FreeDV-015 Codec 2 Rate K Resampler

David Rowe

Revision: 83e59b8 on branch: master

December 10, 2023

1 Introduction

To efficiently transmit spectral amplitude information Codec 2 700C uses a set of algorithms collectively denoted *newamp1*. One of these algorithms is the Rate K resampler which transforms the variable length vectors of spectral amplitude samples to fixed length K vectors suitable for vector quantisation. This document was written in order to explore and possibly improve rate K resampling, and the issues around quantising the resampled vectors. The results of this study are several algorithm innovations and candidate designs for new 700 and 1200 bit/s modes.

2 Resampler Model

Consider a vector $\mathbf{a}$ of L spectral amplitudes, sampled at time $t = nT$ seconds, where n is the frame number, and T is the frame period, typically $T = 0.01$ seconds.

$$\mathbf{a} = \begin{bmatrix} A_1, A_2, \dots A_L \end{bmatrix} \tag{1}$$

A_m is sampled at the frequency $f_m = mF0$ Hz for $m = 1 \dots L$, where $F0$ is the fundamental frequency (pitch) in Hz of the current frame, and L is given by:

$$L = \left\lfloor \frac{F_s}{2F0} \right\rfloor \tag{2}$$

$F0$ and L are time varying as the pitch track evolves over time. For speech sampled at $F_s = 8$ kHz $F0$ is typically in the range of 50 to 400 Hz, giving L in the range of 10 … 80.

To quantise and transmit $\mathbf{a}$, it is convenient to resample $\mathbf{a}$ to a fixed length K element vector $\mathbf{b}$ using a resampling function:

$$\begin{split} \mathbf{y} &= \begin{bmatrix} Y_1, Y_2, \dots Y_L \end{bmatrix} = H(\mathbf{a}) \\ \mathbf{b} &= \begin{bmatrix} B_1, B_2, \dots B_K \end{bmatrix} = R(\mathbf{y}) \end{split} \tag{3}$$

Where $H()$ is a filter function chosen to smooth the spectral amplitude samples A_m while not significantly altering the perceptual quality of the speech; and $R()$ is a resampling function. To model the response of the human ear B_k are sampled on K non-linearly spaced points on the frequency axis:

$$\begin{aligned} f_k &= warp(k,K) \text{ Hz} \quad k = 1 \ldots K \\ warp(1,K) &= 200 \text{ Hz} \\ warp(K,K) &= 3700 \text{ Hz} \end{aligned} \tag{4}$$

where $warp()$ is a frequency warping function. Codec 2 700C uses $K = 20$, and $warp()$ is defined using the Mel function [**?**, p 150] (Figure **??**) which samples the spectrum more densely at low frequencies, and less densely at high frequencies:

$$mel(f) = 2595 log_{10}(1 + f/700) \tag{5}$$

The inverse mapping of f in Hz from $mel(f)$ is given by:

$$f = mel^{-1}(x) = 700(10^{x/2595} - 1); \tag{6}$$

Figure 1: Mel function

We wish to use $mel(f)$ to construct $warp(k,K)$, such that there are K evenly spaced points on the $mel(f)$ axis (Figure **??**). Solving for the equation of a straight line we can obtain $mel(f)$ as a function of k, and hence $warp(k,K)$ (Figure **??**):

$$\begin{aligned} g &= \frac{mel(3700) - mel(200)}{K - 1} \\ mel(f) &= g(k-1) + mel(200) \end{aligned} \tag{7}$$

Substituting (**??**) into the LHS:

$$\begin{aligned} 2595 log_{10}(1 + f/700) &= g(k-1) + mel(200) \\ f = warp(k,K) &= mel^{-1}(g(k-1) + mel(200)) \end{aligned} \tag{8}$$

and the inverse warp function:

$$k = warp^{-1}(f,K) = \frac{mel(f) - mel(200)}{g} + 1 \tag{9}$$

The rate K vector $\mathbf{b}$ is vector quantised for transmission over the channel:

$$\hat{\mathbf{b}} = Q(\mathbf{b}) \tag{10}$$

Figure 2: Linear mapping of $mel(f)$ to Rate K sample index k

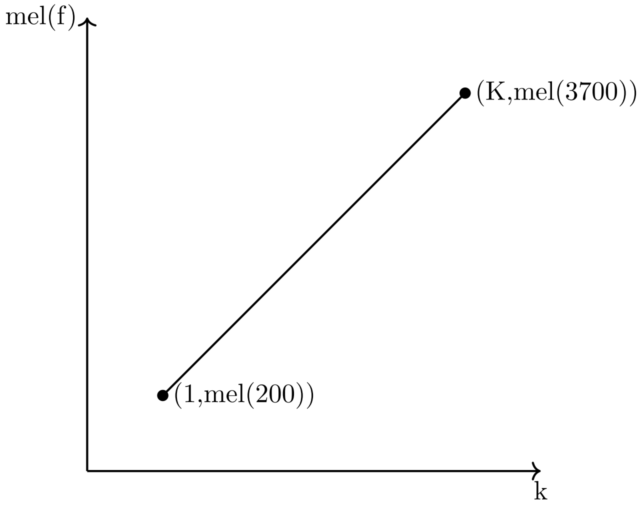


Figure 3: $warp(k,K)$ function for $K=20$

The rate filtered rate L vector can then be recovered by resampling $\hat{\mathbf{b}}$ using another resampling function:

$$\hat{\mathbf{y}} = S(\hat{\mathbf{b}}) \tag{11}$$

A useful error metric is the mean square error:

$$\begin{aligned} E &= \frac{1}{L_{max}-L_{min}+1}\sum_{m=L_{min}}^{L_{max}}(Y_m-\hat{Y}_m)^2 \\ L_{min} &= round(200/F0) \\ L_{max} &= \lfloor 3700/F0 \rfloor \end{aligned} \tag{12}$$

In the case where there is no filtering (e.g. `newamp1`) then:

$$\begin{aligned} H(\mathbf{x}) &= \mathbf{x} \\ \mathbf{y} &= \mathbf{a} \\ \hat{\mathbf{a}} &= \hat{\mathbf{y}} \\ E &= \frac{1}{L_{max}-L_{min}+1}\sum_{m=L_{min}}^{L_{max}}(A_m-\hat{A}_m)^2 \end{aligned} \tag{13}$$

If A_m are in dB, E can be denoted the spectral distortion in dB^2, which can be averaged over a testing database of N frames to obtain mean spectral distortion.

Consider a choice of $warp()$ with linear (non-warped) sampling of the frequency axis, no filtering such that $H(\mathbf{x}) = \mathbf{x}$, and an ideal quantiser Q such that $\hat{\mathbf{b}} = \mathbf{b}$. If $K < L$ information may be lost due to undersampling, which implies $\hat{\mathbf{a}} \neq \mathbf{a}$. With nonlinear sampling, there will be local undersampling where the sampling rate of $\mathbf{b}$ is less than that of $\mathbf{a}$:

$$warp(k+1, K) - warp(k, K) < F0 \tag{14}$$

If A_m is changing rapidly, undersampling may introduce undesirable aliasing, which may manifest as noise that is superimposed on $\mathbf{b}$. This noise may reduce perceptual quality and consume valuable quantiser bits for no benefit. Therefore $H()$ should be chosen to smooth high frequency detail such that local undersampling and uncontrolled aliasing is minimised. This may be restated as choosing $H()$ to minimise E (**??**) when $\hat{\mathbf{b}} = \mathbf{b}$.

3 Rate K Resampler Experiments

3.1 Suggested Experiments

Here are a suggested set of experiments to evaluate the ideas presented in this document. Some of them require informal listening tests, others have objective measures which could be used as the basis for automated tests:

1. The baseline resampling (currently a 2nd order polynomial) is a potential source of distortion. Conduct an experiment to test the theory that E is small (and perceptual quality high) for $K > L$ using linear frequency sampling, $H(\mathbf{x}) = \mathbf{x}$ and large K.

2. How to demonstrate aliasing? Well we can run current rate K code, that uses a 2nd order parabolic resampler. Then compare with a "better" resampler that uses filtering. Perform an informal listening test over a small set of samples. Goal is to show reduced E with similar perceptual quality. Smoothing does reduce information so there will be a trade off. Too much smoothing and perceptual quality will reduce. We shoMel-frequency cepstral coefficients (MFCCs)uld also notice improved VQ performance, as we won't be quantising noise.

3. A useful property is sensitivity to quantisation, which could be defined as $\frac{\partial E}{\partial \mathbf{b}}$. For example, given a 1dB RMS error in the elements of $\mathbf{b}$, what is the impact on E?

4. To minimise bit rate, it is common to transmit $\mathbf{b}$ to the receiver at period T/D seconds, where D is the decimation ratio, and discarding the intermediate $D-1$ frames. A useful property is the ability to smoothly interpolate between transmitted frames $\mathbf{b}_n$ and $\mathbf{b}_{n+D}$ to recover $\mathbf{b}_{n+i}$ where $i = 1 \ldots D-1$. Need a definition for smoothness.

5. Speech evolves slowly over time compared to the $T = 0.01$ second frame period. Adjacent frames of speech parameters such as $\mathbf{a}_n$ and $\mathbf{a}_{n+1}$ have some correlation which can be used to obtain coding efficiency:

$$\begin{aligned}\mathbf{b}_{n+1} &= \mathbf{b}_n + \boldsymbol{\Delta}_n \\ \boldsymbol{\Delta}_{n+1} &= \mathbf{b}_{n+1} - \mathbf{b}_n\end{aligned} \tag{15}$$

   In general $\boldsymbol{\Delta}_n$ can be encoded with less bits than $\mathbf{b}_n$. However consider the case where there is significant noise due to undersampling:

$$\hat{\mathbf{b}}_n = \mathbf{b}_n + \mathbf{n}_n \tag{16}$$

   where $\mathbf{n}_n$ is a vector of noise samples with an unknown distribution. Substituting into (**??**):

$$\boldsymbol{\Delta}_{n+1} = \mathbf{b}_{n+1} - \mathbf{b}_n + \mathbf{n}_{n+1} - \mathbf{n}_n \tag{17}$$

   If $\mathbf{n}_n$ and $\mathbf{n}_{n+1}$ are not well correlated they may become a significant source of noise that is summed with $\boldsymbol{\Delta}_n$, reducing the effectiveness of the quantiser that will need to waste bits quantising the noise. We would therefore expect that in the absence of undersampling noise, delta coding in time should result in increased quantiser efficiency.

6. Small changes in $\mathbf{a}$ input should result in small changes in $\mathbf{b}$ indicating a lack of sensitivity and undersampling noise in R. If R is sensitive, we may notice VQ choices changing from frame to frame for stationary speech.

7. If the sample rate K is sufficiently high (or bandwidth of $\mathbf{a}$ sufficiently constrained), the actual VQ dimension won't matter. The decorrelation properties of the VQ will ensure it achieves the same distortion over a range of dimensions. A large enough dimension K could be chosen to simplify S, which could be linear resampling. It would be good to decouple $warp()$ from K.

