Oval2 Timings

Chris Rackauckas

July 5, 2020

```
using StochasticDiffEq, DiffEqProblemLibrary, Random, Base.Threads
using DiffEqProblemLibrary.SDEProblemLibrary: importsdeproblems; importsdeproblems()
prob =
DiffEqProblemLibrary.SDEProblemLibrary.oval2ModelExample(largeFluctuations=true,useBigs=false)
prob_func(prob,i,repeat) = remake(prob,seed=i)
prob = EnsembleProblem(remake(prob,tspan=(0.0,1.0)),prob_func=prob_func)
js = 16:21
dts = 1.0 ./ 2.0 .^ (js)
trajectories = 1000
fails = Array{Int}(undef,length(dts),3)
times = Array{Float64}(undef,length(dts),3)
```

```
6×3 Array{Float64,2}:
 6.95334e-310  6.95333e-310  6.95333e-310
 6.95333e-310  6.95333e-310  6.95334e-310
 6.95333e-310  6.95333e-310  6.95334e-310
 6.95333e-310  6.95333e-310  6.95333e-310
 6.95333e-310  6.95333e-310  6.95333e-310
 6.95333e-310  6.95333e-310  3.59663e252
```

0.1 Timing Runs

```
sol =
solve(prob,SRIW1(),EnsembleThreads(),abstol=2.0^(-13),reltol=2.0^(-7),maxiters=Int(1e11),qmax=1.125,sa
adaptive_time = @elapsed sol =
solve(prob,SRIW1(),EnsembleThreads(),abstol=2.0^(-13),reltol=2.0^(-7),maxiters=Int(1e11),qmax=1.125,sa
numfails = sum([Int(any(isnan,sol[i]) || sol[i].t[end] != 1) for i in 1:trajectories])
best_adaptive_time = numfails != 0 ? Inf : adaptive_time
println("The number of Adaptive Fails is $numfails. Elapsed time was $adaptive_time")
```

```
The number of Adaptive Fails is 0. Elapsed time was 4.616666083
```

```
sol =
solve(prob,SRI(error_terms=2),EnsembleThreads(),abstol=2.0^(-13),reltol=2.0^(-7),maxiters=Int(1e11),qm
adaptive_time = @elapsed sol =
solve(prob,SRI(error_terms=2),EnsembleThreads(),abstol=2.0^(-13),reltol=2.0^(-7),maxiters=Int(1e11),qm
numfails = sum([Int(any(isnan,sol[i]) || sol[i].t[end] != 1) for i in 1:trajectories])
best_adaptive_time = numfails != 0 ? adaptive_time :
min(best_adaptive_time,adaptive_time)
println("The number of Adaptive Fails is $numfails. Elapsed time was $adaptive_time")
```

```
The number of Adaptive Fails is 0. Elapsed time was 7.547009307
```

```
sol =
solve(prob,SRI(),EnsembleThreads(),abstol=2.0^(-14),reltol=2.0^(-18),maxiters=Int(1e11),qmax=1.125,sav
adaptive_time = @elapsed sol =
solve(prob,SRI(),EnsembleThreads(),abstol=2.0^(-14),reltol=2.0^(-18),maxiters=Int(1e11),qmax=1.125,sav
numfails = sum([Int(any(isnan,sol[i]) || sol[i].t[end] != 1) for i in 1:trajectories])
best_adaptive_time = numfails != 0 ? adaptive_time :
min(best_adaptive_time,adaptive_time)
println("The number of Adaptive Fails is $numfails. Elapsed time was $adaptive_time")
```

```
The number of Adaptive Fails is 0. Elapsed time was 83.207262565
```

```
sol =
solve(prob,SRI(tableau=StochasticDiffEq.constructSRIOpt1()),EnsembleThreads(),abstol=2.0^(-7),reltol=2
adaptive_time = @elapsed sol =
solve(prob,SRI(tableau=StochasticDiffEq.constructSRIOpt1()),EnsembleThreads(),abstol=2.0^(-7),reltol=2
numfails = sum([Int(any(isnan,sol[i]) || sol[i].t[end] != 1) for i in 1:trajectories])
best_adaptive_time = numfails != 0 ? adaptive_time :
min(best_adaptive_time,adaptive_time)
println("The number of Adaptive Fails is $numfails. Elapsed time was $adaptive_time")
```

```
The number of Adaptive Fails is 0. Elapsed time was 0.679515924
```

```
sol =
solve(prob,SOSRI(),EnsembleThreads(),abstol=2.0^(-7),reltol=2.0^(-4),maxiters=Int(1e11),qmax=1.125,sav
adaptive_time = @elapsed sol =
solve(prob,SOSRI(),EnsembleThreads(),abstol=2.0^(-7),reltol=2.0^(-4),maxiters=Int(1e11),qmax=1.125,sav
