Memory allocations test

April 11, 2023

1 Memory allocations test

1.1 Loading packages and functions

[1]:
```
using LinearAlgebra
using OrdinaryDiffEq
using Plots
```

[2]:
```
PATH_SRC="../../src_simd/"
include(string(PATH_SRC,"IRKGL_SIMD.jl"))
using .IRKGL_SIMD


PATH_SRC="../../src_seq/"
include(string(PATH_SRC,"IRKGL_SEQ.jl"))
using .IRKGL_SEQ
```

1.2 Definition of the N-body problem

In Nbody.jl below, the following functions are defined: NbodyEnergy(u,Gm), NbodyODE!(du,u,Gm,t), and NbodyODE1!(du,u,Gm,t), where

$$u = \begin{pmatrix} q_1 & v_1 \\ \vdots & \vdots \\ q_N & v_N \end{pmatrix} \in \mathbb{R}^{2\times 3\times N}, \quad Gm = (G\,m_1, \dots, G\,m_N) \in \mathbb{R}^N.$$

The energy, as a function of the positions $q_1, \dots, q_N \in \mathbb{R}^3$ and the velocities $v_1, \dots, v_N \in \mathbb{R}^3$ of the N bodies is:

$$\text{Energy}(q_1, \dots, q_N, v_1, \dots, v_N) = \frac{1}{2}\sum_{i=1}^{N} m_i \|v_i\|^2 - G \sum_{1\le i<j\le N} \frac{m_i m_j}{\|q_i - q_j\|}.$$

The ODE system of the N-body problem, as a function of the positions $q_1, \dots, q_N \in \mathbb{R}^3$ and the velocities $v_1, \dots, v_N \in \mathbb{R}^3$ of the N bodies is:

$$\begin{aligned} \frac{d}{dt} q_i &= v_i, \\ \frac{d}{dt} v_i &= G \sum_{j\neq i} \frac{m_j}{\|q_j - q_i\|^3} (q_j - q_i). \end{aligned}$$

This system of ODEs can be writen in compact form as

$$\frac{du}{dt} = f(t, u, Gm)$$

[3]:
```
PATH_ODES="../../ODEProblems/"

include(string(PATH_ODES,"Initial5Body.jl"))
include(string(PATH_ODES,"Nbody.jl"))
```

[3]:
```
NbodyODE! (generic function with 2 methods)
```

Back to the top

1.3 Initial value problem: 5-body problem (outer solar system)

We consider $N = 5$ bodies of the outer solar system: the Sun, Jupiter, Saturn, Uranus, and Neptune. The initial values u_{00} are taken from DE430, Julian day (TDB) 2440400.5 (June 28, 1969).

[4]:
```
u0, Gm, bodylist = Initial5Body(Float64)
u0_B, Gm_B, bodylist = Initial5Body(BigFloat)
q0=u0[:,:,1]
v0=u0[:,:,2]
dim=length(size(u0))

N = length(Gm)

show(bodylist)
E0=NbodyEnergy(u0,Gm)
```

```
["Sun" "Jupiter" "Saturn" "Uranus" "Neptune"]
```

[4]:
```
-9.522696242724855e-12
```

Back to the top

1.4 Memory allocations

[5]:
```
t0 = 0.
dt = 500.  # 500.

tF = 100*dt
tF2 = 2*tF
tF4 = 4*tF

prob = ODEProblem(NbodyODE!, u0, (t0,tF), Gm)
prob2 = ODEProblem(NbodyODE!, u0, (t0,tF2), Gm)
```

```
prob4 = ODEProblem(NbodyODE!, u0, (t0,tF4), Gm);
```

1.4.1 Sequential version

[6]:
```
alg_seq=IRKGL_Seq(s=8)

solx=solve(prob,alg_seq,dt=dt, save_everystep=false);
solx2=solve(prob2,alg_seq,dt=dt, save_everystep=false)
solx4=solve(prob4,alg_seq,dt=dt, save_everystep=false);
```

[7]:
```
println(solx.retcode,",tspan=",prob.tspan,",steps=",prob.tspan[2]/
  ↪dt,",length(solx.t)=",length(solx.t))
@time solve(prob,alg_seq,dt=dt, save_everystep=false);
```

```
Success,tspan=(0.0, 50000.0),steps=100.0,length(solx.t)=2
  0.004448 seconds (855 allocations: 57.227 KiB)
```

