

Report for randomFrequency

Simulated with: lib.managers.crankNicolson.dimensionless

Simulation constants:

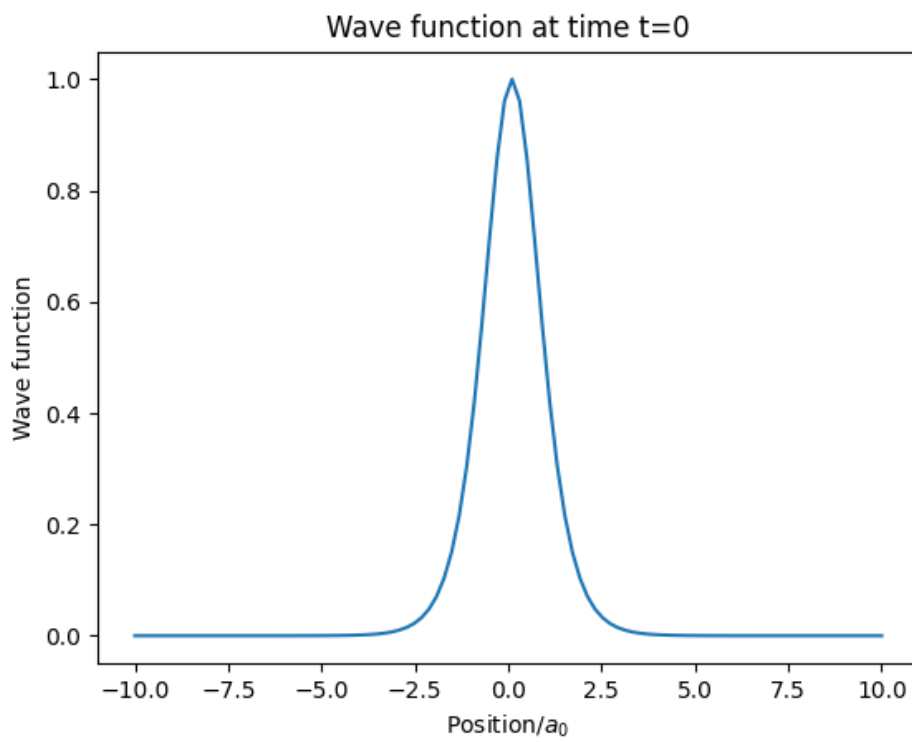
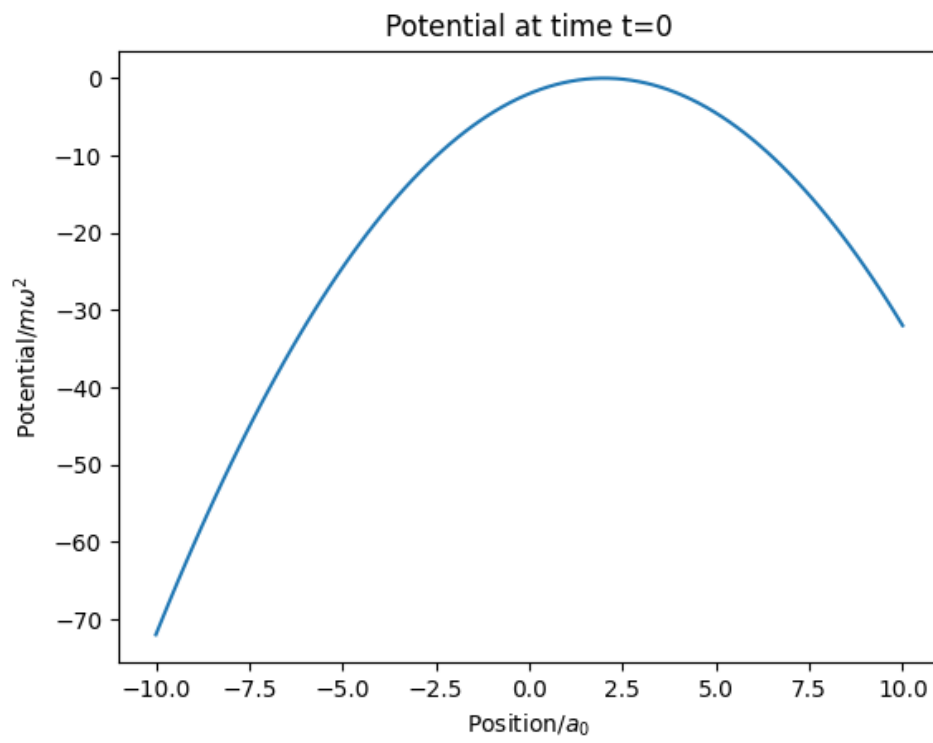
amplitude: 2.000	baseDensity: 1.000	chemicalPotential: 1.000
dt: 0.005	dx: 0.200	g: -1.000
hbar: 1.000	healingLength: 0.707	mass: 1.000
plotFPS: 1000.000	plotPause: 0.001	plotStep: 10
plotYMax: 2	plotYMin: -2	r: 0.125
tCount: 1000	tMax: 5	tMin: 0
velocity: 0.000	w0: 2.000	x0: 0.000
xCount: 100	xMax: 10	xMin: -10