3.2 Experiment 1: rate $K > L$ linear

This experiment tests the ability to resample perfectly ($E = 0$) with $K > L$, no filtering ($H(x) = x$) and linear frequency sampling and was implemented with Octave scripts `ratek1_fbf.m` and `ratek1_batch.m`.

Figure **??** illustrates why the resampler choice is important. In the region of F1, the spectral samples A_m are changing quickly. The parabolic resampler fails to track these changes leading to distortion in a perceptually important feature of the spectrum. When the resampler is viewed as a filter, this can be interpreted as a low pass response in the parabolic resampler, and is most noticeable when K is close to L.

Figure **??** shows the spline and parabolic resampler E, plotted against $F0$ for a 24 second sample containing four speakers. Table **??** is the mean spectral

Figure 4: Frame 50 of big_dog sample with $K = 40$ just larger than $L = 37$, 2nd order parabolic resampler. Note distortion around F1 at 500 Hz

Resampler	mean E dB^2
spline	0.02
para	0.31

Table 1: Mean spectral distortion E for spline and parabolic interpolators

distortion E over the sample. The spline interpolator performs better than the parabolic interpolator. L is time varying, but as it approaches K, E increases, once again showing that interpolators struggle with K close to L.

In this experiment with $K = 80$ both resamplers have low average E (less than 1dB^2). On inspection using `ratek_fbf` the occasional high E frames were found to be unvoiced speech or background noise, where the pitch estimator tends to return low $F0$ (high L) estimates. Fortunately the ear is quite insensitive to spectral distortion in unvoiced of background noise frames, so it is unlikely these errors are audible. However with a lower choice of K we would start to get resampling distortion during voiced speech as K approaches L (Figure **??**), or when $K < L$, aliasing.

Conclusions:

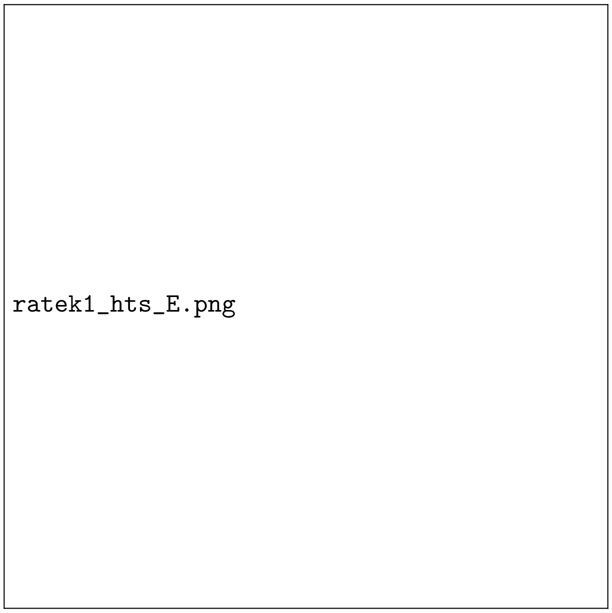
1. The resampler matters, especially when K is close to L. The current rate K Codec 2 700C system uses $K < L$ (at least at high frequencies) with a parabolic resampler and no filtering. This may be suffering from resampling noise that affects VQ performance and speech quality. It seems prudent to use a low distortion resampler. We do not want to add any additional distortion sources to our system.

2. We intend to use $K < L$, which will lead to aliasing distortion. It would be wise to filter A_m prior to resampling to remove the possibility of aliasing and resampler distortion, that is choose $H()$ such that with $\mathbf{b} = \hat{\mathbf{b}}$, E (**??**) is minimised.

3. Once filtered, there will be some minimum value of K_{min} required for $\mathbf{b}$ such that the rate L vector $\hat{\mathbf{f}}$ can be recovered with minimal E. However we are free to choose $K > K_{min}$ as the "bandwidth" of the sampled sequence (and presumably VQ distortion for a given number of bits) will be independent of K when $K > K_{min}$. While a larger K will use more VQ storage, it may simplify resampling at the receiver. If K is large enough, a simple linear resampler may suffice.

4. With non-linear frequency sampling, we may need a high local rate near F1 to accurately represent sharp formants, especially for males. There

Figure 5: Scatter plot of E versus $F0$ for *hts* sample, spline and parabolic resampler



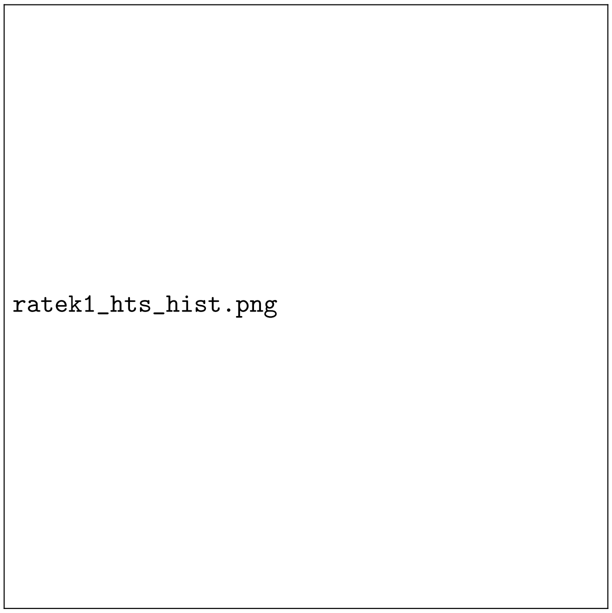
may be other ways to encode this information, for a example a resampling function or VQ that takes into account $F0$ - "sharpening" F1 for low $F0$ speakers.

5. The newamp1 postfilter is an experimentally derived algorithm that raises peaks and lowers troughs in the rate K spectrum, effectively "sharpening" formants. When used after vector quantisation it has a large impact on Codec 2 700C speech quality, especially for male speakers. However it's function is not well understood and in some cases it may be adding distortion. The sensitivity of F1 to low K resampling provides some hints as to why the postfilter is required. Resampler distortion in Figure **??** distorts F1, which is further degraded by vector quantisation. An increased sample rate and better resampling may reduce the need for the postfilter.

3.3 Experiment 2 - Filtering

The goal of this experiment is to filter the spectral amplitudes **a** such that they can be coarsely resampled without resampling or aliasing errors, and with minimal impact on the perceptual quality. The mth filtered sample can be

Figure 6: Histogram of E for spline and parabolic resampler



obtained by:

$$Y_m = \sqrt{\sum_{k=st}^{en} A_k^2 h(k)} \tag{18}$$

where A_k is the kth *linear* harmonic amplitude, $h(k)$ is sequence of samples defining a suitable filter, and st and en define the start and end harmonics used as input to the filtering operation for the mth sample. This formulation filters the harmonic energies, allowing high amplitude samples (peaks) to contribute more than low amplitude samples. This is an approximation of the ears masking behaviour, and was determined by experiment to slightly improve the peak/trough ratio of formants compared to filtering linear or log (dB) A_k.

As we wish to resample on a non-linear frequency axis, we need a set of filters that become wider as frequency increases. We can use the frequency warping function to divide the speech spectrum into N_b bands, with the centre frequency of each band given by:

$$\begin{aligned} f_b &= warp(b, N_b) \text{ Hz} \quad b = 1 \dots N_b \\ warp(1, Nb) &= 200 \text{ Hz} \\ warp(Nb, Nb) &= 3700 \text{ Hz} \end{aligned} \tag{19}$$

For this experiment, we wish to separate the filtering $H()$ from the rate K resampling operation, and perform the filtering on the rate L samples A_m. In this case it is convenient to treat b as a continuous variable. Given the mth harmonic, we can determine the filter band:

$$\begin{aligned} f_b &= mF0 \\ b &= warp^{-1}(f_b, N_b) \end{aligned} \tag{20}$$

A triangular function for $g(k)$ has been selected, which tapers to zero in the centre of adjacent bands $b-1$ and $b+1$. A similar triangular function is used for the filtering stage in the computation of Mel-frequency cepstral coefficients (MFCCs) [**?**]. This function can be defined in terms of the harmonic indexes, as illustrated in Figure **??**.

$$\begin{aligned} st &= max(1, round(warp(b-1, N_b)/F0)) \\ en &= min(L, round(warp(b+1, N_b)/F0)) \\ g(k) &= \frac{k-st}{m-st} \quad k = st \dots m-1 \\ g(m) &= 1 \\ g(k) &= \frac{en-k}{en-m} \quad k = m+1 \dots en \\ g_{sum} &= \sum_{k=st}^{en} g(k) \\ h(k) &= g(k)/g_{sum} \end{aligned} \tag{21}$$

Note the normalising term g_{sum} to ensure the energy of A_m is not altered by the filtering operation. Figure **??** is an example set of filters. It can be seen that N_b controls the filtering. As Nb becomes smaller, $g(k)$ becomes wider, reducing the rate of change of the filtered spectral samples Y_m. While this presumably improves VQ performance, at some point it will also affect the perceptual quality of the speech.

Figure 7: Triangular Filter $h(k)$. Note asymmetry on linear frequency axis

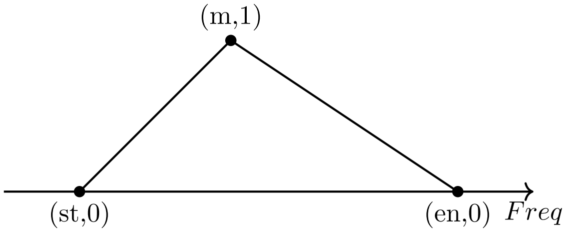
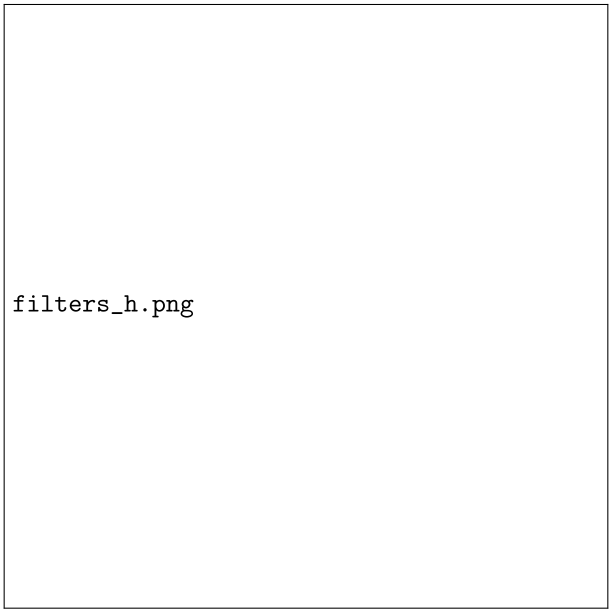


Figure **??** illustrates the effect of varying K for a family of filters by attempting to recover $\hat{\mathbf{y}} = S(R(\mathbf{y}))$ where $\mathbf{y} = H(\mathbf{a})$. As suggested in the conclusion of Experiment 1, once $K > K_{min}$, $\hat{\mathbf{y}}$ can be recovered with very low distortion E, even though $K < L$ for many frames.

