Interlude: Making

The story of this interlude is the story of a battle lost (or at best drawn) against technocentrism. This interlude was started with the intention of coming up with the simplest possible Python example in which a reader could explore what they like about their creation and try out automatic music generation. What it became was a code sample in which a beat is generated that is reasonably interesting, quite easy to explain, and endlessly fiddleable. Is it: a trap; a starting point; a generative worked example; or are we just justifying interesting code with rhetoric? (This is always a danger.)

For the last interlude in this book, we *do not* start with the importing of libraries, exactly, but rather with the installation of new software. In this case, sound libraries are required. Do not get too caught up in why this is not in Python by default: it often is, but this specific platform is not typically intended for music. For now, ignore the streams of nonsense-looking output that may result from software installation. These are noteworthy considerations around data use and reproduction but are beyond the scope of the lesson offered here.

```
!apt install fluidsynth # this is the library that the computer needs to generate sound
!git clone https://github.com/nwhitehead/pyfluidsynth # this connects it to Python
!python ./pyfluidsynth/setup.py install # this installs the downloaded python connnector
Reading package lists . . . Done
Building dependency tree
Reading state information . . . Done
< . . . a bunch of nonsense . . .>
Finished processing dependencies for pyFluidSynth==1.3.1
```

Now we get to import our libraries!

```
from IPython.display import Audio # this lets us play music in the cell
from pyfluidsynth import fluidsynth # this is the Python connector we installed above
import numpy as np # our data analysis library
from random import random, randint, choice # random numbers
```

Contained in the next box is the entirety of our music generation software. It is short! You can do this!

There are arguments in a couple directions here: this might seem daunting, confusing, or exclusionary; it could just as easily feel exploratory, freeing, or revelatory. Creative work can often be like that. Questions remain (as throughout the book): What might we expect you to know or not know? What might we expect a student to know or not know? How does code like this interplay with identity? To what extent is this likely to feel like it is off-putting and to what extent is this an entry to seeing that anyone can do creative work like this? Confidence—perhaps self-efficacy—is needed, but the musical knowledge required is almost nil. For example, you might look at the code below and decide that you do not know why E minor consists

of those numbers. I (Matthew) have no idea why E minor consists of those numbers, but that does not bother me; I know relatively little about music theory, so I just looked it up. Knowing that I can just look it up—and knowing that and knowing who I could ask for help—is more of the creative block here than "not knowing," but resources and power are inequitably distributed.

In short, make some music by fiddling with the following numbers. It really is fun, and it would be surprising if there was not some music made with some instrument accessible from this code was not pleasing to every single person who could read this. Who knows?

```
fl = fluidsynth.Synth()
fl.start()

# the arcane location just below is where the sound software put the
# "sound fonts" - that is, the libraries of instrument sounds by note
sfid = fl.sfload("/usr/share/sounds/sf2/FluidR3_GM.sf2")

# you can mess with this! change the key!

piano_e_minor_notes = [0, 64, 66, 67, 69, 71, 72, 74]

# this is just "blank" and "bass drum" but many other drums exist!
drum_bassdrum_notes = [0, 47]

# 4 on the flour. 16th notes
seqlen = 16

# track ("program") 0 is drums
fl.program_select(0, sfid, 128, 0)

# track 1 is piano
fl.program_select(1, sfid, 0, 0)

s = np.array([])
seconds = 0.07 # how long is a 16th note?

# each of these just selects from the lists above
# either silence or the notes we put in that list
drum_riff = [choice(drum_bassdrum_notes) for _ in range(seqlen)]
piano_riff = [choice(piano_e_minor_notes) for _ in range(seqlen)]
```

```
for i in range(seqlen * 4): # repeat the riff 4 times so it sounds riffy

    drum_note = drum_riff[i % len(drum_riff)] # play the right note
    # the "%" means "mod" - mod is "integer remainder" - like, 4% 3 = 1
    # we do that so we can go around and around a list of the right size
    if 0!= drum_note: # if not silence
        fl.noteon(0,drum_note,20)

    piano_note = piano_riff[i%len(piano_riff)] # same idea as the drums
    if 0!= piano_note:
        fl.noteon(1,piano_note,20)

    s = np.append(s, fl.get_samples(int(44100 * seconds)))
    # just tack the notes on the end

fl.delete() # ok, let's play!

framerate = 44100 # this is what our player expects, like a CD
Audio(s, rate=framerate, autoplay=True) # and let us hear the riff
```

The Left Hand of Data

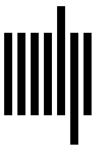
Designing Education Data for Justice

By: Matthew Berland, Antero Garcia

Citation:

The Left Hand of Data: Designing Education Data for Justice
By: Matthew Berland, Antero Garcia
DOI: 10.7551/mitpress/14381.001.0001
ISBN (electronic): 9780262377645
Publisher: The MIT Press
Published: 2024

The open access edition of this book was made possible by generous funding and support from MIT Press Direct to Open



The MIT Press

The MIT Press would like to thank the anonymous peer reviewers who provided comments on drafts of this book. The generous work of academic experts is essential for establishing the authority and quality of our publications. We acknowledge with gratitude the contributions of these otherwise uncredited readers.

This book was set in Stone Serif and Stone Sans by Westchester Publishing Services.

Library of Congress Cataloging-in-Publication Data

Names: Berland, Matthew, author. | Garcia, Antero, author.
Title: The left hand of data : designing education data for justice / Matthew Berland and Antero Garcia.
Description: Cambridge, Massachusetts : The MIT Press, [2024] | Includes bibliographical references and index.
Identifiers: LCCN 2023030088 (print) | LCCN 2023030089 (ebook) | ISBN 9780262547529 (paperback) | ISBN 9780262377652 (epub) | ISBN 9780262377645 (pdf)
Subjects: LCSH: Education—Data processing. | Education—Research. | Educational evaluation.
Classification: LCC LB1028.43 .B45 2024 (print) | LCC LB1028.43 (ebook) | DDC 370.285—dc23/eng/20230718
LC record available at https://lccn.loc.gov/2023030088
LC ebook record available at https://lccn.loc.gov/2023030089