Fonology

um pacote em R para análises fonológicas

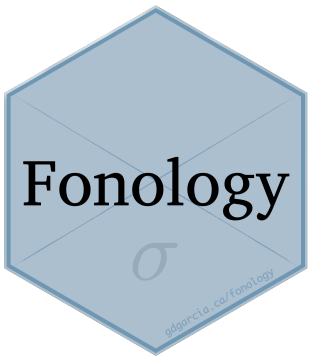
InterAb 14

Guilherme D. Garcia (Université Laval • CRBLM)

1 Introdução

Duas ferramentas complementares para fonologia

☞ Transcrição e análise de dados em R



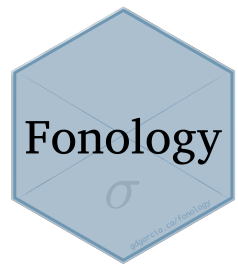
☞ Produção de documentos em Typst

phonokit

2 Fonology

O que é

- Um pacote em R para fonologia (Garcia, 2026a)
- Ideia: automatizar codificação e processamento de dados fonológicos
- Componentes **analítico** e **didático**. Versão atual: `1.0.0`



Instalação etc.

- Visite gdgarcia.ca/fonology para informações e demos detalhadas
- Para instalar o pacote, é necessário ter o pacote `devtools`:

```
library(devtools) # install.packages("devtools")
install_github("guilhermegarcia/fonology")
```

Premissa

- Familiaridade básica com R e com a família `tidyverse`
- `Fonology` : internamente, apenas `▷` ; externamente, ambos os pipes podem ser usados

Transcrição fonêmica

- Disponível para espanhol, francês, italiano e português (acurácia >80%)

```
library(Fonology)
ipa(word = c("Exemplo", "com", "múltiplas", "palavras"))
```

```
           1            2              3             4
"e.ˈzem.plo"       "ˈkom" "ˈmul.ti.plas"  "pa.ˈla.vras"
```

Transcrição de textos

- `ipa()` exige um input tokenizado. E se nosso input for um texto...?
- `cleanText()`: limpeza e tokenização

```
library(tidyverse)
texto = "Este teXto 123# bastante cUrto Não está tokenizado"

texto ▷
  cleanText() ▷
  ipa()
```

```
          1           2           3           4
"ˈes.te"    "ˈtes.to" "bas.ˈtan.te"    "ˈkur.to"
          5           6           7
   "ˈnãw̃"    "es.ˈta" "to.ke.ni.ˈza.do"
```

Transcrição para tabelas

- Normalmente, análises começam com *data frames* ou *tibbles*

```
texto = "Este teXto 123# bastante cUrto Não está tokenizado"

tabela = tibble(palavra = texto ▷ cleanText()) ▷ # Coluna com palavras
  mutate(ipa = palavra ▷ ipa()) # Coluna com transcrição
```

```
# A tibble: 4 × 2
  palavra  ipa
  <chr>    <chr>
1 este     'es.te
2 texto    'tes.to
3 bastante bas.'tan.te
4 curto    'kur.to
```

Textos reais: Os Lusíadas

Tarefa

1. Importar *Os Lusíadas*, limpar e tokenizar o texto
2. Transcrever, silabificar, acentuar palavras lexicais
3. Extrair acento e última sílaba
4. Extrair constituintes da última sílaba de cada palavra

Ferramentas/funções

- `getStress()` : extração de acento a partir de transcrição
- `getWeight()` : extração de perfil de peso (e.g., `LLH`)
- `getSyl()` : extração de uma sílaba
- `syllable()` : extração de constituintes silábicos
- `stopwords_pt` e `stopwords_sp` : listas de palavras funcionais

Textos reais: Os Lusíadas

```
corpus ← lusiadas ▷
  cleanText() ▷ # Limpamos e tokenizamos o texto
  as_tibble() ▷
  rename(word = value) ▷
  filter(
    !word %in% stopwords_pt,
    !is.na(word)
  ) ▷ # Removemos palavras funcionais
  mutate(
    ipa = ipa(word), # Criamos uma coluna para transcrição
    stress = getStress(ipa), # outra para o acento
    weight = getWeight(ipa), # outra para o peso
    finSyl = getSyl(word = ipa, pos = 1), # outra para a sílaba final
    onsetFin = syllable(finSyl, const = "onset"),
    nucFin = syllable(finSyl, const = "nucleus"),
    codaFin = syllable(finSyl, const = "coda"),
    rimaFin = syllable(finSyl, const = "rhyme")
  )
```

