جامعة بيرزيت

BIRZEIT UNIVERSITY

Electrical and Computer Engineering

Spoken Language Processing - Fall 2025/2026

Submission Deadline: Saturday 29 November 2025 (Via Moodle only: itc.birzeit.edu)

Work in groups:

You can work on this assignment in groups of **two** or **three** (maximum three!) students. It should be clear each partner works on which part, and it should be added into the submitted report

Assignment overview (3 parts)

Part A — Data collection & acoustic analysis (Vowels)

Goal: Measure F1, F2, duration, and spectral properties for several vowels.

Procedure:

1. Materials: Pick 5 steady monophthong vowels (e.g., /i, e, a, o, u/). Use carrier phrase or /hVd/ context.

2. Recording: Record 1–5 speakers, 10 tokens per vowel, 16 kHz 16-bit WAV in quiet environment.

3. Acoustic analysis: Use Praat or parselmouth to extract formants at mid-vowel, measure duration, and plot spectrograms.

Deliverables (Part A): Table of measurements, vowel space plot (F1 vs F2), spectrograms & waveform figures, short discussion.

Part B: Measuring and Investigating Pitch Frequency (F_0) from Speech Recordings in Part A

To measure the pitch frequency, use the both methods below:

Method 1 — Using Praat

Open your recording in Praat.

Use Analyze periodicity → To Pitch... (set time step = 0.01s, range = 75–500 Hz).

View and save the pitch track.

Export results: Pitch → Down to Table → Write to text file.

Method 2 — Using Python (parselmouth or librosa)

```
Example using parselmouth:
import parselmouth
snd = parselmouth.Sound("sample.wav")
pitch = snd.to_pitch()
pitch_values = pitch.selected_array['frequency']
times = pitch.xs()
Plotting example:
import matplotlib.pyplot as plt
plt.plot(times, pitch_values)
plt.title("Pitch contour")
plt.xlabel("Time (s)")
plt.ylabel("Frequency (Hz)")
plt.show()
```

Pitch Frequency Analysis & Discussion

Include in your report:

Pitch range (minimum, maximum, and mean F_0).

Pitch contour plot (time vs. frequency).

Interpretation:

How does pitch change during the sentence?

Differences between sustained vowel vs. connected speech.

How does your F_0 compare to typical male/female ranges (male ≈ 85–180 Hz; female ≈ 165–255 Hz)?

Compare the results obtained by Praat method with the results obtained by Python method.

Part C — Source–Filter synthesis and manipulation

1 — Setup & baseline

- Create a project repo / notebook with dependencies: `numpy scipy matplotlib soundfile librosa parselmouth`.
- Define sampling rate `fs = 16000` and duration `1.0 s`.

2 — Implement the glottal source

Two levels of complexity:

A. **Impulse train (simple)**

- Periodic impulses at fundamental frequency `F0` (e.g., 100–140 Hz for male, 200 Hz for female).
- Optionally lowpass filter the impulse train to simulate glottal pulse shape.

B. **LF (Liljencrants–Fant) glottal model (optional)**

- Implement or use existing code for LF waveform for more natural source.

Suggested parameters: `F0 = 120 Hz`, small jitter optional

3 — Implement vocal-tract resonators (formant filters)

- Use 2nd-order IIR resonators (biquad peak filters) for each formant:
    - center frequency = Fi (Hz)
    - bandwidth = Bi (Hz)
- Cascade them (filter source through F1, then F2, then F3).
- Implement helper: `iirpeak(center_freq/(fs/2), Q)` or design biquad coefficients using standard formula.

Typical vowel formant set (male reference, Hz):

- /i/: F1=300, F2=2400, F3=3000
- /e/: F1=500, F2=1900, F3=2500
- /a/: F1=800, F2=1200, F3=2600
- /o/: F1=500, F2=900, F3=2400
- /u/: F1=350, F2=700, F3=2400

Bandwidths: F1 BW≈70 Hz, F2 BW≈120 Hz, F3 BW≈200 Hz (adjustable).