[8]:
```
println(solx2.retcode,",tspan=",prob2.tspan,",steps=",prob2.tspan[2]/
  ↪dt,",length(solx2.t)=",length(solx2.t))
@time solve(prob2,alg_seq,dt=dt, save_everystep=false);
```

```
Success,tspan=(0.0, 100000.0),steps=200.0,length(solx2.t)=2
  0.008994 seconds (1.16 k allocations: 69.727 KiB)
```

[9]:
```
println(solx4.retcode,",tspan=",prob4.tspan,",steps=",prob4.tspan[2]/
  ↪dt,",length(solx4.t)=",length(solx4.t))
@time solve(prob4,alg_seq,dt=dt, save_everystep=false);
```

```
Success,tspan=(0.0, 200000.0),steps=400.0,length(solx4.t)=2
  0.017920 seconds (1.75 k allocations: 94.727 KiB)
```

1.4.2 Simd version

[10]:
```
alg_simd=IRKGL_simd(s=8)


solx=solve(prob,alg_simd,dt=dt, save_everystep=false)
solx2=solve(prob2,alg_simd,dt=dt, save_everystep=false)
solx4=solve(prob4,alg_simd,dt=dt, save_everystep=false);
```

[11]:
```
println(solx.retcode,",tspan=",prob.tspan,",steps=",prob.tspan[2]/
  ↪dt,",length(solx.t)=",length(solx.t))
@time solve(prob,alg_simd,dt=dt, save_everystep=false);
```

```
Success,tspan=(0.0, 50000.0),steps=100.0,length(solx.t)=2
  0.001002 seconds (827 allocations: 55.117 KiB)
```

[12]:
```
println(solx2.retcode,",tspan=",prob2.tspan,",steps=",prob2.tspan[2]/
  ↪dt,",length(solx2.t)=",length(solx2.t))
@time solve(prob2,alg_simd,dt=dt, save_everystep=false);
```

```
Success,tspan=(0.0, 100000.0),steps=200.0,length(solx2.t)=2
  0.001967 seconds (1.13 k allocations: 67.617 KiB)
```

[13]:
```
println(solx4.retcode,",tspan=",prob4.tspan,",steps=",prob4.tspan[2]/
  ↪dt,",length(solx4.t)=",length(solx4.t))
@time solve(prob4,alg_simd,dt=dt, save_everystep=false);
```

```
Success,tspan=(0.0, 200000.0),steps=400.0,length(solx4.t)=2
  0.003403 seconds (1.73 k allocations: 92.617 KiB)
```

1.4.3 Vern9

[14]:
```
sol_Vern =solve(prob,Vern9(), dt=dt, save_everystep=false)
sol_Vern2 =solve(prob2,Vern9(), dt=dt, save_everystep=false)
sol_Vern4 =solve(prob4,Vern9(), dt=dt, save_everystep=false);
```

[15]:
```
println(sol_Vern.retcode,",tspan=",prob.tspan,",steps=",prob.tspan[2]/
  ↪dt,",length(sol_Vern.t)=",length(sol_Vern.t))
@time solve(prob,Vern9(),dt=dt, save_everystep=false);
```

```
Success,tspan=(0.0, 50000.0),steps=100.0,length(sol_Vern.t)=2
  0.000129 seconds (52 allocations: 8.781 KiB)
```

[16]:
```
println(sol_Vern2.retcode,",tspan=",prob2.tspan,",steps=",prob2.tspan[2]/
  ↪dt,",length(sol_Vern2.t)=",length(sol_Vern2.t))
@time solve(prob2,Vern9(),dt=dt, save_everystep=false);
```

```
Success,tspan=(0.0, 100000.0),steps=200.0,length(sol_Vern2.t)=2
  0.000229 seconds (52 allocations: 8.781 KiB)
```

