Linguagens de Programação

no GitHub

Alisson Rosa

1 Introdução

Linguagens de programação nós permitem fornecer regras a quais os computadores entendem, ao longo do tempo foram surgindo inúmeras linguagens com propósitos e estruturas diferentes, aqui vamos analisar o comportamento dessas linguagens no GitHub (gh), que hoje é uma das principais plataformas para versionamento de códigos, assim vamos avaliar:

- Quantidade de *issues* em repositórios para as linguagens estudadas
- Quantidade de *pull requests* (prs) em repositórios que contém tais linguagens ao longo do tempo
- Quantidade totais de repositórios das linguagens.

Todo código a ser desenvolvido aqui será usando a linguagem Julia.

```
include("utils.jl")
using DataFrames
import CSV
using Plots
default(formatter=identity, tickfontsize=7, titlefontsize=12,
legend=:topleft)
#theme(:vibrant)
using Statistics
using Pipe: @pipe


pr = DataFrame(CSV.File("data/prs.csv"));
issues = DataFrame(CSV.File("data/issues.csv"));
total = DataFrame(CSV.File("data/repos.csv"));
```

Precisamos primeiro notar que existiam mais de 400 linguagens disponíveis no gh até 2021, assim vamos começar avaliando as top 5 linguagens e R[1] em quantidade de repositórios

Notamos que as top 3 linguagens são de desenvolvimento web, sendo elas JavaScript, CSS, HTML e no top 4 está o famoso Shell Script, e por último Python. No seguinte gráfico vamos ver como se comporta R em frente a esses gigantes.

[1] R, pois certas pessoas curtem R, certo.. Certo??

Linguagem	Total de Repositórios
JavaScript	1100421
CSS	813443
HTML	779549
Shell	638068
Python	548870

Tabela 1: Top 5 Linguagens em quantidade de repositórios.

```
@pipe total |>
      sort(_, order(:num_repos, rev=true)) |>
      first(_, 5) |>
      select(_, :language => :Linguagem, :num_repos => :N) |>
      df -> bar(df.Linguagem, df.N, fill=["#f3ff33", "#337aff",
                "#ff5e33", "#55ff33", "#233cad"],
            series_annotation=df.N, legend=false)
@pipe  total |>
      filter(:language =>  ==("R"), _) |>
       df -> bar!(df[:,:language] , df[:,:num_repos], legend=false)
```

