Distribuição de *tags* de POS em Cícero e sua variação no tempo e segundo o gênero

FLL 5133 – 2021

Victor Chabu

1 Introdução

No EP 7 desenvolvido neste semestre, avaliou-se a possibilidade de caracterizar a autoria de um texto por meio de marcadores linguísticos referentes a partes do discurso (POS). A hipótese de base era que um autor, escrevendo em um certo gênero, tende a produzir sentenças cujas partes do discurso obedeçam a uma distribuição de probabilidades estável. Embora não seja absurdo pensar que alguns marcadores de POS sejam mais relevantes que outros para esse tipo de análise, e mesmo que um ou outro possa ser por si só determinante, o objetivo ali foi estudá-los individualmente, mas sim em conjunto.

Tomamos $\mu_1, \ldots, \mu_n$ como médias por sentença da densidade[1] dos marcadores $1, \ldots, n$ produzidas por dado autor escrevendo em um determinado gênero, distribuídas com variâncias $\sigma_1^2, \ldots, \sigma_n^2$; avaliamos quantidade:

$$\chi^2 = \sum_{i=1}^{n} \left(\frac{x_i - \mu_i}{\sigma_i} \right)^2, \tag{1}$$

em que x_i representam as médias de marcadores POS por sentença medidas em um texto cuja autoria se queira determinar.

Se cada x_i for uma variável aleatória gaussiana de média μ_i e variância σ_i^2 independente de x_j, $j \neq i$, então χ^2 deverá ser uma variável aleatória distribuída como um qui-quadrado de média n e variância $2n$, e poderemos aplicar um teste de confiança de qui-quadrado padrão para determinar se o texto em análise é compatível com a produção de determinado autor.

Essa abordagem apresenta algumas dificuldades importantes que precisaram ser dirimidas.

- A primeira delas é que as variáveis x_i não podem ser perfeitamente gaussianas, já que, por serem contagens, temos sempre $x_i \geqslant 0$; além disso, temos desvios padrões relativos muitas vezes superiores a 1 (chegamos a ter $\frac{\sigma}{\mu} \sim 40$ para alguns marcadores). Ora, uma variável positiva com desvio padrão maior que a média não pode ser nem mesmo simétrica, muito menos gaussiana.

- Para os textos utilizados como *corpora* naquele EP (*O Guarani* de José de Alencar, *O Abolicionismo* de Luís Gama, e *Dom Casmurro*, de Machado de Assis) houve correlações entre x_i e x_j, $i \neq j$, que chegavam à ordem de 40%, o que descaracterizaria a quantidade χ^2 em (1) como um qui-quadrado mesmo que tivéssemos variáveis $x_1, \ldots, x_n$ perfeitamente gaussianas.

- Não dispúnhamos de nenhum modelo que fornecesse valores teóricos μ_i e σ_i para cada autor, devendo esses ser substituídos por estimadores m_i e s_i calculados a partir de dados obtidos em *corpora* disponíveis para os diversos autores.

  Com relação à média, não há problema, já que podemos absorver a flutuação estatística de m_i naquela de x_i. Com efeito, estimando μ_i em uma amostra com N dados, tomamos $m_i = \mu_i + \varepsilon_i$, em que ε_i é uma variável aleatória de média 0, e obtemos:

  $$x_i - m_i = (x_i - \varepsilon_i) - \mu_i = y_i - \mu_i,$$

  em que y_i é uma variável aleatória com média μ_i e variância:

  $$\sigma_{y_i}^2 = \sigma_i^2 + \frac{\sigma_i^2}{N},$$

  em que $\sigma_{m_i}^2 = \frac{\sigma_i^2}{N}$ é simplesmente a variância do estimador m_i da média para N dados.

[1] A densidade aqui calculada como o número total de ocorrências do marcador de POS em uma sentença dividido pelo tamanho dessa sentença medido em *tokens*.

O mesmo não pode ser feito de maneira direta com relação à variância. Na realidade, se uma variável do tipo

$$g \sim \frac{x-\mu}{\sigma}$$

é gaussiana, sabemos que a variável correspondente substituindo o desvio padrão por seu estimador s,

$$t \sim \frac{x-\mu}{s},$$

será distribuída segundo a função t de Student. Qualitativamente, o t Student assemelha-se à curva normal, exceto que "mais larga", refletindo o fato de que a flutuação do estimador s insere na variável t uma incerteza adicional que g não tinha.

Ora, aqui não é claro nem sequer se as variáveis x_i podem ser adequadamente aproximadas por gaussianas, de modo que a diferença entre t e g será ignorada.

Como resultado, calculamos experimentalmente a variável:

$$\chi^2 = \sum_{i=1}^{n} \frac{(x_i - m_i)^2}{s_i^2\left(1+\frac{1}{N}\right)}. \tag{2}$$

Assim, para a variável aleatória χ^2 em (2), os valores m_i e s_i foram obtidos dos *corpora* disponíveis para cada autor, e x_i foram dados medidos no texto cuja autoria se queria identificar, no caso, *Ubirajara*, de José de Alencar.

A fim de dar conta dos problemas levantados acima, discutiu-se naquele trabalho que o melhor procedimento seria calcular as densidades de marcadores de POS a partir de grupos de médias 50 sentenças aleatoriamente selecionadas de cada texto, restringindo-nos apenas àqueles marcadores cujas distribuições mais se assemelhassem a gaussianas, e que apresentassem menores correlações entre si[2], de modo que, das 23 *tags* disponíveis para língua portuguesa, utilizamos apenas

`ADJ, PROPN, ADP, PUNCT, ADV, SCONJ, AUX, NOUN, CCONJ, VERB, DET, PRON,`

além da variável `Tam` referente ao tamanho médio das sentenças, em *tokens*.

A partir de tal procedimento, calculamos o valor da variável χ^2 definida em (2), seu desvio padrão, e os correspondentes valores para o χ^2 reduzido, considerando os dados obtidos de *Ubirajara* e as estatísticas referentes às distribuições nos diversos textos do *corpus*, chegando aos seguintes resultados:

```
Guarani.txt       | X2 = 26.37 +- 5.10, X2_redux = 2.03 +- 0.39, gdl = 13
DomCasmurro.txt   | X2 = 124.81 +- 5.10, X2_redux = 9.60 +- 0.39, gdl = 13
Abolicionismo.txt | X2 = 75.24 +- 5.10, X2_redux = 5.79 +- 0.39, gdl = 13
```

Repetindo o cálculo diversas vezes para observar a variação desses valores com as diferentes seleções aleatórias de 50 sentenças sobre as quais calculamos das densidades das *tags*, obtivemos valores consistentes:

```
Guarani.txt       | X2 = 25.36 +- 5.10, X2_redux = 1.95 +- 0.39, gdl = 13
DomCasmurro.txt   | X2 = 91.87 +- 5.10, X2_redux = 7.07 +- 0.39, gdl = 13
Abolicionismo.txt | X2 = 94.91 +- 5.10, X2_redux = 7.30 +- 0.39, gdl = 13

Guarani.txt       | X2 = 25.88 +- 5.10, X2_redux = 1.99 +- 0.39, gdl = 13
DomCasmurro.txt   | X2 = 101.48 +- 5.10, X2_redux = 7.81 +- 0.39, gdl = 13
Abolicionismo.txt | X2 = 119.15 +- 5.10, X2_redux = 9.17 +- 0.39, gdl = 13

Guarani.txt       | X2 = 30.65 +- 5.10, X2_redux = 2.36 +- 0.39, gdl = 13
DomCasmurro.txt   | X2 = 107.35 +- 5.10, X2_redux = 8.26 +- 0.39, gdl = 13
Abolicionismo.txt | X2 = 77.51 +- 5.10, X2_redux = 5.96 +- 0.39, gdl = 13
```

Como se pôde ver, em qualquer uma das 4 iterações do programa houve uma clara diferença no valor de χ^2 obtido para José de Alencar ou para os outros autores, o que parece compatível com o fato de que, efetivamente, *Ubirajara* é um romance de José de Alencar em sua fase indianista, assim como *O Guarani*. Entretanto, tais resultados suscitaram diversos questionamentos sobre a compatibilidade das distribuições de *tags* de POS em textos de um mesmo autor em períodos diferentes de sua produção, ou quando ele escreve segundo gêneros literários reconhecidamente distintos, ao mesmo tempo que se pergunta se algumas das *tags* obedeceriam a distribuições mais típicas de cada autor, portanto mais determinantes de autoria.