Figure 8: Filters $h(k)$ for $N_b = 10, m = 1 \ldots L, F0 = 200\text{Hz}, L = 20$. There is no smoothing (length 1 filters) up to 1000 Hz, higher frequencies have progressively more smoothing.



As N_b increases, the filter width reduces such that $h(k) = 1$ for $k = m$ and $h(k) = 0$ for $k \neq m$. Thus $N_b = 100$ (where $N_b = 100 > L_{max}$=80) corresponds to no filtering. As expected, with $N_b = 100$ there is a large spectral distortion E due to undersampling.

Informal listening tests were conducted (3 samples, headphones, spline interpolator, `c2sim` tool) with $A_m = \hat{Y}_m$ to determine the subjective effects of filtering. For $N_b \geq 20$, the speech quality is not significantly affected. For $N_b \leq 15$ the speech became muffled and harder to understand, presumably as the formants become less well defined. The filters have the unfortunate property of reducing the spectral peak/trough ratio which the human ear uses to perceive speech (Figure **??**). A better smoothing function would preserve the peak/trough ratio of formants, while reducing only the frequency resolution (location and width of formants on the frequency axis).

It was also noted that even for $N_b = 100$, the speech quality was not significantly affected, despite the high objective distortion evident in Figure **??** for $Nb = 100$. This is puzzling, as there is significant difference between $\hat{\mathbf{y}}$ and $\mathbf{y}$ due to the aliasing from the rate K resampling $R()$, especially at high frequencies where condition (**??**) is not met. This suggests that the ear is insensitive to noise in the high frequency region, which could be exploited during quantisation

Figure 9: Frame 50 of *big_dog*, plot of filter output $\mathbf{y}$ for $Nb = 10$ and $Nb = 20$. Peak to trough ratio decreases with Nb, reducing intelligibility

ratek2_nb_big_dog_50.png

to optimise bit allocation.

Consider the rate K vectors obtained with ($\mathbf{b_1}$) and without ($\mathbf{b_2}$) filtering:

$$\begin{aligned} \mathbf{b_1} &= R(H(\mathbf{a})) \\ \mathbf{b_2} &= R(\mathbf{a}) \\ \mathbf{b_2} &= R(H(\mathbf{a})) + N(\mathbf{a}) \end{aligned} \tag{22}$$

where $N(\mathbf{a})$ is the noise created by local undersampling of $\mathbf{a}$. If $N(\mathbf{a})$ is even partially uncorrelated with $R(H(\mathbf{a}))$, we would expect to use more bits to quantise $\mathbf{b_2}$ than $\mathbf{b_1}$ to a given level of quantiser distortion E_q given by:

$$E_q = \frac{1}{K}\sum_{k=1}^{K}(b_k - \hat{b}_k)^2 \tag{23}$$

An experiment was conducted to train a two stage VQ (mean removed) with and without filtering, the results are presented in Table **??** and Figure **??**. A 120 second sample (12000 frames) was used to train the VQ. This VQ design and training material is similar to that used for Codec 2 700C. These results suggest that the lack of aliasing noise ($N_b = 20$) results in a smaller quantiser distortion E_q for a given bit allocation.

Figure 10: Mean spectral distortion E against K for a family of filters $H()$ for the 24 second *hts* sample. $N_b = 100$ corresponds to no filtering $H(x) = x$, the *para* case is equivalent to the *newamp1* algorithm rate K resampler.

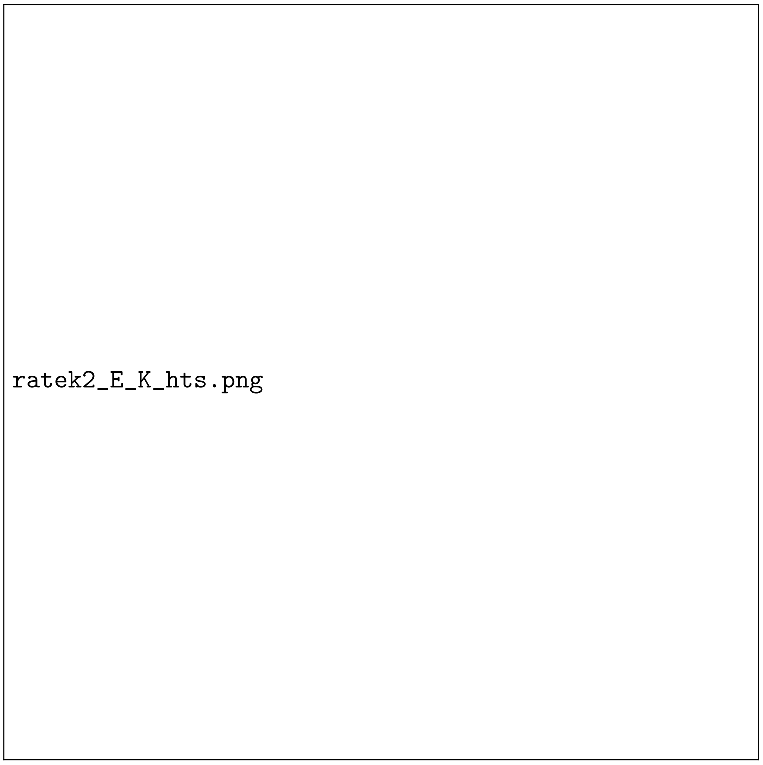


Figure **??** compares the Kmeans and LBG algorithms for VQ training. Kmeans initialises the $M = 512$ size VQ with random samples from the training set, then performs Kmeans iterations until a distortion metric is met. LBG trains a $M = 1$ (0 bit) VQ, then splits and Kmeans trains to get $M = 2$ (1 bit), and repeats the process until the final $M = 512$ size VQ is obtained. For this training set, both algorithms arrive at approximately the same final spectral distortion.

In Figure **??** and Figure **??** we note the 2nd stage performs quite poorly, there appears to be step change in the stage 2 gradient compared to the stage 1. The *mbest* multi-stage search algorithm was tried (using $mbest = 5$ survivors from the first stage) but provided no improvements in E_q. These results suggest:

1. The residual error from stage 1 may be a set of K independent Gaussians, with very little correlation between them. Intuitively, it makes sense that the first VQ stage would train to make the stage 1 residual error variance

Filter	mean E_q dB^2	RMS E_q dB
$N_b = 20$	3.44	1.85
$N_b = 100$ (no filtering)	5.77	2.40

Table 2: Spectral distortion for 18 bit, 2 stage, mean removed VQ, with $K = 30$

in each vector element similar. Exploring the residual vectors in `Octave` shows this is indeed the case - zero mean Gaussian shaped histograms with variances around 7 dB^2 for most elements (curiously $k = 1$ and $k = 30$ higher at 9 and 12 dB^2).

2. The E_q after second stage VQ will reduce slowly as the number of bits increases.

3. The size of K will then be important - it takes more bits to quantise a vector of $K + 1$ independent Gaussians than a vector of size K. This is unlike the first stage (and the theory elsewhere in this study) which states $K > K_{min}$ will have little impact on VQ performance as the information (frequency content) in the vector is the same.

4. A larger single stage VQ may be a better choice. Interpolating the Figure **??** first stage line, a 12 bit single stage VQ would achieve a similar E_q to the 18 bit two stage VQ. It would still be feasible to store and search this VQ on a small machine (the entries in dB could be quantised to 8 bits/element, $K2^{12} = 123$ kbytes).

5. A two stage VQ with smaller K, with a trade off between resampler and VQ distortion. With smaller K, we would expect the second stage Eq to reduce for a given number of bits.

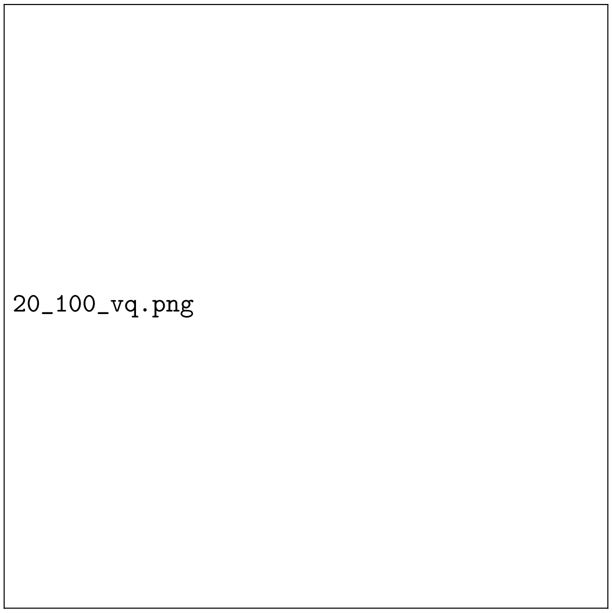
6. Other possibilities are a bug in this analysis (or the software tools), perhaps due to the small size of the training database.

7. An alternative quantiser based on PCA or DCT rather than direct VQ of $\mathbf{b}$. Consider a DCT of $\mathbf{b}$ followed by a DCT. The residual into a second stage would still likely be K Gaussians of similar variance, so may suffer the same poor second stage performance. A method that reduces the dimension of the Gaussians being quantised by the second stage VQ would be useful.

3.4 Efficient Filtering

As L is time varying, it is necessary to determine the filters $g(k)$ for every frame which is computationally inefficient. A more efficient procedure is to resample $\mathbf{a}$ to rate $L_{high} = L_{max}$ vector $\mathbf{x}$ using the resampler $T()$, perform the filtering

Figure 11: Mean VQ spectral distortion E_q against VQ size in bits for $Nb = 20$ and $Nb = 100$. Note the slope of the second stage indicates it performs quite poorly.



$H()$ at rate L_{high}, then resample $R()$ to the no

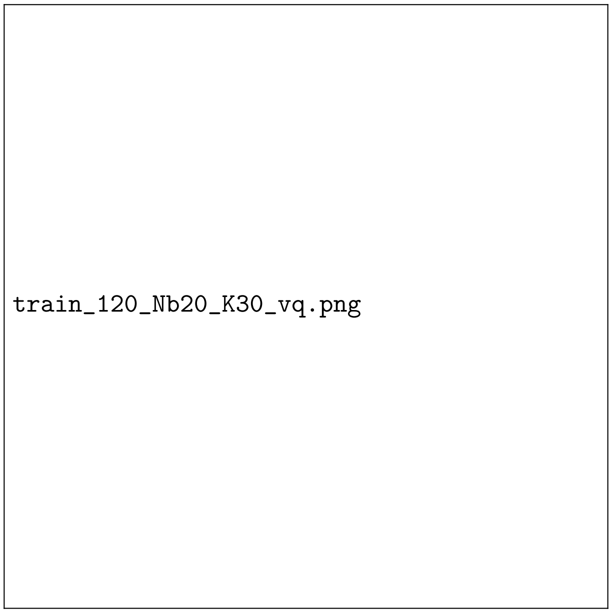
$$\begin{aligned}
\mathbf{x} &= T(\mathbf{a}) \\
\mathbf{y} &= H(\mathbf{x}) \\
\mathbf{b} &= R(\mathbf{y}) \\
\hat{\mathbf{b}} &= Q(\mathbf{b}) \\
\hat{\mathbf{y}}^L &= S(\hat{\mathbf{b}})
\end{aligned} \tag{24}$$

n-linearly spaced rate K vector $\mathbf{b}$. This allows $g(k)$ to be precomputed for rate L_{high}. The process now becomes:

$$\begin{aligned}
\mathbf{x} &= T(\mathbf{a}) \\
\mathbf{y} &= H(\mathbf{x}) \\
\mathbf{b} &= R(\mathbf{y}) \\
\hat{\mathbf{b}} &= Q(\mathbf{b}) \\
\hat{\mathbf{y}}^L &= S(\hat{\mathbf{b}})
\end{aligned} \tag{25}$$

where $\hat{\mathbf{y}}^L$ is sampled at rate L, unlike $\mathbf{y}$ which is sampled at rate L_{high}. Note

Figure 12: Kmeans and LBG VQ training for $Nb = 20$, very similar results for this training set.



that for the purpose of calculating spectral distortion E it may be convenient to configure $S()$ to extract $\hat{\mathbf{y}}^{Lhigh}$ at rate L_{high}.

