An Autoencoder for Transmission of Vocoder Features over Radio Channels

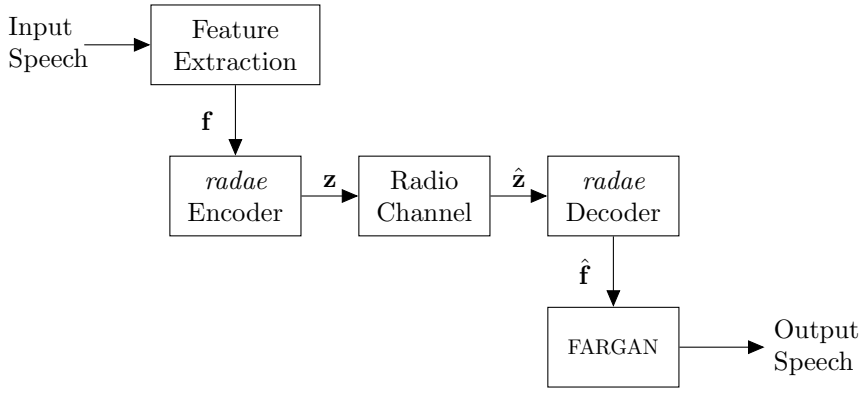
July 4, 2024

1 Introduction

This report presents an autoencoder derived from RDOVAE [4] to send speech over radio channels. Our goal is to determine if reasonable speech quality can be obtained over a channel of bandwidth $B < 3000$ Hz and SNR around 0dB, roughly the lower limit of Single Side Band (SSB) - a popular power and bandwidth efficient form of speech communication.

This document describes the Radio Autoencoder *radae* system, discusses channel simulation and noise calibration, and describes how we combined ML with classical DSP to create a practical proof of concept system that can send speech over real world VHF and HF radio channels. Finally, we compare the capabilities of the Radio Autoencoder to existing systems for speech communication over HF and VHF/UHF radio. TODO - update for later sections.

Figure 1: Radio Autoencoder Concept



The encoder (Figure 1) takes as input a typical set of vocoder features (short term spectrum, pitch, voicing), then applies time based prediction and transforms to produce a set of parameters that can be sent over a channel. This

is similar to vocoders using classical DSP, except Machine Learning (ML) allows us to learn non-linear transforms and prediction, which tend to be more powerful.

In conventional digital speech systems, after the transformation/prediction stage we then quantise to a low bit rate, then use Forward Error Correction (FEC) and modems to send the bits over a channel. However our work takes a novel twist – we train the autoencoder to generate PSK symbols that we send over the channel. It effectively combines quantisation, channel coding, and modulation. The symbols from the autoencoder tend to cluster around +/-1 like BPSK but are continuously valued, so can be considered discrete time, continuously valued PSK.

The encoder and decoder are trained together as an autoencoder with the loss function $L(\mathbf{f}, \hat{\mathbf{f}})$ applied to the vocoder features. We employ the FARGAN vocoder [5] for speech analysis and synthesis, however the concept is applicable to any neural and even classical vocoder with a similar feature set.

Given a vector of vocoder features $\mathbf{f}$, we use an encoder to map them to a dimension d latent vector $\mathbf{z}$ where d is even. Unlike digital modulation, each element z_i of $\mathbf{z}$ is continuously valued and not constrained to a discrete set of points. For bandwidth efficient transmission over the channel the elements of $\mathbf{z}$ are mapped to $d/2$ complex symbols $\mathbf{q}$. Compared to classical digital modulation, the elements of $\mathbf{z}$ can be considered BPSK symbols (continuously valued, analog bits), and the elements of $\mathbf{q}$ analog QPSK symbols.

2 Simulation of AWGN Channels

The autoencoder output $\mathbf{z}$ is updated every $T_z = 1/R_z$ seconds, giving a BPSK symbol rate of:

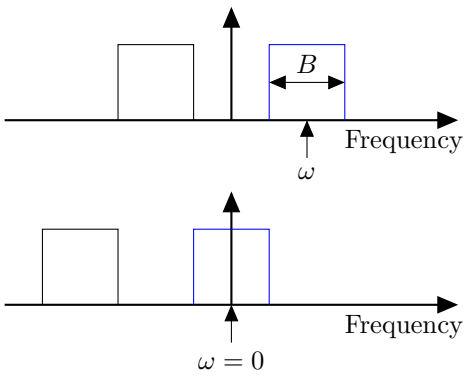
$$R_b = d/T_z \tag{1}$$

For example with $T_z = 0.04, d = 80, R_b = 2000$ symbols/s. The QPSK symbol rate is given by:

$$R_q = \frac{d}{2T_z} \tag{2}$$

For example with $T_z = 0.04, d = 80, R_q = 1000$ symbols/s.

Figure 2: Real sampled off-air signal. We are interested in the blue bandpass interval of bandwidth B, which is single sided and hence complex valued. After shifting to baseband, it's power is unchanged, and it remains complex valued.



We wish to simulate an AWGN channel with a user-defined E_b/N_0, where E_b is the energy of each BPSK symbol, and N_0 is the noise power per unit bandwidth. Consider a real valued signal sampled off air (Figure 2). We will follow convention and define signal and noise power in the "single sided" bandpass interval of the frequency spectrum with bandwidth B centred on ω. As the interval is single sided, we must use complex valued quantities to represent it.

We wish to simulate a bandpass AWGN channel at baseband ($\omega = 0$). This implies a frequency shift of the complex valued signal, but the signal remains complex valued and it's power is unchanged. The negative frequency component on the LHS of Figure 2 is redundant and after frequency shifting can be removed by filtering.

Note that even at baseband we must use complex valued quantities for the signal and noise to represent a bandpass signal of bandwidth B. As a counter-example, given a fixed sample rate B and noise power N, a real valued noise sequence can only represent a bandwidth of $B/2$ which results in doubling the noise density $N_0 = N/(B/2) = 2N_0$ compared to a complex valued noise sequence with the same power.

The energy of each PSK symbol E_q is the signal power S divided by the PSK symbol rate R_q. The noise per unit bandwidth is the total noise power N divided by the bandwidth B of the system. If we are simulating at one sample

per PSK symbol, $B = R_q$:

$$\begin{aligned} \frac{E_q}{N_0} &= \frac{S/R_q}{N/R_q} \\ &= \frac{S}{N} \\ &= \frac{A_q^2}{\sigma^2} \end{aligned} \tag{3}$$

where A_q is the amplitude of each PSK symbol and $\sigma^2 = N$ is the variance of the complex valued noise (mean noise energy per sample). Given a set point E_q/N_0:

$$\sigma = \frac{A_q}{\sqrt{E_q/N_0}} \tag{4}$$

If we use a BPSK system model, we note the amplitude of each BPSK symbol $A_b = A_q/\sqrt{2}$, and the energy of each BPSK symbol $E_b = E_q/2$. Subsituting into 4:

$$\sigma = \frac{A_b}{\sqrt{E_b/N_0}} \tag{5}$$

Which allows us to simulate using a set point E_b/N_0. The complex noise sample r_i can be generated as:

$$r_i = \frac{\sigma}{\sqrt{2}}(\mathcal{N}_{2i}(0,1) + j\mathcal{N}_{2i+1}(0,1)) \tag{6}$$

where $\mathcal{N}_i(0,1)$ is the $i-th$ sample of a unit variance, zero mean, real Gaussian noise source. Note the noise power is split evenly between the real and imaginary arms. Our symbols passing through an AWGN channel can be simulated at complex baseband as:

$$\begin{aligned} \hat{z}_i &= z_i + r_i \\ \hat{q}_i &= q_i + r_i \end{aligned} \tag{7}$$

If the noise is zero mean, we can estimate σ^2 over K noise samples r_i as:

$$\sigma^2 = E[|r_i|^2] = \frac{1}{K}\sum_{i=0}^{K-1} |r_i|^2 \tag{8}$$

2.1 SNR Measurement

In order to compare with other methods of speech communication that have varying bandwidths B, it is useful to formulate expressions for estimating SNR from the BPSK and QPSK symbols. The Signal to Noise ratio (SNR) is given by:

$$\begin{aligned} \frac{S}{N} &= \frac{E_b R_b}{N_0 B} \\ &= \frac{E_q R_q}{N_0 B} \end{aligned} \tag{9}$$

A noise bandwidth B needs to be selected; common choices are $B = R_b$, in which case $S/N = E_b/N_0$; for HF radio $B = 3000$ Hz to compare with existing analog and digital voice waveforms; or $B = 1$ to obtain a normalised C/N_0 carrier power to noise density ratio - useful for comparing waveforms with different bandwidths.

At one sample per symbol, the power, the mean energy of each QPSK symbol over a window of K samples is given by:

$$E_q = E[|q_i|^2] = \frac{1}{K}\sum_{i=0}^{K-1}|q_i|^2 \tag{10}$$

Note the variance function should not be used to calculate E_q, as we cannot guarantee q_i is zero mean. As each QPSK symbol contains 2 BPSK symbols, the energy is split evenly:

$$E_b = E_q/2 \tag{11}$$

For example if the symbol amplitude is $A = 1, E_b = A^2 = 1$, then $E_q = 1+1 = 2$.

For transmission over multipath channels using OFDM we arrange the QPSK symbols as N_c parallel carriers, each running at a symbol rate of $R_s = R_q/N_c$ symbols/s, where R_s is chosen based on delay spread considerations. Typical values for HF modems are $N_c = 20$ and $R_s = 50$ Hz. However the OFDM carriers are arranged such that the total symbol rate over the channel remains constant. So for a given signal power E_q and E_b remain constant (Table 1).