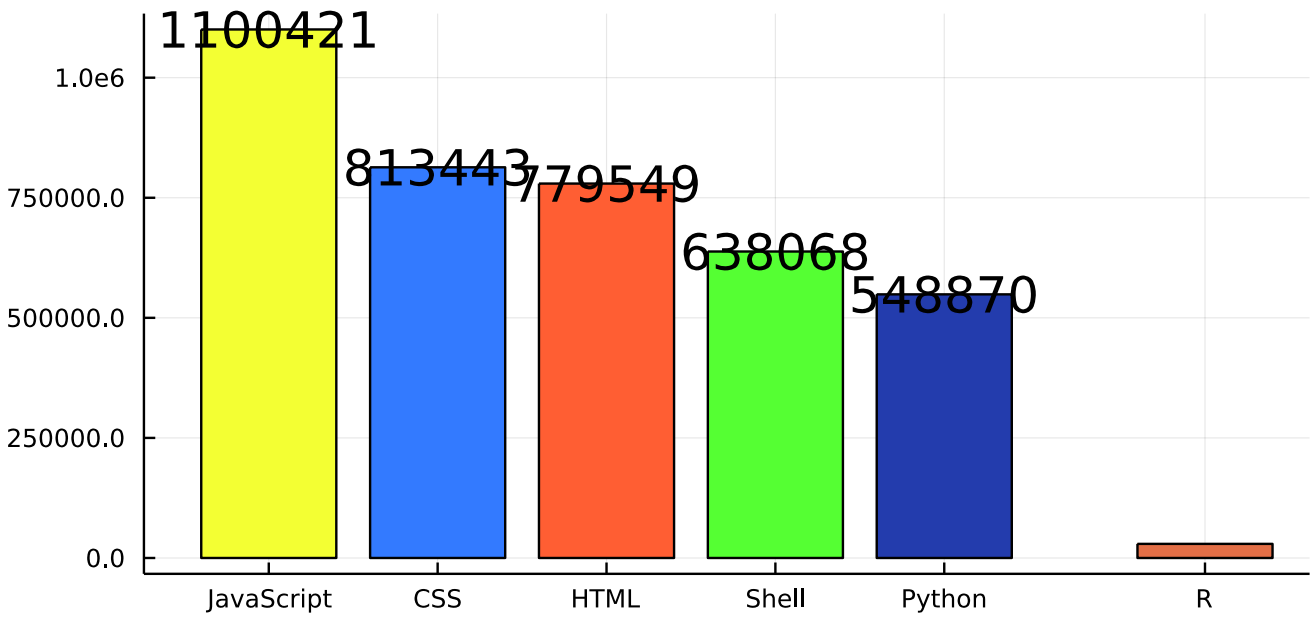


Figura 1: Top 5 Linguagens em quantidade de repositórios e R

Por honestidade, devemos primeiro notar que R é uma linguagem nichada, portanto nunca foi esperado que fosse ter quantidade de repositórios próxima as linguagens anteriormente citadas, porém vale a título de curiosidade.

Seguindo a mesma ideia, vamos analisar as top 5 linguagens que possuem mais prs e issues ao total. Pela Figura 2 notamos que o top 3 se mantém o mesmo tanto pr e issues, entretanto para prs PHP está no top

5 e em issues está no top 4, outra diferença é que Ruby aparece em prs ao invés de C++ que aparece nas issues.

```
p1 = barh(pr, "#5310c8")
p2 = barh(issues, "#0b08c0")

plot(p1,p2, layout=2, title=["Pr" "Issues"])
```

