

Lecture 20: Griffin-Lim and HiFi-GAN Vocoders

Mark Hasegawa-Johnson

These slides are in the public domain.

- 1 Definition of a vocoder: Generate waveforms from latent representations
- 2 Griffin-Lim Algorithm
- 3 Griffin-Lim Algorithm: Signal example
- 4 HiFi-GAN Vocoder
- 5 Conclusions

Outline

- 1 Definition of a vocoder: Generate waveforms from latent representations
- 2 Griffin-Lim Algorithm
- 3 Griffin-Lim Algorithm: Signal example
- 4 HiFi-GAN Vocoder
- 5 Conclusions

What is a vocoder, and why?

Most (not all) TTS operates in three stages:

- 1 **Text normalization:** Identify the meanings of acronyms and digit strings, and decide how punctuation should affect the pronunciation. **Why is it separate?** Rule-based text normalization solves most problems, and is very efficient.
- 2 **Synthesis:** Convert text into a latent representation, e.g., spectrogram, mel-spectrogram, or learned neural embedding. **Why is it separate?** This is the critical step, but it must be trained using paired (speech,text) data, of which kilohours exist.
- 3 **Vocoder:** Convert the latent representation (the “voice code”) into waveforms. **Why is it separate?** This part is trained from paired (speech,spectrogram) data — but the spectrogram can be computed from the speech! Thus megahours exist.

STFT and ISTFT

Let D be the window hop length, then the STFT can be written as

- **STFT:**

$$X_t[k] = \sum_n x[n] w[n - tD] e^{-j\omega_k(n-tD)}$$

- **ISTFT using overlap-add:**

$$x[n] = \frac{\sum_t \frac{1}{N} \sum_{k=0}^{N-1} X_t[k] e^{j\omega_k(n-tD)}}{\sum_t w[n - tD]}$$

... where, usually, we choose the window and the hop length so that $\sum_t w[n - tD]$ is a constant.

◀ ◻ ▶ ◀ ◻ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡ ≡ ↺ 🔍 ↻

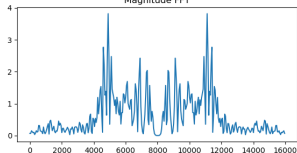
Inverting a spectrogram

Inverting a spectrogram has two problems:

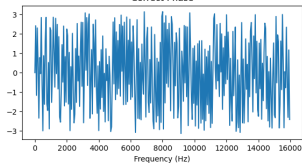
- 1 We don't know the phase of $X_t[k]$
- 2 If $X_t[k]$ was computed from a signal, then it can be inverted. But if it was computed in some other way (e.g., generated by a neural network), then it might not have a valid inverse.

