ChoraleWind: An Expressive Wind-Quartet Dataset for End-to-End Rendering from the Neues Thüringer Choralbuch

Axel Berndt[1] **Aida Amiryan-Stein**[1] **Manuel Peters**[2] **Meinard Müller**[2] **Stefan Balke**[2]
[1]Paderborn University [2] International Audio Laboratories Erlangen
axel.berndt@uni-paderborn.de

ABSTRACT

We introduce ChoraleWind, a dataset along with a framework for a reproducible end-to-end rendering from the Neues Thüringer Choralbuch (NTCB). The dataset comprises 311 four-part chorales and covers the full pipeline from symbolic score encoding to performance-level rendition and synthesized audio. ChoraleWind includes a rule-based performance model that generates expressive timing, dynamics, and articulation, including metric and structural accents as well as phrase-end gestures from high-quality MEI encoding of the NTCB chorales, combined with a wind-instrument synthesis based on physical modeling that produces isolated stems and ensemble mixes. The dataset provides aligned symbolic representations, performance annotations, and multitrack audio, enabling systematic training and evaluation of score-to-audio wind-quartet rendering methods under fully controlled conditions. Rather than aiming at state-of-the-art purely data-driven synthesis, ChoraleWind is designed as a transparent and reproducible testbed for studying expressive performance generation, timbre modeling, and evaluation of wind-quartet rendering systems.

1. INTRODUCTION

Recent years have seen growing interest in computational methods for modeling expressive performance and realistic timbre for wind instruments. In contrast to piano music, however, research on wind music is hindered by the scarcity of openly available, systematically annotated datasets. Existing corpora are often limited in size, heterogeneous in instrumentation, or recorded with substantial cross-talk, which makes them difficult to use as training material for data-driven models and challenging to evaluate in a controlled and reproducible manner.

Previous work such as ChoraleBricks [1] has demonstrated the value of curated wind-ensemble recordings aligned with symbolic representations. Producing such datasets, however, is labor-intensive, costly, and difficult to scale, as it requires professional performers, studio environments, and detailed alignment procedures. Synthetic audio, while not matching the acoustic richness of real performances, offers an attractive complementary approach: it enables controlled experiments, supports self-supervised and contrastive learning setups, and allows large-scale training without recording constraints. At the same time, synthetic datasets can serve as transparent and reproducible baselines for studying expressive performance modeling and score-to-audio rendering.

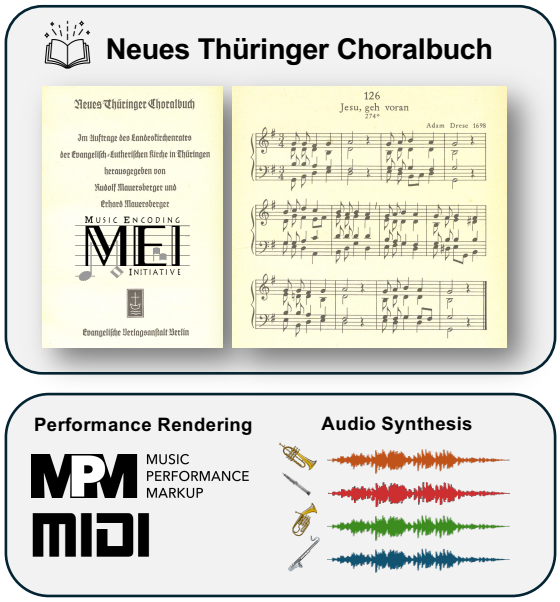

Figure 1. Illustration of the ChoraleWind dataset. Based on high-quality MEI encodings of the Neues Thüringer Choralbuch, we generate expressive renditions using a rule-based performance model. These renditions are then synthesized as multitrack audio for a variety of wind instruments using a synthesizer based on physical modeling.

In this paper, we introduce *ChoraleWind*, an expressive wind-quartet dataset and reproducible end-to-end rendering pipeline based on the "Neues Thüringer Choralbuch" (NTCB). ChoraleWind comprises 311 four-part chorales (**S**oprano, **A**lto, **T**enor, **B**ass) derived from high-quality MEI encodings of the NTCB chorales compiled by Rudolf and Erhard Mauersberger [2, 3]. As illustrated in Figure 1, the dataset covers the full processing chain from symbolic score encoding to expressive performance modeling and synthesized audio.

Expressive timing, dynamics, and articulation are generated using a rule-based performance model that incorporates domain knowledge about chorale performance, including metric and structural accentuation, phrase-end gestures, and controlled humanization. The resulting performance descriptions are rendered into expressive MIDI and synthesized part-wise into multitrack audio using a commercial physical-modeling wind-instrument synthesizer, producing isolated stems for every part as well as

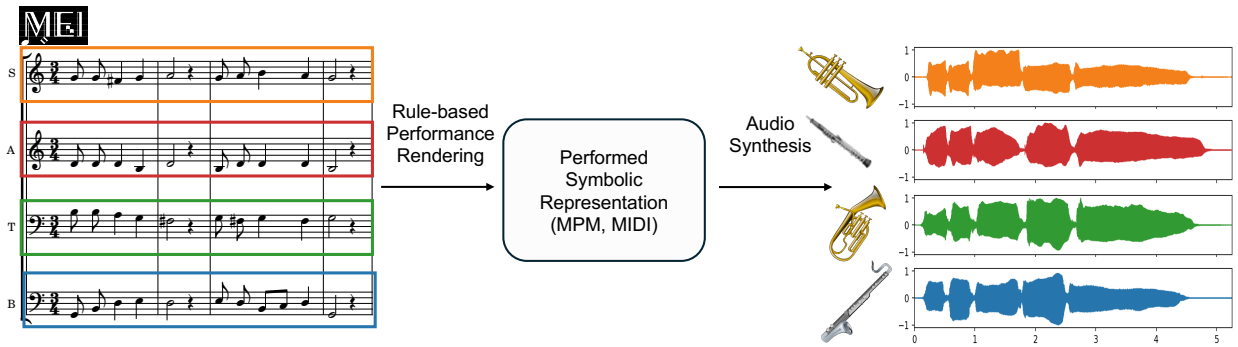

Figure 2. Overview of the rendering pipeline. Starting from the MEI encodings of the SATB chorales, MIDI data is extracted and passed to the performance model. The resulting performance MIDI is then synthesized into multitrack audio recordings.

