Vowel Formant Track Normalization Using Discrete Cosine Transform Coefficients

Josef Fruehwald*
University of Kentucky
josef.fruehwald@uky.edu

Abstract This paper provides an overview of the Discrete Cosine Transform (DCT) as a method for smoothing vowel formant tracks, as well as a procedure to take any speaker normalization method that has been defined for formant point measurements and define an equivalent method to be applied directly to DCT coefficients. This procedure is followed for three established normalization methods, and the difference between DCT normalization and formant point normalization is found to be marginal.

keywords: vowel formants, speaker normalization, vowel intrinsic spectral change, discrete cosine transform

word count: 4933

1 Introduction

Some landmark research in North American dialectology and sociolinguistics of vowels has relied on single point measurements of F1 and F2 to characterize the entire vowel (Labov 2001; Labov et al. 2006). However, there is a wealth of work that makes it clear that full formant dynamics, or Vowel Intrinsic Spectral Change (Nearey and Assmann 1986), have important dialectal and sociolinguistic properties (e.g. Fox and Jacewicz 2009; Risdal and Kohn 2014; Docherty et al. 2015; Tanner et al. 2022). Additionally, vowel formant extraction tools such as the FAVE suite (Rosenfelder et al. 2024), fasttrack (Barreda 2021a; Fruehwald and Barreda 2024), and new-fave (Fruehwald 2025) make full formant tracks available to researchers, and curve fitting techniques are becoming more widely adopted (Sóskuthy 2021).

One open question about the use of full vowel formant tracks is how normalization for differences in vocal tract length between speakers ought to be carried out. There is an *extensive* literature on the topic in the use of point measurements (see Adank et al.

*I would like to thank Santiago Barreda, Kevin McGowan, Dan Villarreal, Jack Rechsteiner, the journal's anonymous reviewers, and the audience at NWAV 52 for feedback on this work.

2004), but how these methods should (or could) be applied to full formant tracks has not been addressed as fully.

In this paper, I will be exploring the use the Discrete Cosine Transform (DCT) to smooth and normalize formant tracks. The DCT was chosen, in part, because it is used in the fasttrack method to optimize the Linear Predictive Coding parameters in formant extraction, and the fasttrack python implementation allows users to save the DCT coefficients. I will demonstrate that it is possible to carry out speaker normalization *directly* on the DCT coefficients, which can then themselves be used for quantitative analysis.

The goal of this paper is not to compare normalization techniques and evaluate their benefits and drawbacks. Rather, it is to demonstrate that if we were to construe any given formant normalization technique as a function $f(x)$ that is applied to measured formant tracks, there is an approximately equivalent function $g(y)$ that can applied to the DCT coefficients of formant tracks. The specific procedure for defining $g(y)$ given $f(x)$ is as follows:

1. Any *non-linear* transformations (log, exponentiation, Mel, Bark, etc) must be applied to formant track data *before* the DCT is applied.
2. For *location* parameters (values added or subtracted in the normalization):
   - They should be estimated using the 0$^{\text{th}}$ DCT parameter.
   - They should be applied to just the 0$^{\text{th}}$ DCT parameter.
3. For *scale* parameters (values divided or multiplied by in the normalization):
   - They should be estimated using the 0$^{\text{th}}$ DCT parameter and multiplied by $\sqrt{2}$.[1]
   - They should be applied to all DCT parameters.

1.1 Notation conventions

There are a number of mathematical formulae in this paper and mathematical notation used throughout. I have chosen to follow the following notation conventions.

$\boldsymbol{F}$, $\boldsymbol{x}$
: A formant track. F is used when discussing formants specifically, and x when discussing a signal in general.

$\bar{\boldsymbol{F}}_{\boldsymbol{i}}$
: For a single vowel token, the average across its entire formant track.

$\boldsymbol{y}$
: A vector of Discrete Cosine Transform coefficients (described below).

[1] This $\sqrt{2}$ constant presumes the "orthogonalization" of the DCT. See Appendix C for more information.

i

A variable for formant numbers. For example F_1 is the first formant, and F_i is the i^{th} formant.

j

A variable for an observation number. For example, x_3 would be the third value from a signal x. This will sometimes be used interchangeably to refer to the j^{th} value in a signal, and sometimes to the j^{th} vowel token in a data set.

k

A variable for a Discrete Cosine Transform coefficient. For example y_0 will be the 0$^{\text{th}}$ coefficient, and y_k will be the k^{th} coefficient.

2 The Discrete Cosine Transform

The Discrete Cosine Transform (DCT) is a general purpose curve fitting method (it is the basis of jpeg image compression (Wallace 1992)). It is sometimes used directly on audio spectra (e.g. Zahorian and Jagharghi 1991; Zahorian and Jagharghi 1993; Guzik and Harrington 2007; Jannedy and Weirich 2017) and has also been used to model and classify vowel formant trajectories (Watson and Harrington 1999; Hillenbrand et al. 2001; Morrison 2009; Williams and Escudero 2014; Williams et al. 2015; Cox and Palethorpe 2019). Versions of the DCT are available in the fftw R package (Mersmann 2024) (which itself calls the fftw C library) and in the scipy python package (Virtanen et al. 2020). It has also been implemented in the emuR R package (Jochim et al. 2024) and the fasttrackpy python package (Fruehwald and Barreda 2024).

The DCT re–describes a continuous function in terms of weights on a bank of cosine functions oscillating at increasing frequencies. There are several definitions of the DCT; for the purpose of this paper, we'll be using the definition of the DCT as used in the python scipy library (Virtanen et al. 2020) both because this is a standard and replicable method, and because this is how the fasttrackpy library implements and saves these parameters (see Section A for more information).

The mathematical definition for the kth DCT is given in Equation 1.

$$y_k = \frac{1}{oN}\sum_{j=0}^{N-1} x_j \cos\left(\frac{\pi k(2j+1)}{2N}\right) \tag{1}$$

$$o = \begin{cases}\sqrt{2} \text{ for } & k = 0\\ 1 \quad \text{for } & k > 0\end{cases}$$

For the purpose of this paper, it's not necessary to understand every component of Equation 1 in detail. Figure 1 annotates the formula with the important pieces that will come into play. A signal x (in our case, a formant track) is multiplied by a cosine function of the same length, then summed. Then, this sum is normalized (divided by the length

of the signal), resulting in the DCT coefficient. The cosine function changes depending on the number of the DCT coefficient.

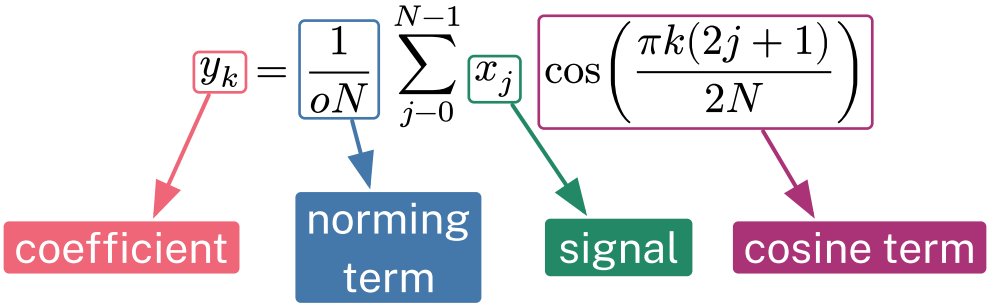

Figure 1: Annotated DCT Definition

Figure 2 plots the 0th through 2nd cosine functions that the coefficients calculated in Equation 1 weight.

