



Yermack 15 апреля 2019 в 07:17

# Джулия в латексе

Программирование, LaTeX, Julia

Tutorial



В научной среде очень важную роль играет визуализация данных и оформление теории. Для удобного и красивого представления формул часто используются инструменты реализующие *LaTeX*-команды, например Markdown и MathJax.

Для Джулии также существует набор пакетов позволяющих использовать синтаксис *LaTeX* 'а, а в связке с средствами символьной алгебры мы получаем мощный инструмент для оперирования формулами.

## Скачиваем и подключаем всё что нужно на сегодня

```
using Pkg
Pkg.add("Latexify")
Pkg.add("LaTeXStrings")
Pkg.add("SymEngine")

using Latexify, LaTeXStrings, Plots, SymEngine
```

## LaTeXStrings

LaTeXStrings небольшой пакет, облегчающий ввод уравнений LaTeX в строковых литералах на языке Julia. При использовании обычных строк в Julia для ввода строкового литерала со встроенными уравнениями LaTeX необходимо вручную избегать всех обратных слешей и знаков доллара: например,  $\alpha^2$  пишется `\\alpha^2\\$`. Кроме того, хотя Julia способна отображать отформатированные уравнения LaTeX (через *MathJax*), с обычными строками такое не работает. Поэтому, пакет LaTeXStrings определяет:

- Класс `LaTeXString` (подтип `String`), который работает как строка (для индексации, преобразования и т. д.), Но автоматически отображается как текст / латекс в `Julia`.
- Строковые макросы `L"..."` и `L"..."..."`, которые позволяют вводить уравнения LaTeX без экранирования от обратной косой

Все потоки   Разработка   Администрирование   Дизайн   Менеджмент   Маркетинг   Научпоп



Войти

Регистрация

```
S = L"1 + \alpha^2"
```

В **REPL** выйдет:

```
"\$1 + \\alpha^2\$"
```

а **Jupyter** отобразит:

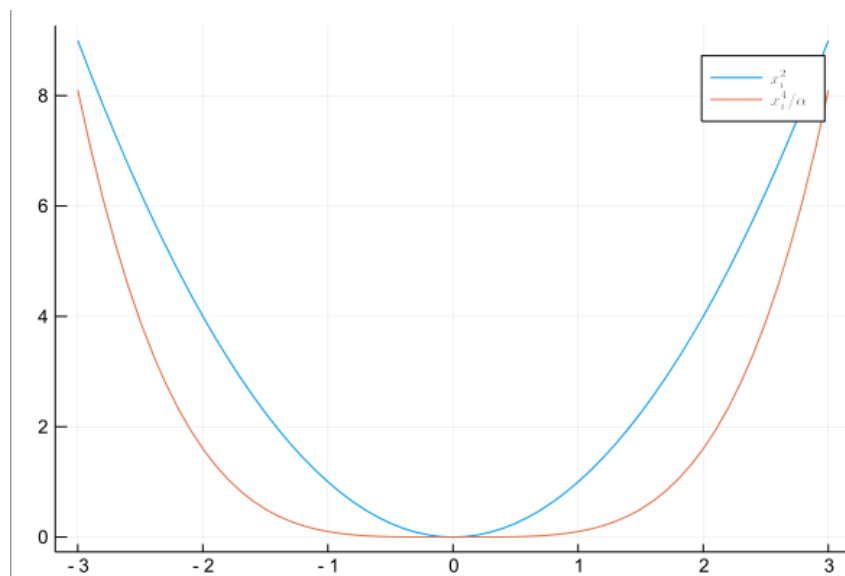
$$1 + \alpha^2$$

Индексация работает как с обычными строками:

```
S[4:7]  
"+ \\a"
```

Такие строки могут быть полезны при оформлении графиков

```
x = [-3:0.1:3...]  
y1 = x.^2  
α = 10  
y2 = x.^4 / α;  
  
plot(x,y1, lab = "\$x^2_i\$")  
plot!(x,y2, lab = L"x^4_i/\alpha")
```



# Latexify

Более функциональным пакетом является [Latexify \(Руководство\)](#). Он предназначен для генерации математики LaTeX из объектов Julia. Этот пакет использует гомологичность Джулии для преобразования выражений в строки в формате LaTeX. *Latexify.jl* предоставляет функции для преобразования ряда различных объектов Julia, в том числе:

