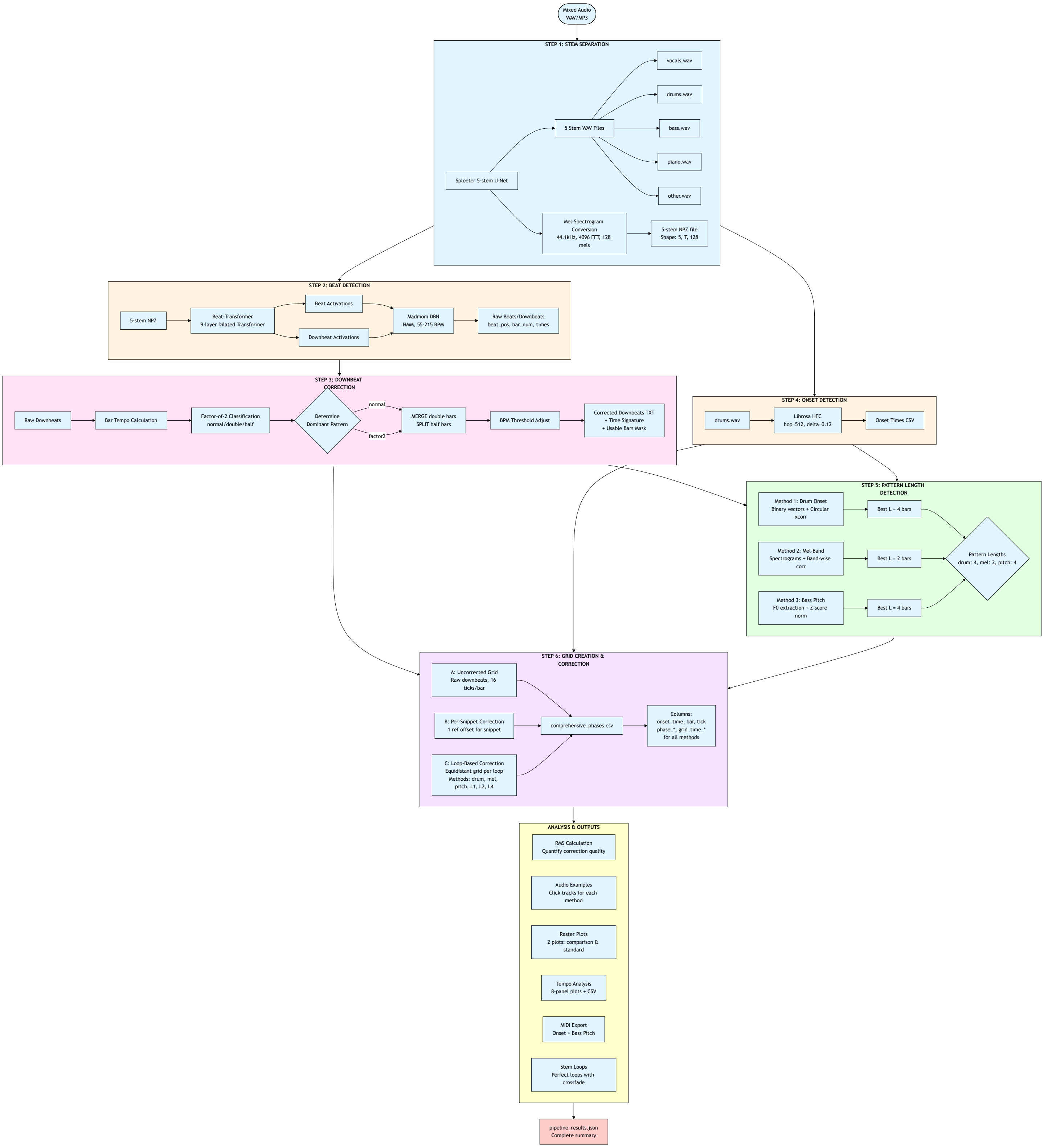
Loop Extractor Pipeline - Mermaid Diagram