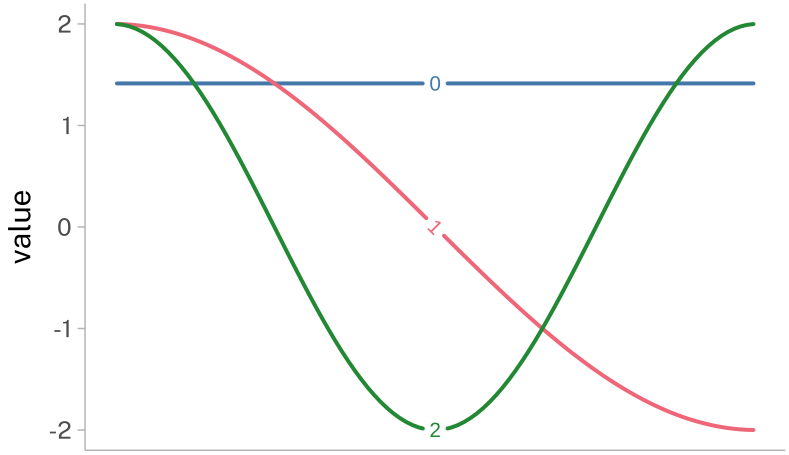

Figure 2: DCT basis functions

The inverse DCT takes the coefficients generated by Equation 1 and returns the original signal. The Inverse DCT equation is given in Equation 2.

$$x_j = \sqrt{2}y_0 + 2\sum_{k=1}^{N-1} y_k \cos\left(\frac{\pi k(2j+1)}{2N}\right) \tag{2}$$

To get the *rate* of change of a signal from its DCT coefficients, we need the first derivative of Equation 2, given in Equation 3

$$\frac{\delta x_j}{\delta j} = -2\sum_{k=1}^{N-1} y_k \frac{\pi k}{N} \sin\left(\frac{\pi k(2j+1)}{2N}\right) \tag{3}$$

To get the *acceleration* in a signal from its DCT coefficients, we need the second derivative of Equation 2, given in

$$\frac{\delta^2 x_j}{\delta j^2} = -2 \sum_{k=1}^{N-1} y_k \left(\frac{\pi k}{N}\right)^2 \cos\left(\frac{\pi k(2j+1)}{2N}\right) \tag{4}$$

3 Benefits of the DCT

There are a number of general benefits to using DCT coefficients for analyzing vowel formant tracks which I will outline here. It's worth noting at this point that several of the benefits only really hold for within-speaker analyses. In order to apply some of these methods to multi-speaker analyses, the DCT coefficients will need to be speaker normalized (the subject of the remainder of this paper).

3.1 Smoothing by truncation

The DCT returns the same number of coefficients as there are data points in the original signal. With all of these DCT coefficients, the original signal can be fully reconstructed by applying the inverse DCT. For example, the F1 formant track in Figure 3 consists of 63 measurement points (the plotted points), so the application of the DCT returns 63 DCT coefficients. When applying the *inverse* DCT to all 63 coefficients, the resulting curve (plotted in the red line) recreates the original signal.

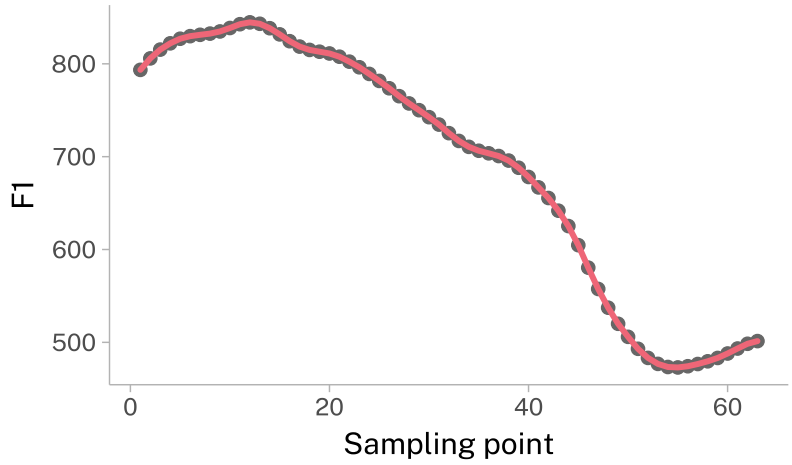

Figure 3: Recreation of the original signal by applying the inverse DCT.

The first 3 DCT coefficients have relatively interpretable meanings which can be seen in their shapes in Figure 2.

- 0th: The overall level of the curve
- 1st: The amount of fall to the curve
- 2nd: The amount of fall-rise to the curve

In fact, several papers have been successful classifying vowel phonemes using just these first three coefficients (Watson and Harrington 1999; Williams et al. 2015). Higher DCT coefficients capture higher frequency oscillations of a signal. For example, the cosine function that the 9th DCT coefficient weights is plotted in Figure 4.

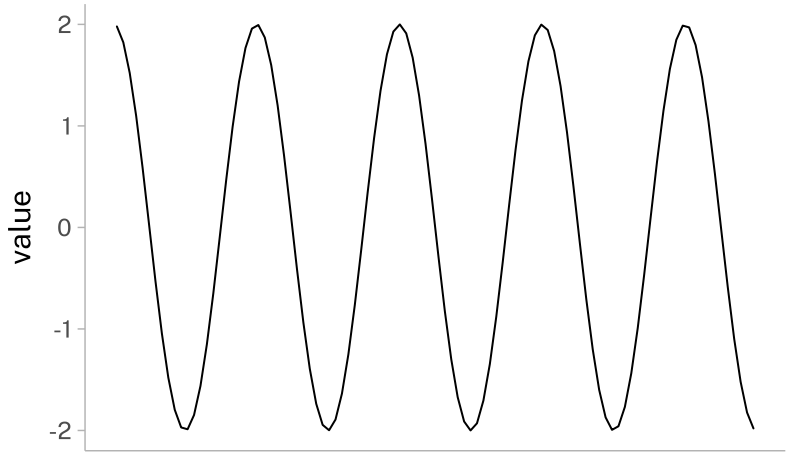

Figure 4: The 9th dct basis function.

While these higher frequency oscillations are important for perfectly recreating the original signal, they are probably not linguistically meaningful with respect to a single vowel's formant track. By truncating the number of DCT coefficients used to describe a formant track to the first 3 (as done in prior work) or 5 (as done in fasttrackpy), we get back a smoothed version of the formant track when applying the inverse DCT.

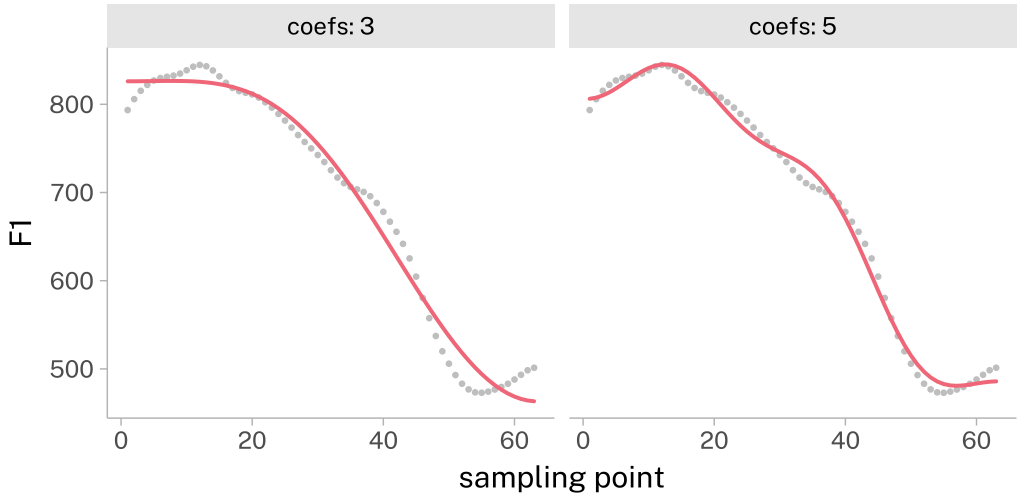

Figure 5: Greater smoothing achieved by using fewer high frequency coefficients

The benefit here is that the degree of desired smoothing can be arbitrarily specified *after* the DCT has been applied by simply using just the first n coefficients. This is in

contrast with some other commonly used curve fitting methods for vowel formants, like cubic regression splines (Sóskuthy 2017, 2021), where the number of basis functions (and therefore coefficients) must be specified *before* the curve is fit, and cannot be altered after the fact.

3.2 Data Compression

While this may be a diminishing concern given increasing computational power, by using just the first 5 DCT coefficients for analysis, instead of full formant track data, the volume of data that needs to be stored or loaded into memory for quantitative analysis is dramatically decreased. As shown in Table 1, the size of the data in the Philadelphia Neighborhood Corpus (PNC) (Labov and Rosenfelder 2011) is about an order of magnitude smaller when using DCT coefficients than when using the raw formant track data.

Table 1: Comparison of data size for formant track data from the Philadelphia Neighborhood Corpus. Both number of lines in comma separated files and total size on disk are shown.

	lines	size
tracks	46,973,521	17 GB
dct coefs	4,943,165	999 MB
ratio	9.5	16.5

3.3 Averaging Formant Tracks

Averaging over formant tracks of different durations is not straightforward, as they will have a different number of sampled values. Figure 6 illustrates this problem with three example formant tracks for the vowel /ay/ from one speaker in the Philadelphia Neighborhood Corpus. It would perhaps be inappropriate to average the final sample from a short token with the aligned midpoint sample from a longer token. Aligning the beginnings and ends tokens in, for example, proportional time, creates a new problem that the samples in between are no longer aligned, making point-wise averaging impossible.