- Выражения,
- Строки,
- Numbers (включая рациональные и комплексные),
- Символические выражения из SymEngine.jl,
- ParameterizedFunctions и ReactionNetworks из DifferentialEquations.jl  
а также массивы всех поддерживаемых типов.

```
ex = :(x/(y+x)^2) # выражение
latexify(ex)
```

$$\frac{x}{(y+x)^2}$$

```
str = "x/(2*k_1+x^2)" # строка
latexify(str)
```

$$\frac{x}{2 \cdot k_1 + x^2}$$

Массив разнотипных элементов:

```
m = [2//3 "e^(-c*t)" 1+3im; :(x/(x+k_1)) "gamma(n)" :(log10(x))]
latexify(m)
```

$$\begin{bmatrix} \frac{2}{3} & e^{-c \cdot t} & 1 + 3i \\ \frac{x}{x+k_1} & \Gamma(n) & \log_{10}(x) \end{bmatrix}$$

Можно задать функцию выводящую формулы и копирующую их в буфер в виде понятном для Хабра:

```
function habr(formula)
    l = latexify(formula)
    res = "\${$display}\${l}$display\${$}\n"
    clipboard(res)
    return l
end

habr(ex)
```

$$\frac{x}{(y+x)^2}$$

```
<p>$<!-- math>$$display$$\frac{x}{\left( y + x \right)^2}$$display$$</math -->$</p>
```

Следует иметь ввиду

```
latexify("x/y") |> display
```

$$\frac{x}{y}$$

```
latexify("x/y") |> print
```

```
$\frac{x}{y}$
```

## SymEngine

SymEngine — пакет предоставляющий символьные вычисления, которые можно визуализировать в вашем Jupyter с помощью Latexify.

Можно задавать символы строками и котировками (quote):

```
julia> a=symbols(:a); b=symbols(:b)
b

julia> a,b = symbols("a b")
(a, b)

julia> @vars a b
(a, b)
```

Зададим матрицу и красиво отобразим ее

```
u = [symbols("u_{$i$}") for i in 1:3, j in 1:3]

3×3 Array{Basic,2}:
u_11  u_12  u_13
u_21  u_22  u_23
u_31  u_32  u_33

u |> habr
```

$$\begin{bmatrix} u_{11} & u_{12} & u_{13} \\ u_{21} & u_{22} & u_{23} \\ u_{31} & u_{32} & u_{33} \end{bmatrix}$$

Предположим у нас вектора

```
C = symbols("Q_b/Q_1")
J = [symbols("J_$i") for i in ['x','y','z']]
h = [0, 0, symbols("h_z")]

3-element Array{Basic,1}:
 0
 0
 h_z
```

которые надо векторно помножить

```
using LinearAlgebra
x = cross

latexify(J×h, transpose = true)
```

$$\begin{bmatrix} J_y \cdot h_z \\ -J_x \cdot h_z \\ 0 \end{bmatrix}$$

Полноценные матричные вычисления:

```
dJ = C*(u*J.^3)×h
latexify( dJ, transpose = true)
habr(ans)
```

$$\begin{bmatrix} \frac{h_z \cdot (u_{21} \cdot J_x^3 + u_{22} \cdot J_y^3 + u_{23} \cdot J_z^3) \cdot \Omega_b}{\Omega_l} \\ \frac{-h_z \cdot (u_{11} \cdot J_x^3 + u_{12} \cdot J_y^3 + u_{13} \cdot J_z^3) \cdot \Omega_b}{\Omega_l} \\ 0 \end{bmatrix}$$

а вот такой нехитрой цепочкой можно найти детерминант и отправить его на Хабр

```
u |> det |> habr
```

$$u_{11} \cdot \left( u_{22} - \frac{u_{12} \cdot u_{21}}{u_{11}} \right) \cdot \left( u_{33} - \frac{u_{13} \cdot u_{31}}{u_{11}} - \frac{\left( u_{23} - \frac{u_{13} \cdot u_{21}}{u_{11}} \right) \cdot \left( u_{32} - \frac{u_{12} \cdot u_{31}}{u_{11}} \right)}{u_{22} - \frac{u_{12} \cdot u_{21}}{u_{11}}} \right)$$

Рекурсивненько! Обратная матрица наверно посчитается сходным образом:

```
u^-1 |> habr
```

Спойлер

Если хотите сделать больно Матжаксу своего браузера, поставьте минус вторую степень (квадрат обратной матрицы)  
Кстати, SymEngine считает производные:

```
dJ[1] |> habr
```

$$\frac{h_z \cdot (u_{21} \cdot J_x^3 + u_{22} \cdot J_y^3 + u_{23} \cdot J_z^3) \cdot \Omega_b}{\Omega_l}$$

```
diff(dJ[1], J[1]) |> habr
```

$$\frac{3 \cdot J_x^2 \cdot h_z \cdot u_{21} \cdot \Omega_b}{\Omega_l}$$

К слову, Джулия может использовать из LaTeX 'а не только формулы, но и графики. И если вы установили *MikTex* и уже скачали *pgfplots*, то с помощью соответствующего окружения его можно сдружить с Джулией, что предоставит возможность строить гистограммы, трехмерные графики, ошибки и рельефы с изолиниями, а потом это интегрировать в LaTeX документ.