numfails = sum([Int(any(isnan,sol[i]) || sol[i].t[end] != 1) for i in 1:trajectories])
best_adaptive_time = numfails != 0 ? adaptive_time :
min(best_adaptive_time,adaptive_time)
println("The number of Adaptive Fails is $numfails. Elapsed time was $adaptive_time")
```

```
The number of Adaptive Fails is 0. Elapsed time was 0.542992857
```

```
sol =
solve(prob,SOSRI(),EnsembleThreads(),abstol=2.0^(-7),reltol=2.0^(-6),maxiters=Int(1e11),qmax=1.125,sav
adaptive_time = @elapsed sol =
solve(prob,SOSRI(),EnsembleThreads(),abstol=2.0^(-7),reltol=2.0^(-6),maxiters=Int(1e11),qmax=1.125,sav
numfails = sum([Int(any(isnan,sol[i]) || sol[i].t[end] != 1) for i in 1:trajectories])
best_adaptive_time = numfails != 0 ? adaptive_time :
min(best_adaptive_time,adaptive_time)
println("The number of Adaptive Fails is $numfails. Elapsed time was $adaptive_time")
```

```
The number of Adaptive Fails is 0. Elapsed time was 0.54516048
```

```
sol =
solve(prob,SOSRI(),EnsembleThreads(),abstol=2.0^(-12),reltol=2.0^(-15),maxiters=Int(1e11),qmax=1.125,s
adaptive_time = @elapsed sol =
solve(prob,SOSRI(),EnsembleThreads(),abstol=2.0^(-12),reltol=2.0^(-15),maxiters=Int(1e11),qmax=1.125,s
numfails = sum([Int(any(isnan,sol[i]) || sol[i].t[end] != 1) for i in 1:trajectories])
best_adaptive_time = numfails != 0 ? adaptive_time :
min(best_adaptive_time,adaptive_time)
println("The number of Adaptive Fails is $numfails. Elapsed time was $adaptive_time")
```

```
The number of Adaptive Fails is 0. Elapsed time was 12.150339919
```

```
sol =
solve(prob,SOSRI(),EnsembleThreads(),abstol=2.0^(-13),reltol=2.0^(-7),maxiters=Int(1e11),qmax=1.125,sa
adaptive_time = @elapsed sol =
solve(prob,SOSRI(),EnsembleThreads(),abstol=2.0^(-13),reltol=2.0^(-7),maxiters=Int(1e11),qmax=1.125,sa
numfails = sum([Int(any(isnan,sol[i]) || sol[i].t[end] != 1) for i in 1:trajectories])
```

```
best_adaptive_time = numfails != 0 ? adaptive_time :
min(best_adaptive_time,adaptive_time)
println("The number of Adaptive Fails is $numfails. Elapsed time was $adaptive_time")
```

```
The number of Adaptive Fails is 0. Elapsed time was 1.146088443
```

```
sol =
solve(prob,SOSRI(),EnsembleThreads(),abstol=2.0^(-12),reltol=2.0^(-15),maxiters=Int(1e11),qmax=1.125,s
adaptive_time = @elapsed sol =
solve(prob,SOSRI(),EnsembleThreads(),abstol=2.0^(-12),reltol=2.0^(-15),maxiters=Int(1e11),qmax=1.125,s
numfails = sum([Int(any(isnan,sol[i]) || sol[i].t[end] != 1) for i in 1:trajectories])
best_adaptive_time = numfails != 0 ? adaptive_time :
min(best_adaptive_time,adaptive_time)
println("The number of Adaptive Fails is $numfails. Elapsed time was $adaptive_time")
```

```
The number of Adaptive Fails is 0. Elapsed time was 12.117790284
```

```
sol =
solve(prob,SOSRI2(),EnsembleThreads(),abstol=2.0^(-12),reltol=2.0^(-15),maxiters=Int(1e11),qmax=1.125,
adaptive_time = @elapsed sol =
solve(prob,SOSRI2(),EnsembleThreads(),abstol=2.0^(-12),reltol=2.0^(-15),maxiters=Int(1e11),qmax=1.125,
numfails = sum([Int(any(isnan,sol[i]) || sol[i].t[end] != 1) for i in 1:trajectories])
best_adaptive_time = numfails != 0 ? adaptive_time :
min(best_adaptive_time,adaptive_time)
println("The number of Adaptive Fails is $numfails. Elapsed time was $adaptive_time")
```

```
The number of Adaptive Fails is 0. Elapsed time was 11.930675559
```

```
sol =
solve(prob,SOSRI2(),EnsembleThreads(),abstol=2.0^(-13),reltol=2.0^(-11),maxiters=Int(1e11),qmax=1.125,
adaptive_time = @elapsed sol =