Resultado

- Um tibble 32.618 x 9 inteiramente codificado

word	ipa	stress	weight	finSyl	onsetFin	nucFin	codaFin	rimaFin
armas	ˈar.mas	penult	HH	mas	m	a	s	as
barões	ba.ˈrõj̃s	final	LH	rõj̃s	r	õj̃	s	õj̃s
assinalados	a.si.na.ˈla.dos	penult	LLH	dos	d	o	s	os
ocidental	o.si.den.ˈtal	final	LHH	tal	t	a	l	al
praia	ˈpra.ja	penult	LL	ja	NA	ja	NA	ja
lusitana	lu.zi.ˈta.na	penult	LLL	na	n	a	NA	a
mares	ˈma.res	penult	LH	res	r	e	s	es

MaxEnt

- Estimar pesos de restrições em uma gramática de entropia máxima (Goldwater & Johnson, 2003)
- Suponha o tableau abaixo (formato tibble): objeto `tableau_maxent_1`

```
# A tibble: 3 × 6
  input output  NoCOMPLEX   DEP   MAX   obs
  <chr> <chr>       <dbl> <dbl> <dbl> <dbl>
1 /CCV/ [CCV]           1     0     0     5
2 /CCV/ [CV.CV]         0     1     0    20
3 /CCV/ [CV]            0     0     1    75
```

☞ **Quais os pesos para maximizarmos os valores observados?**

MaxEnt

```
maxent(tableau_maxent_1)$weights
```

```
nocomplex       dep       max
3.5059833 2.1196889 0.7979331
```

MaxEnt

```
maxent(tableau_maxent_1)$predictions
```

```
# A tibble: 3 × 13
  input output  nocomplex   dep   max   obs harmony max_h exp_h     Z obs_prob
  <chr> <chr>       <dbl> <dbl> <dbl> <dbl>   <dbl> <dbl> <dbl> <dbl>    <dbl>
1 /CCV/ [CCV]           1     0     0     5   3.51   3.51  1     20.0     0.05
2 /CCV/ [CV.CV]         0     1     0    20   2.12   3.51  4.00  20.0     0.2
3 /CCV/ [CV]            0     0     1    75   0.798  3.51 15.0   20.0     0.75
# ℹ 2 more variables: pred_prob <dbl>, error <dbl>
```

3 phono*kit*

O que é

- Um pacote em Typst para fonologia (Garcia, 2026b)
- Ideia: gerar representações fonológicas (IPA, prosódia, autossegmental, etc.)
- Versão atual: `0.4.5` (março 2026)

phonokit

Typst...?

Typst é uma nova linguagem, mais rápida, leve, amigável e moderna do que LaTeX. Visite typst.app para começar com um editor online (consulte tutoriais)

Produção de documentos

- Uma análise termina na comunicação dos resultados: pôster, slide, artigo, livro
- Representações fonológicas podem ser trabalhosas (`tex` , `doc` , `ppt`)
- **phonokit** (Garcia, 2026b) é um pacote para **Typst** que tem três objetivos:
  - criar estruturas fonológicas precisas e de alta qualidade gráfica
  - ter uma sintaxe mínima e intuitiva
  - utilizar uma linguagem moderna e rápida

Fonology → **phonokit**

Duas ferramentas complementares: análise & representação

Transcrição fonética

☞ O pacote utiliza Charis SIL como fonte padrão, mas é possível alterá-la (e.g., Libertinus Sans)

```
#ipa("[tR \\~ a Ns.kRi.'s \\~ a \\~ w]")
```

[tʀãŋs.kɾi.ˈsãw̃]

```
#ipa("['lIt \\v l \\s 'b2R \\schwar ,flaI]")
```

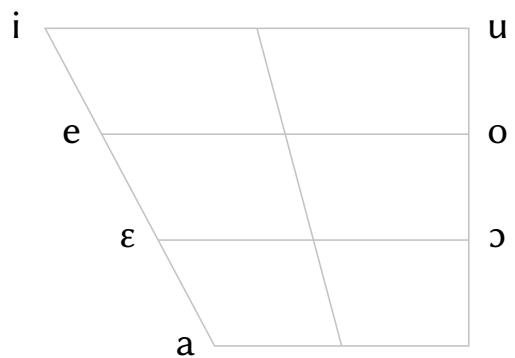
[ˈlɪtˌ ˈbʌɾɚˌflaɪ]

- Referências adaptadas do pacote `tipa` (LaTeX), com otimização de símbolos
- Por exemplo, [ɲ]: `\textltailn` (LaTeX) → `\\nh` (**phono*kit***)