a large variety of ensemble mixes for 16 different wind instruments. Alongside the audio, ChoraleWind provides aligned symbolic representations and performance-level annotations, enabling systematic training and evaluation of score-to-audio wind-quartet rendering methods under fully controlled and reproducible conditions.

Furthermore, ChoraleWind serves as a superset of ChoraleBricks [1], which contains real recordings of ten selected chorales from the same source with thirteen different instruments and human performers. This connection enables direct and systematic comparisons between synthetic and real performances. Rather than aiming at state-of-the-art purely data-driven synthesis, ChoraleWind is designed as a transparent baseline and controlled testbed for studying expressive performance generation, timbre modeling, and reproducible evaluation of wind-quartet rendering systems.

2. RELATED WORK

Only a limited number of datasets including wind music have been released to date. The most prominent examples are Bach10 and URMP. The Bach10 dataset [4] comprises recordings of ten Bach chorales performed on violin, clarinet, saxophone, and bassoon. The URMP dataset [5] contains 44 pieces spanning a variety of instruments and provides approximately 1.2 hours of isolated multitrack audio. A more recent resource is ChoraleBricks [1], which includes ten chorales with isolated recordings of thirteen instruments, enabling different instrument combinations and ensemble permutations. For datasets involving full orchestra recordings, notable examples include the Beethoven Anechoic dataset [6], the Aalto Anechoic recordings [7], Operation Beethoven [1], and the Spheres dataset [8]. Further symbolic encodings for chorales can be found in the "Chorale Corpus" [9].

An emerging trend, also observed in recent datasets such as EnsembleSet [10], SynthSOD [11], and CocoChorales [12], is the use of synthesizers to generate large amounts of data. While this approach enables the creation of substantial datasets at relatively low cost, the resulting variability in timbre, articulation, and modulation still falls short of what can be captured in real recordings. Nevertheless, Wu et al. [12] demonstrated that synthetically generated four-part ensemble data can improve performance in downstream tasks such as music transcription and music source separation, particularly for instruments with limited data availability.

With respect to performance modeling, two main approaches can be distinguished: rule-based systems and data-driven systems. Rule-based systems represent the more "classical" approach. In these systems, core performance parameters such as tempo and dynamics are modeled using explicit rules derived from the underlying score, for example by identifying phrase boundaries and shaping tempo and velocity curves accordingly. An overview of such systems up to 2013 is provided in [13]. In addition to the rule-based systems, data-driven approaches have emerged as an alternative to rule-based systems. For an overview, we refer the reader to [14, 15]. While these approaches give up much of the explicit control offered by rule-based models, they open up new opportunities for analyzing existing performances and generating novel renditions from symbolic scores. However, given the focus on the chorale scenario and the limited availability of suitable training data from real performances, we adopt a rule-based approach similar to the system proposed in [16], aiming at explicit and reproducible performance renderings.

3. SYMBOLIC REPRESENTATION

Figure 2 provides an overview of the ChoraleWind rendering pipeline, which converts symbolically-encoded four-part chorales into expressive wind-quartet performances using a structured rule set tailored to wind-instrument idioms. The pipeline comprises symbolic preprocessing, rule-based expressive modeling, and audio synthesis. In the following, we start by introducing the Music Encoding Initiative (MEI) format and the applied preprocessing steps.

3.1 MEI Encoding

The musical scores for our dataset have been sourced from the critical digital music edition "Neues Thüringer Choralbuch digital". [2] The "Neues Thüringer Choralbuch" (English: "New Thuringian Chorale Book") [3] is a collection

[1] https://openmusic.academy/revs/JwXCJwcEyAXsJDNsU7DtN3, last access: Dec. 2025.

[2] https://github.com/axelberndt/Neues-Thueringer-Choralbuch-digital , last access: Dec. 2025.

