Oval2 Long Run

Chris Rackauckas

April 28, 2019

```
using StochasticDiffEq, DiffEqProblemLibrary, Random
using DiffEqProblemLibrary.SDEProblemLibrary: importsdeproblems; importsdeproblems()
prob =
    DiffEqProblemLibrary.SDEProblemLibrary.oval2ModelExample(largeFluctuations=true,useBigs=false)
```

```
SDEProblem with uType Array{Float64,1} and tType Float64. In-place: true
timespan: (0.0, 500.0)
u0: [0.128483, 1.25685, 0.0030203, 0.0027977, 0.0101511, 0.0422942, 0.23913
5, 0.0008014, 0.0001464, 2.67e-5, 4.8e-6, 9.0e-7, 0.0619917, 1.24443, 0.048
6676, 199.938, 137.427, 1.51802, 1.51802]
```

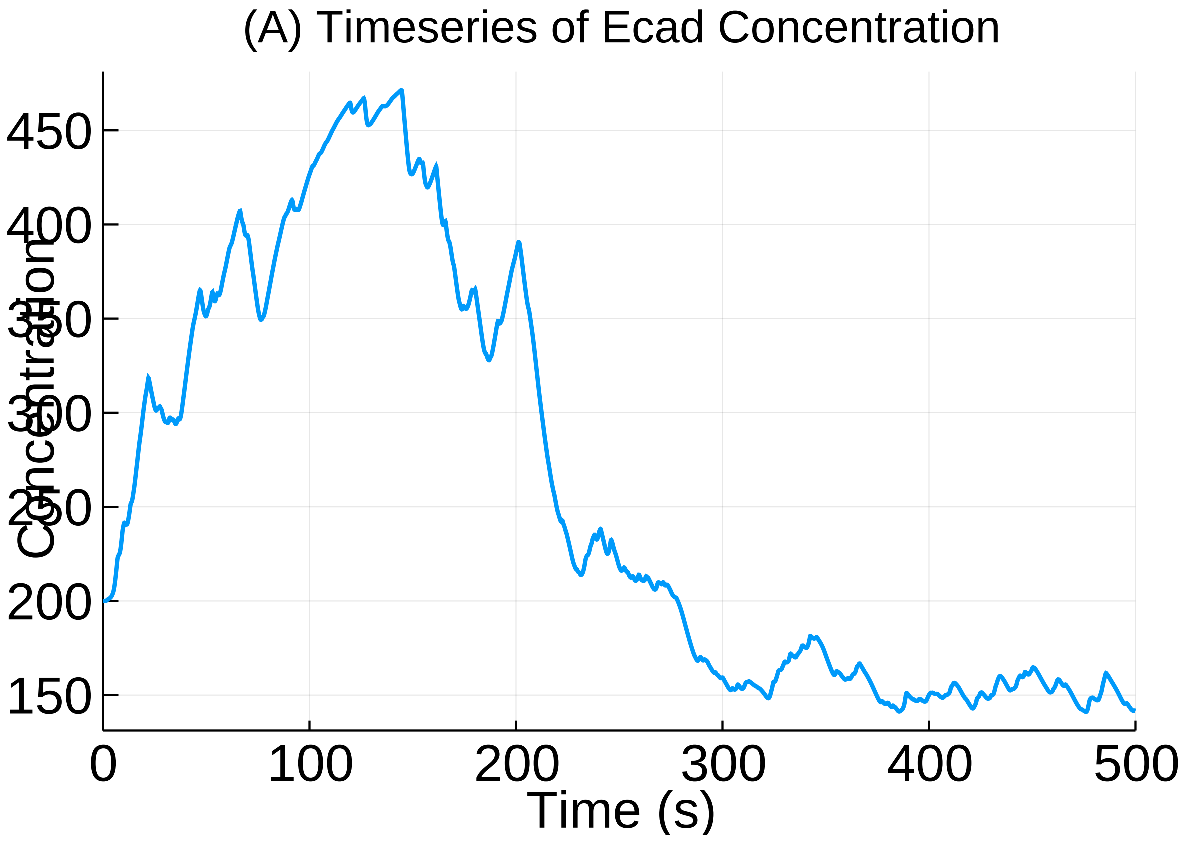
```
Random.seed!(250)
prob = remake(prob,tspan = (0.0,500.0))
sol = solve(prob,SRIW1(),dt=(1/2)^(18),progress=true,qmax=1.125,
          saveat=0.1,abstol=1e-5,reltol=1e-3,maxiters=1e7);
Random.seed!(250)
prob = remake(prob,tspan = (0.0,500.0))
@time sol = solve(prob,SRIW1(),dt=(1/2)^(18),progress=true,qmax=1.125,
    saveat=0.1,abstol=1e-5,reltol=1e-3,maxiters=1e7);
```

```
15.496978 seconds (27.43 M allocations: 4.631 GiB, 28.73% gc time)
```

