Package ‘diffeqr’

March 18, 2024

Type   Package
Title  Solving Differential Equations (ODEs, SDEs, DDEs, DAEs)
Version 2.0.1
Description  An interface to 'DifferentialEquations.jl' <https://diffeq.sciml.ai/dev/> from the R programming language. It has unique high performance methods for solving ordinary differential equations (ODE), stochastic differential equations (SDE), delay differential equations (DDE), differential-algebraic equations (DAE), and more. Much of the functionality, including features like adaptive time stepping in SDEs, are unique and allow for multiple orders of magnitude speedup over more common methods. Supports GPUs, with support for CUDA (NVIDIA), AMD GPUs, Intel oneAPI GPUs, and Apple's Metal (M-series chip GPUs). 'diffeqr' attaches an R interface onto the package, allowing seamless use of this tooling by R users. For more information, see Rackauckas and Nie (2017) <doi:10.5334/jors.151>.

Depends  R (>= 3.4.0)
Encoding UTF-8
License MIT + file LICENSE
URL  https://github.com/SciML/diffeqr
SystemRequirements  Julia (>= 1.6), DifferentialEquations.jl, ModelingToolkit.jl
Imports  JuliaCall
RoxygenNote  7.1.1
Suggests  testthat, knitr, rmarkdown
VignetteBuilder  knitr
NeedsCompilation no
Author  Christopher Rackauckas [aut, cre, cph]
Maintainer  Christopher Rackauckas <me@chrisrackauckas.com>
Repository CRAN
Date/Publication  2024-03-18 11:50:02 UTC


R topics documented:

```
diffeqgpu_setup                   2
diffeq_setup                     3
jitoptimize_ode                  3
jitoptimize_sde                  4
```

Index 6

---

**diffeqgpu_setup**  
*Setup DiffEqGPU*

**Description**

This function initializes the DiffEqGPU package for GPU-parallelized ensembles. The first time will be long since it includes precompilation.

**Usage**

```
diffeqgpu_setup(backend)
```

**Arguments**

- `backend`  
  the backend for the GPU computation. Choices are "CUDA", "AMDGPU", "Metal", or "oneAPI"

**Value**

Returns a `degpu` object which holds the module state of the Julia-side DiffEqGPU package. The core use is to use `degpu$EnsembleGPUKernel()` for choosing the GPU dispatch in the solve.

**Examples**

```
## Not run:  ## diffeqgpu_setup() is time-consuming and requires Julia+DifferentialEquations.jl
degpu <- diffeqr::diffeqgpu_setup(backend)

## End(Not run)
```
diffeq_setup  

Setup diffeqr

Description
This function initializes Julia and the DifferentialEquations.jl package. The first time will be long since it includes precompilation. Additionally, this will install Julia and the required packages if they are missing.

Usage

diffeq_setup(pkg_check = TRUE, ...)

Arguments

pkg_check logical, check for DifferentialEquations.jl package and install if necessary

... Parameters are passed down to JuliaCall::julia_setup

Value

Returns the de object which gives R-side calls to DifferentialEquations.jl’s functions. For example, de$solve calls the DifferentialEquations.solve function, and de$ODEProblem calls the DifferentialEquations.

Examples

## Not run: ## diffeq_setup() is time-consuming and requires Julia+DifferentialEquations.jl
diffeqr::diffeq_setup()

## End(Not run)

jitoptimize_ode  

Jit Optimize an ODEProblem

Description
This function JIT Optimizes and ODEProblem utilizing the Julia ModelingToolkit and JIT compiler.

Usage

jitoptimize_ode(de, prob)
**Arguments**

- `de`: the current diffeqr environment
- `prob`: an ODEProblem

**Value**

Returns an ODEProblem which has been JIT-optimized by Julia.

**Examples**

```r
## Not run: ## diffeq_setup() is time-consuming and requires Julia+DifferentialEquations.jl
de <- diffeqr::diffeq_setup()
f <- function(u, p, t) {
  du1 = p[1]*(u[2]-u[1])
  du2 = u[1]*(p[2]-u[3]) - u[2]
  return(c(du1, du2, du3))
}
u0 <- c(1.0, 0.0, 0.0)
tspan <- c(0.0, 100.0)
p <- c(10.0, 28.0, 8/3)
prob <- de$ODEProblem(f, u0, tspan, p)
fastprob <- diffeqr::jitoptimize_ode(de, prob)
sol <- de$solve(fastprob, de$Tsit5())
```

**Description**

This function JIT Optimizes and SDEProblem utilizing the Julia ModelingToolkit and JIT compiler.

**Usage**

```r
jitoptimize_sde(de, prob)
```

**Arguments**

- `de`: the current diffeqr environment
- `prob`: an SDEProblem

**Value**

Returns an SDEProblem which has been JIT-optimized by Julia.
Examples

## Not run: ## diffeq_setup() is time-consuming and requires Julia+DifferentialEquations.jl

diffeqr::diffeq_setup()

## End(Not run)
Index

diffeq_setup, 3
diffeqgpu_setup, 2

jitoptimize_ode, 3
jitoptimize_sde, 4