[17]:
```
println(sol_Vern4.retcode,",tspan=",prob4.tspan,",steps=",prob4.tspan[2]/
  ↪dt,",length(sol_Vern4.t)=",length(sol_Vern4.t))
@time solve(prob4,Vern9(),dt=dt, save_everystep=false);
```

```
Success,tspan=(0.0, 200000.0),steps=400.0,length(sol_Vern4.t)=2
  0.000440 seconds (52 allocations: 8.781 KiB)
```

1.4.4 Errors in Energy

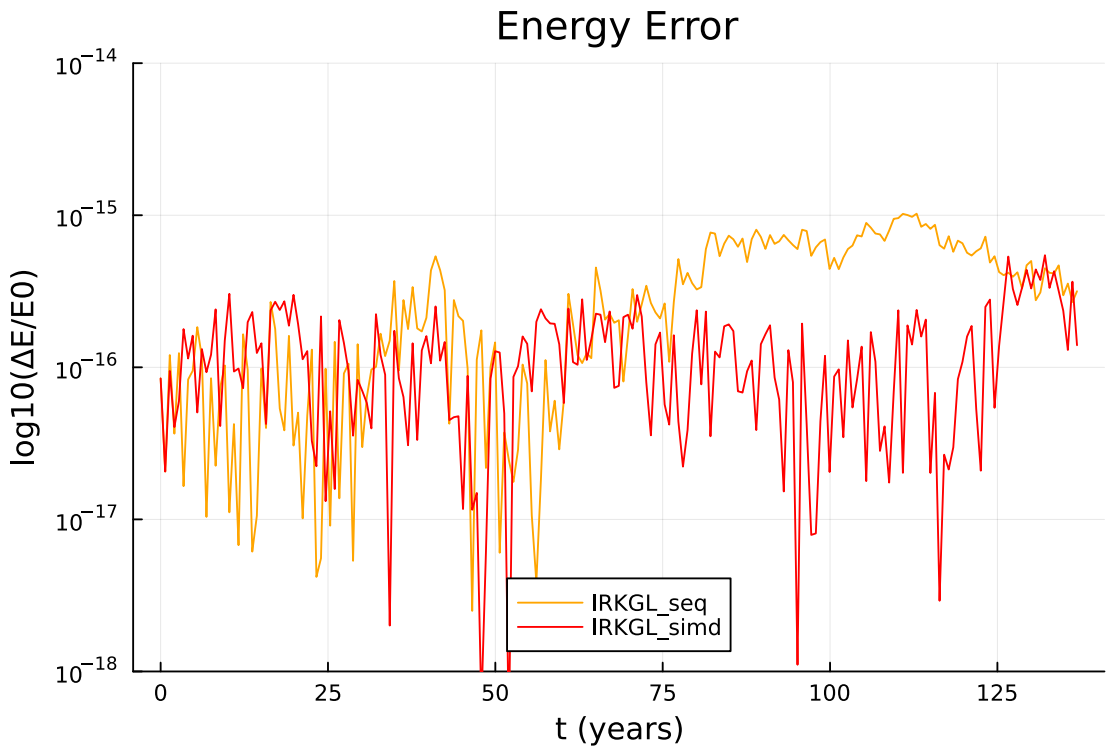
[18]:
```
sol1=solve(prob,alg_seq,dt=dt/2, adaptive=false)
println(sol1.retcode,",length(sol1.t)=",length(sol1.t))

sol2=solve(prob,alg_simd,dt=dt/2, adaptive=false)
println(sol2.retcode,",length(sol2.t)=",length(sol2.t))
```

```
Success,length(sol1.t)=201
Success,length(sol2.t)=201
```

[19]:
```
E0=NbodyEnergy(u0_B, Gm_B)
year=365.25
yrange=(1e-18,1e-14)
ΔE1 = map(x->NbodyEnergy(BigFloat.(x),Gm_B), sol1.u)./E0.-1
ΔE2 = map(x->NbodyEnergy(BigFloat.(x),Gm_B), sol2.u)./E0.-1
plot(title="Energy Error",xlabel="t (years)", ylabel="log10(ΔE/E0)",
     yscale=:log10, ylims=yrange, legend=:bottom)
plot!(sol1.t./year,abs.(ΔE1),label="IRKGL_seq", color="orange")
plot!(sol2.t./year,abs.(ΔE2),label="IRKGL_simd", color="red")
```

[19]:



[ ]: