Generación de música mediante cadenas de Markov

Benjamín Vera, Sebastián Toloza

Facultad de Ciencias Físicas y Matemáticas
Universidad de Chile
Departamento de Ingeniería Matemática
MA4402 Simulación Estocástica: Teoría y Laboratorio

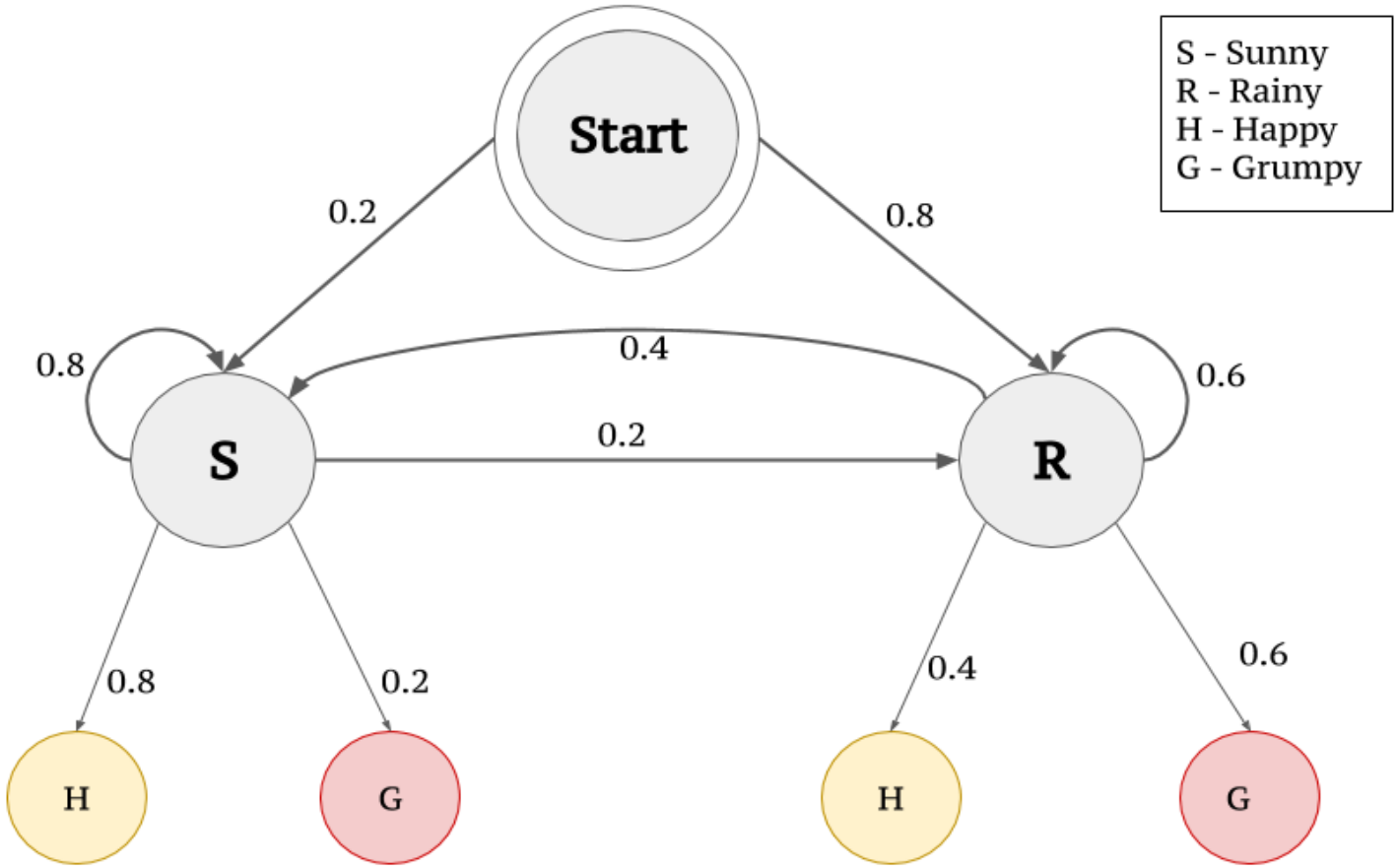
Introducción

La creación de música es una disciplina que a lo largo de su historia ha tenido varias interacciones con el azar, siendo este útil no solo para reemplazar el rol del compositor, sino también para darle herramientas más allá de la partitura convencional. Este proyecto busca explorar las opciones y limitaciones a la hora de generar música con modelos basados en cadenas de Markov.

En particular, se explora la generación de melodías mediante modelos de Markov ocultos (HMM).

Modelos ocultos de Markov (HMM)

Comencemos con un ejemplo ilustrativo.



Modelos ocultos de Markov (HMM)

Introduzcamos un poco de notación. Para un modelo de Markov simple (observable) homogéneo, tenemos las siguientes notaciones:

- $t \in \mathbb{N}$: Tiempo discreto.
- E: Conjunto de estados posibles. En este caso $E = \{0, \ldots, N-1\}$ finito.
- $X_t \in E$: Estado del sistema en el tiempo t.
- $\lambda_x = \mathbb{P}(X_0 = x)$: Probabilidad del estado inicial.
- $\lambda = (\lambda_x)_{x \in E}$: Distribución inicial de la cadena.
- $a_{x,y} = \mathbb{P}(X_1 = y | X_0 = x) \geq 0$: Probabilidad de transición de $x \to y$. Se satisface

$$\forall x \in E : \sum_{y \in E} a_{x,y} = 1$$

- $A = (a_{x,y})_{x,y \in E} \in \mathbb{R}^{N \times N}$: Matriz estocástica de transición.

Modelos ocultos de Markov (HMM)

Ahora, para un modelo de Markov oculto homogéneo, pensamos en los X_t como desconocidos y tenemos además los siguientes elementos:

- O: Conjunto de observaciones posibles. En este caso $O = \{0, \ldots, M-1\}$ finito.