[2] Correlação não implica dependência, mas o contrário sim, de modo que apenas considerar variáveis pouco correlacionadas é uma forma de certificar-se de que sejam também independentes.

À guisa de resposta, precisaríamos estudar a distribuição de *tags* de POS na obra de diversos autores cuja produção se estenda ao longo do tempo e que também se notabilizem pela variedade de temas e gêneros. Um autor com essas características é Marco Túlio Cícero, cuja trabalho estende-se por 40 anos, abrange os mais diversos assuntos, e assume formas bastante distintas, com um amplo conjunto de discursos políticos e judiciais (aqui agrupados no gênero `oratoria`), de correspondência com amigos e familiares (gênero `epistola`), e uma importante obra filosófica (gênero `filosofia`) geralmente no formato de diálogo. Neste artigo, analisaremos a distribuição de POS no *corpus* ciceroniano, a fim de estabelecer uma metodologia que possibilite responder às questões do parágrafo anterior.

Além disso, proporemos algumas modificações serão necessárias nos procedimentos empregados no EP 7, uma vez que ali o *corpus* de cada autor era um conjunto de grupos de 50 sentenças extraídas de um mesmo texto, e na nova análise precisaremos lidar com diversos textos agrupados por gênero ou por período; em outras palavras, enquanto ali testávamos a compatibilidade da distribuição de *tags* de um texto com a do outro, aqui precisaremos testar a distribuição de *tags* em um texto com um conjunto de outros, que não necessariamente possuem as mesmas distribuições entre si. Não é claro, por exemplo, como os resultados seriam afetados se concatenássemos os diversos textos de um gênero, e do texto total resultante fizéssemos as seleções de 50 sentenças calculando daí estatísticas para uma distribuição única por gênero, ou se tomássemos individualmente cada texto e para ele calculássemos as estatísticas para cada um, e delas as estatísticas para uma distribuição de distribuições por gênero. Essa dúvida metodológica reflete uma questão de fundo muito relevante, que é saber se, dentro de um gênero, em um período determinado, as variações de estilo devem-se sobretudo às idiossincrasias do autor, ou às condições tópicas de produção de cada texto.

Uma análise qualitativa de Cícero faz-nos tender fortemente à segunda hipótese. É dele o único caso conhecido de um discurso antigo que sobreviveu até a atualidade em duas versões completas, uma dirigida ao Senado romano e outra à assembleia popular. Nesse discurso, em que Cícero ataca Marco Antônio e defende a posição de Otaviano na guerra civil, nota-se que na elocução senatorial os períodos tendem a ser mais longos, mais complexos, com mais níveis de subordinação e inversões sintáticas, e o próprio texto é mais longo que sua versão concional, mais curta e direta. Não obstante, pode ser que algumas POS variem de forma essencialmente idiossincrática (talvez `ADV` ou `ADJ`?), enquanto que outras (como a prevalência de `SCONJ` para o caso de sucessivas subordinações) realmente dependa mais do contexto e do público alvo.

Assim, neste estudo preliminar consideraremos que cada texto é uma realização de uma variável aleatória valorada em outra distribuição, ou seja, consideraremos que dentro de cada gênero há um intervalo de valores médios e desvios padrões que as distribuições de *tags* de POS podem com alguma probabilidade assumir. Dessa maneira, o i-ésimo texto do gênero g terá uma média de densidade da *tag* de POS p que chamaremos de m_i^p, calculada sobre 48 sentenças[3] aleatoriamente selecionadas do texto, e respectivo desvio padrão da média $\sigma_{m_i^p}$; a média da densidade da *tag* p para o gênero g poderá ser estimada, portanto, por:

$$m_g^p = \sum_i \frac{m_i^p}{\sigma^2_{m_i^p}} \left(\sum_i \frac{1}{\sigma^2_{m_i^p}} \right)^{-1},$$

sendo a variância dessa média estimada por:

$$\frac{1}{\sigma^2_{m_g^p}} = \sum_i \frac{1}{\sigma^2_{m_i^p}};$$

a variância simples dessa distribuição relativa às M distribuições de p no gênero g (uma para cada texto) será estimada pela estatística:

$$\left(s_g^p\right)^2 = \frac{1}{M-1} \sum_i \left(m_i^p - m_g^p\right)^2.$$

Finalmente, avaliaremos a compatibilidade de um texto x – o qual possui densidades médias x^p das *tags* de POS e variâncias da média $\sigma^2_{x^p}$ – com relação ao gênero g por meio da quantidade:

$$\chi^2_g(x) = \sum_p \frac{\left(x^p - m_g^p\right)^2}{\left(s_g^p\right)^2 + \sigma^2_{m_g^p} + \sigma^2_{x^p}}, \tag{3}$$

a qual, espera-se, deve se comportar como um qui-quadrado com $N-1$ graus de liberdade, sendo N a quantidade de *tags* empregada na soma em (3).

Os textos de Cícero utilizados neste trabalho foram obtidos a partir do repositório *The Latin Library*[4], e a *tokenização* e a análise de POS feitas por meio da biblioteca `CLTK` disponível em PYTHON[5]. Todos os arquivos produzidos ao longo do trabalho (códigos, bancos de dados, gráficos, arquivos `csv`, etc.) estarão disponíveis até 31/01/2022 no endereço `https://github.com/vbchabu/fll5133_trabalho_final`.

[3] Essa quantidade foi determinada por tentativa e erro, segundo se constatou que ao agrupar 48 sentenças aleatórias produziam-se variáveis com menores correlações e desvios padrões, resultando em um número maior de *tags* utilizáveis para a análise de χ^2.

[4] Disponível em `https://www.thelatinlibrary.com`, acessado em 20/12/2021.

[5] `http://cltk.org`, acessado em 20/12/2021

2 Metodologia

A metodologia utilizada seguiu a risca a prescrição da fórmula (3) para o cálculo do χ^2; mesmo assim, alguns pontos merecem ser comentados referentes ao processamento dos textos latinos anterior às análises estatísticas e à escolha das *tags* a empregar.

- A ferramenta de análise de POS do `CLTK` provou-se pouco confiável, apresentando uma tendência a classificar uma série de elementos como `NOUN`, ainda que não o fossem, ou ainda que fossem nomes próprios, portanto `PROPN`. Ela se mostrou frágil frente às grandes variações sintáticas permitidas pela língua latina. Na frase *quod me amas bene scio* (lt. *que me amas, bem sei*), *quod* foi classificado como um pronome `PRON`, ainda que em realidade seja uma conjunção subordinante `SCONJ` (o *que* da versão em português); por outro lado, invertendo a ordem do período para *bene scio quod me amas* (lt. *bem sei que me amas*), o *quod* foi corretamente classificado como `SCONJ`. Da mesma maneira, prenomes, que em latim aparecem abreviados, *I.* (Júlio), *Gn.* (Gnaio), *M.* (Marco), *C.* (Caio), etc., foram classificados de maneiras nem sempre difíceis de explicar, oscilando entre numeral, advérbio, nome e nome próprio, ainda que o *tokenizador* do `CLTK` quase sempre (mas não sempre) tenha entendido o ponto como uma abreviação, e não como ponto final.

Por esse motivo, já na fase de pré-processamento do texto resolvemos as siglas de prenomes removendo o ponto da abreviatura (de modo a que o *tokenizador* não quebrasse uma sentença única em duas). Quanto aos problemas da classificação de POS, não houve o que fazer, dado o tempo e o escopo do trabalho. Mesmo assim, alguma classificação foi produzida, portanto podemos dizer que essa classificação é a que ocorre segundo uma variável aleatória. Tal abordagem de "caixa-preta", em que estudamos a distribuição de *tags* que o algoritmo produz, sem que saibamos ao certo por quais razões, permitiu levar o trabalho a cabo e gerou resultados instigantes (ver Seção 3), ainda que às expensas do significado linguístico das variáveis aleatórias estudadas. Uma das perdas sofridas, é que não podemos afirmar que os discursos senatoriais de Cícero possuam períodos compostos mais ou menos complicados que os concionais com base nas médias da *tag* `SCONJ`, pois vimos que o `CLTK` não hesita em classificar conjunções subordinantes como pronomes quando a subordinada aparece, como é frequente em Cícero, antes da oração principal.