In Figure **??**, the *Nb=20 eff* curve shows the results of the efficient filtering algorithm with $\hat{\mathbf{b}} = \mathbf{b}$, which as expected, are very close to the *Nb=20* curve.

3.5 Vector Quantiser Training

Using the algorithms developed above, the 24 minute (144,000 frame) *train.spc* sample was used to train several vector quantisers (Table **??**). Each frame was resampled to rate L_{high}, $N_b = 20$ filtered, and resampled to a training set of rate $K = 30$ vectors. The spectral distortion E_q results for the single stage VQ were disappointing, however the two stage VQ appears to perform quite well, and is a candidate for an improved 1200 bit/s Codec 2 mode. As a sanity check, two 4 second samples from within the training set (*big_dog* and *two_lines*) were tested with results consistent with the entire training data set.

Sample	Vector Quantiser	E_q dB^2	RMS E_q dB
train.spc	1 x 12 bit	4.84	2.20
train.spc	2 x 12 bit	1.7	1.3
big_dog	2 x 12 bit	2.03	1.43
two_lines	2 x 12 bit	2.04	1.43

Table 3: Spectral distortion for vector quantisers trained using *train.spc*

Figure 13: Two stage VQ of frame 61 of *big_dog* (male)

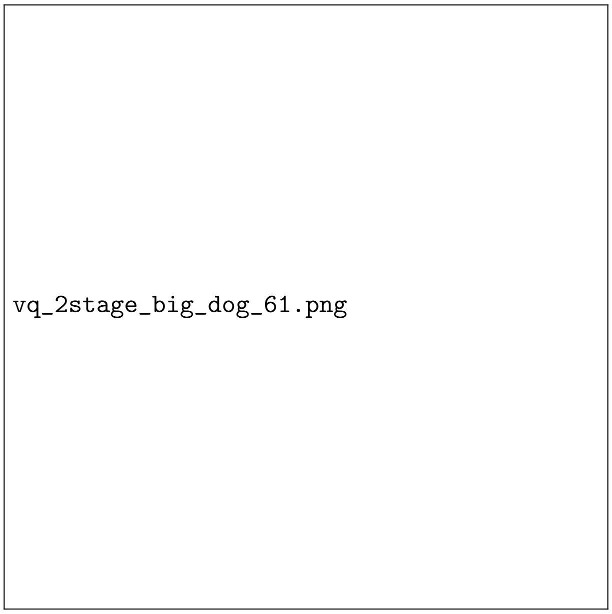
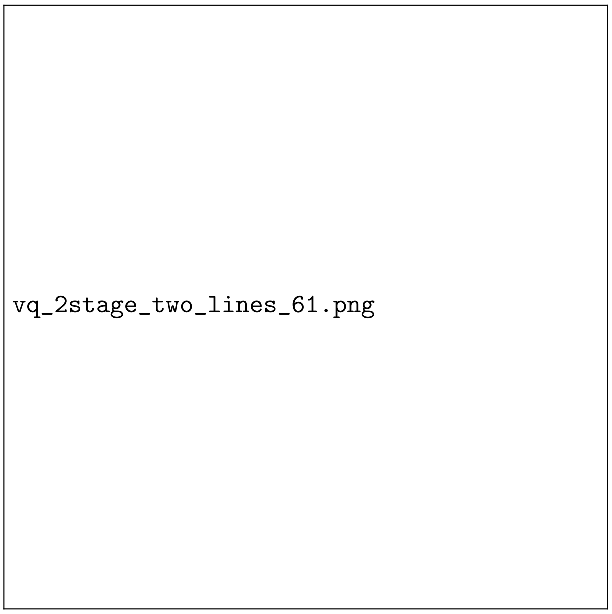


Figure 14: Two stage VQ of frame 61 of *two_lines* (female)



3.6 Pitch Dependant Spectral Envelope

Figures **??** and **??** overlay the speech spectrum and the corresponding upsampled and filtered rate L_{high} vector $\mathbf{y}$.

When $L < K$ (e.g. females), there are many interpolation functions that result in a low E fit. It has been observed that for some frames the peaks in $\mathbf{y}$ do not correspond to the peaks in the speech spectrum. This is acceptable if the quantisation error is small ($\hat{\mathbf{b}} \approx \mathbf{b}$) as the sequence $\mathbf{y}$ can be reconstructed at any interpolated point. However it suggests that $\mathbf{y}$ may be pitch dependant, requiring separate VQ entries to represent essentially the same spectral information from different $F0$ frames, thus increasing the overall bit rate for a given speech quality.

With low $F0$ males, we note $\mathbf{y}$ is also carrying information about the bandwidth of the peak that is not present for high $F0$ speakers. Narrow bandwidth peaks (in particular F1), have been shown experimentally to be important for the perception of male speech. As the bandwidth of F1 broadens, the speech becomes muffled.

Thus with low $F0$ speakers, this also suggests a pitch dependency in the spectral envelope $\mathbf{y}$, and corresponding inefficiency in quantisation.

This pitch dependency explains several results in Codec 2 and the wider literature:

1. Neural speech synthesis use a feature set equivalent to a small N_b, small K $\mathbf{b}$ vectors, and produce very high quality speech. When attempts were made to synthesise speech directly from this feature set using a sinusoidal model, the result was muffled speech. However a Neural model has the ability to use $F0$ as an input feature:

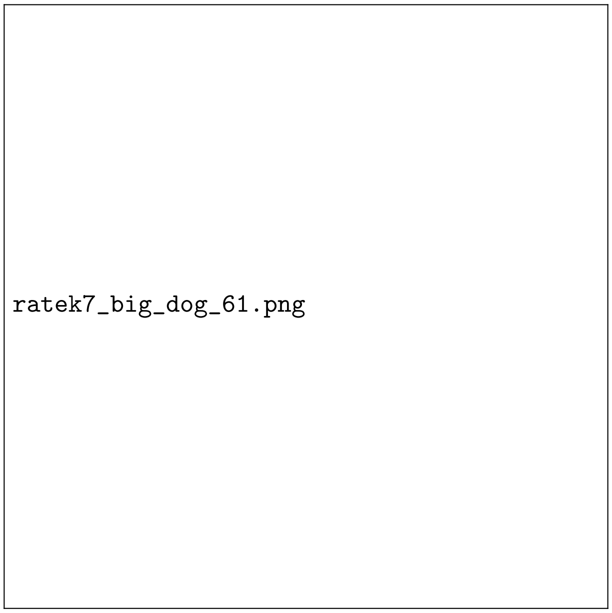
   $$\hat{\mathbf{y}} = F(F0, \mathbf{b}) \tag{26}$$

   and thus forms narrow bandwidth formants in the model output $\hat{\mathbf{y}}$ during low $F0$ speech. The function $F()$ is typically non-linear, so the peak/formant ratio of $\mathbf{y}$ can be enhanced on reconstruction in a way not possible with a linear resampler, overcoming the smoothing effect of low N_b filtering.

2. The Codec 2 *newamp1* algorithm used in Codec 2 700 uses an experimentally derived postfilter that has a large impact in the quality of male speech. It works by expanding the dynamic range of the rate K spectra (multiplying the rate K samples by a constant, then re-normalising to keep energy constant). The peak harmonics become higher than adjacent harmonics, effectively narrowing the bandwidth of peaks.

3. The group delay of a harmonic in the centre of narrow bandwidth filter will be larger compared to adjacent harmonics outside of the peak. If F1 is narrow it will "ring" for longer, and the time domain waveform will not decay completely between pitch periods. This increases the Peak to Average Power Ratio (PAPR) of the synthesised speech, makes it sound less buzzy, and closer to the input speech waveform.

Figure 15: $\mathbf{y}$ for frame 61 of *big_dog* (high L male), note narrow peak at F1



3.7 First Pass VQ Listening tests

Initial listening tests were conducted using a single male (*big_dog*) and female (*two_lines*) samples from inside the training dataset and evaluated using headphones. Samples synthesised with original amplitudes and codec 2 3200 were used as controls. The *phase0* algorithm was used for all samples to synthesise phases. The female sample sounded quite good under quantisation, however the male sounded buzzy and muffled.

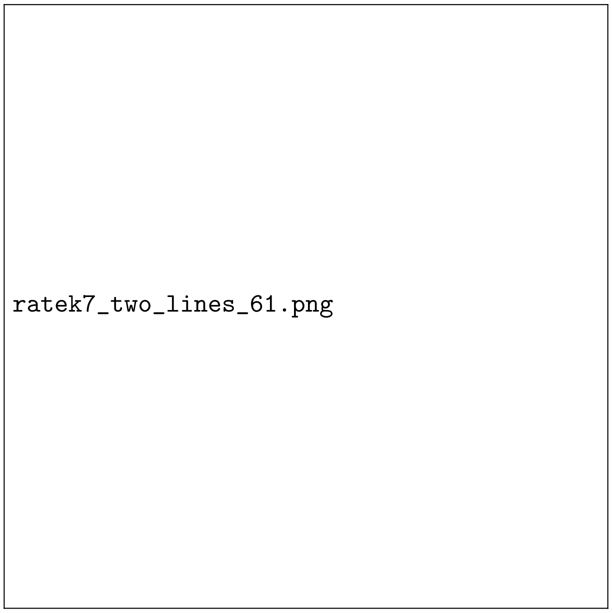
Key observations:

1. A low frequency artefact, possibly due to reconstruction errors beneath $warp(1, K)$ Hz. These appeared to go away when a high pass filter was applied to the synthesised speech.

2. Muffled/buzzy, even without vector quantisation, e.g. just filtered and resampled $\hat{\mathbf{y}}^L = S(\mathbf{b})$. This suggests filtering alone is causing problems, but only for males.

3. Codec 2 3200 sounds fine, this uses a LPC/LSP representation that can accurately model narrow bandwidth formants at all frequencies due to the high bit rate.