На этом с формулами всё, но не с символьными вычислениями: у Джулии еще есть более сложные и интересные решения для символьной алгебры, с которыми мы обязательно разберемся как-нибудь в следующий раз.

Теги: julia, latex, символьные вычисления, матричные операции, формулы

Хабы: Программирование, LaTeX, Julia


 +18



 37

 7,1k

 4



82,7

Карма

2,6

Рейтинг

@Yermack

Пользователь

ПОХОЖИЕ ПУБЛИКАЦИИ

27 сентября 2018 в 08:59  
Символьные вычисления средствами Python. Часть1. Основы

 +23

 31,9k

 143

 12

7 мая 2017 в 15:37  
Профилирование и оптимизация символьных вычислений для будущего сервера

 +6

 3,5k

 25

 12

4 октября 2014 в 22:42  
О вычислении матричной экспоненты

## ВАКАНСИИ

### Golang Backend Developer

от 170 000 до 250 000 Р • ikitlab • Можно удаленно

### Middle | High middle front-end разработчик (React + Typescript)

от 80 000 до 160 000 Р • CSSSR • Можно удаленно

### Программист C# (Senior)

от 180 000 Р • ГК «Системные Технологии» • Калининград • Можно удаленно

### Embedded C Разработчик / Embedded C Software Developer

от 100 000 Р • Flipper Devices Inc. • Москва • Можно удаленно

### Разработка программного обеспечения беспилотных комплексов, C++

от 100 000 до 150 000 Р • Группа «Кронштадт» • Москва

Больше вакансий на Хабр Карьере

## Комментарии 4



**titulusdesiderio** 15 апреля 2019 в 11:25 # 1

↑ +4 ↓

С кликбейтностью заголовка и КДПВ вы не прогадали



**abar** 15 апреля 2019 в 13:18 # 1

↑ 0 ↓

Как раз недавно читал описание языка и задался вопросом о том, пользуется ли кто Джулией вообще.

У нас в университете всё, что видел, было на питоне, в джупитере тот же питон, профессора тоже требуют владение Python/R/C++.  
Нужен ли ещё один язык, если огромный пласт библиотек написан на других языках?



**Yermack** 15 апреля 2019 в 15:57 # 1 2 3 4

↑ 0 ↓

В первую очередь язык создавался для ученых, которым нужно удобство работы с данными и быстрые вычисления. Когда язык начал разрабатываться, питон только становился на ноги в научной среде. У нас же язык непопулярен из-за молодости (речь об окружении), малого количества материалов и инертности целевой аудитории. [juliacomputing.com/case-studies](http://juliacomputing.com/case-studies) — здесь можете посмотреть про серьезные применения.



**technic93** 16 апреля 2019 в 00:39 # 1 2 3 4

↑ 0 ↓

Из статей на хабре про Юлию, она выглядит как питон, но только говорят что быстрый. Код на первый взгляд менее выразительный, хотя может проблема в слабой подсветке синтаксиса хабром. Вроде скрины Juipo выглядят более читабельно.

Только полноправные пользователи могут оставлять комментарии. Войдите, пожалуйста.

САМОЕ ЧИТАЕМОЕ

Сутки

Неделя

Месяц

Налоговая выдаст ЭЦП бесплатно для ИП и юрлиц

+24 34,2k 11 82

У Google появился новый креативный способ убивать SaaS-стартапы

+142 33,4k 85 92

Как одним движением сжечь 10000\$ и получить удар током

+76 20,4k 83 78

Почему я не покупаю новый ноутбук, а работаю на Sony Vaio семейства SVE с 2013 года

+26 17,9k 34 81

IT для тру металлистов: что можно сделать кодом на тяжелом производстве

Мегапост

Ваш аккаунт

Войти

Регистрация

Разделы

Публикации

Новости

Хабы

Компании

Пользователи

Песочница

Информация

Устройство сайта

Для авторов

Для компаний

Документы

Соглашение

Конфиденциальность

Услуги

Реклама

Тарифы

Контент

Семинары

Мегaproекты

Мерч