Waveform	N_c	R_s	R_q	R_b	E_q	E_b
Single Carrier BPSK	1	-	-	2000	-	$S/2000$
Single Carrier QPSK	1	-	1000	2000	$S/1000$	$S/2000$
OFDM QPSK	20	50	1000	2000	$S/1000$	$S/2000$

Table 1: E_b and E_q examples for single and multi-carrier OFDM waveforms for constant carrier power S

2.2 Calibration and Testing

In order to evaluate the ML system early in the development process it is important to ensure the noise is correctly calibrated. The expressions above can be used to check the noise injection process:

1. Set a target E_b/N_0 for the simulation run, and calculate σ using (4).
2. Establish the equivalent target SNR from (9) evaluated using the target E_b/N_0.
3. After the simulation run measure $E_q = E[|q_i|^2]$ over a sample of transmitted symbols. Note that in general $E_q \neq 2$ as the encoder outputs continuous values.
4. Calculate measured SNR using (9) and compare.

The calibration of the noise injection can be checked by replacing the encoder output z_i with discrete PSK symbols to create a digital modem, then measuring the BER at E_b/N_0 points. The theoretical BER over an AWGN channel is:

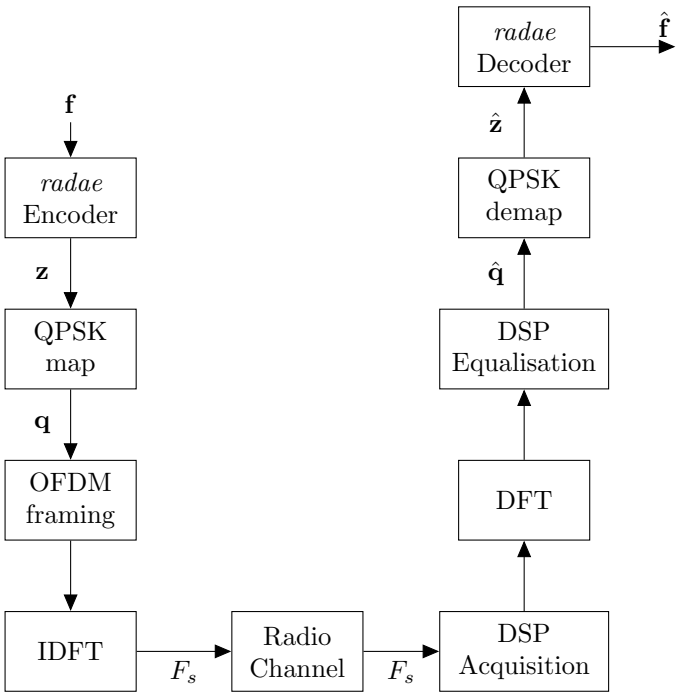
$$BER = 0.5erfc(\sqrt{E_b/N_0}) \tag{12}$$

For a multipath channel:

$$BER = 0.5\left(1 - \sqrt{\frac{E_b/N_0}{E_b/N_0 + 1}}\right) \tag{13}$$

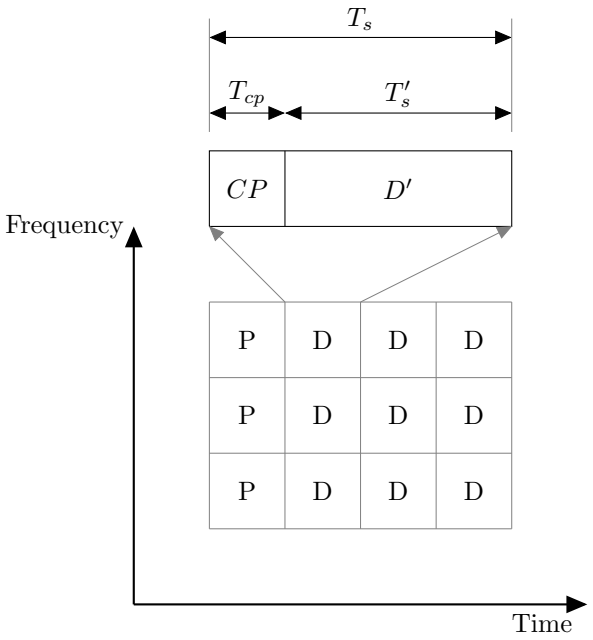
3 Proof of Concept Radio Frequency Tests

Figure 3: ML combined with OFDM and classical DSP synchronisation. Transmitter (Tx) is on the LHS, at RHS is the Receiver (Rx). The sample rate over the channel is $F_s = 8000$ Hz.



Impressive results have been obtained from symbol rate simulations of an OFDM modem, which assumed ideal synchronisation. We would like to verify these results using real radio signals in Over The Cable (OTC) and Over the

Figure 4: OFDM modem frame, P denotes pilot symbol, D payload data symbols. This example has a modem frame of $N_s = 4$ symbols, and $N_c = 3$ carriers. Each symbol D or P is comprised of a T_{cp} second Cyclic Prefix and T_s' second symbol D'.



Air (OTA) tests. This requires building up a rate F_s system, and synchronisation subsystems. For our proof of concept system, the choice was made to use OFDM to handle HF multipath channel, and classical DSP pilot symbol based synchronisation, although we acknowledge potential for ML based synchronisation in future iterations. The combined ML and OFDM/DSP system is illustrated in Figure 3.

The goal is to compare speech quality to SSB at $E_b/N_0 = 0dB$ (approx -3dB SNR in a 3000Hz BW), and work through any issues that prevent the system working over real radio channels. PAPR optimisation will be ignored for the first iteration, as our initial goal is to verify the low E_b/N_0 results suggested by the symbol rate simulations.

The general design of the OFDM frame in Figure 4. QPSK symbols are mapped to a matrix of parallel carriers at a relatively low symbol rate in order to successfully pass through the target HF multipath channel. Pilot symbols are periodically inserted into each OFDM carrier. At the receiver the pilots are used to estimate the time varying phase of the channel (equalisation), and for

initial acquisition (coarse frequency, and frame sync) of the received signal. The disadvantage of pilot based schemes is they consume carrier power that would otherwise be available for the data symbols, and require the symbol rate and hence overall RF bandwidth to be increased to maintain the payload data rate. After the IDFT stage a cyclic prefix is inserted to accommodate inter symbol interference. The cyclic prefix also consumes carrier power and requires an increase in RF bandwidth. The OFDM waveform details, including overheads, are explained in detail in the waveform design spreadsheet [3]. The *Candidate 2* waveform has a RF bandwidth of approximately 1500 Hz, with 500Hz due to synchronisation overhead.

A pilot based sync system was built in PyTorch, and is used for coarse and fine timing, phase and amplitude equalisation. Unlike classical PSK, the ML network is likely to be sensitive to amplitude variations. Phase equalisation also allows small frequency offsets (± 2 Hz) to be handled, sufficient for tests with short samples.

Several phase estimators were prototyped, and evaluated using BER measurements. Maintaining low loss synchronisation at low E_b/N_0 is challenging. Using per-carrier phase estimation makes the system less dependant on fine timing accuracy and gives us the ability to handle multipath, but has higher loss than algorithms that consider all carriers at the same time. As further work a lower latent dimension d and higher E_b/N_0, would allocate more power to pilots, and result in less carriers.

Figure 5 and 6 plots the simulated performance of the OFDM pilot based synchronisation system on AWGN and multipath channels. The *genie* curve is the baseline rate F_s OFDM system with ideal sync, and matches the theoretical BPSK BER curve. Using this baseline system to send ML symbols, we obtain intelligible speech at $E_b/N_0 = -6$, which corresponds to BER=0.24 in a digital modem. We can use the BER=0.24 line to estimate the synchronisation loss at this operating point, which for the *mean6* and *LS* algorithms under realistic conditions is around 2 to 2.4 dB on AWGN, and 2dB on the multipath channel. The *LS* works better on fast fading multipath channels, so was chosen for the HF OTA trials.

Table 2 summarises the sync losses. These are comparable to classical DSP OFDM data modems. Combined with L_p, we estimate a total sync loss of 4dB for this first pass of the ML system combined with classical pilot based synchronisation.

Symbol	Loss (dB)	Description
L_p	0.97	Pilot symbol overhead
L_{cp}	0.97	Cyclic prefix overhead
L_{eq}	2.00	*LS* equalisation (multipath)

Table 2: *Candidate 2* OFDM waveform design sync losses, around 4dB total.

Figure 7: Over The Cable (OTC) VHF test. The *radae* transmitter output is upsampled and shifted to 144.5 MHz using a HackRF. The RF receive power and hence SNR is set by a step attenuator before being received by a RTL-SDR and *gqrx* SDR application. Wave files recorded off air by *gqrx* are presented to the *radae* receiver.

