Sonic Pi & Strudel Learning Path

20-Day Journey: From Basics to Live Performance

Optimized for programmers interested in ambient, classical, and neoclassical live coding

---

Overview

This curriculum is designed for someone with strong programming skills learning Sonic Pi and Strudel from scratch. The focus is on creating ambient, trance, and neoclassical music similar to Nils Frahm and Hania Rani.

Schedule: December 12 - January 1 (20 days) **Time commitment:** 2-3 hours per day **Tools needed:**

- Sonic Pi (download from sonic-pi.net)
- Strudel (web-based at strudel.cc)
- Good headphones or speakers
- MIDI keyboard (optional but recommended)

Essential Techniques for Ambient & Neoclassical Music

Before diving into the daily curriculum, let's understand the core principles that make ambient and neoclassical music work. These concepts will be woven throughout your exercises.

Core Principles of Ambient Music

1. Space and Silence

The Rule: What you DON'T play is as important as what you DO play.

Ambient music breathes. It's not about filling every moment with sound, but about creating space for sounds to resonate and decay.

Key Techniques:

- Long release times (4-16 seconds)
- Gaps between notes/phrases
- Low note density (fewer notes per measure)
- Strategic use of silence

```
# BAD: Too cluttered for ambient

live_loop :cluttered do

  use_synth :piano

  play scale(:c4, :minor).choose

  sleep 0.125  # Too fast!

end

# GOOD: Spacious ambient

live_loop :spacious do

  use_synth :piano

  with_fx :reverb, room: 0.9 do

    play scale(:c4, :minor).choose, release: 4, amp: 0.6

    sleep [4, 6, 8].choose  # Plenty of space

  end

end
```

2. Layering Philosophy

The Ambient Stack (from bottom to top):

1. **Foundation Layer (Sub-bass/Drone)** - 1-2 elements
    - Deep sustained tones (often root notes)
    - Very slow movement or static
    - Frequencies: 30-150 Hz

- Purpose: Anchors the piece emotionally

2. **Harmonic Layer (Pads/Chords)** - 2-3 elements

    - Sustained chords or intervals
    - Slow evolution (change every 4-16 bars)
    - Frequencies: 100-800 Hz
    - Purpose: Creates harmonic context

3. **Melodic Layer (Lead/Motif)** - 1-2 elements

    - Sparse melodic fragments
    - Can be absent for long periods
    - Frequencies: 400-2000 Hz
    - Purpose: Gives the ear something to follow

4. **Textural Layer (Atmosphere)** - 2-4 elements

    - Evolving sounds, field recordings, noise
    - Unpredictable, organic
    - Full frequency spectrum
    - Purpose: Creates depth and interest

5. **Transient Layer (Sparkle/Detail)** - 0-2 elements

    - Occasional high-frequency elements
    - Bell-like sounds, granular textures
    - Frequencies: 2000+ Hz
    - Purpose: Adds air and dimension

Critical Rule: Never have all layers playing at once for extended periods. Build and release.

```
# Example of proper layering

use_bpm 65

# Layer 1: Foundation

live_loop :drone do

  use_synth :prophet

  with_fx :reverb, room: 0.9 do
```

```
    play :d2, release: 16, cutoff: 60, amp: 0.4
    sleep 16
  end
end
# Layer 2: Harmonic (enters after 16 seconds)
live_loop :pads, sync: :drone do
  stop if @section == :intro  # Control when this plays

  use_synth :hollow
  with_fx :reverb, room: 0.85 do
    play chord(:d3, :minor), release: 8, amp: 0.3
    sleep 8
  end
end
# Layer 3: Melodic (sparse, enters after 32 seconds)
live_loop :melody, sync: :drone do
  stop if @section == :intro or @section == :breakdown

  use_synth :piano
  with_fx :reverb, room: 0.7 do
    notes = (scale :d4, :minor).take(5)
```

```
    play notes.choose, release: 2, amp: 0.5
    sleep [8, 12, 16].choose  # Very sparse
  end
end
# Layer 4: Texture (evolving atmosphere)
live_loop :atmosphere, sync: :drone do
  sample :ambi_dark_woosh, rate: rrand(0.3, 0.7), amp: 0.3
  sleep [16, 24, 32].choose
end
# Layer 5: Sparkle (occasional, enters during climax only)
live_loop :sparkle, sync: :drone do
  stop if @section != :climax

  use_synth :pretty_bell
  with_fx :reverb, room: 0.95 do
    if one_in(4)  # Only 25% of the time
      play scale(:d5, :minor).choose, release: 4, amp: 0.2
    end
    sleep 4
  end
end
```

3. Frequency Distribution

The Frequency Balance Rule:

Ambient music typically has:

- **Strong low end** (30-150 Hz): 30-40% of energy
- **Rich midrange** (150-800 Hz): 40-50% of energy
- **Delicate highs** (800+ Hz): 10-20% of energy

This creates a "warm" sound that feels enveloping rather than harsh.

```
# Good frequency distribution

live_loop :low do

  use_synth :sine

  play :d2, release: 8, amp: 0.6  # Strong foundation

  sleep 8

end

live_loop :mid do

  use_synth :hollow

  play chord(:d3, :minor), release: 6, amp: 0.5  # Rich middle

  sleep 8

end

live_loop :high do

  use_synth :pretty_bell

  if one_in(3)

    play :d5, release: 3, amp: 0.25  # Delicate highs
```

```
  end

  sleep 4

end
```

4. Movement and Evolution

The Ambient Paradox: Things must change slowly and continuously, yet feel timeless.

Techniques for evolution:

- **Parameter modulation**: Slowly change filter cutoff, reverb amount, pan
- **Generative variation**: Random note selection within constraints
- **Additive development**: Gradually introduce new layers
- **Subtractive development**: Gradually remove layers
- **Textural shifts**: Change sound character every 1-2 minutes

```
live_loop :evolving_pad do

  use_synth :prophet



  # Cutoff evolves over 64 beats

  cutoff_value = (range 60, 100, step: 0.625).tick(:cutoff)



  with_fx :reverb, room: 0.9 do

    with_fx :lpf, cutoff: cutoff_value do

      play chord(:a2, :minor), release: 8, amp: 0.4

      sleep 8

    end

  end

end
```

```
# Alternative: Continuous random drift

live_loop :drifting_texture do

  use_synth :dark_ambience



  note = (@last_note || 48) + [-2, -1, 0, 1, 2].choose

  note = note.clamp(40, 60)  # Keep in range

  @last_note = note



  play note, release: rrand(6, 10), amp: 0.3

  sleep rrand(4, 8)

end
```

Core Principles of Neoclassical Music

1. The Piano as Foundation

In neoclassical (à la Nils Frahm, Hania Rani), the piano is usually the emotional center.

Piano Techniques:

- **Sparse patterns**: 1-4 notes at a time
- **Rhythmic variation**: Avoid strict metronomic timing
- **Dynamic range**: Soft (pp) to moderately loud (mf), rarely forte
- **Pedal simulation**: Long reverb and release times

```
# Neoclassical piano pattern

live_loop :neo_piano do

  use_synth :piano
```

```
  with_fx :reverb, room: 0.75, mix: 0.5 do
    with_fx :hpf, cutoff: 50 do  # Remove mud
      # Arpeggiated pattern, not block chords
      notes = chord(:d3, :minor7).shuffle.take(3)

      notes.each do |n|
        play n, release: rrand(2, 4), amp: rrand(0.5, 0.7)
        sleep [0.25, 0.5, 0.75].choose  # Irregular timing
      end

      sleep [1, 1.5, 2].choose  # Breathing space
    end
  end
end
```

2. Modal Harmony

Neoclassical often uses **modes** rather than traditional major/minor.

Essential modes for your style:

- **Dorian**: Minor with raised 6th (melancholic but hopeful)
- **Phrygian**: Minor with lowered 2nd (dark, Spanish)
- **Aeolian**: Natural minor (classic sad)
- **Mixolydian**: Major with lowered 7th (wistful)

```
# Exploring modes
```

```
live_loop :modal_exploration do
  use_synth :piano

  # Try each mode for 32 beats
  modes = [:dorian, :phrygian, :aeolian, :mixolydian].ring
  current_mode = modes.tick(:mode_change)

  32.times do
    notes = scale(:d3, current_mode).take(7)
    play notes.choose, release: 2, amp: 0.6
    sleep 1
  end
end
```

3. Rhythmic Complexity Through Simplicity

The Nils Frahm Approach:

- Simple patterns that phase against each other
- Polyrhythms (3 against 4, 5 against 4)
- Intentional "human" timing variations

```
# Polyrhythmic neoclassical
live_loop :base_pulse do
  tick
  sleep 1
```

```
end

# Pattern 1: 4-note cycle

live_loop :pattern_1, sync: :base_pulse do

  use_synth :piano

  notes = (scale :d3, :dorian).take(4)

  play notes.tick, release: 1, amp: 0.6

  sleep 1

end

# Pattern 2: 3-note cycle (creates 3:4 polyrhythm)

live_loop :pattern_2, sync: :base_pulse do

  use_synth :prophet

  notes = (scale :d4, :dorian).take(3)

  play notes.tick, release: 2, amp: 0.4

  sleep 4.0/3  # Divides the bar into 3 instead of 4

end

# Pattern 3: 5-note cycle

live_loop :pattern_3, sync: :base_pulse do

  use_synth :hollow

  notes = (scale :d2, :dorian).take(5)

  play notes.tick, release: 3, amp: 0.3

  sleep 4.0/5  # Divides the bar into 5

end
```

4. The Prepared Piano Sound

Characteristics:

- Slightly detuned or "damaged" sound
- Mix of pure and colored timbres
- Emphasis on attack and decay

```
# Simulating prepared piano

live_loop :prepared do

  use_synth :piano



  with_fx :vowel, vowel_sound: 1 do  # Adds character

    with_fx :reverb, room: 0.6 do

      with_fx :bitcrusher, bits: 12 do  # Subtle degradation

        notes = chord(:a3, :minor)



        # Play notes with slight detuning

        notes.each do |n|

          play n, release: rrand(1.5, 3), amp: 0.6

          play n + rrand(-0.1, 0.1), release: rrand(1.5, 3), amp: 0.3

          sleep 0.125

        end



        sleep 4
```

```
    end
  end
 end
end
```

5. Electronic-Acoustic Fusion

Neoclassical often blends acoustic piano with electronic elements.

