Vibrato Synthesis in Choral Singing via Per-Note Prediction Models

Benjamin Prud'homme
Department of Computer Science
CUNY Graduate Center
New York, NY, United States
bprudhomme@gradcenter.cuny.edu

***Abstract*—Most research in expressive vocal modeling has focused on solo singing, where techniques like vibrato are more prominently exhibited and easier to analyze. In contrast, choral recordings emphasize blend and subtlety, making expressive features harder to detect, annotate, and manipulate. This paper presents a method for injecting realistic vibrato into straight-tone performances drawn from multitrack choral recordings, using predictive models trained on solo singing data. Given manually specified note boundaries, the system extracts pitch and timbral features for each note and predicts vibrato depth and rate. These predictions are used to synthesize vibrato contours, which are applied to the pitch trajectory and re-synthesized using the WORLD vocoder. Evaluation shows that the method produces musically plausible transformations in many cases, particularly when guided by high-quality note annotations. Limitations in segmentation and overprediction are discussed, along with proposed improvements such as intra-note vibrato modeling, voice-part-specific models, and more robust segmentation methods. This work lays the foundation for semi-synthetic expressive tools that allow choral musicians to explore interpretive variants while preserving the realism of human vocal performance.**

***Index Terms*—vibrato modeling, choral synthesis, expressive singing, machine learning, voice transformation**

I. Introduction

Research in singing voice analysis has advanced from early acoustic studies of the singer's formant [1] and phonation modes [2] to modern machine learning models that classify expressive modes such as vibrato, breathiness, and belting [3], [4], [5]. Most of this work, however, has focused on solo singing, where performances are isolated and expressive styles are often labeled at the clip or technique level.

Choral music presents a more complex challenge. While several multitrack choral datasets exist [6], [7], [8] and often include time-aligned MIDI or F0, they typically lack expressive annotations—such as vibrato depth, vocal effort, or phrasing—at either the note or frame level. While this limitation also applies to most solo datasets, expressive traits in choral recordings are particularly difficult to annotate, in part because variation is intentionally subtle: singers blend rather than stand out, making expressive features harder to isolate and quantify. Yet even small differences in vibrato or phrasing can influence blend, tuning, and emotional affect. Synthetic choral datasets like ChoralSynth [9] offer precise control over expressive parameters but lack the natural variability of human ensemble recordings.

This project aims to bridge this gap by exploring vibrato injection as a post-hoc expressive transformation for isolated choral tracks. The goal is to enable controllable expression in choral synthesis, combining the realism of real voices with the flexibility of expressive modeling. I envision a tool that enables choir directors to explore expressive performance variants post-recording, supporting flexible rehearsal and informed musical decisions, while also serving researchers interested in modeling and manipulating choral expressivity. This research emphasizes augmentation, not replacement—empowering human interpretation through computational tools.

II. Related Work

A. Solo Voice Modeling

Expressive singing has long been a focus in voice modeling research. Sundberg's foundational work introduced the concept of the *singer's formant*—a resonance cluster aiding vocal projection [1]—and proposed a taxonomy of four phonation modes: breathy, neutral, resonant, and pressed [2]. This laid the groundwork for acoustic and perceptual studies of vocal expressivity.

Subsequent research leveraged machine learning to analyze expressive technique. Wilkins et al. [4] introduced VocalSet, a dataset of solo singers performing varied techniques across vowels and pitches. In their baseline experiments, they trained models for both vocal technique classification and singer identification, demonstrating that deep networks can distinguish expressive technique categories and individual vocal timbres from isolated recordings. Proutskova et al. [3] applied classifiers to phonation mode detection. More recently, deep neural networks have enabled frame-level classification of expressive singing techniques. Yamamoto et al. [5] proposed a characteristics-aware CRNN model that improves multi-label detection of pitch- and timbre-based techniques, including vibrato.

While this body of work demonstrates the feasibility of learning from controlled, expressive solo recordings, the transition to ensemble singing presents new challenges—particularly in disentangling and modeling expression at the level of individual voices.