Example: We don't know the phase

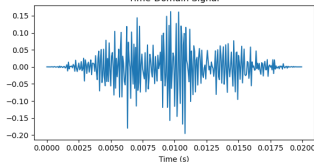
Magnitude FFT



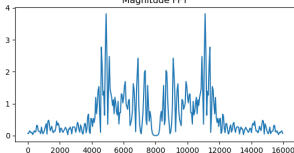
Correct Phase



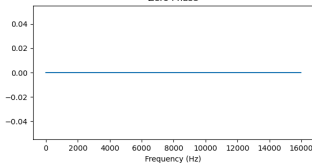
Time-Domain Signal



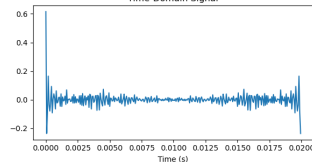
Magnitude FFT



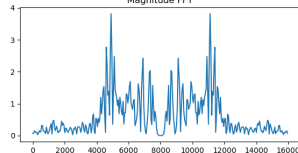
Zero Phase



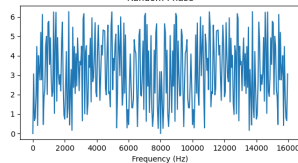
Time-Domain Signal



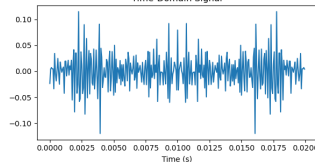
Magnitude FFT



Random Phase

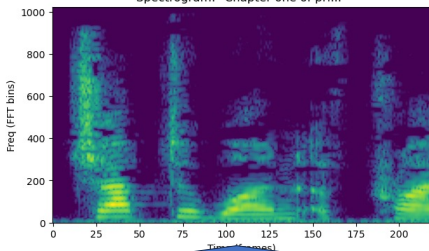


Time-Domain Signal



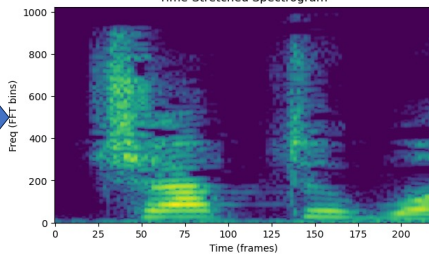
Example: There is no valid inverse

Spectrogram: "Chapter one of pri..."



Time-stretch

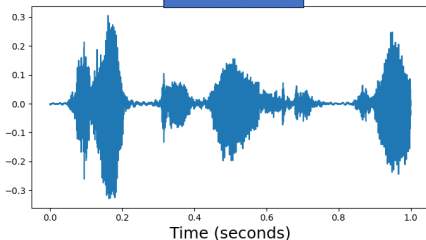
Time-Stretched Spectrogram



?

 $20\log_{10}|\text{STFT}|$

ISTFT



Outline

- 1 Definition of a vocoder: Generate waveforms from latent representations
- 2 Griffin-Lim Algorithm**
- 3 Griffin-Lim Algorithm: Signal example
- 4 HiFi-GAN Vocoder
- 5 Conclusions

Overlap between frames: A linear constraint

- Normally, we invert STFT using overlap-add.
- In order to be a valid STFT, however, you should get the same value of $x[n]$ no matter which window you use to calculate it.
- That means that, if each pair of windows overlap by $L - D$ samples (L is frame length, D is hop length), then those $L - D$ samples need to have exactly the same value, no matter which window you use to calculate them:

$$\frac{\sum_{k=0}^{N-1} X_0[k] e^{j\omega_k n}}{Nw[n]} = x[n] = \frac{\sum_{k=0}^{N-1} X_1[k] e^{j\omega_k (n-D)}}{Nw[n-D]}$$

Overlap between frames: A linear constraint

$X_t[k] = X_{t,r}[k] + jX_{t,i}[k]$ is a valid STFT only if it meets these $L - D$ **linear constraints** for the already-known samples of $x[n]$, the ones where $0 \leq n - tD < L - D$:

$$\sum_{k=0}^{N-1} X_{t,r}[k] \left(\frac{\cos(\omega_k(n - tD))}{Nw[n - tD]} \right) - \sum_{k=0}^{N-1} X_{t,i}[k] \left(\frac{\sin(\omega_k(n - tD))}{Nw[n - tD]} \right) = x[n]$$

Known magnitude: A magnitude constraint

On the other hand, suppose we know the **magnitude** of the STFT we're trying to construct, $M_t[k]$. This is a **nonlinear constraint**:

$$M_t[k] = \sqrt{X_{t,r}^2[k] + X_{t,i}^2[k]}$$

Does a particular magnitude STFT have a valid inverse?

- We want the STFT to have a desired magnitude. This is a **nonlinear constraint**:

$$M_t[k] = \sqrt{X_{t,r}^2[k] + X_{t,i}^2[k]}$$

- In order to be a valid STFT, it must be conjugate symmetric, and overlapping frames need to generate the same samples in the overlap regions. These are **linear constraints**.
- There is no guarantee that there is a valid $X_t[k]$ that satisfies $M[k] = |X_t[k]|$! But to explain the Griffin-Lim algorithm, let's consider a simple example where it is possible to meet both sets of constraints simultaneously, and let's think about how to find the valid STFT.