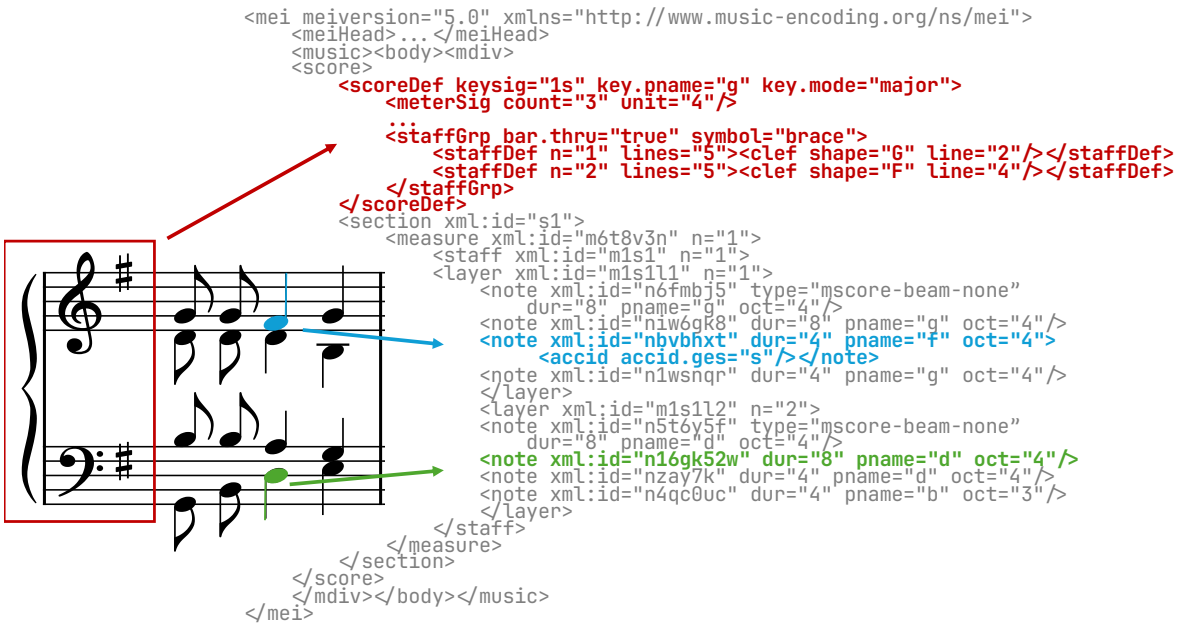

Figure 3. First measure of the chorale "Jesu, geh voran" by Adam Drese and its corresponding MEI encoding. The figure illustrates how score-level information (key, meter, staves) and note-level attributes are represented in MEI.

of more than 300 four-part harmonizations that accompanied the two Protestant chorale books "Deutsches Evangelisches Gesangbuch" and "Evangelisches Kirchengesangbuch". Although these chorale books were succeeded by the current "Evangelisches Gesangbuch" in 1996 and are therefore considered outdated, the chorales of the "Neues Thüringer Choralbuch" remain relevant and in active use due to frequent reprints, such as [17, 18, 19, 20]. Apart from a few exceptions, the vast majority of the harmonizations follow a traditional four-part homophonic cantional style, in which alto, tenor, and bass share the rhythm of the melody, while the "cantus firmus" is consistently placed in the soprano part.

Figure 3 depicts a rendering of the first measure of the chorale "Jesu, geh voran", along with its MEI encoding. Following the original print layout, the four musical parts are encoded across two staves. The upper staff contains tlo he soprano and alto parts as separate layers under a treble clef, while the lower staff similarly encodes tenor and bass under a bass clef. Our MEI encodings follow the international standard (version 5.0 of the "mei-all" schema definition) for digital music editions [2].

Several properties of the MEI encodings are relevant for the subsequent preprocessing steps and are therefore discussed here. The `<meiHead>` section contains detailed metadata, including the (German) song title, book number, composer of the melody, composer of the harmonization, encoder, and license information. The `<music>` section defines one `<mdiv>` (musical division, for example used to separate movements in a symphony) per chorale. Each `<measure>`, `<staff>`, `<layer>`, `<note>`, `<rest>`, and performance instruction carries an `@xml:id` attribute and is therefore addressable.

Tonality and time signature are defined globally at the `<scoreDef>` level (Figure 3, red section), rather than individually per `<staffDef>` as in some other music editions. While tonality remains constant throughout a chorale, time signatures may change—sometimes on a per-measure basis and often without a printed indication. These changes are made explicit in the encoding by inserting a `<scoreDef>` element with an invisible `<meterSig>` between the corresponding measures. Pickup measures are marked with `@metcon="false"` which temporally deactivates metrical conformance. Some chorales, however, do not specify an underlying time signature at all.

Repetitions are encoded using standard repetition bar lines (𝄆 and 𝄇). If the initial section of a chorale is repeated, the opening repetition mark 𝄆 is not printed and therefore not encoded. More complex repetition schemes, such as sections repeated multiple times, are represented using the MEI `<expansion>` mechanism.

3.2 Symbolic Preprocessing

Some preprocessing steps are applied to prepare the symbolic music data for performance generation. For this purpose, we utilize and extend the functionality of the *meico* MEI converter framework [21].

Part separation: Since each musical part is treated individually later on, the two staves are separated into individual parts. Four new staff definitions `<staffDef>` replace the two existing ones in the MEI encoding. Accordingly, each `<measure>` receives new `<staff>` elements, to which the individual `<layer>` elements are reassigned. To remain consistent with the print and common practice, the staff order follows the part order, i. e., (top to bottom) soprano, alto, tenor, bass. As a result, MEI-to-MIDI conversion generates one MIDI track per part in this order.

Addition of IDs: To ensure that all relevant musical elements are addressable, the MEI encoding is checked and missing `@xml:id` attributes are added.

Caesura to breath: The original print, and thus the MEI encoding, use the caesura symbol (') to indicate phrase boundaries. A caesura denotes a "break, pause, or interrup-