- $V_t \in O$: Observación del sistema en el tiempo t, la cual depende solamente del estado X_t que la emite.
- $b_{x,z} = \mathbb{P}(V_t = z | X_t = x) \geq 0$: Probabilidad de ver la observación $z \in O$ dado que el estado oculto es $x \in E$. Se satisface

$$\forall x \in E : \sum_{z \in O} b_{x,z} = 1$$

- $B = (b_{x,z})_{x \in E, z \in O} \in \mathbb{R}^{N \times M}$: Matriz estocástica de observación.

Observar que la existencia de la matriz B implica que las probabilidades de observación son homogéneas en el tiempo.

Modelos ocultos de Markov (HMM)

Así, formalmente se define un HMM como una tripleta (A, B, λ) en donde:

- A denota a la matriz de transición de los estados ocultos
- B denota a la matriz de emisión de los estados observables a partir de un estado oculto
- λ denota a la distribución inicial de la cadena de Markov oculta

Simulación de un HMM

Este es el procedimiento estándar para la simulación de T pasos de un modelo oculto:

- Elegir x_0 según la distribución λ.
- Simular la cadena oculta $(X_t) \sim CM(\lambda, A)$ durante T pasos.
- Para cada uno de los estados x de la cadena, simular el observable que este estado emite con distribución dada por la fila x de la matriz B.
- Armar una lista con los observables obtenidos y retornarla como resultado.

Formulación del problema

Con el objetivo de generar secuencias de notas, interpretamos estas como los observables de un HMM con cierta cantidad de estados ocultos a definir. Lo que se busca es que el modelo pueda, a partir de ciertos datos de entrenamiento (melodías), generar una secuencia de notas que busque *imitar* su estilo.

Esto introduce entonces un problema de *aprendizaje*; es decir, se busca encontrar, dada una secuencia observada, los parámetros del HMM que mejor expliquen este comportamiento para a partir de ellos generar otra secuencia de notas de manera aleatoria.