Layering Strategy:

```
# Acoustic piano (main voice)
live_loop :acoustic do
 use_synth :piano
 with_fx :reverb, room: 0.7 do
  play (scale :f3, :minor).choose, release: 2, amp: 0.8
  sleep 2
 end
end
# Subtle electronic pad (support)
live_loop :electronic, sync: :acoustic do
 use_synth :blade
 with_fx :reverb, room: 0.9 do
  play chord(:f3, :minor), release: 8, amp: 0.2, cutoff: 80
  sleep 8
 end
```

```
end

# Granular texture (atmosphere)

live_loop :granular, sync: :acoustic do

  sample :ambi_glass_hum, rate: rrand(0.5, 1.5), amp: 0.15

  sleep [8, 12, 16].choose

end
```

Arrangement Structures

Ambient Arrangement (20-30 minutes)

INTRO (0-4 min)

├── 0:00 - Silence or barely audible drone

├── 1:00 - First tonal element enters

├── 2:00 - Second layer added

└── 3:00 - Establish key/mood

DEVELOPMENT 1 (4-10 min)

├── Add harmonic complexity

├── Introduce melodic fragments

├── Build density slowly

└── First peak (not climax)

PLATEAU (10-15 min)

├── Maintain full(ish) arrangement

├── Focus on textural evolution

├── Subtle variations

└── Meditative state

DEVELOPMENT 2 (15-22 min)

├── Remove some elements

├── Reintroduce differently

├── Build to main climax

└── Maximum complexity/density

RESOLUTION (22-28 min)

├── Gradual subtraction of layers

├── Return to opening mood

├── Drone/foundation remains

└── Fade to silence or near-silence

OUTRO (28-30 min)

├── Final statement of theme (optional)

└── Complete silence

Neoclassical Arrangement (3-7 minutes)

INTRO (0-30 sec)

└── Solo piano, establishing theme

VERSE 1 (30 sec - 1:30)

├── Theme statement

├── Minimal accompaniment

└── Establish emotional tone

DEVELOPMENT (1:30 - 3:00)

├── Harmonic variation

├── Add electronic elements

├── Rhythmic complexity increases

└── Build intensity

CLIMAX (3:00 - 4:30)

├── Fullest arrangement

├── Polyrhythmic complexity

├── Emotional peak

└── All elements present

BREAKDOWN (4:30 - 5:30)

├── Sudden reduction

├── Return to piano

├── Variation of opening theme

└── Space and breath

OUTRO (5:30 - end)

├── Final thematic statement

├── Gradual reduction

└── End on tonic or suspended

Essential Chord Progressions

For Ambient

Ambient often uses **static harmony** or very slow changes:

```
# Static harmony with movement within
live_loop :ambient_chords do
  use_synth :hollow

  # Stay on Dm for extended period
  chord_notes = chord(:d3, :minor7)

  with_fx :reverb, room: 0.9 do
    # Play different inversions/voicings
    play chord_notes.shuffle.take(3), release: 8, amp: 0.4
    sleep 8
  end
end
# Slow modal progression
live_loop :slow_progression do
  use_synth :prophet

  chords = [
```

```
    chord(:d3, :minor7),    # i7
    chord(:c3, :major7),    # bVII7 (modal borrowed)
    chord(:bb3, :major7),   # bVI7
    chord(:f3, :major7),    # III7
  ]

  with_fx :reverb, room: 0.85 do
    play chords.tick, release: 16, amp: 0.3
    sleep 16  # 16 beats per chord!
  end
end
```

For Neoclassical

Neoclassical uses more traditional progressions but with modal color:

```
# Common neoclassical progressions
live_loop :neo_progression_1 do
  use_synth :piano

  # Progression in D minor (Dorian)
  progression = [
    chord(:d3, :minor7),   # im7
    chord(:g3, :minor7),   # iv7 (Dorian characteristic)
    chord(:c3, :major7),   # bVII7
```

```
    chord(:a3, :minor7),   # v7
  ]

  with_fx :reverb, room: 0.7 do
    progression.each do |ch|
      play ch.take(3), release: 3, amp: 0.6  # Voice leading
      sleep 4
    end
  end
end
# Hania Rani style (more chromatic)
live_loop :neo_progression_2 do
  use_synth :piano

  progression = [
    chord(:a3, :minor),
    chord(:f3, :major),
    chord(:c3, :major),
    chord(:g3, :major),
    chord(:e3, :minor),
    chord(:a3, :minor),
```

```
  ]

  with_fx :reverb, room: 0.65 do
    progression.each do |ch|
      # Arpeggiate instead of block chords
      ch.each do |n|
        play n, release: 2, amp: 0.6
        sleep 0.25
      end
      sleep 1
    end
  end
end
```

Dynamic Control (The Secret Sauce)

Ambient Dynamic Range: pp to mp (very quiet to moderately quiet) **Neoclassical Dynamic Range:** pp to mf (very quiet to moderately loud)

```
# Dynamic shaping in ambient
live_loop :dynamic_ambient do
  use_synth :prophet

  # Build from quiet to medium over 64 beats
```

```
  amp_curve = (range 0.1, 0.5, step: 0.00625).tick(:dynamics)

  with_fx :reverb, room: 0.9 do
    play chord(:d3, :minor), release: 8, amp: amp_curve
    sleep 8
  end
end
# Dynamic contrast in neoclassical
live_loop :dynamic_neo do
  use_synth :piano

  # Alternate between quiet and medium
  dynamics = [0.3, 0.3, 0.6, 0.3, 0.3, 0.7, 0.4, 0.8].ring

  with_fx :reverb, room: 0.7 do
    play scale(:d3, :minor).choose,
      release: 2,
      amp: dynamics.tick
    sleep 1
  end
end
```

Effects Philosophy

Reverb (The Most Important Effect)

For Ambient:

- Room size: 0.8-1.0 (huge spaces)
- Mix: 0.6-0.9 (very wet)
- Damp: 0.3-0.6 (let it ring)

For Neoclassical:

- Room size: 0.5-0.8 (concert hall)
- Mix: 0.4-0.6 (balanced)
- Damp: 0.5-0.7 (controlled)

```
# Ambient reverb

with_fx :reverb, room: 0.95, mix: 0.8, damp: 0.4 do

  use_synth :hollow

  play :c3, release: 8

end

# Neoclassical reverb

with_fx :reverb, room: 0.7, mix: 0.5, damp: 0.6 do

  use_synth :piano

  play :c3, release: 2

end
```

Delay (Depth and Movement)

```
# Ambient: Long, diffuse delays

with_fx :reverb, room: 0.9 do

  with_fx :echo, phase: 1.5, decay: 8, mix: 0.5 do
```

```
    use_synth :blade
    play :c4, release: 3, amp: 0.6
  end
end
# Neoclassical: Rhythmic delays
with_fx :echo, phase: 0.75, decay: 4, mix: 0.3 do
  use_synth :piano
  play :c4, release: 1, amp: 0.7
end
```

Filters (Warmth and Focus)

```
# Low-pass for warmth
with_fx :lpf, cutoff: 90 do
  use_synth :prophet
  play chord(:d3, :minor), release: 8
end
# High-pass for clarity
with_fx :hpf, cutoff: 50 do
  use_synth :piano
  play :c4, release: 2
end
# Slow filter sweep (very ambient)
```

```
live_loop :filter_sweep do
  use_synth :prophet

  cutoff_value = (range 60, 120, step: 0.5).mirror.tick

  with_fx :lpf, cutoff: cutoff_value do
    play chord(:d2, :minor), release: 8, amp: 0.5
    sleep 8
  end
end
```

Week 1: Foundations (Days 1-5)

Day 1: Sonic Pi Basics

Goal: Understand the environment and create your first sounds

Morning Session (1.5 hours):

1. Install Sonic Pi and explore the interface
2. Learn the basic syntax:

```
# Your first sound

play 60

sleep 1

play 64

sleep 1
```

```
play 67

# Using notes instead of MIDI numbers

play :c4

sleep 0.5

play :e4

sleep 0.5

play :g4
```

3. Experiment with:
    - Different note names (`:c4`, `:d4`, `:e4`, etc.)
    - `sleep` durations (0.25, 0.5, 1, 2)
    - The `amp` parameter: `play :c4, amp: 0.5`

Afternoon Session (1 hour):

1. Learn about synths:

```
use_synth :piano

play :c4

use_synth :blade

play :c4

use_synth :hollow

play :c4
```

2. Explore these synths (perfect for your style):
    - `:piano` - classic piano sound
    - `:hollow` - atmospheric pad
    - `:prophet` - warm synth
    - `:blade` - ethereal tones
    - `:dark_ambience` - perfect for ambient work

Exercise 1A: Synth Exploration Create a simple 8-bar melody using different synths. Record yourself explaining what each synth does.

Exercise 1B: First Ambient Layer Apply the foundation layer concept:

```
# Create a drone that lasts 32 seconds

use_synth :prophet

with_fx :reverb, room: 0.9 do

  play :d2, release: 32, amp: 0.4, cutoff: 70

  sleep 32

end
```

Now create three variations:

1. Change the root note to A2
2. Add a second drone 5 semitones higher (very quiet, amp: 0.2)
3. Make the cutoff parameter evolve using `cutoff_slide`

Exercise 1C: Frequency Balance Create three simultaneous loops at different frequency ranges:

- Low (30-150 Hz): Use `:sine` or `:prophet` with notes C2-F2
- Mid (150-800 Hz): Use `:hollow` with notes C3-F3
- High (800+ Hz): Use `:pretty_bell` with notes C5-F5

Make the low 60% volume, mid 40%, high 20%. Listen to how this creates warmth.

Day 2: Loops and Structure

Goal: Build repeating patterns and understand timing

Morning Session (1.5 hours):

```
# Basic loop

loop do
```

```
  play :c4
  sleep 0.5
  play :e4
  sleep 0.5
  play :g4
  sleep 1
end
# Loop a specific number of times
4.times do
  play :c4
  sleep 0.25
end
# Live loops (the foundation of live coding!)
live_loop :melody do
  play :c4
  sleep 0.5
  play :e4
  sleep 0.5
end
```

Afternoon Session (1 hour): Learn about timing and sync:

```
# Multiple synchronized loops
```

```
live_loop :drums do
  sample :drum_bass_hard
  sleep 1
end

live_loop :melody, sync: :drums do
  use_synth :piano
  play :c4
  sleep 0.5
end
```

Exercise 2A: Basic Layering Create three synchronized live loops: one for bass notes, one for melody, one for atmosphere. Start and stop them independently.

Exercise 2B: The 5-Layer Stack Build all 5 layers of the ambient stack (from the Essential Techniques section):

1. Start with foundation (drone)
2. Add harmonic layer after 16 seconds
3. Add melodic layer after 32 seconds
4. Add textural layer after 48 seconds
5. Add sparkle layer after 64 seconds

Use the `@section` variable technique to control when each layer plays.

Exercise 2C: Build and Release Create a 2-minute piece that:

- Builds from 1 layer to 4 layers over 1 minute
- Reduces back to 1 layer over the second minute
- Uses `stop` and conditional logic to control layers

Day 3: Scales and Chords

Goal: Musical theory meets code

Morning Session (1.5 hours):

```
# Working with scales

play scale(:c4, :major)

sleep 1

play scale(:c4, :minor)

sleep 1

play scale(:c4, :dorian)  # Great for ambient!

# Picking notes from scales

play scale(:c4, :minor).choose  # random note

play scale(:c4, :minor).pick(3)  # pick 3 notes

# Chords (essential for your style!)

play chord(:c4, :minor)

sleep 1

play chord(:f3, :minor)

sleep 1

play chord(:g3, :major)
```

Afternoon Session (1 hour): Chord progressions for ambient/neoclassical:

```
# A Nils Frahm-style progression

live_loop :chords do

  use_synth :piano

  use_transpose -12
```

```
  play chord(:a3, :minor), release: 4

  sleep 4

  play chord(:f3, :major), release: 4

  sleep 4

  play chord(:c3, :major), release: 4

  sleep 4

  play chord(:g3, :major), release: 4

  sleep 4

end
```

Exercise 3A: Modal Exploration Create a 16-bar chord progression using minor and major chords. Experiment with the `:minor7`, `:major7`, and `:minor9` chord types.

