


# A Connectionist Approach to Algorithmic Composition

Apresentação por Mila Soares de Oliveira de Souza (BSI – Unirio)



This paper presents a particular type of PDP network for music composition applications.

This paper is intended to provide an indication of the power and range of PDP methods for algorithmic composition and to encourage others to begin exploring this new approach.

# Introdução

## 1989

Até então....

### ▶ Composição Algorítmica

- Compositores determinavam regras de composição
- Computador entregava resultados conforme as regras

# Introdução

- ▶ NOVO PARADIGMA PARA COMPOSIÇÃO:

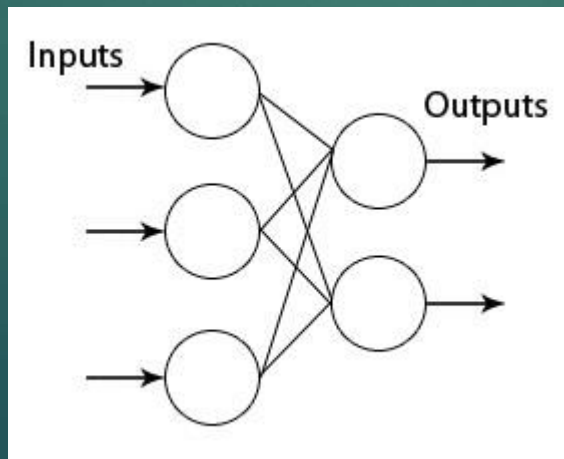
***parallel distributed processing*** (PDP)

*Computation is performed by a collection of several simple processing units connected in a network and acting in cooperation (Rumelhart and McClelland 1986).*

- ▶ Substitui comportamento de obediência de regras por **aprendizagem constante** e **generalização**.

# Conexionismo

- ▶ Cognição humana como o resultado de grandes redes de processamento interativo operando simultaneamente.
- ▶ aprendizagem, representação e processamento de informações são paralelas, distribuídas e interativas por natureza.




The PDP framework involved eight major aspects:

- ▶ A set of **processing units**, represented by a set of integers.
- ▶ An **activation** for each unit, represented by a vector of time-dependent functions.
- ▶ An **output function** for each unit, represented by a vector of functions on the activations.
- ▶ A **pattern of connectivity** among units, represented by a matrix of real numbers indicating connection strength.
- ▶ A **propagation rule** spreading the activations via the connections, represented by a function on the output of the units.
- ▶ An **activation rule** for combining inputs to a unit to determine its new activation, represented by a function on the current activation and propagation.
- ▶ A **learning rule** for modifying connections based on experience, represented by a change in the weights based on any number of variables.
- ▶ An **environment** that provides the system with experience, represented by sets of activation vectors for some subsets of the units.

# Nova abordagem de composição

A nova abordagem para composição algorítmica é:

1. Criar uma rede que pode aprender certos aspectos de estrutura musical
2. Fornecer à rede uma seleção de exemplos musicais, para que dela possa ser aprendidos aspectos estruturais
3. Permitir que a rede use o que aprendeu para construir novas músicas



O interesse é gerar padrões musicais razoáveis em novas situações.

O artigo analisa os seguintes aspectos:

- ▶ Network Design
- ▶ Melody Representation (Pitch, Duration)
- ▶ Training and Running the Neural Network
- ▶ Using the Network for Composition



# Network Design

- ▶ Since music is fundamentally a temporal process, the first consideration in designing a network to learn melodies is how to represent time
- ▶ 2 approaches: **window-time** & **sequential-time**