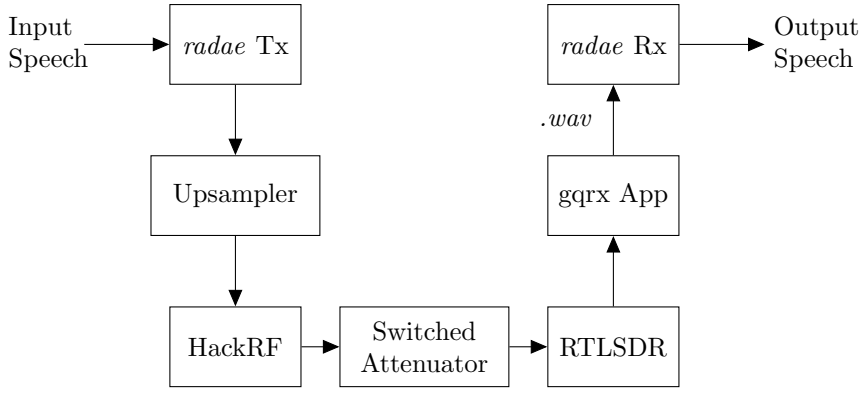
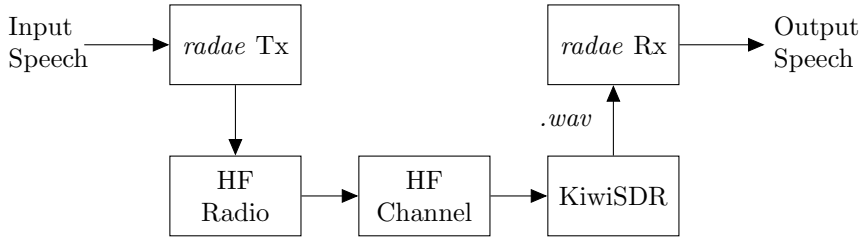


Figure 8: Over The Cable (OTA) HF test. The *radae* transmitter output is fed to the USB sound interface of a COTS HF Radio where it is shifted to HF, amplified, and transmitted over various real world HF channels to a remote KiwiSDR receiver. Wave files recorded off air by the KiwiSDR are presented to the *radae* receiver.



3.1 March 2024 Results

In March 2024, the system was tested over a VHF OTC AWGN path and OTA on a variety of real world Australian HF multipath paths. Figure 7 and Figure 7 describe the experimental configurations. Wave files containing a sine wave tone, compressed SSB, and the *radae* signal were sent over the same channel at the same time. The speech quality obtained was consistent with the simulations and competitive with SSB sent over the same channel at the same time.

Take aways:

1. We have combined ML vocoder, ML autoencoder and classical DSP OFDM to build a system capable of sending speech over radio channels. It is robust to AWGN and multipath channel impairments. Our initial simulation, VHF OTC, and HF OTA tests suggest performance competitive with the analog SSB at the same C/N_0.

2. The total "sync losses" due to pilot, cyclic prefix overheads and non-ideal equalisation are around 4dB. It is conceivable these can be reduced using ML rather than classical DSP.

3. The ML algorithms run at a 40ms frame rate (with some DSP at the OFDM 20ms symbol rate), resulting in modest CPU requirements. The PyTorch simulation code runs several times faster than real time in inference mode.

4. Simulation and initial HF OTA results show surprisingly good performance on multipath channels where the period of the fading (100's of ms) is large compared to the 40ms analysis window of the ML code. Classical DSP would require a 1000-2000 ms interleaver (introducing a algorithmic delay of the same order) for similar robustness to fading on these channels.

5. Unlike regular PSK, we require magnitude estimation and correction, due to the limited dynamic range of ML systems and the wide dynamic range of radio signals.

6. The *Candidate 2* OFDM frame design [3] contains three latent vectors z, so introduces an algorithmic delay of 120ms, due to OFDM framing considerations. This is tolerable in a PTT radio system, but ideally should be reduced.

7. Substituting classical PSK digital symbols and measuring BER is a very useful way to test sync subsystems, and to allows us to verify E_b/N_0 using BER measurements during development.

8. Much of our development effort to date has concerned with the "is this too good to be true" question; to date significantly more effort was put into noise calibration, synchronisation algorithms and testing than the actual ML.

Further work:

1. Try a low dimension latent vector, e.g. $d = 40$, and see if similar speech quality can be obtained at 3dB higher E_b/N_0. This would result in lower sync losses and RF bandwidth for the same channel SNR or C/N_0, as the E_b/N_0 of the pilots would be increased. Does the encoder output still resemble BPSK, or is it training to a higher order constellation?

2. There is significant synchronisation loss from the classical DSP OFDM waveform and associated sync algorithms. Attempt to use the ML network

to perform frequency, phase and amplitude equalisation, with or without passing the pilots to the decoder. An initial attempt (model07) without pilots showed some ability to correct phase, frequency, and magnitude offsets, but resulted in some performance degradation. However this may be acceptable if comparable to the pilot based sync losses. Some pilot or unique word injection may still be required to perform coarse and fine timing estimation using classical DSP running at the sample rate.

3. We may be able to improve on algorithmic delay using ML sync techniques that are less dependant on traditional OFDM framing considerations.

4. Some form or interleaving, or a long time window for the ML network, may improve performance.

5. Work to improve the current classical DSP sync, e.g. from Figure 5 a feedback loop to track out frequency offsets is worth 1 dB.

6. Include PAPR optimisation and rate F_s multipath channels in the training.

7. To better model SSB, locate a better analog compressor, a 3rd party reference implementation in software form would be useful.

8. Definition of lower limit link closure for this use case, for example "CQ CQ, and callsign, enough to produce a QSO report".

9. For a practical implementation, coarse amplitude and timing need to be updated regularly. As we are testing short samples, a single block estimate is used at present.

4 Comparison with Other Speech Waveforms

We wish to compare the potential of our radio autoencoder with existing waveforms used for speech transmission over radio channels. This section assumes the *radae* system can send intelligible speech at an E_b/N_0 of -6dB, with a PAPR of 1dB, both results have been demonstrated in simulation (but not at the time of writing together over real world radio channels). We include a 3dB synchronisation overhead, anticipating modest improvements over the 4dB obtained with our proof of concept system described above.

We start with the assumption that we have a transmitter of C watts, and an AWGN channel with a spectral noise density of N_0 watts/Hz. As the speech waveforms being considered vary in bandwidth we will choose C/N_0 as the SNR metric.

The C/N_0 (in dBHz) at the demodulator input of a terrestrial radio receiver is given by:

$$\frac{C}{N_0} = P_{tx} - PAPR - L_{path} - NF + 174 \tag{14}$$

where P_{tx} is the peak output power of the transmitter, $PAPR$ is the Peak to Average Power Ratio of the waveform, L_{path} is the path loss, NF is the noise

figure of the receiver. For example consider a 400 MHz FM hand held radio over a 1km urban (non line of site) path. The radio has a 1W (30 dBm) power output, $L_{path} = 120$ dB, with noise dominated by ambient EMI such that $NF = 10$ dB. $C/N_0 = 30 - 0 - 120 - 10 + 174 = 74$ dBHz, sufficient for good quality speech (Table 6).

Note that C/N_0 at the demodulator is a function of the waveform PAPR. With all other link properties (e.g. peak output power, noise figure, path loss) being equal, a high PAPR reduces the C/N_0 available at the receiver. We effectively "back off" the transmitter power from the maximum P by the PAPR. We assume the PA is capable of sustaining P watts indefinitely, i.e. it is only the waveform choice that lowers the average power. As PAPR varies by waveform and has an impact on the C/N_0 available the receiver, it should be included in any metric for comparison of waveforms. We define the peak power to spectral noise density ratio P/N_0 as:

$$P/N_0 = C/N_0 + PAPR \tag{15}$$

A waveform that delivers intelligible speech at a low P/N_0 is the target. A low PAPR waveform has other desirable properties, such as greater PA efficiency, longer battery life, and the use of low cost semiconductors in the radio power amplifier electronics.

Waveform	Threshold
Single Sideband	0dB SNR in 3000Hz noise BW, 2400Hz audio bandwidth, Tx speech compressor with 6dB PAPR
Frequency Modulation	-120 dBm quoted for many NBFM radios, 54dB above -174dBm/Hz noise floor
FreeDV 700D	10% PER threshold at -2dB SNR in 3000Hz noise BW
Radio Autoencoder	Intelligible speech at $E_b/N_0 = -6$ dB, $R_b =$ 2000 symbols/s, 3dB sync overhead, a PAPR of 1dB

Table 3: Thresholds for speech link closure for each waveform. The link is considered closed when the speech is barely intelligible to a trained listener.

TODO: include 1st gen VHF/UHF digital voice - I think they go down to -123 dB (5%) BER, but speech quality is sub FM.

Waveform	Threshold C/N_0 calculations (dBHz)
Single Sideband	$0 + 10log_{10}(3000) = 35$
Frequency Modulation	$-120 + 174 = 54$
FreeDV 700D	$-2 + 10log_{10}(3000) = 33$
Radio Autoencoder	$-6 + 10log_{10}(2000) + 3 = 30$

Table 4: Threshold C/N_0 calculations.

Waveform	Abbr	RF BW	PAPR	C/N_0	P/N_0	Δ
Single Sideband	SSB	2400	6	35	41	-10
Frequency Modulation	NBFM	16000	0	54	54	-23
FreeDV 700D	700D	1100	4	33	37	-6
Radio Autoencoder	radAE	1400	1	30	31	0

Table 5: Comparison of link closure by waveform over AWGN channels. A lower P/N_0 is better.

Waveform	Audio BW	C/N_0	P/N_0	Δ
Radio Autoencoder	8000	36	37	0
Frequency Modulation	3000	64	64	-27
Single Sideband	3000	55	61	-24

Table 6: Comparison of good quality "arm chair copy" by waveform over AWGN channels. They are ranked in terms of maximum achievable speech quality. FreeDV 700D has been omitted because of its low speech quality even at high SNR. Even at 20dB SNR there is noticeable noise in received SSB, although DSP based noise reduction may help. Only the radio autoencoder delivers wideband (8000 Hz) speech.

5 CPU and Memory Requirements

Module	MMACS	Memory (kbyte)
Feature Extraction	-	-
RADAE Encoder	80	1000
OFDM Tx	-	-
Totals	80	1000

Table 7: Estimates of RADAE transmitter (Tx) CPU and Memory Requirements.

Module	MMACS	Memory (kbyte)
FARGAN vocoder	300/175	800
RADAE Decoder	80	1000
OFDM Rx	-	-
Totals	380/255	1800

Table 8: Estimates of RADAE receiver (Rx) CPU and Memory Requirements.