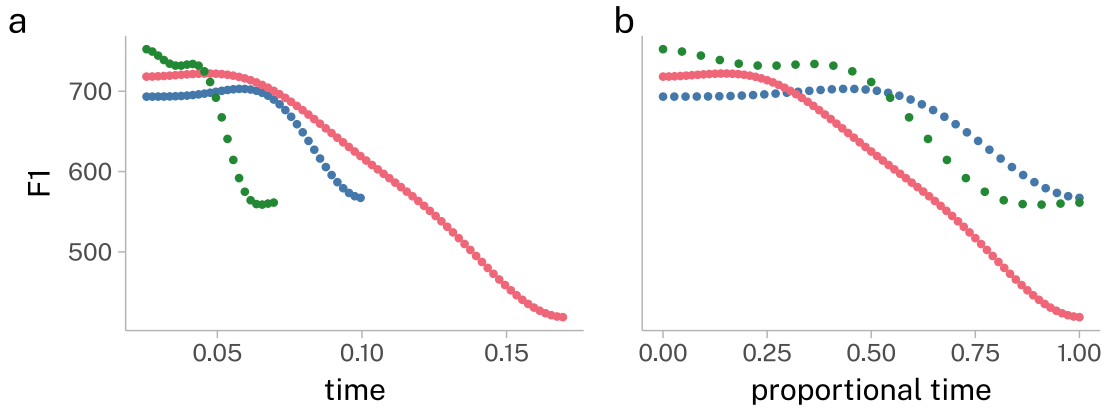

Figure 6: Three F1 formant tracks of different duration. a. Formant values against actual time. b. Formant values against proportional time

Instead of point-wise averaging, or fitting a smoothing model, we can instead average over the DCT coefficients for these three tokens (Table 2), then use these averages as DCT coefficients themselves to get the inverse DCT. The result of this averaging process is plotted in Figure 7.

Table 2: Averaging over DCT coefficients

	0	1	2	3	4
token					
1	471.3	26.0	−17.8	5.6	−0.3
2	428.3	72.6	−14.1	2.3	−4.5
3	473.6	51.0	−13.6	−3.2	7.3
average	457.7	49.9	−15.2	1.5	0.8

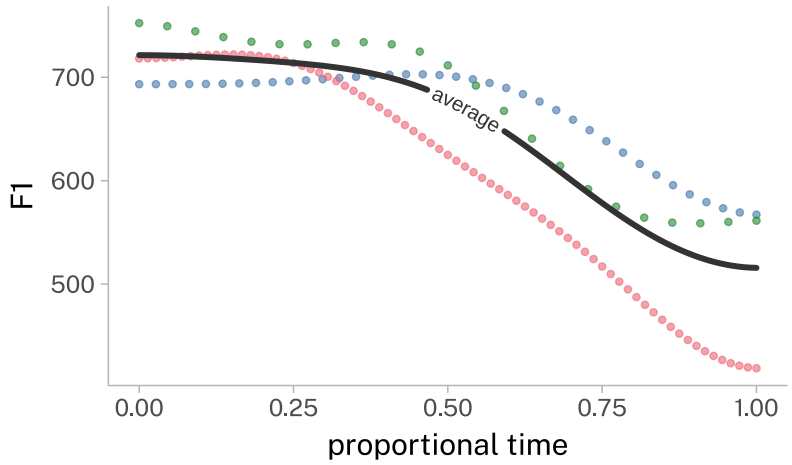

Figure 7: The result of the inverse DCT when applied to the average DCT coefficients of the three formant tracks.

3.4 Modelling with DCT coefficients

Similarly to how we can calculate averages more easily with DCT coefficients, we can also fit regression models more easily. Let's say, for example, we wanted to model the effect of following voicing on the entire F1 trajectory of /ay/ (plotted in Figure 8). To do this with the formant tracks themselves, we would need to fit a generalized additive model with an interaction between the smoothing term and voicing context, taking into account autocorrelation and, perhaps, random smooths by token (Sóskuthy 2021).

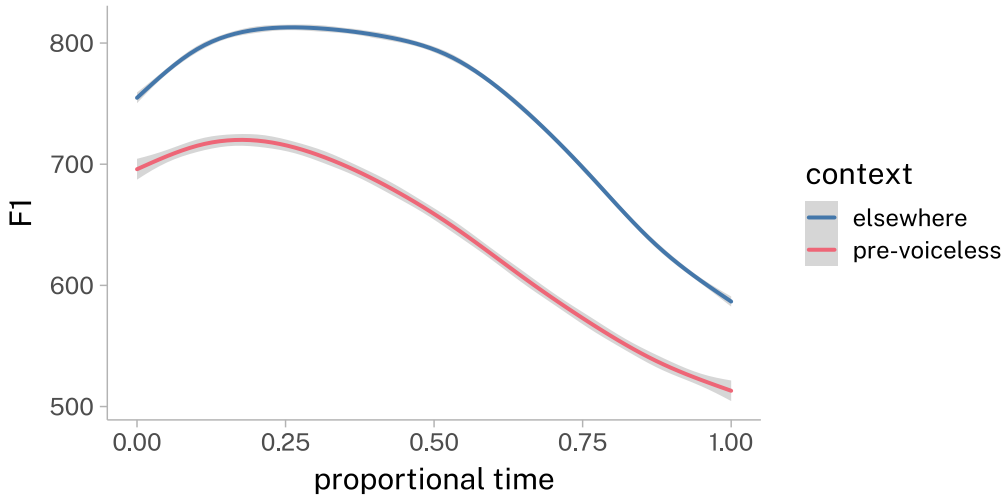

Figure 8: The effect of voicing on F1 formant tracks for one speaker.

With DCT coefficients, on the other hand, we could fit a linear model, using each individual DCT coefficient as the outcome, and the voicing effect as the predictor. This is a form of functional regression (Ramsay and Silverman 2006; Gubian et al. 2015).

Listing 1 shows the R code involved for fitting such a model, with the model results summarized in Table 3.

Listing 1: A model

```
ay_model <- lm(
  cbind(y0, y1, y2, y3, y4) ~ context,
  data = ay_data
)
```

Table 3: Regression coefficients over DCTs

	y0	y1	y2	y3	y4
(Intercept)	523.6	39.0	−21.6	1.5	−3.9
pre-voiceless	−72.5	5.1	13.1	−2.2	1.9

In Table 3, the intercept terms for each DCT coefficient correspond to the average of that coefficient across the reference level (the elsewhere context), and the pre-voiceless terms correspond to the *difference* of the pre-voiceless context from the reference level for each coefficient.

Interestingly, if we apply the inverse DCT to these difference terms, the result is the difference *curve* between the the two voicing contexts, plotted in Figure 9.

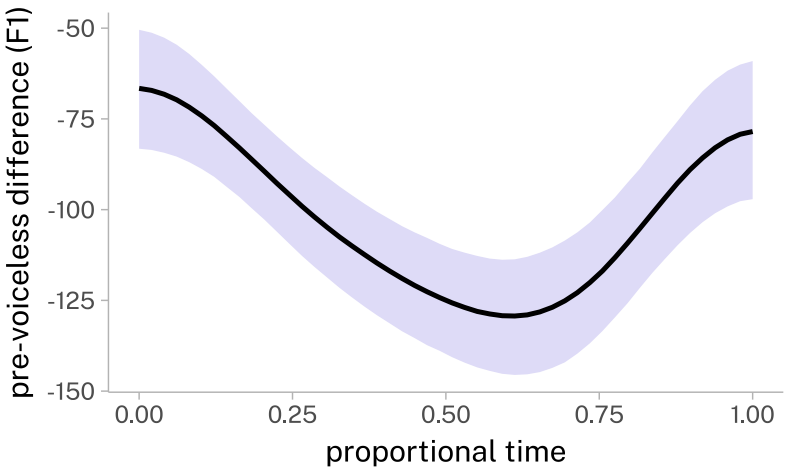

Figure 9: The fitted difference curve between pre-voiceless /ay/ and /ay/ elsewhere.

More complex models exploring the difference in formant dynamics between these two allophones could be specified, but by using DCT coefficients as the outcome variables, we don't need to include time across the formant track in our predictors.

3.5 Rate of change analysis

Given the definition of the first derivative of the signal in Equation 3, it's also possible to analyze the rate of change in formant dynamics. Figure 10 plots the predicted rate of change for these two allophones of /ay/ based on the model fit above. These rate of change curves were calculated directly from the DCT coefficients by plugging them into Equation 3. This kind of analysis could be useful in research on hypo/hyper-articulation, how coarticulation influences sound change, etc.

