Generative Bach Chorale Harmonizations

Stanford University

Teresa Dayrit, Jonathan Jia-An Mak {tedayrit, jmak}@stanford.edu

Goals

- Generate accurate Bach Chorales using various AI techniques and sampling techniques
- Chordal approach rather than note based, and improved sampling to fix problem of "mode collapse" and dull character
- Compare to traditional Bach Chorales using metrics such as rhythm, flow, and technical accuracy.

Related Works

- Generation initially iterative and non-learned, using techniques such as *n*-grams [1].
- Soon enough, Markov Chains and other probabilistic models were used for various situations such as for different moods [2].
- Deep learning with RNNs/LSTMs become gold standard by featurizing the melody and scaling up to determine overarching chord progressions and harmonies [3].
- Novel experimental techniques utilizing GANs arise by dividing MIDI input into bars to retain timestamp information before generating melodies through conditional sampling [4].

Data and Preprocessing

- Raw data consisted of 371 complete Bach Chorales utilizing Music21, a Python music library to access notes and chords
- Separated each chorale into a major/minor bucket, as well as calculated scale degree difference for each note, which is "distance" from current chord
- Created CSP dictionaries for feature factors
- Created fill dictionary based off of transition frequency for chord progressions
- Input/Output: MIDI files separated into chords/Generated MIDI files
- Challenges: Chord representation and respective weighting

Methods

- **Baseline:** Random generation of chords
- ***N*-grams:** Using *n*-grams approach to keep track of previous chords and overarching phrasing using uniform cost search to find lowest cost
- **Fill Sampling:** Probabilistic sampling using log likelihood and also using lowest cost, assigned by $1/m$, where m is the amount of times the chord appears in that specific transition.
- **Features:** Issued a survey examining several musical features for objective weighting
- **Beam Search:** Reduce run time by choosing only top k fills that have the lowest cost
- **Constraint Satisfaction Factors:**
  - Harmonic and technical correctness
    - Follow typical Baroque rules to the same degree Bach did
  - Secondary dominance
    - Correct emphasis on secondary dominant chords, which are harmonic progressions used by composers to inject flair although technically "incorrect"
  - Voice leading
    - Stronger emphasis on having a fluid melody by preventing chords from moving outside of 3 scale degrees to reduce "nonmusical" changes
- **Gibbs Sampling:** Utilized MCMC sampling to mimic "guessing" the following chord
- **Final Model:** Combination of *n*-grams, beam search, CSPs to mimic best loss

$$\mathcal{L}_{\mathbf{x}\sim\theta} = \sum_{i=1}^{N} \alpha(r) + \beta(w) + \gamma(c) + \delta(h)$$