Tables 7 and 8 contain estimates of the CPU and memory resources. Typical PTT radio uses-case are half duplex, so the Tx and Rx would not be required to run at the same time. The receiver dominates, due to the FARGAN vocoder.

MMACS is the number 8-bit multiply and accumulates per second divided by 1E6. The memory is mainly ML weights which are read-only. However memory access needs to be fast to maintain the MMAC rate - each MMAC is assumed to include any required loads. A non-SIMD floating point implementation (where multiple and accumulate are separate operations) would take twice the number of operations when measured in MFLOPS, and four times the memory.

The FARGAN author recommends a 64-bit ARM with NEON extensions as a minimum machine (Cortex-M is likely too small). The current FARGAN vocoder requires 300 MMACs, a slightly lower quality version is available that uses 175 MMACS. The classical DSP Feature Extraction uses comparatively small amounts of CPU and memory, but currently requires floating point hardware.

The RADAE ML CPU hasn't been measured accurately but runs in real time in Python, and is less than the FARGAN decoder (i.e. the FARGAN vocoder dominates the CPU). For an initial estimate we have approximated the RADAE CPU as 25% of FARGAN. The RADAE encoder and decoder require approximately the same CPU, and similar memory (ML weights). It may be possible to reduce the number of weights and CPU, as the optimal size of the RADAE network has not been studied.

The classical DSP OFDM processing requires less resources but hasn't been accurately measured at this time. Comparable modems at similar symbol rates run on a fraction of Cortex-M micro-controllers. Unlike the ML modules, it currently requires floating point hardware. For a first approximation we assume OFDM resource requirements are small compared to the ML modules.

6 Training and Testing Low PAPR Models

This section documents the low PAPR RADAE models developed in May 2024, and improvements from the April 2024 stored file OTA test campaign.

6.1 Mixed Rate Model

We wish to train a model to minimise PAPR over multipath channels. Our previous models (e.g. *model05*) were trained in the frequency domain at rate R_s with the bottleneck applied to the magnitude of real valued symbols z_i. Due to the neat properties of OFDM, we could ignore issues like phase/amplitude equalisation and ISI during training, and just apply magnitude only fading to each PSK symbol. After training, classical DSP techniques were then wrapped around the core ML encoder to develop a practical, rate F_s digital voice system.

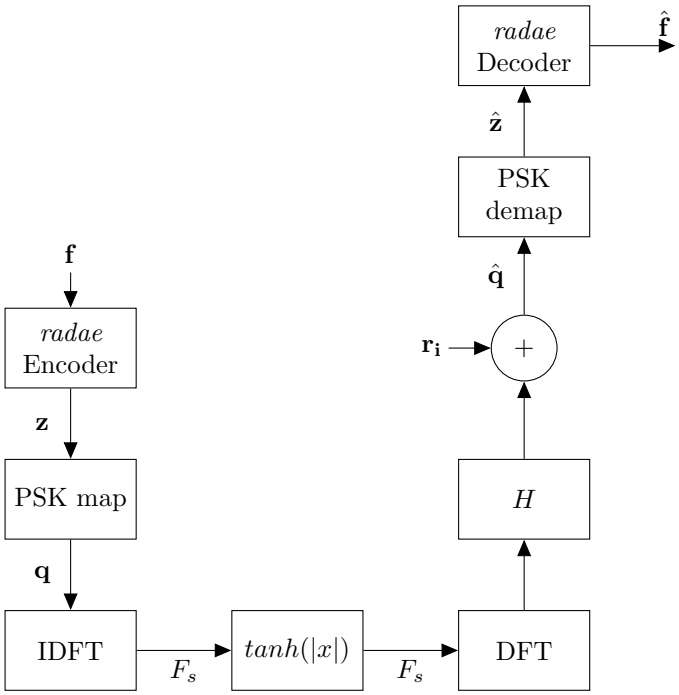
We reasoned that a time domain bottleneck applied to the magnitiude of the complex time domain signal would encourage the network to maximise the RMS power (and hence minimse the PAPR) given the channel noise and peak power constraint of the bottleneck.

To train a model to optimise PAPR we need to apply a bottleneck in the time domain, and simultaneously apply a multipath channel model. Applying

the multipath model in the time domain introduces phase rotations and ISI, which would then require equalisation and removal inside the training loop. While possible this would require significant additional CPU and algorithmic complexity using classical DSP or ML based equalisation, and require training at the higher rate F_s sample rate ($F_s/R_s = 160$ in our use case).

An alternative, "mixed rate" training model was therefore devised (Figure 9). The transmit PSK symbols are IDFTed to the time domain, the bottleneck applied, then immediately DFTed back to the frequency domain. The multipath model is then applied on the frequency domain PSK symbols. Training using this model resulted in signals with a PAPR of around 0.5 dB and S/N performance close to the reference non-PAPR optimised 1-D bottleneck *model05*.

Figure 9: Mixed Rate training for PAPR minimisation and multipath channels. The magnitude of the complex rate F_s time domain signal $x(n)$ passes through a bottleneck simulating the PA compression, before being converted back to the frequency domain where the multipath model H and is applied and noise added using efficient rate R_s OFDM techniques.



After training, pilot symbols and a cyclic prefix are concatenated with each "modem frame" of ML symbols carrying the payload voice information (Figure 10). Classical PAPR compression techniques (gain and compression provided by the bottleneck) are applied to set the pilot signal PAPR to around 3dB. As

the pilot sequence is much shorter than the payload sequence in each "modem frame", the average PAPR remains beneath 1dB. Given pilot symbols with a magntitude of $|p_i| = \sqrt{2}$, the pilot gain is given by:

$$p_g = \frac{M 10^{P_b/20}}{\sqrt{N_c}} \tag{16}$$

where P_b is the backoff in dB, which was set to $P_b = -2$ dB by experment.

6.2 Calibrated Noise

To train for low PAPR we constrain the magnitude of the complex rate F_s time domain transmit signal $|x(n)| \leq 1$. We wish to train for a range of SNRs so require an expression to compute σ for a given target E_b/N_0.

Consider a hybrid ML-DSP system where we constrain $|x(n)|$ in the time domain, but perform all other processing at rate R_s in the frequency domain using classical DSP OFDM techniques. After the bottleneck is applied to $x(n)$, we DFT to transform back to the frequency domain, apply a magnitude only multipath model, than add AWGN noise to simulate the received frequency domain symbols $\hat{q}$.

Given this model, we wish to inject calibrated AWGN noise. Consider one time domain OFDM carrier $x(n)$ with magnitude B. We take the length M DFT to determine the magnitude A_q of the corresponding frequency domain PSK symbol:

$$\begin{aligned} x(n) &= Be^{j(2\pi nk/M+\theta)} \\ X(k) &= \sum_{n=0}^{M-1} x(n)e^{-j(2\pi nk/M)} \\ &= BMe^{j\theta} \\ |X(k)| &= A_q = BM \end{aligned} \tag{17}$$

We have N_c carriers, and the bottleneck constrains the total time domain power of all carriers to be approximately 1:

$$\begin{aligned} N_c B^2 &= 1 \\ B &= \frac{1}{\sqrt{N_c}} \end{aligned} \tag{18}$$

Using (4) we can find an expression for σ given E_b/N_0:

$$\begin{aligned} \sigma &= \frac{A_b}{\sqrt{E_b/N_0}} \\ \sigma &= \frac{A_q}{\sqrt{2E_b/N_0}} \\ \sigma &= \frac{M}{\sqrt{2N_c E_b/N_0}} \end{aligned} \tag{19}$$

The factor of $\sqrt{2}$ on the denominator accounts for A_q being the amplitude of a PSK symbol, whereas (4) is configured for BPSK symbols. For example if $A_b = 1$, $A_q = \sqrt{2}$.

These expressions were used as a baseline for applying noise to PAPR optimised models. It was found the measured Eq/N_0 were around 2.5dB higher than the targets. We observed that the constellation of the trained PSK signals resemble a 2D Gaussian distribution rather than constant magnitude symbols as assumed in (18). Our theory is that in practice the symbol magnitudes A_q vary across carriers and time and are best described as a model-dependant distribution (probably Rayleigh) rather than a constant. Nevertheless the expressions in this section are close enough for training where we sweep E_q/N_0 over a range of SNRs. For evaluation at inference time, we inject noise based on the expressions above as an approximate "set point", then measure and report the actual the SNR statistics for the simulation.

6.3 Simulated Results

The *evaluate.sh* script was used to genereate samples for informal listening, and to compare to SSB (Table 9). E_b/N_0 controls the channel C/No but the relationship varies due to model-model variations. In practice we measure C/No and PAPR for each run of the RADAE simulation, then determine the amount of noise to inject for the SSB sample to obtain the target P/N_0. When applying a multipath model, the transmit power is normalised over the simulation run such that the average carrier power C is the same before and after the multipath model is applied.

The earlier, *model05* RADAE waveform has a PAPR of 9-10dB. Based on our experience of classical DSP OFDM waveforms we reason that by applying classical DSP PAPR reduction techniques this could be reduced to 5dB, with a 1dB loss in performance, leading to an effective PAPR of 6dB, similar to the SSB signal. We have not tested this configuration, but reason that *model17* will have a 6dB advantage over *model05* for the same transmitter peak power.

Waveform	Channel	E_b/N_0	PAPR	C/N_0	P/N_0	SNR
Radio Autoencoder	AWGN	16	0.80	48.21	49.01	13.44
Single Sideband	AWGN	16	6.66	42.35	49.01	7.58
Radio Autoencoder	MPP	16	0.80	48.21	49.01	13.44
Single Sideband	MPP	16	6.66	42.35	49.01	7.58

Table 9: Comparison of PAPR optimised RADAE *model17* to SSB. The low PAPR RADAE waveform shows an improvement of around 6dB in receiver SNR over SSB, although the relationship between speech quality and SNR is of course different for both receivers.