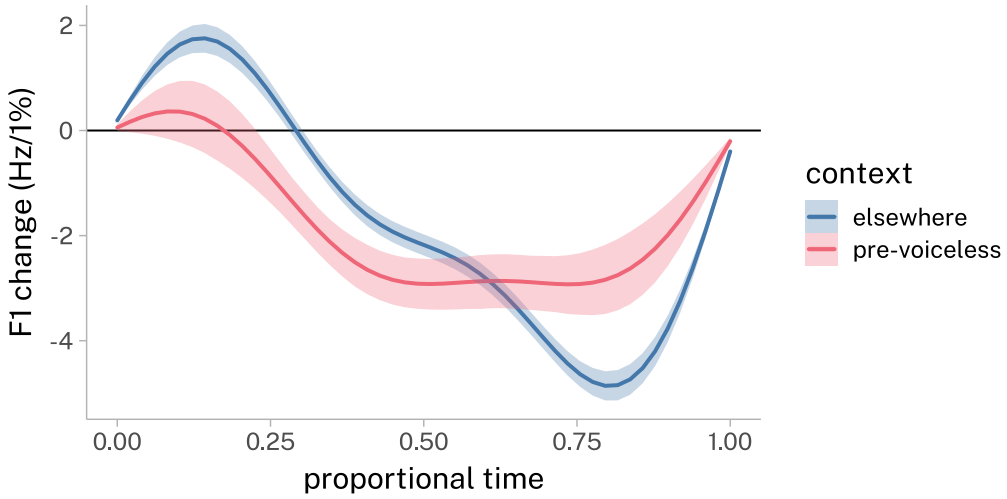

Figure 10: The rate of change of /ay/ allophones. The rate is in expressed in terms of Hz per 0.01 change in proportional time.

3.6 Multi-speaker analysis

The ability to regress over DCT coefficients could be invaluable for research into, for example, diachronic changes in formant dynamics. However this modeling approach, along with the other benefits of DCT coefficients discussed above, is really only well defined for within-speaker analyses. For example, Figure 11 plots formant trajectories for the vowels /iy/, /ey/, /ay/, /uw/, /ow/ and /aw/ from two speakers in the PNC. Specifically, DCT coefficients from each vowel class for each speaker were averaged, and then the inverse DCT applied to result in the plotted trajectories.

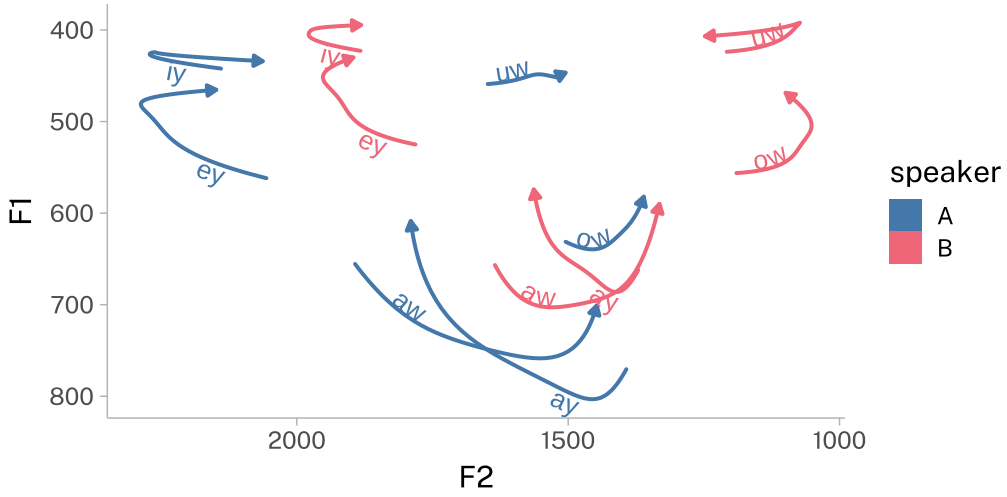

Figure 11: Average vowel trajectories from unnormalized DCT coefficients.

The consistent differences in vowel space size and location between these two speakers are also reflected in their DCT coefficients, such that any analysis combining their data will face the same problems as using unnormalized formant point measurements. For example, Speaker B's /iy/ and /ey/ appear to be backer than Speaker A's, but it's unclear how much of this is due to their overall smaller vowel space size versus having a different vowel space target. In order to conduct multi-speaker analyses using DCT coefficients, the DCT coefficients themselves will need to be normalized.

4 Transforming DCT Coefficients

Nearly all speaker normalization methods involve:

- Shifting the *location* of formant values by a constant l.
- *Scaling* formant values by a constant s.

$$F' = s(F + l) \tag{5}$$

If there are coefficients y_k that when the inverse DCT is applied to them return F, then there are coefficients y'_k that when the inverse DCT is applied to them return F'. The goal of this section is to identify whether and how it is possible to directly transform y_k to y'_k. To achieve this, we'll begin by identifying how shifting and scaling the input signal by our desired l and s affect the resulting DCT coefficients.

4.1 Useful Equivalencies

In order to simplify the notation in the following section, we'll define some equivalencies.[2] The cosine term in Equation 1 will be replaced with ψ_j, a function over the signal index j and the DCT coefficient index k.

[2] I would like to thank a reviewer for the suggested notation for this section.

$$\psi_j(k) = \cos\left(\frac{\pi k(2j+1)}{2N}\right) \tag{6}$$

The normalizing constant will be replaced with c.

$$c = \frac{1}{oN} \tag{7}$$

With these definitions, the DCT can be expressed as

$$y_k = c\sum_{j=0}^{N-1} x_j\psi_j(k) \tag{8}$$

There are some additional important equivalencies we'll leverage below. The first is for when $k = 0$.

$$1 = \psi_j(0) \tag{9}$$

Therefore, summing $\psi_j(0)$ over j will equal N.

$$N = \sum_{j=0}^{N-1} \psi_j(0) \tag{10}$$

Second, when $k > 0$, summing $\psi_j(k)$ over j will equal 0.

$$0 = \sum_{j=0}^{N-1} \psi_j(k); k > 0 \tag{11}$$

4.2 Scaling the input

Let's say we have a signal x that we applied the DCT to to get y_k.

$$y_k = c\sum_{j=0}^{N-1} x_j\psi_j(k) \tag{12}$$

If our normalization method called for scaling x by s, our input to the DCT would now be sx_j. The new DCT coefficients y'_k would be

$$y'_k = c\sum_{j=0}^{N-1} (sx_j)\psi_j(k) \tag{13}$$

As a constant, s can be brought outside the summation.

$$y'_k = sc\sum_{j=0}^{N-1} x_j\psi_j(k)$$

Now, everything to the right of s is the same as Equation 12, so we can substitute in y_k,

$$y'_k = sy_k \tag{14}$$

4.2.1 Scaling conclusion

The result here is that any scaling term applied to an input signal results in the same scaling term being applied to all of its DCT coefficients.

4.3 Shifting the input

Again, let's say we have a signal x that we applied the DCT to to get y_k. If our normalization method called for adding the constant l to x, our input to the DCT would now be $\big(x_j + l\big)$. The new DCT coefficients y'_k would now be

$$y'_k = c\sum_{j=0}^{N-1}\big(x_j + l\big)\psi_j(k) \tag{15}$$

The summation term can be rewritten as the addition of two summation terms.

$$y'_k = c\left(\sum_{j=0}^{N-1} x_j\psi_j(k) + \sum_{j=0}^{N-1} l\psi_j(k)\right) \tag{16}$$

Then, we can distribute the normalizing term c and move the constant l outside of its summation.

$$y'_k = c\sum_{j=0}^{N-1} x_j\psi_j(k) + lc\sum_{j=0}^{N-1}\psi_j(k)) \tag{17}$$

Everything to the left of the addition is the same as Equation 12, so we can substitute in y_k.

$$y'_k = y_k + lc\sum_{j=0}^{N-1}\psi_j(k) \tag{18}$$

We can now use the equivalencies in Equation 10 and Equation 11 to evaluate Equation 18 when $k = 0$ and when $k > 0$.

For $k = 0$, we'll substitute both the sum of $\psi_j(0)$ (Equation 10) and the normalizing constant c (Equation 7) to arrive at

$$y'_0 = y_0 + l\frac{1}{\sqrt{2}N}N = y_0 + \frac{l}{\sqrt{2}} \tag{19}$$

When $k > 0$, the sum of $\psi_j(k)$ is 0 (Equation 11), which eliminates the second term entirely.

$$y'_{k>0} = y_{k>0} + 0lc = y_{k>0} \tag{20}$$

4.3.1 Shifting conclusion

If an input signal is shifted by l, its 0th DCT coefficient will be shifted by $\frac{l}{\sqrt{2}}$, and its remaining DCT coefficients will remain the same.