- Assim como no EP 7, apareceram correlações importantes entre as variáveis aleatórias, e muitas delas exibiram desvios padrões altos demais para que pudessem ser aproximadas por gaussianas. O procedimento de concatenar certa quantidade de sentenças aleatoriamente escolhidas do texto (aqui, 48) foi o suficiente para amenizar o problema na maioria dos casos, mas algumas das amostras de texto, até mesmo por seu estado excessivamente fragmentário, continuaram problemáticas. Foi necessário excluir da análise algumas *tags* e alguns dos textos existentes no *corpus* ciceroniano, porém, com uma quantidade de amostras muito maior que no EP 7, foi necessário automatizar a tarefa de seleção utilizando o algoritmo `tagselect` (ver Apêndice A.4).

Segundo parametrizado, foram excluídos textos em que, para mais da metade das *tags* disponíveis, a razão $\frac{\sigma^p}{m^p}$ superasse 30%, em seguida excluídas *tags* que, nos textos passados pelo teste anterior, apresentassem essa razão maior que 30% (ou exatamente igual a 0). Isso feito, voltamos a excluir *tags*, dessa vez aquelas que apresentassem alguma correlação superior a 50% em mais da metade dos textos restantes, e voltamos a excluir textos, os que apresentavam mais da metade das *tags* restantes com correlação superior a 50%.

Esse processo pode resultar em seleções distintas de *tags* e textos em iterações diferentes do algoritmo de processamento `tagstats` (ver Apêndice A.3), uma vez que ele calcula as médias de densidade das *tags* em grupos de N sentença selecionadas aleatoriamente. Na realidade, aproveitamos desse fato para rodar o processamento diversas vezes, variando o valor de N entorno de 50, e descobrimos que os melhores resultados (mais textos e mais *tags* aprovados por `tagselect`) ocorriam para $N = 48$; não obstante, mesmo com esse número fixado, diversas iterações de `tagstats` resultam em seleções assaz distintas de textos e *tags*. Na melhor iteração que conseguimos, passaram pelos testes apenas 89 dos 155 textos inicialmente disponíveis no *corpus* ciceroniano, sendo 30 do gênero `oratoria`, 33 de `filosofia` e 26 de `epistola`. Quanto às *tags* selecionadas para cada gênero, foram:

`oratoria`: `ADP`, `ADV`, `CCONJ`, `SCONJ`, `VERB`

`filosofia`: `ADJ`, `ADP`, `ADV`, `AUX`, `CCONJ`, `NOUN`, `PRON`, `PUNCT`, `SCONJ`, `VERB`

`epistola`: `ADP`, `ADV`, `AUX`, `CCONJ`, `PRON`, `PUNCT`, `SCONJ`, `VERB`

3 Discussão dos resultados

3.1 Análise no tempo

Vemos nos gráficos das Figuras 1 e 2 que não há nenhuma tendência de variação considerável no tempo das variáveis m_i^p que representam as médias de densidade de ocorrência de *tags* de POS; ao contrário, a dispersão desses valores dentro de um mesmo período é claramente dominante sobre variações que possam ocorrer ao longo do tempo.

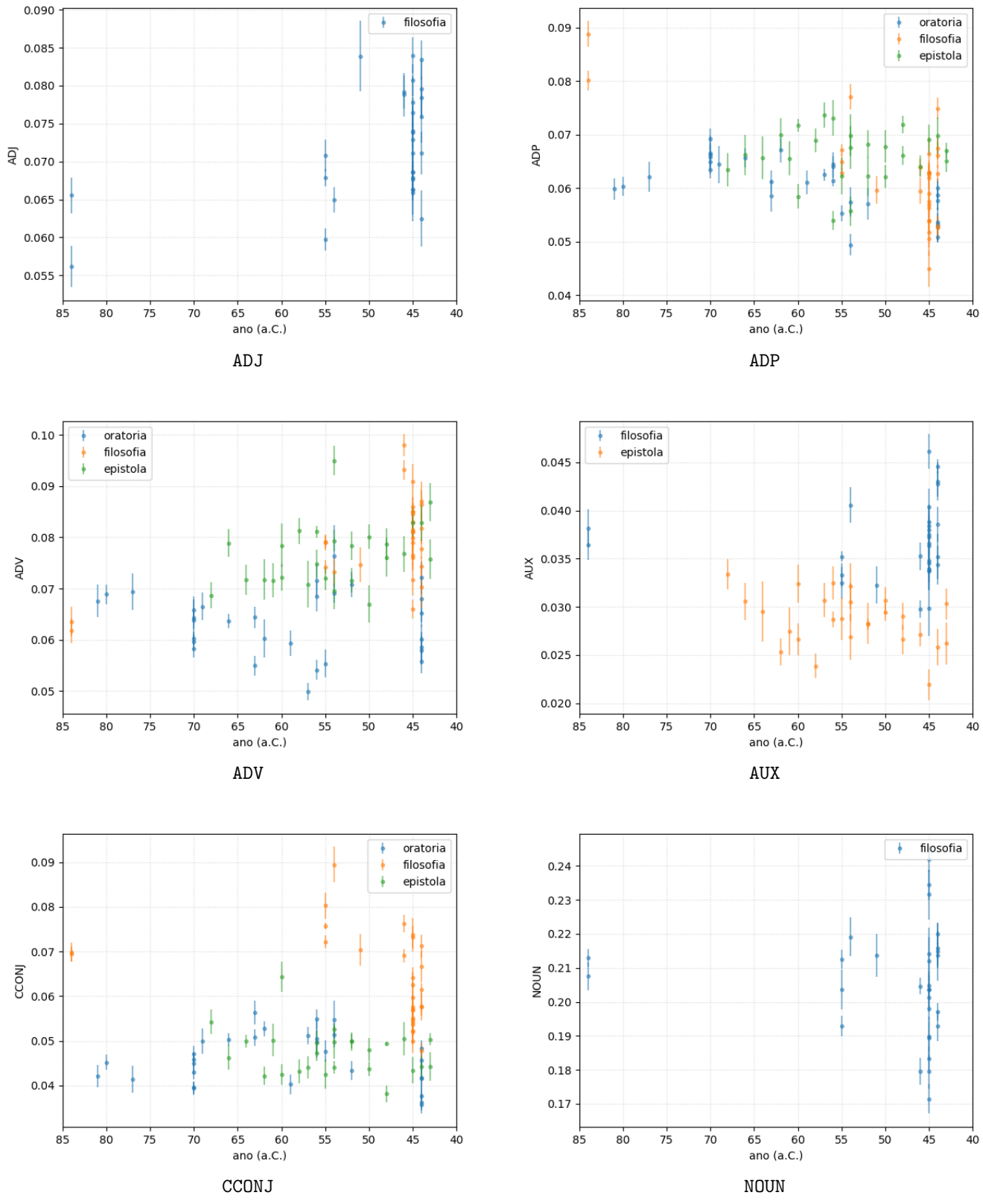
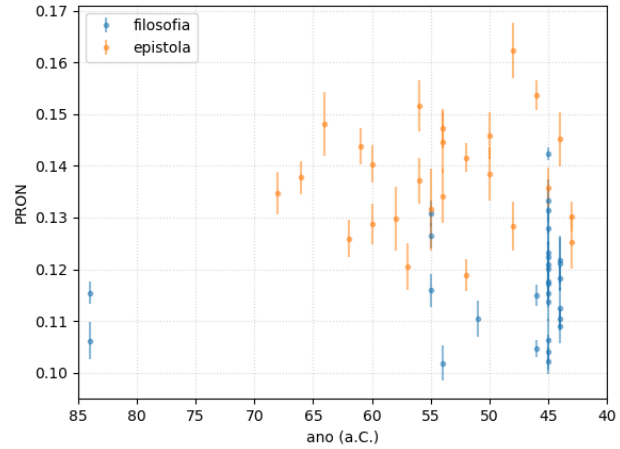
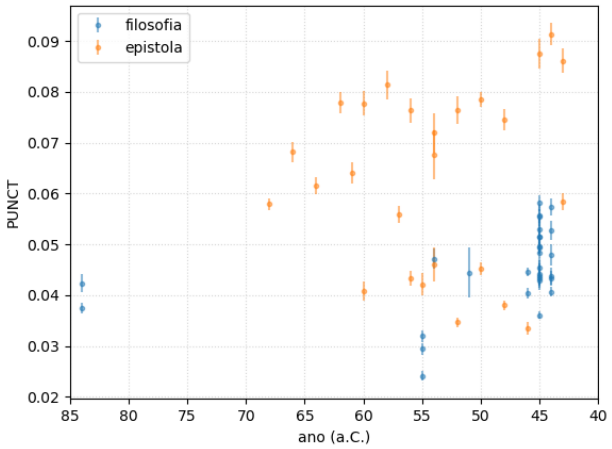


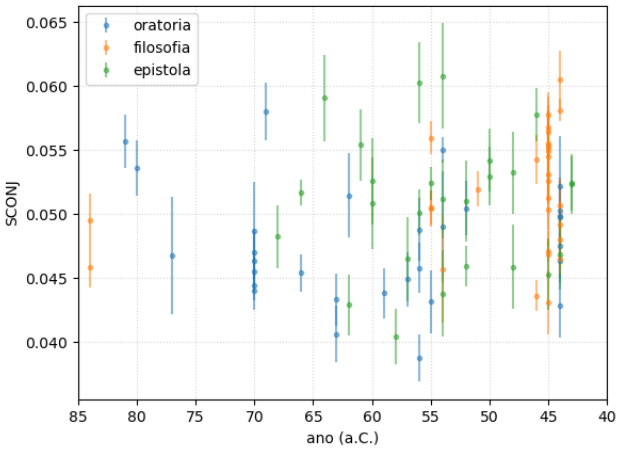
Figura 1: Evolução no tempo das médias, em cada texto, de densidade das *tags* de POS (ocorrências por *token*). Dados referentes a textos pertencentes aos diversos gêneros em questão aparecem discriminados por cor. Continua.