Figure 16: $\mathbf{y}$ for frame 61 of *two_lines* (low L female), note low frequency peak of $\mathbf{y}$ is not at the same frequency as the spectrum.



4. Application of the *phase0* model with the filtered amplitudes ($\hat{\mathbf{y}}^L = S(\mathbf{b})$) is worse than original phases and the filtered amplitudes. Use of original phases removes some of the muffled/buzzy effect. This effect has been observed before.

5. The female sounds fine.

Theory: For low $F0$ speakers, accurate representation of formant bandwidth is important at all frequencies, despite the ear having reduced frequency resolution at high frequencies.

Discussion:

1. The filtering $H()$ broadens high frequency spectral peaks, causing poor quality speech for males. An example can be seen in Figure **??**, near 2000 and 3000 Hz.

2. A physiological reason based on loudness perception is given in [**?**, p 62]. For tone pulses of variable duration (2-20ms) but *equal energy*, the probability of detection by a listener is proportional to the tone duration. This suggests [**?**] the distribution in time of the energy across a pitch cycle is important.

3. This effect can be interpreted in terms of the Peak to Average Power Ratio (PAPR) of the speech waveform. For a fixed total energy, the ear prefers speech waveforms with a low PAPR. This has parallels with amplitude compression algorithms such as those for sound recording and analog radio modulation, which also seek to lower the PAPR in order to improve perceptual loudness and intelligibility in high background noise use cases. Several authors [**?**] [**?**] have explored the problem of buzzy and muffled synthesised speech for low $F0$ speakers and noted peakiness (higher PAPR) in the time domain envelope. In the limiting case where all of the pitch cycle energy is concentrated in an impulse (zero phase speech) [**?**] reported robotic sound quality and decreased intelligibility scores.

4. The theory explains the good results for Codec 2 3200 on the same male. Due to the high bit rate of this mode we can accurately represent narrow bandwidth peaks at any frequency. It also explains why the speech quality declines with the lower rate Codec 2 modes that employ LPC/LSPs. They lack the bits to represent the bandwidth of all spectral peaks.

5. As discussed above, representing narrow peaks for low $F0$ male speakers at many frequencies has the potential for VQ inefficiency and an increased bit rate. A method needs to be found that can preserve the bandwidth of peaks, and still employ a non-linear frequency axis for efficient encoding of the formant centre frequencies.

6. Intuitively, we should be able to represent the speech spectrum with the same number of bits for females as for males.

7. The high quality of speech synthesised from neural networks with low frequency resolution cepstral vectors suggests it is possible to infer or regenerate narrow bandwidth peaks from simpler, lower bit rate (lower N_b) spectral representations. This could be performed via a post filtering algorithm or by training a neural net.

8. If we use the current low N_b filtering, the post filter should be frequency dependant, only narrowing the bandwidth of high frequency peaks.

Table **??** summarises the speech quality for a combination of rate K amplitude modelling (*newamp* algorithm) and phase synthesised using the *phase0* model.

	Original Phase	Synthesised Phase
Original Amplitudes	Good	Good
Rate K amplitudes	Good	Poor

Table 4: Subjective speech quality for a low pitch male sample with Rate K amplitudes and synthesised phase

The *phase0* algorithm fits a minimum phase model to the spectral amplitudes in order to synthesise phase spectra at the decoder. Near spectral peaks this can be approximated by a second order all pole system with time domain impulse response:

$$x(n) = Ae^{-\alpha n} cos(\omega n) \qquad (27)$$

α is the time constant that controls the decay of the time domain envelope. For a wide bandwidth peak α will be large, $x(n)$ will decay quickly and the formant energy will be contained in a shorter period of time. For a narrow bandwidth peak α will be small and $x(n)$ will decay slowly, the format will ”ring” longer and the PAPR will be smaller.

In the frequency domain a small α system results in a rapid phase shift near the peak frequency. If the system is excited by an impulse train (harmonic frequency spectra), a small α results in adjacent harmonics having different phases, and not being aligned in time, shifting time domain energy away from a central peak at the pitch pulse onset.

Group delay is a measure of how long it takes for a tone burst to propagate through a system. In a minimum phase system, group delay is related to the slope of the frequency response [**?**]. For a small α (sharper peaks) group delay will be longer, delaying the attack and decay time of the response (mainly energy at the formant centre frequency) to the impulse exciting the system.

This model and the theory above can be used to explain the results in Table **??**. With original amplitude we have narrow peaks so the *phase0* algorithm produces a small-α $x(n)$ that rings for some time and sounds natural. With rate K amplitudes and original phase, we have a phase spectra from the original speech which despite a broad amplitude peak, results in a time domain signal with energy shifted in time from the pitch pulse onset.

With rate K amplitudes and the *phase0* algorithm we have a wide bandwidth amplitude spectra peak that has a small slope near the peak. Recalling that for minimum phase systems (such as that employed by the *phase0* algorithm), phase shift and group delay are proportional to the slope of the amplitude spectra. Hence adjacent harmonics have small phase shifts, concentrating the energy in time. Viewed in the time domain, a wide bandwidth means a large-α $x(n)$ again with energy concentrated near the start of the impulse.

The theory also explains why high $F0$ speakers sound better. Even for broad spectral peaks, the pitch periods are so short that they run together before having a chance to decay. The ear experiences continuous tones bursts aiding perception of the formant energy.

Possible ways forward:

1. Raise Nb until males sound OK, design VQ and see how well it works. May be inefficient, or there may be some correlations across males that VQ can find.

2. Perform filtering on pre-emphasised speech. Spectral valley energy at frequencies just lower than a high frequency peak can sometimes be higher than the spectral peak due to the slope of the speech spectra. This can causes reduced spectral peak/valley ratios after filtering. For an example see the spectral peak of Figure **??** around 2000 Hz.

3. A freq-dep postfilter. Need to prove we can narrow formants for males, and enhance peak/valley ratio.

4. A NN is one solution - sharp peaks for males (based on F0), and enhance peaks/valley ratio at high frequencies. What would we use as ground truth? Is there a toy example we can train against? How can we make it handle wide dynamic ranges? How do we know we have enough training data? Need to see it enhancing indep of frequency of formant - that would be a good test. Just train on voiced, high energy frames.

3.8 Post Filter Design

We require a function $P()$ that modifies a smoothed vector of spectral magnitude samples $\mathbf{y}$ to sharpen formant peaks and increase peak/valley ratio, in particular at high frequencies.

The filter $P()$ is needed for low $F0$ speakers and voiced speech.

The smoothed vector $\mathbf{y}$ is obtained by filtering $\mathbf{y} = H(\mathbf{x})$, where $H()$ is a filter function defined by the parameter N_b that provides progressively more smoothing as frequency increases. This smoothing reduces the information in $\mathbf{y}$ making it possible to quantise at low bit rates.

Representing the sharp peaks required for male speech is possible by direct quantisation of $\mathbf{x}$ but requires a prohibitively high bit rate for our use case. High $F0$ speakers can be adequately represented by a low N_b vector which suggests there should be a way to reproduce male speech of adequate quality from low N_b vectors. $P()$ should have no effect on high $F0$ speakers, which are already adequately represented by the smoothed spectral samples.

3.9 Group Delay Interpretation

Explanation of why ratek with original phases works. At high frequency, the bandwidth of the spectral peak is broadened. Original phases means the group delay $\tau_g = \frac{d\phi}{d\omega}$ will remain the same for the central harmonic(s) that were part of the original peak. High group delay means a slow response to amplitude envelope changes, in our case an impulse train exciting the system. Harmonic energy in the peak centre will be smeared in time (ringing), lowering PAPR, and having the same probability of detection by the ear as the original waveform.

Adjacent, newly high amplitude peaks have a similar amplitude to the peak but having been outside of the original peak, may have lower group delay when using the original phase spectra. They will therefore decay quickly and not be detected.

The mixture of high group delay and low group delay harmonics may have the same perceptual loudness as with original amplitudes, as the amplitude of the central harmonic(s) is unchanged by broadening the peak, and the adjacent harmonics are not detected due to low group delay.

Another interpretation. With broadened peaks but original phases, we now have a non-minimum phase system as phase response is not related to magnitude response by a Hilbert Transform. This implies the energy is not maximally concentrated in time and group delay is not minimised.

This suggests a post filter design (and indeed explanation of current post filters). Enhancing the spectral envelope by expanding the dynamic range of the envelope $\hat{\mathbf{y}} = \beta\mathbf{y}$ ($\mathbf{y}$ in dB) will increase the slope of the edges of the spectral peak. If we derive a phase spectra using a HT, the group delay will be large at the peak edges, but small in the centre. The amplitudes at the edges will be similar to the central peak. Thus each edge will contribute low PAPR harmonics with increased probability of detection. It remains to be seen if this will will be perceived in the same way as a single peak in the centre of the harmonic, given the poor ability of the ear to separate closely spaced tones at high frequencies.

3.10 Post filter

A post filter has been developed, and is providing a reasonable improvement in speech quality over a small database of speech samples from the *codec2* repository, in particular when a small loudspeaker is used for playback. The post filter reduces the buzzy/muffled effect for low pitch males, and importantly does not reduce the quality of other samples (such as females). It improves intelligibility of several samples, even those with high pitch. It does ”colour” the speech - samples sound different compared to no post filtering, however the overall effect is positive.

The *newamp1* post filter employed in Codec 2 is based on the MBE frequency domain post filter [**?**, Section 8.6, p 267], which is turn based on the frequency domain post filter from McAulay and Quatieri [**?**, Section 4.3, p 148].

Figure **??** illustrates the algorithm in action in the time domain, and Figure **??** shows the effect in the frequency domain. Compared to the *phase0* algorithm alone, the algorithm better distributes the energy of each pitch cycle in time, which can be objectively measured by a reduction in PAPR.

We note that the post filtered signal envelope is more symmetric in time than the original which shows the classic damped exponential envelope. This may not matter to the ear, as our theory suggests it is the PAPR that matters, not the shape of the time domain envelope. In high pitched speech the pitch cycles run together without appreciable envelope decay, so the time domain envelope is nearly constant.

The algorithm acts on both the amplitudes $\mathbf{y}$, and phases $\mathbf{\Omega}$ in two stages. For convenience it currently operates at rate L_{high}, and was developed using unquantised vectors. The amplitude stage:

1. Makes a straight line least squares fit to the amplitude samples $\mathbf{y}$ (in dB) between 200 and 3700 Hz to estimate the spectral slope $\mathbf{s}$.

2. Subtracts the spectral slope to operate on an equalised vector of amplitudes.

3. Multiples each equalised vector sample by a weighting vector $\mathbf{w}$ sampled at the same rate as $\mathbf{y}$ that varies linearly from 1.2 to 2.0 over 0 to 4 kHz,

such that the elements $w(f) = f*(2.0-1.2)/4000+1.2$, to compensate for the effect of the filtering operation $H()$ which tends to reduce the dynamic range at high frequencies.