Exercise 3B: Mode Comparison Create four 16-second sections, each using a different mode:

1. D Dorian - `scale(:d3, :dorian)` - melancholic but hopeful
2. D Phrygian - `scale(:d3, :phrygian)` - dark and mysterious
3. D Aeolian - `scale(:d3, :aeolian)` - natural minor, sad
4. D Mixolydian - `scale(:d3, :mixolydian)` - wistful major

For each, create:

- A sustained chord using the first 3-4 notes
- A simple melody using notes from the scale
- Note how each mode FEELS different

Exercise 3C: Neoclassical Progression Implement this Hania Rani-style progression:

```
# Am - F - C - G - Em - Am

live_loop :progression do

  use_synth :piano
```

```
  chords = [
    chord(:a3, :minor),
    chord(:f3, :major),
    chord(:c3, :major),
    chord(:g3, :major),
    chord(:e3, :minor),
    chord(:a3, :minor),
  ]

  with_fx :reverb, room: 0.7 do
    chords.each do |ch|
      # YOUR CODE: Arpeggiate these chords
      # Play each note separately with 0.25 beat between
      sleep 2  # Each chord gets 2 beats
    end
  end
end
```

Exercise 3D: Voice Leading Take any 4-chord progression. Instead of playing block chords, play them so that:

- Each note moves to the NEAREST note in the next chord
- This creates smooth "voice leading" (very important in neoclassical)

Day 4: Samples and Ambient Textures

Goal: Layer sounds for atmospheric depth

Morning Session (1.5 hours):

```
# Built-in samples

sample :ambi_choir

sleep 4

sample :ambi_dark_woosh

sleep 4

# Manipulating samples

sample :ambi_lunar_land, rate: 0.5  # slower

sample :ambi_lunar_land, rate: 2    # faster

sample :ambi_lunar_land, amp: 0.3   # quieter

# Stretching and reversing

sample :ambi_haunted_hum, beat_stretch: 8

sample :ambi_piano, rpitch: -12  # pitch down
```

Afternoon Session (1 hour): Layering for ambient textures:

```
live_loop :atmosphere do

  sample :ambi_dark_woosh, amp: 0.5, rate: 0.25

  sleep 16

end

live_loop :choir do
```

```
  sample :ambi_choir, amp: 0.3, rate: 0.5, attack: 2, release: 4

  sleep 8

end

live_loop :drone do

  use_synth :hollow

  play :c2, release: 16, amp: 0.4

  sleep 16

end
```

Exercise 4A: Textural Layer Practice Build a 3-minute ambient piece using only samples and drone synths. Focus on creating evolving textures.

Requirements:

- Use at least 3 different ambient samples
- Layer them so they don't all play at once
- Create a sense of evolution (things entering and leaving)
- Apply the frequency balance principle (more low/mid, less high)

Exercise 4B: Ambient Atmosphere Study Recreate the "textural layer" from the 5-layer stack:

- Use 3-4 different samples or synth textures
- Each should be unpredictable (random timing, rate, pitch)
- Create depth by having some sounds far away (quiet, lots of reverb) and some closer
- Nothing should sound "musical" - this is pure atmosphere

Exercise 4C: Sample Manipulation Take ONE sample (`:ambi_choir` or `:ambi_lunar_land`) and create 5 different versions:

1. Original
2. Pitched down 12 semitones (darker)
3. Pitched up 7 semitones (brighter)
4. Stretched to 16 beats
5. Reversed and pitched down

Combine them into a piece where they overlap and interact.

Day 5: Effects and Sound Design

Goal: Transform sounds with effects

Morning Session (1.5 hours):

```
# Basic effects

with_fx :reverb do

  play :c4

end

with_fx :reverb, room: 0.9 do

  play :c4

end

# Chaining effects

with_fx :reverb, room: 0.8 do

  with_fx :echo, phase: 0.5, decay: 4 do

    play :c4

  end

end

# Essential effects for your style:

# :reverb - spatial depth

# :echo - repeating delays

# :lpf - low-pass filter (warmth)
```

```
# :hpf - high-pass filter (clarity)

# :compressor - dynamic control
```

Afternoon Session (1 hour): Building a Hania Rani-style piano piece:

```
live_loop :piano_melody do

  use_synth :piano



  with_fx :reverb, room: 0.8, mix: 0.6 do

    with_fx :hpf, cutoff: 60 do

      notes = (scale :d3, :minor).shuffle.take(4)

      play notes, release: 2, amp: 0.8

      sleep 1

    end

  end

end
```

Exercise 5A: Reverb Comparison Take your Day 3 chord progression and create three versions:

1. Ambient version: room: 0.95, mix: 0.8, damp: 0.4
2. Neoclassical version: room: 0.7, mix: 0.5, damp: 0.6
3. Dry version: No reverb

Listen to how reverb changes the emotional character.

Exercise 5B: Effect Chaining Create a piano melody and experiment with effect order:

```
# Version 1: Reverb then Echo

with_fx :reverb, room: 0.8 do
```

```
  with_fx :echo, phase: 0.5 do
    # your melody
  end
end
# Version 2: Echo then Reverb
with_fx :echo, phase: 0.5 do
  with_fx :reverb, room: 0.8 do
    # your melody
  end
end
```

Notice how the order changes the sound dramatically.

Exercise 5C: Spatial Depth (Essential for Ambient) Create three simultaneous melodies at different "depths":

```
# Close (dry, clear)
live_loop :close do
  use_synth :piano
  with_fx :reverb, room: 0.3, mix: 0.2 do
    play scale(:d4, :minor).choose, amp: 0.8, release: 1
    sleep 2
  end
end
# Medium (balanced)
```

```
live_loop :medium do
  use_synth :blade
  with_fx :reverb, room: 0.7, mix: 0.5 do
    play scale(:d4, :minor).choose, amp: 0.5, release: 2
    sleep 3
  end
end
# Far (washy, distant)
live_loop :far do
  use_synth :hollow
  with_fx :reverb, room: 0.95, mix: 0.85 do
    play scale(:d3, :minor).choose, amp: 0.3, release: 4
    sleep 4
  end
end
```

Exercise 5D: Filter Evolution Create an evolving pad that uses filter automation:

- Start with cutoff at 60 (dark and warm)
- Slowly increase to 110 over 64 beats (brighter)
- Use `cutoff_slide` parameter
- Add this to a full arrangement with bass and melody

Week 2: Intermediate Techniques (Days 6-10)

Day 6: Introduction to Strudel

Goal: Understand Strudel's pattern-based approach

Morning Session (1.5 hours):

1. Open strudel.cc in your browser
2. Understand the mini-notation:

```
// Basic pattern

note("c3 e3 g3 c4").s("piano")

// Rhythm patterns

note("c3 e3 g3").s("piano").slow(2)

// Rests and subdivisions

note("c3 ~ e3 ~").s("piano")

// Stacking (polyphony)

note("c3 e3 g3, c4 e4 g4").s("piano")
```

Afternoon Session (1 hour): Patterns and transformations:

```
// Repeating patterns

note("c3*4 e3*4 g3*8").s("piano")

// Euclidean rhythms (great for ambient!)

note("c3(3,8)").s("piano")

// Speed modulation

note("c3 e3 g3").s("piano").fast(2)
```

```
note("c3 e3 g3").s("piano").slow(4)
```

Exercise: Create the same 8-bar melody you made in Sonic Pi, but in Strudel. Compare the two approaches.

Day 7: Strudel Patterns and Mini-notation

Goal: Master pattern manipulation

Morning Session (1.5 hours):

```
// Pattern operators

note("c3 e3 g3").add(7)  // transpose up

note("<c3 e3 g3 a3>").s("piano")  // cycle through

note("[c3 e3] [g3 a3]").s("piano")  // grouping

// Conditional patterns

note("c3").sometimes(x => x.add(7))

note("c3").often(x => x.rev())

// Pattern functions for ambient

note("c3 e3 g3 c4")

  .s("piano")

  .reverb(0.8)

  .delay(0.5)

  .gain(0.7)
```

Afternoon Session (1 hour): Building evolving patterns:

```
// Generative ambient
```

```
note("<a2 c3 e3 g3>*4")
  .s("piano")
  .add("<0 7 -12>")
  .slow(4)
  .reverb(0.9)
  .delay(0.3)
  .degradeBy(0.1)  // occasional note drops
```

Exercise: Create a pattern that evolves over 32 bars using `<>` alternation and `.sometimes()` transformations.

Day 8: Randomness and Generative Music

Goal: Create self-evolving compositions

Sonic Pi Session (1.5 hours):

```
# Controlled randomness
live_loop :random_melody do
  use_synth :blade

  notes = scale(:c4, :minor_pentatonic)
  play notes.choose, release: 0.5, amp: 0.6
  sleep [0.25, 0.5, 1].choose
end
# Probability-based patterns
```

```
live_loop :bass do
  use_synth :tb303

  if one_in(4)  # 25% chance
    play :c2, release: 0.5, cutoff: 80
  end
  sleep 0.25
end
# Ranged randomness
live_loop :ambience do
  use_synth :hollow

  note = rrand_i(48, 72)  # random note between C3 and C5
  play note, release: rrand(4, 8), amp: 0.3
  sleep rrand(2, 6)
end
```

Strudel Session (1 hour):

```
// Random note selection
note("c3 e3 g3 a3 c4".pickN(4)).s("piano")
// Probabilistic patterns
note("c3").s("piano").sometimesBy(0.3, x => x.add(7))
// Random ranges
```

note(rand.range(48, 72)).s("piano")

Exercise 8A: Controlled Randomness Create a generative piece that plays for 5 minutes without repeating exactly. It should feel intentional, not chaotic.

Requirements:

- Use `scale().choose` for note selection, but limit to ONE scale
- Use `[array].choose` for rhythm, but keep it musical (no 0.01 second sleeps)
- Add probability: use `one_in(n)` to make things happen sometimes
- Ensure it has a clear foundation (drone or bass that doesn't change)

Exercise 8B: Generative Ambient (The Brian Eno Approach) Create a piece where:

- 5 different notes from a scale play at random intervals
- Each note has a random duration between 2-8 seconds
- Each note has a random sleep time between 1-6 seconds
- The piece naturally evolves but never becomes chaotic

```
live_loop :eno_style do

  use_synth :hollow



  notes = scale(:d3, :minor_pentatonic)  # Only 5 notes



  with_fx :reverb, room: 0.95 do

    play notes.choose,

      release: rrand(2, 8),

      amp: rrand(0.3, 0.5)

    sleep rrand(1, 6)

  end

end
```

Add 2-3 more layers with different timing ranges.

Exercise 8C: Probability-Based Composition Create a piece where different events have different probabilities:

- Bass note: plays 100% of the time
- Chord: plays 50% of the time (use `one_in(2)`)
- Melody note: plays 33% of the time (use `one_in(3)`)
- High sparkle: plays 10% of the time (use `one_in(10)`)

This creates organic variation while maintaining structure.