7 Timing Algorithm

Coarse and fine timing is determined by maximising the correlation function at some time offset t samples from [1]:

$$D_t = \boldsymbol{r}^H \boldsymbol{p} \tag{20}$$

where $\boldsymbol{p}$ is a M sample vector of pilot samples:

$$p(n) = \sum_{c=1}^{Nc} P_c e^{j\omega_c n} \quad n = 0, .., M \tag{21}$$

At the transmitter we take $p(n)$ and insert a Cyclic Prefix to extend it to $N_{cp}+M$ samples. The received pilot sequence is denoted $r(n)$. We assume we have a "coarse timing" estimate and have located the received pilot samples to within one symbol. The vector $\boldsymbol{r}$ is the M sample vector extracted from $r(n)$ at some fine time offset t:

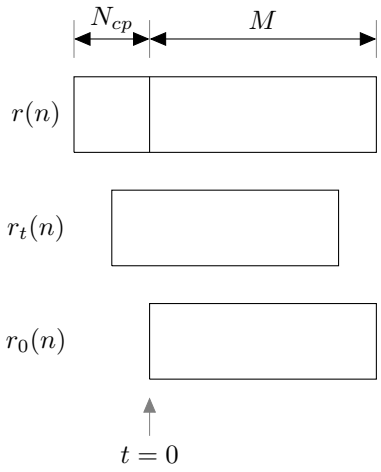
$$\begin{aligned} r(n) &= gp(n) \quad n = -(N_{cp}-1), ..., 0, .., M-1 \\ r_t(n) &= gp(n+t) \quad n = 0, .., M-1, \quad -(N_{cp}-1) \leq t \leq 0 \end{aligned} \tag{22}$$

where g is a complex scalar gain. The correlation as a function of the time offset t is given by:

$$\begin{aligned} D_t &= \sum_{n=0}^{M-1} \sum_{c=1}^{Nc} r_t^*(n) p(n) \\ &= g \sum_{n=0}^{M-1} \sum_{c=1}^{Nc} |P_c|^2 e^{-j\omega_c t} \end{aligned} \tag{23}$$

Now $r_t(n)$ contains a complete M sample cycle of $p(n)$ for any $-(N_{cp}-1) \leq t \leq 0$. We could reason that a peak in $|D_t|$ may occur for any $-(N_{cp}-1) \leq t \leq 0$; however by inspecting (23) we can see that for $t \neq 0$ the $e^{-j\omega_c t}$ phase shift results in the terms of the summation pointing in different "directions" on the complex plane. To some extent these will add destuctively resulting in a reduction in $|D_t|$. Therefore the maxima of $|D_t|$ will tend to be located at $t = 0$ (Figure 11).

Figure 11: The maxima in D_t will tend to be located in the last M samples of the received sequence $r(n)$.



Now consider the pilot sequence passing through a two path multipath channel simulation, where *path1* is the fastest path to our receiver and *path2* the slowest. We set a timing reference at $t = 0$ samples, the start of the received symbol for *path1*. The time difference (delay spread) between the two paths is N_{ds} samples. The timing estimates that maximises D_t will tend to point to the last M samples of either *path1* or *path2*, depending on which path best maximises D_t. The two paths are time varying, but can be considered stationary over the timing estimator analysis window. We denote the two possible timing estimates $t_1 = N_{cp}$ and $t_2 = N_{ds} + N_{cp}$. Given t_1 or t_2, we wish to determine a M sample window of samples starting at t_{sam} for sampling the received OFDM symbol that minimises Inter symbol Interference (ISI).

Figure 12: Multipath channel, our received symbol is the sum of *path1* and *path2*. Shaded regions are ISI from past and future symbols. We must select M samples such that we avoid any ISI.

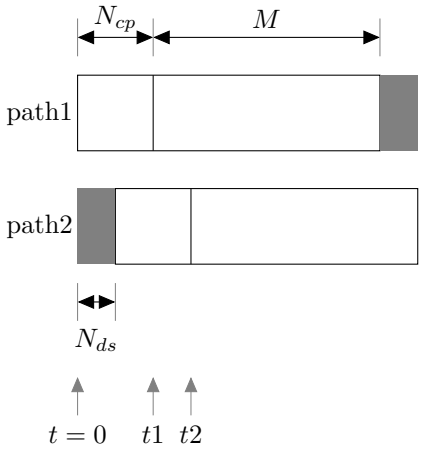


Figure 12 illustrates the two paths and various constraints on selecting a suitable timing offset given we have either t_1 or t_2 from maximising D_t. To demodulate a symbol without ISI, we must select M samples with the starting sample t_{sam} such that:

$$N_{ds} \leq t_{sam} \leq N_{cp} \tag{24}$$

One possible solution for the sampling instant is:

$$t_{sam} = t - N_{ds} \tag{25}$$

where N_{ds} is an estimate of the worst case delay spread. Evaluating for $t = t_1$:

$$\begin{aligned} t_{sam} &= t_1 - N_{ds} \\ &= N_{cp} - N_{ds} \\ &\geq N_{ds} \end{aligned} \tag{26}$$

if $N_{cp} > 2N_{ds}$. Evaluating for $t = t_2$:

$$\begin{aligned} t_{sam} &= t_2 - N_{ds} \\ &= N_{ds} + N_{cp} - N_{ds} \\ &= N_{cp} \end{aligned} \tag{27}$$

This timing algorithm was verified by simulation using data symbols for the MPP (2ms delay spread) channel. A similar algorithm is used for the FreeDV OFDM modes. It has the disadvantage of requiring $N_{cp} > 2N_{ds}$ which is wasteful of power. Further work:

- Research other OFDM timing algorithms.
- Observe the timing algorithm in action for a range of multipath channels, e.g. evolution of the timing estimate over time as the multipath channel evolves.
- In the April 2024 stored file OTA test campaign, no Cyclic Prefix was used (due to a configuration error) however good results were obtained from decoded speech samples, suggesting the RADAE decoder is quite robust to ISI, or perhaps ISI is interpreted as another source of noise and is handled by the decoder without breaking down (but perhaps with some loss in performance). To explore this topic it would be useful to derive an objective measure (e.g. the ML loss function) to determine distortion as a function of cyclic prefix length.
- A method of measuring and observing channel impulse responses (e.g. on international DX paths) would be very useful to determine an appropriate N_{cp}.
- The vector of M samples that maximises D_t (e.g. t_2) may have ISI. It may be better to use a time shifted version of $\boldsymbol{p}$ as the prototype time domain pilot sequence.

8 SNR Estimation

The April 2024 stored file OTA test campaign showed the need for a method of measuring the SNR from off air RADAE signals. In existing FreeDV modes, SNR estimation is performed by estimating the mean and variance of the payload symbol constellation points. However this method relies on known constellation points which is not a valid assumption for ML constellations. In our hybrid DSP-ML system, the only part of the signal that is known to both the Tx and Rx are the pilots. This section presents a SNR estimation algorithm based on the time domain pilot symbol sequence.

Consider the normalised cross correlation function from [1]:

$$C_t^2 = \frac{|\boldsymbol{r}^H \boldsymbol{p}|^2}{\boldsymbol{r}^H \boldsymbol{r}} \tag{28}$$

When we have a good fine timing alignment and small frequency offset, the vector of received samples can be expressed as:

$$\boldsymbol{r} = g\boldsymbol{p} + \boldsymbol{n} \tag{29}$$

where g is a scalar complex gain and $\boldsymbol{n}$ is a vector of noise samples. Expanding (28):

$$\begin{aligned} C_t^2 &= \frac{|(g\boldsymbol{p} + \boldsymbol{n})^H \boldsymbol{p}|^2}{|g|^2 \boldsymbol{p}^H \boldsymbol{p} + 2(g\boldsymbol{p})^H \boldsymbol{n} + \boldsymbol{n}^H \boldsymbol{n}} \\ &= \frac{|g^H \boldsymbol{p}^H \boldsymbol{p} + \boldsymbol{n}^H \boldsymbol{p}|^2}{|g|^2 \boldsymbol{p}^H \boldsymbol{p} + 2(g\boldsymbol{p})^H \boldsymbol{n} + \boldsymbol{n}^H \boldsymbol{n}} \end{aligned} \tag{30}$$

As $\boldsymbol{n}$ and $\boldsymbol{p}$ are uncorrelated we assume the dot product terms of $\boldsymbol{n}$ and $\boldsymbol{p}$ are small compared to the other terms:

$$\begin{aligned} C_t^2 &\approx \frac{|g^H \boldsymbol{p}^H \boldsymbol{p}|^2}{|g|^2 \boldsymbol{p}^H \boldsymbol{p} + \boldsymbol{n}^H \boldsymbol{n}} \\ &= \frac{|g|^2 (\boldsymbol{p}^H \boldsymbol{p})^2}{|g|^2 \boldsymbol{p}^H \boldsymbol{p} + \boldsymbol{n}^H \boldsymbol{n}} \end{aligned} \tag{31}$$

Power is energy per unit time. Let us define the signal power S as the energy per M samples and noise power N as the noise energy per M samples. From (29):

$$\begin{aligned} S &= |g\boldsymbol{p}|^2 \\ &= |g|^2 \boldsymbol{p}^H \boldsymbol{p} \\ N &= \boldsymbol{n}^H \boldsymbol{n} \end{aligned} \tag{32}$$

