

Tutorial: Basis Transformation

PyOR Version: Jeener_Gauri (release date not decided)

Author: Vineeth Thalakkotloor \ Email: vineethfrancis.physics@gmail.com

Import necessary packages and define source code of PyOR

```
In [74]: from IPython.display import display, HTML
display(HTML("<style>.container { width:100% !important; }</style>"))
import sys
sys.path.append('/media/HD2/Vineeth/PostDoc_Simulations/Github/PyOR_V1/Source')

import PythonOnResonance as PyOR

import time
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import rc
%matplotlib notebook
import sympy as sp
from sympy import *
```

Generating Spin System

```
In [75]: Slist1 = [1/2,1/2]
hbarEQ1 = True

System = PyOR.Numerical_MR(Slist1,hbarEQ1)
Sx,Sy,Sz = System.SpinOperator()
Sp,Sm = System.PMoperators(Sx,Sy)
```

Generating Product Operator Basis: PMZ Basis (Hilbert Space)

```
In [76]: # Product Operators Basis
sort = 'negative to positive'
Index = False
Normal = True
Basis_PMZ_D, coh_PMZ, dic_PMZ_D = System.ProductOperators_SpinHalf_PMZ(sort,Index,Normal)
OpB_H = System.String_to_Matrix(dic_PMZ_D, Basis_PMZ_D)

# For Comparing with spindynamica
Basis_PMZ = [OpB_H['Im1Im2'],OpB_H['Im1Iz2'],OpB_H['Iz1Im2'],OpB_H['Im1'],OpB_H['Im2'],OpB_H['Iz1'],OpB_H['Iz2'],OpB_H['Ip1Im2'],OpB_H['Ip1Iz2'],OpB_H['Iz1Ip2'],OpB_H['Ip1Ip2'],OpB_H['Ip2'],OpB_H['Ip1Ip2']]
dic_PMZ = ['Im1Im2','Im1Iz2','Iz1Im2','Im1','Im2','Im1Ip2','Ip1Im2','Iz1Iz2','Iz1','Iz2','Ip1Im2','Ip1Iz2','Iz1Ip2','Ip2','Ip1Ip2']
```

```
In [77]: Matrix(OpB_H["Im1Im2"])
```

Out[77]:

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1.0 & 0 & 0 & 0 \end{bmatrix}$$

Generating Zeeman Hamiltonian (Lab and Rotating Frame)

In [78]:

```
# Gyromagnetic Ratio
Gamma = [System.gammaH1, System.gammaH1]

# B0 Field in Tesla, Static Magnetic field (B0) along Z
B0 = 1.0

# Offset Frequency in Hz
Offset = [10.0, 20.0]

# generate Larmor Frequencies
LarmorF = System.LarmorFrequency(Gamma, B0, Offset)

# Rotating Frame Frequency
OmegaRF = [-System.gammaH1*B0, -System.gammaH1*B0]

# Lab Frame Hamiltonian
Hz_lab = System.Zeeman(LarmorF, Sz)

# Rotating Frame Hamiltonian
Hz = System.Zeeman_RotFrame(LarmorF, Sz, OmegaRF)
```

Larmor Frequency in MHz: [-42.57745869 -42.57746869]

Hamiltonian in Hilbert Space

In [79]:

```
Matrix(Hz/(2.0*np.pi))
```

Out[79]:

$$\begin{bmatrix} -15.0000000000126 & 0 & 0 & 0 \\ 0 & 5.00000000158526 & 0 & 0 \\ 0 & 0 & -5.00000000158526 & 0 \\ 0 & 0 & 0 & 15.0000000000126 \end{bmatrix}$$

Zeeman Basis State

In [80]:

```
Basis_Zeeman_state = System.ZBasis_H(Hz_lab)
```

In [81]:

Basis Zeeman state

Out[81]:

```
[array([1.,
        0.,
        0.,
        0.]),
 array([0.,
        1.,
        0.,
        0.]),
 array([0.,
        0.,
        0.,
        0.])]
```

```

[1.],
[0.]]),
array([[0.],
       [0.],
       [0.],
       [1.]])]
```

Singlet Triplet Basis State

```
In [82]: Basis_ST_state = System.STBasis(Hz_lab)
```

Basis: T_-, T_0, T_+, S_0

```
In [83]: Basis_ST_state
```

```
Out[83]: [array([[1.],
                [0.],
                [0.],
                [0.])),
          array([[0.],
                [0.70710678],
                [0.70710678],
                [0.]]),
          array([[0.],
                [0.],
                [0.],
                [1.])),
          array([[ 0.],
                [ 0.70710678],
                [-0.70710678],
                [ 0.]])]
```