Aprendizaje: Algoritmo de **Baum-Welch**

Como nos interesa el problema del entrenamiento, en donde, dada una secuencia de observaciones y el conjunto de los posibles estados en la cadena oculta, se tiene que determinar los parámetros A, B y λ del modelo, se usa el algoritmo de Baum-Welch, el cual para poder ejecutarse se necesita primero inicializar los parámetros (A, B, λ) de forma aleatoria, y obtener luego los parámetros:

1. Parámetro *forward*

$$\alpha_{x,t} = \mathbb{P}(V_0 = v_0, \ldots, V_t = v_t, X_t = x)$$

2. Parámetro *backward*

$$\beta_{x,t} = \mathbb{P}(V_{t+1} = v_{t+1}, \ldots, V_T = v_T \mid X_t = x)$$

3. Parámetro *gamma*

$$\gamma_{x,t} = \mathbb{P}(X_t = x \mid O)$$

Aprendizaje: Algoritmo de **Baum-Welch**

El parámetro *forward* se calcula según

1. Base:
$$\alpha_{x,0} = \mathbb{P}(V_0 = v_0, X_0 = x) = \lambda_x \cdot b_{x,v_0}$$
2. Inductivo:
$$\alpha_{x,t+1} = \sum_{y \in E} \alpha_{y,t} a_{y,x} b_{x,v_{t+1}}$$
3. Terminal:
$$\mathbb{P}(V_0 = v_0, \ldots, V_{t-1} = v_{t-1}) = \sum_{x \in E} \alpha_{x,t}$$

Aprendizaje: Algoritmo de **Baum-Welch**

El parámetro *backward* se calcula según

1. Base:
$$\beta_{x,T} = 1, \ \ \forall \ \ 0 \leq x \leq N-1$$
2. Inductivo:
$$\beta_{x,t} = \sum_{j=0}^{N-1} a_{xj} b_{x,v_{t+1}} \beta_{j,t+1}$$
3. Terminal:
$$P(O \mid (A, B, \lambda)) = \sum_{j=0}^{N-1} \lambda_j b_{j,v_0} \beta_{j,0}$$

Aprendizaje: Algoritmo de **Baum-Welch**

Y también necesitamos definir la matriz temporal de matrices

$$\xi_t(i,j) = \mathbb{P}(X_t = i, X_{t+1} = j \mid O) = \frac{\alpha_{i,t} a_{ij} b_{j,v_{t+1}} \beta_{j,t+1}}{\mathbb{P}(O \mid (A,B,\lambda))}$$

Tal que $\mathbb{P}(O \mid (A,B,\lambda)) = \sum_{j=0}^{N-1} \alpha_{j,t}\beta_{j,t}$, y se obtienen

$$\gamma_{i,t} = \sum_{j=1}^{N} \xi_t(i,j)$$

$$\sum_{t=1}^{T-1} \gamma_{i,t} = \text{Número esperado de transiciones desde } x_i$$

$$\sum_{t=1}^{T-1} \xi_{i,t} = \text{Número esperado de transiciones de } x_i \text{ a } x_j$$

Aprendizaje: Algoritmo de **Baum-Welch**

Por lo que nos queda determinar los nuevos parámetros $(\bar{A}, \bar{B}, \bar{\lambda})$.