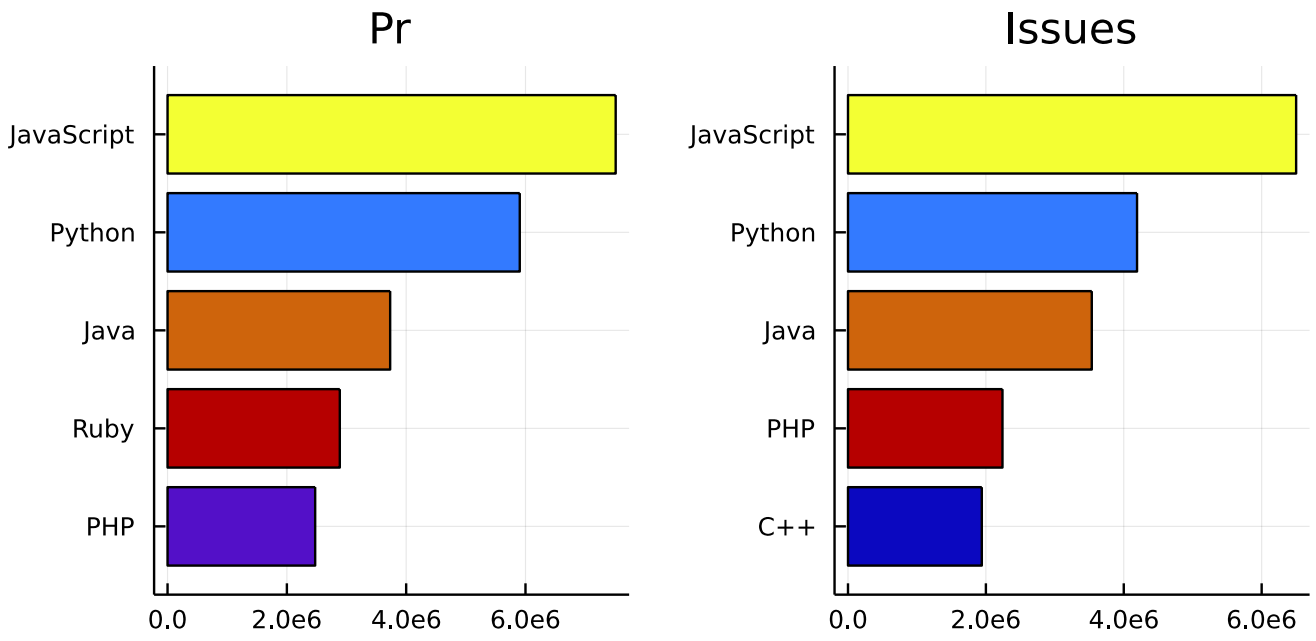


Figura 2: Comportamento médio ao longo dos anos

2 Análise descritiva

É essencial começar avaliando o comportamento médio de prs e issues no gh ao longo dos anos. Notamos alguns pontos fundamentais, tanto prs e issues crescem até atingir seu ápice em 2017, mas os prs diferentes das issues não descressem de forma monótona depois de 2017.

```
p2 = @pipe top_5(pr,:year)  |>
           areaplot(_[:,:year],_[:,:media], title="pr", color = "#ff4933",
           label=false, legend=:topleft)
p1 = @pipe top_5(issues,:year) |>
           areaplot(_[:,:year],_[:,:media], title="issues", label=false, legend=:topleft)

plot(p1,p2,layout= (1,2))
```

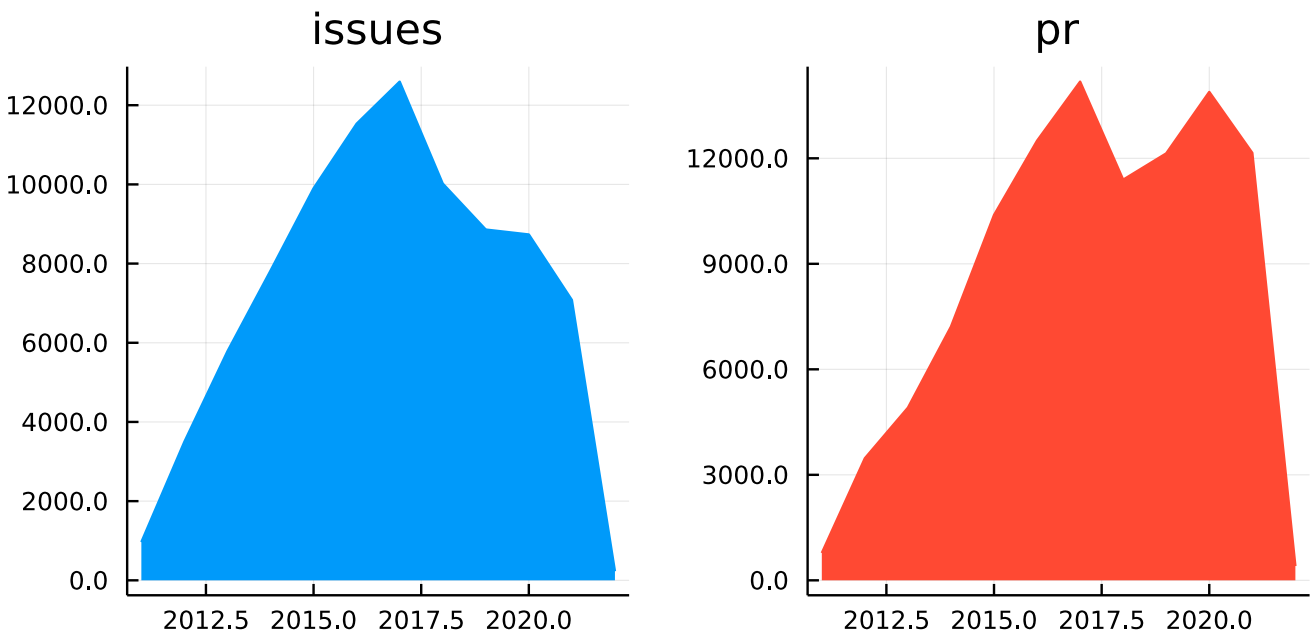


Figura 3: Comportamento médio ao longo dos anos

Mas na Figura 3, avaliamos o comportamento das issues e prs no gh de forma global, independente da linguagem, será que as linguagens individualmente se comportam da mesma maneira? É isso que veremos nas seções seguintes

