Mendes Multistate Model

Samuel Isaacson, Chris Rackauckas

July 4, 2020

Taken from Gupta and Mendes, *An Overview of Network-Based and -Free Approaches for Stochastic Simulation of Biochemical Systems*, Computation, 6 (9), 2018.

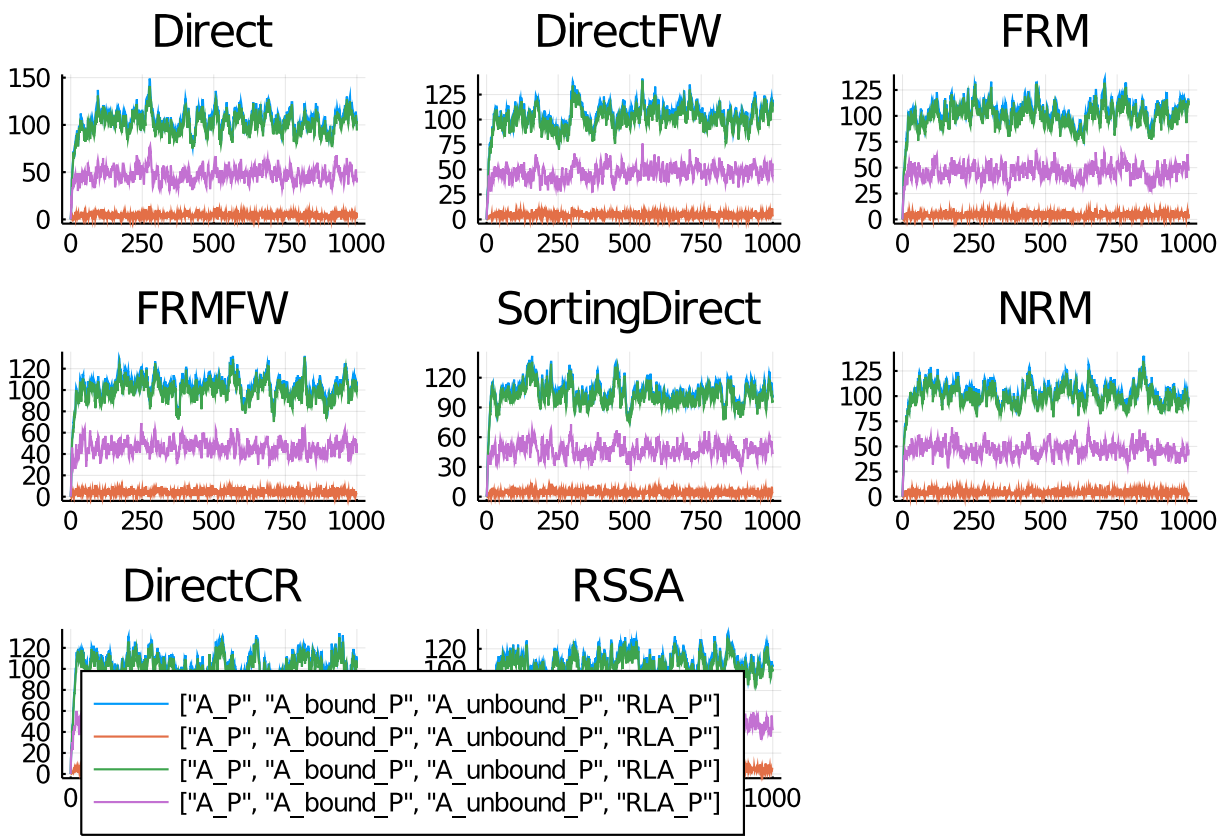
```
using DiffEqBase, DiffEqBiological, DiffEqJump, DiffEqProblemLibrary.JumpProblemLibrary,
Plots, Statistics
gr()
fmt = :png
JumpProblemLibrary.importjumpproblems()
```

