Hamiltonian Problems in Julia

Mizuto Kadowaki

29th July 2020

Using the `HamiltonianProblem`, one only needs to define the Hamiltonian to give a solution for the generalized coordinates and momentum. The following is an introduction.

1 problem-setting

Consider a one-dimensional harmonic oscillator. The Hamiltonian is the following equation.

$$H(q,p) = \frac{p^2}{2m} + \frac{1}{2}m\omega q^2$$

where ω is the angular frequency. From here, we find $q(t)$ and $p(t)$.

2 package

The packages used are the following.

```
julia> using Plots

julia> using DifferentialEquationss
```

3 parameter

```
julia> m = 1 #mass
1

julia> k = rand() #spring constant
0.30543415624828363

julia>   = sqrt(k/m) #angular frequency
0.5526609776782541

julia> p0 = 10*rand() #p(0)
0.006736249180876452

julia> q0 = 10*rand() #q(0)
3.842205019632774
```

4 Hamiltonian

```
julia> H(p , q , param=Float64[]) = (p^2/(2*m)) + (1/2)*m* ^2*q^2
H (generic function with 2 methods)
```

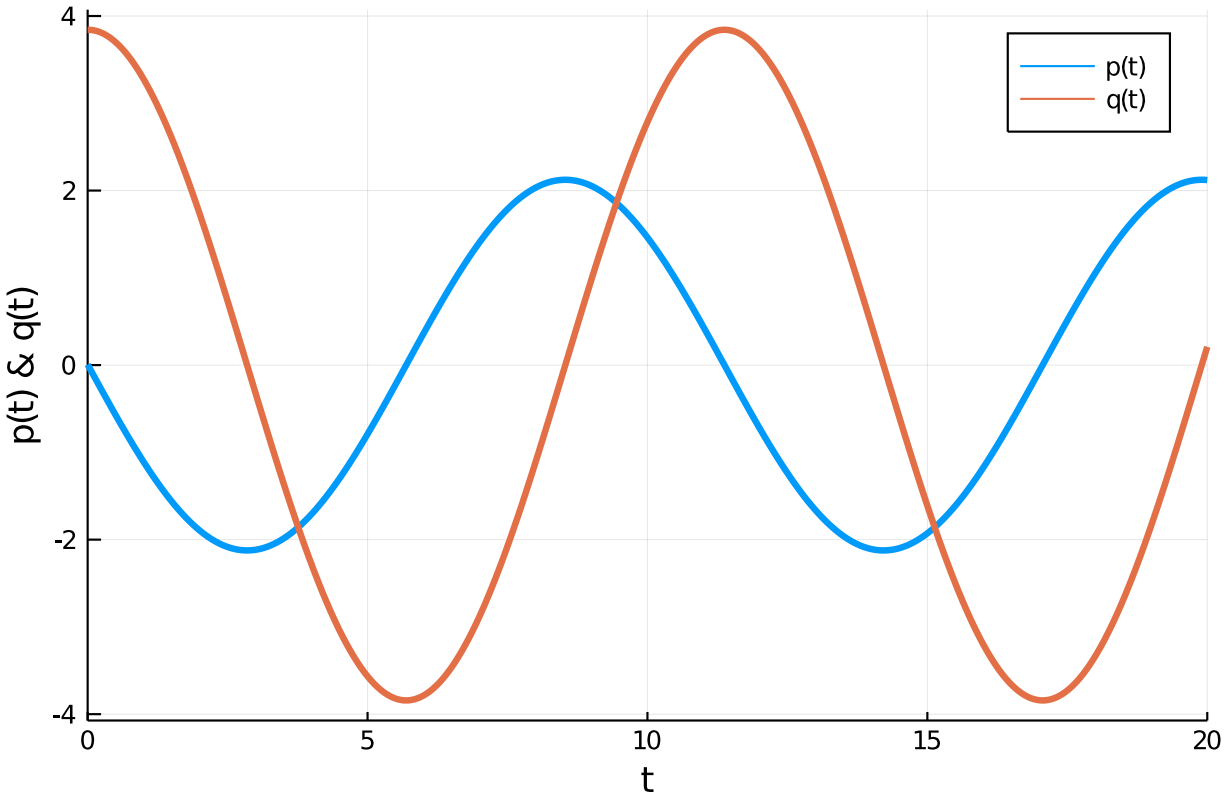
5 Solving

```
julia> prob = HamiltonianProblem(H , p0 , q0 , (0. , 20.))
ODEProblem with uType RecursiveArrayTools.ArrayPartition{Float64,Tuple{Float64,Float64}}
and tType Float64. In-place: false
timespan: (0.0, 20.0)
u0: 0.0067362491808764523.842205019632774

julia> sol = solve(prob , dt = 0.01)
retcode: Success
Interpolation: specialized 6th order interpolation
t: 21-element Array{Float64,1}:
  0.0
  0.01
  0.10999999999999999
  1.1099999999999999
  2.1933107026215417
  3.122623800410331
  4.036738380823853
  5.139993752891735
  6.327114074902692
  7.775317251222163

 10.975536289273403
 12.322514133855227
 13.673366176798647
 14.665096283623349
 15.655518228824686
 16.87732328025448
 18.153247268299875
 19.482027602366838
 20.0
u: 21-element Array{RecursiveArrayTools.ArrayPartition{Float64,Tuple{Float64,Float64}},1
}:
 0.0067362491808764523.842205019632774
 -0.004999200436127123.842213704898604
 -0.122286166942428553.835847816147301
 -1.21694338294701843.1486518513318673
 -1.98596879348776351.360026284268772
 -2.0990329220961863-0.5809521014339368
 -1.681430733822601-2.3465538197966453
 -0.6358138616522475-3.6659428742557156
 0.7320827920247048-3.606659880424913
 1.9402362821452377-1.5612446400967512

 0.464649648169535133.749104792562425
 -1.0621242128376533.3270378888871437
 -2.0283404602997941.1369702481741393
 -2.058642841255347-0.9419281066077786
 -1.4869455204602162-2.7429366743988726
 -0.21306308015964992-3.8228262342312083
 1.2071139448686858-3.16100439751629
 2.0666546287957868-0.882610495118218
```

```
2.12028904955982880.20921630711423803
```