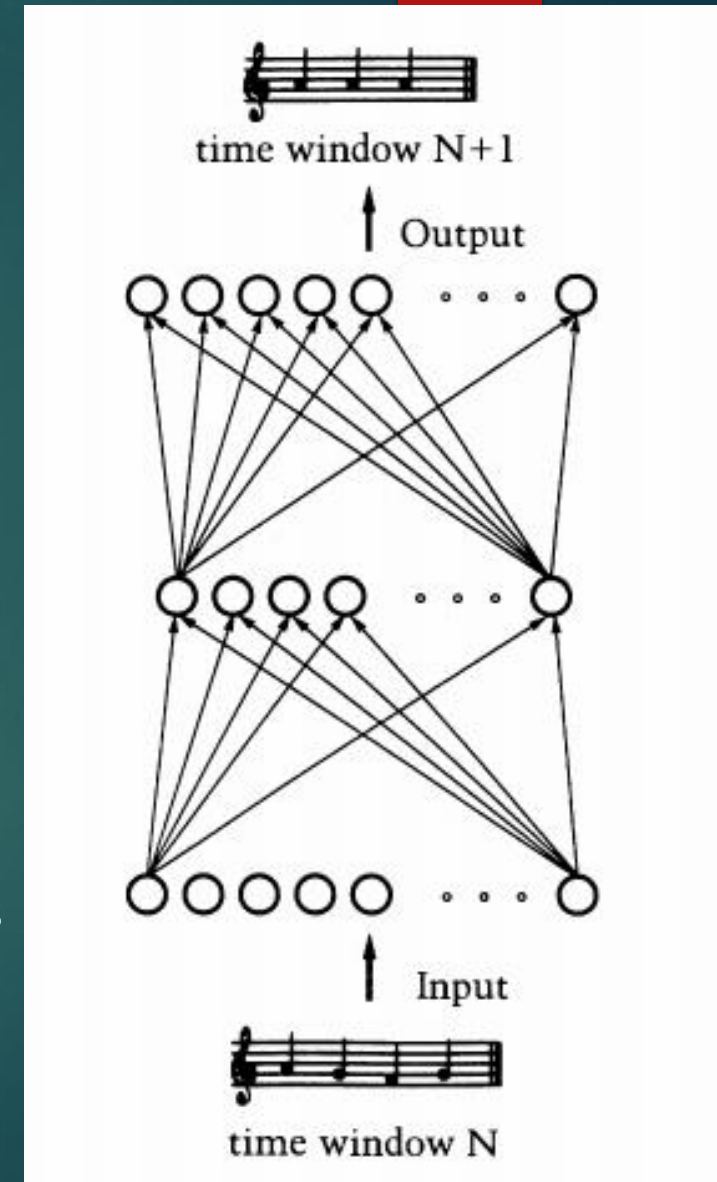
# Network Design

## ▶ Window-time

One way time may be represented is by standard musical notation translated into an ordered **spatial dimension**.

Time flowing from left to right, marked off at regular intervals by measure bars.

- ▶ In essence, the time-as-spatial-position representation converts the problem of learning music into the problem of **learning spatial patterns**.



