qmax Determination

Chris Rackauckas

July 5, 2020

```
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.")
```

```
Solve once to compile.
```

```
sol = solve(prob,EM(),dt=1/2^(18))
Int(sol.u[end][1]!=NaN)
println("Compilation complete.")
```

```
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")
```

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

```
Error: MethodError: no method matching append!(::Nothing, ::Array{DiffEqBas
e.RODESolution{Float64,1,Array{Float64,1},Array{Float64,1},Dict{Symbol,Floa
t64},Array{Float64,1},DiffEqNoiseProcess.NoiseProcess{Float64,1,Float64,Flo
at64,Float64,Array{Float64,1},typeof(DiffEqNoiseProcess.WHITE_NOISE_DIST),t
ypeof(DiffEqNoiseProcess.WHITE_NOISE_BRIDGE),false,ResettableStacks.Resetta
bleStack{Tuple{Float64,Float64,Float64},false},ResettableStacks.ResettableS
tack{Tuple{Float64,Float64,Float64},false},DiffEqNoiseProcess.RSWM{Float64}
,Nothing,RandomNumbers.Xorshifts.Xoroshiro128Plus},DiffEqBase.SDEProblem{Fl
oat64,Tuple{Float64,Float64},false,DiffEqBase.NullParameters,Nothing,DiffEq
Base.SDEFunction{false,typeof(DiffEqProblemLibrary.SDEProblemLibrary.f_line
ar),typeof(DiffEqProblemLibrary.SDEProblemLibrary.σ_linear),LinearAlgebra.U
niformScaling{Bool},typeof(DiffEqProblemLibrary.SDEProblemLibrary.linear_an
alytic),Nothing,Nothing,Nothing,Nothing,Nothing,Nothing,Nothing,Nothing,Not
hing,Nothing,Nothing,Nothing},typeof(DiffEqProblemLibrary.SDEProblemLibrary
.σ_linear),Base.Iterators.Pairs{Union{},Union{},Tuple{},NamedTuple{(),Tuple
{}}},Nothing},StochasticDiffEq.SRIW1,StochasticDiffEq.LinearInterpolationDa
ta{Array{Float64,1},Array{Float64,1}},DiffEqBase.DEStats},1})
Closest candidates are:
  append!(!Matched::BitArray{1}, ::Any) at bitarray.jl:766
  append!(!Matched::Plots.Series, ::Any...) at /builds/JuliaGPU/DiffEqBench
marks.jl/.julia/packages/Plots/3INZP/src/utils.jl:839
  append!(!Matched::Array{T,1} where T, ::AbstractArray{T,1} where T) at ar
ray.jl:953
  ...
```

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

```
Error: UndefVarError: test_mc not defined
```

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
```

```
k was 1. The number of Adaptive Fails is 0. Elapsed time was 212.263834703
k was 2. The number of Adaptive Fails is 0. Elapsed time was 196.562096925
k was 3. The number of Adaptive Fails is 0. Elapsed time was 183.422153273
k was 4. The number of Adaptive Fails is 0. Elapsed time was 183.136726901
k was 5. The number of Adaptive Fails is 0. Elapsed time was 193.83005651
k was 6. The number of Adaptive Fails is 0. Elapsed time was 206.312577872
k was 7. The number of Adaptive Fails is 0. Elapsed time was 204.510439647
k was 8. The number of Adaptive Fails is 0. Elapsed time was 203.879277337
```

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,num_monte=num_runs,abstol=2.0^(-13),reltol=0,qmax=qs
    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
```

```
Problem 1
Error: MethodError: no method matching append!(::Nothing, ::Array{DiffEqBas
e.RODESolution{Float64,1,Array{Float64,1},Array{Float64,1},Dict{Symbol,Floa
t64},Array{Float64,1},DiffEqNoiseProcess.NoiseProcess{Float64,1,Float64,Flo
at64,Float64,Array{Float64,1},typeof(DiffEqNoiseProcess.WHITE_NOISE_DIST),t
ypeof(DiffEqNoiseProcess.WHITE_NOISE_BRIDGE),false,ResettableStacks.Resetta
bleStack{Tuple{Float64,Float64,Float64},false},ResettableStacks.ResettableS
tack{Tuple{Float64,Float64,Float64},false},DiffEqNoiseProcess.RSWM{Float64}
,Nothing,RandomNumbers.Xorshifts.Xoroshiro128Plus},DiffEqBase.SDEProblem{Fl
oat64,Tuple{Float64,Float64},false,DiffEqBase.NullParameters,Nothing,DiffEq
Base.SDEFunction{false,typeof(DiffEqProblemLibrary.SDEProblemLibrary.f_line
ar),typeof(DiffEqProblemLibrary.SDEProblemLibrary.σ_linear),LinearAlgebra.U
niformScaling{Bool},typeof(DiffEqProblemLibrary.SDEProblemLibrary.linear_an
alytic),Nothing,Nothing,Nothing,Nothing,Nothing,Nothing,Nothing,Nothing,Not
hing,Nothing,Nothing,Nothing},typeof(DiffEqProblemLibrary.SDEProblemLibrary
.σ_linear),Base.Iterators.Pairs{Union{},Union{},Tuple{},NamedTuple{(),Tuple
{}}},Nothing},StochasticDiffEq.SRIW1,StochasticDiffEq.LinearInterpolationDa
ta{Array{Float64,1},Array{Float64,1}},DiffEqBase.DEStats},1})
Closest candidates are:
  append!(!Matched::BitArray{1}, ::Any) at bitarray.jl:766
  append!(!Matched::Plots.Series, ::Any...) at /builds/JuliaGPU/DiffEqBench
marks.jl/.julia/packages/Plots/3INZP/src/utils.jl:839
  append!(!Matched::Array{T,1} where T, ::AbstractArray{T,1} where T) at ar
ray.jl:953
  ...
```

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

0.3 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","qmaxDetermination.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/AdaptiveSDE/Project.toml`
[78ddff82-25fc-5f2b-89aa-309469cbf16f] DiffEqMonteCarlo 0.15.1
[77a26b50-5914-5dd7-bc55-306e6241c503] DiffEqNoiseProcess 5.0.2
[a077e3f3-b75c-5d7f-a0c6-6bc4c8ec64a9] DiffEqProblemLibrary 4.8.0
[2dcacdae-9679-587a-88bb-8b444fb7085b] ParallelDataTransfer 0.5.0
[91a5bcdd-55d7-5caf-9e0b-520d859cae80] Plots 1.5.3
[789caeaf-c7a9-5a7d-9973-96adeb23e2a0] StochasticDiffEq 6.24.0
[8ba89e20-285c-5b6f-9357-94700520ee1b] Distributed
[9a3f8284-a2c9-5f02-9a11-845980a1fd5c] Random
```