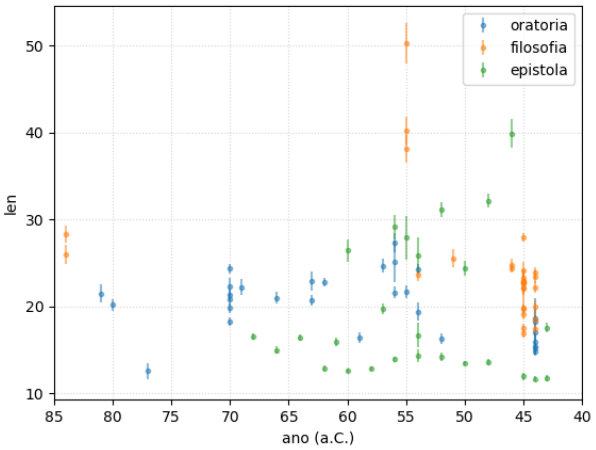
PRON



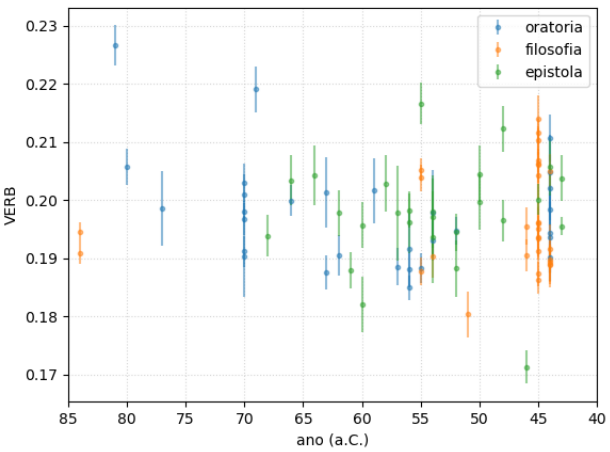
PUNCT



SCONJ



Tam



VERB

Figura 2: Evolução no tempo das médias, em cada texto, de densidade das *tags* de POS (ocorrências por *token*). Dados referentes a textos pertencentes aos diversos gêneros em questão aparecem discriminados por cor. Continuação.

3.2 Análise de χ^2

Exibimos na Figura 3 os dados m_i^p histogramados para cada p, agrupados por cor segundo o gênero de i:

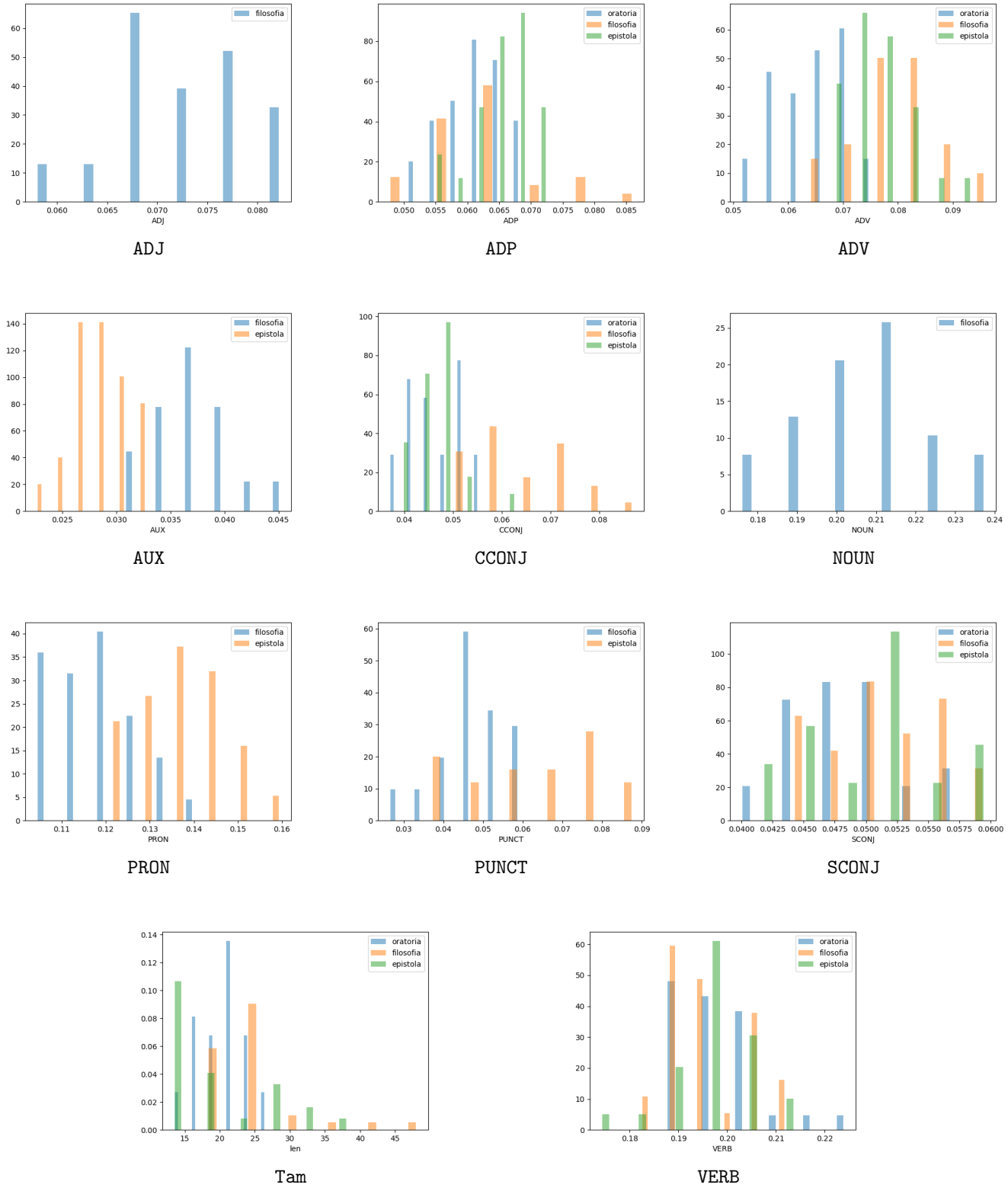
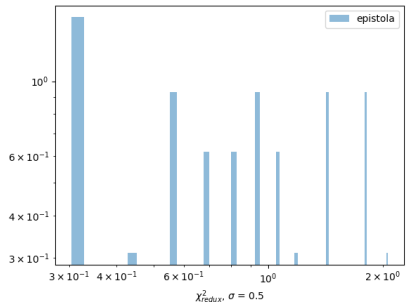
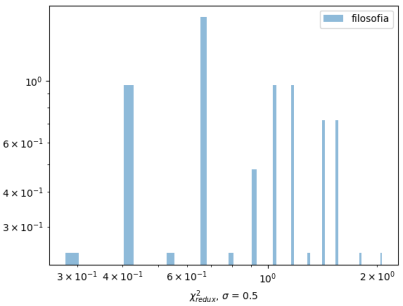


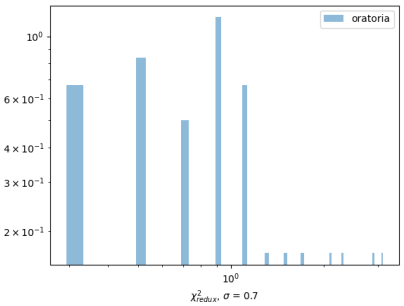
Figura 3: Distribuição das densidades de *tags* de POS (ocorrências pro *token*) para cada *tag*. Os dados referentes a textos classificados nos diversos gêneros aparecem discriminados por cor. Nos gráficos somente estão presentes os dados relativos a textos para cujo gênero a *tag* em questão tenha sido selecionada segundo discutido (ver o algoritmo `tagselect` no Apêndice A.4).



