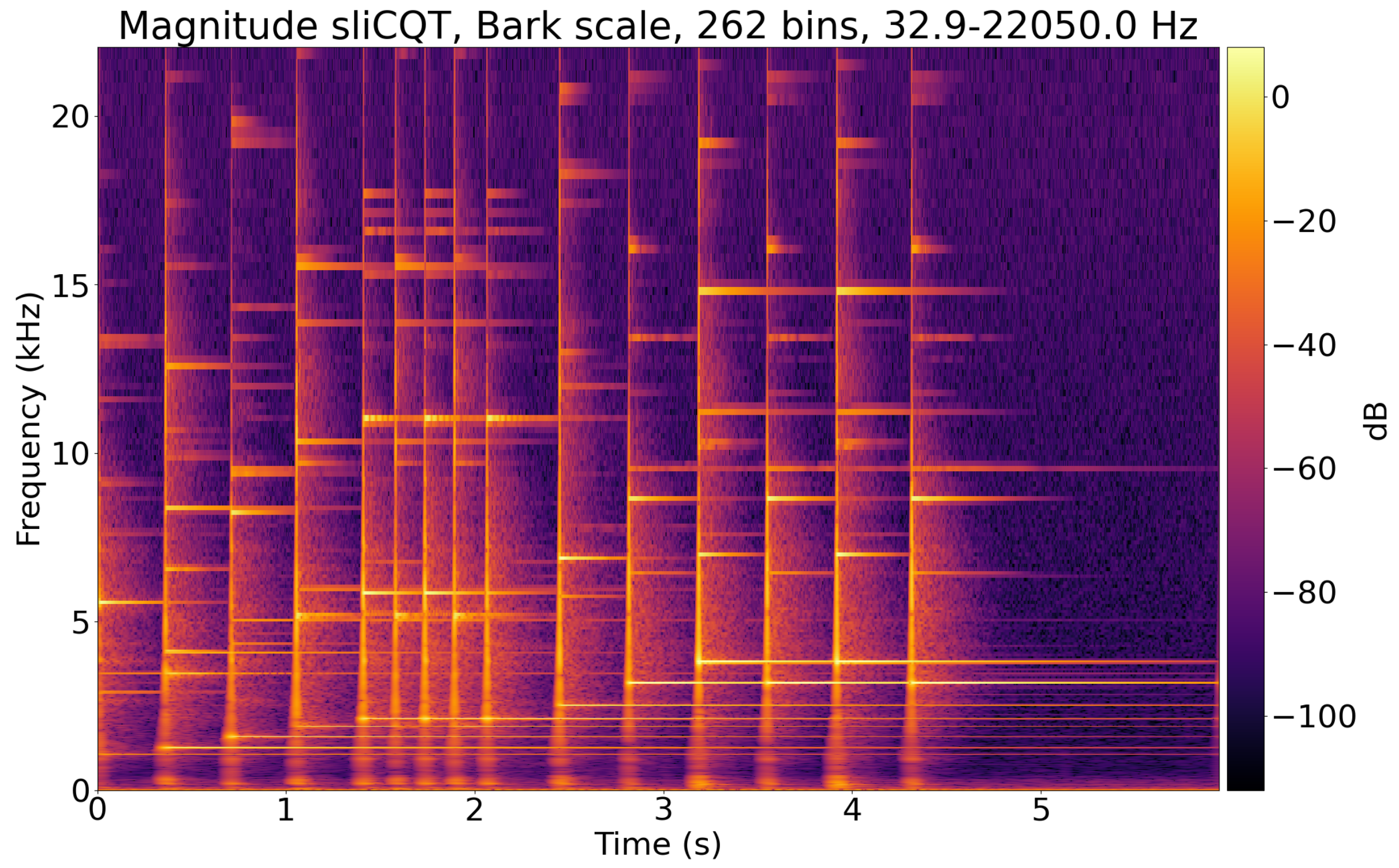