Exercise 8D: Constrained Randomness Use `rrand_i()` to generate notes, but constrain them:

```
live_loop :constrained do

  use_synth :piano



  # Generate a random note, but keep it in a musically useful range

  note = rrand_i(48, 72)  # C3 to C5



  # Only play if it's in our chosen scale

  if scale(:d3, :minor).include?(note)

    play note, release: 2, amp: 0.6

  end



  sleep [1, 2, 3].choose

end
```

This prevents random chaos while allowing variation.

Day 9: MIDI and External Control

Goal: Connect to external instruments and controllers

Morning Session (1.5 hours):

```
# MIDI in Sonic Pi

# Check available MIDI ports

puts midi_ports

# Send MIDI notes

midi_note_on :c4, port: "your_midi_device"

sleep 1

midi_note_off :c4, port: "your_midi_device"

# Control external synths

live_loop :external do

  midi_note_on scale(:c3, :minor).choose,

               velocity: rrand_i(70, 100),

               port: "your_midi_device"

  sleep 0.5

  midi_all_notes_off port: "your_midi_device"

end

# Receive MIDI input

live_loop :midi_in do
```

```
  use_real_time

  note, velocity = sync "/midi/your_controller/0/1/note_on"

  synth :piano, note: note, amp: velocity / 127.0

end
```

Afternoon Session (1 hour): Strudel MIDI setup and OSC communication between Sonic Pi and Strudel.

Exercise: If you have a MIDI keyboard, create a setup where you play chords on the keyboard and Sonic Pi generates arpeggios from those chords.

Day 10: Review and First Performance Piece

Goal: Consolidate learning and create a complete piece

Full Day Project (3 hours):

Create a 5-7 minute composition that demonstrates all you've learned.

REQUIRED ELEMENTS:

- ✅ Proper 5-layer ambient stack OR neoclassical piano + electronic fusion
- ✅ Clear chord progression (use one from the Essential Techniques section)
- ✅ Melody that breathes (space between phrases)
- ✅ Bass/foundation that anchors the piece
- ✅ At least one textural/atmospheric element
- ✅ At least one generative/random element (but controlled)
- ✅ Proper dynamics (build and release, not constant volume)
- ✅ Effects that create depth (reverb, delay, filters)
- ✅ Frequency balance (strong low end, rich midrange, delicate highs)

STRUCTURE (choose ONE):

Option A: Ambient Structure

1. **Intro (0-1 min):** Drone/atmosphere only, very quiet
2. **Build (1-2 min):** Add harmonic layer, then bass
3. **Development (2-4 min):** Add melody and textures, reach first peak

4. **Plateau (4-5 min):** Maintain complexity, focus on evolution
5. **Breakdown (5-6 min):** Remove layers, create space
6. **Outro (6-7 min):** Return to opening mood, fade out

Option B: Neoclassical Structure

1. **Intro (0-30 sec):** Solo piano establishing theme
2. **Verse (30s-1:30):** Piano + subtle pad
3. **Development (1:30-3:00):** Add bass, textures, rhythmic complexity
4. **Climax (3:00-4:30):** Full arrangement, polyrhythms, emotional peak
5. **Breakdown (4:30-5:30):** Sudden reduction, back to piano
6. **Outro (5:30-end):** Final statement, resolution

TECHNICAL REQUIREMENTS:

- Use at least 5 `live_loop`s
- Use `@section` variables to control arrangement
- Include comments explaining your choices
- Save it as `day_10_composition.rb`

EVALUATION CHECKLIST:

- ☐ Does it have clear sections?
- ☐ Does it build and release tension?
- ☐ Are there moments of silence/space?
- ☐ Is the frequency balance good (not too bright)?
- ☐ Does it evoke emotion?
- ☐ Would you want to listen to it again?

BONUS CHALLENGES:

- Create both a Sonic Pi AND a Strudel version
- Record it and listen back tomorrow with fresh ears
- Share it with a friend and get feedback

Week 3: Advanced Techniques (Days 11-15)

Day 11: Advanced Timing and Polyrhythms

Goal: Create complex rhythmic relationships

Morning Session (1.5 hours):

```
# Different time signatures
live_loop :four_four do
  sample :bd_haus
  sleep 1
end
live_loop :three_four, sync: :four_four do
  sample :elec_plip
  sleep 0.75
end
# Polyrhythms - Nils Frahm often uses these!
live_loop :base do
  tick
  sleep 1
end
live_loop :rhythm1, sync: :base do
  sample :perc_bell, amp: 0.3
  sleep 1.5  # Creates a 2:3 polyrhythm
end
live_loop :rhythm2, sync: :base do
  use_synth :piano
```

```
  play scale(:d3, :minor).tick, release: 0.5

  sleep 2.66666  # Creates a 3:8 relationship

end
```

Afternoon Session (1 hour): Strudel polyrhythms:

```
// Different pattern lengths

stack(

  note("c3*4").s("piano"),

  note("e3*3").s("piano").slow(1.5),

  note("g3*5").s("piano").slow(2)

)

// Euclidean polyrhythms

stack(

  note("c2(3,8)").s("sawtooth"),

  note("e2(5,8)").s("sawtooth"),

  note("g2(4,7)").s("sawtooth")

)
```

Exercise 11A: Basic Polyrhythm Create a piece with at least three different rhythmic layers that interact in interesting ways.

Start simple:

```
# 4 against 3 (most common polyrhythm)

live_loop :four do

  sample :perc_bell, amp: 0.6
```

```
  sleep 1

end

live_loop :three, sync: :four do

  sample :perc_bell, amp: 0.4, rate: 1.5

  sleep 4.0/3  # This creates the 3-feel

end
```

Exercise 11B: Nils Frahm Polyrhythm Study Create a three-layer polyrhythm piece:

- Layer 1: 4-note pattern repeating (baseline)
- Layer 2: 3-note pattern (creates 4:3 polyrhythm)
- Layer 3: 5-note pattern (creates 4:5 polyrhythm)

```
use_bpm 90

live_loop :base do

  tick

  sleep 1

end

# Your 4-note pattern (1 beat per note)

live_loop :pattern_4, sync: :base do

  use_synth :piano

  notes = (scale :d3, :dorian).take(4)

  play notes.tick, release: 0.5, amp: 0.7

  sleep 1

end

# Your 3-note pattern (plays 3 notes over 4 beats)
```

```
live_loop :pattern_3, sync: :base do
  use_synth :prophet
  notes = (scale :d4, :dorian).take(3)
  # YOUR CODE: Calculate the sleep time
  # Hint: 4 beats / 3 notes = ?
end

# Your 5-note pattern (plays 5 notes over 4 beats)
live_loop :pattern_5, sync: :base do
  use_synth :hollow
  notes = (scale :d2, :dorian).take(5)
  # YOUR CODE: Calculate the sleep time
end
```

Exercise 11C: Phasing Patterns Create two identical melodies that play at slightly different speeds:

```
live_loop :melody_1 do
  use_synth :piano
  (scale :a3, :minor).take(8).each do |n|
    play n, release: 0.5, amp: 0.6
    sleep 0.5
  end
end

live_loop :melody_2 do
```

```
  use_synth :piano
  (scale :a3, :minor).take(8).each do |n|
    play n, release: 0.5, amp: 0.4
    sleep 0.51  # Slightly slower - they drift apart!
  end
end
```

Let this run for 2 minutes and hear how they phase in and out.

Exercise 11D: Irregular Time Signatures Create a piece in 7/8 time (very common in neoclassical):

```
live_loop :seven_eight do
  use_synth :piano

  # 7 beats grouped as 3+2+2
  pattern = [3, 2, 2]  # Beat groupings
  notes = (scale :d3, :minor).shuffle.take(3)

  pattern.zip(notes).each do |duration, note|
    play note, release: duration * 0.8, amp: 0.7
    sleep duration * 0.25  # Each number = number of 8th notes
  end
end
```

Day 12: Advanced Sound Design

Goal: Craft unique timbres and textures

Morning Session (1.5 hours):

```
# Sonic Pi synth parameters deep dive

use_synth :prophet

play :c3,

  cutoff: 80,        # filter frequency

  res: 0.7,          # resonance

  attack: 0.5,       # envelope attack

  decay: 0.2,        # envelope decay

  sustain: 2,        # envelope sustain

  release: 1,        # envelope release

  amp: 0.8

# Creating evolving pads

live_loop :evolving_pad do

  use_synth :hollow


  with_fx :reverb, room: 0.9 do

    with_fx :lpf, cutoff: rrand(60, 100), cutoff_slide: 4 do

      play chord(:a2, :minor),

        release: 8,
```

```
        amp: 0.5,
        cutoff: 90
      sleep 8
    end
  end
end
# Granular-style textures
live_loop :grains do
  use_synth :dsaw

  64.times do
    play rrand(60, 84),
      release: 0.1,
      amp: 0.05,
      cutoff: rrand(70, 110)
    sleep 0.0625
  end
end
```

Afternoon Session (1 hour):

```
// Strudel sound design
note("c2")
  .s("sawtooth")
```

```
  .cutoff("<400 800 1200 2000>/4")

  .resonance(20)

  .gain(0.7)

  .room(0.9)

// Layered sounds

stack(

  note("c2").s("sine").gain(0.5),

  note("c2").s("sawtooth").cutoff(800).gain(0.3),

  note("c2").s("triangle").delay(0.5).gain(0.2)

)
```

Exercise 12A: Signature Sounds Design three unique "instrument" patches: one for bass, one for pads, one for leads. Save the parameter settings.

Bass Requirements:

- Frequency range: 30-150 Hz
- Should be felt more than heard
- Long sustain or very short (no medium)
- Example starting point:

```
use_synth :prophet

play :d2,

  release: 8,

  cutoff: 70,

  res: 0.5,

  amp: 0.6
```

Pad Requirements:

- Should create atmosphere without dominating
- Long attack (0.5-2 seconds)
- Long release (4-8 seconds)
- Occupies midrange frequencies
- Example:

```
use_synth :hollow

play chord(:d3, :minor),

  attack: 1,

  release: 8,

  cutoff: 90,

  amp: 0.4
```

Lead Requirements:

- Clear and defined
- Can cut through the mix
- Expressive (use filters, envelopes)
- Example:

```
use_synth :blade

play :a4,

  attack: 0.1,

  release: 2,

  cutoff: 100,

  amp: 0.7
```

Exercise 12B: Evolving Pad Masterclass Create a pad sound that evolves over 64 beats using parameter slides:

```
live_loop :evolving_pad do
```

```
  use_synth :prophet


  with_fx :reverb, room: 0.9 do
    with_fx :lpf,
      cutoff: 60,
      cutoff_slide: 8 do |fx|


      play chord(:a2, :minor),
        release: 16,
        cutoff: 60,
        cutoff_slide: 8,
        amp: 0.5


      # Slide the cutoff up
      control fx, cutoff: 110


      sleep 8
    end
  end
end
```

Make it:

- Start dark (cutoff: 60)
- Brighten over time (cutoff: 110)
- Have the filter change create emotional movement

Exercise 12C: Prepared Piano Simulation Create a "prepared piano" sound (like Nils Frahm's felt piano):

```
with_fx :vowel, vowel_sound: 1 do

  with_fx :reverb, room: 0.6, mix: 0.4 do

    with_fx :hpf, cutoff: 80 do  # Remove some low end

      with_fx :compressor do  # Even out dynamics

        use_synth :piano



        # Play with slight detuning for character

        play :c4, release: 3, amp: 0.7

        play :c4 + 0.05, release: 3, amp: 0.3  # Slight detune



        sleep 1

      end

    end

  end

end
```

Experiment with:

- Different vowel_sound values (1-5)
- Different amounts of detuning (+0.05 to +0.2)
- Adding `:bitcrusher` for lo-fi character

Exercise 12D: Granular-Style Textures Create a granular cloud texture (modern ambient technique):

```
live_loop :granular_cloud do

  use_synth :dsaw



  with_fx :reverb, room: 0.95 do

    # Play many short notes rapidly

    32.times do

      play rrand(60, 84),  # Random pitch

        release: rrand(0.05, 0.2),  # Very short

        amp: 0.1,  # Very quiet individually

        cutoff: rrand(70, 110),

        pan: rrand(-0.5, 0.5)  # Stereo spread



      sleep 0.125  # 32nd notes

    end

  end

end
```

This creates a shimmering texture rather than individual notes.