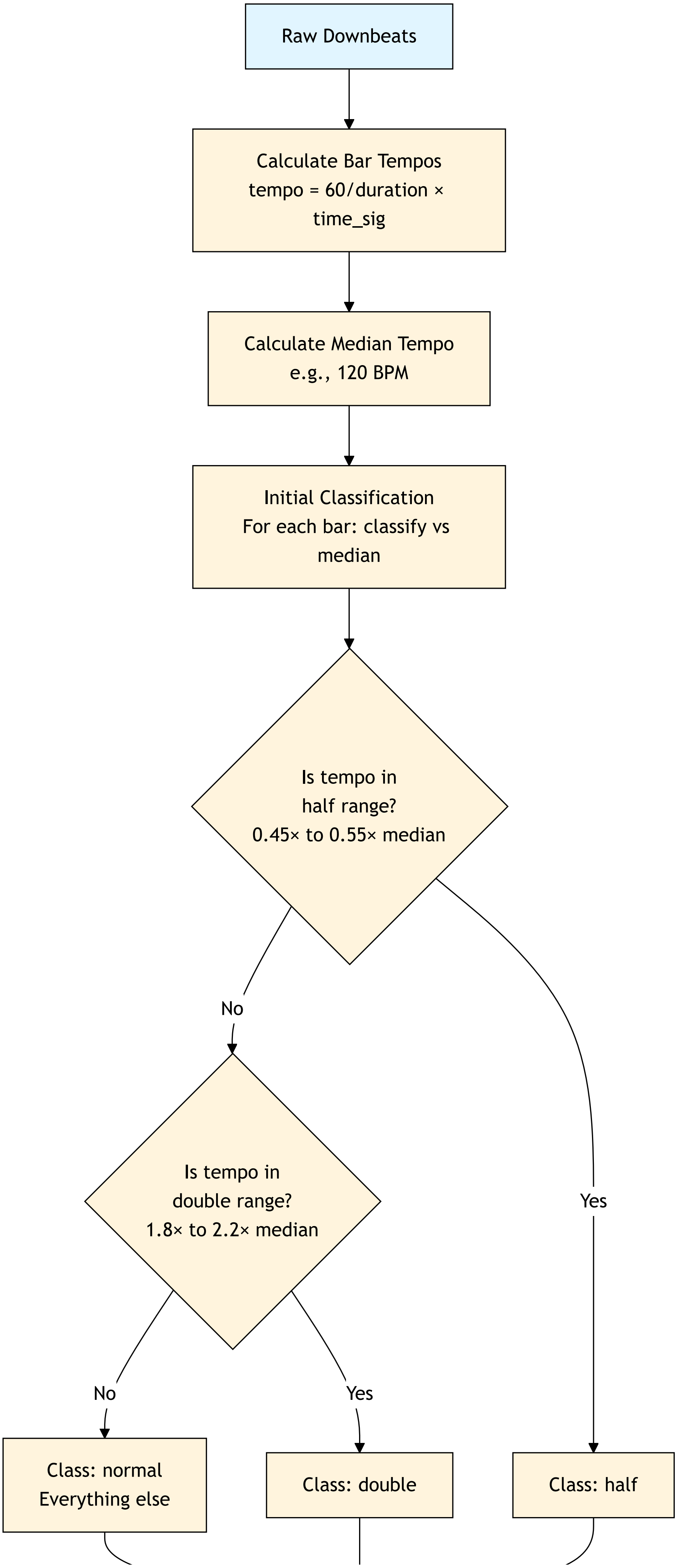
Author: Alexander Krause, TU Berlin

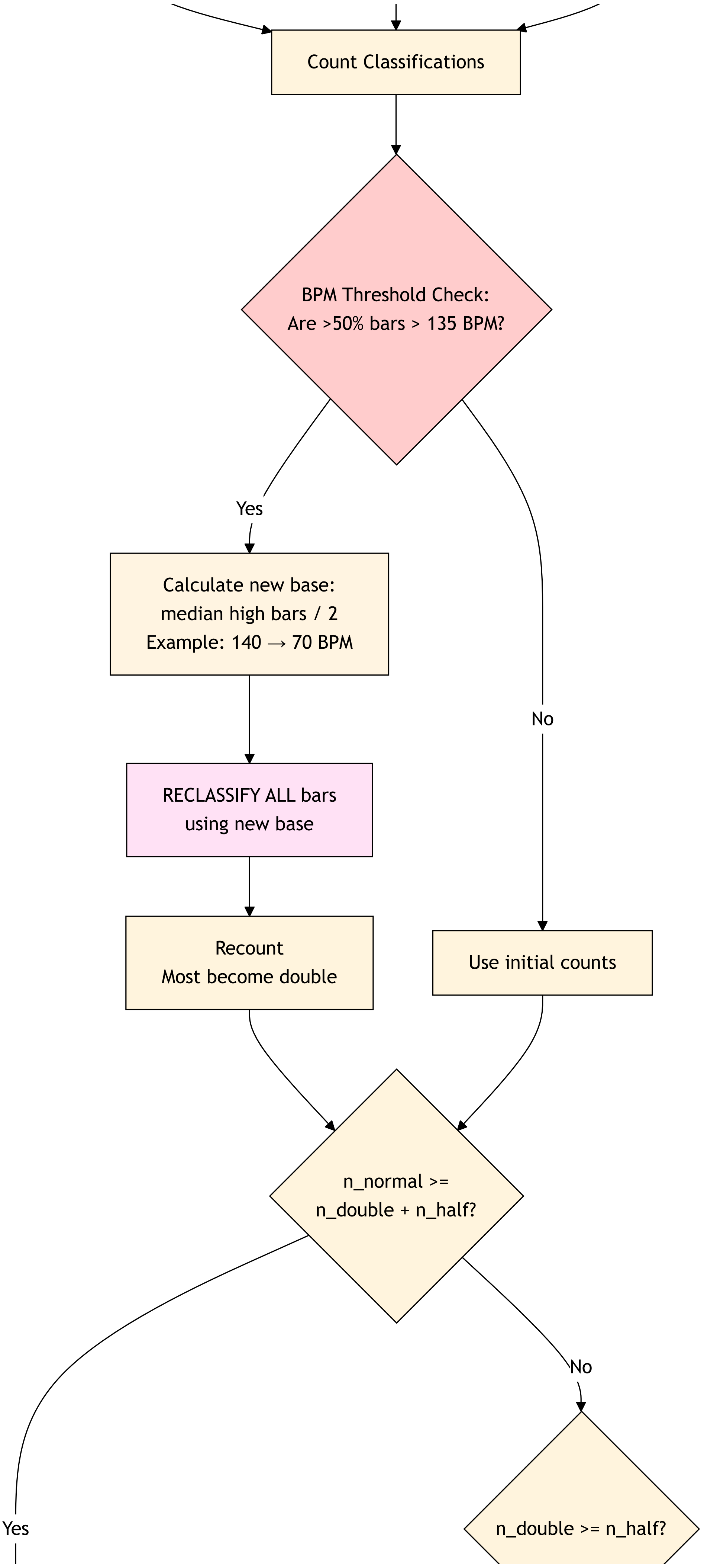
Co-Author: Claude Code (Anthropic)