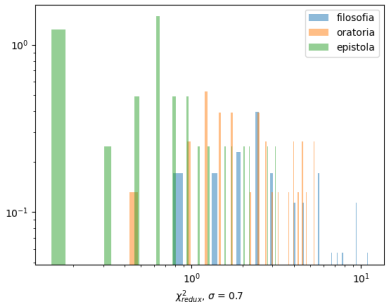
$g =$ `epistola`



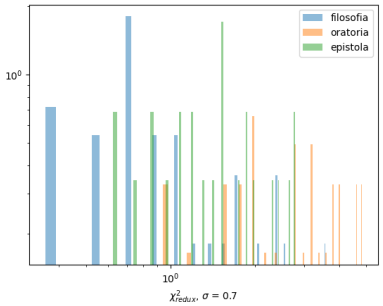
$g =$ `filosofia`



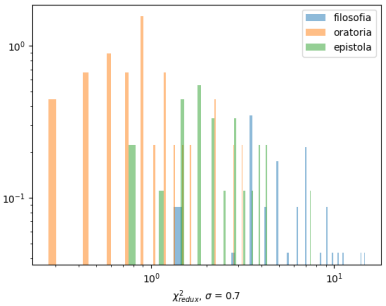
$g =$ `oratoria`



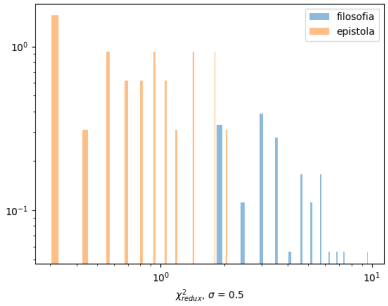
$g =$ `epistola`



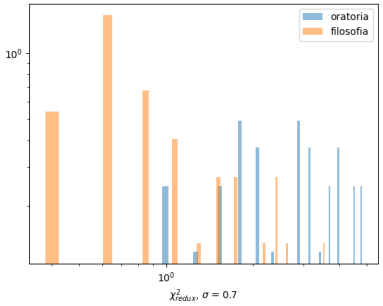
$g =$ `filosofia`



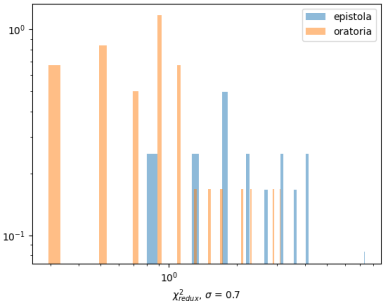
$g =$ `oratoria`



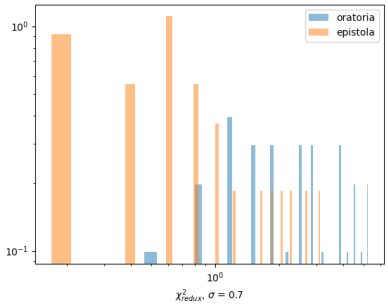
$g =$ `epistola`



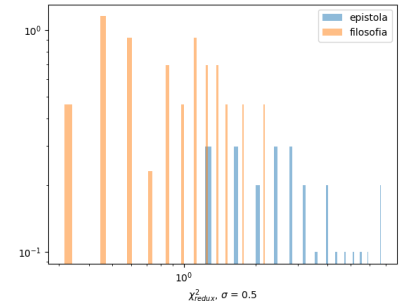
$g =$ `filosofia`



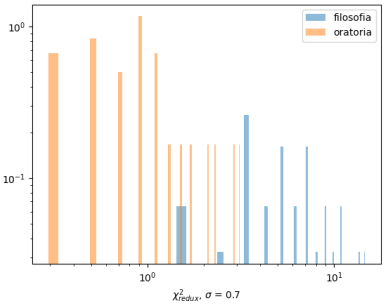
$g =$ `oratoria`



$g =$ `epistola`



$g =$ `filosofia`



$g =$ `oratoria`

Figura 4: Distribuição dos valores de χ^2_g reduzido, para os diversos valores de g (gênero). Os dados referentes a textos classificados segundo os distintos gêneros aparecem discriminados por cor. As *tags* de POS utilizadas para o cálculo de χ^2_g são aquelas selecionadas concomitantemente para os gêneros em análise segundo discutido (ver o algoritmo `tagselect` no Apêndice A.4), e podem ser identificadas a partir da Figura 3.

Nota-se que, fixando um gênero g, diversas das variáveis $g \ni i \mapsto m_i^p$ aparentam normalidade, embora algumas, sobretudo para $p =$ `PRON` em $g =$ `filosofia`, presumivelmente se comportem mais como distribuições de Poisson, as quais, em todo caso, costumam descrever processos de contagens para poucas contagens, deixando-se aproximar pela gaussiana conforme a quantidade de contagens aumenta. Quanto às correlações entre as variáveis, ainda que não seja essencial utilizar no cálculo de χ^2 variáveis descorrelacionadas (mas sim independentes!), é importante notar que existe um padrão complexo de correlação entre as diversas variáveis, sugerindo que eles não possam ser bem aproximadas por variáveis aleatórias independentes. Na Figura 6, observamos como a *tag* `NOUN` aparece fortemente correlacionada com `PRON` para o gênero `filosofia`, ao mesmo tempo que aparenta pouca correlação com `ADJ` (dois últimos gráficos da figura); isso certamente se reflete em correlações entre `ADJ` e `PRON` e as demais *tags*. Por exemplo, na mesma figura observamos correlação entre `PRON` e `SCONJ` quando tomamos separadamente as distribuições de `filosofia` e `oratoria`; por inspeção visual, percebe-se que se as tomássemos conjuntamente, a correlação ficaria bastante atenuada.

Já na Figura 4, exibimos as distribuições da função χ_g^2 definida em (3), para diversos valores de g. As *tags* empregadas no somatório de (3) são aquelas cujos dados estão disponíveis concomitantemente para todos os gêneros em comparação em cada gráfico. Assim, na primeira fileira da figura, os gráficos referem-se às distribuições de χ_g^2 definidos com as *tags* selecionadas por `tagselect` para `oratoria`, `filosofia` e `epistola`, descritas na Seção 2; na segunda fileira, os gráficos trazem χ_g^2 em que se utilizam apenas as *tags* na intersecção dos três conjuntos, e nas demais fileiras, *tags* nas intersecção dos conjuntos de *tags* referentes aos dois gêneros em tela. Outras seleções de *tags* seriam possíveis (desde que haja dados disponíveis para cada uma), por exemplo aquelas que, visualmente, nas Figuras 5 e 6, mostrem-se mais adequadas para separar dados de um ou outro gênero.

Qualquer que seja o conjunto de *tags* utilizadas no somatório de χ_g^2, os resultados permanecem inalterados do ponto de vista qualitativo: os textos pertencentes ao gênero g sistematicamente apresentam valores de χ_g^2 mais baixos que os demais, a ampla maioria deles dentro de um intervalo de um desvio padrão entorno da média, e o formato das distribuições de $g \ni i \mapsto \chi_g^2(x_i)$ é compatível com o que se esperaria de uma genuína distribuição de χ^2. Para textos que não sejam do gênero g, na maioria das vezes os valores de χ_g^2 superam a média por mais que dois ou mesmo três desvios padrões; é o caso dos textos de `filosofia` quando comparados com χ_g^2 para $g =$ `epistola`, usando as *tags* desses dois gêneros (primeira coluna, terceira fileira da figura). Contudo, essa é uma conclusão longe de ser geral, por exemplo no cálculo de χ_g^2 em que se utilizam *tags* dos três gêneros, notadamente para $g =$ `epistola` (primeira coluna, segunda fileira) e para $g =$ `filosofia` (segunda coluna, segunda fileira). Assim, demonstra-se a compatibilidade das distribuições $g \ni i \mapsto m_i^p$ enquanto observações de variáveis aleatórias m_g^p distintas para cada gênero e mutuamente independentes (dentro dos conjuntos de *tags* selecionadas por *tagselect*), ainda que para alguns valores de p as médias e variâncias dessas distribuições possam coincidir entre os gêneros, como veremos adiante na Seção 3.3.