◀ ◻ ▶ ◀ ◻ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡ ▶ ↺ 🔍 ↻

The Griffin-Lim Algorithm

The Griffin-Lim algorithm tries to find a valid STFT using the following approach:

- 1 Initialize with random phase, $\phi_t[k] \sim U(0, 2\pi)$:

$$X_t[k] = M_t[k]e^{j\phi_t[k]}$$

- 2 Repeat the following two steps, over and over, until there is little change from one iteration to the next:
 - Find an $X_t[k]$ that satisfies the linear constraints:

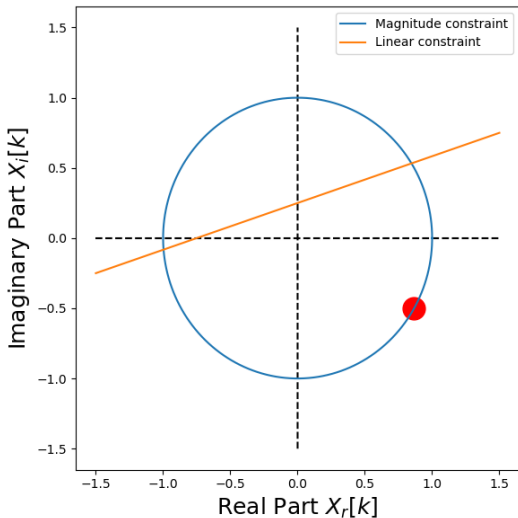
$$X_t[k] \leftarrow \text{STFT} \{ \text{ISTFT} \{ X_t[k] \} \}$$

- Rescale so that it meets the magnitude constraint:

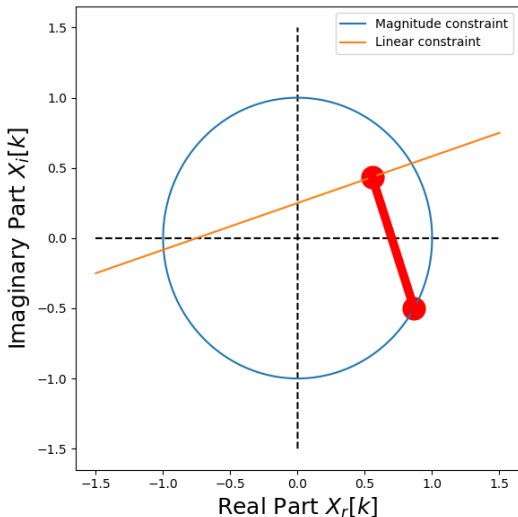
$$X_t[k] \leftarrow M_t[k]e^{j\angle X_t[k]}$$

- 3 Terminate: $x[n] = \text{ISTFT} \{ X_t[k] \}$

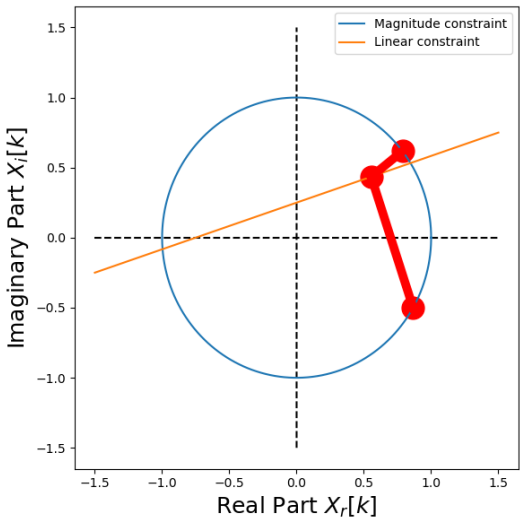
Griffin-Lim initialization: Random phase



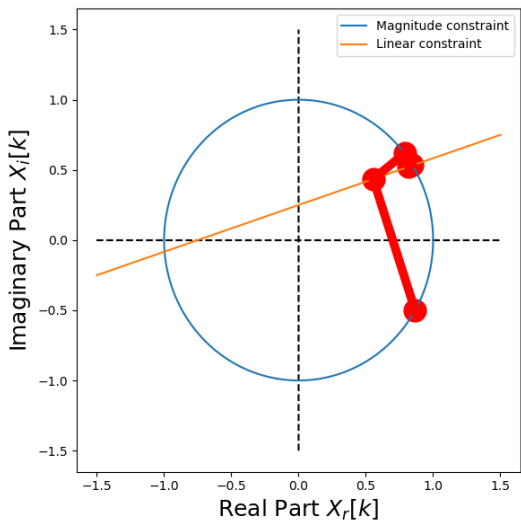
Orthogonal projection ($X_t[k] \leftarrow \text{STFT} \{ \text{ISTFT} \{ X_t[k] \} \}$)



Adjusting the magnitude ($X_t[k] \leftarrow M_t[k]e^{j\angle X_t[k]}$)