4 — Synthesis pipeline

- Generate source `s[n]`.
- Cascade through resonators → `y[n]`.
- Normalize amplitude; save `.wav` using `soundfile.write`.
- Plot waveform, spectrogram (`plt.specgram` or librosa.display.specshow), and compute formants (via parselmouth) to verify.

5 — Compare with natural vowels

- Use a small dataset of recorded vowels (student recordings or example WAVs).
- Extract formants using `parselmouth` or Praat for both natural and synth.
- Plot vowel space (F1 vs F2) for natural vs. synthetic.

.

Part 6 — Optional perceptual test

- Prepare 5–10 synthesized tokens and corresponding natural tokens.
- Get 10–20 naive listeners to do forced-choice vowel identification or rate naturalness (1–5).
- Report confusion matrix or mean ratings.

Suggested Evaluation Metrics

- **Formant error:** |F1_synth − F1_natural|, |F2_synth − F2_natural| (Hz) averaged over vowels.
- **Visual:** Spectrogram and waveform comparisons.
- **Psychoacoustic:** Listener identification accuracy or MOS (mean opinion score).

Useful Code Snippets (Python)

1) Simple impulse train source

```
import numpy as np

fs = 16000
duration = 1.0
F0 = 120
t = np.linspace(0, duration, int(fs*duration), endpoint=False)
source = np.zeros_like(t)
period = int(fs / F0)
source[::period] = 1.0
# gentle lowpass to shape glottal (optional)
from scipy.signal import butter, filtfilt
b,a = butter(2, 400/(fs/2))
source = filtfilt(b,a, source)
```

2) Resonator (IIR peak) helper + cascade

```
from scipy.signal import iirpeak, lfilter

def resonator(center_freq, bandwidth, fs):
    Q = center_freq / bandwidth
    b,a = iirpeak(center_freq/(fs/2), Q)
    return b, a

def apply_resonators(x, formants, fs):
    y = x.copy()
    for (f, bw) in formants:
        b,a = resonator(f, bw, fs)
        y = lfilter(b, a, y)
    return y

formants_a = [(800,70),(1200,120),(2600,200)]
y = apply_resonators(source, formants_a, fs)
```

3) Spectrogram + saving

```
import matplotlib.pyplot as plt
import soundfile as sf

# normalize
y = y / np.max(np.abs(y)) * 0.9
sf.write('synth_a.wav', y, fs)

plt.specgram(y, NFFT=1024, Fs=fs, noverlap=512)
plt.title('Synth /a/ spectrogram')
plt.show()
```

4) Estimate formants using parselmouth (Praat bindings)

```
import parselmouth
snd = parselmouth.Sound('synth_a.wav')
formants = snd.to_formant_burg()
mid = snd.get_total_duration() / 2
f1 = formants.get_value_at_time(1, mid)
f2 = formants.get_value_at_time(2, mid)
print('Synth formants:', f1, f2)
```

Suggested Extensions (if time permits)

- Replace impulse train with **LF glottal model** for realism.
- Implement **formant bandwidth tuning** to match natural spectra better.
- Add **aspiration/noise** component for unvoiced consonants or breathiness.
- Build a small GUI to change F1/F2 interactively and hear results.
- Implement time-varying formants to synthesize diphthongs or simple CV syllables.

Report structure & submission

Submit a 2–4 page report containing: introduction, methods, results (Parts A–C), discussion, conclusion, references. Include code and data as a folder.

Optional: short demo video (3–5 minutes) with audio examples.

Tools & starter code (notebook provided)

The accompanying Jupyter notebook provides minimal, ready-to-run Python examples for formant extraction (parselmouth), synthesis (scipy), and plotting.

Academic integrity & ethics

When collecting participant data, obtain consent. Do not include identifying information in submission.

Good luck — Dr. Abualsoud Hanani