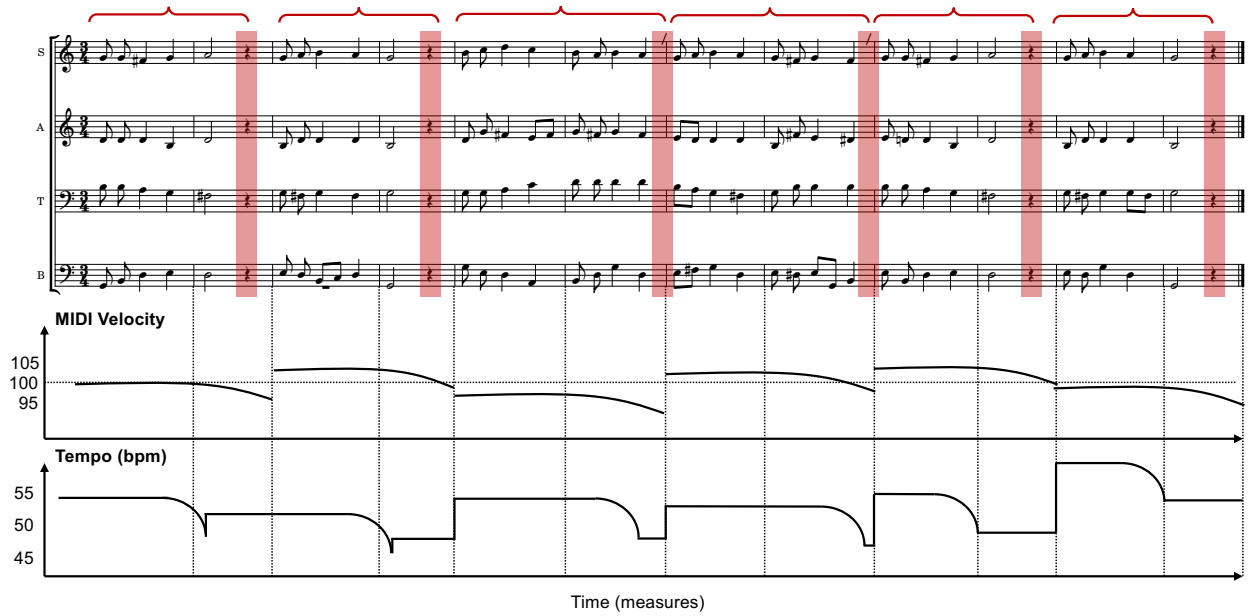

Figure 4. Phrasing mechanism of the performance model illustrated based on the chorale "Jesu, geh voran" by Adam Drese and its harmonization by Rudolf Mauersberger. The top panel shows the four parts (SATB) in individual staff notation, where red arcs indicate the detected phrases and red boxes highlight general rests and caesuras across all parts. The lower panels displays the resulting tempo and velocity curves generated by the performance model.

tion in the normal tempo of a composition"[3] and is commonly interpreted as a breath mark. While tempo modulation at phrase endings is handled by performance modeling, caesuras do not encode the articulatory modifications needed to create a short breathing break. Therefore, all `<caesura>` elements are renamed to `<breath>`.

Breath insertions: Caesura or breath marks are typically omitted where rests in the melody provide a natural breathing opportunity, which is unproblematic. However, they are also missing immediately before repetition end marks (:‖). As these positions always correspond to phrase endings in the dataset but would otherwise go undetected, `<breath>` elements are inserted at the end of the corresponding measures.

Score expansion: To ensure that repetitions are rendered in performance, the score is expanded into a through-composed form. Repeated sections are copied and inserted into the musical sequence, with all copied elements receiving unique IDs and all ID references updated accordingly.

Fix fractured measures: Repetition marks are encoded as bar lines (e. g., `<measure n="5" right="rptend">`) but are often placed within a measure. This results in fractured measures: a shortened measure before the repetition mark and a remaining segment after it or at the start of the repeated section. Performance modeling would interpret these as time signature changes and apply inappropriate metrical accentuation patterns, although the logical time signature remains unchanged. To prevent this, we suppress all subsequent time signature changes that sum to their preceding time signature.

In the ChoraleWind dataset, we provide several files that represent both the score and its rendered performance. These include the original MEI file, which contains two systems (soprano and alto in the first system, tenor and bass in the second), as well as a preprocessed MEI version in which each part is placed in a separate system. In addition, we provide an MPM file encoding the performance parameters, an MSM file that serves as an intermediate representation for exporting MEI to MIDI, and, finally, a performed MIDI file.

[3] See MEI Guidelines, `https://music-encoding.org/guidelines/v5/elements/caesura.html`, last access: Dec. 2025.

4. PERFORMANCE RENDERING

Expressive performance parameters such as timing, dynamics, and articulation, are generated using curated rules that emulate typical wind-instrument ensemble practice. Performance decisions are based on musical properties and modeled in the Music Performance Markup (MPM) format [22]. The MPM API, part of the *meico* framework, renders these into expressive MIDI sequences. This process also produces augmented MEI encodings, adding each note's MIDI velocity (`@vel`) and onset/offset times in milliseconds (`@tstamp.real`, `@tstamp2.real`). Figure 4 shows the sheet music and the resulting velocity and tempo curves for the chorale "Jesu, geh voran". In the following, we describe the underlying rules in detail.

Articulations: During conversion, all articulation instructions from the MEI score are placed into part-specific MPM `<articulationMap>`s and then merged into a global map. Some articulations, particularly breath instructions (former caesuras; cf. Section 3.2), apply to all parts. This ensures all parts breathe together, not just the soprano. The MPM API provides default definitions for standard articulations, which are slightly adjusted, e. g., the breath articulation shortens note duration by 200 ms to allow a break. Text-based articulations are supported but not used here due to the absence of lyrics in most sources.

Metrical accentuation: We implemented standard ac-

#	Instrument	Abbr.	Parts	Dur. (hh:mm:ss)
1	Flute	fl	S	03:10:31
2	Oboe	ob	S	03:10:31
3	Soprano Saxophone	ss	S	03:10:31
4	Alto Saxophone	as	S, A	06:21:02
5	Clarinet	cl	S, A	06:21:02
6	Flugelhorn	fh	S, A	06:21:02
7	Trumpet	tp	S, A	06:21:02
8	English Horn	eh	A	03:10:31
9	Tenor Saxophone	ts	A, T	06:21:02
10	Baritone	bar	T, B	06:21:02
11	Bass Clarinet	bcl	T, B	06:21:02
12	Baritone Saxophone	bs	T, B	06:21:02
13	Bassoon	bsn	T, B	06:21:02
14	French Horn	fho	T, B	06:21:02
15	Trombone	tb	T, B	06:21:02
16	Tuba	tba	B	03:10:31
		$\sum$	27	85:44:14

Table 1. Overview of the synthesized instruments in ChoraleWind, including abbreviations, the SATB parts each instrument covers, and the overall duration of isolated audio tracks.