Iterate until the change from one iteration to the next becomes small



The Griffin-Lim Algorithm

- 1 Initialize with random phase, $\phi_t[k] \sim U(0, 2\pi)$:

$$X_t[k] = M_t[k]e^{j\phi_t[k]}$$

- 2 Repeat the following two steps, over and over, until there is little change from one iteration to the next:
 - Find an $X_t[k]$ that satisfies the linear constraints:

$$X_t[k] \leftarrow \text{STFT} \{ \text{ISTFT} \{ X_t[k] \} \}$$

- Rescale so that it meets the magnitude constraint:

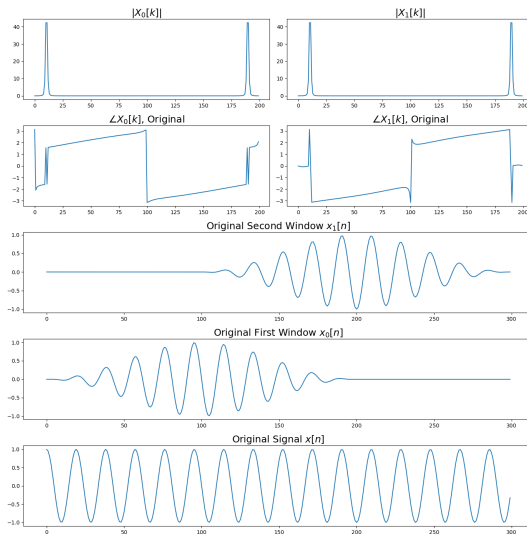
$$X_t[k] \leftarrow M_t[k]e^{j\angle X_t[k]}$$

- 3 Terminate: $x[n] = \text{ISTFT} \{ X_t[k] \}$

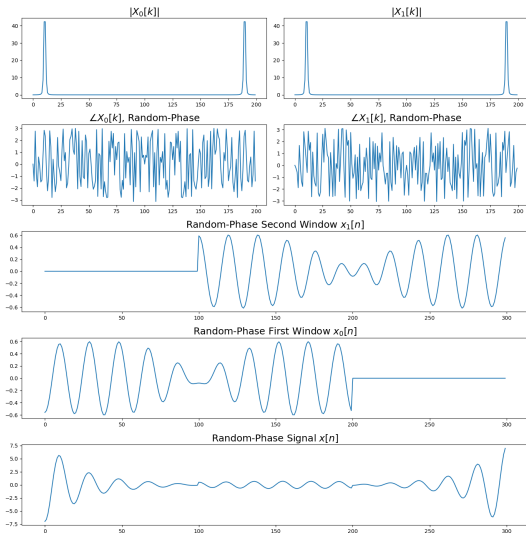
Outline

- 1 Definition of a vocoder: Generate waveforms from latent representations
- 2 Griffin-Lim Algorithm
- 3 Griffin-Lim Algorithm: Signal example
- 4 HiFi-GAN Vocoder
- 5 Conclusions

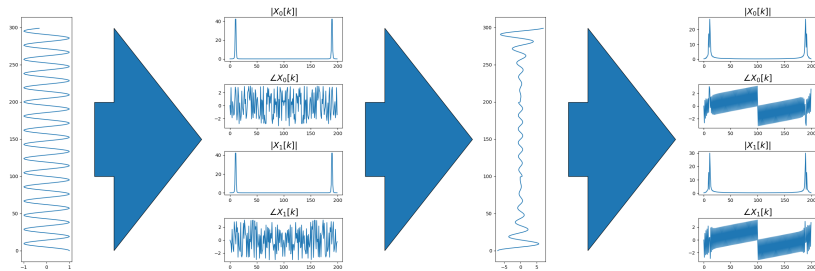
STFT of a cosine: A valid STFT



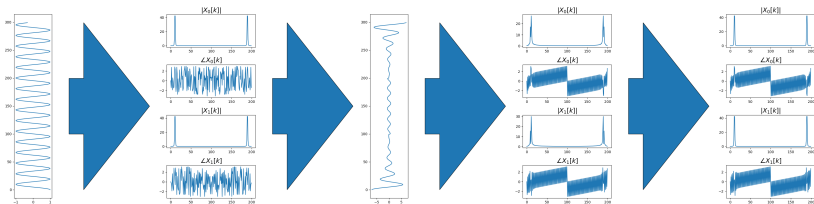
Step 1: Randomize the phase, Step 2a: Overlap-add resynthesis