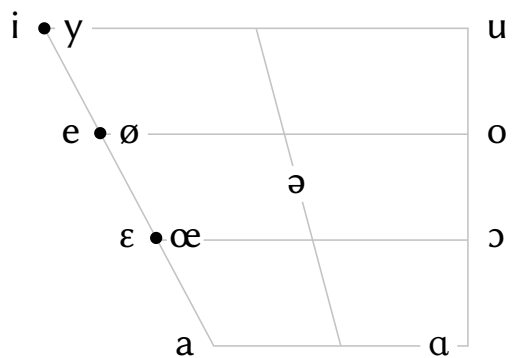
Inventários fonêmicos

☞ Trapézios vocálicos com algumas línguas pré-configuradas

```
#vowels("portuguese")
```



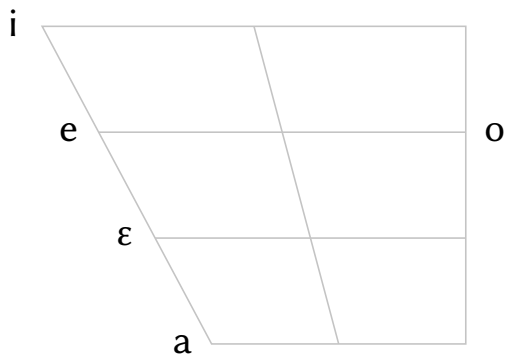

```
#vowels("french")
```



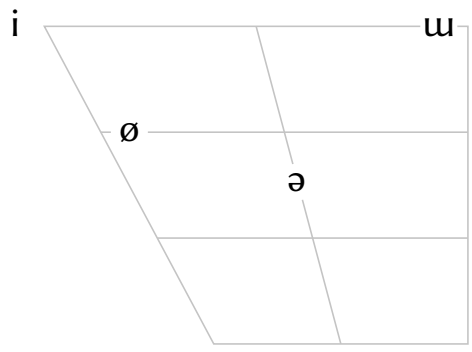
Inventários fonêmicos

☞ Trapézio com input (string) personalizado

```
#vowels("aeoiE")
```



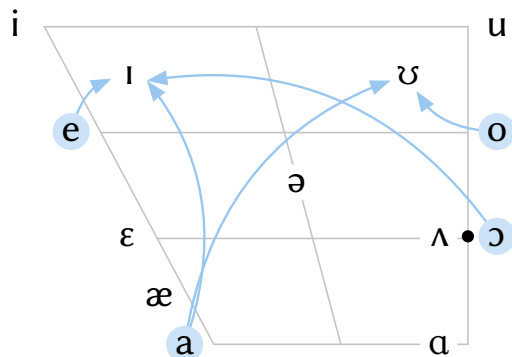

```
#vowels("\\o iW@")
```



Inventários fonêmicos

☞ Trapézio com setas, highlights, etc.

```
#vowels(
    "english",
    arrows: (
      ("a", "U"),
      ("a", "I"),
      ("e", "I"),
      ("O", "I"),
      ("o", "U"),
    ),
    arrow-color: blue.lighten(60%),
    curved: true,
    highlight: ("a", "e", "o", "O"),
    highlight-color: blue.lighten(80%),
  )
```



Inventários fonêmicos

☞ Tabela consonantal (input = string) com línguas pré-definidas ou lista personalizada

```
#consonants("portuguese", scale: 0.6)
```

	Bilabial	Labiodental	Dental	Alveolar	Postalveolar	Retroflex	Palatal	Velar	Uvular	Pharyngeal	Glottal
Plosive	p b			t d				k g			
Nasal	m			n			ɲ				
Trill											
Tap or Flap				ɾ							
Fricative		f v		s z	ʃ ʒ			x			
Lateral fricative											
Approximant	w						j	w			
Lateral approximant				l			ʎ				

Inventários fonêmicos

☞ Tabela consonantal (input = string) com línguas pré-definidas ou lista personalizada

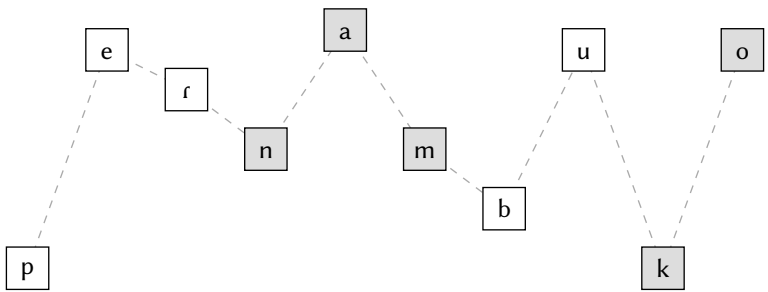
```
#consonants("french", scale: 0.6)
```