3.3 Análise de dispersão

As análises de χ^2 permitiram confirmar que um mesmo autor produz distribuições de densidade de POS diferentes segundo escreve em diversos gêneros; entretanto, também evidenciou a dificuldade de separar textos por gênero com base apenas nos valores de χ^2, algo que havíamos feito para atribuir autoria, no EP 7. Isso se deve ao fato de as diferenças de densidade média entre um gênero e outro serem, via de regra, ordens de grandeza menores que seus respectivos desvios padrões. Entretanto, com uma análise mais fina é possível identificar características que classificam os textos como pertencentes a um ou outro gênero. Em muitos dos gráficos de dispersão das Figuras 5 e 6 vemos claramente como `epistola` e `filosofia` ocupam regiões distintas do espaço $p_x \times p_y$; na dispersão `PRON` × `AUX` pode-se mesmo separar esses gêneros por um classificador linear! A identificação do gênero `oratoria` parece menos clara, pois em muitos casos sobrepõe-se com `epistola`[6], ainda assim não impossível, considerando `ADV` × `ADP`, em que parece se separar dos demais, ou diversos outros, como `CCONJ` × `SCONJ`, em que se separa bem de `filosofia`.

É importante lembrar que a comparação de dispersão por duplas de *tags* dá-se apenas por motivos de visualização gráfica; eventualmente, se comparássemos três ou mais *tags* poderíamos determinar de maneira mais precisa características não óbvias dos textos, envolvendo diversas relações entre as médias de densidade de POS, que os classificariam segundo gênero. Tal estudo poderia ser realizado por meio da estruturação de redes neurais que utilizem que utilizem como *embedding* para os textos a classificar os valores de m_i^p para as *tags* p disponíveis.

4 Conclusão

As densidades médias de *tags* de POS, em Cícero, variam apreciavelmente, mas sem relação direta com o tempo, podendo em um mesmo ano apresentar variações muito maiores que em décadas. Além disso, constatamos que, para certas *tags*, essas médias distribuem-se normalmente de formas próprias a cada gênero e independentes entre si. Apesar disso, não é possível utilizá-las em testes de χ^2 a fim de classificar os textos de Cícero em gênero, embora elas sejam uma forma de *embedding* que se demonstra promissor para a estruturação de redes neurais que façam tal classificação.

[6] Talvez porque em suas correspondências Cícero não deixa de discursar de maneira retoricamente conformada, ainda que no âmbito privado, enquanto que na obra filosófica haja uma preponderância de diálogos em que as falas têm mais traços de oralidade e espontaneidade?

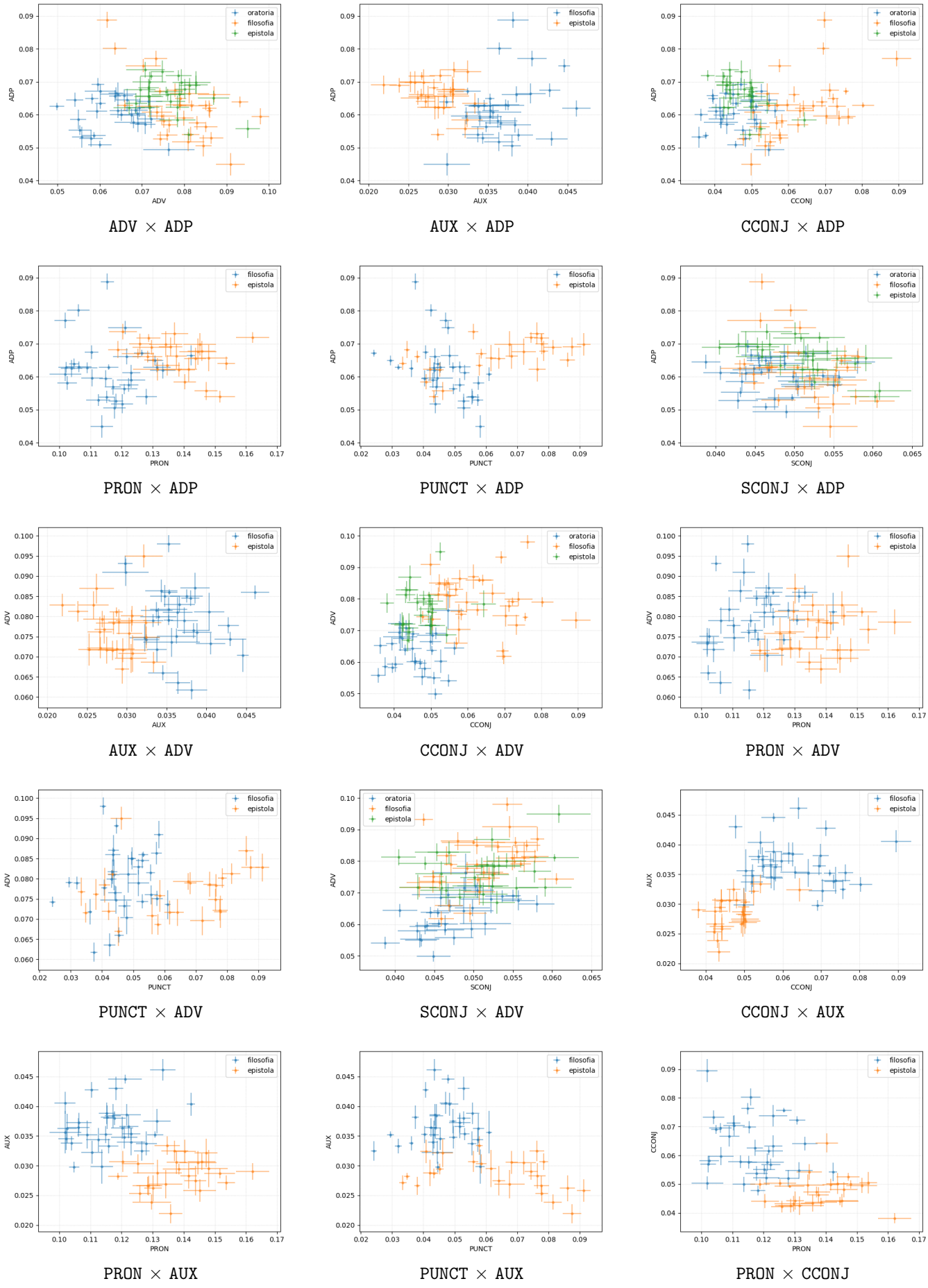


Figura 5: Gráficos de dispersão para pares de médias de densidades de *tags* de POS (ocorrências por *token*), discriminados por gênero. Continua.

