as Professor of Electrical and Computer Engineering at the University of Illinois in Urbana-Champaign. Steve is a member of ARRL and a Fellow of the IEEE.

Joe Taylor was first licensed as KN2ITP in 1954, and has since held call signs K2ITP, WA1LXQ, W1HFV, VK2BJX and K1JT. He was Professor of Astronomy at the University of Massachusetts from 1969 to 1981 and since then Professor of Physics at Princeton University, serving there also as Dean of the Faculty for six years. He was awarded the Nobel Prize in Physics in 1993 for discovery of the first orbiting pulsar, leading to observations that established the existence of gravitational waves. After retirement he has been busy developing and enhancing digital protocols for weak-signal communication by Amateur Radio, including JT65 and WSPR. He chases DX from 160 meters through the microwave bands.

References

[1]J. Taylor, K1JT, "The JT65 Communications Protocol", *QEX*, Sep-Oct 2005, pp 3-12. Available also available at **physics.princeton.edu/pulsar/K1JT/JT65.pdf**.

[2]R. Kötter and A. Vardy, "Algebraic soft-decision decoding of Reed-Solomon codes", *IEEE Transactions on Information Theory*, Vol. 49, pp 2809-2825, 2003.

[3]WSJT Home Page: **www.physics.princeton.edu/pulsar/K1JT/**.

[4]Shu Lin and Daniel J. Costello, **Error Control Coding, 2nd Edition**, Pearson-Prentice Hall, 2004.

[5]Camille Leroux, Saied Hemati, Shie Mannor, Warren J. Gross, "Stochastic Chase Decoding of Reed-Solomon Codes," *IEEE Communications Letters*, Vol. 14, No. 9, pp. 863-865, 2010.

[6]Soo-Woong Lee and B. V. K. Vijaya Kumar, "Soft-Decision Decoding of Reed-Solomon Codes Using Successive Error-and-Erasure Decoding," *IEEE "GLOBECOM"* Proceedings, 2008.

[7] Chang-Ming Lee and Yu T. Su, "Stochastic Erasure-Only List Decoding Algorithms for Reed-Solomon Codes," *IEEE Signal Processing Letters*, Vol. 16, pp 691-694, 2009.

[8]Source code for all programs in the WSJT project is stored in a Subversion repository at Sourceforge: **https://sourceforge.net/projects/wsjt/**.

[9]Errors-and erasures decoder for the Berlekamp-Massey algorithm written by Phil Karn, KA9Q, **www.ka9q.net/code/fec/**.