B. Expressive Control and Post-Production

Post-production editing of expressive vocal features—such as pitch and vibrato—is common in commercial solo music. Ben Duinker [10] describes how tools like Auto-Tune have shaped vocal production by enabling fine-grained pitch correction and encouraging a trend toward sonic refinement. While editing is routine in solo recordings, it is typically more restrained in choral contexts, where uniformity and ensemble blend are prioritized. Karim Blondy [11], an audio post-production specialist, notes that pitch correction is occasionally applied in classical choir performances, though such interventions remain rare. This restraint reflects both practical and aesthetic considerations. Many choral traditions prioritize subtle ensemble blend and 'naturalness' over polished individual perfection, valuing the immersive feel of group performance [12]. Moreover, solo-oriented vocal modeling tools may not generalize well to ensemble contexts, where the need for vocal blend and subtle expressivity introduces distinct modeling challenges.

C. Choral Datasets

Advancing computational analysis of choral singing has long been hindered by a scarcity of multitrack choral datasets. Helena Cuesta addressed this gap in her doctoral work [7], contributing several foundational resources. Some were introduced earlier [6], [8] and later consolidated in her dissertation.

Key datasets include:

- **Choral Singing Dataset (CSD)** [6]: Semi-professional SATB choir recorded by section, with F0 and note annotations.
- **ESMUC Choir Dataset (ECD)** [7]: 12-singer student choir recorded with individual and room mics, annotated with F0 and notes.
- **Dagstuhl ChoirSet (DCS)** [8]: Amateur SATB ensemble with multiple mic types per singer, beat annotations, and time-aligned scores.
- **Cantoría Dataset** [7]: Professional SATB quartet specializing in Iberian Renaissance music, with F0 annotations.

Despite certain limitations acknowledged by Cuesta [7]—such as annotation granularity and recording artifacts like vocal bleed—these datasets nevertheless provide invaluable material for analyzing ensemble blend, intonation, and individual vocal contributions in real choral performances.

ChoralSynth [9], a fully synthetic SATB synthesis model, offers explicit symbolic control over expressive features such as dynamics and (potentially) vibrato. While useful for controlled experiments and synchronized output, its generated audio lacks the nuanced timbral richness of live performance. Additionally, it does not align with my goal of augmenting real recordings with expressive features rather than replacing them.

III. Methodology

A. Data and Preprocessing

1) VocalSet Dataset and Selection: As introduced in Section II-C, I used the VocalSet dataset [4] to train my vibrato prediction models. The dataset contains recordings of 20 singers (9 female, 11 male), spanning a range of vocal techniques, vocalizations (arpeggios, long tones, scales, excerpts), and five vowels (a, e, i, o, u). Figure 1 illustrates this structure.

To align with typical choral performance, I curated a subset focused on sustained phonation and expressive dynamics relevant to choral singing. Specifically, I selected:

- **Vibrato** (from Arpeggios, Long Tones, and Scales).
- **Fast Forte**, **Fast Piano**, **Slow Forte**, and **Slow Piano** (from Arpeggios and Scales).
- **Forte**, **Pianissimo**, and **Messa di Voce** (from Long Tones).

Techniques such as Belt, Breathy, and Vocal Fry were excluded from my primary training set, as they were deemed less characteristic of traditional choral singing styles, where timbral blend and uniformity are often prioritized. In addition, Straight recordings were excluded as they do not service the goal of modeling vibrato. This focused data selection strategy aimed to train my model on vocal expressions most relevant to my goal of realistically enhancing choral recordings with controlled vibrato.

2) Audio Feature Extraction: For each selected VocalSet recording, I extracted frame-level acoustic features suitable for vibrato modeling: fundamental frequency (F0), root mean square (RMS) energy, and the spectral envelope, all at a consistent frame rate.

Each `.wav` file was loaded at its native 44.1 kHz sampling rate using `librosa` [13]. To remove silence from the beginning and end, I computed RMS energy (frame length 2048, hop length 512) and trimmed the audio to the region between the first and last frames where the linear RMS amplitude exceeded 0.005. Files with insufficient active frames were excluded. This trimming step was important not only to focus on sung content, but also because early testing showed that silence at the edges often caused Dynamic Time Warping (DTW) to fail or misalign in later stages.