4. Restores the spectral slope.
5. Scales the energy in post filtered vector to be the same as the original vector.

$$\begin{aligned}\mathbf{y}' &= (\mathbf{y}-\mathbf{s}) \odot \mathbf{w} + \mathbf{s} \\ \hat{\mathbf{y}} &= \mathbf{y}' + 10\log_{10}(||\mathbf{y}||^2/||\mathbf{y}'||^2)\end{aligned} \tag{28}$$

where $\odot$ represents element by element multiplication, all vectors are in dB, and the operation $||\mathbf{x}||^2$ calculates the total *linear* energy in the vector $\mathbf{x}$.

Using informal listening tests during development it was found that too much post filtering in the amplitude stage (especially at low frequencies) resulted in hollow, unnatural sounding speech. Therefore a second stage that affects just the harmonic phases was introduced. The second stage weights the input to the Hilbert Transform used to calculate the phase spectra:

$$\mathbf{\Omega} = HT(1.5*\hat{\mathbf{y}}) \tag{29}$$

where $HT()$ is a the Hilbert Transform function that returns a vector of phase samples $\mathbf{\Omega}$ from a vector of amplitude samples $\mathbf{y}$. This was found to provide a further dispersion of the time domain energy, without additional modification of the magnitude spectrum.

The `plphase2.m` Octave script was used for development of the post filter described in this section. It supports frame-by-frame single stepping through speech samples, and switching various option in and out while plotting the signal in the time, frequency, and group delay domains. The `ratek2_batch.m` Octave script (*ratek2_model_postfilter* function) provides batch processing of speech samples, and the top level `ratek_resampler.sh` shell script processes multiples speech files using various options, generating wave files for subjective listening tests. *hts1a* frame 44 and *big_dog* frame 61 are two test frames useful for visualising the problem, and effect of the post filter.

Inspecting *hts1a* with `plphase2.m` provides a good example of the effect of the post filtering on group delay. This low pitch sample has frames where a single harmonic defines formants, demonstrating just how narrow male formant bandwidths can be.

Complex PAPR (CPAPR) is defined as:

$$\begin{aligned}CPAPR &= 10\log_{10}\frac{\max|s(n)|^2}{\text{mean}|s(n)|^2} \\ s(n) &= \sum_{m=1}^{L} A_m e^{j(\omega_0 mn+\theta_m)}\end{aligned} \tag{30}$$

CPAPR is sensitive to the time domain magnitude envelope of the synthesised speech signal.

3.11 Post Filter Further Work

We now have a better understanding of how frequency domain post filters such as those used in the *newamp1* algorithm improve speech quality:

1. Increasing the formant/anti-formant ratio better defines the formant locations for the ear, increasing intelligibility. This helps improve speech quality for males and females.

2. For low pitch males, the post filter distributes the time domain speech energy more equally across the pitch period than the minimum phase *phase0* model alone, increasing the probability of detection of formant energy by the ear, reducing muffling and increasing intelligibility.

The post filter algorithm presented here was developed by experiment, and there is much room for further development. The weights and partition between the amplitude and phase stages were chosen by experiment. Too much filtering in the amplitude stage (especially at low frequencies) resulted in hollow sounding speech.

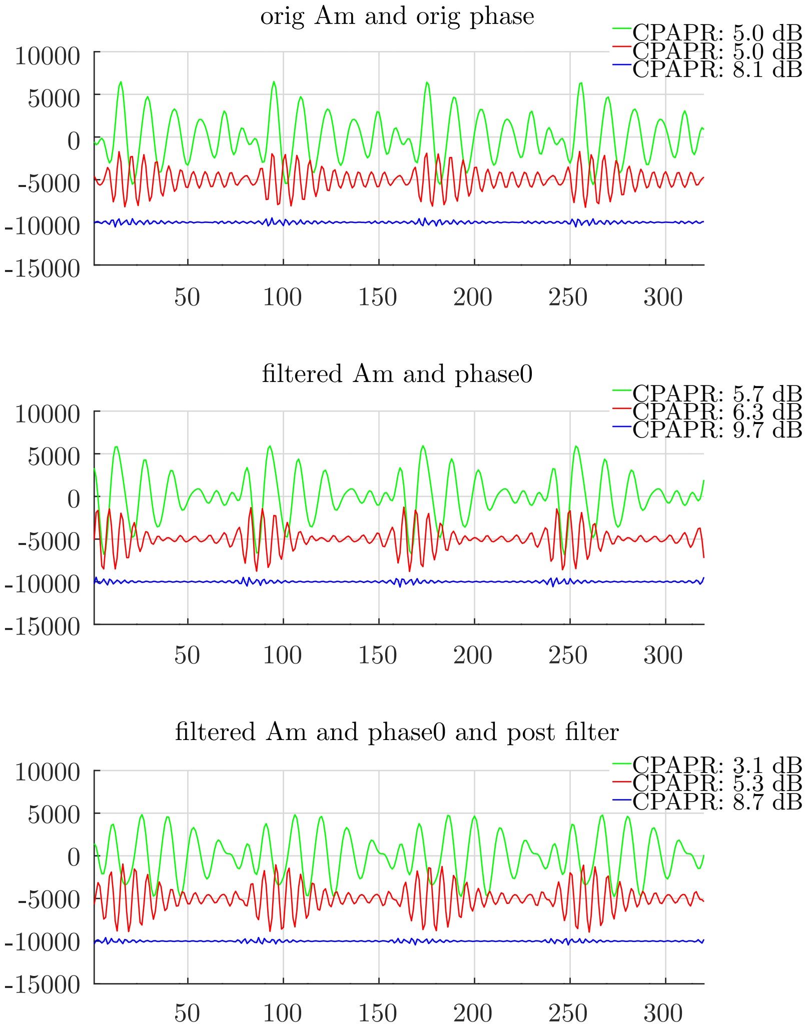
The slope removal/equalisation phase deserves further examination. In the *newamp1* post filter, $\mathbf{s}$ was set to a fixed 6dB. Others [**?**, Section 4.3, p 148] have fit a first order all pole model.

The accuracy of the fit of $\mathbf{y}$ to $\mathbf{a}$ varies from frame to frame, so the amount of post filtering required to obtain a distribution of energy in time similar to the original varies. A more formal definition of energy distribution could lead to an algorithm to better calculate a phase spectra.

For low frequency harmonics, it is important to have phase continuity between frames to prevent roughness. The algorithm does not currently track or regulate phase continuity between adjacent frames.

The question of where to apply the post filter (e.g. before or after quantisation) is open. Equalisation of the speech spectra early in the signal processing chain would likely lead to lower VQ distortion, if the effect on speech quality is acceptable.

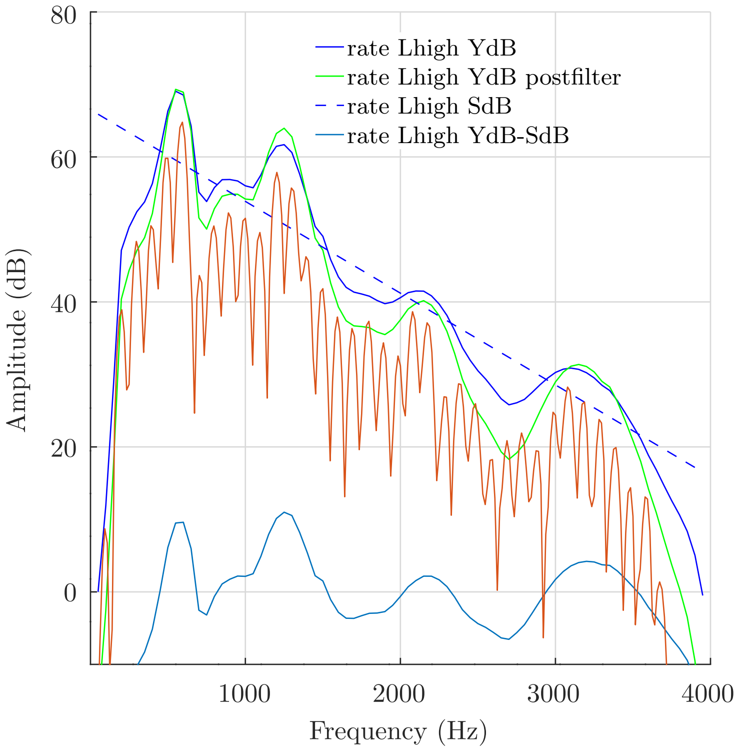
Figure 17: Time domain: 40ms centred on big_dog frame 61; green 0-1k, red 1-2, blue 2-4 kHz. Note middle subplot has energy more concentrated in time compared to top. With the post filter applied (bottom), energy is less concentrated


orig Am and orig phase

filtered Am and phase0

filtered Am and phase0 and post filter

Figure 18: Frequency domain view of post filter: note post filter effect increases with frequency. Also shown is the least squares spectral slope estimate, and equalised vector (bottom)



4 Vector Quantiser Experiments

A range of VQ designs were explored, strongly influenced by the excellent introduction presented in [?]. Using *train.spc*, the distortion of each VQ E_q in dB^2 for a range of bits was plotted. The Makefile `ratek_resampler.mk` was used to automate the process. Figure **??** presents a range of experimental VQ results:

- All of the VQ designs are "mean removed"; we remove the mean of the vector and transmit this separately as side information. This results in some loss of VQ efficiency but reduces storage and search complexity. As further work it would be interesting to explore alternatives to this design choice, especially when combined with dynamic range compression.

- The $K = 20$ *scalar* line shows the performance of scalar quantiser at the same bit rate. Scalar quantisers encode each VQ element independently and cannot take into account correlation between elements, making them generally less efficient. As the number of bits allocated to the VQ increases, correlation is removed and the slope starts to approach that of the scalar quantiser.

- The $4dB^2$ reference is a rough acceptability bar for communications quality speech, based on historical Codec 2 use cases.

- *lbg* is a two stage, 24 bit quantiser with $K = 30$. *k20* is similar but uses $K = 20$, and shows very similar performance with reduced storage requirements. The break in performance from stage1 to stage2 is clearly evident from the change in slope. [?] suggests this break is due to the loss of correlation in the stage 1 residual. Maximising the size of the first stage VQ is important for low bit rate speech coding.

- *sub* uses just a subset of the $K = 20$ elements, for example ignoring the first few and last few elements. This leads to a more efficient VQ at the expense of reduced speech quality as the LF and HF response is lost. Informal listening tests were not encouraging, the subjective quality lost some intelligibility.

- *split* uses two smaller single stage VQs e.g. $K_1 = 10$ and $K_2 = 10$ to quantise the $K = 20$ elements. The has lower performance, e.g. at $4dB^2$ *split* requires $2 \times 10 = 20$ bits compared to $12 + 3 = 15$ bits for *k20*, as the correlation between high and low frequencies is lost.

- *stf* is combined time and frequency VQ, where two vectors 20ms apart are combined and VQ-ed together, encoding a 40ms by $K = 20$ time-frequency vector. This attempts to gain VQ efficiencies due to correlations in time as well as frequency. To make the VQ size manageable it is split into two $K = 10$ by 40ms vectors. Results were disappointing, with no significant advantage compared to *k20* at a 20ms update rate. However this experiment was performed before energy normalisation (see below) was implemented, so as further work it would be useful to repeat this experiment.