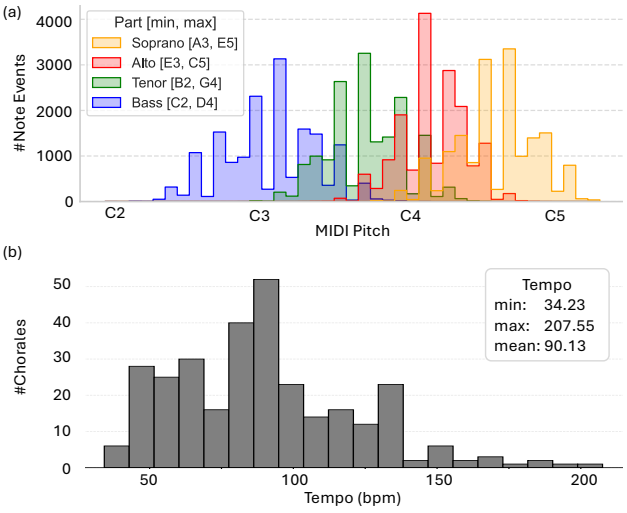

Figure 5. Histograms of (a) the musical pitches for the four parts (SATB) and (b) the overall tempi of the chorales.

centuation patterns for all relevant time signatures, a pattern handling pieces which start with a shortened measure, and a fallback pattern for other cases. Following historical notation conventions, the strength of the accent depends on two features. First, if no time signature is defined, no accentuation is applied. Otherwise, the bar line method is consulted: absent bar lines or "takt" lines [4] deactivate accentuation. Mensural lines between staves indicate a latent time signature, so accents are scaled to 50% (± 8 MIDI velocity units [5]) of the full accentuation (± 16 MIDI velocity units) used with regular bar lines. Depending on the synthesizer, this scaling factor may be adjusted, as each interprets velocity differently [23].

Basic tempo and dynamics: The baseline tempo of each chorale is derived from the average phrase length, assuming that a phrase is sung and played on a single breath. We employ a reference breath cycle duration of approximately 4.3 seconds. However, while this works well for the vast majority of chorales in our dataset, there are several outlier cases with rather short or long phrases or missing indications for phrase boundaries (rendering the whole piece in one extremely long phrase, technically speaking). Furthermore, we add a subtle random offset of ± 3 bpm to the basic tempo of each phrase to emulate natural human variation. This is modeled as a correlated variation with a Brownian distribution to avoid erratic behavior. For dynamics we assume a basic loudness of 100 MIDI velocity units. Like with basic tempo, basic loudness is also varied with each new phrase, this time in a range of ± 5 velocity units via an (uncorrelated) uniform distribution.

Phrasing: Breath marks and general rests define phrase boundaries, guiding tempo and dynamics shaping. However, general rests can be unreliable: some chorales include them only in the melody while other parts continue playing. Rests shorter than one beat are also ignored. As shown in Figure 4, towards phrase endings, dynamics decrease to 70% of the basic MIDI velocity and tempo slows down to 95% of the base tempo (ritardando); by the end of the verse, tempo reaches 78%.

Randomization: Each note's onset (via MPM's Compensating Triangle Distribution), velocity (Gaussian), and duration (uncorrelated Triangle) are individually randomized per musical part. This emulates the natural imprecision of human performances, commonly called "humanization", and prevents mechanically exact renderings. Offset magnitudes can be adjusted to reflect the ensemble's assumed professionalism.

Velocity range normalization: If calculated velocities fall outside the MIDI range, we apply a nonlinear compression that preserves the relative dynamic structure while ensuring valid values.

The output of the performance rendering model is an MPM file, which describes the performance in a structured format. Velocity and tempo contours are represented as time-continuous Bézier functions. To control subsequent synthesizers, these MPM files are exported as MIDI files, providing a widely supported interface while preserving the expressive timing and dynamics.

5. AUDIO SYNTHESIS

For each chorale, we generate an expressive MIDI file containing all parts as separate MIDI tracks and synthesize audio using a commercially available physical-modeling wind instrument engine, for a first overview on the topic we refer to e. g., [24]. [6] The engine is provided as a VST plugin, which we load into the digital audio workstation Reaper. [7] In the future, other instrument-specific synthesis approaches such as [25] could be considered. [8] Within Reaper, we prepare a project in which each instrument is assigned to a separate track, as listed in Table 1. For every chorale, we render 27 isolated tracks with 16 unique

[4] "short bar line through a subset of staff lines", according to the MEI Guidelines, `https://music-encoding.org/guidelines/v5/data-types/data.BARMETHOD.html` , last access: Dec. 2025.

[5] MIDI velocity is a unitless, integer-valued parameter in [0:127] that encodes the note intensity.

[6] `https://audiomodeling.com/swam-engine/`, last access: Dec. 2025.

[7] `https://www.reaper.fm/`, last access: Dec. 2025.

[8] As a freely available alternative, SoundFonts could be employed; however, our preliminary experiments have yielded mixed results.