From the trimmed signal (y_{trimmed}), I estimated the F0 contour and corresponding timestamps (t) using `librosa.pyin` [14], and interpolated missing values. I then computed frame-aligned RMS energy using `librosa.feature.rms` and extracted the spectral envelope with `pyworld.cheaptrick` [15]. The resulting time-aligned features were stored as compressed NumPy (`.npz`) files for downstream modeling (Section III-B).

3) Note Segmentation: To enable per-note analysis, I aligned each VocalSet recording with symbolic MIDI transcriptions manually derived from its corresponding score. For "Long Tones" exercises—where notes are continuous in the score but separated by breaths in performance—I manually inserted short rests to prevent alignment artifacts. MIDI files were parsed and adjusted using `PrettyMIDI` [16].

To improve alignment, I shifted each MIDI to start at time zero and scaled its onsets and offsets in order to match the final note's offset with the duration of the trimmed audio (Section III-A2). This avoided per-file tempo tuning and improved DTW robustness in testing. An added benefit is that it suggests

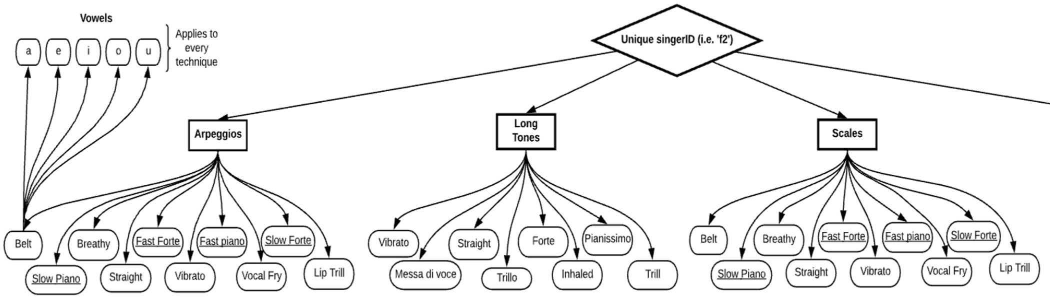

Fig. 1. Breakdown of vocalizations and techniques in the VocalSet dataset. The diagram shows how various techniques (e.g., Straight, Vibrato, Belt, dynamic variations) are nested within four main performance contexts (Arpeggios, Long Tones, Scales, Excerpts) and apply across five standard vowels. Figure adapted from Wilkins et al. [4].

a strategy for future tools to ingest score-based input without requiring manual rest trimming or tempo adjustment.

For alignment, I extracted two feature matrices at a consistent frame rate:

- From audio: onset strength (via `librosa.onset.onset_strength`) and a quantized pitch contour (via `librosa.pyin`, converted to MIDI numbers with unvoiced regions set to 0).
- From MIDI: a note onset envelope (1.0 at each note onset, 0.0 otherwise) and a framewise MIDI pitch vector.

Both were normalized and stacked into 2D matrices, then aligned using `librosa.sequence.dtw` [13], with MIDI as the reference and Euclidean distance as the cost metric. The resulting warping path was used to update the timing of each note in the MIDI file, yielding note boundaries quantized to frame resolution.

These note annotations were saved as CSV files and served as the second half of the model's input. While this process yields detailed boundaries, some imprecisions remain, as discussed in Section V.

B. Vibrato Prediction Models

1) Input Features and Ground Truth: For each segmented note (Section III-A3), I extracted the corresponding slices of F0, RMS, and spectrogram data from the `.npz` files described in Section III-A2. These were passed to the pyAMPACT toolkit [17] to compute ground-truth vibrato depth and rate. pyAMPACT defines vibrato depth as twice the amplitude of the dominant frequency component in the detrended F0 contour, and vibrato rate as the frequency (in Hz) of that component, as identified by the peak of the FFT.

The model input vector included: mean F0 (Hz), mean RMS energy, note duration (s), and voice part (soprano=0, alto=1, tenor=2, bass=3). Frame-level spectrograms were avoided as inputs to prevent encoding the very vibrato I aim to predict.

2) Model Architecture and Training: I used the `HistGradientBoostingRegressor` from scikit-learn for its interpretability, fast training, and ability to capture nonlinear interactions without feature scaling. The training set followed VocalSet's official singer-based split, ensuring independence between training and test singers. The test set included one of seven sopranos, one of two altos, one of four tenors, and two of seven basses; all others were used for training.