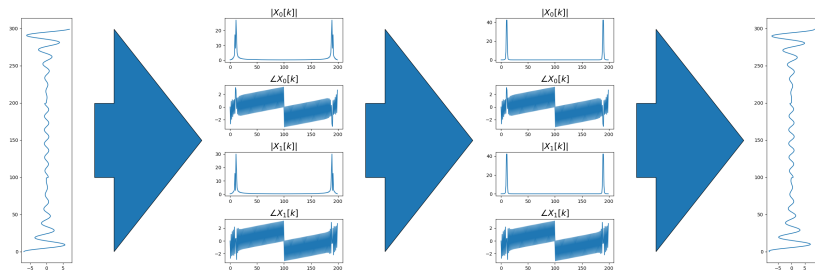
Step 2b: Take the STFT again



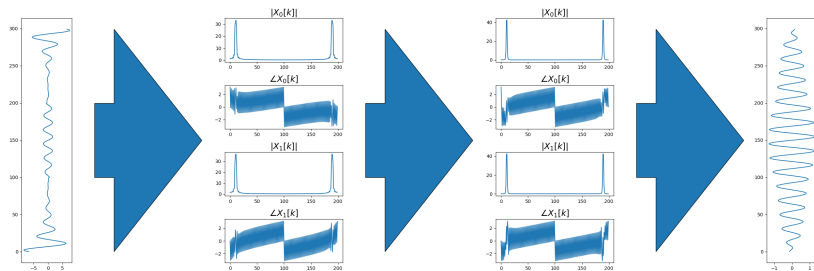
Step 2c: Fix the magnitude



Step 3a: Overlap-add resynthesis again



... and iterate again and again, until convergence

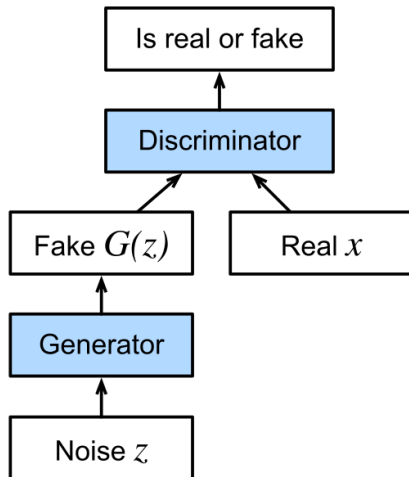


Outline

- 1 Definition of a vocoder: Generate waveforms from latent representations
- 2 Griffin-Lim Algorithm
- 3 Griffin-Lim Algorithm: Signal example
- 4 HiFi-GAN Vocoder**
- 5 Conclusions

A **generative adversarial network** is two neural nets, trained simultaneously.

- The **discriminator** is trained to distinguish real from fake signals
- The **generator** is trained to fool the discriminator
- Training has an unstable Nash equilibrium at which the generator exactly matches the distribution of real signals, and the discriminator guesses at random



Generative adversarial network

- There are many different discriminators, $1 \leq k \leq K$, each of which computes:

$$D_k(\mathbf{x}) = \begin{cases} 1 & \text{Signal is real} \\ 0 & \text{Signal is fake} \end{cases}$$

The parameters of D_k are trained to minimize

$$\mathcal{L}_{Adv}(D_k; G) = \mathbb{E} \left[(D_k(\mathbf{x}) - 1)^2 + (D_k(G(\mathbf{s})))^2 \right]$$

- The generator converts latent $\mathbf{s}$ into waveform $\mathbf{x}$:

$$\mathbf{x} = G(\mathbf{s})$$

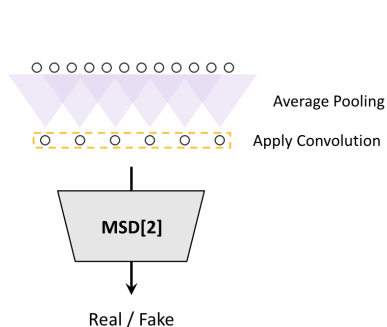
The parameters of G are trained to minimize

$$\sum_{k=1}^K \mathcal{L}_{Adv}(G; D_{1:K}) = \mathbb{E} \left[\sum_{k=1}^K (D_k(G(\mathbf{s})) - 1)^2 \right]$$