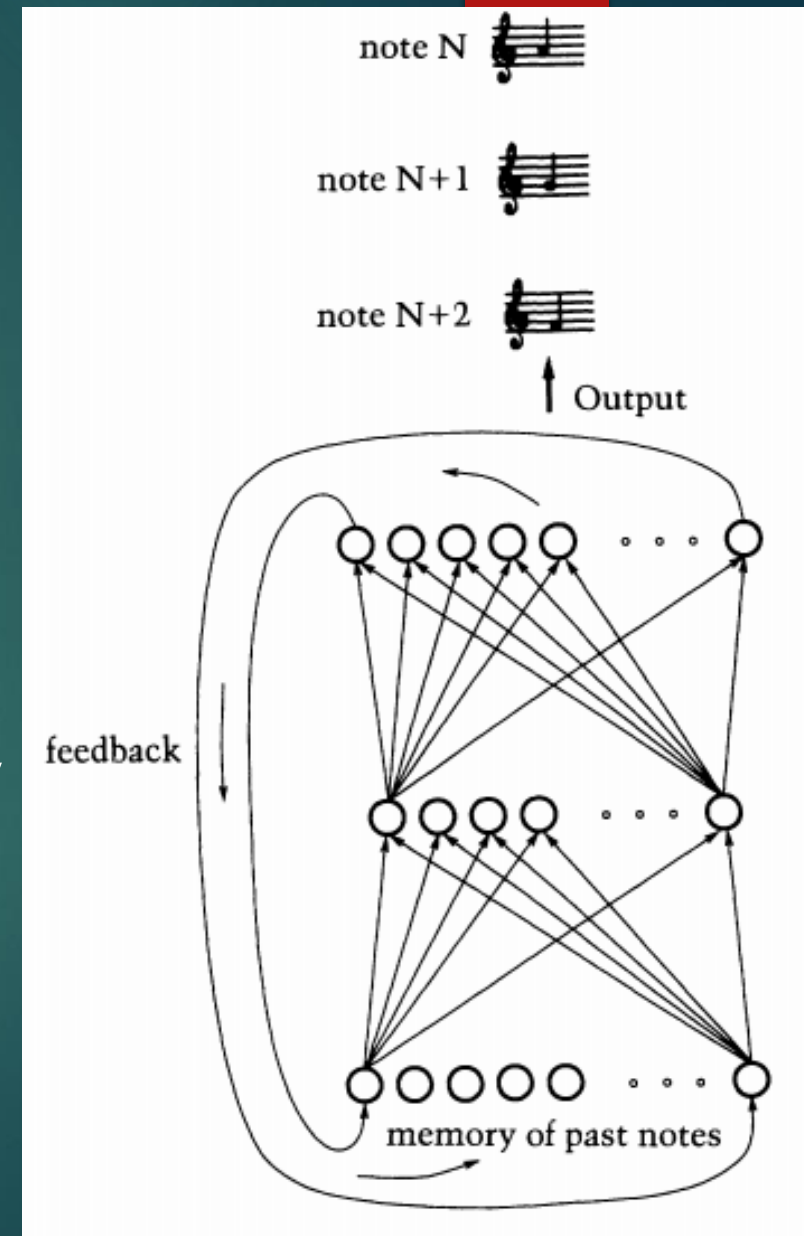
# Network Design


## ► Sequential-time

Treat music as a **sequential phenomenon**, with notes being produced one after another in succession.

Sequential network learns to produce a **sequence of single notes** based on some memory of past notes.

► Time is represented by the relative position of a note in the sequence, rather than the spatial position of a note in a window of units.



- 
- ▶ Basically, the windowed-time pattern associator network produces a **static output** given its input: one window of time in produces one window of time out.
  - ▶ The sequential network, on the other hand, **cycles repeatedly** to yield a sequence of successively produced outputs.

# Melody Representation

- ▶ The representation will determine the number and interpretation of the output units and of the corresponding context units.
- ▶ Our representation need only capture two types of information: the **pitch** and **duration** of each note.

# Melody Representation

## Pitch

- ▶ Either the **actual value** of each pitch can be specified, or the **relative transitions (intervals)** between successive pitches can be used.

**1. Actual value:** output units corresponding to each of the possible pitches

One unit for middle C, one unit for C#, etc.

- ▶ **A-B-C -> {A, B, C}**

**2. Intervals:** output units corresponding to pitch changes of various sizes.

One unit would designate a pitch change of + 1 half step, another a change of -3 half steps, etc.

**A-B-C -> {A, +2, + 1}**

(where the first pitch must be specified so we know where to start)

# Melody Representation

## Pitch

**Pitch-Intervals** approach is **appealing**:

- ▶ Not restricted to in the pitch range it can cover
- ▶ Melodies are encoded in a more-or-less key-independent fashion

*This key-independent learning is useful for letting the network discover common structure between melodies.*

*C-G-E-F and F#-C#-A#-B -> network would be likely to miss any connection between the two, since they use totally different output units.  
In a pitch-interval representation, though, these two phrases would be coded identically.*

- ▶ Allows transposition of entire melody just by changing the initial pitch.

# Melody Representation

## Pitch

**Pitch-Intervals** approach has a **major drawback**:

- ▶ When an error is made in the production of a sequence of intervals, the rest of the melody will be transposed to a different key, relative to the segment before the error
- ▶ One error in the interval sequence will thus globally affect the performance of the whole melody

In contrast, mistakes in the output using the actual-pitch representation are purely local. Only the pitch of the individual wrong note will be altered.



# Melody Representation

## Duration

Time-slice representation:

- ▶ melody could be divided into equally spaced time slices of some fixed length
- ▶ each output in the sequence would correspond to the pitch during one time slice.

A-B-C

quarter-note, eighth-note, and dotted quarternote -> time slices of eighth-note duration =