3 Linguagens Globalmente

Aqui vamos avaliar o comportamento de 5 linguagens: Java, JavaScript, PHP, Python e Ruby. Pela Figura 4, podemos ter um vislumbre do comportamento de prs para essas linguagens ao longo do tempo

```
fav_langs(pr, ["Java","Python", "JavaScript", "Ruby", "PHP"]) |>
      df -> plot(df[:,:year], df[:,:media], g = df[:,:name])
```

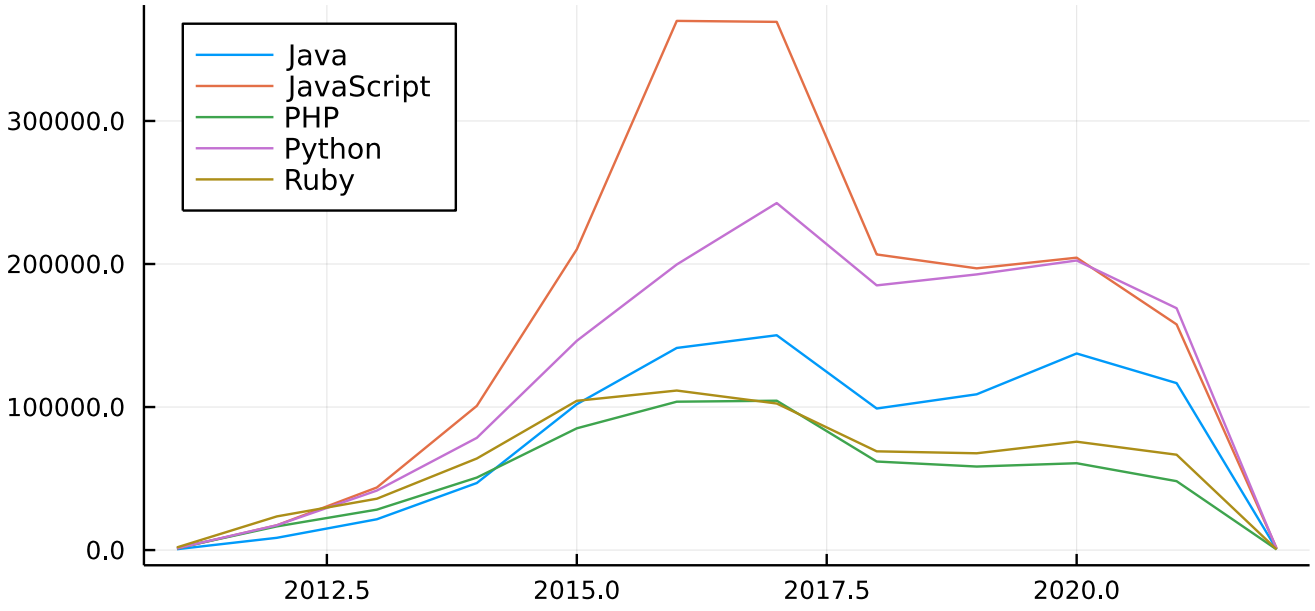


Figura 4: Comportamento médio de prs

Note que JavaScript domina desde o começo, porém Python avançou muito nos últimos anos, e em 2021 finalmente ultrapassou JavaScript, outra linguagem que andou tendo um decréscimo considerável é PHP. Vale ressaltar o pico de todas as linguagens em 2017, e depois um levo decréscimo, mas somente PHP e Ruby continuaram caindo consideravelmente nos últimos anos. Pela Figura 5, nota-se que Ruby tem um comportamento diferente em termos de issues que em prs, pois fica abaixo de todas as linguagens praticamente em todos os anos, Java para issues está mais próximo de Python do que estava em termos de prs, e Java depois de 2017 só decresce em issues, algo que não aconteceu em prs.

```
fav_langs(issues, ["Java","Python", "JavaScript", "Ruby", "PHP"]) |>
      df -> plot(df[:,:year], df[:,:media], g = df[:,:name])
```

