Adiabatic master equation with spin-fluctuators

Huo Chen

November 25, 2020

0.1 Model setup

In this example notebook, we solve a master equation of the form

$$\dot{\rho} = -i[-Z + \delta(s)Z, \rho] + \mathcal{L}(\rho)$$

via the quantum trajectories method. In the above expression, $\mathcal{L}$ is the Davies generator from the adiabatic master equation. $\delta(s)$ is a classical stochastic process generated by summing multiple telegraph processes:

$$\delta(t) = \sum_{i=1}^{N} T_i(t) \ ,$$

where $T_i(t)$ switches randomly between $\pm b_i$ with a rate γ_i. In the following code block, we show how to construct such a process:

```
using OrdinaryDiffEq, OpenQuantumTools
using Plots, StatsBase

H = DenseHamiltonian([(s)->1.0], [-σz], unit=:ħ)
u0 = PauliVec[1][1]

coupling = ConstantCouplings(["Z"], unit=:ħ)
# number of fluctuators
num = 10
# The values of b created here are in angular frequency units:
bvec = 0.01 * ones(num)
# γ_i
γvec = log_uniform(0.01, 1, num)
# create the fluctuator coupling interaction
fluctuator_ensemble = EnsembleFluctuator(bvec, γvec);
interaction_fluctuator = Interaction(coupling, fluctuator_ensemble)
# create the Ohmic coupling interaction
ohmic_bath = Ohmic(1e-4, 4, 16)
interaction_ohmic = Interaction(coupling, ohmic_bath)
# merge these two bath objects into `InteractionSet`
interactions = InteractionSet(interaction_fluctuator, interaction_ohmic)
annealing = Annealing(H, u0, interactions=interactions)
```

```
Annealing with hType QTBase.DenseHamiltonian{Complex{Float64}} and uType Ar
ray{Complex{Float64},1}
u0 with size: (2,)
```

0.2 Dynamics

We solve the dynamics and calculate $\langle X \rangle$ during the evolution:

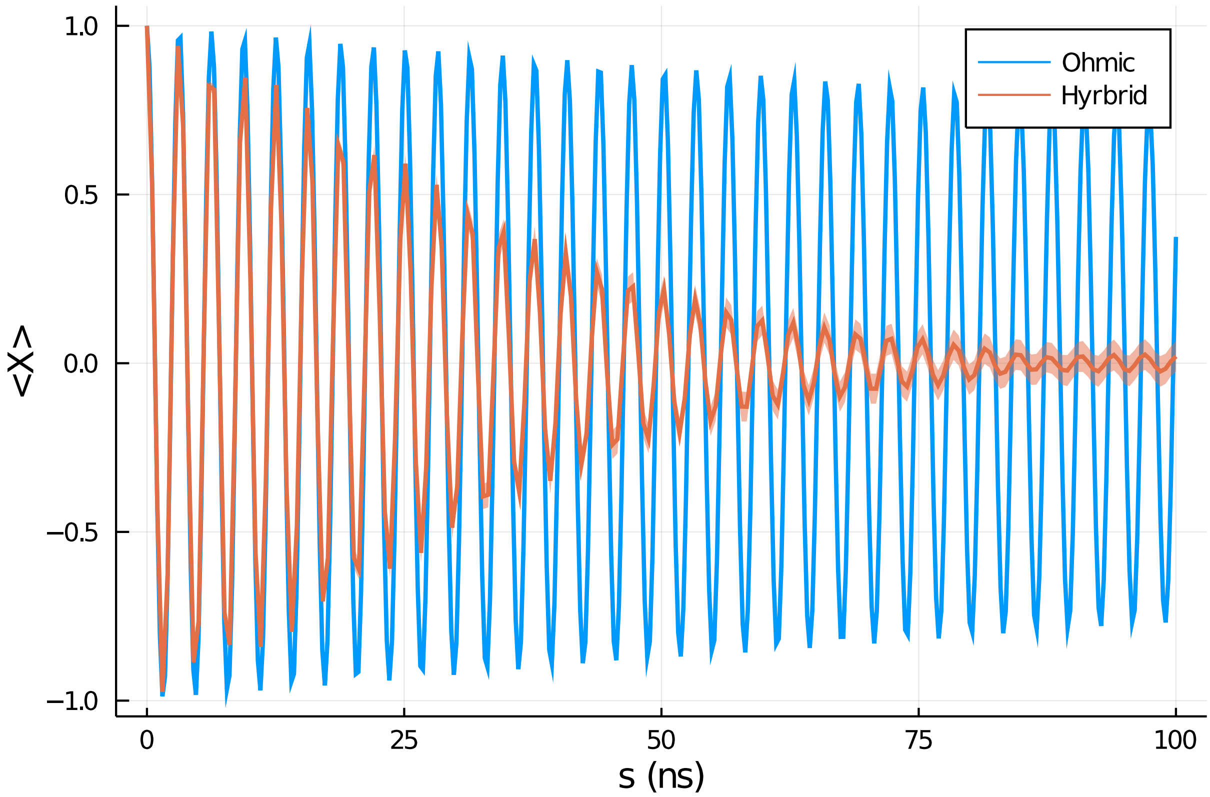
```
tf = 100
prob = build_ensembles(annealing, tf, :ame, ω_hint = range(-2, 2, length=100))
t_list = range(0,tf,length=200)
sol = solve(prob, Tsit5(), EnsembleSerial(), trajectories=1000, reltol=1e-6,
saveat=t_list)

dataset = []
st(s, so) = normalize(so(s, continuity=:left))
for s in t_list
    push!(dataset, [real(st(s, so)' * σx * st(s, so)) for so in sol])
end

pop_mean = []
pop_rmse = []
for data in dataset
    p_mean = sum(data)/1000
    p_rmse = sqrt(sum((x)->(x-p_mean)^2, data))/1000
    push!(pop_mean, p_mean)
    push!(pop_rmse, 2*p_rmse)
end
```

We also solve the dynamics with a pure Ohmic bath, i.e., the adiabatic master equation:

```
a_list = range(0,tf,length=400)
annealing_ame = Annealing(H, u0, coupling=coupling, bath=ohmic_bath)
sol_ame = solve_ame(annealing_ame, tf, alg=Tsit5(), ω_hint = range(-2, 2, length=100),
    reltol=1e-6, saveat=a_list)
ame_x = [real(tr(u*σx)) for u in sol_ame.u]
```

We compare $\langle X \rangle$ obtained using the above two models:

```
plot(a_list, ame_x, label="Ohmic", linewidth=2)
plot!(t_list, pop_mean, ribbon=pop_rmse, label="Hyrbrid", linewidth=2)
xlabel!("s (ns)")
ylabel!("<X>")
```



1.0
0.5
0.0
−0.5
−1.0
<X>
0
25
50
75
100
s (ns)
Ohmic
Hyrbrid