{A, A, B, C, C, C}

# Melody Representation

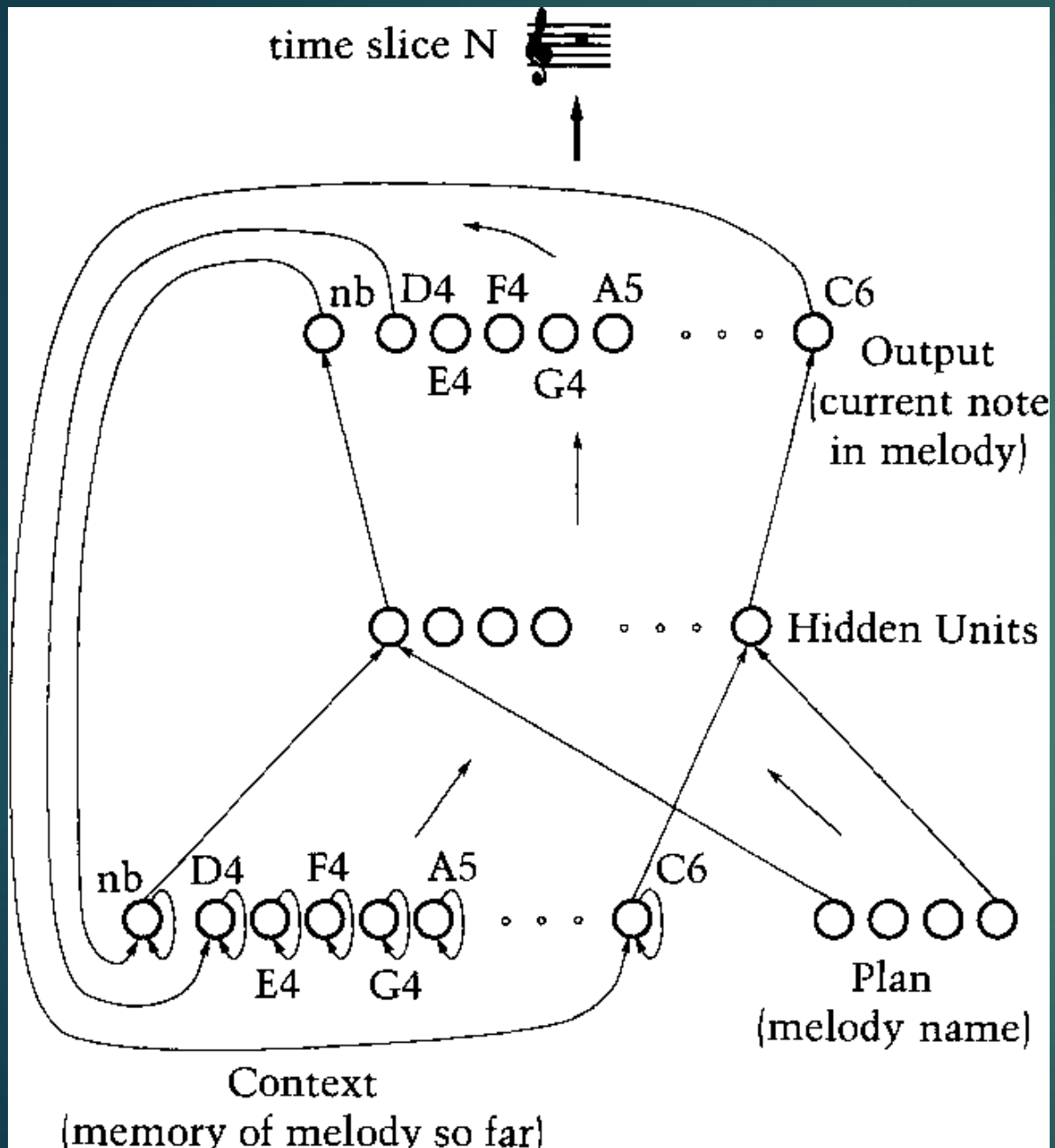
## Duration

Time-slice representation:

- ▶ Need an indication of the time slices on which any notes begin.

*{ A, A } = two notes of pitch A, each lasting one time slice? or one note of pitch A that is two time slices in duration?*

- ▶ Therefore, an additional **note-begin marking** unit is added to the output (and context).
- ▶ The presence of the note-begin unit also **complicates attempts at representing polyphonic sequences.**



- ▶ The sequential network design used for compositional purposes in this paper.

The current musical representation requires note-begin (nb) and pitch (D4-C6) units, as shown.

# Training and Running the Network

The length of the training process depends on:

- ▶ the size of the network
- ▶ the number of training melodies
- ▶ their relatedness to each other
- ▶ and (at least) two network training parameters
  - ▶ learning rate
  - ▶ Momentum

Low learning rates (on the order of 0.05) and high momenta (around 0.8) seem appropriate

# Training and Running the Network

To give approximate figures on the number of epochs:

**5 output units** and **15 hidden units** learned:

- ▶ One short melody (34 time steps, as in the examples to come) in **2,000 epochs**.
- ▶ For two such melodies, 5,500 epochs were required
- ▶ For four, 8,500 epochs were needed

When **8 hidden units** were used, it took nearly **50,000 epochs** to learn the same two melodies that took 5,500 epochs in the previous case.