Equation 1: Loss function over top 4 features

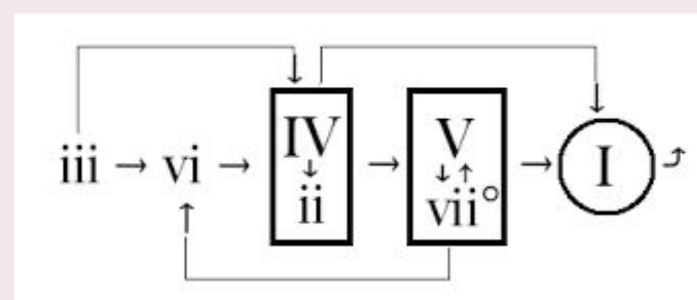


Figure 1: Finite State Diagram for technical correctness

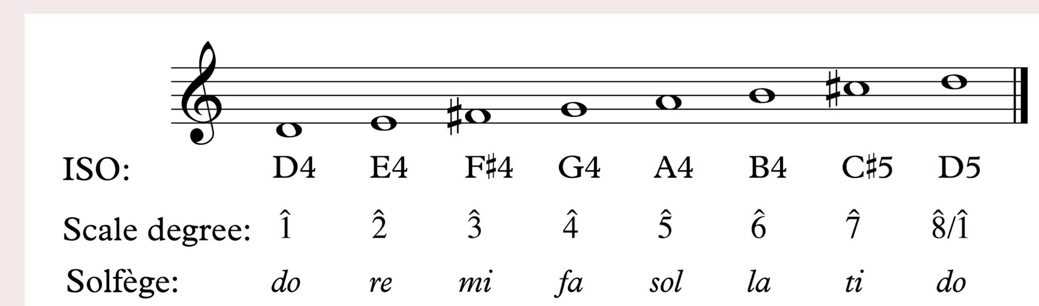
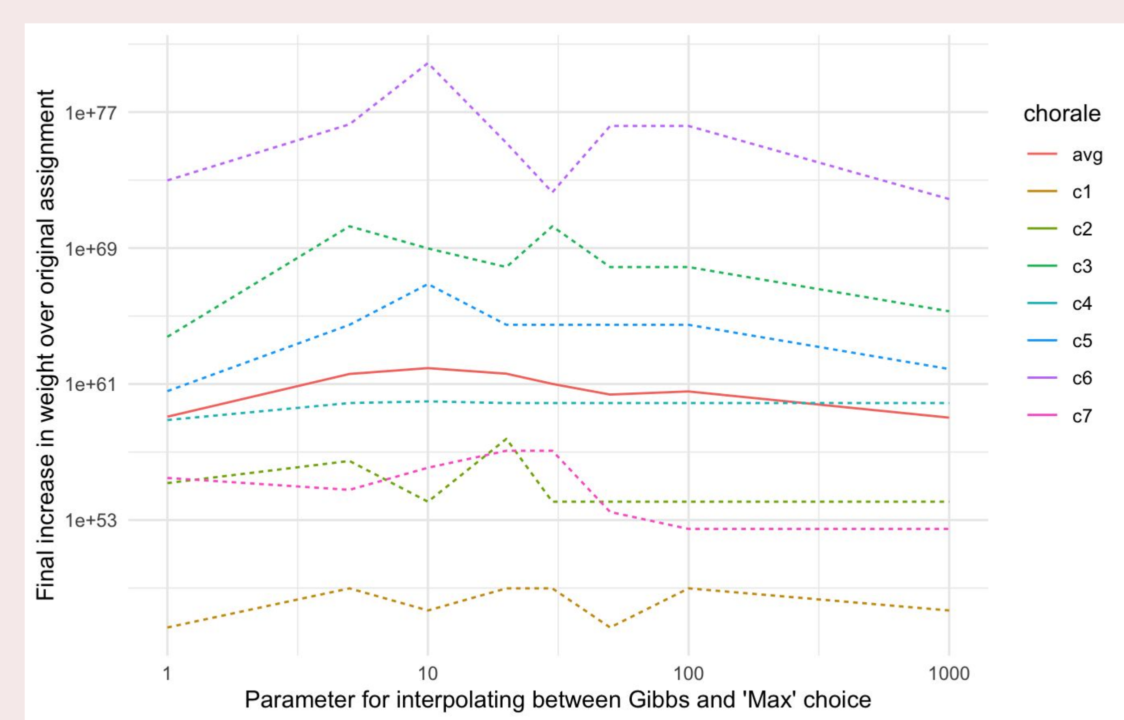


Figure 2: Scale Degree Separation governs frequency for selecting fills

Results

Figure 3: Baseline Chorale (Randomly Generated)

Figure 4: Trigram Chorale (Lowest Cost)



Weighted CSP results for different methods of sampling. For most chorales, it is best to augment the weight of the "best" solution by a factor of ~10, so as not to lead the harmonization in a sideways direction. However, overweighting the best assignment can get the CSP stuck in a local optimum.

Model Results

Model Name	Repetition	Weak Beats	Incorrectness	Rhythm	Loss
Baseline (Random)	0.35	0	0.57	0.87	0.460
Unary (Probabilistic)	0.85	0.12	0.51	0.75	0.561
Unary (Lowest Cost)	0.40	0.04	0.51	0.92	0.453
Bigram	0.83	0.11	0	0.81	0.305
Trigram	**0.73**	**0.09**	**0**	**0.75**	**0.271**
Tetragram	0.74	0.09	0	0.74	0.273
Bach	0.40	0.01	0	0.7	0.169

Discussion

- Initial CSP assignment generation using trigrams; has the lowest loss since it is able to capture the 3-beat harmonic phrasing (predominant, dominant, tonic) typically found in Baroque/Bach music
- Pure Gibbs Sampling injects too much noise, especially since a single poorly sampled variable then affects all of the following generations
- Interpolating between Gibbs and 'Max' sampling maximizes weights

Future Work

- Tune Gibbs Sampling by reweighting probabilities
- Utilize auxiliary variables in CSP to allow for n-ary factors involving more diverse harmonic rhythm

References

1. Cheng, Heng-Tze, et al. "Automatic chord recognition for music classification and retrieval." 2008 IEEE International Conference on Multimedia and Expo. IEEE, 2008.
2. Ramanto, Adhika Sigit, and Nur Ulfa Maulidevi. "Markov Chain Based Procedural Music Generator with User Chosen Mood Compatibility." International Journal of Asia Digital Art and Design Association 21.1 (2017): 19-24.
3. Lim, Hyungui, Seungyeon Rhyu, and Kyogu Lee. "Chord generation from symbolic melody using BLSTM networks." arXiv preprint arXiv:1712.01011 (2017).
4. Yang, Li-Chia, Szu-Yu Chou, and Yi-Hsuan Yang. "MidiNet: A convolutional generative adversarial network for symbolic-domain music generation." arXiv preprint arXiv:1703.10847 (2017).