4.4 Non-linear transforms of the input

Some speaker normalization procedures involve transformations of the signal that are non-linear. The most common one is taking the logarithm of formant values. If we logged an input signal x, its DCT coefficients would be given as

$$y'_k = c \sum_{j=0}^{N-1} \log(x_j)\psi_j(k) \tag{21}$$

There are no additional equivalences we can leverage here to re-express y'_k as a transformation of y_k. In order to calculate y'_k, we must directly transform the original signal, then apply the DCT.

4.5 DCT Coefficient Transformation Summary

In summary, if we want to shift our formant values by some value l and scale them by some value s:

- We need to add $\frac{l}{\sqrt{2}}$ to the 0th DCT coefficient (Equation 19).
- We don't add anything to the remaining DCT coefficients (Equation 20).
- We need to scale all DCT coefficients by s (Equation 14).

Additionally, for non-linear transformations of formant frequencies, there is no way to transform the DCT coefficients of the original values directly into DCT coefficients of the transformed values. Instead, the entire DCT needs to be applied to the transformed formants directly.

5 Estimating normalization parameters

Now that we understand how to shift DCT coefficients by l and scale them by s, we need to establish how to get these l and s normalization parameters from DCT coefficients. The two most common normalization parameters are the mean and standard deviation over vowels, so we will focus on those.

5.1 Calculating means

Let's begin with F as a formant track for a single vowel token, and y_k as the coefficients we get from applying the DCT to F. As we already established in Equation 9, $\psi_j(0) = 1$, which means the 0th DCT coefficient can be given as

$$y_0 = \frac{1}{\sqrt{2}}\frac{1}{N}\sum_{j=0}^{N-1} F_j \tag{22}$$

The sum of all values in a vector divided by the number of values in the vector is the mean of that vector, which we can represent with $\bar{F}$. So, the 0th coefficient is equal to the mean of F, divided by $\sqrt{2}$.

$$y_0 = \frac{1}{\sqrt{2}}\bar{F} \tag{23}$$

And therefore

$$\bar{F} = \sqrt{2}y_0 \tag{24}$$

The $\sqrt{2}$ constant will carry through any functions that are weighted sums. So if the function $f(x)$ is a weighted sum,

$$f(\bar{F}) = \sqrt{2}f(y_0) \tag{25}$$

The mean is a weighted sum, so

$$\text{mean }(\bar{F}) = \sqrt{2}\,\text{mean }(y_0) \tag{26}$$

5.1.1 Shifting by the mean

Let's use $\text{mean}(\bar{F})$ as our desired location parameter l by which to shift all vowel tokens. We've already established in Equation 19 and Equation 20 that this will involve only changes to the 0th DCT coefficient for the j^{th} token. If we substitute Equation 26 into Equation 19, we get

$$y'_{0j} = y_{0j} - \frac{\sqrt{2}\,\text{mean }(y_0)}{\sqrt{2}} = y_{0j} - \text{mean }(y_0) \tag{27}$$

Because $\sqrt{2}$ cancels out in Equation 27, to center all DCT coefficients on the mean across tokens, we just need to center y_0 on the mean across y_0.

5.2 Calculating standard deviations

The standard deviation over $\bar{F}$ is *not* a weighted sum, but as it happens for the standard deviation specifically, there is still a straightforward relationship between formant values and y_0. Firstly, the standard deviation across $\bar{F}$ can be given as

$$\text{sd}(\bar{F}) = \sqrt{\frac{\sum\left(\bar{F}_j - \text{mean}(\bar{F})\right)^2}{N}} \tag{28}$$

Substituting in Equation 24 and Equation 26, we have

$$\text{sd}(\bar{F}) = \sqrt{\frac{\sum\left(\sqrt{2}y_{0j} - \sqrt{2}\,\text{mean}(y_0)\right)^2}{N}} \tag{29}$$

This can be simplified in a few steps.

$$\mathrm{sd}(\bar{F}) = \sqrt{\frac{\sum \sqrt{2}^2 \big(y_{0j} - \mathrm{mean}(y_0)\big)^2}{N}} \tag{30}$$

$$\mathrm{sd}(\bar{F}) = \sqrt{\frac{\sum 2\big(y_{0j} - \mathrm{mean}(y_0)\big)^2}{N}}$$

$$\mathrm{sd}(\bar{F}) = \sqrt{2\frac{\sum \big(y_{0j} - \mathrm{mean}(y_0)\big)^2}{N}}$$

$$\mathrm{sd}(\bar{F}) = \sqrt{2}\sqrt{\frac{\sum \big(y_{0j} - \mathrm{mean}(y_0)\big)}{N}}$$

Resulting in

$$\mathrm{sd}(\bar{F}) = \sqrt{2}\,\mathrm{sd}(y_0) \tag{31}$$

5.2.1 Scaling by the standard deviation

Let's use $\frac{1}{\mathrm{sd}(\bar{F})}$ as our scaling parameter s by which to scale all vowel tokens. By substituting Equation 31 into Equation 14, we get

$$y'_{kj} = \frac{1}{\sqrt{2}\,\mathrm{sd}\ (y_0)} y_{kj} \tag{32}$$

So, when scaling by the standard deviation, or by a linear function over y_0, we need to multiply the result by $\sqrt{2}$, and then use it as a scaling parameter over all DCT coefficients.

6 Example Application

In this section, we'll take three normalization methods that were defined over formant point measurements, and redefine DCT normalization methods as outlined above. These three methods were chosen because they cover all three possibilities of shifting and scaling.

Table 4: Example Normalization Methods

Method	l (shift)	s (scale)
ΔF (Johnson 2020)	0	$\frac{1}{\delta(F)}$
Nearey (Nearey 1978)	$-\ \mathrm{mean}\ (\log F)$	1
Lobanov (Lobanov 1971)	$-\ \mathrm{mean}\ (F_i)$	$\frac{1}{\mathrm{sd}\ (F_i)}$

We'll be using Equation 5 (repeated here) as the generalized normalization equation that all three methods have in common.

$$F' = s(F + l) \tag{33}$$

6.1 ΔF

Johnson (2020) proposes a vocal tract length normalization method that scales all formants, but does not shift them. In the original paper, a quantity ΔF is defined, but for the purpose of this paper, we'll define it as a function $\delta(F)$.

$$\delta(F) = \sum_{i=1}^{M} \sum_{j=1}^{N} \frac{F_{ij}}{i - 0.5} \tag{34}$$

Where i is the formant number. The scaling and shifting parameters s and l for this method are

$$s = \frac{1}{\delta(F)};\ l = 0 \tag{35}$$

Plugging these parameters in to Equation 33, we get

$$F'_{ij} = \frac{1}{\delta(F)} F_{ij} \tag{36}$$

The $\delta(F)$ function is a weighted sum, so by Equation 25 we get

$$\delta(F) = \sqrt{2}\delta(y_0) \tag{37}$$

Since this method involves only scaling, we can plug Equation 37 into Equation 14 to get the DCT coefficients of the normalized formants as

$$y'_{kij} = \frac{1}{\sqrt{2}\delta(y_0)} y_{kij} \tag{38}$$

I used Equation 38 to normalize the DCT coefficients for F1 through F3 for two speakers from the PNC. Figure 12 compares the resulting inverse DCT on the unnormalized vs Δ F normalized formant tracks, focusing on just the long vowels.

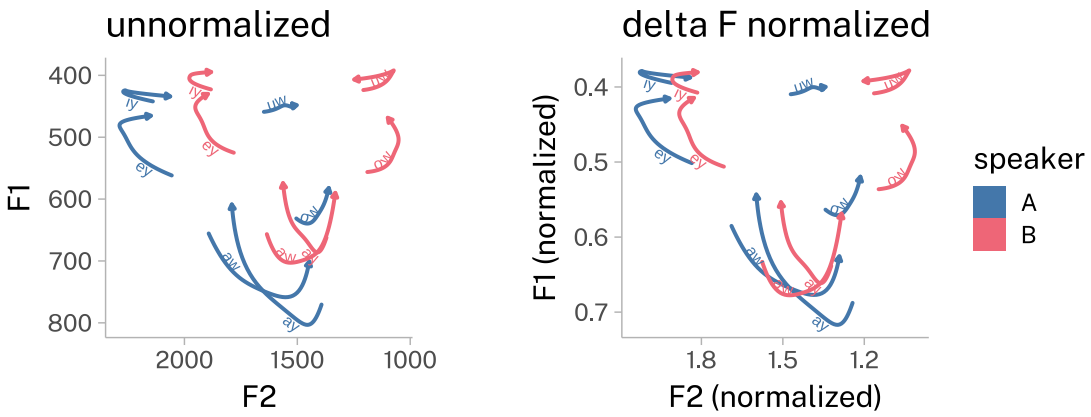

Figure 12: Unnormalized vs Delta F normalized formant tracks.