En efecto, la distribución inicial se obtiene mediante

$$\bar{\lambda}_i = \gamma_{i,0} = \text{Frecuencia esperada en } x_i \text{ a tiempo } t = 0$$

Los coeficientes de $\bar{A}$ vienen dados Por

$$\bar{a}_{ij} = \frac{\text{Número esperado de transiciones de } x_i \text{ a } x_j}{\text{Número esperado de transiciones desde } x_i}$$

Es decir,

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_{i,t}}$$

Aprendizaje: Algoritmo de **Baum-Welch**

Y para $\bar{B}$, tenemos

$$\bar{b}_{j,k} = \frac{\text{Número esperado de veces en el estado } j \text{ y observando } v_k}{\text{Número esperado de veces en el estado } j}$$

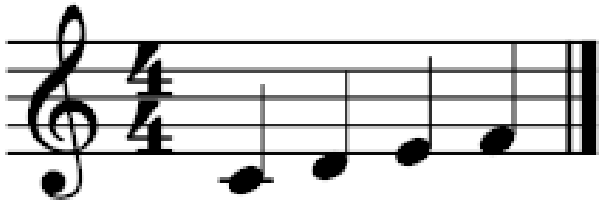
Es decir,

$$\bar{b}_{j,k} = \frac{\sum_{t=1}^{T} \gamma_{i,t} 1_{O_t = v_k}}{\sum_{t=1}^{T} \gamma_{i,t}}$$

Con lo cual de forma iterativa generamos nuevos parámetros (volviéndolos a usar en el modelo como input) hasta que estos converjan a un "punto fijo", es decir, un óptimo local.

Implementación

Se trabajó con el estándar `MIDI`. A través de la librería `mido` de `Python`.

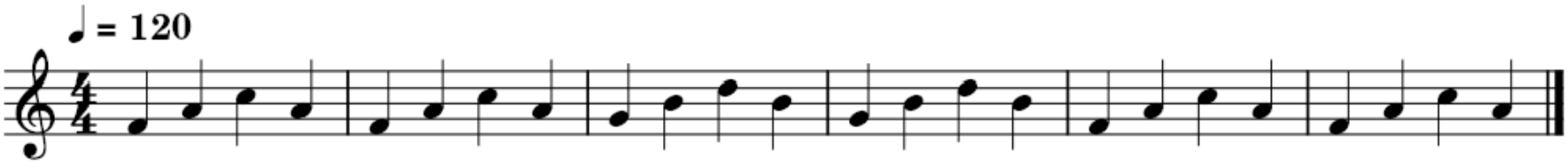


```
    MetaMessage('midi_port', port=0, time=0),
    Message('note_on', channel=0, note=60, velocity=80, time=0),
    Message('note_on', channel=0, note=60, velocity=0, time=455),
    Message('note_on', channel=0, note=62, velocity=80, time=25),
    Message('note_on', channel=0, note=62, velocity=0, time=455),
    Message('note_on', channel=0, note=64, velocity=80, time=25),
    Message('note_on', channel=0, note=64, velocity=0, time=455),
    Message('note_on', channel=0, note=65, velocity=80, time=25),
    Message('note_on', channel=0, note=65, velocity=0, time=455),
    MetaMessage('end_of_track', time=1)])
])
```