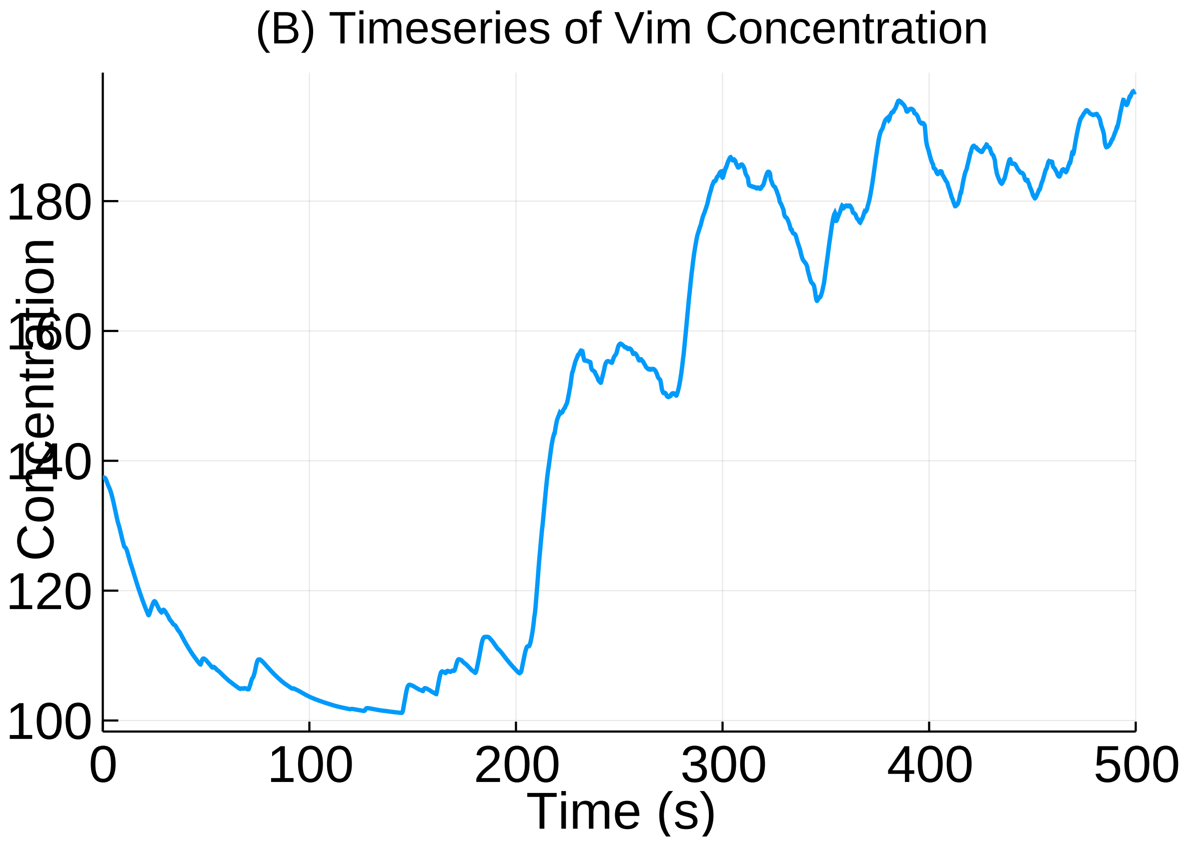
```
println(maximum(sol[:,2]))
```

```
199.93762312284827
```

```
using Plots; gr()
lw = 2
lw2 = 3
p1 = plot(sol,vars=(0,16),
          title="(A) Timeseries of Ecad Concentration",xguide="Time (s)",
          yguide="Concentration",guidefont=font(16),tickfont=font(16),
          linewidth=lw,leg=false)
```



```
p2 = plot(sol,vars=(0,17),
        title="(B) Timeseries of Vim Concentration",xguide="Time (s)",
        yguide="Concentration",guidefont=font(16),
        tickfont=font(16),linewidth=lw,leg=false)
```



```
prob = remake(prob,tspan = (0.0,1.0))
```

```julia
## Little Run
sol = solve(prob,EM(),dt=(1/2)^(20),
          progressbar=true,saveat=0.1)
println("EM")
```

```
EM
```

```julia
@time sol = solve(prob,EM(),dt=(1/2)^(20),
          progressbar=true,saveat=0.1)
```

```
1.027161 seconds (1.00 M allocations: 640.905 MiB, 49.77% gc time)
```

```julia
sol = solve(prob,SRI(),dt=(1/2)^(18),adaptive=false,
            progressbar=true,save_everystep=false)
println("SRI")
```

```
SRI
```

```julia
@time sol = solve(prob,SRI(),dt=(1/2)^(18),adaptive=false,
          progressbar=true,save_everystep=false)
```

```
1.129157 seconds (786.75 k allocations: 184.043 MiB, 14.32% gc time)
```

```julia
sol = solve(prob,SRIW1(),dt=(1/2)^(18),adaptive=false,
          adaptivealg=:RSwM3,progressbar=false,qmax=4,saveat=0.1)
println("SRIW1")
```

```
SRIW1
```

```julia
@time sol = solve(prob,SRIW1(),dt=(1/2)^(18),adaptive=false,
          adaptivealg=:RSwM3,progressbar=false,qmax=4,saveat=0.1)
```

```
0.553655 seconds (786.61 k allocations: 184.027 MiB, 32.68% gc time)
```

```julia
sol = solve(prob,SRI(),dt=(1/2)^(18),
          adaptivealg=:RSwM3,progressbar=false,qmax=1.125,
          saveat=0.1,abstol=1e-6,reltol=1e-4)
println("SRI Adaptive")
```

```
SRI Adaptive
```

```julia
@time sol = solve(prob,SRI(),dt=(1/2)^(18),
          adaptivealg=:RSwM3,progressbar=false,qmax=1.125,
          saveat=0.1,abstol=1e-6,reltol=1e-4)
```

```
0.176920 seconds (152.61 k allocations: 27.344 MiB, 11.63% gc time)
```

```julia
@show length(sol.t)
```

```
length(sol.t) = 11
```

```julia
sol = solve(prob,SRIW1(),dt=(1/2)^(18),
          adaptivealg=:RSwM3,progressbar=false,qmax=1.125,
          saveat=0.1,abstol=1e-6,reltol=1e-4)
println("SRIW1 Adaptive")
```

```
SRIW1 Adaptive
```

```julia
@time sol = solve(prob,SRIW1(),dt=(1/2)^(18),
          adaptivealg=:RSwM3,progressbar=false,qmax=1.125,
          saveat=0.1,abstol=1e-6,reltol=1e-4)
```

```
0.036006 seconds (81.60 k allocations: 14.623 MiB)
```

```
@show length(sol.t)
```

```
length(sol.t) = 11
11
```