Transformation Between Zeeman state and Singlelet state

```
In [84]: U_Z_ST = System.Transform_StateBasis(Basis_Zeeman_state, Basis_ST_state)
Matrix(U_Z_ST)
```

```
Out[84]: 
$$\begin{bmatrix} 1.0 & 0 & 0 & 0 \\ 0 & 0.707106781186547 & 0 & 0.707106781186547 \\ 0 & 0.707106781186547 & 0 & -0.707106781186547 \\ 0 & 0 & 1.0 & 0 \end{bmatrix}$$

```

```
In [85]: Matrix(System.State_BasisChange(Basis_Zeeman_state[3], U_Z_ST))
```

```
Out[85]: 
$$\begin{bmatrix} 0 \\ 0.707106781186547 \\ -0.707106781186547 \\ 0 \end{bmatrix}$$

```

J Coupling Hamiltonian

```
In [86]: Jlist = np.zeros((len(Slist1), len(Slist1)))
          Jlist[0][1] = 1
```

```
Hj = System.Jcoupling(Jlist, Sx, Sy, Sz)
Matrix(Hj)
```

```
Out[86]:
```

$$\begin{bmatrix} 1.5707963267949 & 0 & 0 & 0 \\ 0 & -1.5707963267949 & 3.14159265358979 & 0 \\ 0 & 3.14159265358979 & -1.5707963267949 & 0 \\ 0 & 0 & 0 & 1.5707963267949 \end{bmatrix}$$

Transformation of J Coupling Hamiltonian from Zeeman to Singlet Triplet

```
In [87]: Hj_ST = System.Operator_BasisChange(Hj, U_Z_ST)
Matrix(System.Matrix_Round(System.Matrix_Tol(Hj_ST/(2.0*np.pi), 1.0e-5), 2))
```

```
Out[87]:
```

$$\begin{bmatrix} 0.25 & 0 & 0 & 0 \\ 0 & 0.25 & 0 & 0 \\ 0 & 0 & 0.25 & 0 \\ 0 & 0 & 0 & -0.75 \end{bmatrix}$$

Basis Kets

```
In [88]: Kets = System.Basis_Ket()
Kets
```

```
Out[88]: ['|1/2, 1/2>|1/2, 1/2>',
'|1/2, 1/2>|1/2, -1/2>',
'|1/2, -1/2>|1/2, 1/2>',
'|1/2, -1/2>|1/2, -1/2>']
```

Basis Bras

```
In [89]: Bras = System.Basis_Bra()
Bras
```

```
Out[89]: ['<1/2, 1/2|<1/2, 1/2|',
'<1/2, 1/2|<1/2, -1/2|',
'<1/2, -1/2|<1/2, 1/2|',
'<1/2, -1/2|<1/2, -1/2|']
```

Product Operator Basis: Zeeman (Hilbert Space)

```
In [90]: Basis_Zeeman, dic_Zeeman = System.ProductOperators_Zeeman(Hz_lab)
```

```
In [91]: Matrix(Basis_Zeeman[0])
```

```
Out[91]:
```

$$\begin{bmatrix} 1.0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

```
In [92]: # Dictionary
dic_Zeeman
```

```
Out[92]: ['|1/2,1/2>|1/2,1/2><1/2,1/2|<1/2,1/2|',
'|1/2,1/2>|1/2,1/2><1/2,1/2|<1/2,-1/2|',
'|1/2,1/2>|1/2,1/2><1/2,-1/2|<1/2,1/2|',
'|1/2,1/2>|1/2,1/2><1/2,-1/2|<1/2,-1/2|',
'|1/2,1/2>|1/2,-1/2><1/2,1/2|<1/2,1/2|',
'|1/2,1/2>|1/2,-1/2><1/2,1/2|<1/2,-1/2|',
'|1/2,1/2>|1/2,-1/2><1/2,-1/2|<1/2,1/2|',
'|1/2,1/2>|1/2,-1/2><1/2,-1/2|<1/2,-1/2|',
'|1/2,-1/2>|1/2,1/2><1/2,1/2|<1/2,1/2|',
'|1/2,-1/2>|1/2,1/2><1/2,1/2|<1/2,-1/2|',
'|1/2,-1/2>|1/2,1/2><1/2,-1/2|<1/2,1/2|',
'|1/2,-1/2>|1/2,1/2><1/2,-1/2|<1/2,-1/2|',
'|1/2,-1/2>|1/2,-1/2><1/2,1/2|<1/2,1/2|',
'|1/2,-1/2>|1/2,-1/2><1/2,1/2|<1/2,-1/2|',
'|1/2,-1/2>|1/2,-1/2><1/2,-1/2|<1/2,1/2|',
'|1/2,-1/2>|1/2,-1/2><1/2,-1/2|<1/2,-1/2|']
```