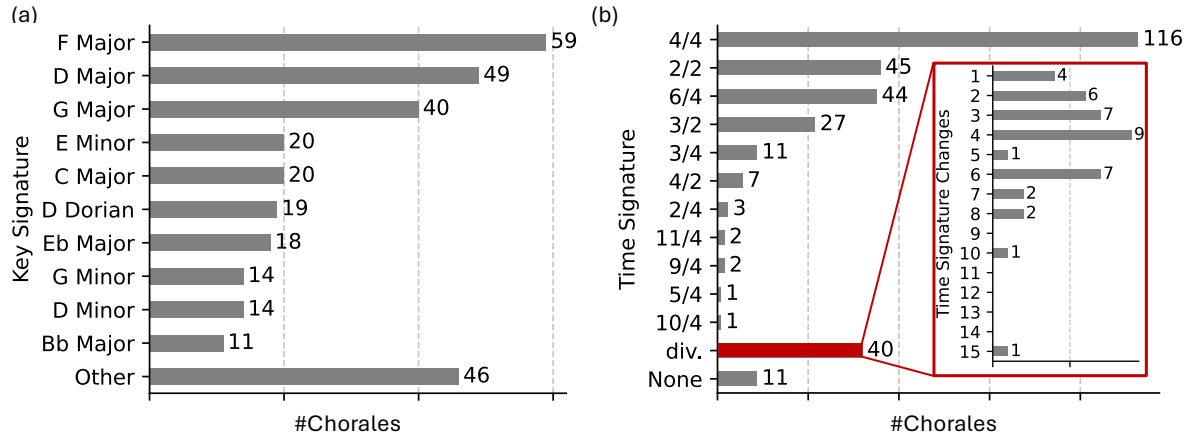

Figure 6. Distribution of key and time signatures in the dataset. (a) Distributions of top 10 key signatures. The category "Other" aggregates all remaining, less frequent keys. (b) Distributions of time signatures. The category "div." indicates that the chorale was notated with time signature changes, as depicted in the red box. The category "None" indicates that no time signature was present in the MEI file.

instruments. Note that some instruments can cover multiple parts; e. g., the trumpet is used for both the soprano and alto part. The mapping from instruments to parts is determined by matching playable pitch ranges and by the authors' practical experience. All instruments are tuned to a tuning frequency of 442 Hz.

The entire synthesis process is automated using Reaper's internal scripting language, ReaScript, which is accessed via its Lua API. Using Lua requires no additional setup, and the script can be executed directly from within Reaper. Compared to sample-based synthesizers, physical-modeling synthesizers require less memory and offer greater flexibility in adapting the generated sound. In our initial renderings, all synthesizer parameters are kept at their default settings without any reverb. Executing the full synthesis pipeline took approximately 90 minutes on a notebook equipped with an AMD Ryzen AI 5 Pro processor and 32 GB of RAM. The Reaper project file, as well as the Lua script, are available on the accompanying website. [9]

The isolated parts were intentionally exported without artificial reverberation. Although reverberation and stereo panning are important for perceptually realistic ensemble sound, ChoraleWind is designed as a machine learning dataset. This requires precise alignment between audio and annotations, which reverberation would compromise through temporal smearing that obscures note onsets and weakens the correspondence to symbolic labels. By providing dry signals, we prioritize annotation fidelity and controllability, while allowing users to add reverberation and spatialization as needed.

As listed in Table 1, rendering a single instrument for one part yields approximately 3 hours of audio. Rendering all parts and instruments results in approximately 86 hours of isolated audio tracks. Considering the possible four-part ensemble combinations (7 Soprano, 6 Alto, 7 Tenor, and 7 Bass) for 311 chorales, this theoretically amounts to approximately 640,000 distinct ensemble configurations.

In addition to providing the complete dataset for download via the accompanying website, we offer multitrack recordings of ten chorales that can be explored using the `trackswitch.js` player [26]. These chorales are identical to those used in the ChoraleBricks dataset and therefore enable direct comparisons between real performances and the corresponding synthetic renderings. Moreover, folder names and file naming conventions are consistent with ChoraleBricks, allowing the use of the same existing Python toolbox for sampling ensembles from the dataset. Further information is available on the website.

[9] `https://audiolabs-erlangen.de/resources/MIR/2026-ChoraleWind`, last access: Mar. 2026.

6. STATISTICS AND ANNOTATIONS

In the following, we summarize essential statistics for the ChoraleWind dataset regarding pitch, tempo, key, and time signatures. Figure 5a shows the distribution of pitches for the four SATB parts in MIDI notation. The lowest pitch observed is C2 in the bass part, while the highest pitch is E5 in the soprano part. The histogram indicates that each part occupies a characteristic pitch range; however, the ranges overlap in the boundary regions.

Figure 5b illustrates the distribution of tempi across the entire chorale corpus, directly extracted from the metadata of the MPM's tempo maps. The minimum tempo is 34.23 bpm in "O Traurigkeit, o Herzeleid." This chorale is characterized by many short phrases which in combination with the breath cycle duration of 4.3 seconds leads to a very slow tempo. In contrast, the maximum tempo of 207.55 bpm occurs in "Herr, du hast alles, Himmel und Erden", which contains extended passages without breaks or caesuras. Overall, the average tempo across the corpus is 90.13 bpm.

Figure 6a shows the distribution of the ten most frequent key signatures in the corpus. Of these, six correspond to major modes and four to minor modes (including D Dorian). Moreover, 255 of the 311 chorales (approximately 82%) are in keys with at most three sharps or flats, making them particularly suitable for performance by wind instruments. The category "Other" aggregates less frequent modes, such as E Phrygian or F Mixolydian, and illustrates the level of detail captured in the corpus annotations.

Figure 6b shows the distribution of time signatures in the corpus. A total of 116 chorales (approximately 37%) are notated in 4/4, followed by 45 chorales (approximately 14%) in 2/2. In addition, 40 chorales (approximately 13%)

exhibit changing time signatures. This is mainly due to the fact that some chorales originally do not specify a time signature; representing these pieces in modern notation required the introduction of changing time signatures. However, in 11 chorales no time signature is given.

