tested 190817 using *SpinDynamica* 3.0.1 under *Mathematica* 11.0

InversionRecovery

this shows the simulation of an inversion recovery pulse sequence for an ensemble of single spin-s-1/2. The main interest is how the relaxation superoperator may be thermalized to give the correct thermal equilibrium state.

```
Needs["SpinDynamica`"]
```

```
SpinDynamica version 3.0.1 loaded
```

ModifyBuiltIn: The following built-in routines have been modified in SpinDynamica:
{Chop, Dot, Duration, Exp, Expand, ExpandAll, NumericQ, Plus, Power, Simplify, Times, WignerD}.
Evaluate ??*symbol* to generate the additional definitions for *symbol*.

Inversion Recovery without thermalization

ideal infinitesimal pulses are represented by RotationSuperoperators

Set up

```
SetSpinSystem[1]
```

SetSpinSystem: the spin system has been set to $\{\{1, \frac{1}{2}\}\}$

SetBasis: the state basis has been set to ZeemanBasis$\left[\{\{1, \frac{1}{2}\}\}, \text{BasisLabels} \to \text{Automatic}\right]$.

define the inversion-recovery sequence.

Note the use of None to indicate a delay

a 10 ms delay has been added before the sequence, in order to see what happens more clearly

```
IRsequence[τ_] := {{None, 10 × 10^-3}, RotationSuperoperator[{π, 0}], {None, τ}}
```

define a relaxation superoperator

in this case the relaxation model corresponds to isotropic random fields

```
relaxationstrength = 1;
```

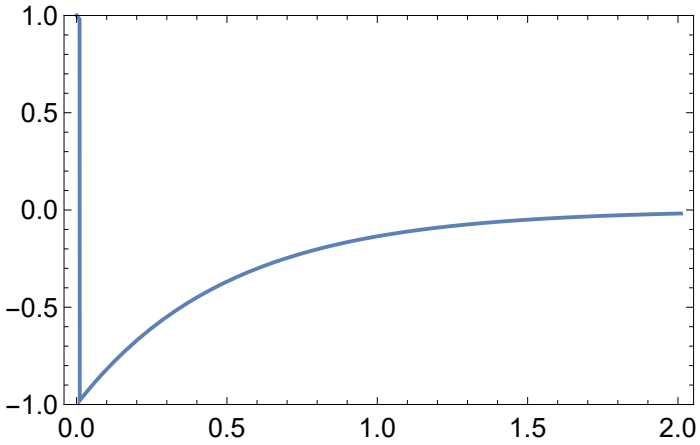
```
RelaxationSuperoperator = -relaxationstrength Sum[(-1)^m
    DoubleCommutationSuperoperator[opT[1, {1, m}], opT[1, {1, -m}]], {m, -1, 1}];
```

trajectory of z-magnetization, starting from z

```
Iztraj = Trajectory[
  opI["z"] → opI["z"],
  IRsequence[2],
  BackgroundGenerator -> RelaxationSuperoperator
 ]
```

```
TrajectoryFunction[ {{0, 2.01}} , <>]
```

```
Plot[Iztraj[t], {t, 0, EventDuration[IRsequence[2]]},
 Frame → True, PlotStyle → Thick, PlotRange → {-1, 1}]
```



note that the final state has zero magnetization. This is because the relaxation superoperator has not been thermalized

Inversion Recovery with thermalization

Set up

```
SetSpinSystem[1]
```

SetSpinSystem: the spin system has been set to $\{\{1, \frac{1}{2}\}\}$

define the inversion-recovery sequence.