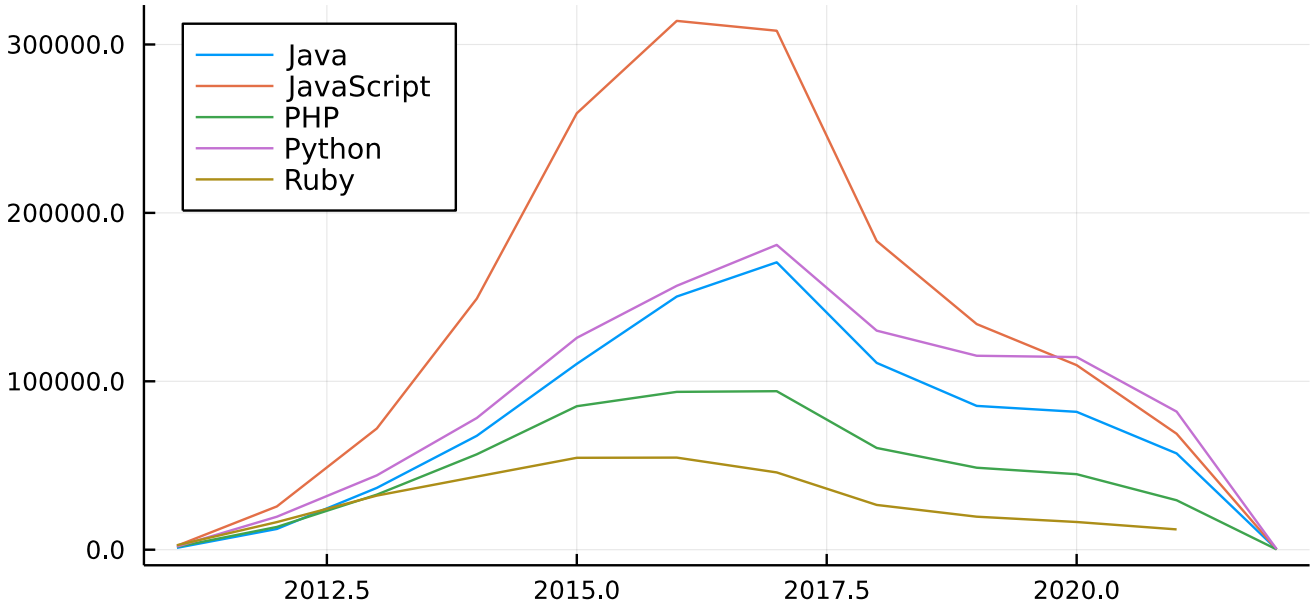


Figura 5: Comportamento médio de issues

Notamos portanto, um comportamento bem claro, prs cresceram até atingir seu apogeu em 2017, aonde depois, em geral tendem a cair, as únicas linguagens das estudadas que recuperaram crescimento foram Java e Python. Como vimos antes na análise global, as issues somente decrescem depois de 2017 e o mesmo aconteceu individualmente para as linguagens estudadas aqui.

4 Linguagens para Dados

Nessa seção vamos avaliar o comportamento especificamente de R e Julia, que são duas das três principais linguagens para trabalhar com dados no momento. Vimos que issues decrescem a partir de 2017 monotonicamente de forma geral e também para as linguagens estudadas nessa subseção, a fig portanto nos informa quais as 5 linguagens mantiveram mais anos de crescimento após 2017

```
lagged = @pipe issues[issues.year.>2016, :] |>
               groupby(_, [:year, :name]) |>
               combine(_, :count => mean) |>
               groupby(_, :name) |>
               combine(_, :count_mean => (lag => :count));
lagged[:, :count] =  ifelse.(lagged[:, :count] .> 0, 1, 0);
top_5_count(lagged, :name)
```

	name	total
	String31	Float64
1	ColdFusion	3.0
2	Dart	3.0
3	Nunjucks	3.0
4	ActionScript	2.0
5	Go	2.0

Das 5 linguagens 3 tiveram 3 anos de crescimento de issues se comparado ao ano anterior, ou seja: nenhuma delas manteou-se crescente desde o começo.

4.1 Breve Introdução

R é uma implementação open-source da Linguagem S, teve seu lançamento oficial em 1995, sendo desenvolvida por Ross Ihaka e Robert Gentleman, sendo assim seu nome "R" vem da letra inicial do seus criadores e também do dialeto com "S". R tem suas raízes na estatística computacional, passou seus 10 primeiros anos sendo primariamente uma linguagem acadêmica, mas depois do desenvolvimento de pacotes como `tidyverse` se tornou um ambiente propicio para ciência de dados.

