qmax Determination

Chris Rackauckas

May 8, 2021

```
qs = 1.0 .+ 2.0.^(-5:2)
times = Array{Float64}(undef,length(qs),4)
means = Array{Float64}(undef,length(qs),4)

using StochasticDiffEq, DiffEqProblemLibrary, Random,
      Plots, ParallelDataTransfer, DiffEqMonteCarlo, Distributed
Random.seed!(99)

using DiffEqProblemLibrary.SDEProblemLibrary: importsdeproblems; importsdeproblems()
full_prob =
DiffEqProblemLibrary.SDEProblemLibrary.oval2ModelExample(largeFluctuations=true,useBigs=false)
import DiffEqProblemLibrary.SDEProblemLibrary: prob_sde_additivesystem,
            prob_sde_additive, prob_sde_2Dlinear, prob_sde_linear, prob_sde_wave
prob = remake(full_prob,tspan=(0.0,1.0))

println("Solve once to compile.")
sol = solve(prob,EM(),dt=1/2^(18))
Int(sol.u[end][1]!=NaN)
println("Compilation complete.")
num_runs = 10000

probs = Vector{SDEProblem}(undef,3)
p1 = Vector{Any}(undef,3)
p2 = Vector{Any}(undef,3)
p3 = Vector{Any}(undef,3)
## Problem 1
probs[1] = prob_sde_linear
## Problem 2
probs[2] = prob_sde_wave
## Problem 3
probs[3] = prob_sde_additive

println("Setup Complete")

## Timing Runs

function runAdaptive(i,k)
  sol = solve(prob,SRIW1(),dt=1/2^(8),abstol=2.0^(-15),reltol=2.0^(-10),
              verbose=false,maxIters=Int(1e12),qmax=qs[k])
  Int(any(isnan,sol[end]) || sol.t[end] != 1)
end

#Compile
monte_prob = EnsembleProblem(probs[1])
test_mc =
solve(monte_prob,SRIW1(),dt=1/2^(4),adaptive=true,trajectories=1000,abstol=2.0^(-1),reltol=0)
```

```
DiffEqBase.calculate_monte_errors(test_mc);
```

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

0.1 qmax test on Oval2 Model

```
for k in eachindex(qs)
  global times
  Random.seed!(99)
  adaptiveTime = @elapsed numFails = sum(map((i)->runAdaptive(i,k),1:num_runs))
  println("k was $k. The number of Adaptive Fails is $numFails. Elapsed time was
$adaptiveTime")
  times[k,4] = adaptiveTime
end
```

```
Error: UndefVarError: num_runs not defined
```

0.2 qmax test on other problems

```
for k in eachindex(probs)
  global probs, times, means, qs
  println("Problem $k")
  ## Setup
  prob = probs[k]

  for i in eachindex(qs)
    msim =
solve(monte_prob,dt=1/2^(4),SRIW1(),adaptive=true,trajectories=num_runs,abstol=2.0^(-13),reltol=0,qmax
    test_msim = DiffEqBase.calculate_monte_errors(msim)
    times[i,k] = test_msim.elapsedTime
    means[i,k] = test_msim.error_means[:final]
    println("for k=$k and i=$i, we get that the error was $(means[i,k]) and it took
$(times[i,k]) seconds")
  end
end
```

```
Error: UndefVarError: probs not defined
```

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