To filter out implausible vibrato patterns—often caused by segmentation or pitch-tracking errors—I excluded notes with depth greater than 50 Hz or rate outside the 4–10 Hz range. These thresholds removed rare outliers while preserving musically meaningful data, reducing the dataset from 20,774 to 16,153 notes (11,848 training and 4,305 test). Performance was evaluated on the test set using Mean Absolute Error (MAE) and coefficient of determination (R^2).

C. Transformation of Multitrack Choral Recordings

I applied the trained vibrato models to straight-tone recordings from the CSD dataset, using note boundaries from time-aligned MIDI files. Automatic segmentation (Section III-A3), while effective on VocalSet, was unreliable on CSD—particularly when note transitions involved voiced consonants or subtle pitch shifts. For each note, I extracted frame-level F0, RMS, and spectrogram features (Section III-A2), constructed a 4D summary vector (Section III-B1), and manually specified the voice part. This vector was passed to the trained models to predict vibrato depth (d) and rate (r).

The vibrato contour was synthesized by adding a sinusoidal deviation to the F0 trajectory: $f_0'(t) = f_0(t) + \frac{d}{2} \cdot \sin(2\pi r t)$, where t is the frame time vector. The modified pitch contour was passed to the WORLD vocoder for re-synthesis, using the original spectral envelope and aperiodicity.

Each transformed note was crossfaded with its original version using a 30 ms fade-in/fade-out window to ensure smooth transitions. This process was applied independently to each note, resulting in a transformed recording with pitch-appropriate, expressive vibrato.

IV. Results

We report quantitative and qualitative results for the vibrato models and their transformations.

A. Model Performance

I evaluated the vibrato models on a held-out test set from VocalSet. The vibrato depth model achieved a Mean Absolute Error (MAE) of 5.817 Hz and an R^2 score of 0.175, while

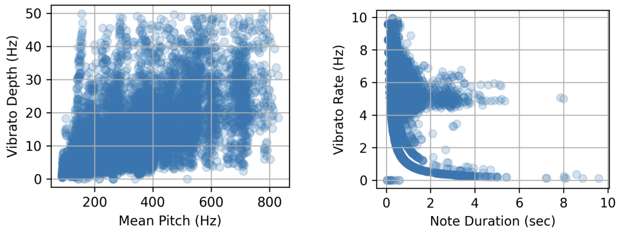

Fig. 2. Predicted vibrato parameters on the training set. **Left:** Vibrato depth (Hz) as a function of mean note pitch. **Right:** Vibrato rate (Hz) as a function of note duration (seconds). Patterns reflect musically grounded tendencies, such as greater vibrato depth for higher pitches and slower vibrato for longer notes.

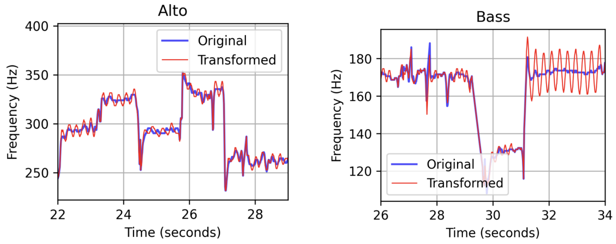

Fig. 3. Example pitch contours before and after vibrato transformation. **Left:** Alto voice with subtle vibrato. **Right:** Bass voice with exaggerated modulation from depth overprediction on a long, low-pitched note.

the vibrato rate model achieved an MAE of 1.440 Hz and an R^2 of 0.179. These values indicate that both models were able to capture general trends relating acoustic features to vibrato characteristics, though much of the variance in the target values remains unexplained.