Wave function:

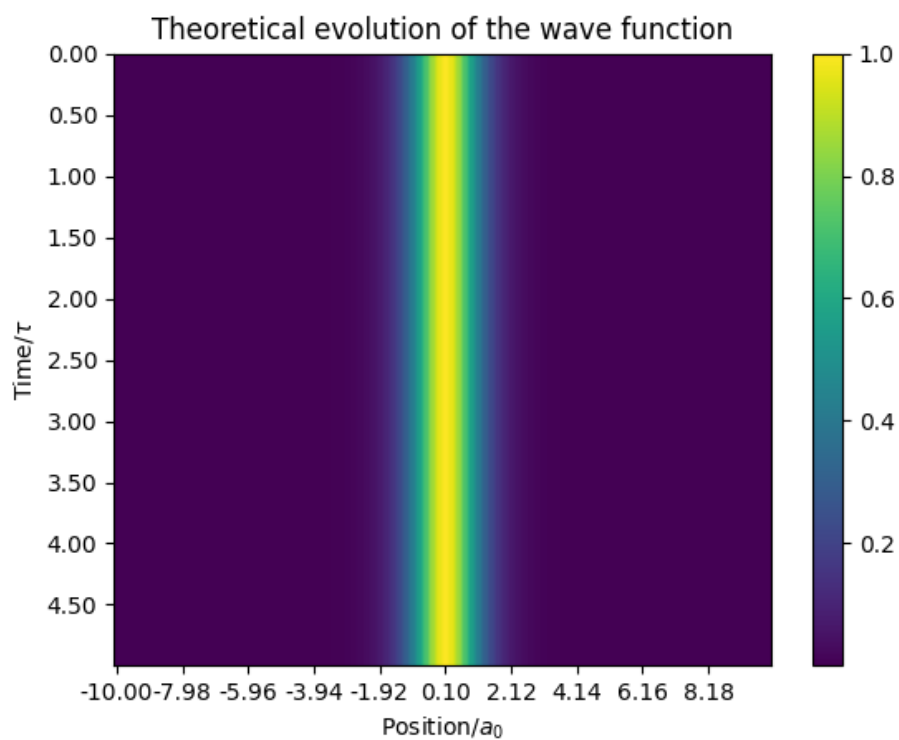
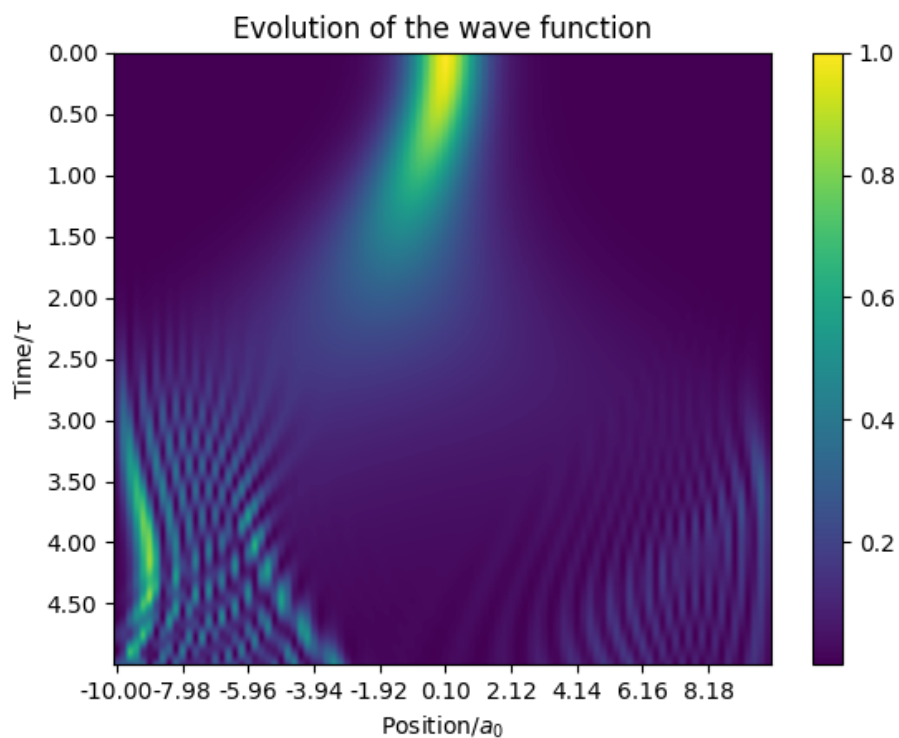
```
def brightSoliton(x, t, constants):    v = constants["velocity"]    g = constants["g"]    x0 = constants["x0"]    eta = jnp.sqrt((v**2 + 2) / (-2 * g))    kappa = jnp.sqrt(2 / (v**2 + 2))    spacePart = eta / jnp.cosh(((x - x0) - v * t) / kappa) * jnp.exp(1j * (x - x0) * v)    timePart = jnp.exp(1j * (1 / 2 - v**2 / 4) * t)    return spacePart * timePart
```