The combination of MEI, MPM/MSM, and MIDI files provides rich annotation and performance-level information for the audio renderings. The MEI files include detailed score metadata (e.g., title, composer, key) as well as full note-level information, enabling analyses of melody, harmony, and part interactions. The MPM files encode expressive parameters such as tempo and articulation, allowing extraction of continuous velocity and tempo trajectories for synthesis or modeling. The MIDI files offer a convenient representation for machine learning tasks, providing precise onset and pitch information for each note event.

7. CONCLUSIONS AND POTENTIAL APPLICATIONS

We introduced *ChoraleWind*, a large-scale expressive wind-quartet dataset derived from the *Neues Thüringer Choralbuch* (NTCB), along with a fully reproducible end-to-end rendering pipeline. It provides aligned scores, performance annotations, and multitrack audio for 311 four-part chorales, covering the full chain from MEI to expressive MIDI and physical-modeling synthesis. The dataset enables controlled evaluation of score-to-audio rendering through precise ground-truth annotations of pitch, timing, dynamics, and articulation, which are hard to obtain from real recordings. By combining a rule-based performance model with physical-modeling synthesis, ChoraleWind captures key aspects of expressive wind-quartet performance while remaining fully reproducible and extensible.

Future extensions of the performance model include incorporating sung text to inform expressive timing and emotional coloring, as well as modeling long-range phrasing across multiple phrases. Currently, velocity and tempo contours are sampled at note onsets and held constant over each note; more advanced representations of continuous contours could enable more expressive and musically coherent synthesis.

Although synthetic, the dataset supports a broad range of applications, such as audio synthesis, audio–score alignment, and automatic music transcription (AMT). For audio synthesis, ChoraleWind can serve as a baseline approach for score-to-audio models, expressive MIDI generation, timbre modeling, and ensemble rendering, e. g., using DDSP-based [27] or diffusion-based [28] approaches to synthesize wind instrument sounds directly from score. For audio–score alignment [29, 30], ChoraleWind allows evaluation at the instrument level. This granularity enables detailed analyses of alignment behavior, such as identifying which instruments are prioritized (e. g., trumpets in the soprano or tuba in the bass), yielding insights that support the refinement of alignment methods. In AMT, early work focused primarily on piano recordings [31]. While recent Transformer-based models [32] extend transcription to other instruments, performance on diverse datasets like URMP [5] remains limited. With its varied instrumentation and modular design, ChoraleWind provides a valuable resource for evaluating and improving AMT models across a broader range of instruments.

Its compatibility with *ChoraleBricks* [1] enables direct comparisons between synthetic and real performances, establishing a transparent reference point for evaluating methods and advancing research across the aforementioned applications.

Acknowledgments

The music encoding is supported by KreativInstitut.OWL: a consortium consisting of Ostwestfalen-Lippe (OWL) University of Applied Sciences and Arts, Detmold University of Music, and Paderborn University, funded by the Ministry of Economic Affairs, Industry, Climate Action and Energy of the State of North Rhine-Westphalia, Germany. The International Audio Laboratories Erlangen are a joint institution of the Friedrich-Alexander-Universität Erlangen Nürnberg (FAU) and Fraunhofer Institute for Integrated Circuits IIS. MP, MM, and SB are supported by the Deutsche Forschungsgemeinschaft (DFG; German Research Foundation) under grant numbers 500643750 (MU 2686/15-1) and 555525568 (MU 2686/18-1).

8. REFERENCES

[1] S. Balke, A. Berndt, and M. Müller, "ChoraleBricks: A Modular Multitrack Dataset for Wind Music Research," *Transaction of the International Society for Music Information Retrieval (TISMIR)*, vol. 8, no. 1, pp. 39–54, 2025.

[2] A. Hankinson, P. Roland, and I. Fujinaga, "The Music Encoding Initiative as a Document-Encoding Framework," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Miami, Florida, USA, October 2011, pp. 293–298.

[3] R. Mauersberger and E. Mauersberger, *Neues Thüringer Choralbuch*, 6th ed., H. Peter, Ed. Berlin, Germany: Evangelische Verlagsanstalt Berlin, 1990.

[4] Z. Duan and B. Pardo, "Soundprism: An Online System for Score-Informed Source Separation of Music Audio," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 6, pp. 1205–1215, 2011.

[5] B. Li, X. Liu, K. Dinesh, Z. Duan, and G. Sharma, "Creating a Multitrack Classical Music Performance Dataset for Multimodal Music Analysis: Challenges, Insights, and Applications," *IEEE Transactions on Multimedia*, vol. 21, no. 2, pp. 522–535, 2019.

[6] C. Böhm, D. Ackermann, and S. Weinzierl, "A multi-channel anechoic orchestra recording of Beethoven's Symphony no. 8 op. 93," *Journal of the Audio Engineering Society*, vol. 68, no. 12, pp. 977–984, 2021.

[7] M. Schedl, D. Hauger, M. Tkalčič, M. Melenhorst, and C. C. S. Liem, "A Dataset of Multimedia Material about Classical Music: PHENICX-SMM," in *Proceedings of International Workshop on Content-Based Multimedia Indexing (CBMI)*, 2016, pp. 1–4.

[8] J. Garcia-Martinez, D. Diaz-Guerra, J. Anderson, R. Falcon-Perez, P. Cabañas-Molero, T. Virtanen, J. J. Carabias-Orti, and P. Vera-Candeas, "The Spheres Dataset: Multitrack Orchestral Recordings for Music Source Separation and Information Retrieval," *arXiv preprint arXiv:2511.21247*, 2025.

[9] K. Gerhardt and M. Kirsch, "Choralbücher from Northern Germany c. 1800 - a Dataset for Studies in Hymnology, Music Culture and Figured Bass," *Transactions of the International Society for Music Information Retrieval*, vol. 7, no. 1, pp. 1–10, 2024.