To gain further insight into the behavior of the models, I visualized their predictions across the feature space. Figure 2 shows two side-by-side plots: the left plot displays predicted vibrato depth as a function of mean pitch, while the right plot shows predicted vibrato rate as a function of note duration. While the depth–pitch relationship is not strictly linear, higher vibrato depths tend to cluster around mid- to high-pitched notes, especially above 200 Hz. This aligns with stylistic tendencies in classical singing, where upper voices are more likely to employ expressive vibrato. The right plot reveals a strong inverse relationship between vibrato rate and note duration: predicted vibrato rate decreases sharply as duration increases. This behavior aligns with musical intuition—short notes, when sung with vibrato, tend to require faster oscillations to fit within their limited time span, while longer notes allow for slower, more stable vibrato. The model appears to have captured this dependency effectively, making note duration one of the most influential features in vibrato rate prediction.

B. *Transformation Observations*

Applied to straight-tone CSD recordings, the vibrato injection pipeline often produced natural-sounding results, particularly when aligned with sustained vowels, enhancing expressiveness without introducing artifacts. The fade-in/fade-out window around each transformed note was critical for avoiding discontinuities and ensuring smooth transitions between segments. This crossfading also helped preserve realism by retaining the original audio in noisy or unpitched regions—such as breaths, plosives, or ambient room sounds—that could otherwise have acquired unnatural oscillations from pitch modification. The ability to selectively transform voiced regions was central to the perceptual quality of the output.

However, a recurring issue emerged in some bass recordings: the model occasionally overpredicted vibrato depth on long, low-pitched notes, leading to exaggerated modulation. Figure 3 (right) shows one such case, where the transformed pitch oscillates unnaturally at the end of a sustained note. In contrast, the alto transformation (left) shows a more stylistically appropriate result, with smooth and subtle vibrato added to stable pitches.

V. Discussion

While the transformation pipeline produced promising results across diverse voice types and recordings, several limitations point to directions for future improvement. First, the note segmentation pipeline (Section III-A3), though effective on VocalSet during development, was less reliable on choral recordings. It struggled to detect boundaries involving subtle pitch changes or voiced consonants without clear amplitude drops. This motivated the use of aligned MIDI boundaries in CSD. However, segmentation errors may also have affected some VocalSet examples used in training. Manual verification of over 20,000 notes was infeasible, leaving some VocalSet segments vulnerable to alignment error. Second, while encoding voice part as a categorical feature helped the model generalize across singers and registers, it may have been too coarse to capture finer-grained differences. Future work could explore models trained separately for each SATB part, or conditioning on tessitura, pitch range, or vowel context. Third, the current system applies a fixed vibrato depth and rate per note, whereas real singers often vary vibrato expressively within a note—starting subtly and intensifying over time. Capturing this intra-note variation, perhaps via time series models or neural trajectory generators, could improve realism. Finally, segmentation could be further refined. Incorporating features like spectral flatness or vowel detection, or training a dedicated segmentation model on labeled choral audio, may improve onset and offset accuracy—particularly in reverberant or polyphonic contexts.

VI. Conclusion

This work presents a method for injecting vibrato into straight-tone choral recordings using models trained on note-level features. Given note boundaries, the system predicts vibrato depth and rate from pitch and timbral features, then applies the resulting contours using the WORLD vocoder. While some overprediction and segmentation sensitivity remain, the method often yields pitch-appropriate, perceptually natural vibrato.

This approach lays the groundwork for tools that could let choir directors or ensemble leaders explore expressive variants in rehearsal recordings—supporting stylistic experimentation while preserving the realism of human voices.

References

[1] Johan Sundberg, "The acoustics of the singing voice," *Scientific American*, vol. 236, no. 3, pp. 82–91, Mar. 1977. [Online]. Available: https://faculty.washington.edu/losterho/Sundberg.pdf

[2] Johan Sundberg, *The Science of the Singing Voice*. DeKalb, IL: Northern Illinois University Press, 1987.