6.2 Nearey

Nearey normalization (Nearey 1978) involves a single shift in location, and has been recently argued to more closely reflect listeners perceptual categorization (Barreda 2021b).

Importantly, the first step of Nearey normalization involves taking the logarithm of formant values. As discussed in Section 4.4, there is no way to directly transform DCT coefficients of formants in Hz to DCT coefficients in log(Hz). Instead, the formants must first be logged, and then have the DCT applied. For the sake of notational brevity, for this section assume that any F refers to a log transformed formant, and any y_k refers to the DCT coefficients of that log transformed formant.

The scaling (s) and shifting (l) parameters for Nearey normalization are:

$$s = 1;\ l = -\ \text{mean}\ (F) \tag{39}$$

Plugging this into Equation 33, we get

$$F'_{ij} = F_{ij} - \text{mean}\ (F) \tag{40}$$

Since this involves shifting by the mean, we can use Equation 27 to define the DCT normalization like so:

$$y'_{0ij} = y_{0ij} - \text{mean}\ (y_0) \tag{41}$$

$$y'_{k>0ij} = y_{k>0ij}$$

The 0th DCT parameter of every token will be shifted by the mean across y_0, and all other DCT parameters will stay the same.

Using Equation 41, I Nearey normalized the DCT coefficients for F1 through F3 for the same two speakers from the PNC as before. Figure 13 compares the resulting inverse DCT on the unnormalized vs normalized formant tracks.

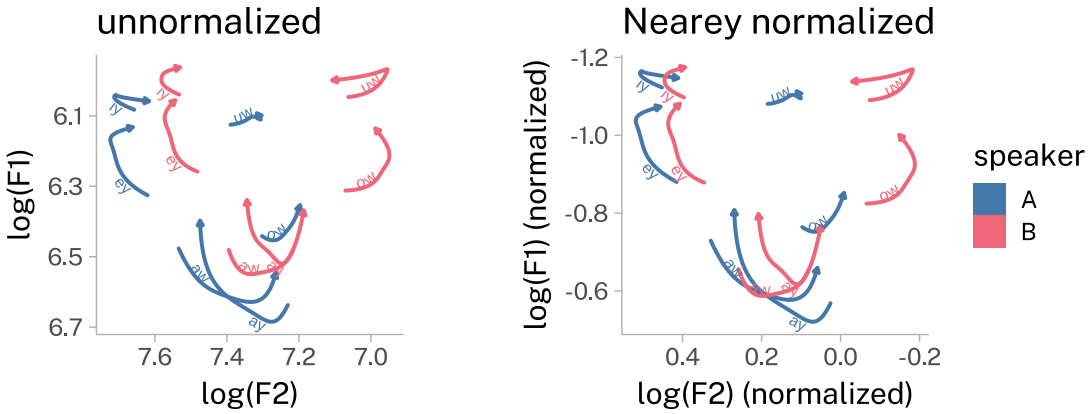

Figure 13: Unnormalized vs Nearey normalized formant tracks.

6.3 Lobanov

When applied to other domains of data analysis, Lobanov normalization (Lobanov 1971) is called z-scoring. The scaling (s) and shifting (l) parameters for Lobanov normalization are:

$$s = \frac{1}{\text{sd } (F_i)};\ l = -\text{ mean } (F_i) \tag{42}$$

Plugging these parameters into Equation 33, we get the following definition of Lobanov normalization.

$$F'_{ij} = \frac{1}{\text{sd } (F_i)}\big(F_{ij} - \text{mean } (F_i)\big) \tag{43}$$

The shifting parameter will only affect the 0$^{\text{th}}$ coefficient, and the scaling parameter will affect all coefficients. Plugging the equivalencies from Equation 27 and Equation 31 into Equation 43, we get the following definition for Lobanov normalizing the DCT coefficients.

$$y_{0ij} = \frac{1}{\sqrt{2}\,\text{sd } (y_{0i})}\big(y_{0ij} - \text{mean } (y_{0i})\big) \tag{44}$$

$$y_{k>0;ij} = \frac{1}{\sqrt{2}\,\text{sd } (y_{0i})} y_{k>0;ij}$$

Again, using Equation 44, I Lobanov normalized the DCT coefficients for F1 and F2 for the same two speakers from the Philadelphia neighborhood corpus. Figure 14

compares the resulting average formant tracks when applying the inverse DCT to the normalized coefficients.

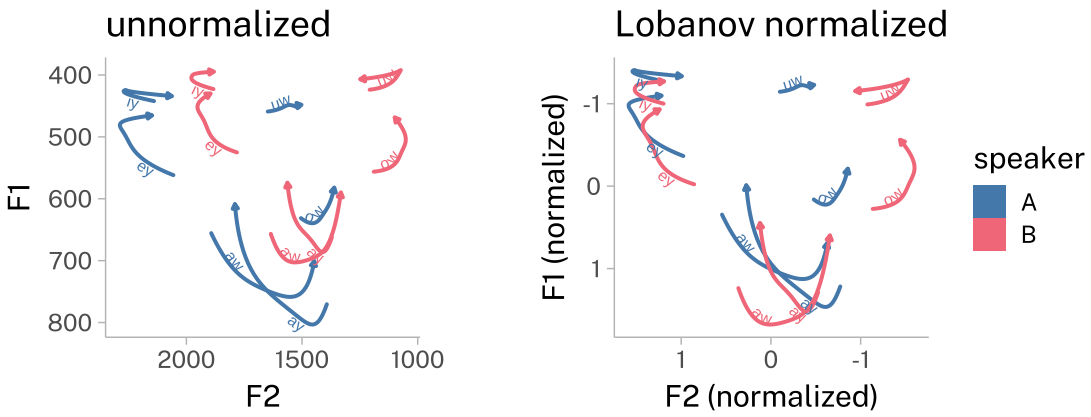

Figure 14: Unnormalized vs Lobanov normalized formant tracks.

6.4 Comparison to point-based normalization

It's visually clear that none of the normalization methods resulted in complete overlap of the formant trajectories for these two speakers. However, the goal of speaker normalization is to eliminate (socio)linguistically irrelevant differences, while preserving meaningful differences. While different approaches have been taken to evaluating how well any given normalization method achieves this goal (Adank et al. 2004; Barreda 2021b) the goal of this paper is just to define formant *track* normalization based on formant *point* normalization methods. As such, we'll briefly compare the formant track results above to the same point-based methods.

For this comparison, the same two speakers' formant track data was used, and formant values at 50% of the duration taken as the point measurement. These point measures were then normalized using the ΔF, Nearey, and Lobanov methods, as defined above. Then, for the same set of vowels, the average F1 and F2 was estimated. Figure 15 plots the result of all three methods as well as the unnormalized data.

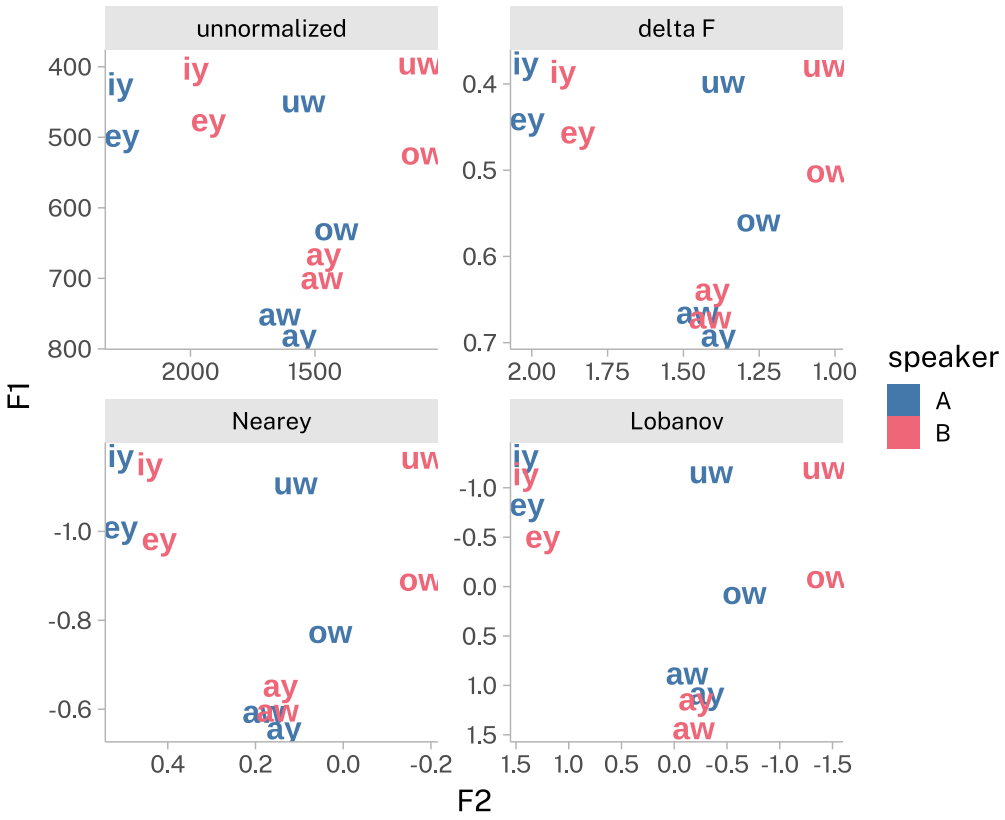

Figure 15: Comparison of three different formant point normalization methods to unnormalized formants.