1 Plot solutions by each method

```
methods = (Direct(),DirectFW(),FRM(),FRMFW(),SortingDirect(),NRM(),DirectCR(),RSSA())
shortlabels = [string(leg)[12:end-2] for leg in methods]
jprob   = prob_jump_multistate
tf      = 10.0*jprob.tstop
prob    = DiscreteProblem(jprob.u0, (0.0,tf), jprob.rates)
rn      = jprob.network
varlegs = ["A_P", "A_bound_P", "A_unbound_P", "RLA_P"]
varsyms = [
    [:S7,:S8,:S9],
    [:S9],
    [:S7,:S8],
    [:S7]
]
varidxs = []
for vars in varsyms
    push!(varidxs, [findfirst(isequal(sym),rn.syms) for sym in vars])
end

p = []
for (i,method) in enumerate(methods)
    jump_prob = JumpProblem(prob, method, rn, save_positions=(false,false))
    sol = solve(jump_prob, SSAStepper(), saveat=tf/1000.)
    solv = zeros(1001,4)
    for (i,varidx) in enumerate(varidxs)
        solv[:,i] = sum(sol[varidx,:], dims=1)
    end
    if i < length(methods)
        push!(p, plot(sol.t,solv,title=shortlabels[i],legend=false,format=fmt))
    else
        push!(p,
plot(sol.t,solv,title=shortlabels[i],legend=true,labels=varlegs,format=fmt))
```

```
    end
end
plot(p...,format=fmt)
```



2 Benchmarking performance of the methods

```
function run_benchmark!(t, jump_prob, stepper)
    sol = solve(jump_prob, stepper)
    @inbounds for i in 1:length(t)
        t[i] = @elapsed (sol = solve(jump_prob, stepper))
    end
end
```

```
run_benchmark! (generic function with 1 method)
```

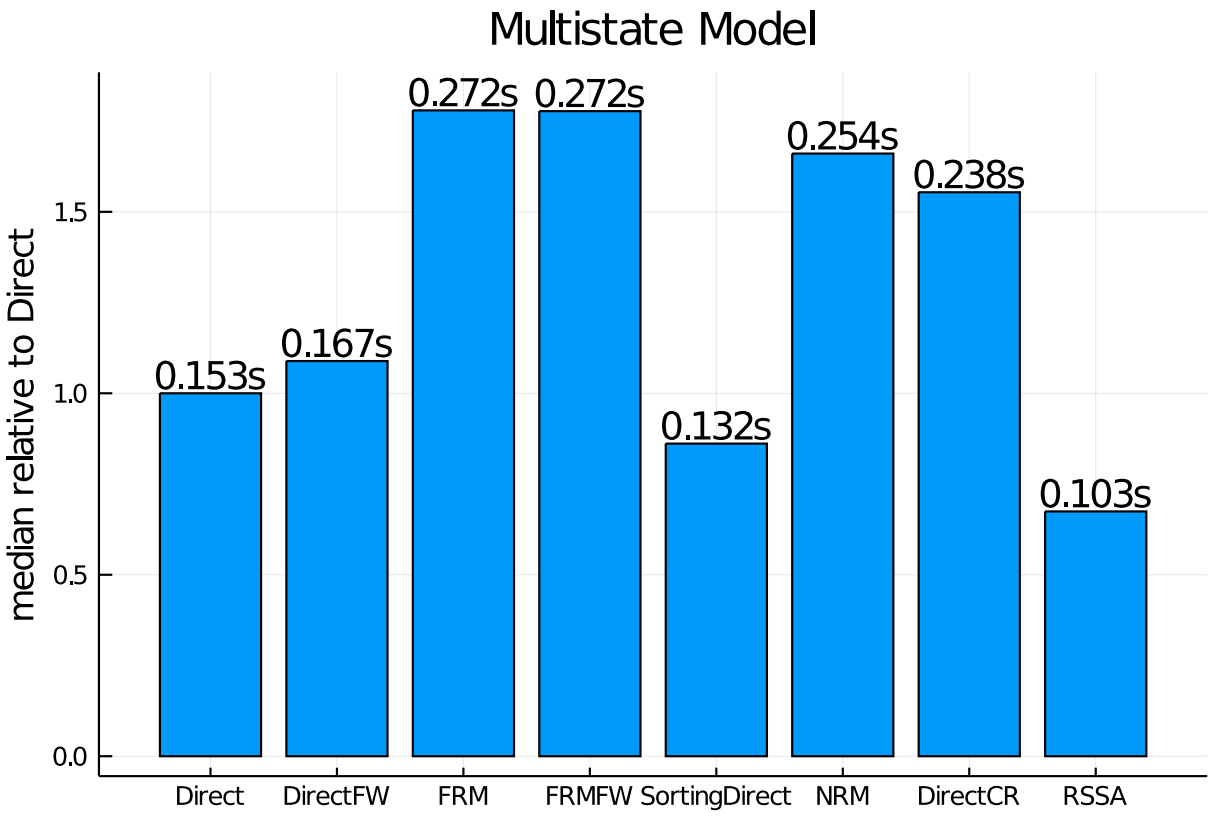
```
nsims = 100
benchmarks = Vector{Vector{Float64}}()
for method in methods
    jump_prob = JumpProblem(prob, method, rn, save_positions=(false,false))
    stepper = SSAStepper()
    t = Vector{Float64}(undef, nsims)
    run_benchmark!(t, jump_prob, stepper)
    push!(benchmarks, t)
end

medtimes = Vector{Float64}(undef,length(methods))
stdtimes = Vector{Float64}(undef,length(methods))
avgtimes = Vector{Float64}(undef,length(methods))
for i in 1:length(methods)
    medtimes[i] = median(benchmarks[i])
    avgtimes[i] = mean(benchmarks[i])
```

```
    stdtimes[i] = std(benchmarks[i])
end
using DataFrames

df =
DataFrame(names=shortlabels,medtimes=medtimes,relmedtimes=(medtimes/medtimes[1]),avgtimes=avgtimes,
std=stdtimes, cv=stdtimes./avgtimes)

sa = [text(string(round(mt,digits=3),"s"),:center,12) for mt in df.medtimes]
bar(df.names,df.relmedtimes,legend=:false, fmt=fmt)
scatter!(df.names, .05 .+ df.relmedtimes, markeralpha=0, series_annotations=sa, fmt=fmt)
ylabel!("median relative to Direct")
title!("Multistate Model")
```




```
using DiffEqBenchmarks
DiffEqBenchmarks.bench_footer(WEAVE_ARGS[:folder],WEAVE_ARGS[:file])
```

