Kapitza Pendulum

Modules

The following modules refer to the algebra

$\mathcal{L}_n = \{h_1, \ldots, h_n\}$,

that fulfils the commutation rule

$[h_i, h_j] = i\,\hbar \sum_{k=1}^{n} c_{i,j,k}\, h_k$,

characterized by the structure constants $c_{i,k,j}$.

LieGetMa

LieGetMa[c, J, vα] generates the M transformations of the for $M_k = e^{-Q_k}$ for $k = 1, \ldots,$

n and $M_{n+1} = M_1 \ldots M_n$ is an $n \times n$ matrix and M is an $n \times n \times n$ tensor corresponding to the structure constants c.

- **c** : $n \times n \times n$ tensor containing the structure constants,
- **J** : $n \times n$ matrix, $h' = Jh$ is a new representation of the $\mathcal{L}_\mathrm{n}$,
- **vα** : dimension n list $v\alpha = \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ containing the transformation parameters for the U_A transformation.

```
LieGetMa[c_, J_, vα_] := Module[{dim, k, Q, M},
 dim = Dimensions[c][[1]];
 k = Dimensions[J][[1]];
 M = Table[0, {k1, 1, k + 1}, {k2, 1, dim},
    {k3, 1, dim}];
 M[[k + 1]] = IdentityMatrix[k];
 Do[

   Q =
    Table[Sum[c[[k2, k3, k4]] J[[k1, k2]] vα[[k1]],
      {k2, 1, dim}], {k3, 1, dim}, {k4, 1, dim}];
  M[[k1]] = MatrixExp[-Q];
  M[[k + 1]] = M[[k + 1]].M[[k1]];
  , {k1, 1, k}];
 M
 ]
```

LieGetNu

LieGetNu[M,J] generates the *v* matrix where *v* is an n×n matrix.

- **M** : $n \times n \times n$ tensor containing the transformation matrices,
- **J** : $n \times n$ matrix, $h' = Jh$ is a new representation of the $\mathcal{L}_n$.

```
LieGetNu[M_, J_] := Module[{dim, Mk, Ik, νt, νt1},
  dim = Dimensions[M[[1]]][[1]];
  Ik = Normal[SparseArray[{{1, 1} → 1}, dim]];
  νt1 = Ik.J;
  Do[
   Mk = M[[k1]];
   Ik = Normal[SparseArray[{{k1, k1} → 1}, dim]];
   νt = νt1.Mk + Ik.J;
   νt1 = νt;
   , {k1, 2, dim}];
  Transpose[νt]
 ]
```

LieTrans

LieTrans[M, J, va, vα, k, t] transforms the coeficients l into va' under the k'th transformation U_k corresponding to the M_k matrix. Under this transformation the original Floquet operator $H - p_t = va^T h - p_t$ is transformed into $H' - p_t = U_k (H - p_t) U_k = va^T M_k h - \dot{\alpha}_k h_k - p_t = (va')^T h - p_t$ where va' is the new set of coefficients.

- **M** : $n \times n \times n$ tensor containing the transformation matrices,
- **J** : $n \times n$ matrix, $h' = Jh$ is a new representation of the $\mathcal{L}_n$,
- **va** : dimension n list $va = \{a_1, a_2, ..., a_n\}$ containing the coefficients of the original Floquet operator,

- **vα** : dimension n list $v\alpha = \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ containing the transformation parameters for the U_A transformation,
- **k** : integer that tags the number of transformation to be used,
- **t** : time parameter.

```
LieTrans[M_, J_, va_, vα_, k_, t_] :=
 va.M[[k]] - D[vα[[k]], t] J[[k]]
```

LieGetu

LieGetu[M, J, va, vα, t] transforms the original coefficients va into u under the complete transformation U_A corresponding to $M_a = M_1 M_2 \ldots M_n$. Under this transformation the original Floquet operator $H - p_t = va.h - p_t$ is transformed into $H' - p_t = U_A (H - p_t) U_A = va^T M_a h - \dot{\alpha}^T v^T h_k - p_t = u^T h - p_t$.

- **M** : is an $n \times n \times n$ tensor containing the transformation matrices,
- **J** : $n \times n$ matrix, $h' = Jh$ is a new representation of the $\mathcal{L}_n$,
- **va** : dimenison n list va = $\{a_1, a_2, \ldots, a_n\}$ containing the coefficients of the original Floquet operator,
- **vα** : dimension n list containing the transformation parameters for the U_A transformation. The α parameters must be functions of the time parameter t.
- t : time parameter.

```
LieGetu[M_, J_, va_, vα_, t_] := Module[{k, vu, vw},
  k = Dimensions[vα][[1]];
  vu = va;
  Do[
   vw = LieTrans[M, J, vu, vα, k1, t];
   vu = vw;
   , {k1, 1, k}];
  vu
 ]
```

LieGetDifEqLambda

LieGetDifEqLambda[J,*vi*,v*α*,v*β*,*λ*,ci] calculates a list containing the differential equations with respect to the auxiliary parameter λ that connects the α and β parameters.