	Bilabial	Labiodental	Dental	Alveolar	Postalveolar	Retroflex	Palatal	Velar	Uvular	Pharyngeal	Glottal
Plosive	p b			t d				k g			
Nasal	m			n			ɲ				
Trill				r							
Tap or Flap											
Fricative		f v		s z	ʃ ʒ						
Lateral fricative											
Approximant	w						j	w			
Lateral approximant				l							

Sonoridade

☞ Representação visual do princípio de sonoridade (Parker, 2011)

```
#sonority("peR.nam.bu.ko", scale: 0.7)
```

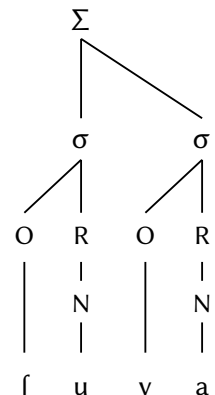


Representação prosódica

☞ Sílaba e pé métrico

```
#syllable("maR")
```

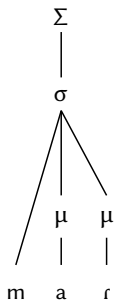

```
#foot("'Su.va")
```



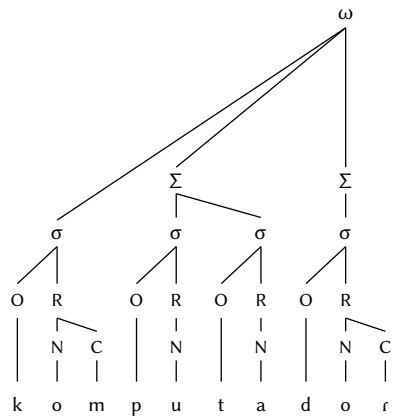
Representação prosódica

☞ Representação moraica e palavra prosódica

```
#foot-mora("maR", coda: true, scale: 0.8)
```



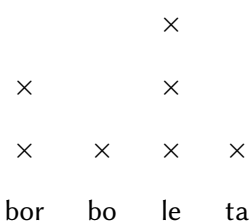

```
#word("kom.('pu.ta).('doR)", scale: 0.7)
```



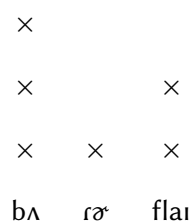
Representação prosódica: grade métrica

☞ Input como string (esq) ou tuple com suporte para IPA (dir)

```
#met-grid("bor2.bo1.le3.ta1")
```



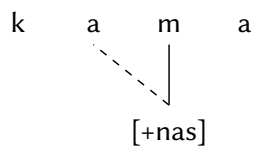

```
#met-grid(("b2", 3), ("R \\shwar", 1), ("flaI", 2))
```



Fonologia autossegmental

☞ Processos de assimilação com representações intuitivas

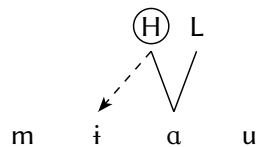
```
#autoseg(
    ("k", "a", "m", "a"),
    features: ("", "", "[+nas]", ""),
    links: ((2,1),),
    spacing: 1.0,
    arrow: false,
)
```



Fonologia autossegmental

☞ Processos tonais com representações intuitivas

```
#autoseg(
    ("m", "1", "A", "u"),
    features: ("", "", ("H", "L"), ""),
    tone: true,
    links: (((2,0),1),),
    highlight: ((2,0),),
    spacing: 1.0,
    arrow: true,
)
```



SPE

☞ Matrizes de traço e regras

```
#feat-matrix("\\ae")
```

/æ/

$$\begin{bmatrix} \text{+syllabic} \\ \text{−consonantal} \\ \text{+sonorant} \\ \text{+continuant} \\ \text{+voice} \\ \text{−high} \\ \text{+low} \\ \text{+front} \\ \text{−back} \\ \text{−round} \end{bmatrix}$$

```
#feat("+son", "-approx") #a-r #feat(sym.alpha +
[#smallcaps("place")]) / #blank()\]#sub[#sym.sigma]
#feat("-son", "-cont", "-del rel", sym.alpha +
[#smallcaps("place")])
```

$$\begin{bmatrix} \text{+son} \\ \text{−approx} \end{bmatrix} \rightarrow \left[\alpha\text{PLACE}\right] / \underline{\quad}\,]_{\sigma} \begin{bmatrix} \text{−son} \\ \text{−cont} \\ \text{−del rel} \\ \alpha\text{PLACE} \end{bmatrix}$$