For new modes, it was decided to proceed with a single stage 12 bit VQ, and 3 stage $3 \times 9 = 27$ bit VQ. The performance of these VQs is shown in Figure **??**. A 12 bit VQ at a 30ms update rate requires 400 bits/s, leaving 300 bits for the mean (energy) and pitch in a candidate 700 bits/s mode. The 27 bit VQ using 9 bits/s stage has modest storage and search complexity, and requires 900 bits/s at a 30ms update rate. Combined with 300 bits for the mean (energy) and pitch this VQ is a candidate for a 1200 bits/s mode. Improvements in quantisation efficiency allow the use of a 30ms update rate which will improve intelligibility compared to existing Codec 2 modes at these rates that use a 40ms update rate.

Figure 19: Vector Quantiser Distortion for a range of VQ designs

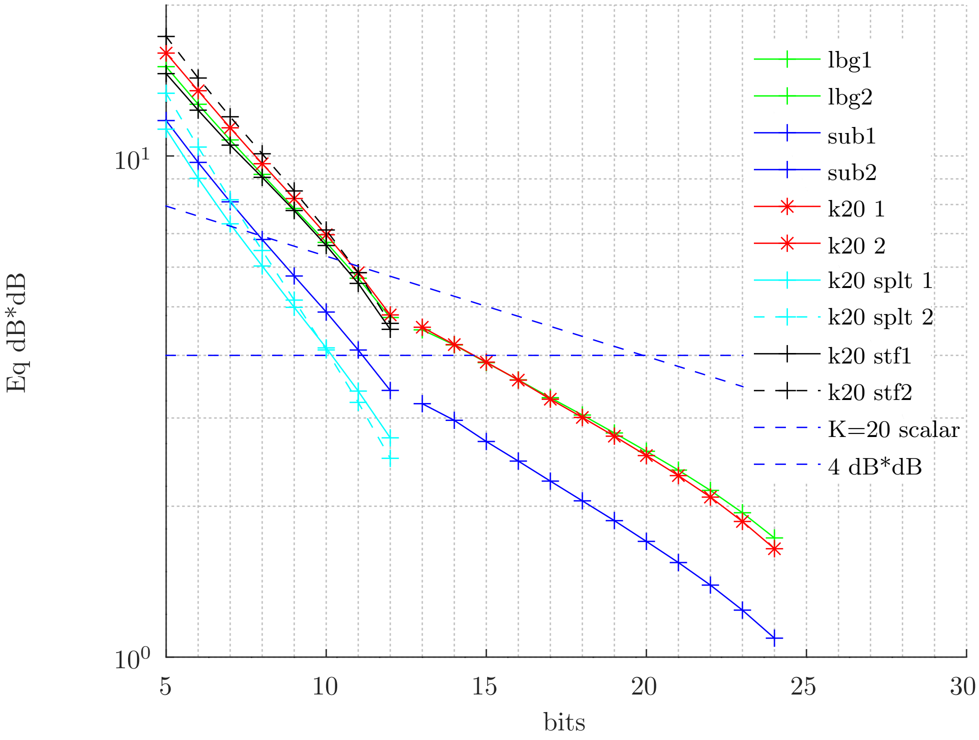
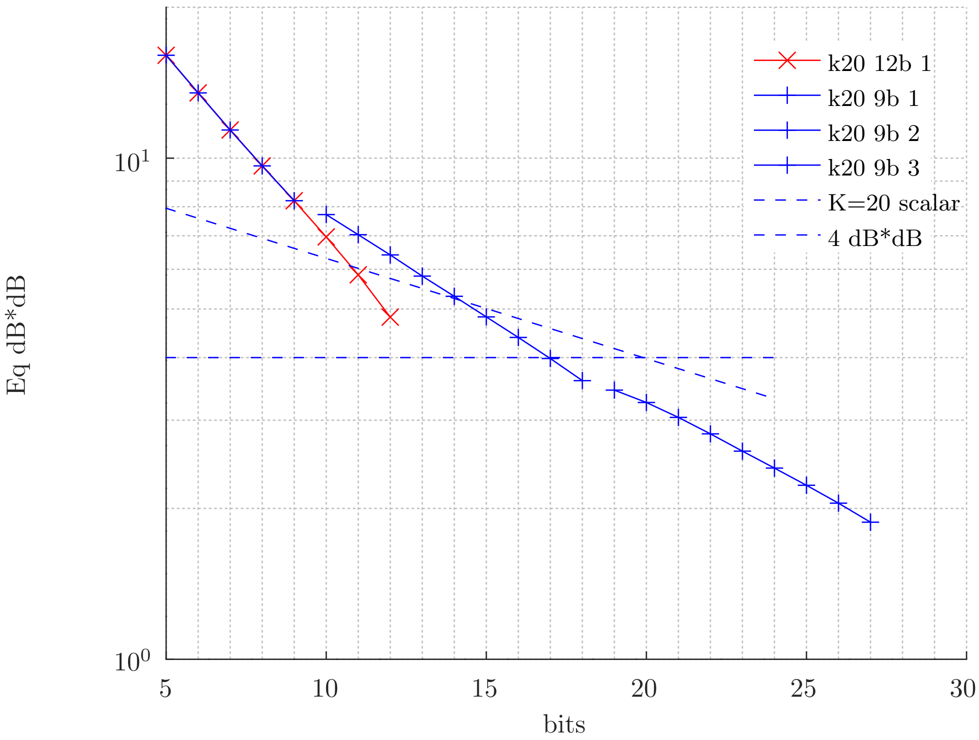


Figure 20: Vector Quantiser Distortion for 12 bit (700 bits/s mode) and 27 bit (1200 bits/s) candidates



5 Microphone Equaliser

The input speech may be subject to arbitrary filtering, for example due to the microphone frequency response, room acoustics, and anti-aliasing filter. This filtering is fixed or slowly time varying. The filtering biases the target vectors away from the VQ training material, resulting in significant additional distortion. The filtering does not greatly affect the input speech quality, however the VQ performance distortion increases and the output speech quality is reduced.

Consider the kth element of the error vector $\mathbf{e}^j$ for the jth target vector:

$$e^j_k = (t^j_k - f^j_k - g^j - v^j_k)^2 \tag{31}$$

where t^j_k is the target, f_k is an estimate of the filter, g^j the scalar gain (mean) added to the target, and v^j_k the chosen VQ entry. Solving for $\mathbf{f}^j$ is a highly non-linear problem, as g^j is a function of t_k^j and f_k^j; and v_k^j is a function of the preceding three variables. However as the function is differentiable it can be solved by gradient descent. We first find the derivative:

$$\frac{\partial e_k^j}{\partial f_k^j} = -2(t_k^j - f_k^j - g^j - v_k^j) \tag{32}$$

We then adjust f_k^j to move in the opposite direction to the gradient towards a (probably local) minima of $\mathbf{e}$:

$$f_k^{j+1} = f_k^j + 2\Delta(t_k^j - f_k^j - g^j - v_k^j) \tag{33}$$

where Δ is some small constant. The procedure them becomes:

- At iteration j, we have $\mathbf{t}^j$ and $\mathbf{f}^j$.
- Find $g^j = \frac{1}{K}\sum_{k=1}^{K}(t_k^j - f_k^j)$
- Search VQ to minimise:

$$E_i = |\mathbf{u} - \mathbf{v}_i|^2 \tag{34}$$

$$\mathbf{u} = \mathbf{t}^j - \mathbf{f}^j - g^j \tag{35}$$

  where $\mathbf{v}_i$ is the ith VQ entry. Let i_{min} be the VQ entry where E_i is minimised.

- Update the equaliser filter estimate:

$$\mathbf{f}^{j+1} = \mathbf{f}^j + 2\Delta(\mathbf{t}^j - \mathbf{f}^j - g^j - \mathbf{v}_{i_{min}}^j) \tag{36}$$

A unit test for this algorithm (known as *EQ2*) is in `tmic_eq.m`.

6 Energy Normalisation and Resampling

Resampling the frequency domain vectors (e.g. from rate L to rate K) is a common operation above. This section discuses how to ensure frame energy remains constant across the resampling operations. This is necessary to handle the following use cases:

- In unvoiced or transition frames the pitch estimator output $F0$ is undefined. However the energy of adjacent frames in the time or frequency domain will only vary slightly. If $F0$ is not smooth, it is still desirable for the energy of adjacent frames represented by $\mathbf{b}$ in frame f and $f+1$ to be similar.
- To reduce bit rate we often decimate $\mathbf{b}$ in time then interpolate at the decoder. For example we may attempt to reconstruct $\mathbf{b}_{f+1}$ by linear interpolation of $\mathbf{b}_f$ and $\mathbf{b}_{f+2}$. Despite $F0$ variations, we would like the energy of adjacent frames to be similar to facilitate smooth interpolation.

The initial procedure for resampling did not normalise energy, leading to the energy of $\mathbf{b}$ being $F0$ dependant. When synthesising unvoiced speech energy variations lead to audible clicks (especially when combined with pitch estimation errors during voiced speech or voicing errors when $F0$ was random).

Consider a sampling operation between rate L and rate K. We would like the energy e in the vector at both sample rates to be the same:

$$e = \sum_{m=1}^{L} |A_m|^2 = c \sum_{k=1}^{K} |B_k|^2 \tag{37}$$

Note here we assume A_m and B_k are linear (not dB). Solving for c:

$$c = \frac{\sum_{m=1}^{L} |A_m|^2}{\sum_{k=1}^{K} |B_k|^2} \tag{38}$$

The normalised rate K vector elements are then:

$$\hat{B}_m = \sqrt{c} B_m \tag{39}$$

So the procedure for resampling is:

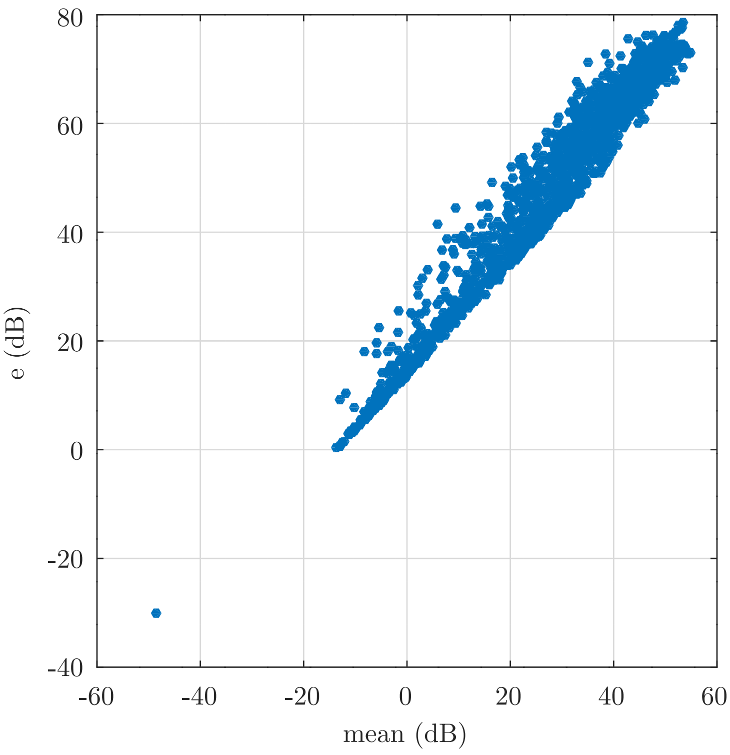
- Resample from $\mathbf{a}$ to $\mathbf{b}$ using a spline, or similar low distortion resampler.
- calculate c
- The output normalised rate K vector is $\hat{\mathbf{b}} = \sqrt(c)\mathbf{b}$
- When $\mathbf{b}$ is expressed in dB, a convenient result is $\hat{B}_{m\_dB} = 10log_{10}(c) + B_{m\_dB}$

This procedure should be used for any resampling, e.g. from rate L to rate K, or rate K to rate L, to ensure constant energy is preserved across all resampling operations. A unit test for this procedure is the function `test_norm_energy()` in the file `newamp_700c.m`.