2.1 Appendix

These benchmarks are a part of the DiffEqBenchmarks.jl repository, found at: https://github.com/JuliaDi

To locally run this tutorial, do the following commands:

```
using DiffEqBenchmarks
DiffEqBenchmarks.weave_file("Jumps","Mendes_multistate_example.jmd")
```

Computer Information:

```
Julia Version 1.4.2
```

```
Commit 44fa15b150* (2020-05-23 18:35 UTC)
Platform Info:
  OS: Linux (x86_64-pc-linux-gnu)
  CPU: Intel(R) Core(TM) i7-9700K CPU @ 3.60GHz
  WORD_SIZE: 64
  LIBM: libopenlibm
  LLVM: libLLVM-8.0.1 (ORCJIT, skylake)
Environment:
  JULIA_DEPOT_PATH = /builds/JuliaGPU/DiffEqBenchmarks.jl/.julia
  JULIA_CUDA_MEMORY_LIMIT = 2147483648
  JULIA_PROJECT = @.
  JULIA_NUM_THREADS = 8
```

Package Information:

```
Status: `/builds/JuliaGPU/DiffEqBenchmarks.jl/benchmarks/Jumps/Project.toml`
[a93c6f00-e57d-5684-b7b6-d8193f3e46c0] DataFrames 0.21.4
[2b5f629d-d688-5b77-993f-72d75c75574e] DiffEqBase 6.40.4
[eb300fae-53e8-50a0-950c-e21f52c2b7e0] DiffEqBiological 4.3.0
[c894b116-72e5-5b58-be3c-e6d8d4ac2b12] DiffEqJump 6.9.3
[a077e3f3-b75c-5d7f-a0c6-6bc4c8ec64a9] DiffEqProblemLibrary 4.8.0
[1dea7af3-3e70-54e6-95c3-0bf5283fa5ed] OrdinaryDiffEq 5.41.0
[91a5bcdd-55d7-5caf-9e0b-520d859cae80] Plots 1.5.3
[10745b16-79ce-11e8-11f9-7d13ad32a3b2] Statistics
```