Expressing (31) in terms of S and N and solving for S/N:

$$\begin{aligned} C_t^2 &= \frac{S \boldsymbol{p}^H \boldsymbol{p}}{S + N} \\ S/N &= \frac{C_t^2}{\boldsymbol{p}^H \boldsymbol{p} - C_t^2} \end{aligned} \tag{33}$$

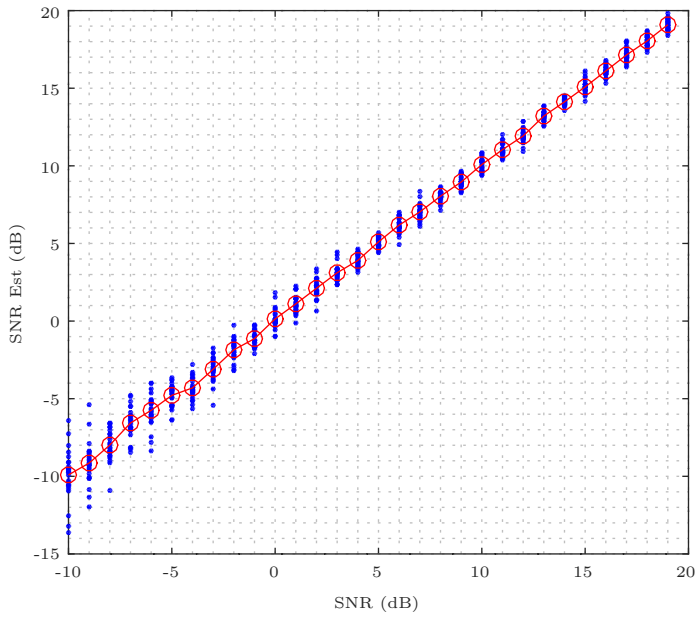
This algorithm was simulated in *est_snr.py* with reasonable results over the range -10 to +20 dB SNR on an AWGN channel (Figure 13). While there is significant variance in the estimates at low SNRs, the mean appears correct which suggests averaging SNR estimates over multiple modem frames should lead to reasonable SNR estimates.

The estimator was integrated into *inference.py* and tested on rate F_s AWGN and MPP signals. The estimated SNR values were adjusted to reference a 3000 Hz noise bandwidth. The estimator perfomed well for AWGN channels, however Table 10 shows it performs poorly for multipath channels. The problem may be the assumption in (29) of a single phase $arg[g]$ describing the channel. In practice multipath channels have varying phase across frequency so the received sequence $\boldsymbol{r}$ may be quite distorted compared to $\boldsymbol{p}$, lowering the SNR estimates.

Target Eb/No (dB)	20	10	3	0
Target SNR (dB)	18.24	8.24	1.24	-1.76
Measured SNR (dB)	19.73	9.78	2.78	-0.22
Estimated SNR (dB)	9.57	5.93	0.43	-2.2
Delta (dB)	-10.16	-3.85	-2.35	-1.98

Table 10: SNR estimator running in *inference.py* on a MPP channel. AWGN performance was much better, with the estimate delta being around 0.25 dB

Figure 13: Scatter plot of actual versus estimated SNR values for a 15 carrier waveform, $M = 160$ sample pilot sequence, AWGN channel.



Further work:

- Research other SNR estimators more robust to multipath channels.
- For the purpose of stored file experiments it is acceptable to send a separate signals (e.g. a carrier or set of known symbols) a few seconds before or after the test signal.
- Consider training a ML based SNR estimator. The SNR of each training sequence could be considered an output on the network and a loss function term added.

9 Acquisition

In a practical PTT voice system a transmitter will start and stop transmissions at a time unknown to the remote receiver. When a received signal is present, it will have unknown timing and frequency offsets. An acquisition system is required to determine if a valid signal is present, and to determine estimates of the frequency and timing offsets. It must also determine when transmission ceases, to prevent decoding when no signal is present.

Given the lack of structure in the ML symbols, the only "known" part of the signal is the periodic pilot sequence $r(n) = gp(n)$, send once per modem frame. We have therefore chosen to use classical DSP techniques for acquisition. This is typically detected by maximising D_t (20) over a grid of time and frequency offsets. Once the pilot sequence is detected, a state machine is used to control when decoding of the received ML symbols starts and stops.

9.1 Pilot Detection and N_c

From (23), if we assume constant pilot magnitude $|P_c| = P$ and ideal timing the maximum is:

$$|D_{tmax}| = gMN_cP^2 \tag{34}$$

This maximum is compared to a threshold D_{thresh} that is computed from σ, the standard deviation of $|D_t|$:

$$D_{thresh} = \sigma\sqrt{-ln(P_{thresh})} \tag{35}$$

using a sufficiently low probability P_{thresh}. If $|D_t| > D_{thresh}$, we can conclude that $p(n)$ has been detected. Let us consider the probability of detection of two different $p(n)$ with the same S/N but different N_c. If we assume channel noise dominates σ, then the same noise power N will produce the same D_{thresh} for any N_c. This reduces the problem to comparing D_t for two $p(n)$ sequences with different N_c but the same power.

We denote the two choices of N_c as N_{c1} and N_{c2}. The signals are a series of impulses in the frequency domain of magnitude P. The power can be conveniently computed in the frequency domain as:

$$\begin{aligned} S_1 &= g_1^2 N_{c1} P^2 \\ S_2 &= g_2^2 N_{c2} P^2 \end{aligned} \tag{36}$$

To achieve the same S/N for constant N we set $S_1 = S_2$ and solve for g_1:

$$g_1 = g_2\sqrt{\frac{N_{c2}}{N_{c1}}} \tag{37}$$

Computing the ratio of the two D_t maxima:

$$\begin{aligned} |D_{t1}| &= g_1 M N_{c1} P^2 \\ |D_{t2}| &= g_2 M N_{c2} P^2 \\ \frac{D_{t1}}{D_{t2}} &= \sqrt{\frac{N_{c1}}{N_{c2}}} \end{aligned} \tag{38}$$

For example doubling the number of carriers $N_{c1}/N_{c2} = 2$ increases the maximum by $D_{t1}/D_{t2} = \sqrt{2}$, significantly increasing the probability of detection for a given S/N.

Table 11 shows some experimental results for *model17* and *model18*. The acquisition performance of *model17* ($N_c = 30$) supports the theory that a larger

N_c gives better acquisition performance. The high SNR D_{thresh} baseline is due to the σ of the noise free $|D_t|$ samples at non-ideal timing and frequency offsets. The mean $D_{t1}/D_{t2} = 1.39$ (across all SNRs) compared to 1.41 predicted by (38).

Model	N_c	C/N_0	D_{thresh}	D_{tmax}	N_{frames}
17	30	132	3.1	5.2	4
18	15	132	3.1	4.1	4
17	30	42	3.2	5.4	4
18	15	42	3.1	4.0	4
17	30	35	3.6	5.8	4
18	15	35	3.3	4.2	8
17	30	32	4.1	5.8	4
18	15	32	3.5	4.2	24
17	30	30	4.6	6.5	4
18	15	30	4.3	4.2	68

Table 11: *dt_test.sh* simulation results for AWGN channel for *model17* and *model18* waveforms. N_{frames} is the number of frames it takes to sync using the *rx.py* state machine. These results suggest a larger N_c (*model17*) has better performance.

9.2 Pilot Detection over Multiple Frames

The pilot detector computes $|D_t|$ as the inner product of

$$D_t = \sum_{n=0}^{M-1} r^*(n)p(n) \tag{39}$$

When we have good time alignment with the received pilot:

$$\begin{aligned} r(n) &= gp(n) + noise(n) \\ D_{tmax} &= \sum_{n=0}^{M-1} g^* p^*(n)p(n) + \sum_{n=0}^{M-1} noise(n) * p(n) \\ |D_{tmax}| &\approx gN_cMP^2 \end{aligned} \tag{40}$$

as the noise and pilot samples are uncorrelated so their inner product is small. Consider the other possible signals $r(n)$ that may be processed by the pilot detector:

$$r(n) = \begin{cases} r(n) = gp(n-t) + noise(n), & t \neq 0 \\ r(n) = noise(n) \\ r(n) = gtx(n) + noise(n) \end{cases} \tag{41}$$

In the cases (41) we model $|D_t|$ as a Rayleigh distributed random variable with scale parameter σ_r and probability density function:

$$f(x;\sigma_r) = \frac{x}{\sigma_r^2} e^{-x^2/(2\sigma_r^2)}, \quad x \geq 0 \tag{42}$$

The cases in (41) are further explained:

1. Non-time aligned pilot and additive noise.
2. Noise only (no signal present).
3. Non-pilot symbols from the transmitter (e.g. ML symbols) with additive noise. This is a common scenario in streaming voice, where we often attempt to acquire the signal well after transmission has started. For high SNRs, the power of $gtx(n)$ will dominate the received signal power and also the power (or variance σ^2) of $|D_t|$. Note that for PAPR optimised waveforms, $gtx(n)$ may be several dB higher in power than the pilot $gp(n)$, degrading the ratio of D_{tmax} compared to σ compared to non-PAPR optimised waveforms.