Exercise 12E: Three-Layer Synthesis Layer three different synths for ONE note to create a complex timbre:

```
define :layered_note do |note|

  # Layer 1: Fundamental (low, warm)
```

```
  use_synth :sine
  play note, release: 4, amp: 0.5

  # Layer 2: Harmonics (mid, body)
  use_synth :saw
  play note, release: 4, amp: 0.3, cutoff: 90

  # Layer 3: Sparkle (high, air)
  use_synth :pretty_bell
  play note + 12, release: 3, amp: 0.2
end
with_fx :reverb, room: 0.8 do
  layered_note(:a3)
  sleep 4
end
```

Create your own custom layered sound for your performance.

Day 13: Live Coding Performance Techniques

Goal: Learn to code smoothly during performance

Morning Session (1.5 hours):

Key concepts:

1. **Never stop the music** - modify live_loops while they're running
2. **Comment out, don't delete** - keep options available
3. **Use variables for quick changes**
4. **Prepare variations in advance**

```
# Performance-ready code structure

use_bpm 90

# Global variables for live control

volume = 1.0

complexity = 4

# Base layer (rarely changed)

live_loop :foundation do

  use_synth :prophet

  play :a2, release: 8, amp: 0.3 * volume

  sleep 8

end

# Main performance layer

live_loop :melody do

  use_synth :piano



  notes = scale(:a3, :minor).take(complexity)



  # Uncomment different variations during performance

  # play notes.choose, release: 0.5, amp: 0.8 * volume
```

```
  # play notes.shuffle.take(2), release: 1, amp: 0.8 * volume

  play notes.ring.tick, release: 0.5, amp: 0.8 * volume



  sleep 0.5

end

# Build/release layer (add/remove during performance)

# live_loop :texture do

#   sample :ambi_choir, rate: 0.25, amp: 0.4 * volume

#   sleep 16

# end
```

Afternoon Session (1 hour): Practice techniques:

1. Start with minimal code running
2. Add one live_loop at a time
3. Modify parameters smoothly (use `_slide` parameters)
4. Practice stopping and restarting loops
5. Build muscle memory for common patterns

Exercise: Perform a 10-minute improvisation. Record it. Don't worry about mistakes - focus on keeping the music flowing.

Day 14: Integration and Hybrid Performance

Goal: Combine Sonic Pi and Strudel in one performance

Morning Session (1.5 hours):

```
# Sonic Pi as the foundation

use_bpm 90
```

```
live_loop :sonic_bass do
  use_synth :tb303
  play scale(:d2, :minor).tick,
    release: 0.5,
    cutoff: 70,
    amp: 0.8
  sleep 1
end
live_loop :sonic_pads do
  use_synth :hollow
  with_fx :reverb, room: 0.9 do
    play chord(:d3, :minor), release: 8, amp: 0.3
    sleep 8
  end
end
# Send sync to Strudel via OSC or MIDI clock
```

Strudel patterns (running simultaneously):

```
// Strudel for complex patterns and melodies
setcps(90/60/4)  // Match BPM
note("<d4 f4 a4 c5>*4")
  .s("piano")
  .add("<0 7 -5 12>/8")
```

```
  .sometimes(x => x.rev())

  .gain(0.7)

  .reverb(0.5)
```

Afternoon Session (1 hour): Design a performance template:

- Sonic Pi: Bass, pads, percussion, effects
- Strudel: Melodies, arpeggios, complex patterns
- Plan transitions between sections

Exercise: Create a 15-minute hybrid performance structure with clear sections and transitions.

Day 15: Composition Study

Goal: Analyze and recreate elements of your favorite artists

Full Day (3 hours):

Part 1: Deep Analysis (1.5 hours)

Choose 2 pieces to analyze in depth:

1. One ambient piece (e.g., Brian Eno, Tim Hecker, or similar)
2. One neoclassical piece by Nils Frahm or Hania Rani

For EACH piece, document:

A. Structure Analysis

- Total length
- Number of distinct sections (intro, build, climax, etc.)
- Timing of each section
- How transitions happen (sudden vs. gradual)
- Draw a simple diagram of the arrangement

B. Harmonic Analysis

- What key/mode is it in?

- List the chord progression (if identifiable)
- Is it modal or tonal?
- How often do chords change?
- Are there any surprising harmonic moments?

C. Layering Analysis

- How many distinct layers can you hear?
- List them by frequency range (low/mid/high)
- Which layers are constant? Which come and go?
- What's the maximum number of simultaneous layers?
- Draw a timeline showing when each layer is active

D. Rhythmic Analysis

- What's the approximate BPM?
- Is there a clear pulse or is it floating?
- Are there polyrhythms?
- How does rhythm create forward motion (or not)?

E. Timbral Analysis

- What instruments/sounds can you identify?
- How much reverb/space is there?
- Is it warm or bright (frequency balance)?
- What makes the sound unique?

F. Dynamics Analysis

- What's the loudest moment?
- What's the quietest moment?
- How does volume change over time?
- Are there sudden dynamic shifts?

ANALYSIS TEMPLATE:

PIECE: [Name] by [Artist]

LENGTH: [X:XX]

STRUCTURE:

0:00 - 1:30 | Intro - Solo piano, quiet

1:30 - 3:00 | Build - Add strings

etc.

HARMONY:

Key: D minor (Dorian)

Progression: Dm - C - Bb - F (repeating)

Harmonic rhythm: Changes every 4 bars

LAYERS:

1. Bass drone (constant)

2. Piano (enters 0:30)

3. Strings (enters 1:30)

4. Percussion (enters 2:00)

[Draw timeline]

etc.

Part 2: Recreation (1 hour)

Choose ONE section from ONE piece (30-60 seconds).

Goal: Recreate the ESSENCE, not the exact notes.

Recreation Checklist:

- ☐ Same BPM (±5)
- ☐ Similar number of layers
- ☐ Similar frequency balance
- ☐ Similar spatial quality (reverb)
- ☐ Similar emotional character
- ☐ Similar harmonic movement

What you DON'T need to match:

- Exact notes/melody
- Exact synth sounds
- Every tiny detail

```
# Example recreation approach for Nils Frahm style

use_bpm 85  # Match the original BPM

# Foundation layer (like in original)

live_loop :foundation do

  use_synth :prophet

  play :d2, release: 16, cutoff: 70, amp: 0.4

  sleep 16

end

# Piano pattern (inspired by original)

live_loop :piano_pattern do

  use_synth :piano



  with_fx :reverb, room: 0.7 do

    # Your interpretation of the rhythmic feel

    pattern = [1, 0.5, 0.5, 1, 1.5]

    notes = (scale :d3, :dorian).shuffle.take(5)



    pattern.zip(notes).each do |duration, note|

      play note, release: duration, amp: 0.6
```

```
    sleep duration
  end
 end
end

# Add other layers as in original...
```

Part 3: Transformation (30 minutes)

Take elements from your recreation and create something NEW:

Transformation Options:

1. **Harmonic Variation:**
    - Same rhythm, different chord progression
    - Same progression, different mode (e.g., Dorian → Phrygian)
2. **Rhythmic Variation:**
    - Same chords, different rhythmic pattern
    - Same melody, different meter (4/4 → 7/8)
3. **Timbral Variation:**
    - Same notes, completely different sounds
    - Acoustic → Electronic or vice versa
4. **Structural Variation:**
    - Rearrange the sections
    - Add a new section the original doesn't have

Deliverable: Create a new 2-minute piece that:

- Uses ideas from the original
- Sounds distinctly different
- Shows you understood WHY the original worked

BONUS ANALYSIS EXERCISES:

Exercise A: Compare and Contrast Take two similar pieces (e.g., two Hania Rani pieces) and identify:

- What makes them similar?
- What makes them different?
- What is her "signature" approach?

Exercise B: Historical Context Research:

- Who influenced Nils Frahm? (Arvo Pärt, Brian Eno, etc.)
- What makes "neoclassical" different from classical?
- How does modern ambient differ from 1970s ambient?

Exercise C: Failed Recreation Try to recreate something and intentionally note what DOESN'T work:

- What elements are harder to capture?
- What did you learn from the failures?
- What shortcuts did you discover?

This is extremely valuable - failed attempts teach you more than successes!

REFLECTION QUESTIONS:

1. What surprised you most in your analysis?
2. What technique could you use immediately?
3. What would take more practice to master?
4. Which artist's approach resonates more with you?
5. What did you learn about your own taste?

Common Mistakes and How to Fix Them

Mistake 1: Too Much Reverb

Symptom: Everything sounds muddy and washed out.

Fix:

```
# BAD: Everything drowning in reverb
with_fx :reverb, room: 1.0, mix: 1.0 do
  # All your code
end

# GOOD: Selective reverb
live_loop :dry_element do
  use_synth :piano
  with_fx :reverb, room: 0.3, mix: 0.2 do  # Mostly dry
    play :c4, release: 1, amp: 0.7
    sleep 1
  end
end

live_loop :wet_element do
  use_synth :hollow
  with_fx :reverb, room: 0.9, mix: 0.7 do  # Very wet
    play :c3, release: 8, amp: 0.3
    sleep 8
  end
end
```

Rule: Not everything needs massive reverb. Create depth through contrast.

Mistake 2: Too Many Layers at Once

Symptom: Sounds cluttered, can't hear individual elements.

Fix:

```
# BAD: All 5 layers playing all the time

live_loop :layer1 do

  # ...

end

live_loop :layer2 do

  # ...

end

live_loop :layer3 do

  # ...

end

live_loop :layer4 do

  # ...

end

live_loop :layer5 do

  # ...

end

# GOOD: Layers come and go

@section = :intro
```

```
live_loop :layer1 do
  # Foundation - always plays
end

live_loop :layer2 do
  stop if @section == :intro  # Doesn't play in intro
end

live_loop :layer3 do
  stop if [:intro, :breakdown].include?(@section)  # Only plays during build/climax
end
```

Rule: Maximum 3-4 layers at any given time. Let the music breathe.

