

Informática para a Musicologia (IPM) 2024/25

Jupyter Notebooks

Docente: [J.N. Oliveira](#)

Departamento de Informática da U. Minho, em colaboração com a  ENSICO

Aula de 15-Out: Experiência com bibliotecas Haskell para *Computer-Aided Musicology*

Manipulação de sequências que representam música. Interação com notação Abc. Compassos e polifonia.

Representação formal da polifonia. O combinador `P`. Exemplo: Cânone nr.1 dos *Canones diversi super thema regium* do BWV 1079 de J.S. Bach (1685-1750). Movimento reverso (função 'reverse').

Imitação e repetição.

⚠ Importante: correr sem mexer as células seguintes.

```
In [ ]: :opt no-lint
:m Data.Ratio
:m Data.Char
:m Data.List
```

```
In [ ]: :l ../src/Cp.hs
:l ../src/Reducer.hs
:l ../src/Ipm.hs
:l ../src/Abc.hs
```

```
In [ ]: :l ../src/CS.hs
```

Compassos

Voltemos a:

```
In [ ]: m = ["A", "^G", "A", "B", "c", "B"]
r = [3%4, 3%4, 3%4, 3%4, 3%4, 3%4, 3%4]
mel = zip m r
----
abcplease mel
```

4.1 - A visualização seguinte acrescenta barras de compasso ternário, cf. $3/4$:

```
In [ ]: abcPlayM "C" "3/4" mel
```

```
In [ ]: abcPlayM "A" "3/4" mel
```

👉 Sobre abcplease e abcPlayM

Designação	Significado	Descrição detalhada
abcplease	mostrar partitura	abcplease m mostra a partitura de m sem armação de clave nem compasso, nem barras de compasso
abcPlayM	mostrar partitura	abcPlay K C m - mostra a partitura de m com armação de clave K, compasso C e barras de compasso (regular) deduzidas a partir de C

4.2 - Fazer a mesma coisa para `carnaval_serrano`, usando compasso binário ($2/4$) e quaternário ($4/4$):

```
In [ ]: abcPlayM undefined
```

```
In [ ]: abcPlayM undefined
```

Polifonia

 Dadas duas linhas melódicas m e n , $P [m,n]$ representa a sua execução simultânea, ie. *polifónica*.

A notação estende a mais do que duas vozes, por exemplo $P [m,n,p]$ etc.

Por vezes usamos a abreviatura $m \# n$ quando há só duas vozes, cf:

```
In [ ]: m # n = P [m,n]
```

4.3 - Executar o seguinte exemplo de polifonia a duas vozes:

```
In [ ]: soprano = mel
bass      = [("A",3 % 4),("E",3 % 4),("A",3 % 4),("G",3 % 4),("C",3 % 4),("G",3 % 4)]
---
abcPlayM "C" "3/4" (soprano # bass)
```

Acrescentar uma parte de tenor, também em mínimas com ponto (editar):

```
In [ ]: tenor = zip ["C","E","C","D","E","D"] [3 % 4,3 % 4,3 % 4,3 % 4,3 % 4,3 % 4]
--- etc
```

4.4 - Como anteriormente, para ouvir o ABC gerado correr `abcShow` (célula seguinte) e copiar para o *clip-board*; de seguida entrar no [editor ABC on-line](#), colar e pôr a tocar.

```
In [ ]: abcShow
```

Caso de estudo

Cânone nr.1 dos *Canones diversi super thema regium* do **BWV 1079** de J.S. Bach (1685-1750):

Analogia:



Canones diversi super thema regium.

Canon a 2.

1. 

Primeira parte do cânone - o *tema régio* (a):

In []:

```
a = royalTheme
----
abcPlayM "Eb" "C" a
```

In []: abcShow

Segunda parte do cânone - o contratema (b):

```
In [ ]: b = counterTheme
      ---
      abcPlayM "Eb" "C" b
```

```
In [ ]: abcPlayM "Eb" "C" (reverse (a++b))
```

4.5 - Construir o cânone todo, a duas partes, sabendo que código (a) ++ (b) está em contraponto consigo própria, mas em movimento retrógrado. Mostrá-lo em partitura.