Resultados

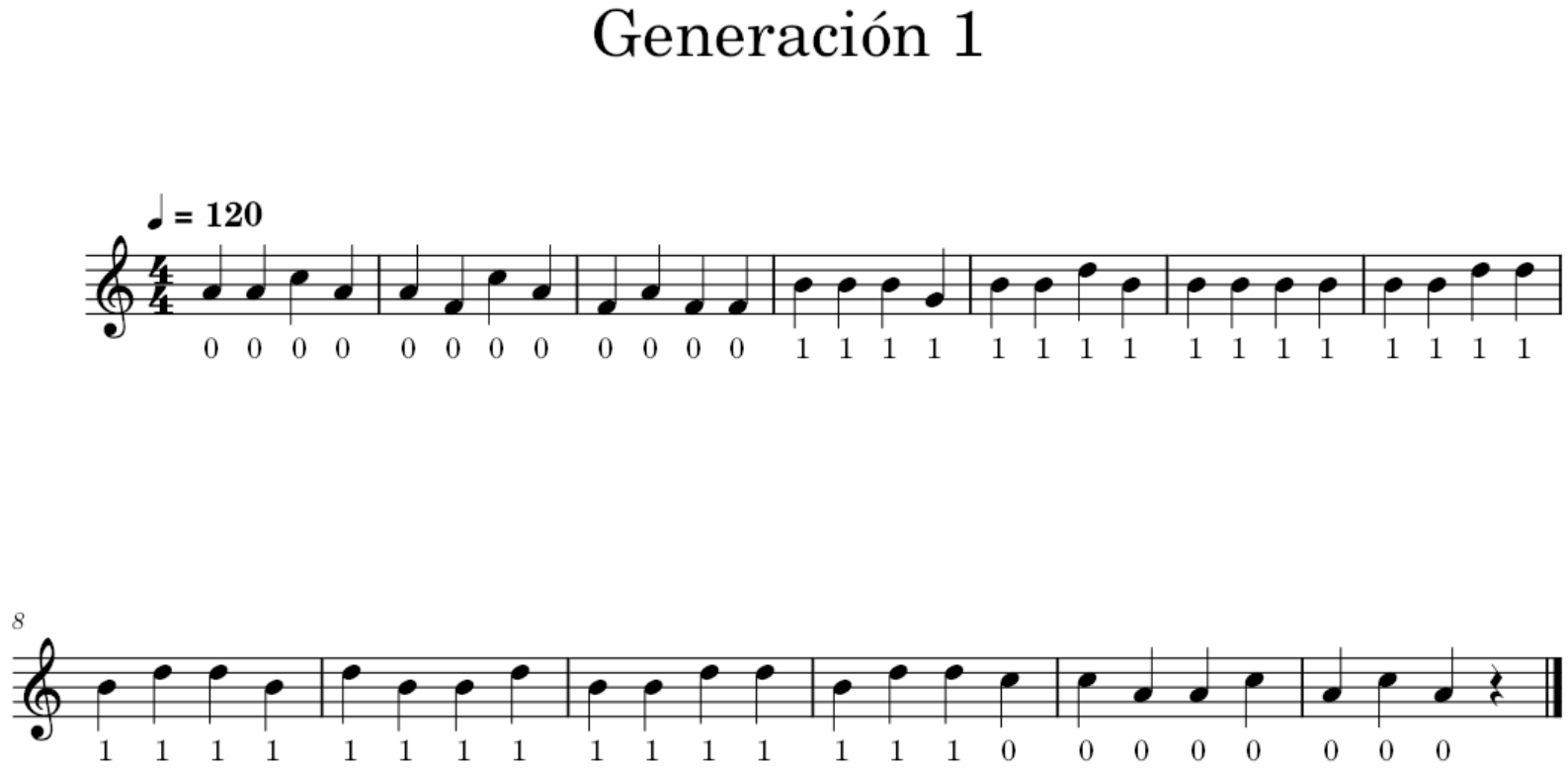
Un caso en que tratamos de forzar ciertos estados ocultos al modelo.