Mistake 3: No Dynamic Range

Symptom: Everything is the same volume, no emotional arc.

Fix:

```
# BAD: Static amplitude
live_loop :static do
  play :c4, amp: 0.5
  sleep 1
end

# GOOD: Dynamic shaping
live_loop :dynamic do
  # Volume increases over 32 beats, then decreases
```

```
  amp_value = (range 0.2, 0.8, step: 0.01875).mirror.tick


  play :c4, amp: amp_value
  sleep 1
end
```

Rule: Plan your dynamics. Where's the quietest moment? The loudest?

Mistake 4: Too Much High Frequency Content

Symptom: Sounds harsh, tiring to listen to, not warm.

Fix:

```
# BAD: Bright and harsh
use_synth :saw
play :c4, cutoff: 130, amp: 0.8

# GOOD: Warm and inviting
use_synth :saw
with_fx :lpf, cutoff: 85 do
  play :c4, cutoff: 90, amp: 0.6
end

# BETTER: Frequency-conscious layering
live_loop :low_end do
  use_synth :sine
```

```
  play :c2, release: 8, amp: 0.6  # Lots of bass
  sleep 8
end
live_loop :mids do
  use_synth :hollow
  play chord(:c3, :minor), release: 6, amp: 0.5  # Rich midrange
  sleep 8
end
live_loop :highs do
  use_synth :pretty_bell
  if one_in(3)  # Sparse highs
    play :c5, release: 3, amp: 0.25  # Quiet
  end
  sleep 4
end
```

Rule: Ambient should be 70% low/mid frequencies, 30% high frequencies.

Mistake 5: Ignoring Space and Silence

Symptom: Notes never stop, no breathing room.

Fix:

```
# BAD: Non-stop notes
live_loop :cluttered do
```

```
  play (scale :c4, :minor).choose
  sleep 0.25  # Too fast!
end
# GOOD: Space between phrases
live_loop :spacious do
  # Play 3 notes
  3.times do
    play (scale :c4, :minor).choose, release: 1, amp: 0.6
    sleep 0.5
  end

  sleep 4  # Silence! Let the reverb decay
end
```

Rule: Aim for 30-50% silence in ambient music.

Mistake 6: Forgetting About Bass

Symptom: Music feels thin, not grounding.

Fix:

```
# BAD: No low end
live_loop :thin do
  use_synth :piano
```

```
  play scale(:c4, :minor).choose  # Only playing in middle register
  sleep 1
end
# GOOD: Proper bass foundation
live_loop :bass do
  use_synth :prophet
  play :c2, release: 8, amp: 0.5, cutoff: 70  # Deep bass
  sleep 8
end
live_loop :melody do
  use_synth :piano
  play scale(:c4, :minor).choose, amp: 0.6
  sleep 1
end
```

Rule: Always have something happening below 150 Hz.

Mistake 7: Random Chaos (Not Musical Randomness)

Symptom: Sounds like a computer glitching, not intentional.

Fix:

```
# BAD: Uncontrolled randomness
live_loop :chaos do
  play rrand(20, 100)  # Could be ANY note!
```

```
  sleep rrand(0.01, 5)  # Could be ANY timing!
end
# GOOD: Constrained randomness
live_loop :controlled do
  # Stay within a scale
  notes = scale(:c3, :minor_pentatonic)
  play notes.choose, release: 2, amp: 0.6

  # Musical timing choices
  sleep [1, 2, 3, 4].choose
end
```

Rule: Use randomness within musical constraints (scales, sensible timing).

Mistake 8: No Clear Arrangement Structure

Symptom: Piece meanders, no sense of journey.

Fix: Plan your structure BEFORE coding:

```
MY PIECE STRUCTURE:
0:00-1:00 | INTRO    | Drone only, @intensity = 0
1:00-2:30 | BUILD 1  | Add harmony, @intensity = 1
2:30-4:00 | CLIMAX   | Full arrangement, @intensity = 2
4:00-5:00 | BREAK    | Remove layers, @intensity = 3
```

5:00-6:00 | OUTRO | Back to drone, @intensity = 0

Use variables to control sections:

```
@intensity = 0  # Start at intro

live_loop :control do
  # Change intensity manually or programmatically
  # Uncomment the section you're working on:

  # @intensity = 1  # Build
  # @intensity = 2  # Climax
  # @intensity = 3  # Breakdown

  sleep 16
end
```

Rule: Know where you're going before you start coding.

Mistake 9: Ignoring the Piano's Natural Character

Symptom: Piano sounds robotic, not expressive.

Fix:

```
# BAD: Mechanical piano
live_loop :robot_piano do
  use_synth :piano
  play :c4, amp: 0.5, release: 1
```

```
  sleep 1
end
# GOOD: Humanized piano
live_loop :human_piano do
  use_synth :piano

  # Vary velocity
  velocity = rrand(0.5, 0.8)

  # Vary timing slightly (not perfectly quantized)
  timing_variation = rrand(-0.02, 0.02)

  play :c4, amp: velocity, release: rrand(0.8, 1.2)
  sleep 1 + timing_variation
end
# BETTER: Musical dynamics and phrasing
live_loop :musical_piano do
  use_synth :piano

  with_fx :reverb, room: 0.7 do
    # Phrase 1: Growing louder
```

```
    4.times do |i|
      amp = (0.4 + (i * 0.1))
      play (scale :a3, :minor).tick, release: 1.5, amp: amp
      sleep 1
    end

    sleep 2  # Breathing space

    # Phrase 2: Softer
    4.times do
      play (scale :a3, :minor).tick, release: 1, amp: 0.4
      sleep 1
    end

    sleep 4  # Longer breath
  end
end
```

Rule: Add humanity through variation, not perfection.

Mistake 10: Overusing the Same Chord Progression

Symptom: Gets boring after the first minute.

Fix:

```
# BAD: Same 4 chords forever
live_loop :boring do
  [chord(:c3, :minor), chord(:f3, :major),
   chord(:g3, :major), chord(:c3, :minor)].each do |ch|
    play ch
    sleep 4
  end
end
# GOOD: Progression with variation
live_loop :interesting do
  progression_a = [
    chord(:a3, :minor7),
    chord(:f3, :major7),
    chord(:c3, :major7),
    chord(:g3, :major7)
  ]

  progression_b = [
    chord(:a3, :minor7),
    chord(:f3, :major7),
    chord(:c3, :major7),
```

```
    chord(:e3, :minor7)  # Different ending!
  ]

  # Play progression_a twice
  2.times do
    progression_a.each do |ch|
      play ch, release: 3, amp: 0.6
      sleep 4
    end
  end

  # Then progression_b once (creates question/answer)
  progression_b.each do |ch|
    play ch, release: 3, amp: 0.6
    sleep 4
  end
end
```

Rule: Create variations within your progressions to maintain interest.

Quick Diagnostic Checklist

If your piece isn't working, ask:

Sound Quality:

- [ ] Is there too much reverb? (Do elements blend into mud?)
- [ ] Is it too bright/harsh? (Add low-pass filter)
- [ ] Is the bass present? (Check for sub-150 Hz content)
- [ ] Are effects appropriate? (Less is often more)

Arrangement:

- [ ] Are there too many layers? (Maximum 4 simultaneous)
- [ ] Is there enough space/silence? (Aim for 30-50%)
- [ ] Does it build and release? (Dynamic arc)
- [ ] Can I identify distinct sections? (Intro/build/climax/outro)

Musical:

- [ ] Does it stay in one scale/mode? (Avoid random note chaos)
- [ ] Is the timing musical? (Avoid random micro-timings)
- [ ] Does the harmony make sense? (Chord relationships)
- [ ] Is there too much or too little repetition?

Emotional:

- [ ] Does it evoke a feeling?
- [ ] Would I want to listen to this again?
- [ ] Does it go on a journey?
- [ ] Are there memorable moments?

Week 4: Performance and Refinement (Days 16-20)

Day 16: Building a Live Set

Goal: Create a 20-30 minute performance-ready set

Structure Planning (1 hour):

OPENING (0-5 min)

- Slow build from silence

- Introduce main atmosphere

- Establish tonal center

DEVELOPMENT (5-15 min)

- Section 1: Add rhythm and bass

- Section 2: Introduce main themes

- Section 3: Variation and complexity

CLIMAX (15-20 min)

- Fullest arrangement

- Most energy and complexity

RESOLUTION (20-25 min)

- Gradual reduction

- Return to opening themes

- Fade to silence

ENCORE/VARIATION (25-30 min, optional)

- Alternative take on themes

- Improvisational section

Coding Session (2 hours): Implement this structure with:

- Clear section markers
- Transition plans
- Backup variations
- Emergency "save" patterns

Day 17: Performance Practice

Goal: Rehearse your set multiple times

Morning (1.5 hours):

- Perform your entire set start to finish
- Don't stop for mistakes
- Record the audio

Afternoon (1.5 hours):

- Listen to the recording critically
- Note problem areas
- Practice transitions between sections
- Refine timing and dynamics

Evening:

- Perform again and record
- Compare to morning session
- Identify improvements

Day 18: Visual and Stage Presence

Goal: Think beyond just the code

Morning Session (1.5 hours):

1. Learn about screen sharing and projection:
    - What code should be visible?
    - Font size for audience visibility
    - Syntax highlighting schemes
2. Practice explaining what you're doing:
    - Brief verbal cues during performance
    - How to make coding visible/understandable
3. Physical setup:

- Equipment positioning
- Lighting considerations
- Backup systems

Afternoon Session (1 hour): Add visual elements:

```
# Sonic Pi can trigger visuals via OSC

# Send messages to Processing, TouchDesigner, etc.

live_loop :visuals do

  osc_send "localhost", 7000, "/intensity", rrand(0.3, 0.8)

  sleep 0.25

end
```

Exercise: Perform your set while recording both screen and audio. Watch it back as if you're an audience member.

Day 19: Improvisation and Interaction

Goal: Develop spontaneous creativity

Session 1 (1 hour): Constraint-based improvisation Set constraints and improvise for 15 minutes:

- Only use 3 notes
- Only use one synth
- Only use samples, no synths
- Only use rhythm, no pitch

Session 2 (1 hour): Theme variations Pick a simple theme (4-note melody) and improvise variations:

- Harmonic variations (different chords)
- Rhythmic variations (different timings)
- Textural variations (different sounds)
- Structural variations (different forms)

Session 3 (1 hour): Reactive improvisation Have a friend play music or sounds, and improvise in response using Sonic Pi/Strudel.