In []:

NB: tal como acima, para ouvir o ABC gerado correr `abcShow`, copiar código `abc` que é mostrado para o *clip-board* e entrar no [editor ABC on-line](#) para o pôr a tocar.

4.6 - De volta a `mel` acima, construir na célula seguinte `contra` tal que

```
| (mel ++ contra) # reverse (mel ++ contra)
```

faça sentido harmónico.

Visualizar o resultado e ouvi-o usando `abcShow` (etc).

In []:

```
In [ ]: abcShow
```

4.7 - A célula seguinte contém as 5 partes da *Earle of Essex Galiard* de [John Dowland](#) (1563-1626) tal como foi trabalhada anteriormente em ABC. Colocá-la em partitura, ouvi-la e experimentar acrescentar mais notas às partes mais incompletas, a partir deste [PDF](#).

```
In [ ]: cantus = [("G",3 % 8),("A",1 % 8),("B",1 % 4),("G",1 % 4),("g",1 % 2),("f",3 % 4),("_e",1 % 4),("d",1 % 2),
bassus  = [("G",1 % 2),("G",3 % 4),("A",1 % 4),("B",3 % 2),("B",1 % 2),("F",1 % 2),("G",1 % 2),("D",
quintus = [("B",3 % 4),("A",1 % 4),("G",1 % 2),("d",3 % 8),("e",1 % 8),("f",1 % 2),("B",3 % 4),("B",
tenor   = [("d",1 % 2),("g",1 % 2),("d",3 % 4),("c",1 % 4),("B",1 % 2)]
altus   = [("G",1 % 2),("G",3 % 8),("A",1 % 8),("B",1 % 4),("c",1 % 4),("d",3 % 4),("c",1 % 4),("B",1 % 2),
```

```
In [ ]:
```

Repetição e imitação

4.8 - Conseguem identificar que melodia será a seguinte, se em fá maior?

```
In [ ]: fj = [("F",1 % 4), ("G",1 % 4), ("A",1 % 4),("F",1 % 4),("F",1 % 4),
("G",1 % 4), ("A",1 % 4), ("F",1 % 4),("A",1 % 4),("B",1 % 4),
("c",1 % 2), ("A",1 % 4), ("B",1 % 4),("c",1 % 2),("c",3 % 16),
("d",1 % 16),("c",1 % 8), ("B",1 % 8),("A",1 % 4),("F",1 % 4),
("c",3 % 16),("d",1 % 16),("c",1 % 8),("B",1 % 8),("A",1 % 4),
("F",1 % 4), ("F",1 % 4), ("C",1 % 4),("F",1 % 2),("F",1 % 4),
("C",1 % 4), ("F",1 % 2)]
```

Façam-na tocar na próxima célula (não se esqueçam: em fá maior)

```
In [ ]: abcPlayM "F" "C" fj
abcShow
```

4.9 - A operação `delay t m` atrasa a linha melódica `m` `t`-unidades de tempo. Por exemplo:

```
In [ ]: (abcPlayM "F" "3/4" . P) [ mel , delay (2%4) mel ]
```

Usando `delay`, construir o cânone a 4 vozes (uníssono) sobre `fj` de forma a que a 4ª voz só execute a melodia uma vez e as outras vozes nunca parem de cantar:

In []:

```
v1= undefined
v2= undefined
v3= undefined
v4= undefined
canone = (P [v1,v2,v3,v4])
-----
abcPlayM "F" "C" canone
```

In []:

```
abcShow
```

4.10 - A operação $a \otimes p$ repete p a a -vezes, por exemplo $2 \otimes "ab" = "abab"$. Olhando para

Fre - re Jac - ques, Fre - re Jac - ques, dor - mez vous?
 Dor - mez vous? Son - nez les ma - ti - nes!
 Son - nez les ma - ti - nes! Din - dan - don. Din - dan - don.

determinar i e os padrões p_0 a p_3 por forma a se ter a igualdade:

In []:

```
i = undefined
n = i ⊗ p0 ++ i ⊗ p1 ++ i ⊗ p2 ++ i ⊗ p3
-----
n==m
```