Potential function:

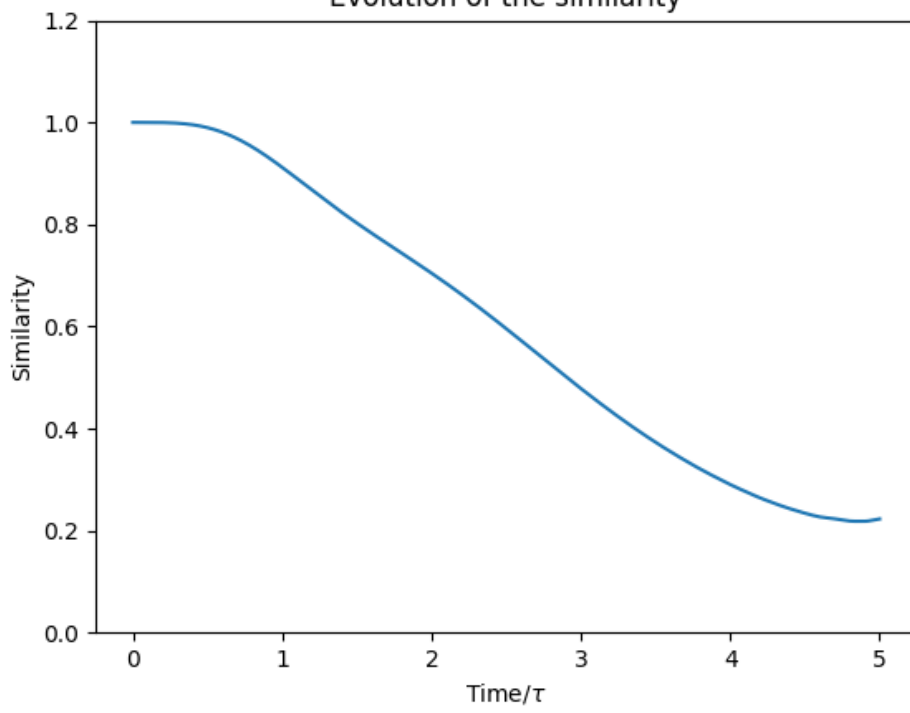
```
def V(x, t, constants):    x0 = constants["amplitude"] * jnp.cos(t * constants["w0"])    return -((x - x0) ** 2) / 2
```



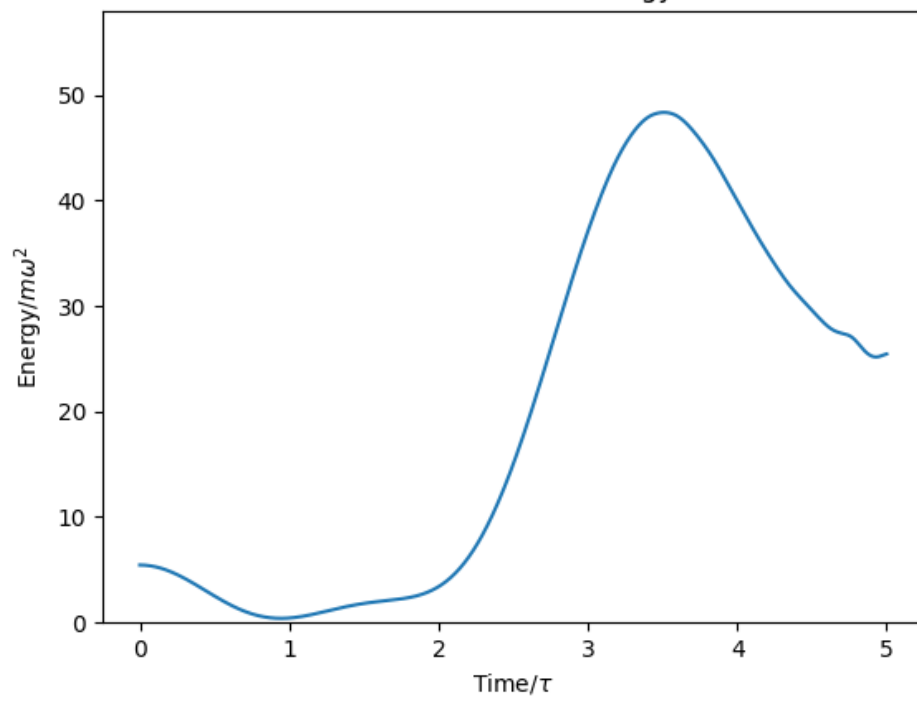
Results



Evolution of the similarity



Evolution of the energy



Evolution of the norm