We would expect to see a peak (40) in $|D_t|$ every modem frame or every N_{mf} samples. Consider the case where we sum $|D_t|$ from two adjacent modem frames:

$$\begin{aligned} |D_{t12}| &= \left|\sum_{n=0}^{M-1} r^*(n)p(n)\right| + \left|\sum_{n=0}^{M-1} r^*(n+N_{mf})p(n)\right| \\ |D_{t12}| &= |D_{t1}| + |D_{t2}| \end{aligned} \tag{43}$$

This can be seen as the sum of two random variables. Unfortunately there is no closed form expression for the sum of Rayleigh distributions [2]. Let us attempt to use the properties of Gaussian random variables to find an approximation. The variance of the sum of two Gaussian random variables is:

$$Var(X1 + X2) = Var(X1) + Var(X2) + 2Cov(X1, X2) \tag{44}$$

Due to the additive noise component and time varying nature of $tx(n)$ we assume for all cases in (41) $|D_{t1}|$ and $|D_{t2}|$ are independent so $2Cov(|D_{t1}|, |D_{t2}|) = 0$. We also assume they have the same variance $Var(|D_{t1}|) = Var(|D_{t2}|)$ as the noise or signal power components in (41) are likely to have the same power when separated by just one modem frame:

$$\begin{aligned} Var(|D_{t1}| + |D_{t2}|) &= Var(|D_{t1}|) + Var(|D_{t2}|) + 2Cov(|D_{t1}|, |D_{t2}|) \\ \sigma_{12}^2 &= \sigma^2 + \sigma^2 \\ \sigma_{12} &= \sqrt{2}\sigma \end{aligned} \tag{45}$$

Observing that the Rayleigh scale parameter is proportional to σ, we apply this approximation to the Rayleigh scale parameter to obtain an approximation of the PDF of the sum of two Rayleigh RVs, each have a scale parameter σ_r:

$$\begin{aligned} \sigma_{r12} &\approx \sqrt{2}\sigma_r \\ f_{12}(x;\sigma_r) &\approx \frac{x}{2\sigma_r^2} e^{-x^2/(4\sigma_r^2)}, \quad x \geq 0 \end{aligned} \tag{46}$$

Figure 14: PDF approximation from (46) and simulation histogram for sum of two Rayleigh RVs $X1$ and $X2$.

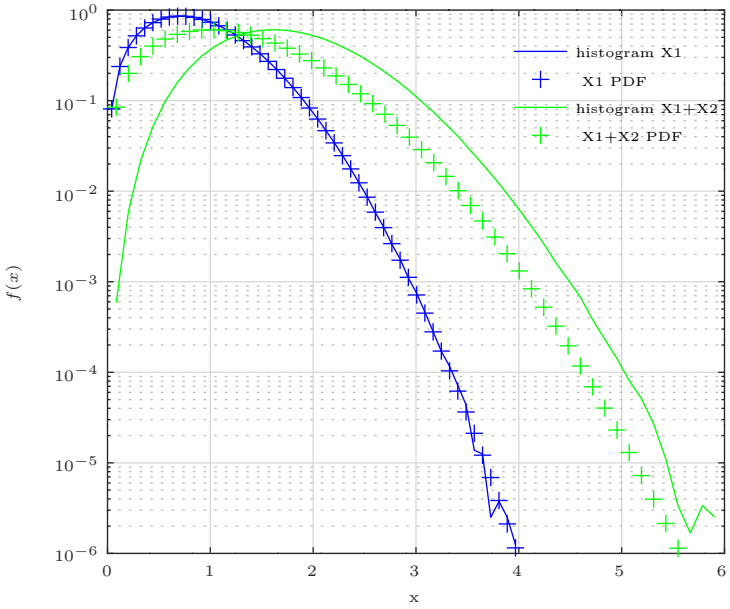


Figure 14 shows the results of using this approximation. In the region of interest $f(x) \approx 1 \times 10^{-4}$ the probability density derived from (46) is around 5 times the measured histogram. Now for a *single* Rayleigh random variable the probability of a sample of $|D_t|$ exceeding x is the area of the region to the right of x or $1 - CDF(x)$:

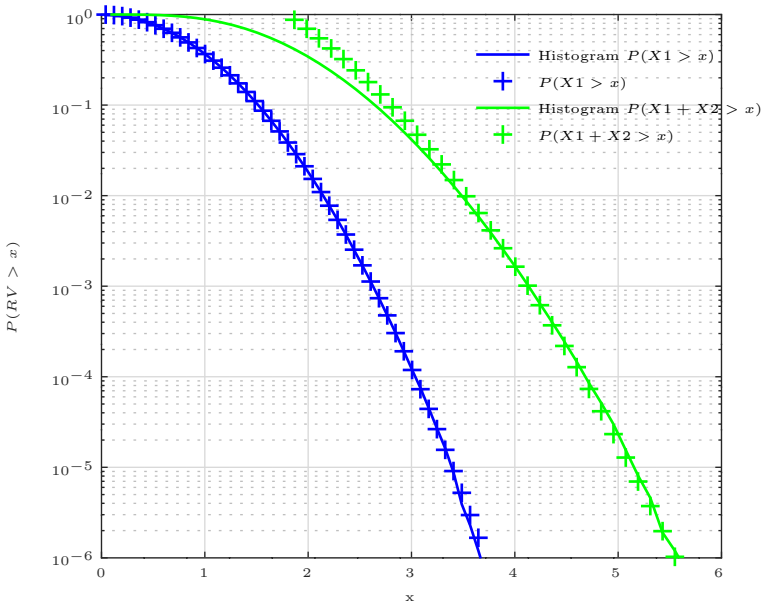
$$P(|D_t| > x, \sigma_r) = e^{-x^2/(2\sigma_r^2)} \tag{47}$$

Applying (46) and the correction factor of 5 from Figure (14) to the sum of two Rayleigh RVs:

$$\begin{aligned} P(|D_{t1}| + |D_{t2}| > x, \sigma_r) &\approx 5e^{-x^2/(2\sigma_{r12}^2)} \\ &= 5e^{-x^2/(4\sigma_r^2)} \end{aligned} \tag{48}$$

Figure 15 compares (48) to a histogram of simulated values.

Figure 15: Probability of the sum of two Rayleigh RVs exceeding a threshold. Equation (48) has a good match to the simulation histogram for $P(RV > x) < 1 \times 10^{-2}$.



Consider the detector output due to a correctly aligned pilot signal $|D_t| = |D_{tmax}|$. When we sum two adjacent modem frames:

$$\begin{aligned} |D_{tmax12}| &= |D_{tmax1}| + |D_{tmax2}| \\ &= 2|D_{tmax}| \end{aligned} \tag{49}$$

Given $|D_{t1}|$ and $|D_{t2}|$ have the same scale parameters σ_r, we can use (47) and (48) we can compare the probability of false detection of a pilot, which occurs when a sample of $|D_t|$ from the cases (41) exceeds a threshold D_{tmax}:

$$\begin{aligned} P &= P(|D_t| > |D_{tmax}|, \sigma_r) \\ &= e^{-|D_{tmax}|^2/(2\sigma_r^2)} \\ P_{12} &= P(|D_{t1}| + |D_{t2}| > |D_{tmax12}|, \sigma_r) \\ &= 5e^{-(|D_{tmax12}|/(4\sigma_r)^2} \\ &= 5e^{-(|2D_{tmax}|^2/(4\sigma_r))^2} \\ &= 5\left(e^{-|D_{tmax}|^2/(2\sigma_r^2)}\right)^2 \\ P_{12} &= 5P^2 \end{aligned} \tag{50}$$

For example if $P = 1 \times 10^{-3}$, $P_{12} = 5 \times 10^{-6}$, a significant improvement. Note this relationship holds for any σ_r and D_{tmax}. We can now set out a procedure for pilot detection:

1. Sample $|D_{t1}|$ and $|D_{t2}|$ over a range of time and frequency offsets.
2. Determine the sample $|D_{tmax12}|$ from the set of summed samples $|D_{t12}|$.
3. Determine σ_r. Note this the Rayleigh scale parameter for $|D_{t1}|$ or $|D_{t2}|$ alone, not the sum. One convenient way is to use the Rayleigh distribution property $\sigma_r = \frac{E[|D_{t1}|]}{\sqrt{\pi/2}}$. It may be useful to average across both RVs $\sigma_r = (\sigma_{r1} + \sigma_{r2})/2$.
4. Evaluate (48) to determine the probability of this being a non-pilot sample, as input to the sync state machine. Alternatively this can be framed in terms of comparing $|D_{tmax12}|$ to a target threshold D_{thresh}, calculated for a desired false detection probability.

9.3 Pilot Tracking

Once synchronised we continue to search for the pilot sequence over a small timing and frequency window, in order to track time and frequency offsets as they evolve and detect the sudden absence of the pilot signal that would indicate transmission has stopped.

The pilot detection algorithm in Section 9.2 requires a statistic of $|D_t|$ (such as mean or variance) in order to calculate the Rayleigh scale parameter σ_r. This must contains samples that are mainly from the cases in (41) where the pilot is not strongly correlated with the received signal. During acquisition this is achieved by sampling $|D_t|$ over a large grid of time and frequency offsets.

Once synchronised we do not wish to calculate a large number of $|D_t|$ samples due to CPU constraints. This section presents a low complexity algorithm for estimating σ_r from the received signal. Consider a single sample of the random variable $|D_t|$ at time t:

$$\begin{aligned} |D_t| &= \sum_{n=0}^{M-1} r^*(n)p(n) \\ &= \sum_{n=0}^{M-1} \sigma_{rx} r_n^*(n)\sigma_p p_n(n) \\ &= \sigma_{rx}\sigma_p \sum_{n=0}^{M-1} r_n^*(n)p_n(n) \end{aligned} \tag{51}$$

where σ_{rx} and σ_p are the standard deviation of $r(n)$ and $p(n)$. It can be seen that $|D_t|$ is proportional to σ_{rx} and σ_p. Using a simulation *est_sigma_dt.py* an approximation for the standard deviation of $|D_t|$ was found to be:

$$Std(|D_t|) \approx \frac{\sigma_{rx}\sigma_p}{\sqrt{5}} \tag{52}$$

Using the properties of Rayleigh distribution the scale factor can be found:

$$\begin{aligned}\sigma_r &= \frac{Std(|D_t|)}{\sqrt{(4-\pi)/2}} \\ &= \frac{\sigma_{rx}\sigma_p}{\sqrt{10-5\pi/2}}\end{aligned} \tag{53}$$

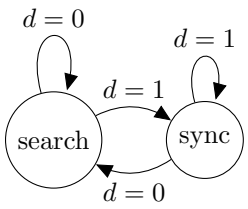
9.4 Sync State Machine

Figure 16 illustrates the basic operation of the sync state machine. In the *search* state we maximise D_t over a grid of time and frequency values. When $|D_t| > D_{thresh}$ we enter the *sync* state. In the sync state D_t is sampled over a smaller range of timing offsets to track sample clock offsets and the optimum sampling instant in a multipath channel. When the signal disappears, D_t will drop beneath D_{thresh} and we will return to the *search* state. The pilot detector output d is defined as:

$$d = \begin{cases} 0, & |D_t| < D_{thresh} \\ 1, & |D_t| \geq D_{thresh} \end{cases} \tag{54}$$

The state machine "clock speed" is once per modem frame (120ms for the current, *Candidate 2* waveform).