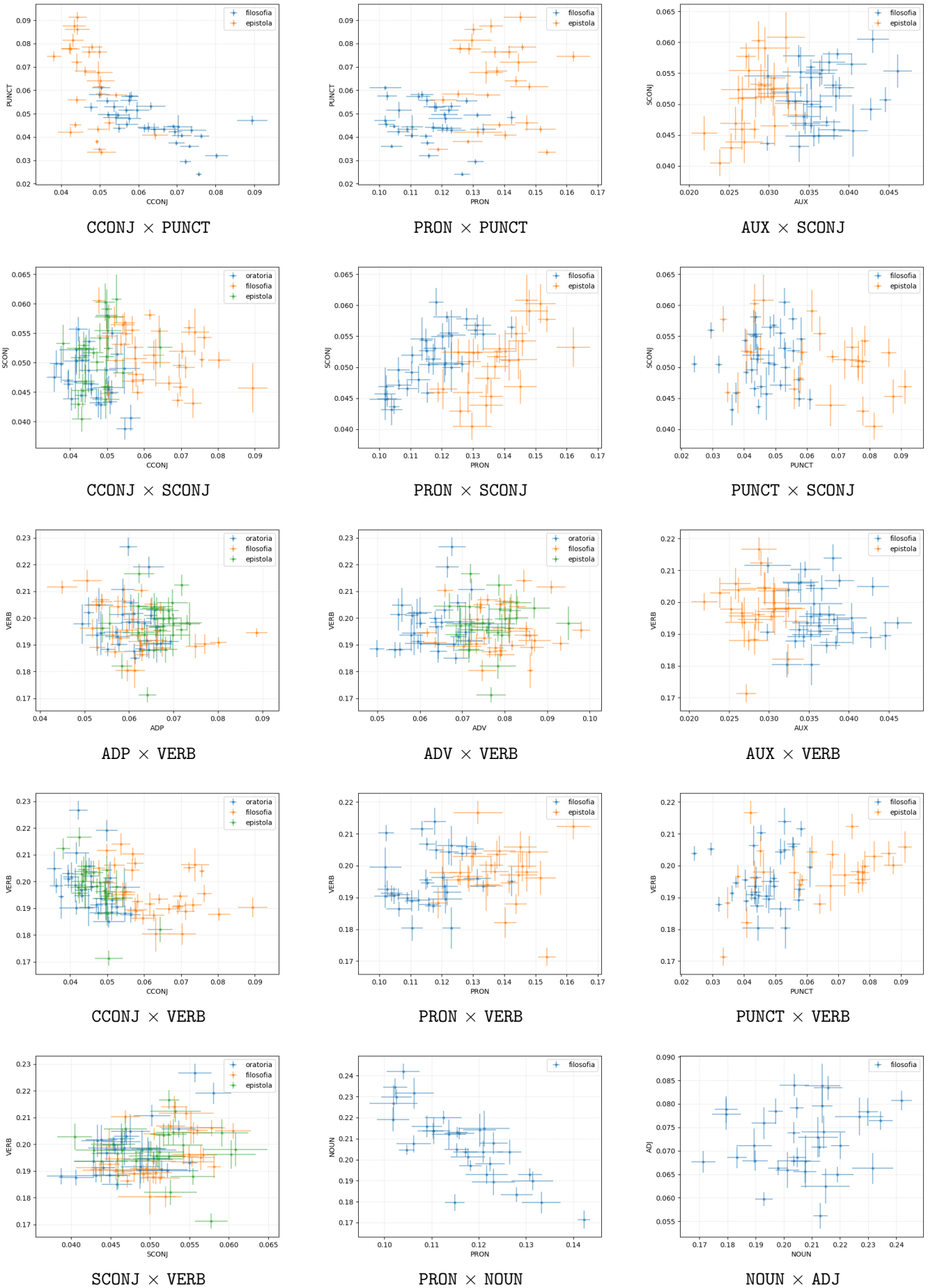


Figura 6: Gráficos de dispersão para pares de médias de densidades de *tags* de POS (ocorrências por *token*), discriminados por gênero. Continuação.

A Código Python

Todos os códigos abaixo (e ainda outros, assim como gráficos e diversos dados, brutos e processados), encontram-se disponíveis em `https://github.com/vbchabu/fll5133_trabalho_final` até 31/01/2022.

A.1 `amostras.py`

Define a classe `Amostra` com atributos relativos a textos em análise e métodos para gravação e leitura desses dados em arquivo, já fazendo a análise de `POS`.

```
from re import sub
from pandas import DataFrame,read_csv
from cltk.nlp import NLP
from cltk.sentence.lat import LatinPunktSentenceTokenizer

nlp = NLP(language='lat')
sent_tokenizer = LatinPunktSentenceTokenizer().tokenize

class Amostra():

    def __init__(self,autor,obra,dados=None):
        self.autor = autor
        self.obra = obra
        if dados is None:
            dados = self.__filtrar__()
            dados = sent_tokenizer(dados.decode())
            dados = ( nlp(sent) for sent in dados )
            dados = ( (sent.tokens,sent.pos,[ sent.words[i].dependency_relation
   for i in range(len(sent.words)) ]) for sent in dados )
        self.dados = DataFrame(dados,columns=['tokens','POS','DET'])

    def __filtrar__(self):
        with open(self.autor+'/'+self.obra+'.txt','rb') as arquivo:
            texto = arquivo.read()
        texto = sub(b'<head>[^^]*</head>',b'',texto)              #limpa
   cabecalhos
        texto = sub(b'<p class=pagehead>[^<]*</p>',b'',texto)
        texto = sub(b'<div class="footer">[^^]*</div>',b'',texto) #limpa rodape
        texto = sub(b'nbsp;?',b'',texto)          #limpa parte do codigo html
        texto = sub(b'<[^>]*>',b'',texto)         #limpa as demais chaves html
        texto = sub(b'\[[ivxlc]*\]',b'',texto)    #limpa a numeracao dos passos
        texto = sub(b'[!?]', b'\.', texto)        #limpa a pontuacao, exceto \.
   que e necessario para tokenizar
        texto = sub(b'([A-Z])\.',b'\g<1>',texto) #resolve siglas de prenomes
   tirando o \.
        texto = sub(b'([A-Z][a-z])\.',b'\g<1>',texto)
        texto = texto.lower()
        texto = texto.replace(b'v',b'u')          #troca v por u
        texto = texto.replace(b'j',b'i')          #troca j por i
        texto = sub(b'[^abcdefghiklmnopqrstuxyz\. ]',b'',texto)   #limpa todo o
   resto
        return texto

    def __len__(self):
        return len(self.dados)

    def __repr__(self):
        return 'Amostra({},{})'.format(self.autor,self.obra)

    def to_file(self,path):
```

```
        self.dados.append({'tokens':self.autor,'POS':self.obra},ignore_index=True
    ).to_csv(path,index=False)
        #autor e obra sao enxertados na ultima linha do frame a ser salvo

def from_file(path):
    frame = read_csv(path)
    autor,obra = frame.iloc[-1].drop('DET')
    dados = frame.iloc[:-1]
    for col in dados.columns:                #transforma a string lida do arquivo
        dados[col] = dados[col].apply(eval) #em verdadeiro dado tipo list
    return Amostra(autor,obra,dados)
```

A.2 processo_nlp.py

Script que efetivamente processa os *corpora* fornecidos e salva os dados para posterior análise.

```
from multiprocessing import Pool,cpu_count
from amostras import Amostra
from pathlib import Path

def exec(obra):
    amostra = Amostra('cicero',obra)
    amostra.to_file('dados/{}-{}.csv'.format(amostra.autor,amostra.obra))

def main():
    obras = { item.parts[-1].replace('.txt','') for item in Path('cicero/').glob(
    '*.txt') }
    #ja_feito = { item.parts[-1].replace('cicero-','') for item in Path('dados/')
    .glob('*') }
    #obras = obras.difference(ja_feito)
    #back up para o caso de interrupcao do processamento dos dados

    with Pool(cpu_count()-1) as pool:
        i = 0
        n = len(obras)
        print('Feito: {:3d} de {}.'.format(i,n))
        for _ in pool.imap_unordered(exec,obras):
            i += 1
            print('Feito: {:3d} de {}.'.format(i,n))
        print('Pronto.')

if __name__ == '__main__': main()
```

A.3 tagstats.py

Processa dados linguísticos oriundos de amostras da classe `Amostra` segundo a metodologia descrita, e salva as diversas estatísticas obtidas em arquivos `csv`.

```
from collections import Counter
from pandas import DataFrame
from tagsbank import tags
from pathlib import Path

N_sentencas_a_homogeneizar = 48
#N obtido por tentativa e erro de modo a maximizar o numero de tags e de tabelas
#a serem selecionadas por tagselect

def randomizador(original,N):
```

```
11      frame = original.copy()
12      randomizado = DataFrame(columns=frame.columns)
13      while len(frame) >= N:
14          smp = frame.sample(N)
15          frame.drop(smp.index,inplace=True)
16          dic = { col:[ item for row in smp[col] for item in row ] for col in smp.
        columns }
17          randomizado = randomizado.append(dic,ignore_index=True)
18      return randomizado
19
20  def _media_tags(processado,randomizado,tipo):
21      processado[tipo] = randomizado[tipo].apply(Counter)
22      for tag in tags[tipo]:
23          coluna = processado[tipo].apply(Counter.get,args=[tag,0])/processado['len
        ']
24          if not coluna.std() == 0:
25              processado[tag] = coluna
26      processado.drop(tipo,axis=1,inplace=True)
27
28  def process(randomizado,N):
29      processado = DataFrame()
30      processado['len'] = randomizado.tokens.str.len()/N
31      _media_tags(processado,randomizado,'POS')
32      _media_tags(processado,randomizado,'DET')
33      return processado
34
35  def exec(pack):
36      amostra,N = pack
37      dados = process(randomizador(amostra.dados,N),N)
38      tabelar(dados,amostra.autor,amostra.obra)
39
40  def tabelar(dados,autor,obra):
41      dados.to_csv('dados_processados/contagens-{}-{}.csv'.format(autor,obra))
42      dados.corr().to_csv('dados_processados/correlacoes-{}-{}.csv'.format(autor,
        obra))
43      tabela = dados.describe()
44      tabela.loc['ratio'] = tabela.loc['std']/tabela.loc['mean']
45      tabela.loc[['mean','std','ratio','count']].to_csv('dados_processados/
        descricao-{}-{}.csv'.format(autor,obra))
46
47  def main():
48      from amostras import from_file
49      from multiprocessing import Pool,cpu_count
50
51      N = N_sentencas_a_homogeneizar
52      amostras = ( from_file(path) for path in Path('dados/').glob('cicero-*') )
53      amostras = ( (amostra,N) for amostra in amostras if len(amostra) >= 2*N )
54      #o empacotamento (amostra,N) e para passar varios parametros para
        imap_unordered
55
56      with Pool(1) as pool:
57          list(pool.imap_unordered(exec,amostras))
58
59  if __name__ == '__main__': main()
```