Entrenamiento



Resultados

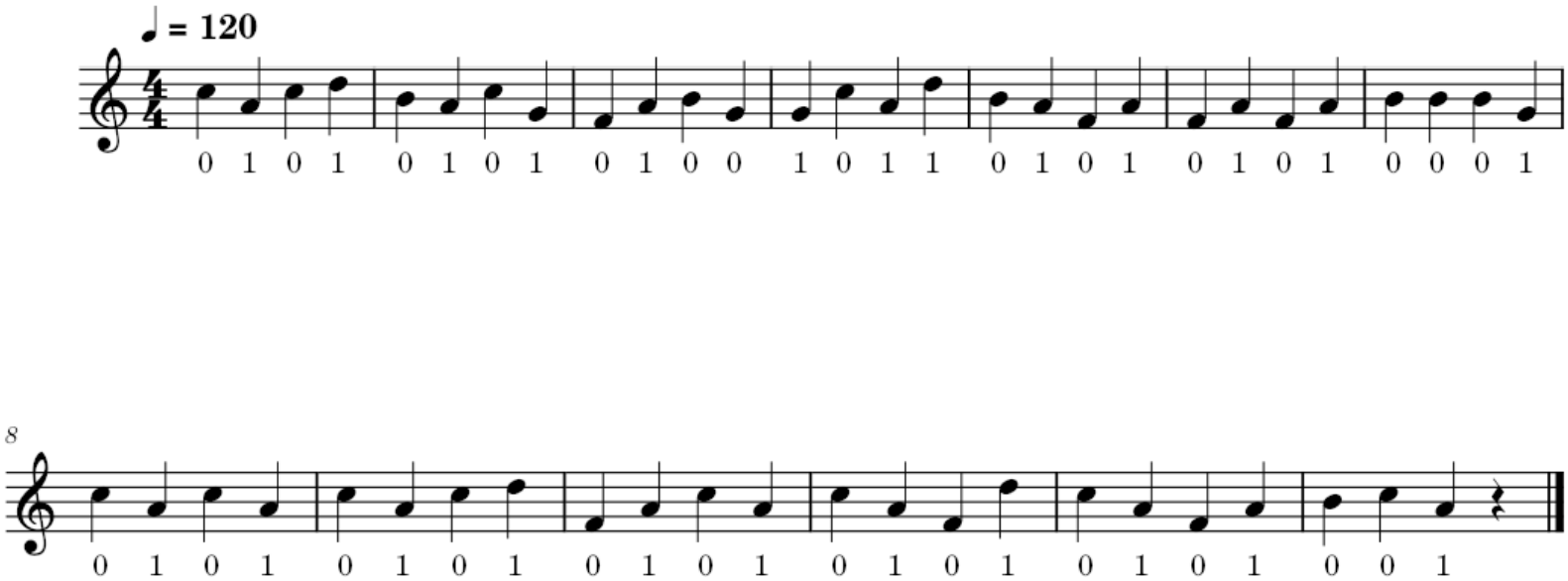
El modelo puede aprender lo que hubiésemos esperado.



Resultados

Pero esto depende del estado inicial del algoritmo de Baum-Welch.