Hamiltonian in Liouville Space (Zeeman basis)

```
In [93]: Hz_L = System.CommutationSuperoperator(Hz)
Matrix(System.Matrix_Round(Hz_L/(2.0*np.pi),2))
```

```
Out[93]:
```

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	-20.0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	-10.0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	-30.0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	20.0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	10.0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	-10.0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	10.0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	-10.0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	-20.0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	30.0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	10.0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	20.0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Product Operator Basis: Zeeman Liouville

```
In [94]: Basis_Zeeman_L = System.ProductOperators_ConvertToLiouville(Basis_Zeeman)
```

```
In [95]: Matrix(Basis_Zeeman_L[0])
```

```
Out[95]:
```

$$\begin{bmatrix} 1.0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Product Operator Basis: PMZ Liouvillie

```
In [96]: Basis_PMZ_L = System.ProductOperators_ConvertToLiouville(Basis_PMZ)
```

```
In [97]: Matrix(Basis_PMZ_L[0])
```

```
Out[97]:
```

Eigen Frequencies: Hilbert

```
In [98]: System.EigFreq_ProductOperator_H(Hz, OpB_H["Im1Im2"])
```

```
Out[98]: 30.0000000000025157
```

Eigen Frequencies: Liouvillie

```
In [99]: for i in range(System.Ldim):
        print("Eigen frequency of " + dic_PMZ[i] + "(" + str(coh_PMZ[i]) + ") :", System.EigFr
```

Eigen frequency of Im1Im2(-2) : 30.0000000000025157
Eigen frequency of Im1Iz2(-1) : 9.999999656141906
Eigen frequency of Iz1Im2(-1) : 19.999999317026997
Eigen frequency of Im1(-1) : 9.99999998427322
Eigen frequency of Im2(-1) : 20.00000000159783
Eigen frequency of Im1Ip2(0) : -10.000000003170511
Eigen frequency of Ip1Im2(0) : 10.000000003170511
Eigen frequency of Iz1Iz2(0) : 0.0
Eigen frequency of Iz1(0) : 0.0
Eigen frequency of Iz2(0) : 0.0
Eigen frequency of ID(0) : 0.0
Eigen frequency of Ip1Iz2(1) : -9.999999656141906
Eigen frequency of Iz1Ip2(1) : -19.999999317026997
Eigen frequency of Ip1(1) : -9.99999998427322
Eigen frequency of Ip2(1) : -20.00000000159783
Eigen frequency of Ip1Ip2(2) : -30.000000000025157

Basis Transformation Matrix: From Zeeman to PMZ Basis (Liouvillie Space)

```
In [100... U_Z_PMZ = System.Transform_StateBasis(Basis_Zeeman_L, Basis_PMZ_L)
```

```
In [101... Matrix(U_Z_PMZ)
```

```
Out[101... 
```

0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0.70710676908493	0	0.707106781186547	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0.70710676908493	0	0.707106781186547	0	0
0	0	0	0	0	-1.
0	0	0	0	0	0
0	0	0	0	0	0
1.0	0	0	0	0	0
0	-0.70710676908493	0	0.707106781186547	0	0
0	0	-0.70710676908493	0	0.707106781186547	0
0	0	0	0	0	0

```
In [102... Matrix(System.State_BasisChange(Basis_Zeeman_L[0], U_Z_PMZ))
```

```
Out[102...
```

```
[ 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 0
 1.0
 0
 0
 0]
```

Liouvillie Bracket

In [103...

```
System.Liouville_Bracket(Basis_Zeeman_L[0], Hz_L/(2.0*np.pi), Basis_Zeeman_L[0])
```

Out[103...

0.0

In [104...

```
System.Liouville_Bracket(Basis_PMZ_L[0], Hz_L/(2.0*np.pi), Basis_PMZ_L[0])
```

Out[104...

30.0000000000025157

Hamiltonian in Liouvillie Space (PMZ basis)

In [105...

```
Hz_L_PMZ = System.Operator_BasisChange(Hz_L, U_Z_PMZ)
```

In [106...

```
Matrix(System.Matrix_Round(System.Matrix_Tol(Hz_L_PMZ/(2.0*np.pi), 1.0e-5), 2))
```

Out[106...