# Using the Network for Composition

**15 output units** - **1 note-begin unit** and **14 pitch units** corresponding to the pitches in the key of C from D4 to C6, with no accidentals.

**5 context units**, all with self-feedback connections of strength 0.7, and 15 hidden units.

All of the original trained melodies used were **34 time slices in length**, padded at the end with rests (as shown in the music notation).

Each **time slice** corresponds to an **eighth-note** in duration, so that the original melodies are all approximately four measures long.

# Using the Network for Composition

**15 output units** - **1 note-begin unit** and **14 pitch units** corresponding to the pitches in the key of C from D4 to C6, with no accidentals.

**5 context units**, all with self-feedback connections of strength 0.7, and 15 hidden units.

All of the original trained melodies used were **34 time slices in length**, padded at the end with rests (as shown in the music notation).

Each **time slice** corresponds to an **eighth-note** in duration, so that the original melodies are all approximately four measures long.

# Using the Network for Composition

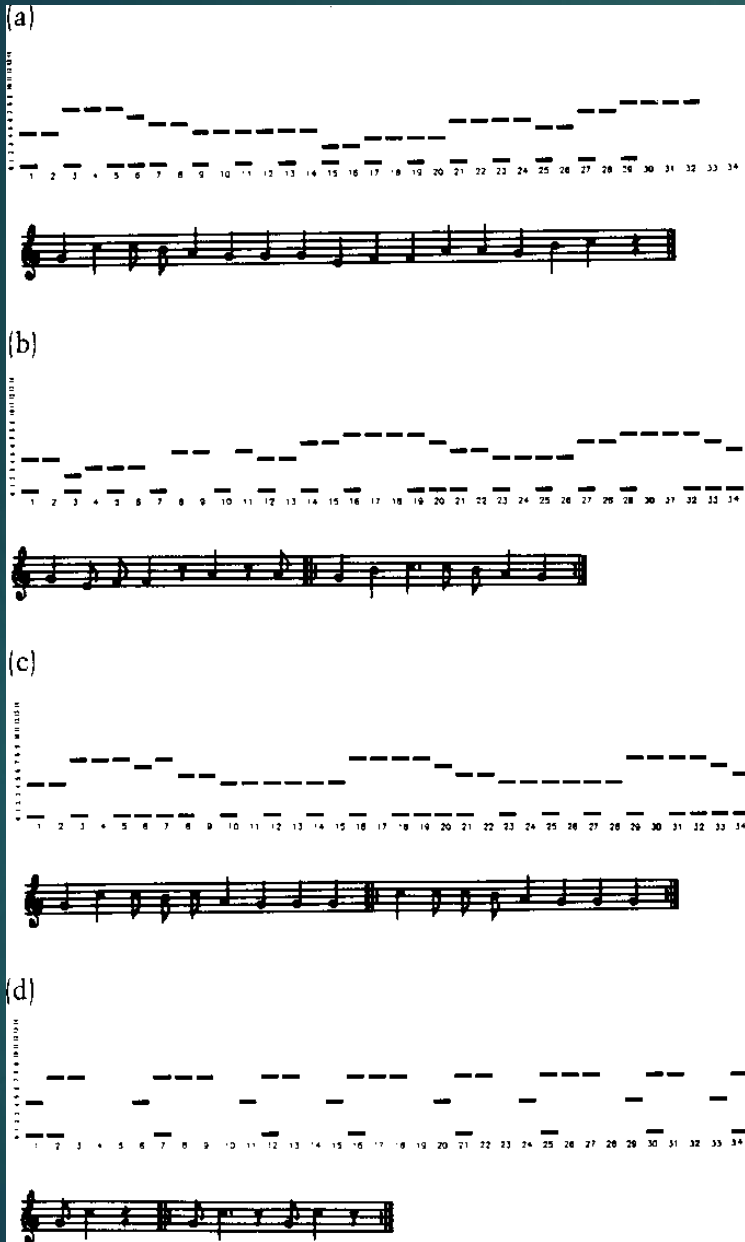
- ▶ **Extrapolating from a Single Plan**

If our network is trained on only one melody, then we have a single plan from which to extrapolate and choose new plans.