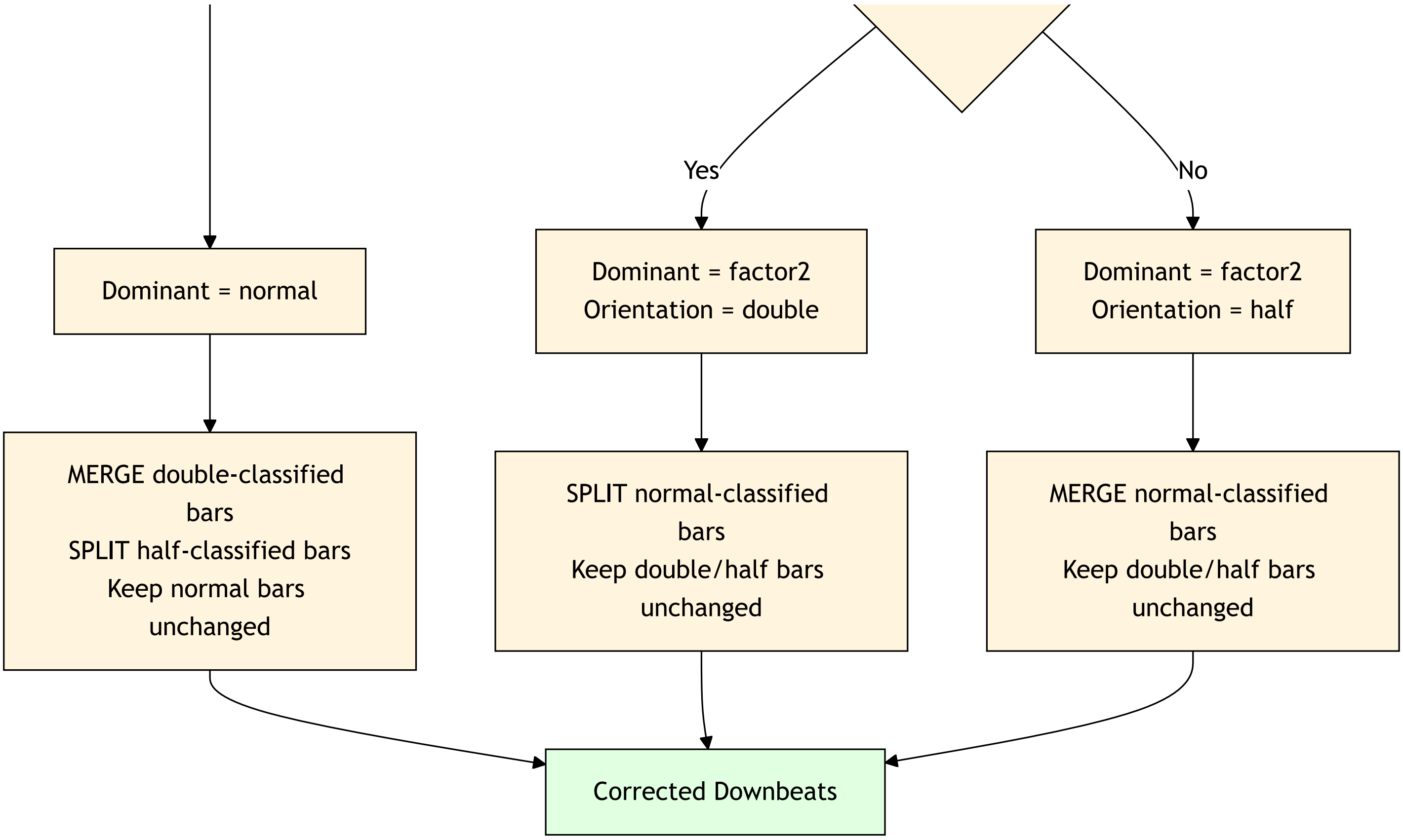
Main Pipeline Flow


Mixed Audio
WAV/MP3
STEP 1: STEM SEPARATION
Spleeter 5-stem U-Net
5 Stem WAV Files
vocals.wav
drums.wav
bass.wav
piano.wav
other.wav
Mel-Spectrogram Conversion 44.1kHz, 4096 FFT, 128 mels
5-stem NPZ file Shape: 5, T, 128
STEP 2: BEAT DETECTION
5-stem NPZ
Beat-Transformer 9-layer Dilated Transformer
Beat Activations
Downbeat Activations
Madmom DBN HMM, 55-215 BPM
Raw Beats/Downbeats beat_pos, bar_num, times
STEP 3: DOWNBEAT CORRECTION
Raw Downbeats
Bar Tempo Calculation
Factor-of-2 Classification normal/double/half
Determine Dominant Pattern
normal
factor2
MERGE double bars SPLIT half bars
BPM Threshold Adjust
Corrected Downbeats TXT + Time Signature + Usable Bars Mask
STEP 4: ONSET DETECTION
drums.wav
Librosa HFC hop=512, delta=0.12
Onset Times CSV
STEP 5: PATTERN LENGTH DETECTION
Method 1: Drum Onset Binary vectors + Circular xcorr
Best L = 4 bars
Method 2: Mel-Band Spectrograms + Band-wise corr
Best L = 2 bars
Method 3: Bass Pitch F0 extraction + Z-score norm
Best L = 4 bars
Pattern Lengths drum: 4, mel: 2, pitch: 4
STEP 6: GRID CREATION & CORRECTION
A: Uncorrected Grid Raw downbeats, 16 ticks/bar
B: Per-Snippet Correction 1 ref offset for snippet
C: Loop-Based Correction Equidistant grid per loop Methods: drum, mel, pitch, L1, L2, L4
comprehensive_phases.csv
Columns: onset_time, bar, tick phase_*, grid_time_* for all methods
ANALYSIS & OUTPUTS
RMS Calculation Quantify correction quality
Audio Examples Click tracks for each method
Raster Plots 2 plots: comparison & standard
Tempo Analysis 8-panel plots + CSV
MIDI Export Onset + Bass Pitch
Stem Loops Perfect loops with crossfade
pipeline_results.json Complete summary

Detailed Step 3: Downbeat Correction Logic


Raw Downbeats
Calculate Bar Tempos
tempo = 60/duration × time_sig
Calculate Median Tempo
e.g., 120 BPM
Initial Classification
For each bar: classify vs median
Is tempo in half range?
0.45× to 0.55× median
No
Yes
Is tempo in double range?
1.8× to 2.2× median
No
Yes
Class: normal
Everything else
Class: double
Class: half


Count Classifications
BPM Threshold Check:
Are >50% bars > 135 BPM?
Yes
Calculate new base:
median high bars / 2
Example: 140 → 70 BPM
RECLASSIFY ALL bars
using new base
Recount
Most become double
No
Use initial counts
n_normal >=
n_double + n_half?
Yes
No
n_double >= n_half?


Yes
No
Dominant = normal
Dominant = factor2
Orientation = double
Dominant = factor2
Orientation = half
MERGE double-classified bars
SPLIT half-classified bars
Keep normal bars unchanged
SPLIT normal-classified bars
Keep double/half bars unchanged
MERGE normal-classified bars
Keep double/half bars unchanged
Corrected Downbeats

How 3-Beat and 5-Beat Bars are Corrected

The downbeat correction automatically fixes bars that were misdetected as having 3 or 5 beats when the dominant time signature is 4/4:

Misdetected Bar	Actual Beats	Duration vs 4/4	Tempo Classification	Correction Action
3-beat bar	3 beats	75% of normal	~1.33x tempo → "double"	**MERGED** with next bar to create one 4/4 bar
5-beat bar	5 beats	125% of normal	~0.8x tempo → "half"	**SPLIT** into two bars (4+4 beats)

Example: If BeatTransformer detects a sequence like:

```
Bar 1: 4 beats ✓
Bar 2: 3 beats ✗ (too short)
Bar 3: 1 beat   (remainder from merge)
Bar 4: 4 beats ✓
```

After correction:

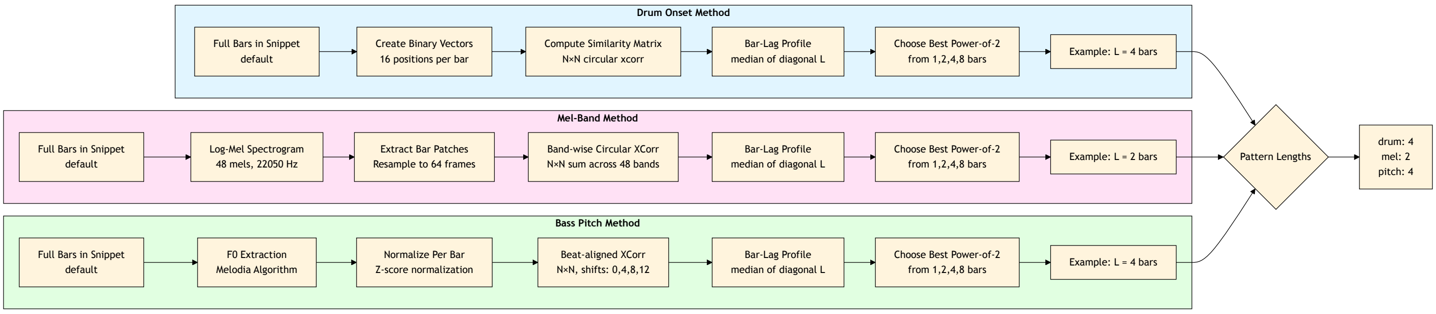
```
Bar 1: 4 beats ✓
Bar 2: 4 beats ✓ (merged Bar 2 + Bar 3)
Bar 3: 4 beats ✓
```

This tempo-based classification and correction ensures consistent 4/4 bars for downstream analysis, even when the beat detection model makes occasional errors.