Broadly similar patterns emerge in the normalized point values as did in the normalized formant tracks (e.g. Speaker A's fronter /uw/ and/ ow/ in all normalizations, and speaker B's slightly backer /iy/ and /ey/ in the delta F and Nearey normalizations).

6.5 Discussion

In this section, we've seen that normalization methods originally defined for formant point measurements can be directly translated into DCT coefficient normalization methods, and that the results on formant tracks are broadly similar to the results on formant points.

To briefly reiterate and refine the process of defining a DCT normalization procedure:

1. Any non-linear transformations must be applied to the formant tracks *before* the DCT is applied.
2. Any normalization parameters based on weighted sums (like the mean) or the standard deviation can be estimated using the 0^{th} DCT coefficient.
3. Location (l) parameters can be directly added or subtracted from the 0^{th} DCT coefficient only.

4. Scale parameters (s) need to be multiplied by $\sqrt{2}$, and applied to all DCT coefficients.

7 Conclusion

The Discrete Cosine Transform is a formant track smoothing method that has already found usage in phonetic research on vowel intrinsic spectral change. The use of DCT coefficients comes with a number of concrete benefits to research on vowel formant dynamics. The large body of research on speaker normalization can be drawn upon to directly normalize DCT coefficients, as approximately equivalent normalization functions can be defined for DCT coefficients based on the definition of the formant normalization functions.

Appendices

A The scipy implementation of the DCT

The scipy documentation for the DCT describes three ways the DCT can be "normalized", and two ways the DCT can be "orthogonalized" or "non-orthogonalized". All of these options on the DCT alter the terms to the left of the sum in the DCT formula. Let's define and simplify these components.

I'll use S to indicate the sum function, which is defined as

$$S_k(x) = \sum_{j=0}^{N-1} x_j \cos\left(\frac{\pi k(2j+1)}{2N}\right)$$

This term is unaltered by any of the different options scipy offers. Any given DCT implementation can be given as

$$y_k = ocS_k(x)$$

Where o is the orthogonalization term and c is the normalization constant.

The orthogonalization term is the easiest to define.

$$o = \begin{cases} 1 & \text{if orth} = \text{False} \\ \frac{1}{\sqrt{2}} & \text{if orth} = \text{True} \end{cases}$$

The scipy documentation provides the mathematical definition for "backward" normalization constant only, but the "forward" normalization can be inferred from its output.

$$c = \begin{cases} 2 & \text{if norm} = \text{backward} \\ \frac{1}{N} & \text{if norm} = \text{forward} \end{cases}$$

As a demonstration by example, we can define a python function for just the sum function, then apply it to the formant track in Figure 16.

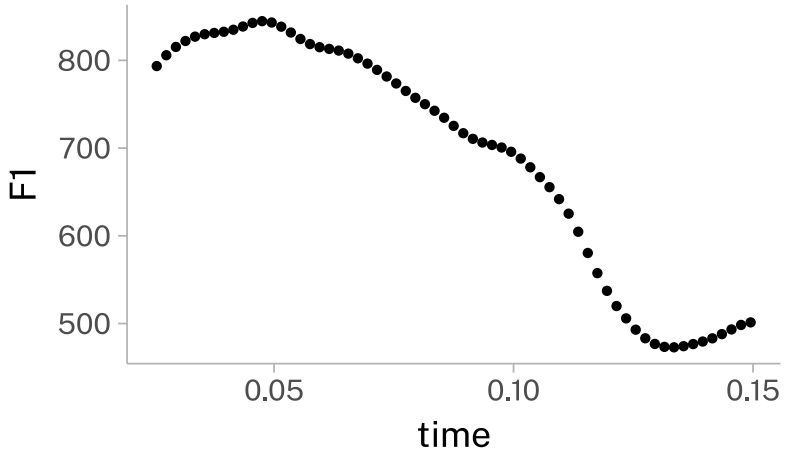

Figure 16: Demonstration formant track

Listing 2: Definition of the DCT sum function

```
import numpy as np
from scipy.fft import dct

def S(x, k):
  j = np.arange(len(x))
  N = len(x)

  result = sum(
      x * np.cos(
        (np.pi * k * ((2*j)+1))/
        (2*N)
      )
    )

  return result
```

We can get the result of the sum function for the 0th and 1st DCT coefficients to then examine the outcome of the different normalizations.

Listing 3: Sum terms of the DCT

```
s_0 = S(f1, 0)
s_1 = S(f1, 1)
```

At this point, we can also get the 0th and 1st DCT coefficients from the scipy implementation.

Listing 4: Application of the scipy DCT

```
dct_backward = dct(f1, norm = "backward")
dct_forward  = dct(f1, norm = "forward")
```

The normalizing constant for `norm = "backward"` is documented to be 2, so multiplying `s_0` and `s_1` by 2 should be equal to the 0th and 1st coefficients in `dct_backward`.

Listing 5: Backward DCT normalization

```
np.array([
  2 * np.array([s_0, s_1]),
  dct_backward[0:2]
])
```

```
array([[87057.43, 11509.54],
       [87057.43, 11509.54]])
```

If the normalizing constant for `norm="forward"` is $\frac{1}{N}$, dividing `s_0` and `s_1` by the length of the input vector should be equal to the 0th and 1st coefficients in `dct_forward`.

Listing 6: forward DCT normalization

```
N = len(f1)

np.array([
  np.array([s_0, s_1]) / N,
  dct_forward[0:2]
])
```

```
array([[690.93,  91.35],
       [690.93,  91.35]])
```

Admittedly, it would be more ideal to be able to reference the actual forward normalization constant from the scipy documentation, but it is not provided.

B The DCT Basis

While the formula in Equation 1 can be used to calculate the DCT coefficients, the formula to calculate the DCT basis functions in Figure 2 is different. If B is a matrix of the basis functions, the k^{th} basis function will be in its columns. To get B, we apply the DCT with backward normalization to an identity matrix I (that is, a matrix with 1s along the diagonal, and 0s elsewhere). The orthogonalization term o is included in Equation 45.

$$B_{jk} = 2o \sum_{j=0}^{N-1} I_{jk} \cos\left(\frac{\pi k(2j+1)}{2N}\right) \tag{45}$$

This can be quickly implemented using the scipy DCT implementation like so:

Listing 7: Getting the DCT basis functions

```
N = len(f1)
dct_basis = dct(np.eye(N), norm = "backward", orthogonalize = True)
```

C The choice of orthogonalization

The choice of "orthogonalizing" the DCT coefficients, that is, dividing the 0th coefficient by $\sqrt{2}$, does introduce some awkwardness into the normalization procedures described here. Using the orthogonalized DCT was a design decision within fasttrackpy due to its reliance on regression-based DCT coefficients.

As a practical issue, formant tracking sometimes returns missing, or NA values for some, but not all, time points along a formant track. With missing values, the DCT cannot be directly applied. However, the DCT coefficients can be approximated by linear regression, using the DCT basis as the "predictors".

Listing 8: Comparison of direct vs regression-based DCT

```
dct_reg = np.linalg.lstsq(dct_basis, f1, rcond=None)[0]
dct_direct = dct(f1, norm = "forward", orthogonalize = True)

np.array([
  dct_reg[0:3],
  dct_direct[0:3]
])
```

```
array([[488.56,  91.35, -23.14],
       [488.56,  91.35, -23.14]])
```

Orthogonalizing the first coefficient was the only option that resulted in the same coefficients for both regression and direct DCT within the scipy implementation. Without orthogonalizing the first coefficient, the 0th coefficient is not equal between the regression based DCT and direct DCT.