- **J** : $n \times n$ matrix, $h' = J.h$ is a new representation of the $\mathcal{L}_n$,
- ***vi*** : inverse of the n×n matrix v calculated with LieGetNu[M,J],
- **v*α*** : dimension n list vα =
  $\{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ containing the transformation parameters for the U_A. The α parameters must be functions of the auxiliary parameter λ,
- **v*β*** : dimension n list vβ =
  $\{\beta_1, \beta_2, \ldots, \beta_n\}$ containing the transformation parameters for the U_B. The β parameters are NOT functions of the auxiliary parameter λ,
- ***λ*** : is the auxiliary parameter that helps relate the α and β transformation parameters.

ci : is a Boolean variable. If ci == True,
the initial conditions $\alpha_1[0] == 0$, $\alpha_2[0] == 0$, ...,
$\alpha_n[0] == 0$ is appended to the list of differential equations. If on ci == False then the output is just the list of differential equations,

- **λ** : is the auxiliary parameter that helps relate the α and β transformation parameters.

```
LieGetDifEqLambda[J_, νi_, vα_, vβ_, λ_, ci_] :=
 Module[{dim, v},
  dim = Dimensions[vα][[1]];
  v = νi.vβ;
  If[ci == True,
   Join[Table[D[vα[[k1]], λ] == v[[k1]], {k1, 1, dim}],
    Table[(vα[[k1]] /. λ → 0) == 0, {k1, 1, dim}]],
   Table[D[vα[[k1]], λ] == v[[k1]], {k1, 1, dim}]
  ]
 ]
```

Main Program

Definition of the structure constants $c_{i,j,k}$

The elements of this algebra are given by the operators $h_1 = 1$, $h_2 = x$, $h_3 = p$, $h_4 = m\,\omega_0{}^2\,x^2 + p^2$. The following lines define the algebra dimension and the structure constants.

```
n = 4;
d = Table[0, {k1, 1, n}, {k2, 1, n}, {k3, 1, n}];
d[[2, 3, 1]] = 1;
d[[3, 2, 1]] = -1;
d[[4, 2, 3]] = -2;
d[[2, 4, 3]] = 2;
d[[4, 3, 2]] = 2 m^2 ω0^2;
d[[3, 4, 2]] = -2 m^2 ω0^2;

R = IdentityMatrix[n];
Ri = Inverse[R];
c =
  Table[Sum[R[[k1, m1]] R[[k2, m2]] Ri[[m3, k3]]
      d[[m1, m2, m3]], {m1, 1, n}, {m2, 1, n},
     {m3, 1, n}], {k1, 1, n}, {k2, 1, n}, {k3, 1, n}];
```

Since J=$\mathcal{I}$, the structure of the algebra elements is preserved.

```
J = IdentityMatrix[n];
```

Derivation of the time differential equations for α_i(t)

We calculate u using Eq. (13). Notice that in this case vα = {$\alpha_1(t)$, ..., α_n(t)} is a function of time.

```
vα = Table[Subscript[α, k1][t], {k1, 1, n}];
va = Table[Subscript[a, k1], {k1, 1, n}];
Ma = LieGetMa[c, J, vα];
vu = LieGetu[Ma, J, va, vα, t];
MatrixForm[Simplify[ExpToTrig[vu]]]
```

$$\left(\begin{array}{c} a_1 - a_3\,\alpha_2[t] + a_4\,\alpha_2[t]^2 + a_2\,\alpha_3[t] + m^2\,a_4\,\omega_0^2\,\alpha_3[t]^2 \\ \mathrm{Cos}[2\,m\,\omega_0\,\alpha_4[t]]\,\left(a_2 + 2\,m^2\,a_4\,\omega_0^2\,\alpha_3[t] - \alpha_2'[t]\right) - m\,\mathrm{Sin}[2\,m\,\omega_0\, \\ \frac{\mathrm{Sin}[2\,m\,\omega_0\,\alpha_4[t]]\,\left(a_2 + 2\,m^2\,a_4\,\omega_0^2\,\alpha_3[t] - \alpha_2'[t]\right)}{m\,\omega_0} + \mathrm{Cos}[2\,m\,\omega_0\,\alpha_4[t]] \\ a_4 - \alpha_4'[t] \end{array}\right.$$