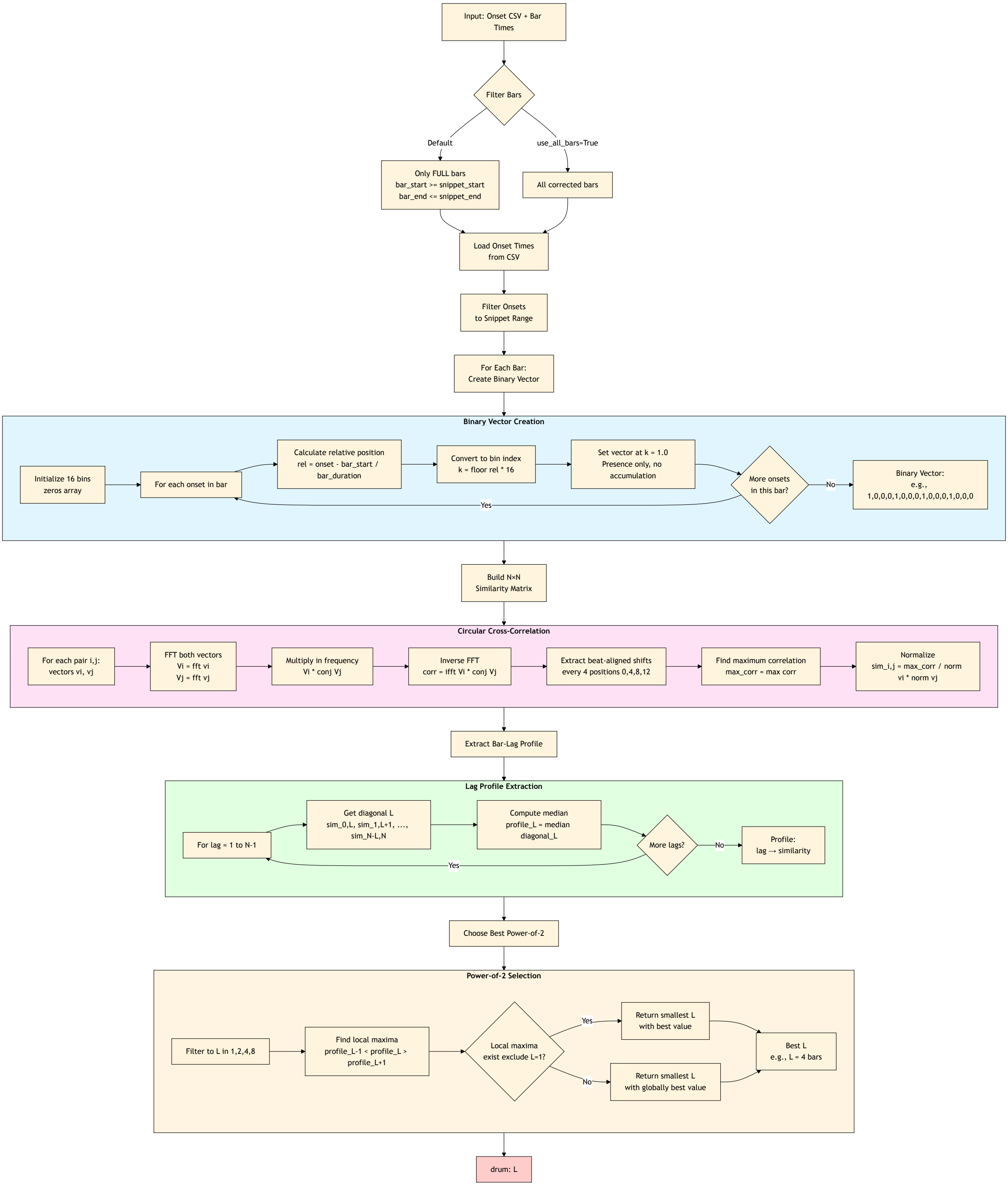
Detailed Step 5: Pattern Length Detection Methods

Important: By default, the three methods analyze only **FULL bars within the snippet boundaries** (bars where both start AND end times fall within the snippet). This ensures pattern detection focuses on the analyzed region. You can optionally set `use_all_bars=True` to analyze all corrected bars regardless of snippet boundaries.

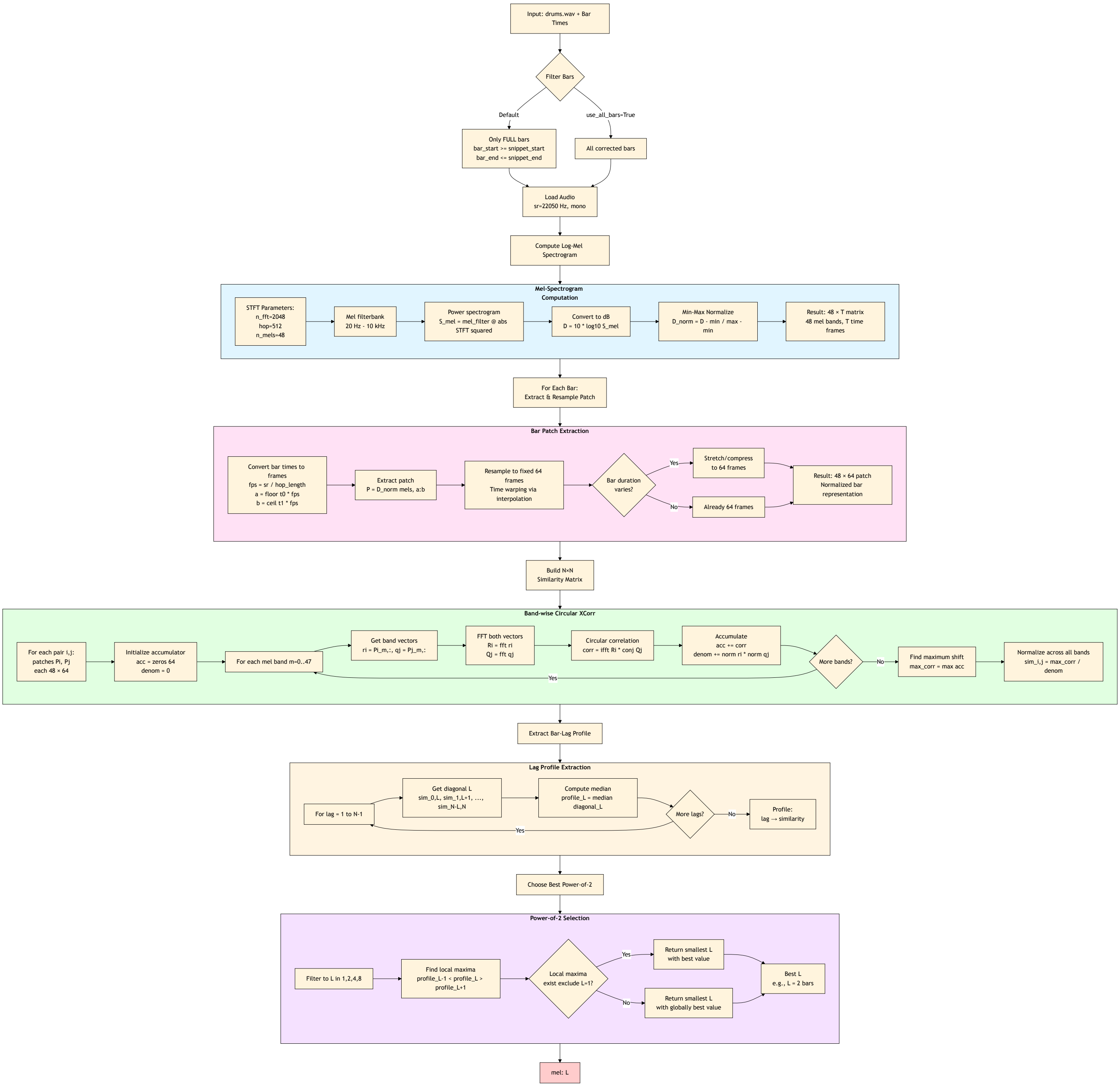
Each method creates an N×N similarity matrix comparing every bar to every other bar, then extracts the diagonal at lag L to measure periodicity. The median similarity at each lag determines the best pattern length.