Figure 16: Sync state machine



Practical experience has taught us that reliable HF acquisition systems are challenging to build, they are often a source of hard to fix bugs and acquisition across a wide frequency range requires relatively high CPU. It is therefore prudent to design carefully. Table 12 sets out some requirements for the acquisition system. No acquisition system is perfect, so we define behaviour in terms of probabilities and mean event times. To design an acquisition system we set reasonable "mean time" requirements, then derive the probabilities required to meet these requirements. The ability to meet these these requirements can be determined by analysis and verified by simulation.

	Requirement	Symbol	Value
A1	Frequency offset		± 50 Hz
A2	Prob we enter sync with no signal present	$P(d=0\|\overline{sig})$	
A3	Prob we enter sync with signal present	$P(d=1\|sig)$	
A4	Prob we leave sync with signal present	$P(d=0\|sig)$	
A5	Prob we leave sync with no signal present	$P(d=0\|\overline{sig})$	
A6	Mean time between false syncs with no signal present	T_{false}	> 120 s
A7	Mean time to sync when signal present	T_{sync}	< 0.5 s
A8	Mean time to lose sync when signal present	T_{false_unsync}	> 600 s
A9	Mean time to lose sync when no signal present	T_{unsync}	< 0.5 s

Table 12: Acquisition system requirements. The probabilities will be a function of SNR and channel type (multipath is always harder).

9.5 Acquisition Further work

- Use ML to perform acquisition. Question of suitable feature set (e.g. FFT mag samples), or do we run at rate F_s.
- A different pilot distribution might be smarter, e.g. staggered through carriers.
- In high SNR channels, $D_{tmax} >> D_{thresh}$ so we could make a decision in one frame, speeding acquisition. We could solve (35) for the probability, and make a one frame sync decision if probability is very high.

10 Glossary

Symbol	Explanation	Units
B	noise or signal bandwidth	Hz
C	RMS carrier power $C = S$ for this study	
C/N_0	Carrier power/spectral noise density	
d	dimension of latent vector $\mathbf{z}$	
E_b/N_0	energy per BPSK symbol on spectral noise density	
E_q/N_0	energy per QPSK symbol on spectral noise density	
N	total noise power	Watts
N_c	Number of carriers	
N_0	Noise power in 1 Hz of bandwidth	
P/N_0	Peak power/spectral noise density	
$\mathbf{q}$	vector of QPSK symbols	
q_i	single QPSK symbol, element of $\mathbf{q}$	
R_b	BPSK symbol rate	symbols/second
R_q	QPSK symbol rate	symbols/second
R_s	OFDM per carrier QPSK symbol rate	symbols/second
R_z	latent vector update rate	Hz
SNR	signal to noise Ratio	
S	total signal (carrier) power	Watts
T_b	BPSK symbol period	seconds
T_q	QPSK symbol period	seconds
T_s	OFDM per carrier QPSK symbol period	seconds
T_z	time between latent vector updates	seconds
r_i	noise sample	
$\mathbf{z}$	Autoencoder output latent vector	
z_i	single latent vector element of $\mathbf{z}$, a BPSK symbol	

Table 13: Glossary of Symbols

11 References

[1] Low SNR FreeDV Modes. `https://github.com/drowe67/misc/blob/master/freedv_low/freedv_low.pdf`.

[2] J. Hu and N.C. Beaulieu. Accurate simple closed-form approximations to rayleigh sum distributions and densities. *IEEE Communications Letters*, 9(2):109–111, 2005.

[3] David Rowe. FreeDV-032 Radio Autoencoder Waveform Design (spreadsheet).

[4] Jean-Marc Valin, Jan Büthe, and Ahmed Mustafa. Low-bitrate redundancy coding of speech using a rate-distortion-optimized variational autoencoder, 2023.

[5] Jean-Marc Valin, Ahmed Mustafa, and Jan Büthe. Very low complexity speech synthesis using framewise autoregressive gan (fargan) with pitch prediction, 2024.

Figure 5: OFDM pilot based synchronisation algorithm performance, tested by measuring the BER obtained using discrete PSK symbols on an AWGN channel. With ideal sync, the autoencoder produces intelligible speech at $E_b/N_0 = -6$ dB which corresponds to BER=0.24. Several algorithms, combined with gain and frequency offsets were simulated.

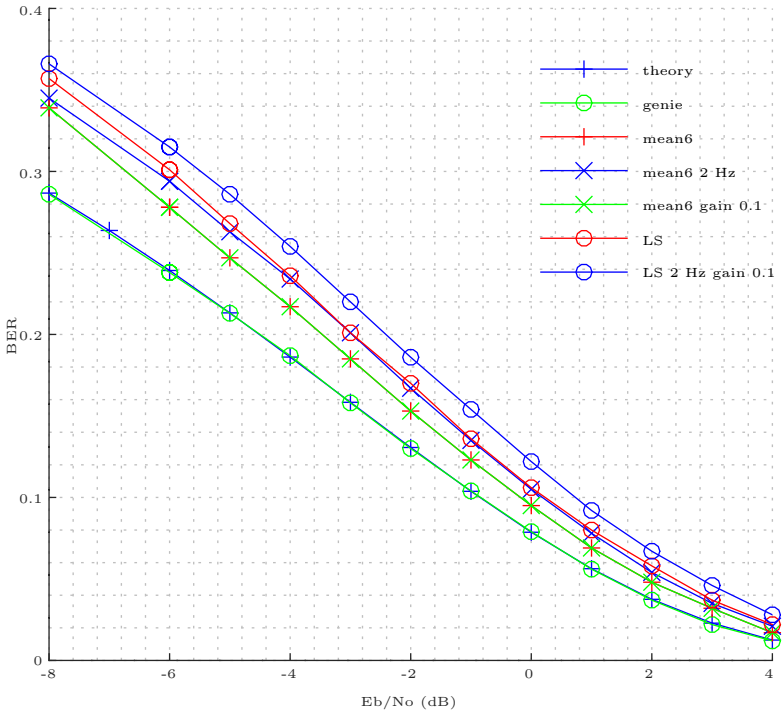


Figure 6: OFDM pilot based synchronisation algorithm performance, tested by measuring the BER obtained using discrete PSK symbols on an multipath (MPP) channel. The "genie" curve is slightly better than theory, probably due to an insufficiently long run. At the BER=0.24 threshold, the *LS* algorithm has a 2dB loss compared to the "genie" curve

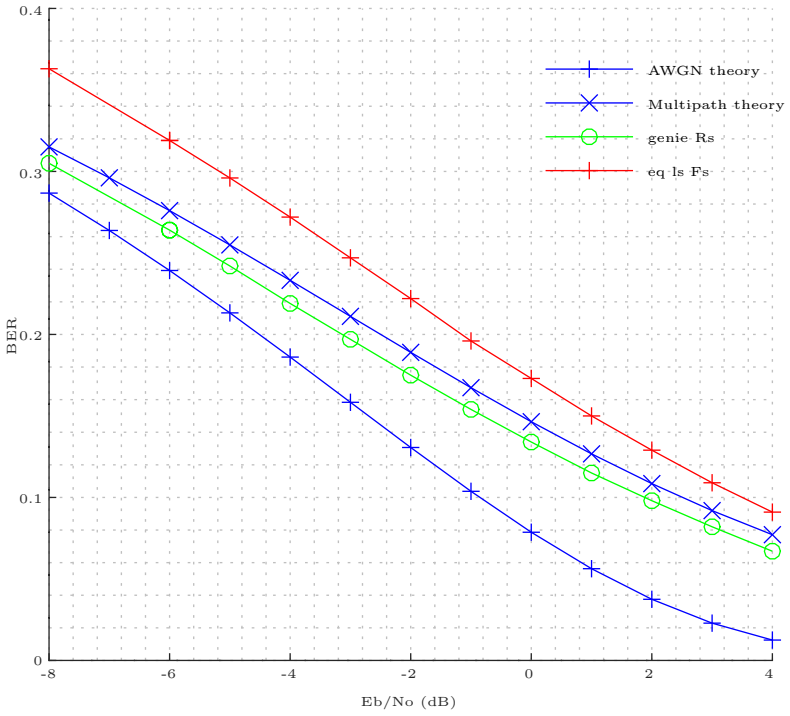


Figure 10: Inference time PAPR optimised system, showing integration with classical DSP OFDM components. Pilot symbols $\mathbf{p}$ are inserted into the modem frame with a hand tuned gain p_g to adjust PAPR. The pilots are constant and known to both the Tx and Rx. A Cyclic Prefix (CP) appended to each time domain symbol provides protection from ISI. Acquisition and equalisation is perfomed using classical pilot-based DSP.