Generación 2

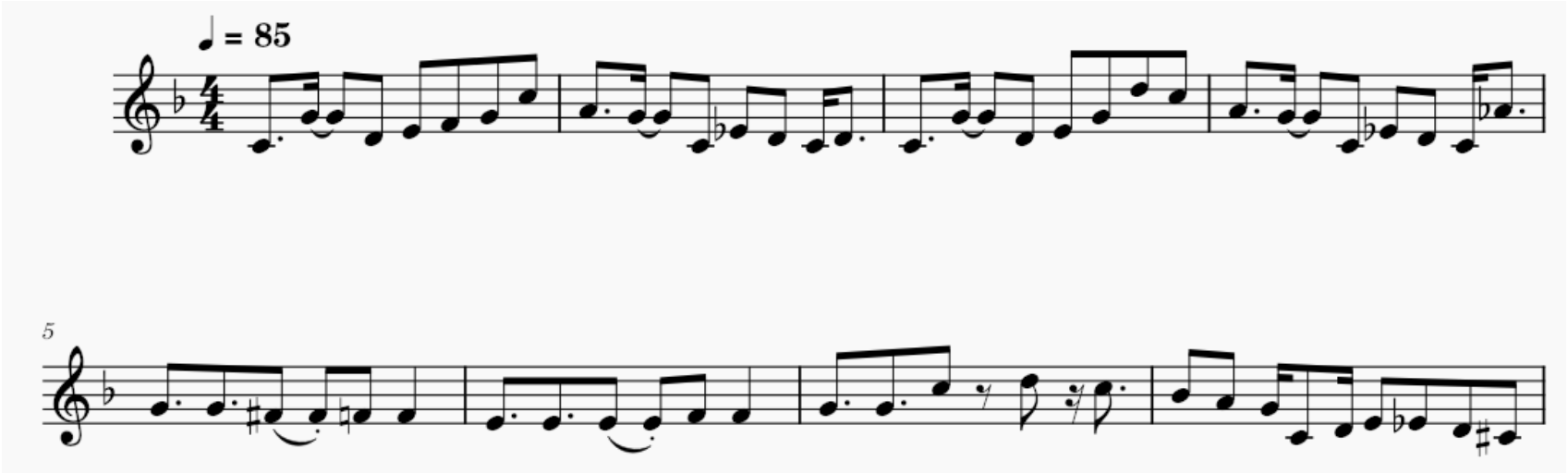


Resultados

Para casos más generales, se usó la siguiente data de entrenamiento:

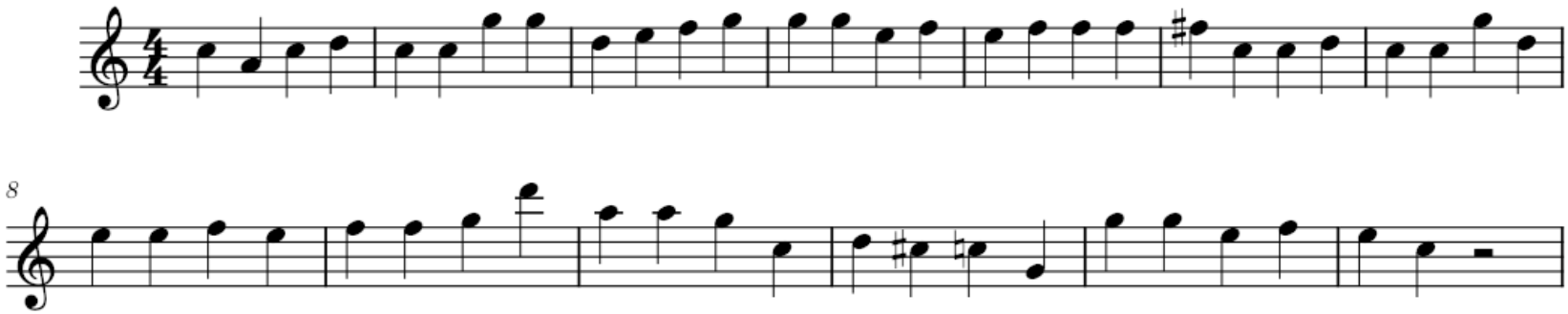
Resultados

Melodía de entrenamiento: *Oreburgh City*



Resultados

Melodía generada ($N = 15$ estados ocultos)



Pendientes

Realizado y obtenido lo anterior, existen muchas opciones de mejora del modelo, tales como:

1. Incorporación de ritmos
2. Considerar extensiones del modelo, por ejemplo, aumentando el número de estados, incorporando
   - Pares de notas en vez de notas singulares
   - Duraciones de las notas
   - Acordes

Conclusiones

- El modelo logra recuperar la información de sus datos de entrenamiento.
- Por la naturaleza de los modelos de Markov no se logra que las melodías generadas tengan una estructura global o de largo plazo.
- No fue posible incorporar ritmos al desarrollo del proyecto. Es un aspecto que las referencias vistas trabajan poco.

Referencias

1. Shapiro, I. & Huber, M., *Markov Chains for Music Generation*, *Journal of Humanistic Mathematics*, Volume 11 Issue 2 (July 2021), pages 167. DOI:10.5642/jhummath.2021.08. Disponible en https:\scholarship.claremont.edu/jhm/vol11/iss2/8

2. Yanchenko, A. K. (2017). Classical Music Composition Using Hidden Markov Models. *Duke University.*

3. Jurafsky, D. & Martin, J.H (2000). Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Prentice Hall PTR, USA.

Generación de música mediante cadenas de Markov

Benjamín Vera, Sebastián Toloza

Facultad de Ciencias Físicas y Matemáticas
Universidad de Chile
Departamento de Ingeniería Matemática
MA4402 Simulación Estocástica: Teoría y Laboratorio