O projeto da linguagem Julia teve seu início em meados de 2009, muitos dos seus criadores usavam certas ferramentas que funcionavam para tarefas especificas mas em outras eram terríveis, portanto Julia surge sendo uma linguagem que deveria aglomerar tudo que certas linguagens faziam de bom, como ser rápida como C, ser boa em estatística como o R, ser de proposito geral como Python, ser poderosa em algebra linear como Matlab, e repetindo ser **rápida** como C, sendo assim, foi lançada oficialmente em 2012, no momento que o autor vos escreve, Julia está fazendo 10 anos, mesmo com somente 10 anos de idade Julia já é usada pela NASA e também pelo INPE.
Vamos começar avaliando a quantidade total de repositórios para cada uma das duas linguagens.

```
total[findall(in(["R", "Julia"]), total[:, :language]), :] |>
df -> bar(df[:, :language], df[:, :num_repos], g=df[:, :language],
        fill=["#2B5FE9", "#AA30FF"])
```

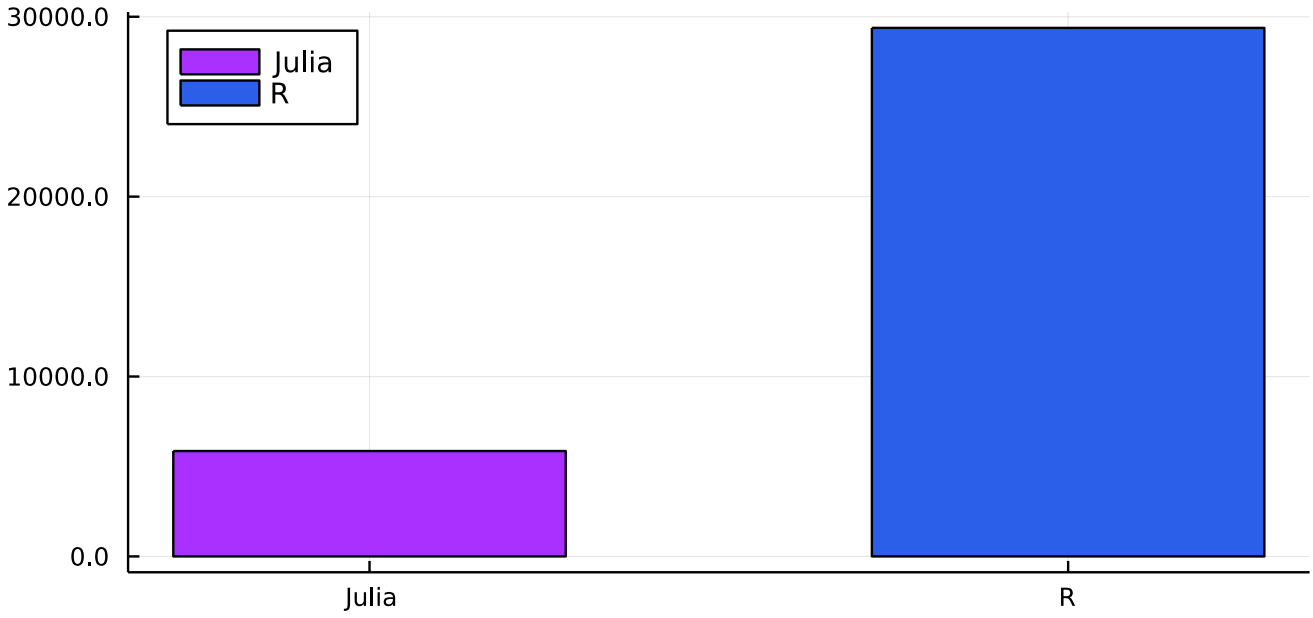


Figura 6: Quantidade de repositórios disponíveis para R e Julia

Podemos ver que R possui muito mais repositórios disponíveis no gh, porém R é uma linguagem que está a bastante tempo no mercado, ao contrário de Julia. Como dito anteriormente, R nos seus primeiros anos foi uma linguagem primariamente acadêmica, isso significa que a maioria dos seus repositórios são para ferramentas para a própria linguagem, e é nessa fase que Julia se situa atualmente.