[3] Polina Proutskova, Christophe Rhodes, Tim Crawford, and Geraint Wiggins, "Breathy, resonant, pressed: Automatic detection of phonation mode from recordings of sung vocalizations," *Journal of New Music Research*, vol. 42, no. 2, pp. 171–186, 2013. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/09298215.2013.821496

[4] Julia Wilkins, Prem Seetharaman, Alison Wahl, and Bryan Pardo, "VocalSet: A singing voice dataset," in *Proc. 19th Int. Society for Music Information Retrieval Conf. (ISMIR)*, Paris, France, 2018. [Online]. Available: https://zenodo.org/record/1203819

[5] Yuya Yamamoto, Juhan Nam, and Hiroko Terasawa, "PrimaDNN: A characteristics-aware DNN customization for singing technique detection," in *Proc. 31st European Signal Processing Conf. (EUSIPCO)*, Helsinki, Finland, 2023. [Online]. Available: https://arxiv.org/abs/2306.14191

[6] Helena Cuesta, Emilia Gómez, Agustín Martorell, and Francisco Loáiciga, "Analysis of intonation in unison choir singing," in *Proc. 15th Int. Conf. on Music Perception and Cognition (ICMPC) and 10th ESCOM Conf.*, Graz, Austria, 2018. [Online]. Available: https://repositori-api.upf.edu/api/core/bitstreams/15ccc8b2-093e-4c8b-9839-76972057c578/content

[7] Helena Cuesta, "Data-driven pitch content description of choral singing recordings," Ph.D. dissertation, Dept. of Information and Communication Technologies, Universitat Pompeu Fabra, 2022. [Online]. Available: https://www.tdx.cat/handle/10803/673924

[8] Sebastian Rosenzweig, Helena Cuesta, Christof Weiß, Frank Scherbaum, Emilia Gómez, and Meinard Müller, "Dagstuhl ChoirSet: A multitrack dataset for MIR research on choral singing," *Trans. Int. Soc. Music Info. Retrieval*, vol. 3, no. 1, pp. 98–110, 2020. [Online]. Available: http://transactions.ismir.net/articles/10.5334/tismir.48/

[9] Jyoti Narang, Viviana De La Vega, Xavier Lizarraga, Oscar Mayor, Hector Parra, Jordi Janer, and Xavier Serra, "ChoralSynth: Synthetic dataset of choral singing," unpublished manuscript, arXiv preprint arXiv:2311.08350, Nov. 2023. [Online]. Available: https://arxiv.org/abs/2311.08350

[10] Ben Duinker, "Auto-Tune as instrument: Trap music's embrace of a repurposed technology," *Popular Music*, vol. 43, no. 2, pp. 1–25, Feb. 2025. [Online]. Available: https://doi.org/10.1017/S0261143024000369

[11] Karim Blondy, "Auto-Tune and classical music: Harmony or heresy?," *La Scena Musicale*, Apr. 2024. [Online]. Available: https://myscena.org/karim-blondy/auto-tune-and-classical-music-harmony-or-heresy/

[12] Robert Philip, *Performing Music in the Age of Recording*. New Haven, CT: Yale University Press, 2004.

[13] Brian McFee, Colin Raffel, Dawen Liang, Daniel P. W. Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto, "librosa: Audio and music signal analysis in Python," in *Proc. Python in Science Conf. (SciPy)*, 2015, pp. 18–24. [Online]. Available: https://doi.curvenote.com/10.25080/Majora-7b98e3ed-003

[14] Matthias Mauch and Simon Dixon, "pYIN: A fundamental frequency estimator using probabilistic threshold distributions," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP)*, 2014, pp. 659–663. [Online]. Available: http://ieeexplore.ieee.org/document/6853678/

[15] Masanori Morise, Fumiya Yokomori, and Kenji Ozawa, "WORLD: A vocoder-based high-quality speech synthesis system for real-time applications," *IEICE Trans. Inf. and Syst.*, vol. E99.D, no. 7, pp. 1877–1884, 2016. [Online]. Available: https://www.jstage.jst.go.jp/article/transinf/E99.D/7/E99.D_2015EDP7457/_article/-char/en

[16] Colin Raffel and Daniel P. W. Ellis, "Intuitive analysis, creation, and manipulation of MIDI data with `pretty_midi`," in *Proc. 15th Int. Society for Music Information Retrieval Conf. (ISMIR)*, 2014. [Online]. Available: https://github.com/craffel/pretty-midi

[17] Johanna Devaney, Daniel McKemie, and Alex Morgan, "pyAMPACT: A score-audio alignment toolkit for performance data estimation and multi-modal processing," *Proc. 25th Int. Society for Music Information Retrieval Conf. (ISMIR)*, 2024. [Online]. Available: http://arxiv.org/abs/2412.05436