Compare these results with the ones in Eqs. (82)-(85).
The simplified differential equations for $\alpha_i(t)$ are obtained from Eq. (15)

```
ν = LieGetNu[Ma, J];
νi = Inverse[ν];
ℰ = Simplify[νi.vu];
MatrixForm[ℰ]
```

$$\left(\begin{array}{c} a_1 - a_3\,\alpha_2[t] + a_4\,\alpha_2[t]^2 - m^2\,a_4\,\omega_0^2\,\alpha_3[t]^2 - \alpha_1'[t] \\ a_2 + 2\,m^2\,a_4\,\omega_0^2\,\alpha_3[t] - \alpha_2'[t] \\ a_3 - 2\,a_4\,\alpha_2[t] - \alpha_3'[t] \\ a_4 - \alpha_4'[t] \end{array}\right)$$

Compare these results with the ones in Eqs. (87)-(90).

Relation between α(t) and β(t) via the solution of the λ differential equations

Using Eq. (16) we workout the λ differential equations for the $\alpha_i(\lambda,t)$ parameters. Note that in this case v$\alpha = \{\alpha_1(\lambda),\ \ldots,\ \alpha_n(\lambda)\}$ is a function of λ therefore, M_α and ν have to be recalculated. In **LieGetDifEqLamda** the condition **ci** is set to **True** in order to include the initial conditions.

```
vα = Table[Subscript[α, k1][λ], {k1, 1, n}];
Ma = LieGetMa[c, J, vα];
ν = LieGetNu[Ma, J];
νi = Inverse[ν];
vβ = Table[Subscript[β, k1], {k1, 1, n}];
difeqsλ =
  Simplify[
   ExpToTrig[LieGetDifEqLambda[J, νi, vα, vβ, λ,
     True]]];
MatrixForm[difeqsλ]
```

$$\begin{pmatrix} (\mathrm{Cos}[2\,m\,\omega_0\,\alpha_4[\lambda]]\,\beta_2 + m\,\mathrm{Sin}[2\,m\,\omega_0\,\alpha_4[\lambda]]\,\beta_3\,\omega_0)\,\alpha_3[\lambda] + \alpha_1{}'[\lambda] \\ \mathrm{Cos}[2\,m\,\omega_0\,\alpha_4[\lambda]]\,\beta_2 + m\,\mathrm{Sin}[2\,m\,\omega_0\,\alpha_4[\lambda]]\,\beta_3\,\omega_0 == \alpha_2{}'[\lambda] \\ \mathrm{Cos}[2\,m\,\omega_0\,\alpha_4[\lambda]]\,\beta_3 == \frac{\mathrm{Sin}[2\,m\,\omega_0\,\alpha_4[\lambda]]\,\beta_2}{m\,\omega_0} + \alpha_3{}'[\lambda] \\ \beta_4 == \alpha_4{}'[\lambda] \\ \alpha_1[0] == 0 \\ \alpha_2[0] == 0 \\ \alpha_3[0] == 0 \\ \alpha_4[0] == 0 \end{pmatrix}$$

Compare these results with Eqs. (95)-(98).

These equations are simple enough that we can attempt to solve them with **DSolve**.

```
sol = Simplify[DSolve[difeqsλ, vα, λ][[1]]]
```

$$\left\{\alpha_4[\lambda] \to \lambda\,\beta_4,\ \alpha_2[\lambda] \to \frac{\mathrm{Sin}[m\,\lambda\,\beta_4\,\omega_0]\,(\mathrm{Cos}[m\,\lambda\,\beta_4\,\omega_0]\,\beta_2 + m\,\mathrm{Sin}[m\,\lambda\,\beta_4\,\omega_0]\,\beta_3\,\omega_0)}{m\,\beta_4\,\omega_0},\right.$$
$$\alpha_3[\lambda] \to \frac{-2\,\mathrm{Sin}[m\,\lambda\,\beta_4\,\omega_0]^2\,\beta_2 + m\,\mathrm{Sin}[2\,m\,\lambda\,\beta_4\,\omega_0]\,\beta_3\,\omega_0}{2\,m^2\,\beta_4\,\omega_0^2},$$
$$\alpha_1[\lambda] \to \frac{1}{16\,m^3\,\beta_4^2\,\omega_0^3}\Big(-8\,m\,\mathrm{Cos}[2\,m\,\lambda\,\beta_4\,\omega_0]\,\mathrm{Sin}[m\,\lambda\,\beta_4\,\omega_0]^2\,\beta_2\,\beta_3\,\omega_0 - \beta_2^2\,(-4\,\mathrm{Sin}[2\,m\,\lambda\,\beta_4\,\omega_0] + \mathrm{Sin}[4\,m\,\lambda\,\beta_4\,\omega_0] + 4\,m\,\lambda\,\beta_4\,\omega_0) + m^2\,\omega_0^2\,\big(16\,m\,\lambda\,\beta_1\,\beta_4^2\,\omega_0 + \beta_3^2\,(\mathrm{Sin}[4\,m\,\lambda\,\beta_4\,\omega_0] - 4\,m\,\lambda\,\beta_4\,\omega_0)\big)\Big)\left.\vphantom{\frac{1}{1}}\right\}$$

