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

Amplitude-Modulated Rf field

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

```
SetSpinSystem[1]
```

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

Amplitude-Modulated rf field simulated in the laboratory frame; no relaxation

this example shows the simulation of evolution under a time-dependent Hamiltonian. In this case a laboratory frame simulation is performed using an amplitude-modulated rf field

```
ω0 = 2 π 10^6
```

2 000 000 π

```
ωnut = 2 π 50 × 10^3
```

100 000 π

```
τ360 = 2 π / ωnut
```

$\frac{1}{50\,000}$

```
τ90 = τ360 / 4; τ180 = τ360 / 2;
```

the following syntax shows how to define a time-dependent Hamiltonian function, in this case a cosine modulated rf field

```
HamiltonianFunction = Function[t, ω0 opI["z"] + ωnut (2 opI["x"] Cos[ω0 t])]
```

Function[t, ω0 opI[z] + ωnut (2 opI[x] Cos[ω0 t])]

in this simple example, the time-dependent Hamiltonian is applied for the duration of a 180° pulse. In general, any sequence of events of various types may be defined.

```
events = {{HamiltonianFunction, τ180}}
```

{{Function[t, ω0 opI[z] + ωnut (2 opI[x] Cos[ω0 t])], $\frac{1}{100\,000}$}}

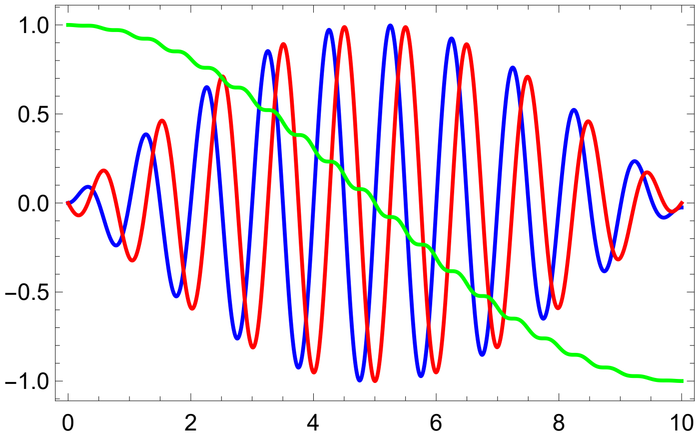
```
{Ixtraj, Iytraj, Iztraj} =
 Trajectory[opI["z"] -> {opI["x"], opI["y"], opI["z"]}, events]
```

{TrajectoryFunction[{{0, 10. × 10^{-6}}} , <>],
 TrajectoryFunction[{{0, 10. × 10^{-6}}} , <>], TrajectoryFunction[{{0, 10. × 10^{-6}}} , <>]}