30.0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	10.0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	20.0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	10.0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	20.0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	-10.0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	10.0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	-10.0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	-20.0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	-10.0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	-20.0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

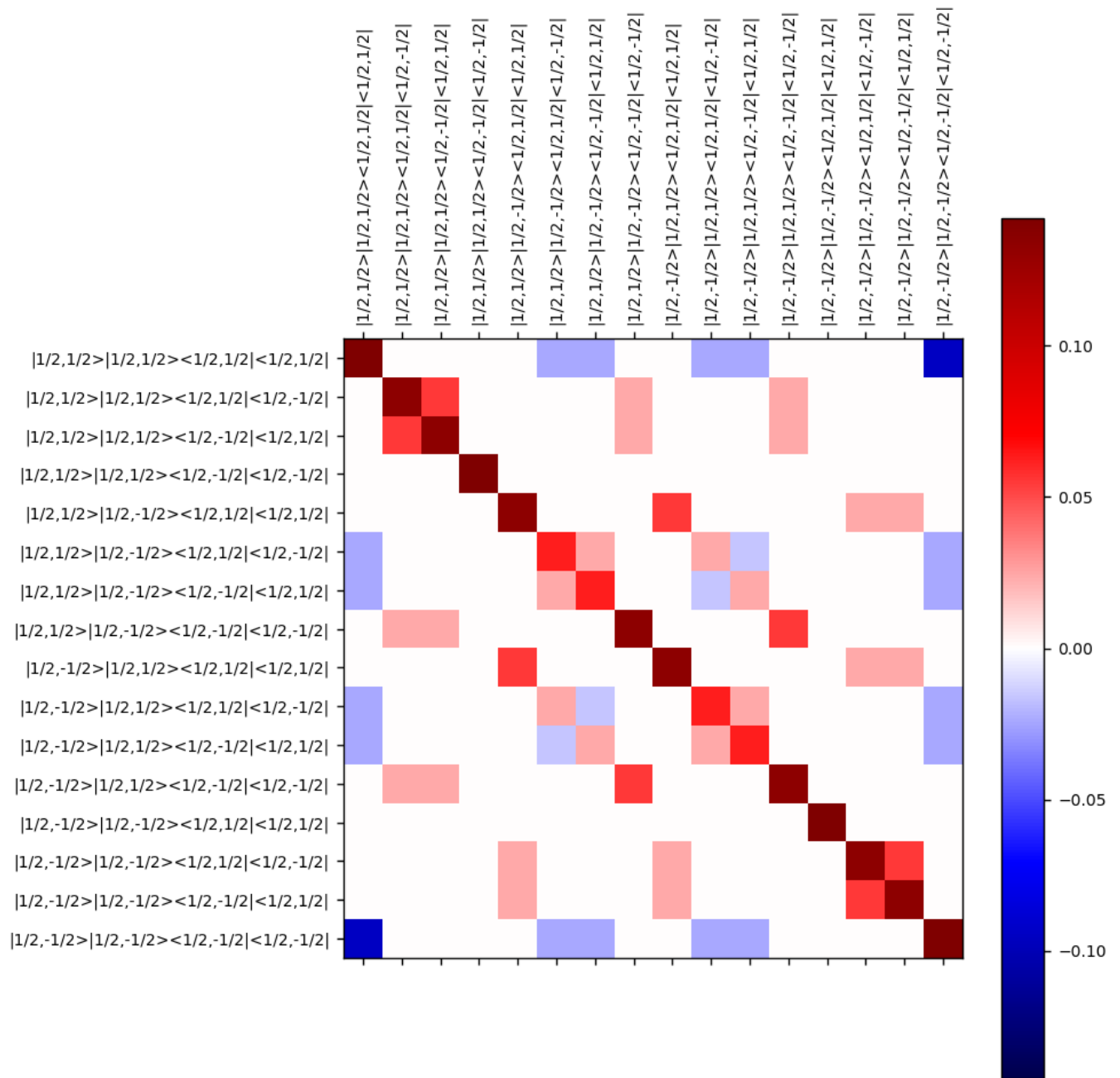
Dipolar Homonuclear Relaxation Superoperator

In [107...

```
R = None
Rprocess = "Auto-correlated Dipolar Homonuclear"
tau = 10.0e-12
bIS = [20.0e3]
System.Relaxation_Parameters(LarmorF, OmegaRF, tau, bIS)
R_L = System.Relaxation_L(Rprocess, R, Sx, Sy, Sz, Sp, Sm)
```