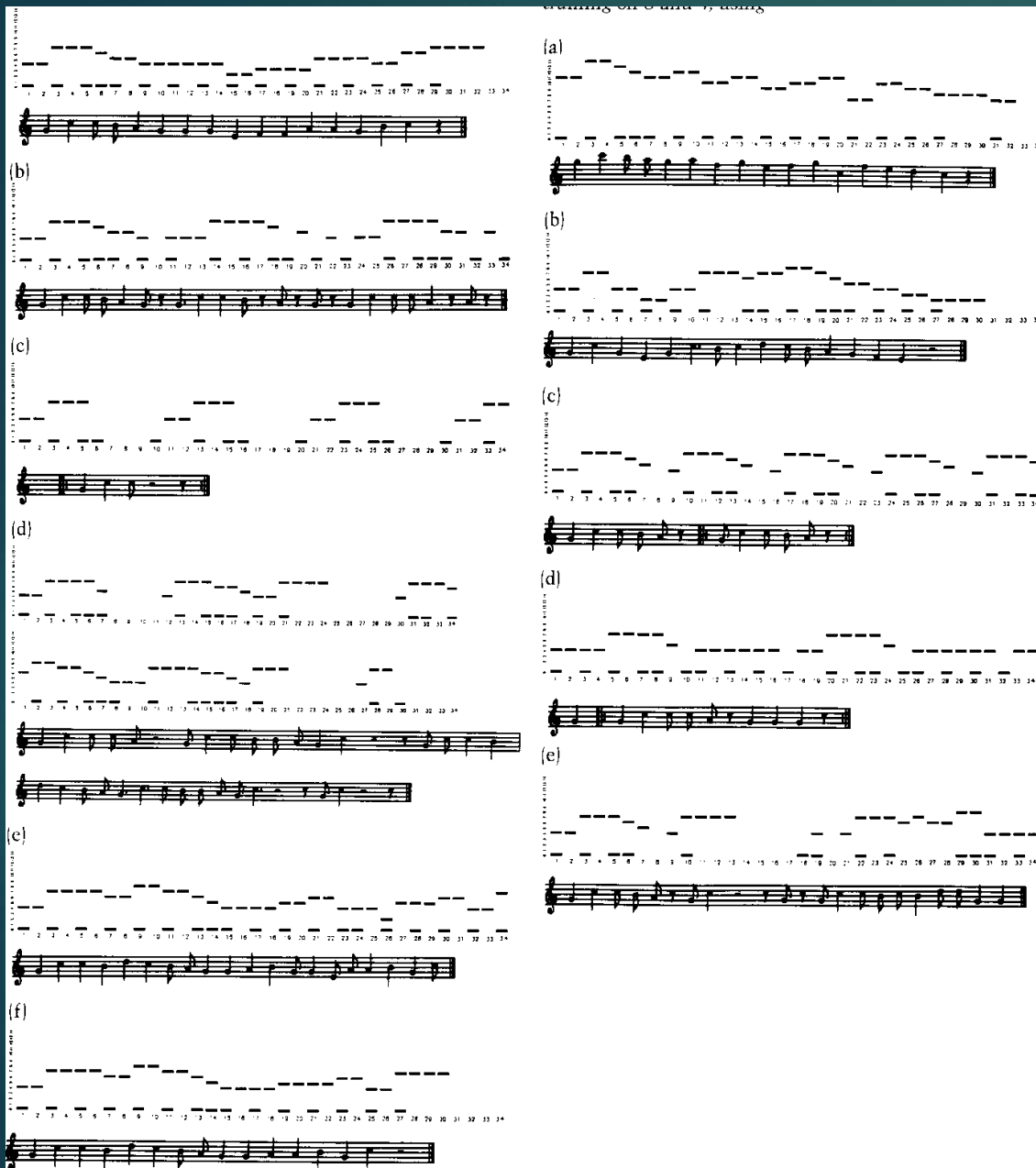
- ▶ **Interpolating Between Multiple Plans**

If we train our original network with more than one melody, each having a different plan, then we can generate new melodies by specifying new plans interpolated between the trained ones.





► Network output using extrapolation from a single melody.



▶ Network output using interpolation between two melodies.

# Referências

- ▶ M. TODD, Peter. **A Connectionist Approach to Algorithmic Composition**. Disponível em:  
<https://www.semanticscholar.org/paper/A-Connectionist-Approach-to-Algorithmic-Composition-Todd/81e0a57abde1bf2cc4b7cd772e0573e92069e8ef>. Acesso em:  
8 de nov. 2018