Agora vamos ver o comportamento das duas linguagens ao longo dos últimos anos

```
p1 = fav_langs(pr, ["R","Julia"]) |>
      df -> plot(df[:,:year], df[:,:media], g = df[:,:name], title = "Pull Requests")
p2 = fav_langs(issues, ["R","Julia"]) |>
      df -> plot(df[:,:year], df[:,:media], g = df[:,:name], title = "Issues")

plot(p1,p2,layout=(1,2))
```

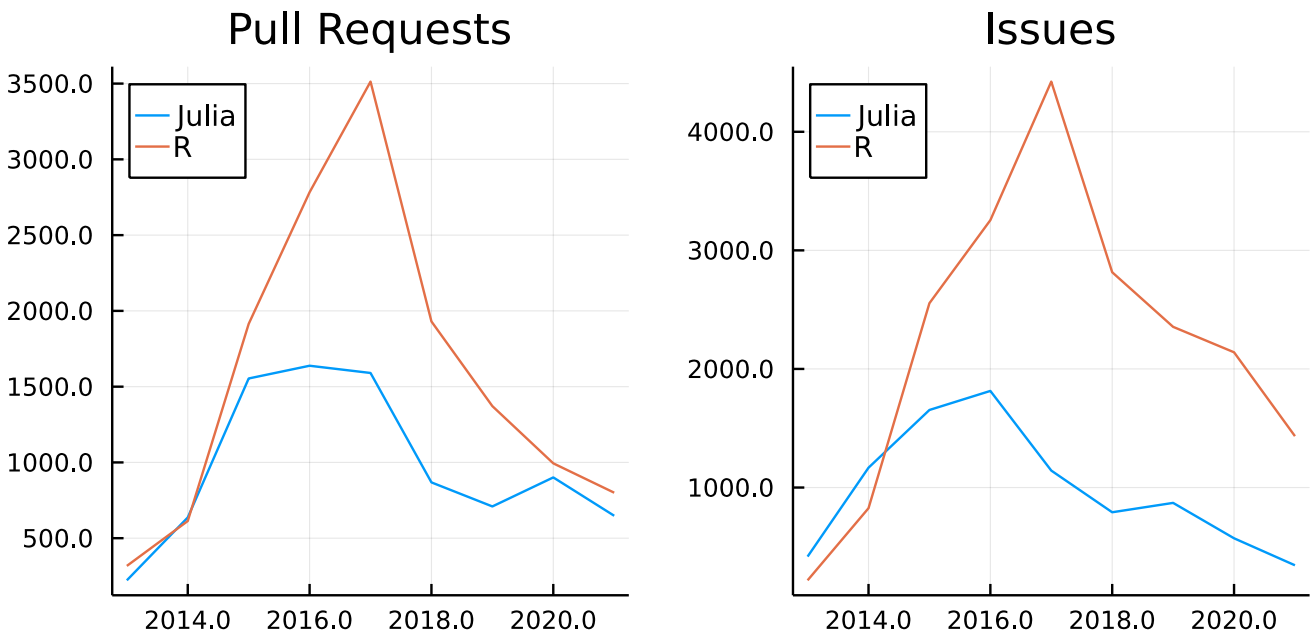


Figura 7: Média de prs e issues para R e Julia

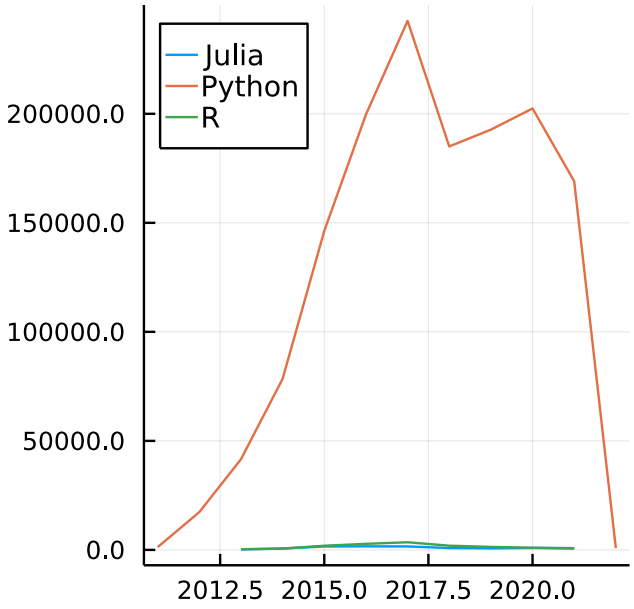
Nota-se que em 2014 Julia teve mais prs e issues que R e ambas as linguagens atingem seu pico em 2017, um comportamento global como vimos anteriormente, porém para issues Julia teve mais de 2012 até 2014 e também seu pico de issues foi em 2016 e não 2017 como R.