Listing 9: Comparison of direct vs regression-based non-orthogonalized DCT

```
dct_nonorth_basis = dct(np.eye(N), norm = "backward", orthogonalize = False)

dct_nonorth_reg = np.linalg.lstsq(dct_nonorth_basis, f1, rcond=None)[0]
dct_nonorth_direct = dct(f1, norm="forward", orthogonalize=False)

np.array([
  dct_nonorth_reg[0:3],
  dct_nonorth_direct[0:3]
])
```

```
array([[345.47,  91.35, -23.14],
       [690.93,  91.35, -23.14]])
```

Since the design decision to orthogonalize the DCT coefficients was made within fasttrackpy, which was the tool used to arrive at these DCT coefficients in this paper, this was also the version of the DCT used here.

References

Adank, Patti, Roel Smits & Roeland van Hout. 2004. A comparison of vowel normalization procedures for language variation research. *The Journal of the Acoustical Society of America* 116(5). 3099–3107. https://doi.org/10.1121/1.1795335.

Barreda, Santiago. 2021a. Fast Track: Fast (nearly) automatic formant-tracking using Praat. *Linguistics Vanguard* 7(1). 20200051. https://doi.org/10.1515/lingvan-2020-0051.

Barreda, Santiago. 2021b. Perceptual validation of vowel normalization methods for variationist research. *Language Variation and Change* 33(1). 27–53. https://doi.org/10.1017/S0954394521000016.

Cox, Felicity & Sallyanne Palethorpe. 2019. Vowel variation in a standard context across four major Australian cities. (Ed.) Sasha Calhoun, Paola Escudero, Marija Tabain & Paul Warren. *Proceedings of the 19th International Congress of Phonetic Sciences*. Australasian Speech Science; Technology Association Inc.; International Phonetic Association. 577–581. https://assta.org/proceedings/ICPhS2019/papers/ICPhS_626.pdf.

Docherty, Gerard, Simón Gonzalez & Nathaniel Mitchell. 2015. Static vs dynamic perspectives on the realization of vowel nucleii in West Australian English. (Ed.) The Scottish Consortium for ICPhS 2015. *Proceedings of the 18th International Congress of Phonetic Sciences* 956. https://www.internationalphoneticassociation.org/icphs-proceedings/ICPhS2015/Papers/ICPHS0956.pdf.

Fox, Robert Allen & Ewa Jacewicz. 2009. Cross-dialectal variation in formant dynamics of American English vowels. *The Journal of the Acoustical Society of America* 126(5). 2603–2618. https://doi.org/10.1121/1.3212921.

Fruehwald, Josef. 2025. new-fave. v1.1.1. https://doi.org/10.5281/ZENODO.14837885.

Fruehwald, Josef & Santiago Barreda. 2024. Fasttrackpy. v0.5.3. https://doi.org/10.5281/ZENODO.10212099.

Gubian, Michele, Francisco Torreira & Lou Boves. 2015. Using Functional Data Analysis for investigating multidimensional dynamic phonetic contrasts. *Journal of Phonetics* 49. 16–40. https://doi.org/10.1016/j.wocn.2014.10.001.

Guzik, Karita M. & Jonathan Harrington. 2007. The quantification of place of articulation assimilation in electropalatographic data using the similarity index (SI). *Advances in Speech Language Pathology* 9(1). 109–119. https://doi.org/10.1080/07268600601094294.

Hillenbrand, James M., Michael J. Clark & Terrance M. Nearey. 2001. Effects of consonant environment on vowel formant patterns. *The Journal of the Acoustical Society of America* 109(2). 748–763. https://doi.org/10.1121/1.1337959.

Jannedy, Stefanie & Melanie Weirich. 2017. Spectral moments vs discrete cosine transformation coefficients: Evaluation of acoustic measures distinguishing two merging German fricatives. *The Journal of the Acoustical Society of America* 142(1). 395–405. https://doi.org/10.1121/1.4991347.

Jochim, Markus, Raphael Winkelmann, Klaus Jaensch, Steve Cassidy & Jonathan Harrington. 2024. *emuR: Main package of the EMU speech database management system*. https://github.com/IPS-LMU/emuR.

Johnson, Keith. 2020. The ΔF method of vocal tract length normalization for vowels. *Laboratory Phonology: Journal of the Association for Laboratory Phonology* 11(1). 10. https://doi.org/10.5334/labphon.196.

Labov, William. 2001. *Principles of linguistic change. Volume 2: Social factors. Language in society*. Oxford: Blackwell.

Labov, William, Sherry Ash & Charles Boberg. 2006. *The atlas of North American English: Phonetics, phonology and sound change*. New York: Mouton de Gruyter.

Labov, William & Ingrid Rosenfelder. 2011. The Philadelphia Neighborhood Corpus.

Lobanov, Boris. 1971. Classification of Russian vowels spoken by different listeners. *Journal of the Acoustical Society of America* 49. 606–608. https://doi.org/10.1121/1.1912396.

Mersmann, Olaf. 2024. Fftw: Fast FFT and DCT based on the FFTW library. https://CRAN.R-project.org/package=fftw.

Morrison, Geoffrey Stewart. 2009. Likelihood-ratio forensic voice comparison using parametric representations of the formant trajectories of diphthongs. *The Journal of the Acoustical Society of America* 125(4). 2387–2397. https://doi.org/10.1121/1.3081384.

Nearey, Terrance M. 1978. *Phonetic feature systems for vowels*. PhD thesis. https://sites.ualberta.ca/~tnearey/Nearey1978_compressed.pdf.

Nearey, Terrance M. & Peter F. Assmann. 1986. Modeling the role of inherent spectral change in vowel identification. *The Journal of the Acoustical Society of America* 80(5). 1297–1308. https://doi.org/10.1121/1.394433.

Ramsay, James & Bernard W. Silverman. 2006. *Functional data analysis*. New York: Springer.

Risdal, Megan L. & Mary E. Kohn. 2014. Ethnolectal and generational differences in vowel trajectories: Evidence from African American English and the Southern Vowel System. *Penn Working Papers in Linguistics* 20(2). 139–148. https://doi.org/20.500.14332/45004.

Rosenfelder, Ingrid, Josef Fruehwald, Keelan Evanini, Scott Seyfarth, Christian Brickhouse, Kyle Gorman, Hillary Prichard & Jiahong Yuan. 2024. FAVE: Forced alignment and vowel extraction. v2.0.3. https://doi.org/10.5281/ZENODO.593309.

Sóskuthy, Márton. 2017. Generalised additive mixed models for dynamic analysis in linguistics: A practical introduction. https://doi.org/10.48550/arXiv.1703.05339.

Sóskuthy, Márton. 2021. Evaluating generalised additive mixed modelling strategies for dynamic speech analysis. *Journal of Phonetics* 84. 101017. https://doi.org/10.1016/j.wocn.2020.101017.

Tanner, James, Morgan Sonderegger & Jane Stuart-Smith. 2022. Multidimensional acoustic variation in vowels across English dialects. In, 72–82. Seattle, Washington: Association for Computational Linguistics. https://doi.org/10.18653/v1/2022.sigmorphon-1.8.

Virtanen, Pauli, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, et al. 2020. SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature Methods* 17(3). 261–272. https://doi.org/10.1038/s41592-019-0686-2.

Wallace, Gregory K. 1992. The JPEG still picture compression standard. *IEEE Transactions on Consumer Electronics* 38(1). xviii–xxxiv. https://doi.org/10.1109/30.125072.

Watson, Catherine I. & Jonathan Harrington. 1999. Acoustic evidence for dynamic formant trajectories in Australian English vowels. *The Journal of the Acoustical Society of America* 106(1). 458–468. https://doi.org/10.1121/1.427069.

Williams, Daniel & Paola Escudero. 2014. A cross-dialectal acoustic comparison of vowels in Northern and Southern British English. *The Journal of the Acoustical Society of America* 136(5). 2751–2761. https://doi.org/10.1121/1.4896471.

Williams, Daniel, Jan-Willem van Leussen & Paola Escudero. 2015. Beyond North American English: Modelling vowel inherent spectral change in British English and Dutch. (Ed.) The Scottish Consortium for ICPhS 2015. *Proceedings of the 18th International Congress of Phonetic Sciences* 596. https://www.internationalphoneticassociation.org/icphs-proceedings/ICPhS2015/Papers/ICPHS0596.pdf.

Zahorian, Stephen A. & Amir J. Jagharghi. 1991. Speaker normalization of static and dynamic vowel spectral features. *The Journal of the Acoustical Society of America* 90(1). 67–75. https://doi.org/10.1121/1.402350.

Zahorian, Stephen A. & Amir Jalali Jagharghi. 1993. Spectral-shape features versus formants as acoustic correlates for vowels. *The Journal of the Acoustical Society of America* 94(4). 1966–1982. https://doi.org/10.1121/1.407520.