Day 20: Final Performance and Reflection

Goal: Culminate your learning journey

Morning: Final Rehearsal (1.5 hours)

- Run through entire set
- Make final adjustments
- Ensure all code is commented and organized

Afternoon: Performance (1 hour)

- Record a final performance
- Invite friends/family to watch (in person or stream)
- Embrace imperfection - live coding is about the journey

Evening: Reflection (30 minutes)

- Listen to your performance
- Write down:
    - What you learned
    - What surprised you
    - What you want to explore next
    - Three goals for continued practice

Additional Resources

Essential Sonic Pi Resources

- Official Tutorial: Built into Sonic Pi (Help > Tutorial)
- Sonic Pi Forum: in-thread.sonic-pi.net
- Sam Aaron's performances on YouTube
- "Sonic Pi - The Live Coding Synth for Everyone" by Sam Aaron

Essential Strudel Resources

- Strudel Documentation: strudel.cc/learn
- Tidal Cycles documentation (Strudel's parent language)
- TOPLAP community (livecoding.toplap.org)

Music Theory for Live Coders

- Scales to explore: Minor, Dorian, Phrygian, Minor Pentatonic, Harmonic Minor
- Chord progressions: i-VI-III-VII, i-v-VI-III, i-VII-VI-V (all in minor)
- For your style, focus on: Sustained tones, sparse arrangements, modal harmony

Community and Inspiration

- Algorave videos on YouTube
- TOPLAP performances
- Eulerroom (community of live coders)
- Andrew Sorensen (Extempore/Impromptu creator)
- Anna Xambó (research on live coding performance)

Practice Routines (Post-December)

Daily (20-30 minutes)

- Warm up with a simple improvisation
- Learn one new technique or parameter
- Code for 15 minutes without stopping

Weekly (2-3 hours)

- Create one complete short piece (3-5 minutes)
- Experiment with a new concept
- Watch/analyze other live coders' performances

Monthly

- Perform (even if just for yourself)
- Share code with the community
- Set new learning goals

Final Tips

1. **Start Simple**: Resist the urge to create complexity immediately. The most beautiful ambient music is often the simplest.

2. **Listen First**: Spend as much time listening to your code as writing it.

3. **Version Control**: Save different versions of your pieces. Use Git or just dated files.

4. **Record Everything**: You'll create happy accidents you'll want to remember.

5. **Join the Community**: Share your work, ask questions, attend online events.

6. **Embrace Mistakes**: In live coding, "mistakes" often become the most interesting moments.

7. **Your Style**: Don't try to copy Nils Frahm or Hania Rani - let their work inspire your own unique voice.

8. **Technical Issues**: Always have a backup plan. Technology fails - especially during performances!

9. **Perform Often**: Even 5-minute performances for friends will accelerate your learning.

10. **Have Fun**: This is the most important rule. If you're not enjoying it, change something.

Technique Deep-Dives: Essential Exercises

These exercises can be practiced any time to master specific techniques essential for ambient and neoclassical music.

Technique 1: Creating Evolving Drones

The Goal: A drone that changes subtly over time without losing its hypnotic quality.

Exercise:

```
live_loop :master_drone do
  use_synth :prophet

  # Slowly evolving cutoff (takes 128 beats to complete cycle)
  cutoff_value = (range 60, 100, step: 0.3125).mirror.tick(:cutoff)

  # Slowly evolving amplitude (different cycle length creates complexity)
  amp_value = (range 0.3, 0.5, step: 0.0025).mirror.tick(:amp)

  with_fx :reverb, room: 0.95 do
    with_fx :lpf, cutoff: cutoff_value do
      play :d2,
        release: 16,
        amp: amp_value,
        cutoff: cutoff_value

      sleep 16
    end
  end
end
```

Variations to try:

- Add a second drone a fifth higher
- Slowly pan left and right
- Modulate the reverb room size
- Layer 3 drones at different octaves

Technique 2: Arpeggio Patterns (Neoclassical)

The Goal: Create flowing, organic arpeggios like Hania Rani uses.

Exercise:

```
live_loop :arpeggio do
  use_synth :piano

  # Define chord
  chord_notes = chord(:a3, :minor7)

  with_fx :reverb, room: 0.7 do
    # Arpeggiate up
    chord_notes.each do |note|
      play note, release: 1.5, amp: 0.7
      sleep 0.25
    end

    # Then down (excluding root to avoid repetition)
    chord_notes.reverse[1..-1].each do |note|
```

```
    play note, release: 1.5, amp: 0.6
    sleep 0.25
  end
 end
end
```

Advanced variations:

- Random note order: `chord_notes.shuffle`
- Skip some notes: `if one_in(3)`
- Irregular timing: `sleep [0.25, 0.5, 0.75].choose`
- Change chord every 4 bars

Technique 3: Textural Clouds

The Goal: Create shimmering, atmospheric textures that sit in the background.

Exercise:

```
live_loop :cloud do
 use_synth :hollow

 # Define note pool
 notes = scale(:d4, :minor_pentatonic)

 with_fx :reverb, room: 0.95, mix: 0.9 do
  with_fx :hpf, cutoff: 80 do
   # Play cluster of notes
```

```
    4.times do
      play notes.choose,
        release: rrand(2, 6),
        amp: rrand(0.1, 0.2),
        pan: rrand(-0.7, 0.7)
      sleep rrand(0.1, 0.5)
    end

    sleep rrand(3, 8)  # Long gap between clusters
  end
 end
end
```

Purpose: These textures add richness without drawing attention.

Technique 4: Call and Response

The Goal: Two voices that interact, like a conversation.

Exercise:

```
# Voice 1: The question
live_loop :call do
 use_synth :piano

 with_fx :reverb, room: 0.7 do
```

```
    notes = (scale :d4, :minor).take(3)

    notes.each do |n|
      play n, release: 1, amp: 0.7
      sleep 0.5
    end

    sleep 2  # Wait for response
  end
end
# Voice 2: The answer
live_loop :response, sync: :call do
  use_synth :blade

  sleep 2  # Wait for call to finish

  with_fx :reverb, room: 0.8 do
    notes = (scale :d5, :minor).take(3).reverse

    notes.each do |n|
      play n, release: 0.8, amp: 0.5
```

```
    sleep 0.5
  end

  sleep 2
 end
end
```

Technique 5: Rhythmic Displacement

The Goal: Play the same pattern but shift when it starts (very common in neoclassical).

Exercise:

```
# Original pattern (starts on beat 1)
live_loop :original do
 use_synth :piano

 pattern = [1, 0.5, 0.5, 1, 1]
 notes = (scale :a3, :minor).take(5)

 with_fx :reverb, room: 0.6 do
  pattern.zip(notes).each do |duration, note|
   play note, release: duration * 0.8, amp: 0.6
   sleep duration
  end
```

```
  end
end
# Displaced pattern (starts on beat 2)
live_loop :displaced, sync: :original do
  sleep 1  # Offset by 1 beat

  use_synth :prophet

  pattern = [1, 0.5, 0.5, 1, 1]
  notes = (scale :a4, :minor).take(5)

  with_fx :reverb, room: 0.7 do
    pattern.zip(notes).each do |duration, note|
      play note, release: duration * 0.8, amp: 0.4
      sleep duration
    end
  end
end
```

This creates beautiful rhythmic tension.

Technique 6: Dynamic Swells

The Goal: Create crescendos and decrescendos (essential for emotional impact).

Exercise:

```
live_loop :swell do
  use_synth :hollow


  # Crescendo (quiet to loud)
  with_fx :reverb, room: 0.9 do
    play chord(:d3, :minor),
      attack: 4,      # 4 beats to reach full volume
      sustain: 4,     # Hold for 4 beats
      release: 4,     # 4 beats to silence
      amp: 0.6


    sleep 12
  end
end
# Add melodic interest during swell
live_loop :swell_melody, sync: :swell do
  use_synth :piano
```

```
  sleep 4  # Wait for attack phase


  # Play during sustain (most energy)
  with_fx :reverb, room: 0.7 do
    4.times do
      play (scale :d4, :minor).choose, release: 1, amp: 0.7
      sleep 1
    end
  end


  sleep 4  # Let release phase finish
end
```

Technique 7: Harmonic Rhythm Control

The Goal: Control how often chords change to create tension/release.

Exercise:

```
define :play_progression do |chord_duration|
  progression = [
    chord(:a3, :minor),
    chord(:f3, :major),
    chord(:c3, :major),
    chord(:g3, :major),
```

```
  ]

  use_synth :piano

  with_fx :reverb, room: 0.75 do
    progression.each do |ch|
      play ch, release: chord_duration * 0.8, amp: 0.6
      sleep chord_duration
    end
  end
end
# Fast harmonic rhythm (tension, forward motion)
live_loop :fast_changes do
  play_progression(1)  # Changes every beat
end
# Slow harmonic rhythm (calm, meditative)
live_loop :slow_changes do
  play_progression(8)  # Changes every 8 beats
end
```

Discovery: Slow harmonic rhythm = ambient feel. Fast = neoclassical feel.

Technique 8: Pedal Point (Drone + Movement)

The Goal: Keep one note constant while harmony moves above it.

Exercise:

```
# The pedal (constant note)
live_loop :pedal do
  use_synth :sine
  play :a2, release: 16, amp: 0.5
  sleep 16
end
# Moving harmony above
live_loop :movement, sync: :pedal do
  use_synth :piano

  chords = [
    chord(:a3, :minor),   # Consonant (stable)
    chord(:f3, :major),   # Mildly dissonant
    chord(:c3, :major),   # More dissonant
    chord(:g3, :major),   # Tension!
  ]

  with_fx :reverb, room: 0.7 do
```

```
    chords.each do |ch|
      play ch, release: 3, amp: 0.6
      sleep 4
    end
  end
end
```

This creates beautiful tension as chords clash then resolve with the pedal.

Technique 9: Metric Modulation

The Goal: Make it feel like the tempo changes without actually changing BPM.

Exercise:

```
use_bpm 90
# Standard 4/4 feel
live_loop :original_tempo do
  sample :bd_haus
  sleep 1
end
# Play triplets (3 notes per beat) - feels faster
live_loop :triplet_feel, sync: :original_tempo do
  use_synth :piano

  notes = (scale :d3, :minor).take(3)
```

```
  3.times do
    play notes.tick, release: 0.3, amp: 0.6
    sleep 1.0/3  # Three notes in one beat
  end
end
# Play half-time - feels slower
live_loop :half_time, sync: :original_tempo do
  use_synth :prophet

  play (scale :d2, :minor).choose, release: 2, amp: 0.4
  sleep 2  # Every two beats
end
```

Technique 10: The Ambient Build Template

The Goal: A reusable template for building ambient pieces.

Complete Exercise:

```
use_bpm 70
# Control variables
@intensity = 0  # 0 = intro, 1 = build, 2 = climax, 3 = outro
# Layer 1: Always present
```

```
live_loop :foundation do
  use_synth :prophet
  with_fx :reverb, room: 0.95 do
    play :d2, release: 16, amp: 0.4, cutoff: 70
    sleep 16
  end
end
# Layer 2: Enters during build
live_loop :harmony do
  stop if @intensity < 1

  use_synth :hollow
  with_fx :reverb, room: 0.9 do
    play chord(:d3, :minor), release: 8, amp: 0.3
    sleep 8
  end
end
# Layer 3: Enters during climax
live_loop :melody do
  stop if @intensity < 2

  use_synth :piano
```

```
  with_fx :reverb, room: 0.7 do
    notes = (scale :d4, :minor).take(5)
    play notes.choose, release: 2, amp: 0.6
    sleep [4, 6, 8].choose
  end
end
# Layer 4: Only during climax
live_loop :sparkle do
  stop if @intensity != 2

  use_synth :pretty_bell
  with_fx :reverb, room: 0.95 do
    if one_in(3)
      play (scale :d5, :minor).choose, release: 4, amp: 0.3
    end
    sleep 2
  end
end
# Controller: Change @intensity to build/release
# @intensity = 1  # Build starts
# @intensity = 2  # Climax
```

@intensity = 3 # Outro (only foundation plays)

How to use:

1. Start with `@intensity = 0`
2. Every 32 seconds, increment: `@intensity = 1`, then `@intensity = 2`, etc.
3. Practice timing your changes to create natural builds

Practice Routine: Daily 15-Minute Drills

Pick ONE technique per day and spend 15 minutes:

Minutes 1-5: Code the example **Minutes 6-10:** Modify one parameter at a time **Minutes 11-15:** Create your own variation

Weekly Cycle:

- Monday: Drones
- Tuesday: Arpeggios
- Wednesday: Textures
- Thursday: Call & Response
- Friday: Swells
- Saturday: Pedal Points
- Sunday: Free composition using all techniques

Challenge Projects

Once you complete the 20 days, try these:

Project 1: Album

Create a 30-minute "album" of 5-6 pieces using only Sonic Pi or Strudel.