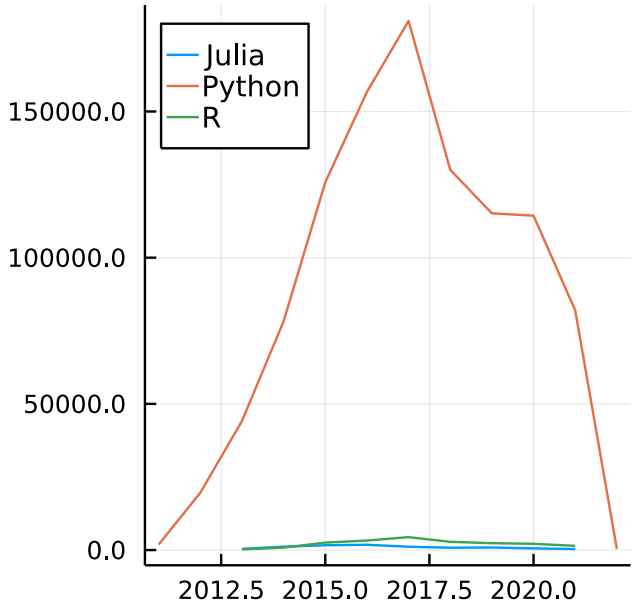
4.2 Python entra no game

Evidentemente não podemos esquecer de Python, pois como vimos está no top 5 de linguagens com mais repositórios disponíveis, atualmente conta com vários pacotes para análise de dados, e é a principal linguagem para deep learning, então é claro que não poderia faltar a comparação com suas concorrentes nesse ramo. Notamos que há uma diferença desproporcional em termos de prs e issues a comparação com R e Julia.

```
p1 = fav_langs(pr, ["R","Julia","Python"]) |>
      df -> plot(df[:,:year], df[:,:media], g = df[:,:name])
p2 = fav_langs(issues, ["R","Julia", "Python"]) |>
      df -> plot(df[:,:year], df[:,:media], g = df[:,:name])

plot(p1,p2,layout=(1,2))
```





Sabemos que Python é de proposito geral e uma linguagem bem estabelecida, então evidentemente tende a ocupar mais espaço que R e Julia. Um ponto fundamental, é que se filtrarmos em repositórios com vies de análise de dados/ machine learning, será que Python tende a manter essa distância das outras linguagens? Infelizmente tal filtragem seria puramente arbitrária, pois como definimos o que é um repositório de análise de dados? Qualquer maneira de definir pode facilmente não incluir pontos relevantes, assim a resposta para a pergunta anterior fica a cargo do leitor.

5 Conclusão momentânea

Vimos que JavaScript leva vantagem de todas as linguagens em todos os anos, e também fica claro a discrepância que JavaScript tem das outras linguagens, pois pela Figura 3 vemos que o valor médio geral para pr e issues é muito inferior que o de JavaScript, Python vem brigando pela superioridade com JavaScript ao longo dos anos, sendo que em 2021 o superou nas métricas estudadas aqui. Vimos também que Julia é uma linguagem nova e em ascensão, pois atualmente ela está na fase em que R se encontrava a 10 anos atrás, porém já possuindo algumas aplicações no mercado. Python tem muito destaque sobre as outras linguagens que tem viés para análise de dados, mas vale ressaltar que python não nasceu com esse propósito.
Será que o que observamos aqui vai se manter daqui 10 anos? Será que a linguagem Julia tem potencial para superar R e Python? Python vai continuar em ascensão? São muitas perguntas e no momento poucas respostas, só o tempo e empenho das comunidades trarão as respostas desejadas.