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

0.1 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("AdaptiveSDE","Oval2LongRun.jmd")
```

Computer Information:

```
Julia Version 1.1.0
Commit 80516ca202 (2019-01-21 21:24 UTC)
Platform Info:
  OS: Linux (x86_64-pc-linux-gnu)
  CPU: Intel(R) Xeon(R) CPU E5-2680 v4 @ 2.40GHz
  WORD_SIZE: 64
  LIBM: libopenlibm
  LLVM: libLLVM-6.0.1 (ORCJIT, haswell)
```

Package Information:

```
Status: `/home/crackauckas/.julia/environments/v1.1/Project.toml`
[c52e3926-4ff0-5f6e-af25-54175e0327b1] Atom 0.8.5
[bcd4f6db-9728-5f36-b5f7-82caef46ccdb] DelayDiffEq 5.2.0
[bb2cbb15-79fc-5d1e-9bf1-8ae49c7c1650] DiffEqBenchmarks 0.1.0
[459566f4-90b8-5000-8ac3-15dfb0a30def] DiffEqCallbacks 2.5.2
[f3b72e0c-5b89-59e1-b016-84e28bfd966d] DiffEqDevTools 2.7.2+
[77a26b50-5914-5dd7-bc55-306e6241c503] DiffEqNoiseProcess 3.1.0
[055956cb-9e8b-5191-98cc-73ae4a59e68a] DiffEqPhysics 3.1.0
[a077e3f3-b75c-5d7f-a0c6-6bc4c8ec64a9] DiffEqProblemLibrary 4.1.0
[0c46a032-eb83-5123-abaf-570d42b7fbaa] DifferentialEquations 6.3.0
[b305315f-e792-5b7a-8f41-49f472929428] Elliptic 0.5.0
[e5e0dc1b-0480-54bc-9374-aad01c23163d] Juno 0.7.0
[7f56f5a3-f504-529b-bc02-0b1fe5e64312] LSODA 0.4.0
[c030b06c-0b6d-57c2-b091-7029874bd033] ODE 2.4.0
[54ca160b-1b9f-5127-a996-1867f4bc2a2c] ODEInterface 0.4.5
[09606e27-ecf5-54fc-bb29-004bd9f985bf] ODEInterfaceDiffEq 3.1.0
```

```
[1dea7af3-3e70-54e6-95c3-0bf5283fa5ed] OrdinaryDiffEq 5.5.0
[65888b18-ceab-5e60-b2b9-181511a3b968] ParameterizedFunctions 4.1.1
[91a5bcdd-55d7-5caf-9e0b-520d859cae80] Plots 0.24.0
[d330b81b-6aea-500a-939a-2ce795aea3ee] PyPlot 2.8.1
[90137ffa-7385-5640-81b9-e52037218182] StaticArrays 0.10.3
[789caeaf-c7a9-5a7d-9973-96adeb23e2a0] StochasticDiffEq 6.1.1
[c3572dad-4567-51f8-b174-8c6c989267f4] Sundials 3.3.0+
[92b13dbe-c966-51a2-8445-caca9f8a7d42] TaylorIntegration 0.4.1
[44d3d7a6-8a23-5bf8-98c5-b353f8df5ec9] Weave 0.9.0
```