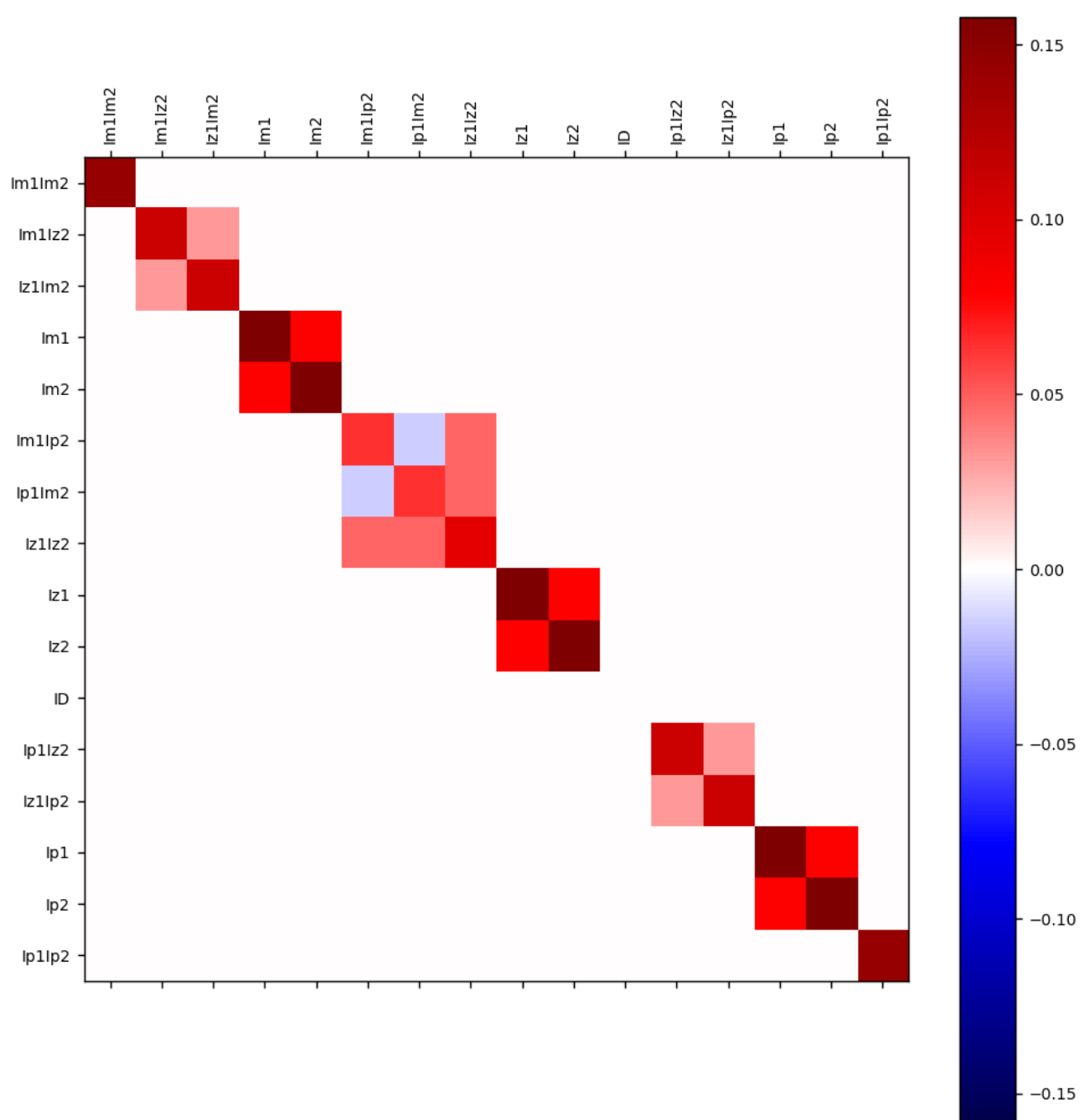
In [108...

```
# Relaxation Superoperator in Zeeman basis
System.MatrixPlot(1, R_L.real, dic_Zeeman, dic_Zeeman)
```



In [109...

```
# Relaxation Superoperator in PMZ basis
R_L_PMZ = System.OperatorBasisChange(R_L.real,U_Z_PMZ).real
System.MatrixPlot(2,R_L_PMZ,dic_PMZ,dic_PMZ)
```



Any suggestion? write to me

vineethfrancis.physics@gmail.com