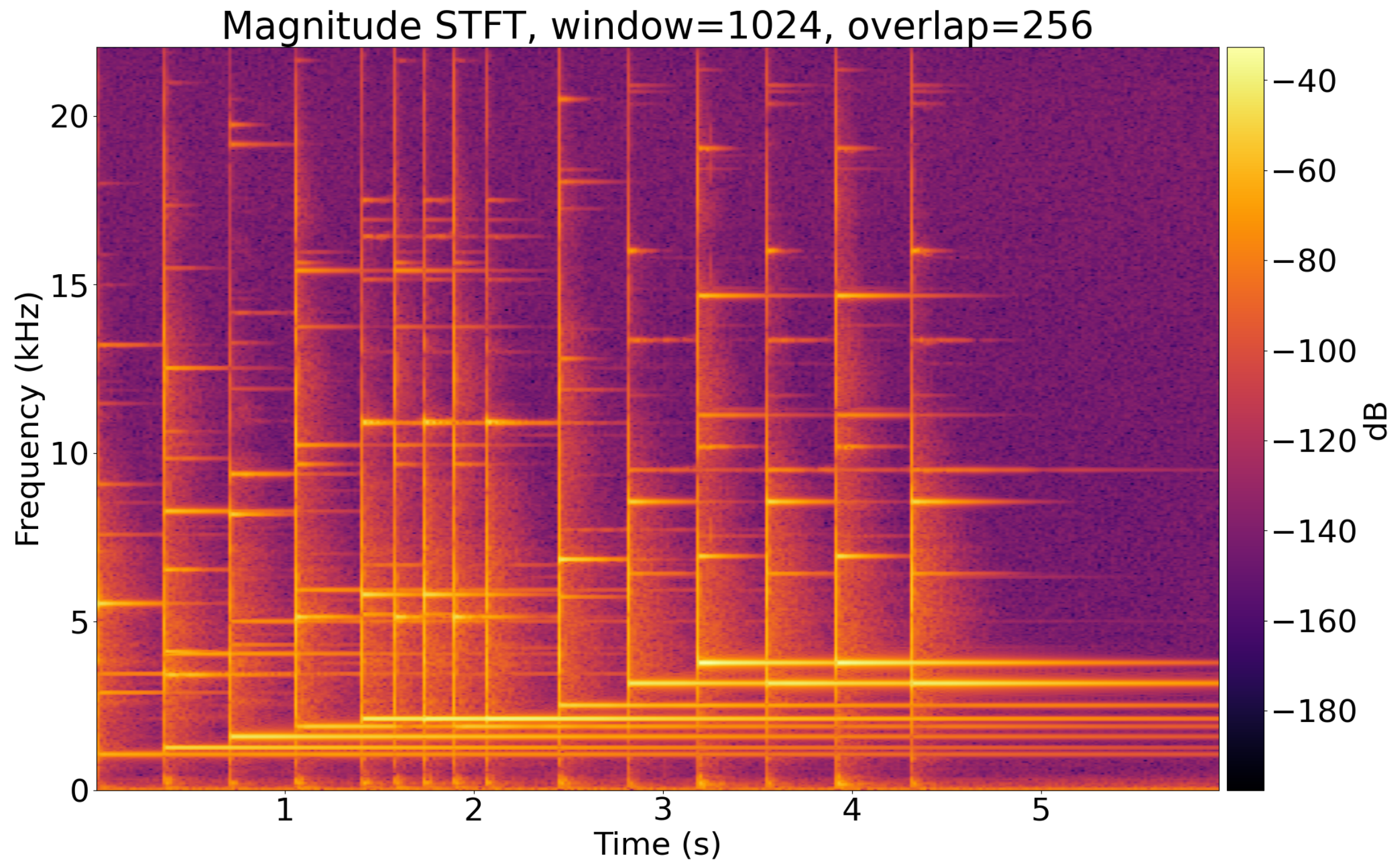
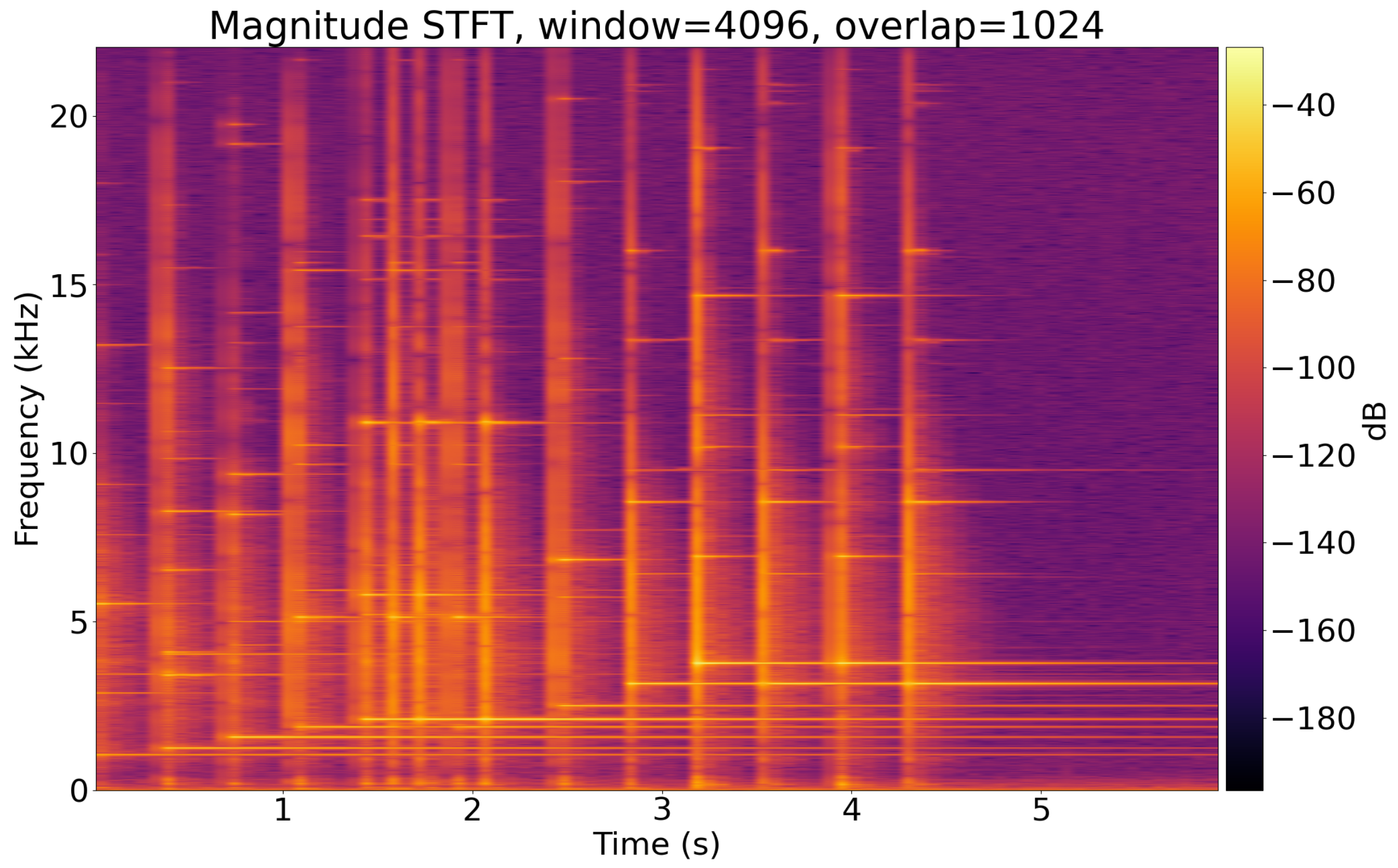