7 Mean and Energy Quantisation

The current approach for vector quantising $\mathbf{b}$ is to remove the mean $\mu = \frac{1}{K}\sum_{k=1}^{K} B_k$, and send that separately as side information, along with the vector quantised, mean removed vector $\hat{\mathbf{c}} = Q(\mathbf{c}) = Q(\mathbf{b} - \mu)$. The mean is correlated with the frame energy, but they are not quite the same thing. Figure **??** shows that for a given energy e, a range of means is possible. Coarsely quantising μ resulted in unusual variations in peak amplitude in the synthesised speech at the decoder, as clipping μ to some maximum does not guarantee frame energy will be constant. We are interested in coarse quantisation of frame energy to reduce bit rate, and clipping to reduce dynamic range. It is therefore preferable to quantise and transmit frame energy, which can be converted to $\hat{\mu}$ at the decoder.

Figure 21: Scatter plot of **b** energy against mean for *train.spc*



The frame energy to be quantised and sent to the decoder is given by:

$$e = 10log_{10}\left(\sum_{k=1}^{K} 10^{(\hat{C_k}+\hat{\mu})/10}\right) \tag{40}$$

Given $\hat{e} = Q(e)$ and $\hat{\mathbf{c}}$ at the decoder we obtain $\hat{\mu}$:

$$\hat{\mu} = \hat{e} - 10log_{10}\left(\sum_{k=1}^{K} 10^{\hat{C_k}/10}\right) \tag{41}$$

$\hat{\mathbf{b}}$ can then be recovered with:

$$\hat{\mathbf{b}} = \hat{\mathbf{c}} + \hat{\mu} \tag{42}$$

A unit test for this procedure is the function `test_energy_to_mean_conversion()` in the file `newamp_700c.m`.

8 Miscellaneous Topics

8.1 Background Noise and Clicks

High level background noise is handled poorly by Codec 2 at present. The key issue is the *phase0* phase model, which can produce perceptually annoying clicks (impulses) in the presence of random input signals. With random signals, the pitch estimator produces random pitch estimates. Occasionally, the voicing estimator fails, and declares a noise like signal to be voiced. The combination of low $F0$, voiced synthesis, and flat spectra synthesises speech with clicks (all time domain energy for the frame concentrated in one impulse).

An example of a spurious impulse can be seen in Figure **??** around sample 4800 (most obvious in the bottom plot). This is an unvoiced segment where one frame with a low $F0$ estimate has accidentally been declared voiced.

The `postfilter.c` algorithm attempts to address this. This algorithm estimates the background noise level. When a harmonic is beneath the threshold it randomises the phase of the harmonic. It handles modest levels of background noise reasonably well. However this algorithm relies on a level threshold, above the threshold it can't distinguish background noise from speech. It therefore breaks down when at higher noise levels, resulting in harsh sounding background noise with clicks.

Possible solutions:

- Multiple voicing bits like MELP or MBE, so that the entire spectra is not declared voiced.
- A better phase synthesis model, as discussed in Section **??**. Codec 2 should never synthesise an impulse - they are not present in either voiced speech, unvoiced speech, or background noise signals.
- An algorithm to detect and suppress background noise signals, such that they are not transmitted to the decoder.

8.2 Pitch Estimator Errors

Some time was spent tracking down errors in the synthesised speech, and several impulse noises were tracked down to pitch estimator errors. A similar breakdown to background noise sometimes occurs during transition of voiced speech segments when the pitch estimator returns an incorrect estimate. The combination of low $F0$ and voiced speech decision causes a perceptually annoying time domain click. As these are high energy segments of speech, the error is particularly noticeable.

A possible solution is a pitch tracker that considers 1-2 few frames forward or backwards in time.

8.3 Hilbert Compression on Synthesis

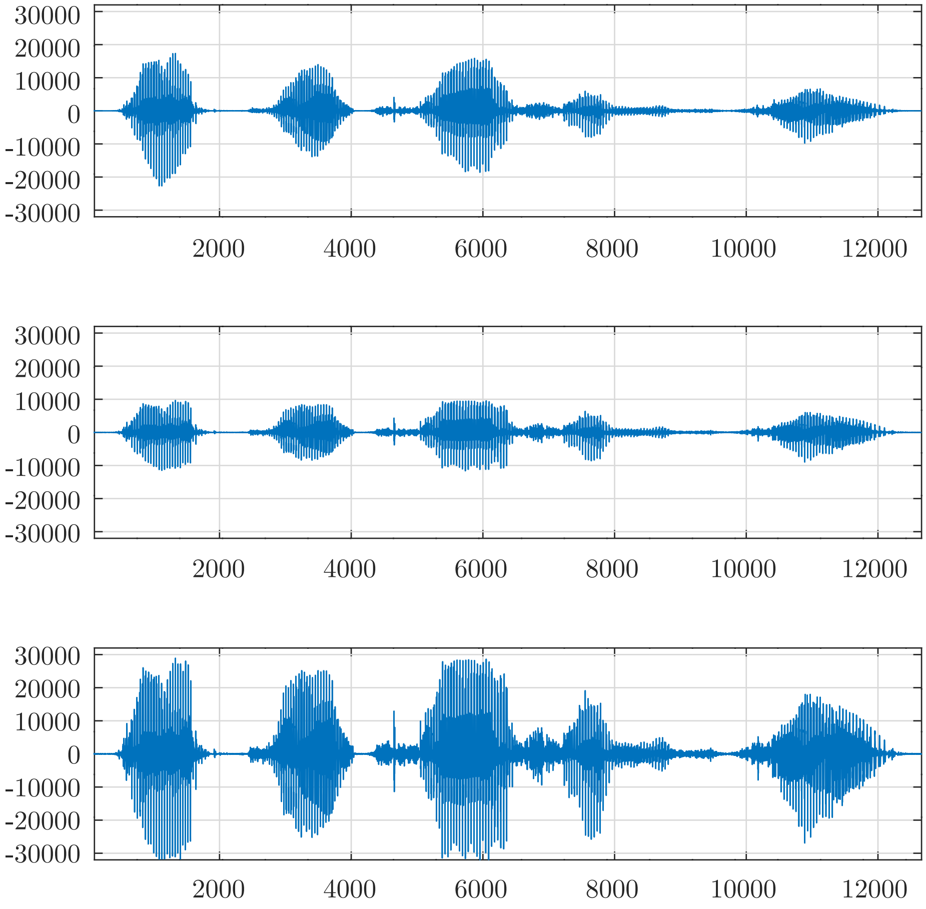
In SSB it is common to apply some compression to audio signals at the transmitter to maximise the RMS transmitted power and hence received SNR. Some FM transmitters also apply clipping to input audio to similarly increase average deviation and maximise the RMS audio level output from the discriminator.

A Hilbert Compressor (HC) was added to the `synthesise()` function in `sine.c`. Unlike SSB or FM, this implements compression at the **receiver**. The decoder has gain applied (e.g. 10dB), then time domain Hilbert clipping to limit the amplitude excursions of the time domain synthesised signal. This has several effects:

- The RMS level of the speech signal is increased, so it sounds louder. Many people consider ”louder is better”.
- Clipping distortion is minimised, as Hilbert Clipping (clipping the absolute value of the analytical version of the synthesised signal) reduces clipping distortion compared to a traditional clipper that operates on the real valued signal.
- It is a form of automatic gain control. Louder signals are limited in amplitude, so the ratio of softer to louder speech is reduced. This makes the Codec 2 voice link less fussy about mic gain settings. A little extra ”mic gain” will results in some relatively benign Hilbert Clipping.
- As the peak/RMS ratio is reduced, we use small speakers and audio amplifiers more efficiently. The speech sounds louder for a given speaker/audio power amplifier.
- The energy quantiser also compresses the speech dynamic range, as the maximum level of the energy quantiser tends to ”flat top” the speech waveform without undue distortion. It could be argued this is a form of frequency domain/frame based Hilbert Clipping. How the energy limiting compliments the time domain HC is a topic for future work (e.g. only one may be required).
- It is still possible to overdrive the clipper, and care must be taken such that the intelligibility of the already low fidelity signal is not further reduced. How best to use the HC is a useful topic for further work. One idea is make the compressor gain adjustable by the end user at the Rx station.

The Hilbert Compressor may also be useful for existing Codec 2 modes.

Figure 22: Effect of energy limiting and Hilbert Clipper when synthesising *forig* sample. Top is the vanilla synthesised speech, in the middle the energy limiting during 4 bit energy quantisation and at bottom we have applied 10dB gain and Hilbert Clipping. The bottom sample sounds much louder with no noticeable distortion. Also visible around sample 4800 is a spurious impulse (see Section **??**).



9 Summary of Software

In this section the software used for the R&D presented in this report is summarised. Further instructions can generally by found at the top of each script. The Octave scripts come in two flavours (a) frame-by-frame (fbf) that allow single stepping through samples while plotting various waveforms and vectors, and (b) ”batch” that process entire speech files, producing WAV files ready for informal listening tests.

Name	Description
`c2sim`	Codec 2 encoder and decoder that can exchange signal vectors with external scripts
`plphase2.m`	fbf experiment to explore original and phase0 synthesised phase combined with rate K amplitude model
`plphase3.m`	fbf script to explore issues with clicks in synthesised audio
`ratek3_batch.m`	batch processing script with many options for rate K resampling, postfilters and VQ
`ratek_resampler.sh`	Ties C and Octave code together to generate VQ training data, train VQs, and run various experiments
`ratek_resampler_plot.m`	Plotting curves of VQ performance from `ratek_resampler.mk`
`ratek_resampler.mk`	Generates VQ training data, trains various VQs, and generated plots of VQ performance
`newamp_700c.m`	Library of rate K processing functions and tests, code developed for this report at bottom
`tmic_eq.m`	Unit test for microphone equaliser
`ratek_plots.m`	Generates plots for this report

Table 5: Summary of software used for the work described in this report