Project 2: Remix

Take an existing classical piece and create a live-coded interpretation.

Project 3: Collaboration

Find another live coder online and create a collaborative piece (asynchronously or in real-time).

Project 4: Visual Integration

Connect your code to a visual system (Processing, TouchDesigner, or even simple p5.js).

Project 5: Instrument Design

Create a custom "instrument" in Sonic Pi - a live_loop that responds to specific inputs in complex ways.

Your Journey Awaits

This curriculum is a starting point, not a rulebook. Feel free to:

- Skip sections that don't interest you
- Spend more time on areas you love
- Combine days or reorder them
- Go deeper into specific techniques

The beauty of live coding is that it combines your programming skills with musical creativity. There's no "right way" - only your way.

Most importantly: **Make sound every single day**. Even 10 minutes of noodling in Sonic Pi will keep your skills sharp and your creativity flowing.

Good luck, and enjoy your journey into the world of live coded music!

"The computer is not a thing, it's a place. It's a musical instrument in the same way a piano is."
— Sam Aaron, creator of Sonic Pi

Quick Reference Guide

Essential Scales for Ambient/Neoclassical

```
# Minor family (melancholic, emotional)

scale(:d3, :minor)          # Natural minor - sad
```

```
scale(:d3, :dorian)          # Dorian - hopeful melancholy (Nils Frahm favorite)

scale(:d3, :phrygian)        # Phrygian - dark, mysterious

scale(:d3, :aeolian)         # Same as natural minor

scale(:d3, :harmonic_minor)  # Exotic, classical

# Pentatonic (safe, consonant)

scale(:d3, :minor_pentatonic)  # Only 5 notes, always sounds good

scale(:d3, :major_pentatonic)  # Happy, open

# Major family (bright, hopeful)

scale(:d3, :major)           # Classic major

scale(:d3, :mixolydian)      # Major with b7 - wistful

scale(:d3, :lydian)          # Bright, dreamy

# Exotic (use sparingly)

scale(:d3, :whole_tone)      # Ambient, floating

scale(:d3, :chromatic)       # All 12 notes
```

Essential Chord Types

```
# Basic triads

chord(:a3, :major)      # Happy - A, C#, E

chord(:a3, :minor)      # Sad - A, C, E

# Seventh chords (richer, more sophisticated)

chord(:a3, :major7)     # Dreamy - A, C#, E, G#
```

```
chord(:a3, :minor7)     # Smooth sad - A, C, E, G
chord(:a3, :dom7)       # Tense - A, C#, E, G
# Extended chords (very neoclassical)
chord(:a3, :minor9)     # Add 9th - lush
chord(:a3, :major9)     # Add 9th - ethereal
chord(:a3, :add9)       # Triad + 9th
# Suspended (floating, unresolved)
chord(:a3, :sus2)       # No 3rd, has 2nd
chord(:a3, :sus4)       # No 3rd, has 4th
```

Essential Synths by Use Case

For Bass/Foundation:

```
:sine       # Pure, clean sub-bass
:prophet    # Warm, analog bass
:tb303      # Acidic, moving bass
:subpulse   # Deep rumble
```

For Pads/Atmosphere:

```
:hollow     # Ethereal, airy
:dark_ambience  # Mysterious, deep
:prophet    # Warm, analog
:pad        # Classic synth pad
:blade      # Smooth, rich
```

For Melody/Lead:

```
:piano      # Classic, expressive
:blade      # Smooth lead
:pretty_bell  # Delicate, sparkly
:kalimba    # Percussive, bright
```

For Texture/Color:

```
:dsaw       # Granular textures
:noise      # White noise (percussion)
:bnoise     # Brown noise (rumble)
:gnoise     # Grey noise (hiss)
```

Essential Effects Settings

Reverb (most important!):

```
# Ambient: Large, wet
with_fx :reverb, room: 0.9, mix: 0.8, damp: 0.4
# Neoclassical: Medium, balanced
with_fx :reverb, room: 0.7, mix: 0.5, damp: 0.6
# Dry/close: Small, dry
with_fx :reverb, room: 0.3, mix: 0.2, damp: 0.8
```

Delay:

```
# Ambient: Long, washy
```

```
with_fx :echo, phase: 1.5, decay: 8, mix: 0.5

# Rhythmic: Tempo-synced

with_fx :echo, phase: 0.75, decay: 4, mix: 0.3

# Subtle: Short, quiet

with_fx :echo, phase: 0.25, decay: 2, mix: 0.15
```

Filters:

```
# Warmth: Low-pass

with_fx :lpf, cutoff: 85  # Remove highs

# Clarity: High-pass

with_fx :hpf, cutoff: 50  # Remove mud

# Evolving: Slow sweep

with_fx :lpf, cutoff: 60, cutoff_slide: 8 do |fx|

  control fx, cutoff: 110  # Slide to brighter

end
```

Common Tempos by Style

```
# Ambient

use_bpm 60   # Very slow, meditative

use_bpm 70   # Slow, spacious

use_bpm 80   # Medium-slow

# Neoclassical

use_bpm 80   # Contemplative
```

use_bpm 90   # Classic tempo (Nils Frahm often uses)

use_bpm 100 # Slightly upbeat

Uptempo neoclassical

use_bpm 120 # Rhythmic, energetic

Amplitude (Volume) Guidelines

Foundation/bass

amp: 0.5-0.7   # Solid presence

Pads/harmony

amp: 0.3-0.5   # Supportive, not dominant

Melody/lead

amp: 0.6-0.8   # Clear, audible

Texture/atmosphere

amp: 0.1-0.3   # Subtle, background

Sparkle/details

amp: 0.2-0.4   # Delicate, occasional

Release Times by Layer

Bass/drone

release: 8-16   # Very sustained

Pads

release: 4-8   # Sustained

```
# Melody

release: 1-3    # Clear but flowing

# Percussion/plucks

release: 0.1-0.5  # Short, articulate

# Textures

release: 2-6    # Medium, evolving
```

Time Signatures

```
# Standard

# 4/4 - most common (4 beats per bar)

# Interesting variations

# 3/4 - waltz time (elegant)

# 5/4 - asymmetrical (Take Five)

# 7/8 - very neoclassical (grouped as 3+2+2 or 2+2+3)

# To create 7/8 feel in Sonic Pi:

pattern = [3, 2, 2]  # Beat groupings

pattern.each do |beats|

  play :c4

  sleep beats * 0.25  # Each unit = 1/8 note

end
```

Polyrhythm Calculations

```
# 3 against 4 (common, beautiful)

# Pattern 1: sleep 1 (plays 4 times in 4 beats)

# Pattern 2: sleep 4.0/3 (plays 3 times in 4 beats)

# 5 against 4

# Pattern 1: sleep 1

# Pattern 2: sleep 4.0/5

# 7 against 4

# Pattern 1: sleep 1

# Pattern 2: sleep 4.0/7

# General formula:

# If you want X notes in Y beats:

# sleep Y.to_f / X
```

Performance Mode Shortcuts

```
# Quick variables for live control

$volume = 1.0

$complexity = 4

$section = :intro

# Use in loops

live_loop :melody do

  if $section != :intro
```

```
    play (scale :d4, :minor).take($complexity).choose,
      amp: 0.6 * $volume
  end
  sleep 1
end
# Change during performance:
# $volume = 0.5      # Quieter
# $complexity = 7    # More complex
# $section = :climax # Change section
```

Emergency "Save" Patterns

If things go wrong during performance:

```
# Drone reset - always sounds good
live_loop :save do
  use_synth :prophet
  with_fx :reverb, room: 0.9 do
    play :d2, release: 16, amp: 0.5
    sleep 16
  end
end
# Simple piano - always musical
```

```
live_loop :save_piano do
  use_synth :piano
  with_fx :reverb, room: 0.7 do
    play (scale :d3, :minor_pentatonic).choose,
      release: 2, amp: 0.6
    sleep [2, 3, 4].choose
  end
end
```

Mixing Tips (Volume Balance)

The Frequency Pyramid:

High (2kHz+): ▲ 20% of energy - delicate, sparkly

|

Mid (200-2kHz): ███ 40% of energy - body, richness

|

Low (30-200Hz): ████ 40% of energy - foundation, warmth

In practice:

- Bass/drone: amp 0.5-0.7
- Pads: amp 0.3-0.5
- Melody: amp 0.6-0.8
- Highs: amp 0.2-0.4

Sample BPMs of Famous Artists

- Brian Eno: 60-80 BPM

- Nils Frahm: 85-110 BPM
- Hania Rani: 80-100 BPM
- Max Richter: 70-90 BPM
- Ólafur Arnalds: 75-95 BPM

Key Signatures by Emotion

For Sadness/Melancholy:

- D minor, A minor, E minor
- F# minor (very emotional)

For Hope/Wistfulness:

- G major, D major, A major
- D Dorian, A Dorian

For Mystery/Darkness:

- C# minor, G# minor
- E Phrygian, A Phrygian

For Peace/Meditation:

- C major, F major
- Pentatonic scales (any root)

The Golden Rules (Print This!)

1. **Space > Density** - Less is more
2. **Bass > Treble** - Foundation first
3. **Evolution > Repetition** - Things must change
4. **Release > Attack** - Long decay = ambient
5. **Reverb > Delay** - Space before time
6. **Consonance > Dissonance** - Beautiful before clever
7. **Structure > Chaos** - Even randomness needs rules
8. **Emotion > Technique** - Feel before complexity
9. **Layers In/Out** - Build and release
10. **Listen > Code** - Ears are the final judge

Final Words

You now have everything you need to create beautiful ambient and neoclassical music with code. Remember:

- **Start simple** - A single drone can be profound
- **Trust your ears** - If it sounds good, it is good
- **Embrace mistakes** - Happy accidents are gifts
- **Practice daily** - Even 15 minutes compounds
- **Share your work** - Community feedback accelerates learning
- **Be patient** - Mastery takes time
- **Have fun** - Joy is the point

Your journey from December 12 to January 1 is just the beginning. The real learning happens in the months and years after, as you develop your unique voice.

Now go make some beautiful sound.

🎹✨🎵