solve(prob,SOSRI2(),EnsembleThreads(),abstol=2.0^(-13),reltol=2.0^(-11),maxiters=Int(1e11),qmax=1.125,
numfails = sum([Int(any(isnan,sol[i]) || sol[i].t[end] != 1) for i in 1:trajectories])
best_adaptive_time = numfails != 0 ? adaptive_time :
min(best_adaptive_time,adaptive_time)
println("The number of Adaptive Fails is $numfails. Elapsed time was $adaptive_time")
```

```
The number of Adaptive Fails is 0. Elapsed time was 5.67502337
```

```
sol =
solve(prob,SOSRI2(),EnsembleThreads(),abstol=2.0^(-13),reltol=2.0^(-11),maxiters=Int(1e11),qmax=1.125,
adaptive_time = @elapsed sol =
solve(prob,SOSRI2(),EnsembleThreads(),abstol=2.0^(-13),reltol=2.0^(-11),maxiters=Int(1e11),qmax=1.125,
numfails = sum([Int(any(isnan,sol[i]) || sol[i].t[end] != 1) for i in 1:trajectories])
best_adaptive_time = numfails != 0 ? adaptive_time :
min(best_adaptive_time,adaptive_time)
println("The number of Adaptive Fails is $numfails. Elapsed time was $adaptive_time")
```

```
The number of Adaptive Fails is 0. Elapsed time was 5.65182531
```

```
for j in eachindex(js)
  println("j = $j")
  sol
=solve(prob,EM(),EnsembleThreads(),dt=dts[j],maxiters=Int(1e11),save_everystep=false,verbose=false,tra
  t1 = @elapsed sol =
solve(prob,EM(),EnsembleThreads(),dt=dts[j],maxiters=Int(1e11),save_everystep=false,verbose=false,traj
  numfails = sum([Int(any(isnan,sol[i]) || sol[i].t[end] != 1) for i in 1:trajectories])
  println("The number of Euler-Maruyama Fails is $numfails. Elapsed time was $t1")
  fails[j,1] = numfails
  times[j,1] = t1
end
```

```
j = 1
The number of Euler-Maruyama Fails is 20. Elapsed time was 13.112608104
j = 2
The number of Euler-Maruyama Fails is 5. Elapsed time was 25.981692926
j = 3
The number of Euler-Maruyama Fails is 1. Elapsed time was 52.034032394
j = 4
The number of Euler-Maruyama Fails is 0. Elapsed time was 103.012684876
j = 5
The number of Euler-Maruyama Fails is 0. Elapsed time was 204.886064772
j = 6
The number of Euler-Maruyama Fails is 0. Elapsed time was 411.100743688
```

```
for j in 1:4
  println("j = $j")
  sol
=solve(prob,SRIW1(),EnsembleThreads(),dt=dts[j],maxiters=Int(1e11),save_everystep=false,verbose=false,
  t1 = @elapsed sol =
solve(prob,SRIW1(),EnsembleThreads(),dt=dts[j],maxiters=Int(1e11),save_everystep=false,verbose=false,t
  numfails = sum([Int(any(isnan,sol[i]) || sol[i].t[end] != 1) for i in 1:trajectories])
  println("The number of SRIW1 Fails is $numfails. Elapsed time was $t1")
  fails[j,3] = numfails
  times[j,3] = t1
end
```

```
j = 1
The number of SRIW1 Fails is 971. Elapsed time was 0.506223672
j = 2
The number of SRIW1 Fails is 977. Elapsed time was 0.599828934
j = 3
The number of SRIW1 Fails is 978. Elapsed time was 0.550247673
j = 4
The number of SRIW1 Fails is 981. Elapsed time was 0.527410425
```

```
js = 17:21
dts = 1.0 ./2.0 .^ (js)
for j in 1:6
  println("j = $j")
  sol
=solve(prob,ImplicitEM(),EnsembleThreads(),dt=dts[j],maxiters=Int(1e11),save_everystep=false,verbose=f
  t1 = @elapsed sol =
solve(prob,ImplicitEM(),EnsembleThreads(),dt=dts[j],maxiters=Int(1e11),save_everystep=false,verbose=fa
  numfails = sum([Int(any(isnan,sol[i]) || sol[i].t[end] != 1) for i in 1:trajectories])
  println("The number of Implicit-EM Fails is $numfails. Elapsed time was $t1")
end
```

```
j = 1
The number of Implicit-EM Fails is 35. Elapsed time was 0.246821009
j = 2
The number of Implicit-EM Fails is 31. Elapsed time was 0.246915375
j = 3
The number of Implicit-EM Fails is 41. Elapsed time was 0.352916163