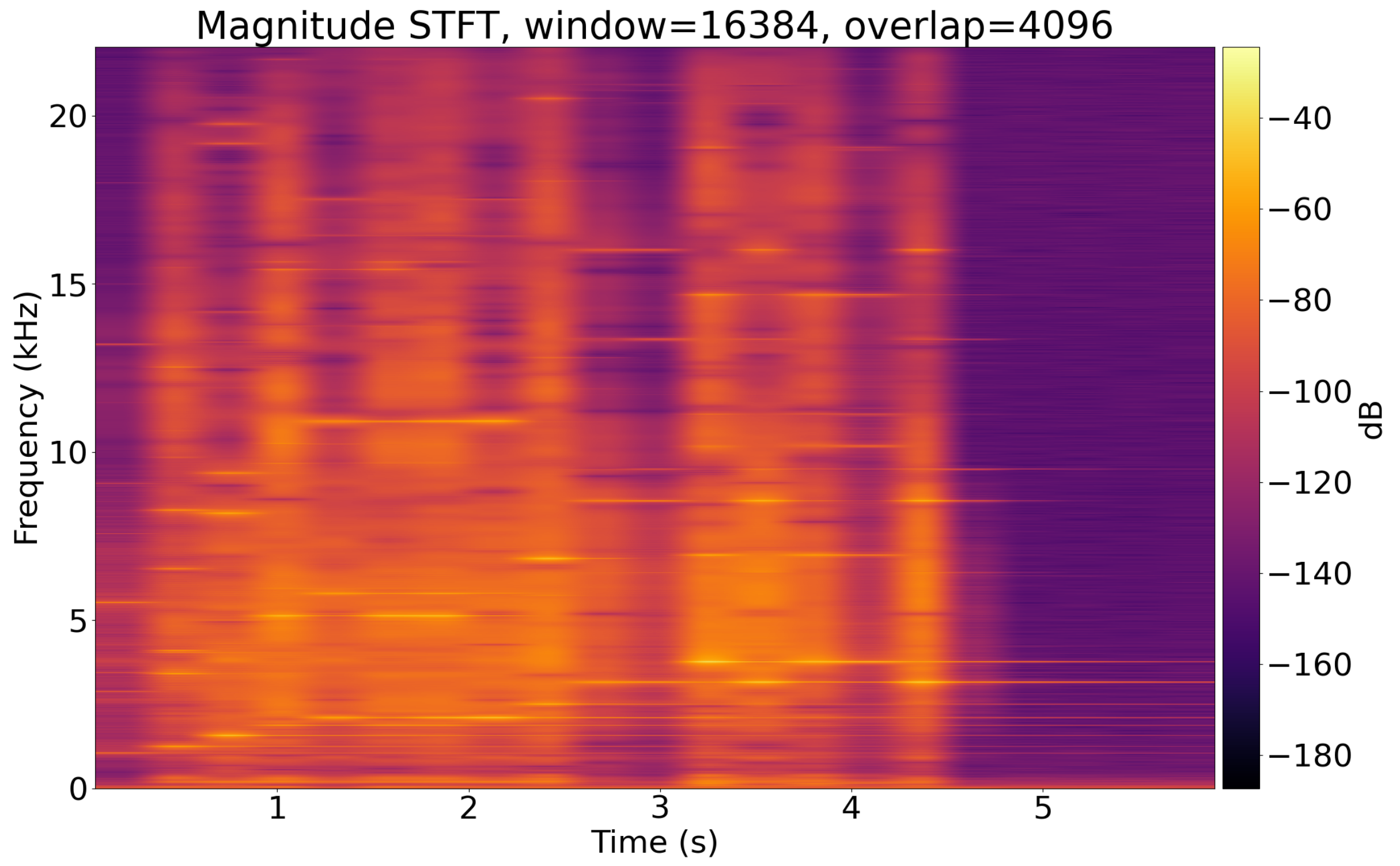
sliCQT