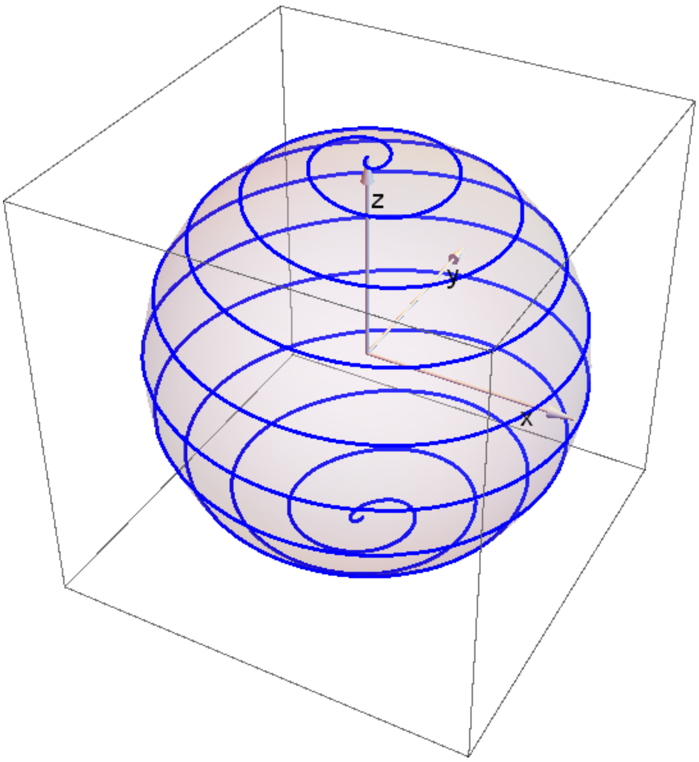
```
EventDuration[events]
```

$\frac{1}{100\,000}$

```
Plot[
 Evaluate[Through[{Ixtraj, Iytraj, Iztraj}[tμs × 10^-6]]],
 {tμs, 0, EventDuration[events] × 10^6 },
 Frame → True,
 PlotStyle → {{Thick, Blue}, {Thick, Red}, {Thick, Green}},
 LabelStyle → Directive[Medium, FontFamily → "Helvetica"]
]
```




```
Show[
 Graphics3D@{Opacity[0.2], Sphere[{0, 0, 0}, 1]},
 ParametricPlot3D[
  Through[{Ixtraj, Iytraj, Iztraj}[t]], {t, 0, EventDuration[events] },
  Boxed → True, Axes → None, PlotStyle → {{Thick, Blue}}
 ],
 Axes3D[]
]
```



Amplitude-Modulated rf field simulated in the laboratory frame; with relaxation in this case a strong relaxation superoperator is included in the BackgroundGenerator. Evolution over 5 complete nutations is simulated.

```
ω0 = 2 π 10^6;
ωnut = 2 π 50 × 10^3;
τ360 = 2 π / ωnut;
τ90 = τ360 / 4;  τ180 = τ360 / 2;
```

```
HamiltonianFunction = Function[t, ω0 opI["z"] + ωnut × (2 opI["x"] Cos[ω0 t])]
```

```
Function[t, ω0 opI[z] + ωnut (2 opI[x] Cos[ω0 t])]
```

the relaxation superoperator defined below corresponds to fluctuating random fields along the z-axis.

```
relaxationstrength = 10^5;
```

```
RelaxationSuperoperator =
  -relaxationstrength DoubleCommutationSuperoperator[opT[1, {1, 0}], opT[1, {1, 0}]]
```

$-100\,000$ DoubleCommutationSuperoperator$[I_{1z}, I_{1z}]$

```
events = {{HamiltonianFunction, 5 τ360}}
```

{{Function[t, ω0 opI[z] + ωnut (2 opI[x] Cos[ω0 t])], $\frac{1}{10\,000}$}}

```
traj = Trajectory[
   opI["z"] -> opI["z"],
   events,
   BackgroundGenerator -> RelaxationSuperoperator
  ]
```

TrajectoryFunction[{{0, $100. \times 10^{-6}$}} , <>]