[10] S. Sarkar, E. Benetos, and M. Sandler, "EnsembleSet: A New High Quality Dataset for Chamber Ensemble Separation," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Bengaluru, India, 2022, pp. 625–632.

[11] J. Garcia-Martinez, D. Diaz-Guerra, A. Politis, T. Virtanen, J. J. Carabias-Orti, and P. Vera-Candeas, "SynthSOD: Developing an Heterogeneous Dataset for Orchestra Music Source Separation," *IEEE Open Journal of Signal Processing*, vol. 6, pp. 129–137, 2025.

[12] Y. Wu, J. Gardner, E. Manilow, I. Simon, C. Hawthorne, and J. Engel, "The Chamber Ensemble Generator: Limitless High-Quality MIR Data via Generative Modeling," 2022. [Online]. Available: http://arxiv.org/abs/2209.14458

[13] A. Kirke and E. R. Miranda, *An Overview of Computer Systems for Expressive Music Performance*. London: Springer London, 2013, pp. 1–47. [Online]. Available: https://doi.org/10.1007/978-1-4471-4123-5_1

[14] C. E. Cancino-Chacón, M. Grachten, W. Goebl, and G. Widmer, "Computational Models of Expressive Music Performance: A Comprehensive and Critical Review," *Frontiers in Digital Humanities*, vol. Volume 5 - 2018, 2018.

[15] A. Kirke and E. R. Miranda, *Performance Creativity in Computer Systems for Expressive Performance of Music*. Cham: Springer International Publishing, 2021, pp. 521–584. [Online]. Available: https://doi.org/10.1007/978-3-030-72116-9_19

[16] A. Friberg, R. Bresin, and J. Sundberg, "Overview of the KTH rule system for musical performance," *Advances in Cognitive Psychology*, vol. 2, no. 2, p. 145, 2006.

[17] S. Bauer, K. Klek, and B. Reich, Eds., *Orgelsätze zum Evangelischen Gesangbuch*. Munich, Germany: Verband Evangelische Kirchenmusik in Württemberg, Strube Verlag, 1996, Ausgabe für die Evangelische Landeskirche in Württemberg.

[18] M. Heinig, H. Rau, and D. Schuberth, Eds., *Orgelbuch zum Evangelischen Gesangbuch*. Kassel, Germany: Evangelische Kirche Deutschland, Bärenreiter-Verlag, 1993.

[19] Sächsische Posaunenmission, Ed., *Wachet auf – Ein Posaunenbuch*, 9th ed. Radebeul, Germany: Sächsische Posaunenmission e.V., 2002.

[20] F. Plewka, Ed., *Der Tag ist nicht mehr fern: Bläsermusik zum Weihnachtsfestkreis*. Munich, Germany: Posaunenwerk der Evangelischen Kirche in Mitteldeutschland, Strube Verlag, 2023.

[21] A. Berndt, S. Waloschek, and A. Hadjakos, "Meico: A Converter Framework for Bridging the Gap between Digital Music Editions and its Applications," in *Proceedings of the Audio Mostly Conference*. Wrexham, North Wales, UK: ACM, 2018.

[22] A. Berndt, "The Music Performance Markup Format and Ecosystem," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Bengaluru, India, 2021, pp. 50–57.

[23] R. B. Dannenberg, "The Interpretation of MIDI Velocity," in *Proc. of the Int. Computer Music Conf. (ICMC)*. Tulane University, New Orleans, USA: International Computer Music Association, Nov. 2006, pp. 193–196.

[24] D. H. Keefe, "Physical Modeling of Wind Instruments," *Computer Music Journal*, vol. 16, no. 4, pp. 57–73, 1992.

[25] I. Derenyi and R. B. Dannenberg, "Synthesizing Trumpet Performances," in *Proceedings of the International Computer Music Conference (ICMC)*, Ann Arbor, Michigan, USA, 1998.

[26] N. Werner, S. Balke, F.-R. Stöter, M. Müller, and B. Edler, "trackswitch.js: A Versatile Web-Based Audio Player for Presenting Scientific Results," in *Proceedings of the Web Audio Conference (WAC)*, London, UK, 2017.

[27] J. Engel, L. Hantrakul, C. Gu, and A. Roberts, "DDSP: Differentiable Digital Signal Processing," in *Proceedings of the International Conference on Learning Representations (ICLR)*, Virtual, 2020. [Online]. Available: https://openreview.net/forum?id=B1x1ma4tDr

[28] B. Maman, J. Zeitler, M. Müller, and A. H. Bermano, "Multi-Aspect Conditioning for Diffusion-Based Music Synthesis: Enhancing Realism and Acoustic Control," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 33, pp. 68–81, 2025.

[29] R. B. Dannenberg and C. Raphael, "Music Score Alignment and Computer Accompaniment," *Communications of the ACM, Special Issue: Music Information Retrieval*, vol. 49, no. 8, pp. 38–43, 2006.

[30] M. Müller, A. Arzt, S. Balke, M. Dorfer, and G. Widmer, "Cross-Modal Music Retrieval and Applications: An Overview of Key Methodologies," *IEEE Signal Processing Magazine*, vol. 36, no. 1, pp. 52–62, 2019.

[31] E. Benetos, S. Dixon, Z. Duan, and S. Ewert, "Automatic Music Transcription: An Overview," *IEEE Signal Processing Magazine*, vol. 36, no. 1, pp. 20–30, 2019.

[32] B. Maman and A. H. Bermano, "Unaligned Supervision for Automatic Music Transcription in The Wild," in *Proceedings of the International Conference on Machine Learning (ICML)*, Baltimore, Maryland, USA, 2022, pp. 14 918–14 934.