Note the use of None to indicate a delay

a 10 ms delay has been added before the sequence starts, in order to see what happens more clearly

```
IRsequence[τ_] := {{None, 10 × 10^-3}, RotationSuperoperator[{π, 0}], {None, τ}}
```

define the Hamiltonian to use for thermalization

```
ω0 = -500 × 10^6 × 2 π;
```

```
H0 = ω0 opI["z"]
```

$-1\,000\,000\,000\,\pi\ \mathrm{I}_{1\,z}$

define the thermalized relaxation superoperator

```
? ThermalizeSuperoperator
```

ThermalizeSuperoperator[*sop*,*Hlab*,*Temperature*,*options*] corrects the eigenoperators of a superoperator so that a Boltzmann population distribution under the laboratory frame Hamiltonian *Hlab* is established at long evolution times. The syntax *ThermalizeSuperoperator*[*sop*,{*Hlab*,*Temperature*},*options*] may also be used. The *options* may include *HighTemperatureApproximation*→*False* or *HighTemperatureApproximation*→*True*.

```
relaxationstrength = 1;

RelaxationSuperoperator =
  -relaxationstrength Sum[ (-1) ^m
     DoubleCommutationSuperoperator[opT[1, {1, m}], opT[1, {1, -m}]], {m, -1, 1}];

ThermalizedRelaxationSuperoperator =
  ThermalizeSuperoperator[
   RelaxationSuperoperator,
   H0,
   Temperature → 300
  ];
```

SetOperatorBasis: the operator basis has been set to ShiftAndZOperatorBasis[{{1, 1/2}}, Sorted → CoherenceOrder].

define the thermal equilbrium density operator, and the thermal equilibrium magnetization

```
? ThermalEquilibriumDensityOperator
```

ThermalEquilibriumDensityOperator[*Hlab, Temperature*,*options*] sets up a thermal equilibrium density operator under the laboratory frame Hamiltonian *Hlab* at the specified temperature. The syntax *ThermalEquilibriumDensityOperator*[{*Hlab, Temperature*},*options*] may also be used. The *options* may include *HighTemperatureApproximation*→*False* or *HighTemperatureApproximation*→*True*.

```
ρeq = ThermalEquilibriumDensityOperator[H0, Temperature → 300]
```

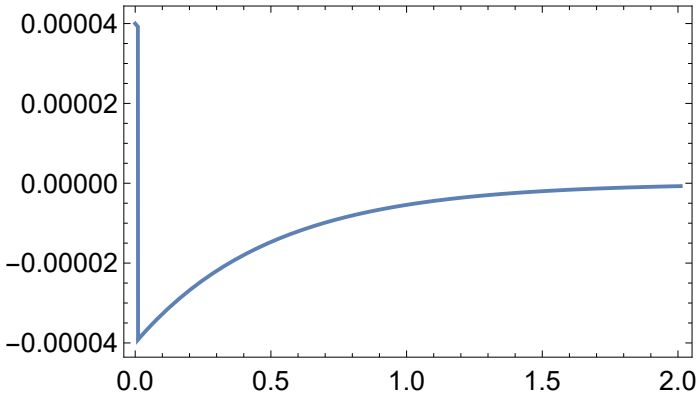
```
Operator[<< .. >> , OperatorType → Hermitian ]
```

trajectory of z-magnetization, starting from z

```
Iztraj = Trajectory[
  ρeq → opI["z"],
  IRsequence[2],
  BackgroundGenerator -> ThermalizedRelaxationSuperoperator
 ]
```

```
TrajectoryFunction[ {{0, 2.01}} , <>]
```

```
Plot[Iztraj[t], {t, 0, EventDuration[IRsequence[2]]},
 Frame → True, PlotStyle → Thick, PlotRange → All]
```



The trajectory may be normalized against thermal equilibrium magnetization as follows:

```
Iztraj = Trajectory[
  ρeq → opI["z"],
  IRsequence[2],
  BackgroundGenerator -> ThermalizedRelaxationSuperoperator,
  NormalizationFactor → OperatorAmplitude[ρeq → opI["z"]]
 ]
```

```
TrajectoryFunction[ {{0, 2.01}} , <>]
```

```
M0 = OperatorAmplitude[opI["z"], ρeq];
M0 // EngineeringForm
```

39.994×10^{-6}

```
Plot[Iztraj[t], {t, 0, EventDuration[IRsequence[2]]},
 Frame → True, PlotStyle → Thick, PlotRange → All]
```