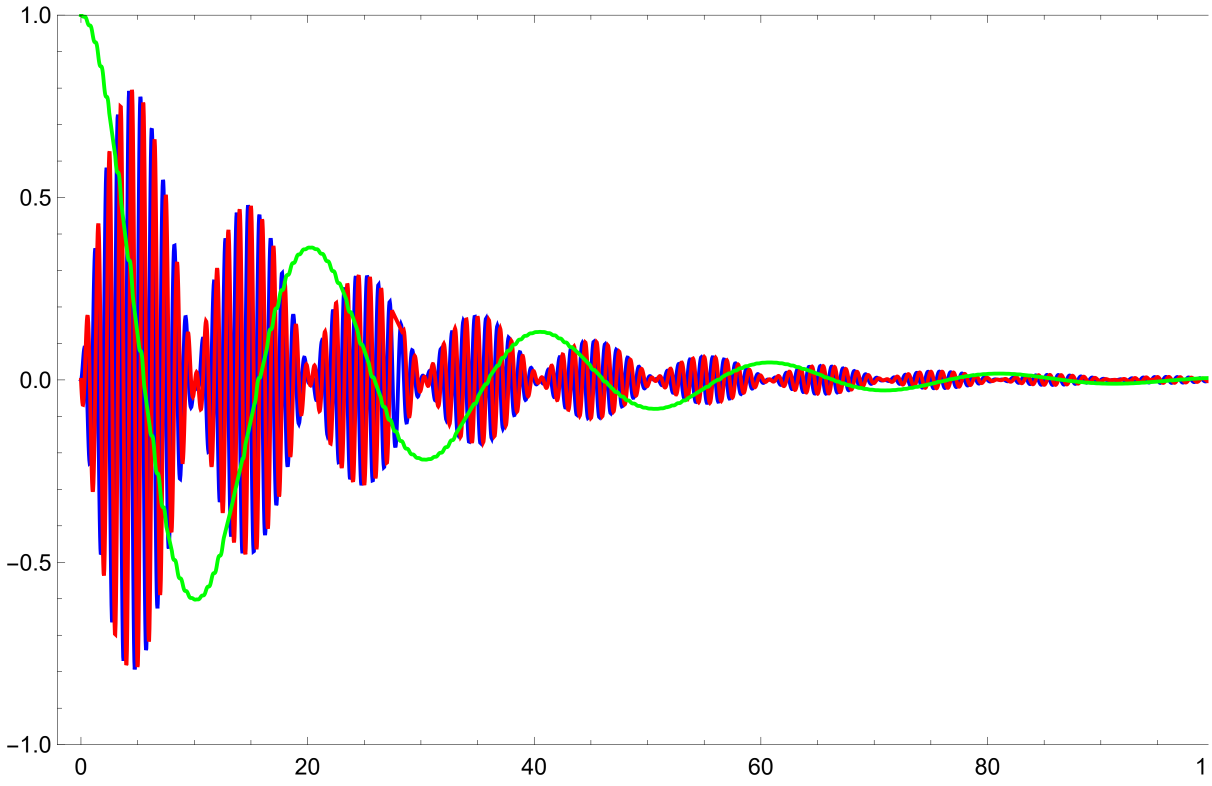
```
Trajectory[
 opI["z"] -> {opI["x"], opI["y"], opI["z"]},
 events,
 BackgroundGenerator -> RelaxationSuperoperator
]
```

{TrajectoryFunction[{{0, $100. \times 10^{-6}$}} , <>],
 TrajectoryFunction[{{0, $100. \times 10^{-6}$}} , <>],
 TrajectoryFunction[{{0, $100. \times 10^{-6}$}} , <>]}

```
{Ixtraj, Iytraj, Iztraj} =
 Trajectory[
   opI["z"] -> {opI["x"], opI["y"], opI["z"]},
   events,
   BackgroundGenerator -> RelaxationSuperoperator
  ]
```

{TrajectoryFunction[{{0, $100. \times 10^{-6}$}} , <>],
 TrajectoryFunction[{{0, $100. \times 10^{-6}$}} , <>],
 TrajectoryFunction[{{0, $100. \times 10^{-6}$}} , <>]}

```
Plot[Evaluate[Through[{Ixtraj, Iytraj, Iztraj}[tμs × 10^-6]]],
 {tμs, 0, EventDuration[events] × 10^6}, Frame → True,
 PlotStyle → {{Thick, Blue}, {Thick, Red}, {Thick, Green}},
 PlotRange → {-1, 1}, LabelStyle → Directive[Medium, FontFamily → "Helvetica"]]
```



```
Show[
 Graphics3D@{Opacity[0.2], Sphere[{0, 0, 0}, 1]},
 ParametricPlot3D[
  Through[{Ixtraj, Iytraj, Iztraj}[t]], {t, 0, EventDuration[events]},
  Boxed → True, Axes → None, PlotStyle → {{Thick, Blue}}
 ],
 Axes3D[]
]
```