1. STFT: stationary Gabor transform. Fixed time-frequency resolution
2. sliCQT: realtime/slice-wise implementation of NSGT (Nonstationary Gabor transform)[1]
3. TF transforms with Fourier coefficients, varying TF resolution, perfect inverse. Musical/auditory frequency scales e.g. log2/CQT, ERBlet transform
4. sliCQT demonstrates good tonal/transient representation, and displays more musical information than the STFT


Magnitude sliCQT, Bark scale, 262 bins, 32.9-22050.0 Hz

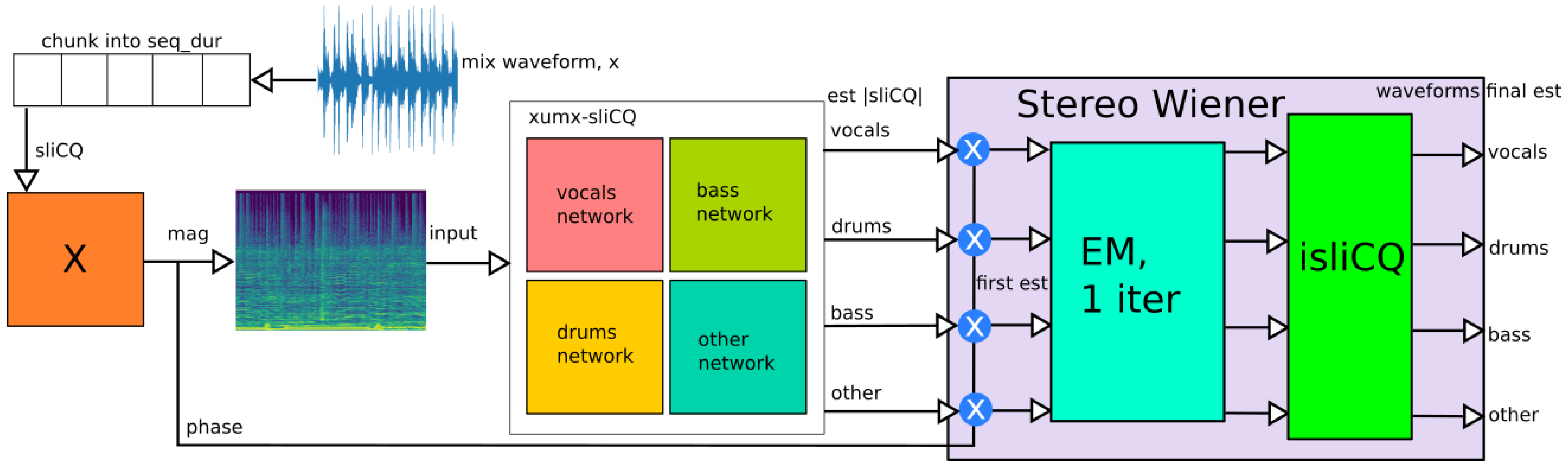

Magnitude STFT, window=1024, overlap=256

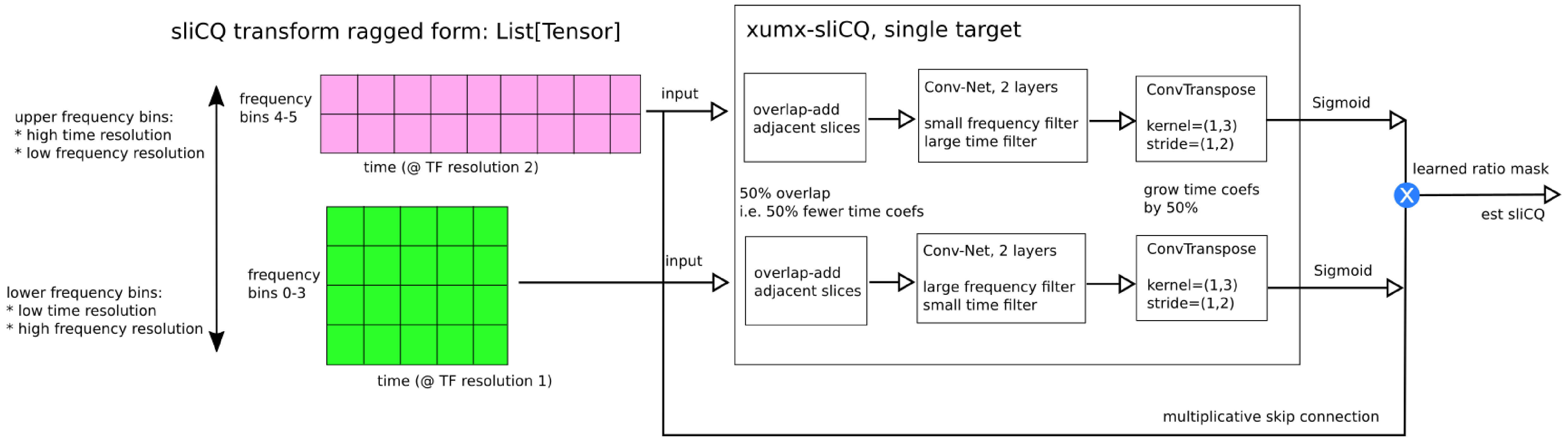

Magnitude STFT, window=4096, overlap=1024


Magnitude STFT, window=16384, overlap=4096

xumx-sliCQ

1. Simple models use magnitude spectrogram; phase and waveforms are more complicated. For waveform, use phase of mix (aka "noisy phase")
2. Choose sliCQT params by maximizing SDR of "noisy phase" oracle: $\hat{\boldsymbol{X}}_{\text{target}} = |\boldsymbol{X}_{\text{target}}| \cdot \measuredangle \boldsymbol{X}_{\text{mix}}$; **7.42 dB** vs. 6.23 on MUSDB18-HQ validation set
3. sliCQT output: list of complex 2D Time × Frequency tensors of Fourier coefficients, bucketed by time resolution. Different temporal frame rate per bucket
4. Overall system mostly similar to UMX/XUMX[2]: convolutional layers[3] applied to each bucket of sliCQT





Results and future work

1. PyTorch implementation of sliCQT: `https://github.com/sevagh/nsgt`
2. xumx-sliCQ: `https://github.com/sevagh/xumx-sliCQ`; **3.6 dB** vs. 4.64 (umx), 5.54 (x-umx) on MUSDB18-HQ test set (trained only on MUSDB18-HQ)
3. **Future:** Better sliCQT[4] + other ideas: `https://gitlab.com/sevagh/xumx_slicq_extra/-/tree/main/sliceq22-ideas`

[1] Holighaus et al., 2013, "A framework for invertible, real-time constant-Q transforms"; Balazs et al., 2011, "Theory, implementation and applications of nonstationary Gabor frames".

[2] Stöter et al., 2019, "Open-Unmix: A reference implementation for music source separation"; Sawata et al., 2021, "All for one and one for all: improving music separation by bridging networks".

[3] Grais et al., 2021, "Multi-band multi-resolution fully convolutional neural networks for singing voice separation".

[4] Schörkhuber et al., 2014, "A Matlab toolbox for efficient perfect reconstruction time-frequency transforms with log-frequency resolution".