Setting *λ*=1 we can obtain a relation between *α*(t) and *β*(t) of the form (18).

```
vα1 = Table[Subscript[α, k1], {k1, 1, n}];
eqs = Table[vα1[[k1]] == ((vα[[k1]] /. sol) /. {λ → 1}),
  {k1, 1, n}]
```

$$\left\{\alpha_1 == \frac{1}{16\,m^3\,\beta_4^2\,\omega_0^3}\Big(-8\,m\,\mathrm{Cos}[2\,m\,\beta_4\,\omega_0]\,\mathrm{Sin}[m\,\beta_4\,\omega_0]^2\,\beta_2\,\beta_3\,\omega_0 - \beta_2^2\,(-4\,\mathrm{Sin}[2\,m\,\beta_4\,\omega_0] + \mathrm{Sin}[4\,m\,\beta_4\,\omega_0] + 4\,m\,\beta_4\,\omega_0) + m^2\,\omega_0^2\,\big(16\,m\,\beta_1\,\beta_4^2\,\omega_0 + \beta_3^2\,(\mathrm{Sin}[4\,m\,\beta_4\,\omega_0] - 4\,m\,\beta_4\,\omega_0)\big)\Big),\right.$$
$$\alpha_2 == \frac{\mathrm{Sin}[m\,\beta_4\,\omega_0]\,(\mathrm{Cos}[m\,\beta_4\,\omega_0]\,\beta_2 + m\,\mathrm{Sin}[m\,\beta_4\,\omega_0]\,\beta_3\,\omega_0)}{m\,\beta_4\,\omega_0},$$
$$\alpha_3 == \frac{-2\,\mathrm{Sin}[m\,\beta_4\,\omega_0]^2\,\beta_2 + m\,\mathrm{Sin}[2\,m\,\beta_4\,\omega_0]\,\beta_3\,\omega_0}{2\,m^2\,\beta_4\,\omega_0^2},$$
$$\left.\alpha_4 == \beta_4\right\}$$

Relation between *α*(t) and *β*(t) via the eigenvalue one

eigenvectors of M_α^T

Working out the inverse relation between α and β might be difficult in this case given the complexity of the previous equations. Therefore, it is convenient to obtain a relation between α(t) and β(t) by obtaining the eigenvalue one eigenvectors of M_α^T. There are two eigenvalue one eigenvectors.

```
vα = Table[Subscript[α, k1], {k1, 1, n}];
Ma = LieGetMa[c, J, vα];
Mat = Transpose[Ma[[n + 1]]];
eval = Simplify[Eigenvalues[Mat]];
evec = Simplify[Eigenvectors[Mat]];
eval[[2]]
ρ1 = Simplify[ExpToTrig[evec[[2]]]]
eval[[3]]
ρ2 = Simplify[evec[[3]]]
vβ = γ1 ρ1 + γ2 ρ2
```

1

$\{0, -m\,\omega_0\,(-\mathrm{Cot}[m\,\alpha_4\,\omega_0]\,\alpha_2 + m\,\alpha_3\,\omega_0),\ \alpha_2 + m\,\mathrm{Cot}[m\,\alpha_4\,\omega_0]\,\alpha_3\,\omega_0,\ 1\}$

1

$\{1, 0, 0, 0\}$

$\{\gamma_2, -m\,\gamma_1\,\omega_0\,(-\mathrm{Cot}[m\,\alpha_4\,\omega_0]\,\alpha_2 + m\,\alpha_3\,\omega_0),\ \gamma_1\,(\alpha_2 + m\,\mathrm{Cot}[m\,\alpha_4\,\omega_0]\,\alpha_3\,\omega_0),\ \gamma_1\}$

Compare this last result with the one in Eq. (100).