Drum Onset Method
Full Bars in Snippet default
Create Binary Vectors 16 positions per bar
Compute Similarity Matrix N×N circular xcorr
Bar-Lag Profile median of diagonal L
Choose Best Power-of-2 from 1,2,4,8 bars
Example: L = 4 bars
Mel-Band Method
Full Bars in Snippet default
Log-Mel Spectrogram 48 mels, 22050 Hz
Extract Bar Patches Resample to 64 frames
Band-wise Circular XCorr N×N sum across 48 bands
Bar-Lag Profile median of diagonal L
Choose Best Power-of-2 from 1,2,4,8 bars
Example: L = 2 bars
Bass Pitch Method
Full Bars in Snippet default
F0 Extraction Melodia Algorithm
Normalize Per Bar Z-score normalization
Beat-aligned XCorr N×N, shifts: 0,4,8,12
Bar-Lag Profile median of diagonal L
Choose Best Power-of-2 from 1,2,4,8 bars
Example: L = 4 bars
Pattern Lengths
drum: 4
mel: 2
pitch: 4

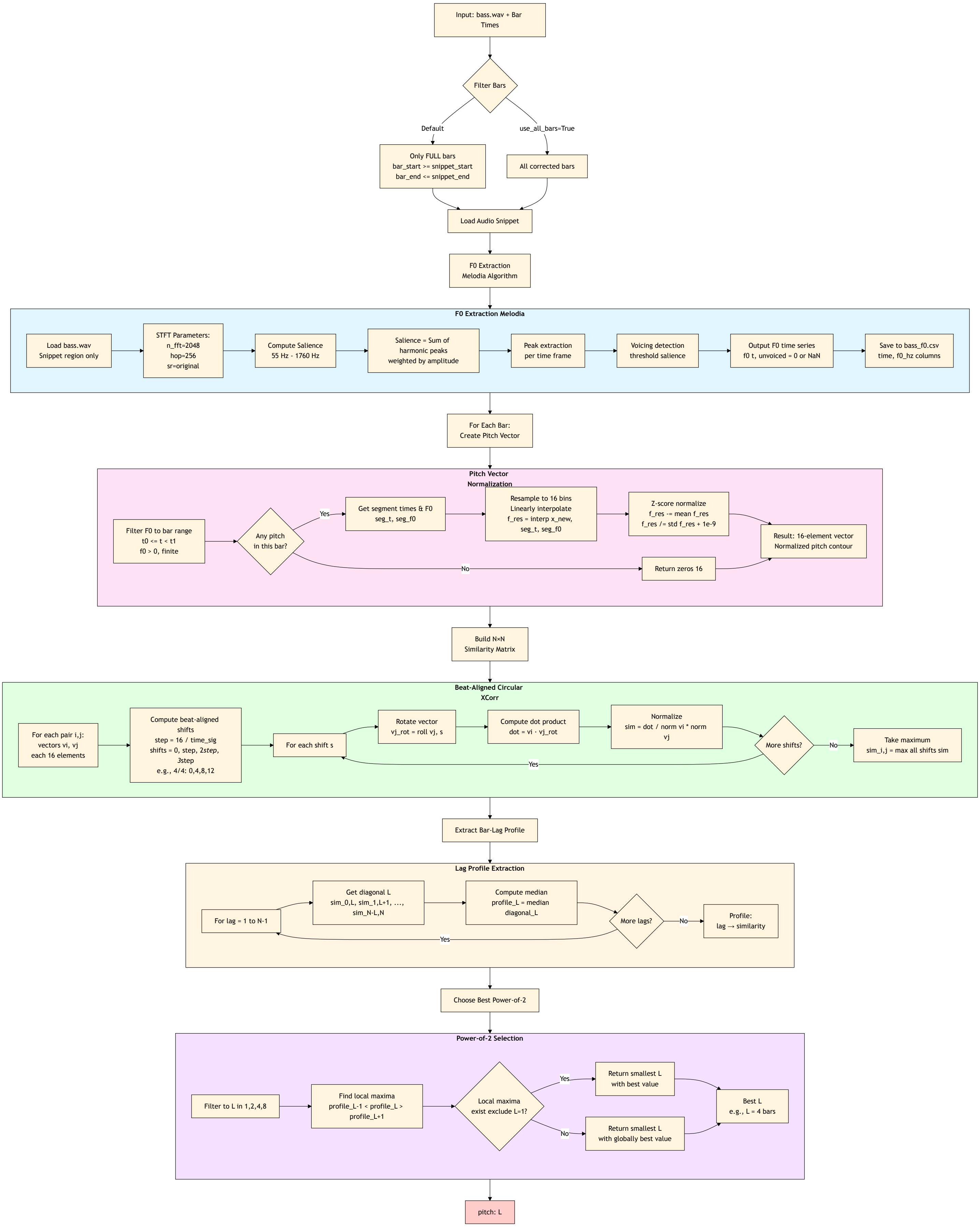
Method 1: Drum Onset Pattern Detection (Detailed)



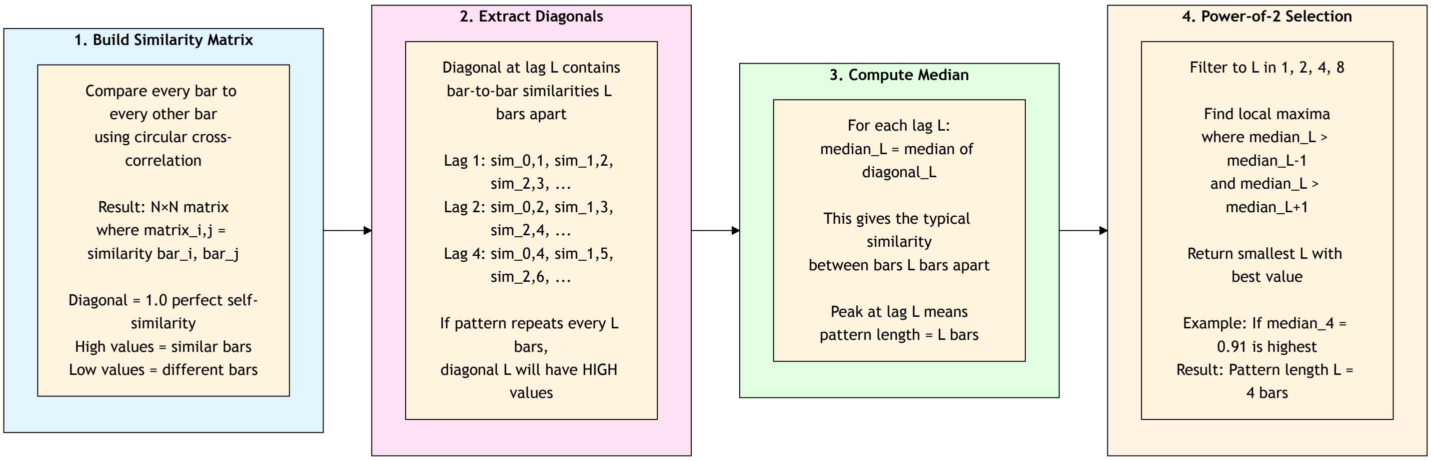
Method 2: Mel-Band Pattern Detection (Detailed)



Method 3: Bass Pitch Pattern Detection (Detailed)



Pattern Detection: Similarity Matrix Visualization



1. Build Similarity Matrix

Compare every bar to every other bar using circular cross-correlation

Result: N×N matrix where matrix_i,j = similarity bar_i, bar_j

Diagonal = 1.0 perfect self-similarity
High values = similar bars
Low values = different bars

2. Extract Diagonals

Diagonal at lag L contains bar-to-bar similarities L bars apart

Lag 1: sim_0,1, sim_1,2, sim_2,3, ...
Lag 2: sim_0,2, sim_1,3, sim_2,4, ...
Lag 4: sim_0,4, sim_1,5, sim_2,6, ...