j = 4
The number of Implicit-EM Fails is 33. Elapsed time was 0.262675363
j = 5
The number of Implicit-EM Fails is 31. Elapsed time was 0.255392482
j = 6
Error: BoundsError: attempt to access 5-element Array{Float64,1} at index [
6]
```

```
js = 17:21
dts = 1.0 ./ 2.0 .^(js)

for j in 1:6
  println("j = $j")
  sol
=solve(prob,ImplicitRKMil(),EnsembleThreads(),dt=dts[j],maxiters=Int(1e11),save_everystep=false,verbos
  t1 = @elapsed sol =
solve(prob,ImplicitRKMil(),EnsembleThreads(),dt=dts[j],maxiters=Int(1e11),save_everystep=false,verbose
  numfails = sum([Int(any(isnan,sol[i]) || sol[i].t[end] != 1) for i in 1:trajectories])
  println("The number of Implicit-RKMil Fails is $numfails. Elapsed time was $t1")
end
```

```
j = 1
The number of Implicit-RKMil Fails is 28. Elapsed time was 4.426184871
j = 2
The number of Implicit-RKMil Fails is 31. Elapsed time was 4.42317794
j = 3
The number of Implicit-RKMil Fails is 31. Elapsed time was 4.494598556
j = 4
The number of Implicit-RKMil Fails is 16. Elapsed time was 4.47645517
j = 5
The number of Implicit-RKMil Fails is 23. Elapsed time was 4.32511253
j = 6
Error: BoundsError: attempt to access 5-element Array{Float64,1} at index [
6]
```

```
for j in 1:6
  println("j = $j")
  sol
=solve(prob,RKMil(),EnsembleThreads(),dt=dts[j],maxiters=Int(1e11),save_everystep=false,verbose=false,
  t1 = @elapsed sol =
solve(prob,RKMil(),EnsembleThreads(),dt=dts[j],maxiters=Int(1e11),save_everystep=false,verbose=false,t
  numfails = sum([Int(any(isnan,sol[i]) || sol[i].t[end] != 1) for i in 1:trajectories])
  println("The number of RKMil Fails is $numfails. Elapsed time was $t1")
  fails[j,2] = numfails
  times[j,2] = t1
end
```

```
j = 1
The number of RKMil Fails is 3. Elapsed time was 2.049488406
j = 2
The number of RKMil Fails is 5. Elapsed time was 2.0438551
j = 3
The number of RKMil Fails is 4. Elapsed time was 2.045340218
j = 4
The number of RKMil Fails is 5. Elapsed time was 1.998770594
j = 5
The number of RKMil Fails is 4. Elapsed time was 2.012845096
j = 6
Error: BoundsError: attempt to access 5-element Array{Float64,1} at index [
6]
```

```
using Plots
lw = 3
p2 =
plot(dts,times,xscale=:log2,yscale=:log2,guidefont=font(16),tickfont=font(14),yguide="Elapsed
Time (s)",xguide=L"Chosen $\Delta t$",top_margin=50px,linewidth=lw,lab=["Euler-Maruyama"
"RK-Mil" "RosslerSRI"],legendfont=font(14))
```

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

```
plot!(dts,repmat([best_adaptive_time],11),linewidth=lw,line=:dash,lab="ESRK+RSwM3",left_margin=75px)
```

```
Error: UndefVarError: repmat not defined
```

```
scatter!([2.0^(-20);2.0^(-20);2.0^(-18)],[times[5,1];times[5,2];times[3,3]],markersize=20,c=:red,lab="
plot(p2,size=(800,800))
```

```
Error: UndefVarError: p2 not defined
```

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

0.2 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("StiffSDE","Oval2Timings.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/StiffSDE/Project.toml`
[f3b72e0c-5b89-59e1-b016-84e28bfd966d] DiffEqDevTools 2.22.0
[77a26b50-5914-5dd7-bc55-306e6241c503] DiffEqNoiseProcess 5.0.2
[a077e3f3-b75c-5d7f-a0c6-6bc4c8ec64a9] DiffEqProblemLibrary 4.8.0
[91a5bcdd-55d7-5caf-9e0b-520d859cae80] Plots 1.5.3
[789caeaf-c7a9-5a7d-9973-96adeb23e2a0] StochasticDiffEq 6.24.0
[37e2e46d-f89d-539d-b4ee-838fcccc9c8e] LinearAlgebra
[9a3f8284-a2c9-5f02-9a11-845980a1fd5c] Random
[10745b16-79ce-11e8-11f9-7d13ad32a3b2] Statistics
```