6 Plot

```
julia> plot(sol , label = ["p(t)" "q(t)"] , ylabel = "p(t) & q(t)" , lw = 3)
```

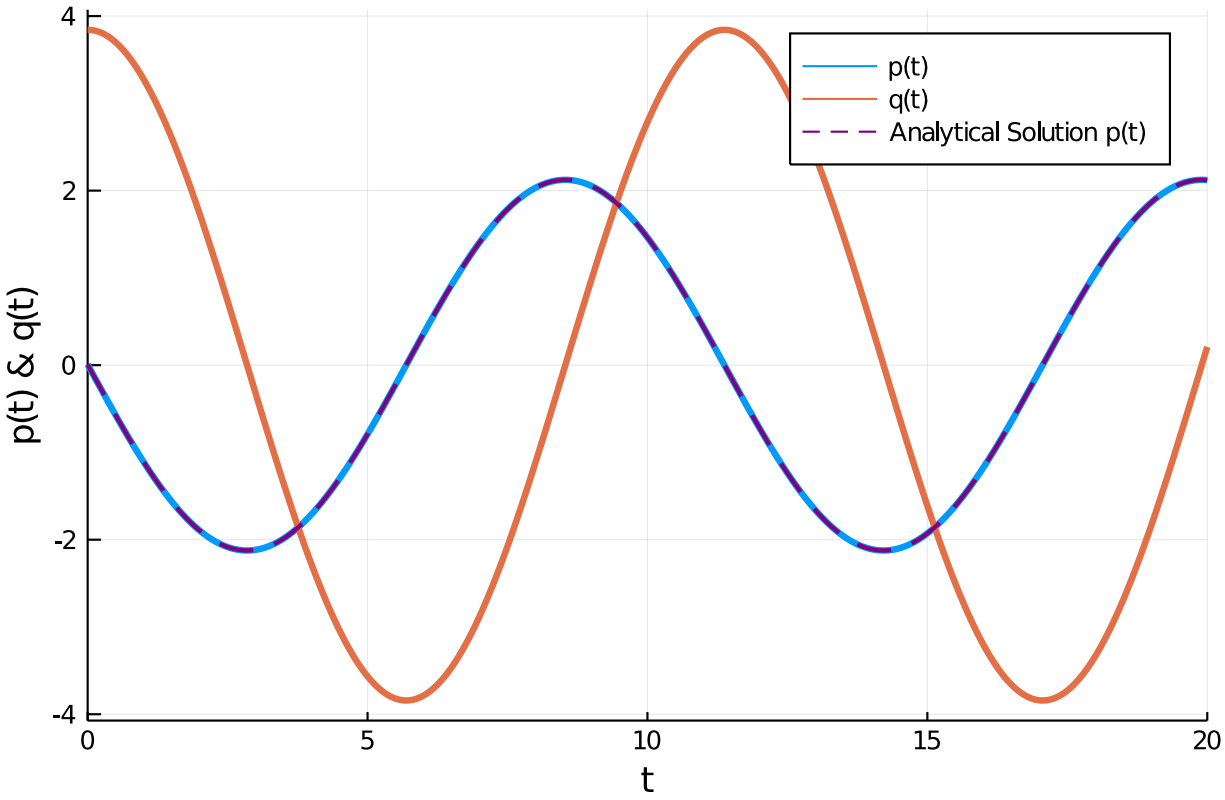


7 analytical solution

The analytical solution obtained by solving the differential equations is

$$q(t) = q(0)\cos\omega t + \frac{p(0)}{m\omega}\sin\omega t$$
$$p(t) = -m\omega q(0)\sin\omega t + p(0)\cos\omega t$$

```
julia> Q(t) = q0*cos( *t) + (p0/(m* ))*sin( *t) #q(t)
Q (generic function with 1 method)

julia> P(t) = -m* *q0*sin( *t) + p0*cos( *t) #p(t)
P (generic function with 1 method)

julia> t = 0:0.01:20
0.0:0.01:20.0

julia> plot!(t , P , ls =:dash , label = "Analytical Solution p(t)" , lw = 2 , color =
:purple)
```




```
julia> plot!(t , Q , ls =:dash , label = "Analytical Solution q(t)" , lw = 2 , color =
:green)
```