If pattern repeats every L bars, diagonal L will have HIGH values

3. Compute Median

For each lag L: median_L = median of diagonal_L

This gives the typical similarity between bars L bars apart

Peak at lag L means pattern length = L bars

4. Power-of-2 Selection

Filter to L in 1, 2, 4, 8

Find local maxima where median_L > median_L-1 and median_L > median_L+1

Return smallest L with best value

Example: If median_4 = 0.91 is highest
Result: Pattern length L = 4 bars

Example: 8-Bar Pattern with L=4 Repetition

Similarity Matrix (8×8):

```
       Bar0  Bar1  Bar2  Bar3  Bar4  Bar5  Bar6  Bar7
Bar0 │ 1.00  0.45  0.32  0.28  0.92  0.41  0.29  0.25 │
Bar1 │ 0.45  1.00  0.43  0.31  0.44  0.89  0.40  0.28 │
Bar2 │ 0.32  0.43  1.00  0.42  0.31  0.43  0.87  0.39 │
Bar3 │ 0.28  0.31  0.42  1.00  0.27  0.30  0.41  0.85 │
Bar4 │ 0.92  0.44  0.31  0.27  1.00  0.43  0.30  0.26 │
Bar5 │ 0.41  0.89  0.43  0.30  0.43  1.00  0.42  0.29 │
Bar6 │ 0.29  0.40  0.87  0.41  0.30  0.42  1.00  0.40 │
Bar7 │ 0.25  0.28  0.39  0.85  0.26  0.29  0.40  1.00 │
```

Diagonal Extraction:

- **Lag 1:** [0.45, 0.43, 0.42, 0.27, 0.43, 0.42, 0.40] → median = **0.42**
- **Lag 2:** [0.32, 0.31, 0.42, 0.31, 0.43, 0.87] → median = **0.37**
- **Lag 3:** [0.28, 0.31, 0.42, 0.27, 0.30, 0.41] → median = **0.31**
- **Lag 4:** [**0.92**, 0.89, **0.87**, **0.85**] → median = **0.88** ⭐ **PEAK!**
- **Lag 5:** [0.41, 0.43, 0.41] → median = **0.41**

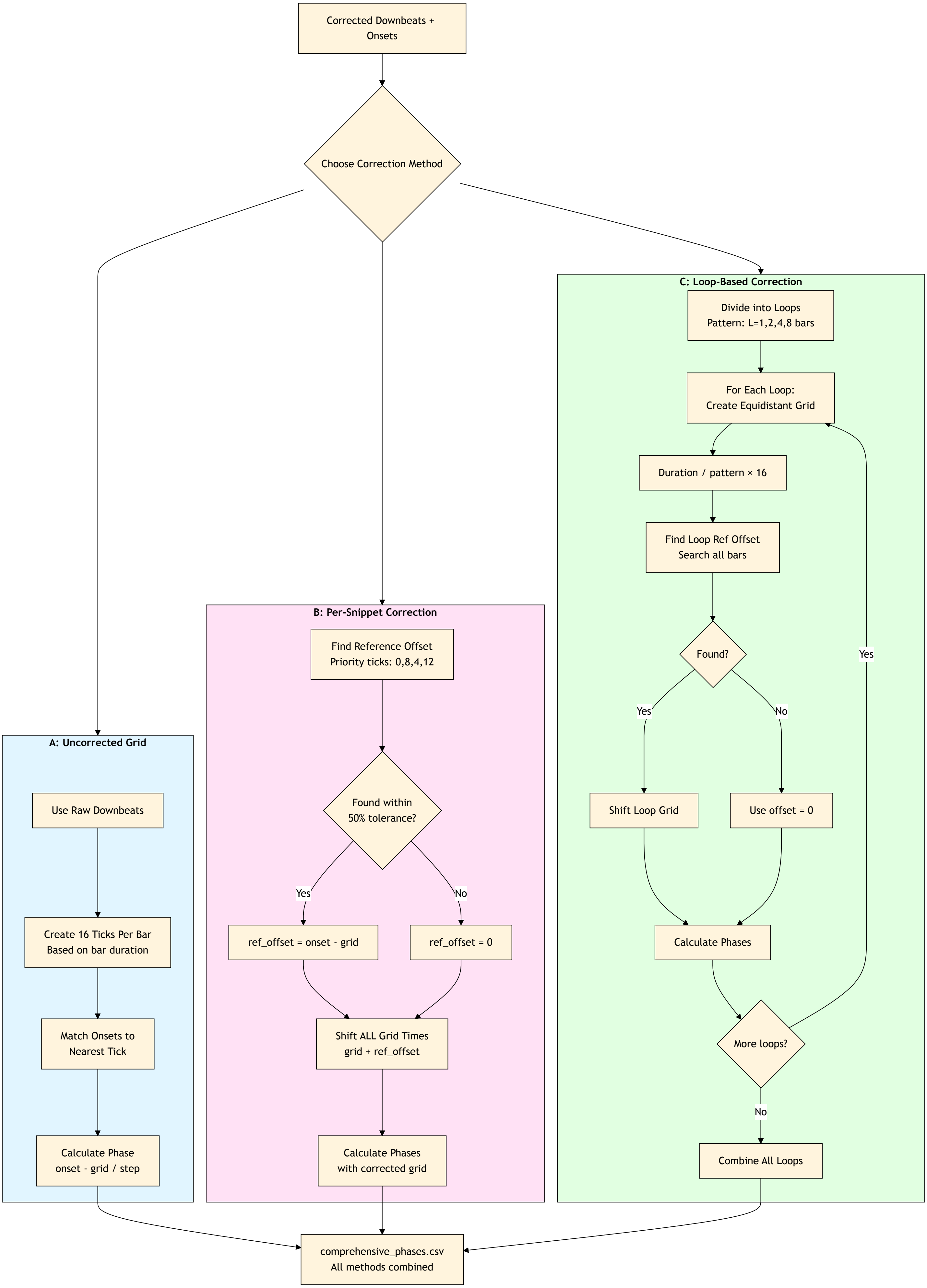
Result: L = 4 bars (highest median at lag 4)

Interpretation:

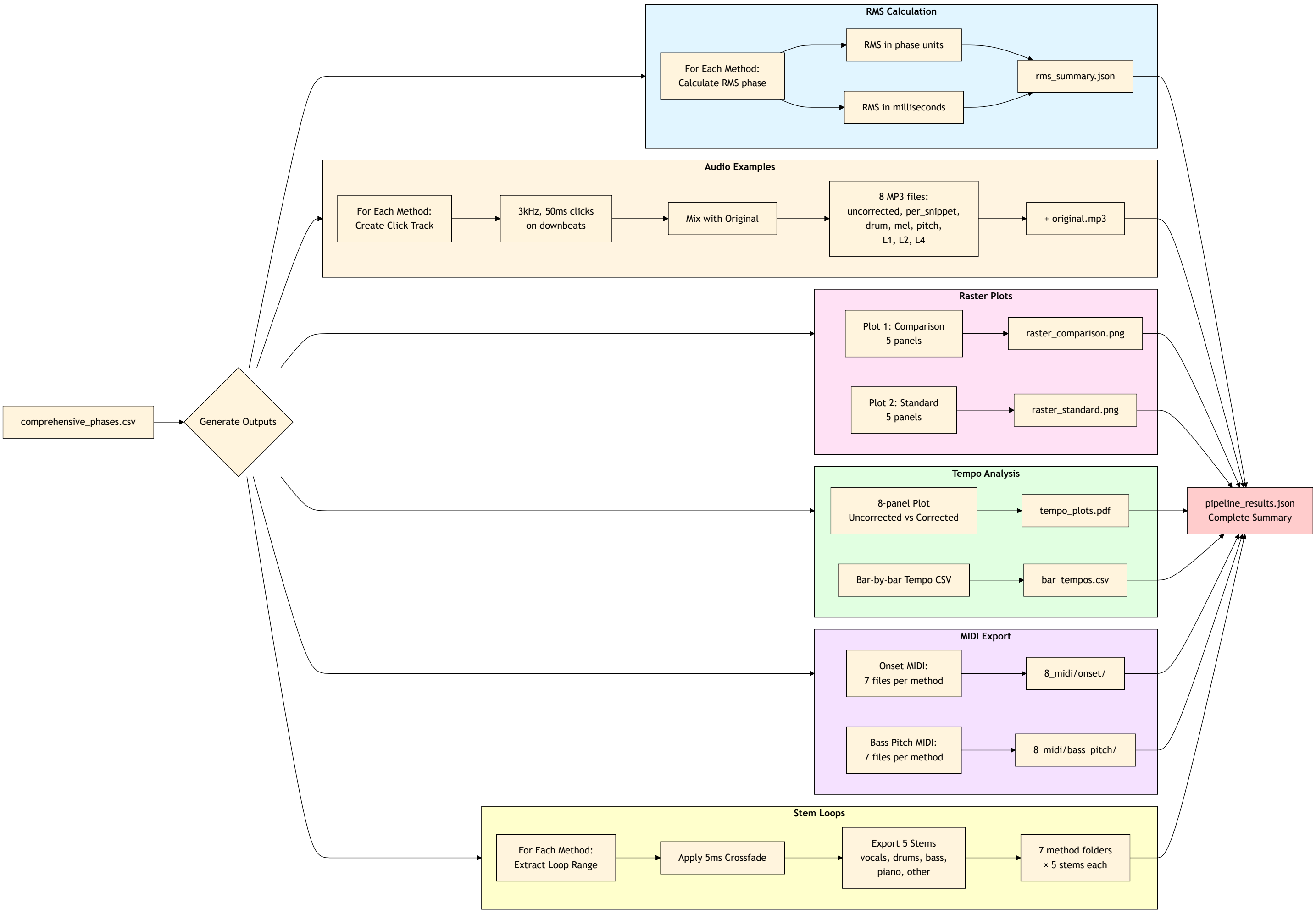
- Bar 0 and Bar 4 are very similar (0.92)
- Bar 1 and Bar 5 are very similar (0.89)
- Bar 2 and Bar 6 are very similar (0.87)
- Bar 3 and Bar 7 are very similar (0.85)

- This indicates a **4-bar repeating pattern**

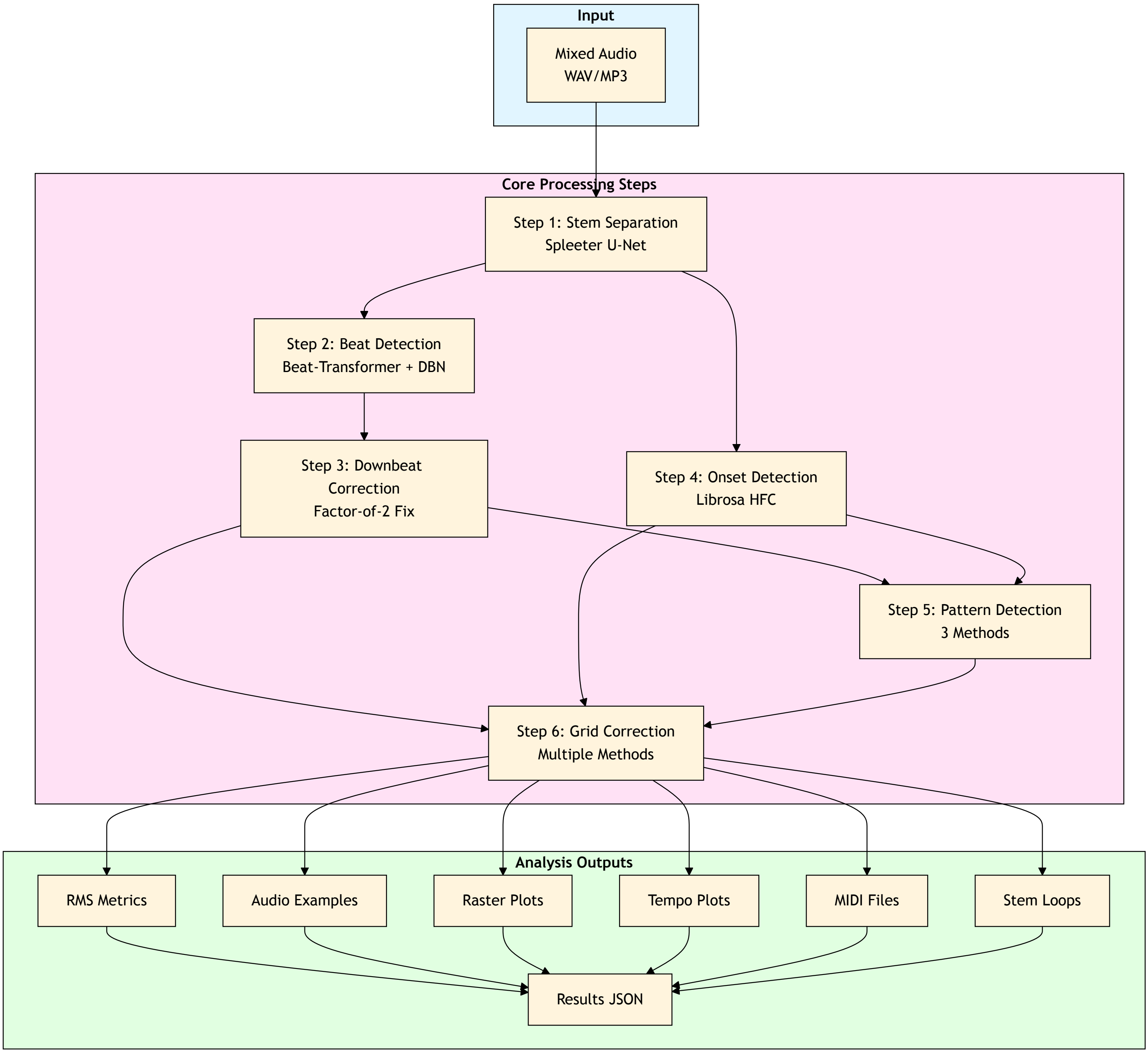
Detailed Step 6: Grid Correction Methods


Corrected Downbeats + Onsets
Choose Correction Method
A: Uncorrected Grid
Use Raw Downbeats
Create 16 Ticks Per Bar
Based on bar duration
Match Onsets to Nearest Tick
Calculate Phase
onset - grid / step
B: Per-Snippet Correction
Find Reference Offset
Priority ticks: 0,8,4,12
Found within 50% tolerance?
Yes
No
ref_offset = onset - grid
ref_offset = 0
Shift ALL Grid Times
grid + ref_offset
Calculate Phases
with corrected grid
C: Loop-Based Correction
Divide into Loops
Pattern: L=1,2,4,8 bars
For Each Loop:
Create Equidistant Grid
Duration / pattern × 16
Find Loop Ref Offset
Search all bars
Found?
Yes
No
Shift Loop Grid
Use offset = 0
Calculate Phases
More loops?
Yes
No
Combine All Loops
comprehensive_phases.csv
All methods combined

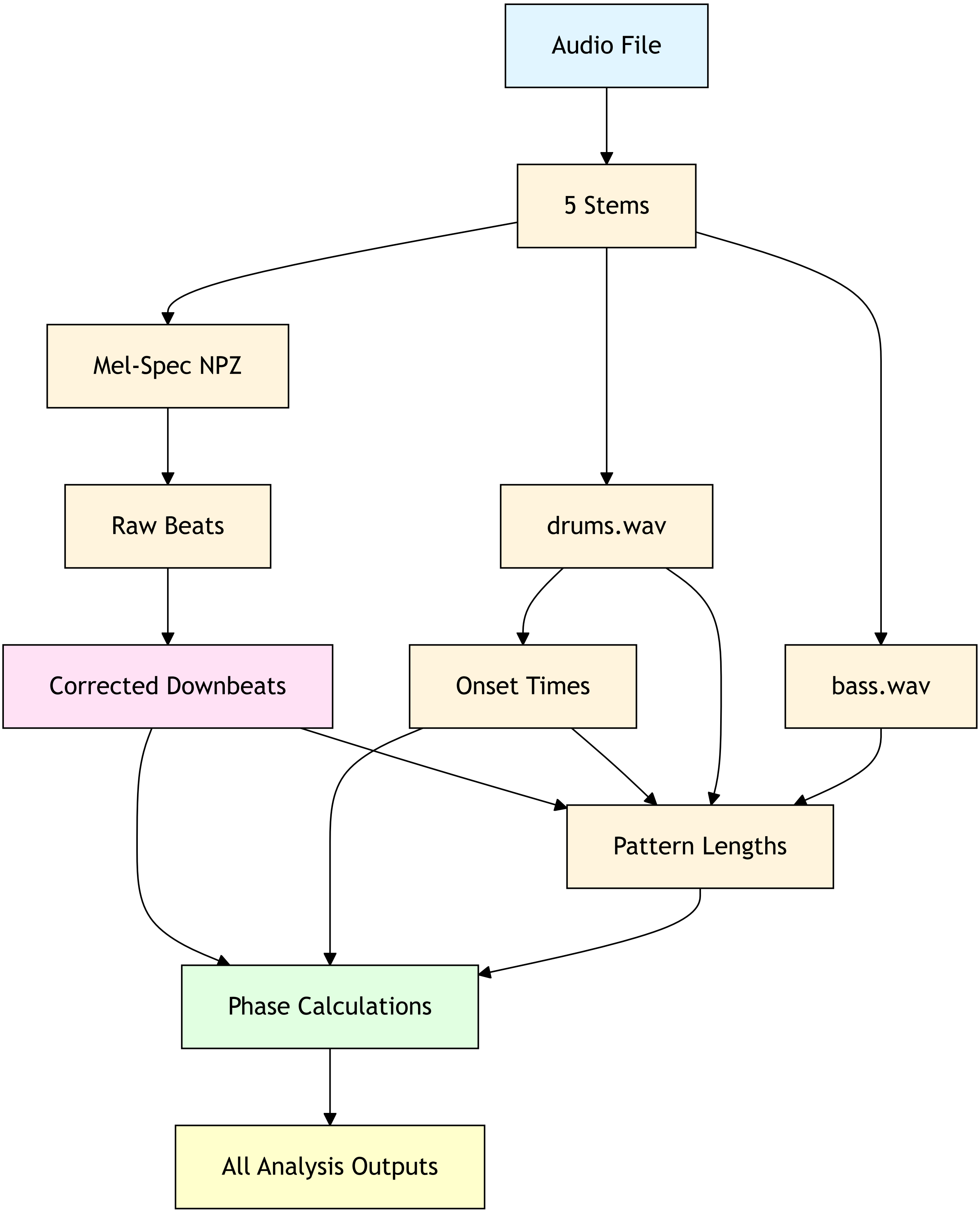
Output Generation Flow


comprehensive_phases.csv
Generate Outputs
RMS Calculation
For Each Method: Calculate RMS phase
RMS in phase units
RMS in milliseconds
rms_summary.json
Audio Examples
For Each Method: Create Click Track
3kHz, 50ms clicks on downbeats
Mix with Original
8 MP3 files: uncorrected, per_snippet, drum, mel, pitch, L1, L2, L4
+ original.mp3
Raster Plots
Plot 1: Comparison 5 panels
raster_comparison.png
Plot 2: Standard 5 panels
raster_standard.png
Tempo Analysis
8-panel Plot Uncorrected vs Corrected
tempo_plots.pdf
Bar-by-bar Tempo CSV
bar_tempos.csv
MIDI Export
Onset MIDI: 7 files per method
8_midi/onset/
Bass Pitch MIDI: 7 files per method
8_midi/bass_pitch/
Stem Loops
For Each Method: Extract Loop Range
Apply 5ms Crossfade
Export 5 Stems vocals, drums, bass, piano, other
7 method folders × 5 stems each
pipeline_results.json Complete Summary

Complete Pipeline Architecture



Data Dependencies



Legend

- **Blue boxes**: Input/intermediate audio data
- **Pink boxes**: Beat/rhythm detection

- **Green boxes**: Pattern/timing analysis
- **Purple boxes**: Grid correction
- **Yellow boxes**: Final outputs
- **Red boxes**: Summary/results files

Notes

- Steps 2 and 4 can run in parallel
- Step 6 produces the most important output: `comprehensive_phases.csv`
- All correction methods are calculated simultaneously and stored in one CSV
- Users can compare methods using RMS metrics or by listening to audio examples