A.4 tagselect.py

Seleciona as *tags* de POS a serem utilizadas em análises de χ^2 a partir de critérios definidos.

```
######################################################################
## ESTE SCRIPT SERVE APENAS PARA RETORNAR AS TUPLAS EM dic_dados ##
######################################################################

from pandas import read_csv
from tagsbank import tags
from pathlib import Path
from collections import namedtuple,Counter
T = namedtuple('Tupla',['path','ano','tabela'])
G = namedtuple('Genus',['arquivos','tags'])
N = namedtuple('Nome',['arquivo','ano'])

limiar_tab = 0.3                    #max ratio permitida em metade ou mais da tabela
limiar_tag = 0.3                    #max ratio permitida para tag em tabela passada pelo teste anterior
limiar_cor_tab = 0.5                #max corr permitida em metade ou mais de tabela passada pelo teste a frente
limiar_cor_tag = 0.5                #max correlacao permitida entre tags
inexiste = {'ratio':limiar_tag*999} #se tag nao existe em uma tabela, nao pode ser usada

generos = ['filosofia','oratoria','epistola']

def _condicao_dados(tabela):
    if len([ None for col in tabela.columns if tabela[col]['ratio'] >= limiar_tab ]) >= len(tabela.columns)/2:
        return False
    return True

def _condicao_tags(tag,dados):
    for dado in dados:
        if dado.tabela.get(tag,inexiste)['ratio'] >= limiar_tag: return False
    return True

def _condicao_tupla(tupla,dados):
    if len([ None for dado in dados if abs(dado.tabela[tupla[0]][tupla[1]]) >= limiar_cor_tag ]) >= len(dados)/2:
        return True
    return False

def _condicao_corr(tabela,matriz):
    if len([ None for tupla in matriz if (abs(tabela[tupla[0]][tupla[1]]) >= limiar_cor_tab) ]) >= len(matriz)/2:
        return False
    return True

def _arquivo(path):
    return path.name.replace('descricao-cicero-','').replace('csv','txt')

def _cond_arq(path,obras,genero):
    return obras.query('arquivo == @_arquivo(@path)').genero.iloc[0] == genero

#recebe como string 'filosofia', 'epistola' ou 'oratoria'
#seleciona dentre todos os arquivos os que correspondem ao genero da string
def banco_de_dados(genero):
    obras = read_csv('cicero.csv')
```

```
51      lista = Path('dados_processados/').glob('descricao-cicero-*')
52      return [ T(path,int(obras.query('arquivo == @_arquivo(@path)').ano.iloc[0]),
        read_csv(path,index_col=0)) for path in lista if _cond_arq(path,obras,genero)
        ]
53
54  def seletor(genero):
55      dados = banco_de_dados(genero)
56
57      #comecamos por olhar as variancias se estao dentro dos limiares
        parametrizados
58      #exclui tabelas com metade ou mais das ratio acima do limiar_tab
59      dados = [ dado for dado in dados if _condicao_dados(dado.tabela) ]
60
61      #exclui tags que, nas tabelas aprovadas, possuam ratio acima do limiar_tag
62      chaves = { key:None for key in tags.keys() }
63      for key in chaves.keys():
64          chaves[key] = [ tag for tag in tags[key] if _condicao_tags(tag,dados) ]
65
66      #agora vamos olhar as correlacoes
67      dados = [ T(dado.path,dado.ano,read_csv(str(dado.path).replace('descricao','
        correlacoes'),index_col=0)) for dado in dados ]
68      chaves = chaves['POS']#+chaves['DET'] #neste trabalho analisaremos apenas as
        tags POS
69      dados = [ T(dado.path,dado.ano,dado.tabela[chaves].loc[chaves]) for dado in
        dados ]
70      #apenas queremos olhar as correlacoes entre as tags que permanecem na analise
71
72      matriz = [ (tag1,tag2) for tag1 in chaves for tag2 in chaves ]
73      matriz = [ tupla for tupla in matriz if chaves.index(tupla[0]) < chaves.index
        (tupla[1]) ]
74
75      #limpa tags fortemente correlacionadas com outras tags segundo
        _condicao_tupla
76      suspeito = Counter([ tag for tupla in matriz if _condicao_tupla(tupla,dados)
        for tag in tupla ]).most_common(1)
77      while suspeito:
78          suspeito = suspeito[0][0]
79          chaves.remove(suspeito)
80          matriz = [ tupla for tupla in matriz if suspeito not in set(tupla) ]
81          suspeito = Counter([ tag for tupla in matriz if _condicao_tupla(tupla,
        dados) for tag in tupla ]).most_common(1)
82      del suspeito
83      dados = [ T(dado.path,dado.ano,dado.tabela[chaves].loc[chaves]) for dado in
        dados ]
84
85      #exclui tabelas com mais da metade das entradas (nao repetidas) com
        correlacao acima do limiar_cor
86      dados = [ N(dado.path,dado.ano) for dado in dados if _condicao_corr(dado.
        tabela,matriz) ]
87
88      return G(dados,set(chaves))
89
90  dic_dados = { genero:seletor(genero) for genero in generos }
```

A.5 chi_quadrado.py

Calcula as estatísticas referentes às distribuições de distribuições para cada gênero, e define um método para calcular o χ^2 de um texto (com dados fornecidos como *output* de `tagstats`) com respeito a um gênero.

```
from tagselect import generos,dic_dados
import matplotlib.pyplot as plt
from pandas import read_csv
import numpy as np
from collections import namedtuple
T = namedtuple('T',['ano','tabela','arquivo'])
D = namedtuple('D',['media','varmedia','varsimples'])

tags_gen = { genero:dic_dados[genero].tags for genero in generos }

valores = { genero:dic_dados[genero].arquivos for genero in generos  }
valores = { genero:[ T(arquivo.ano,read_csv(arquivo.arquivo,index_col=0)[tags_gen
    [genero].union({'len'})],arquivo.arquivo) for arquivo in valores[genero] ] for
     genero in generos }
#a informacao sobre ano sera importante no estudo da variacao das POS tags no
    tempo, mas nao aqui

def media(genero,tag):
    medias = np.array([ v.tabela[tag]['mean'] for v in valores[genero] ])
    variancias = np.array([ v.tabela[tag]['std']**2/v.tabela[tag]['count'] for v
    in valores[genero] ])
    pesos = 1/variancias
    return (medias*pesos).sum() / pesos.sum()

def var_media(genero,tag):
    variancias = np.array([ v.tabela[tag]['std']**2/v.tabela[tag]['count'] for v
    in valores[genero] ])
    return (1/variancias).sum()**-1

def var_simples(genero,tag):
    medias = np.array([ v.tabela[tag]['mean'] for v in valores[genero] ])
    return medias.var()*len(medias)/(len(medias)-1)
    #funcao np.var() fornece variancia populacional, precisamos da
    #variancia amostral, por isso o reescalonamento por n/(n-1)

statisticas = { genero:{ tag:D(media(genero,tag),var_media(genero,tag),
    var_simples(genero,tag)) for tag in tags_gen[genero] } for genero in generos }

#devolve qui-quadrado reduzido e incerteza
def chi(tabela,stats,tags):
    medias = np.array([ tabela[tag]['mean'] for tag in tags ])
    incertezas = np.array([ tabela[tag]['std']**2/tabela[tag]['count'] for tag in
     tags ])
    dados = np.array([ stats[tag].media for tag in tags ])
    variancias_media = np.array([ stats[tag].varmedia for tag in tags ])
    variancias_simples = np.array([ stats[tag].varsimples for tag in tags ])
    chi_quad = ( (medias - dados)**2 / (variancias_simples + variancias_media +
    incertezas) ).sum()
    gdl = len(tags)-1
    return chi_quad/gdl, np.sqrt(2/gdl)
```