OT

☞ Tableaux dinâmicos com sombreamento automático

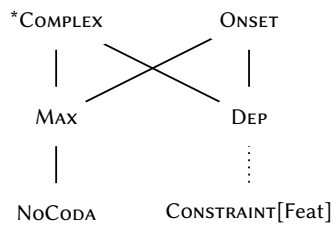
```
#tableau(
    input: "kraTa",
    candidates: ("kra.Ta", "ka.Ta", "ka.ra.Ta"),
    constraints: ("Max", "Dep", "*Complex"),
    violations: (
    ("", "", "*"),
    ("*!", "", ""),
    ("", "*!", ""), ),
    winner: 0,
    dashed-lines: (1,),
    shade: true
)
```

/kraθa/	Max	Dep	*Complex
☞ kra.θa			*
ka.θa	*!		
ka.ra.θa		*!	

Diagramas Hasse

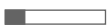
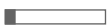
☞ Função para visualização de rankings em OT

```
#hasse(
    (
    ("*Complex", "Max", 0),
    ("*Complex", "Dep", 0),
    ("Onset", "Max", 0),
    ("Onset", "Dep", 0),
    ("Max", "NoCoda", 1),
    ("Dep", "Constraint[Feat]", 1, "dotted"),
    ),
    node-spacing: 3,
 )
```



MaxEnt

☞ Tableaux MaxEnt: cálculo automático, visualização (opcional) de probabilidades

		$w = 2.5$	$w = 1.8$	$w = 0.5$			
/kraθa/	MAX	DEP	*COMPLEX	h_i	e^{-h_i}	P_i	
[kra.θa]	0	0	1	0.5	0.607	0.71	
[ka.ra.θa]	0	1	0	1.8	0.165	0.194	
[ka.θa]	1	0	0	2.5	0.082	0.096	

- Argumento `sort: true` para ordenar candidatos por P_i

MaxEnt

☞ Tableaux MaxEnt com cálculo automático e visualização (opcional) de probabilidades

```
#maxent(
    input: "kraTa",
    candidates: ("[kra.Ta]", "[ka.Ta]", "[ka.ra.Ta]"),
    constraints: ("Max", "Dep", "*Complex"),
    weights: (2.5, 1.8, 0.5),
    violations: (
      (0, 0, 1),
      (1, 0, 0),
      (0, 1, 0),
    ),
    visualize: true, // ← visualização de probabilidades
    sort: true, // ← ordenamento de candidatos
)
```

4 Considerações finais

Ideia geral

phonokit

- Fonology → **agilidade** na codificação fonológica a partir de dados escritos
- phonokit → **qualidade** e **precisão** em representações fonológicas

☞ Quarto (Posit) já oferece suporte a Typst (estes slides): ambiente de trabalho **unificado**

IDEs/editores com suporte excelente para ambas as ferramentas: VS Code, Positron, NeoVim, etc.

Feedback, bugs, dúvidas 

Fonology `gdgarcia.ca/fonology`

phonokit `gdgarcia.ca/phonokit`

guilhermegarcia/fonology

guilhermegarcia/phonokit

Agradecimentos

O pacote Fonology está ligado a um projeto financiado pelo governo canadense sobre estatística lexical e padrões fonológicos em aquisição de línguas

- Université Laval
- *Social Sciences and Humanities Research Council of Canada* (SSHRC)
- **Bolsistas**: Nicolas Bustos, Emmy Dumont, Matéo Levesque, Linda Wong,

obrigado!

Referências

Garcia, G. D. (2026a). *Fonology: Phonological Analysis in R*. https://gdgarcia.ca/fonology

Garcia, G. D. (2026b). *phonokit: a toolkit to create phonological representations in Typst*. Zenodo. https://doi.org/10.5281/zenodo.18434478

Goldwater, S., & Johnson, M. (2003). Learning OT constraint rankings using a Maximum Entropy model. *Proceedings of the Stockholm Workshop on Variation within Optimality Theory*, 111-120.

Parker, S. (2011). Sonority. In M. van Oostendorp, C. J. Ewen, E. Hume, & K. Rice (éds.), *The Blackwell Companion to Phonology* (p. 1160-1184). Wiley Online Library. https://doi.org/10.1002/9781444335262.wbctp0049