Jump process using DiffEqBiological

Simon Frost (@sdwfrost), 2020-05-11

Introduction

This implementation defines the model as a combination of two jump processes, infection and recovery, simulated using the [Doob-Gillespie algorithm](#).

Libraries

```
using DiffEqBiological
```

```
Error: ArgumentError: Package DiffEqBiological not found in current path:
- Run `import Pkg; Pkg.add("DiffEqBiological")` to install the DiffEqBiolog
ical package.
```

```
using Random
using DataFrames
using StatsPlots
using BenchmarkTools
```

Transitions

```
sir_model = @reaction_network sir_rn begin
  0.5/1000, s + i --> 2i
  0.25, i --> r
end
```

```
Error: LoadError: UndefVarError: @reaction_network not defined
in expression starting at none:1
```

Time domain

```
tmax = 40.0
tspan = (0.0,tmax);
```

```
(0.0, 40.0)
```

For plotting, we can also define a separate time series.

```
δt = 0.1
t = 0:δt:tmax;
```

```
0.0:0.1:40.0
```

Initial conditions

```
u0 = [990,10,0]; # S,I,R
```

```
3-element Array{Int64,1}:
 990
  10
   0
```

Random number seed

We set a random number seed for reproducibility.

```
Random.seed!(1234);
```

```
MersenneTwister(UInt32[0x000004d2], Random.DSFMT.DSFMT_state(Int32[-1393240
018, 1073611148, 45497681, 1072875908, 436273599, 1073674613, -2043716458,
1073445557, -254908435, 1072827086  ...  -599655111, 1073144102, 367655457, 1
072985259, -1278750689, 1018350124, -597141475, 249849711, 382, 0]), [0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0  ...  0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0], UInt128[0x00000000000000000000000000000000, 0x00000
000000000000000000000000000, 0x00000000000000000000000000000000, 0x00000000
000000000000000000000000, 0x00000000000000000000000000000000, 0x00000000000
000000000000000000000, 0x00000000000000000000000000000000, 0x00000000000000
00000000000000000000, 0x00000000000000000000000000000000, 0x00000000000000000
000000000000000  ...  0x00000000000000000000000000000000, 0x00000000000000000
000000000000000, 0x00000000000000000000000000000000, 0x00000000000000000000
000000000000, 0x00000000000000000000000000000000, 0x00000000000000000000000
000000000, 0x00000000000000000000000000000000, 0x00000000000000000000000000
000000, 0x00000000000000000000000000000000, 0x00000000000000000000000000000
000], 1002, 0)
```

Running the model

Running this model involves:

- Setting up the problem as a `DiscreteProblem`;

- Adding the jumps and setting the algorithm using `JumpProblem`; and

- Running the model, specifying `SSAStepper`

```
prob = DiscreteProblem(u0,tspan)
```

```
Error: UndefVarError: DiscreteProblem not defined
```

```
prob_jump = JumpProblem(prob,Direct(),sir_model)
```

```
Error: UndefVarError: Direct not defined
```

```
sol_jump = solve(prob_jump,SSAStepper());
```

```
Error: UndefVarError: SSAStepper not defined
```

Post-processing

In order to get output comparable across implementations, we output the model at a fixed set of times.

```
out_jump = sol_jump(t);
```

```
Error: UndefVarError: sol_jump not defined
```

We can convert to a dataframe for convenience.

```
df_jump = DataFrame(out_jump')
```

Error: UndefVarError: out_jump not defined

```
df_jump[!,:t] = out_jump.t;
```

Error: UndefVarError: out_jump not defined

Plotting

We can now plot the results.

```
@df df_jump plot(:t,
    [:x1 :x2 :x3],
    label=["S" "I" "R"],
    xlabel="Time",
    ylabel="Number")
```