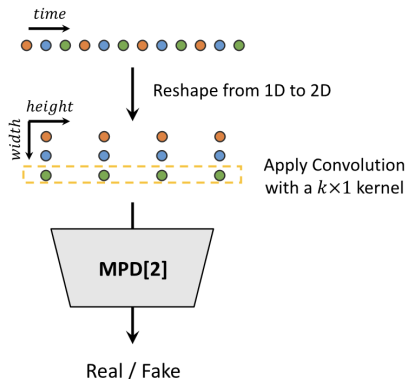
The HiFi-GAN idea: Many views of the speech signal

- Multi-scale discriminators (MSD): Waveform samples are average-pooled at different scales. From each, a ConvNet discriminates real vs. fake
- Multi-period discriminators (MPD): Waveform samples are downsampled at different downsampling periods. From each, a ConvNet discriminates real vs. fake
- Multi-window (not in HiFi-GAN): Spectrograms computed with different windows are evaluated real vs. fake
- Multi-transform (not in HiFi-GAN): Constant-Q transform, wavelet transform, etc are all computed from the waveform and discriminated

The HiFi-GAN idea: Many views of the speech signal

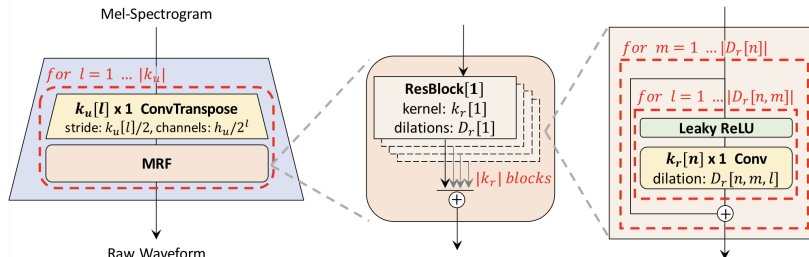


Three MSDs average pool 1, 2, or 4 samples, respectively (shown: 4)



Three MPDs downsample by 2, 3, 5, 7, or 11 samples, respectively (shown: 3)

The HiFi-GAN generator



- Stage 1: ConvTranspose = convolution combined with $\times 2$ upsampling
- Stage 2: Multi-receptive field (MRF) = residual signal modifications generated by many different dilated convolutions

Further improving the generator: Feature matching

- For every natural waveform $\mathbf{x}$, compute its spectrogram $\mathbf{s}$, then generate $\hat{\mathbf{x}} = G(\mathbf{s})$ from that. Losses include:
 - ① Adversarial loss
 - ② Spectrogram-matching: $\|\mathbf{s} - \hat{\mathbf{s}}\|_1$, $\hat{\mathbf{s}}$ is spectrogram generated from $\hat{\mathbf{x}}$.
 - ③ Feature-matching: $\|\mathbf{f}_k(\mathbf{x}) - \mathbf{f}_k(\hat{\mathbf{x}})\|_1$, $\mathbf{f}_k$ are features computed by D_k .
- Result: Naturalness MOS of 4.36 ± 0.07 , vs. 4.02 ± 0.08 for WaveNet, 4.45 ± 0.06 for real speech.

Outline

- 1 Definition of a vocoder: Generate waveforms from latent representations
- 2 Griffin-Lim Algorithm
- 3 Griffin-Lim Algorithm: Signal example
- 4 HiFi-GAN Vocoder
- 5 Conclusions

Conclusions

- 1 Griffin-Lim = Initialize with random phase, then repeat the following two steps, over and over, until there is little change from one iteration to the next:

- Find an $X_t[k]$ that satisfies the linear constraints:

$$X_t[k] \leftarrow \text{STFT} \{ \text{ISTFT} \{ X_t[k] \} \}$$

- Rescale so that it meets the magnitude constraint:

$$X_t[k] \leftarrow M_t[k] e^{j\angle X_t[k]}$$

- 2 HiFi-GAN = learn to generate signals that fool many different types of discriminator networks (multi-scale, multi-period)