Error: UndefVarError: df_jump not defined

Benchmarking

```
@benchmark solve(prob_jump,FunctionMap())
```

Error: UndefVarError: FunctionMap not defined

Appendix

Computer Information

```
Julia Version 1.4.1
Commit 381693d3df* (2020-04-14 17:20 UTC)
Platform Info:
  OS: Linux (x86_64-pc-linux-gnu)
  CPU: Intel(R) Core(TM) i7-1065G7 CPU @ 1.30GHz
  WORD_SIZE: 64
  LIBM: libopenlibm
  LLVM: libLLVM-8.0.1 (ORCJIT, icelake-client)
Environment:
  JULIA_NUM_THREADS = 4
```

Package Information

```
Status `~/.julia/environments/v1.4/Project.toml`
[46ada45e-f475-11e8-01d0-f70cc89e6671] Agents 3.1.0
[c52e3926-4ff0-5f6e-af25-54175e0327b1] Atom 0.12.11
[6e4b80f9-dd63-53aa-95a3-0cdb28fa8baf] BenchmarkTools 0.5.0
[a134a8b2-14d6-55f6-9291-3336d3ab0209] BlackBoxOptim 0.5.0
[2445eb08-9709-466a-b3fc-47e12bd697a2] DataDrivenDiffEq 0.2.0
```

```
[a93c6f00-e57d-5684-b7b6-d8193f3e46c0] DataFrames 0.21.0
[ebbdde9d-f333-5424-9be2-dbf1e9acfb5e] DiffEqBayes 2.14.0
[459566f4-90b8-5000-8ac3-15dfb0a30def] DiffEqCallbacks 2.13.2
[c894b116-72e5-5b58-be3c-e6d8d4ac2b12] DiffEqJump 6.7.5
[1130ab10-4a5a-5621-a13d-e4788d82bd4c] DiffEqParamEstim 1.14.1
[0c46a032-eb83-5123-abaf-570d42b7fbaa] DifferentialEquations 6.14.0
[31c24e10-a181-5473-b8eb-7969acd0382f] Distributions 0.23.2
[634d3b9d-ee7a-5ddf-bec9-22491ea816e1] DrWatson 1.11.0
[587475ba-b771-5e3f-ad9e-33799f191a9c] Flux 0.8.3
[28b8d3ca-fb5f-59d9-8090-bfdbd6d07a71] GR 0.49.1
[523d8e89-b243-5607-941c-87d699ea6713] Gillespie 0.1.0
[7073ff75-c697-5162-941a-fcdaad2a7d2a] IJulia 1.21.2
[4076af6c-e467-56ae-b986-b466b2749572] JuMP 0.21.2
[e5e0dc1b-0480-54bc-9374-aad01c23163d] Juno 0.8.2
[093fc24a-ae57-5d10-9952-331d41423f4d] LightGraphs 1.3.3
[1914dd2f-81c6-5fcd-8719-6d5c9610ff09] MacroTools 0.5.5
[ee78f7c6-11fb-53f2-987a-cfe4a2b5a57a] Makie 0.9.5
[961ee093-0014-501f-94e3-6117800e7a78] ModelingToolkit 3.6.0
[76087f3c-5699-56af-9a33-bf431cd00edd] NLopt 0.6.0
[429524aa-4258-5aef-a3af-852621145aeb] Optim 0.21.0
[1dea7af3-3e70-54e6-95c3-0bf5283fa5ed] OrdinaryDiffEq 5.38.1
[91a5bcdd-55d7-5caf-9e0b-520d859cae80] Plots 1.3.1
[428bdadb-6287-5aa5-874b-9969638295fd] SimJulia 0.8.0
[05bca326-078c-5bf0-a5bf-ce7c7982d7fd] SimpleDiffEq 1.1.0
[f3b207a7-027a-5e70-b257-86293d7955fd] StatsPlots 0.14.6
[789caeaf-c7a9-5a7d-9973-96adeb23e2a0] StochasticDiffEq 6.23.0
[fce5fe82-541a-59a6-adf8-730c64b5f9a0] Turing 0.12.0
[44d3d7a6-8a23-5bf8-98c5-b353f8df5ec9] Weave 0.10.0
```