Exact solution for quantum-classical hybrid mode of a classical harmonic oscillator quadratically coupled to a degenerate two-level quantum system

Settings

restart;
with(*Physics*) :

Hamiltonian
$alias(H = H(q, p))$:

$alias\left(f_1 = f_1(q), f_2 = f_2(q), f_3 = f_3(q)\right)$:

Wavefunction
$alias\left(\Upsilon_1 = \Upsilon_1(q, p, t), \Upsilon_2 = \Upsilon_2(q, p, t)\right)$:

$\Upsilon := Vector\left(\left[\Upsilon_1, \Upsilon_2\right]\right)$:

Pauli matrices
$alias\left(\sigma_1 = Matrix(Psigma[1]), \sigma_2 = Matrix(Psigma[2]), \sigma_3 = Matrix(Psigma[3])\right)$:

Quanutm-classical hybrid equation of motion for the Koopman wavefunction Υ

Hybrid equation of motion

$$HybridEq := -I\cdot\hbar\cdot\frac{\partial}{\partial t}\Upsilon - I\cdot\hbar\cdot\left(\frac{\partial}{\partial p}H\cdot\frac{\partial}{\partial q}\Upsilon - \frac{\partial}{\partial q}H\cdot\frac{\partial}{\partial p}\Upsilon - \left(\frac{\partial}{\partial q}f_1\cdot\sigma_1 + \frac{\partial}{\partial q}f_2\cdot\sigma_2 + \frac{\partial}{\partial q}f_3\cdot\sigma_3\right).\frac{\partial}{\partial p}\Upsilon\right) - \frac{1}{2}\cdot\left(q\cdot\frac{\partial}{\partial q}H\cdot\Upsilon + q\cdot\left(\frac{\partial}{\partial q}f_1\cdot\sigma_1 + \frac{\partial}{\partial q}f_2\cdot\sigma_2 + \frac{\partial}{\partial q}f_3\cdot\sigma_3\right).\Upsilon + p\cdot\frac{\partial}{\partial p}H\cdot\Upsilon\right) + H\cdot\Upsilon + \left(f_1\cdot\sigma_1 + f_2\cdot\sigma_2 + f_3\cdot\sigma_3\right).\Upsilon :$$

Hybrid density matrix expressed through the Koopman wave function

Define the matrix components

$$\mathrm{D}_{1,1} := 2\cdot\left|\Upsilon_{1}\right|^{2} + \Re\left(q\cdot\Upsilon_{1}\cdot\frac{\partial}{\partial q}conjugate\left(\Upsilon_{1}\right) + p\cdot\Upsilon_{1}\cdot\frac{\partial}{\partial p}conjugate\left(\Upsilon_{1}\right) + 2\cdot I\cdot\hbar\cdot\frac{\partial}{\partial q}\Upsilon_{1}\cdot\frac{\partial}{\partial p}conjugate\left(\Upsilon_{1}\right)\right):$$

$$\mathrm{D}_{1,2} := 2\cdot\Upsilon_{1}\cdot conjugate\left(\Upsilon_{2}\right) + I\cdot\hbar\cdot\left(\frac{\partial}{\partial q}\Upsilon_{1}\cdot\frac{\partial}{\partial p}conjugate\left(\Upsilon_{2}\right) - \frac{\partial}{\partial q}conjugate\left(\Upsilon_{2}\right)\cdot\frac{\partial}{\partial p}\Upsilon_{1}\right) + \frac{\Upsilon_{1}}{2}\cdot\left(q\cdot\frac{\partial}{\partial q}conjugate\left(\Upsilon_{2}\right) + p\cdot\frac{\partial}{\partial p}conjugate\left(\Upsilon_{2}\right)\right) + \frac{conjugate\left(\Upsilon_{2}\right)}{2}\cdot\left(q\cdot\frac{\partial}{\partial q}\Upsilon_{1} + p\cdot\frac{\partial}{\partial p}\Upsilon_{1}\right):$$

$$\mathrm{D}_{2,2} := 2\cdot\left|\Upsilon_{2}\right|^{2} + \Re\left(q\cdot\Upsilon_{2}\cdot\frac{\partial}{\partial q}conjugate\left(\Upsilon_{2}\right) + p\cdot\Upsilon_{2}\cdot\frac{\partial}{\partial p}conjugate\left(\Upsilon_{2}\right) + 2\cdot I\cdot\hbar\cdot\frac{\partial}{\partial q}\Upsilon_{2}\cdot\frac{\partial}{\partial p}conjugate\left(\Upsilon_{2}\right)\right):$$

Hybrid density matrix

$$d := \begin{bmatrix} \mathrm{D}_{1,1} & \mathrm{D}_{1,2} \\ conjugate\left(\mathrm{D}_{1,2}\right) & \mathrm{D}_{2,2} \end{bmatrix}:$$

Classical density

$$\rho_{classical} := \mathrm{D}_{1,1} + \mathrm{D}_{2,2}:$$

Density matrix for the quantum subsystem

$$\rho_{quant} := map\left(x \rightarrow Int\left(x, \left[q = -\infty..+\infty, p = -\infty..+\infty\right]\right), d\right):$$

Example 1: free particle classical Boltzmann state

$$ExampleFreeParticle := \left\{\Upsilon_{1} = \frac{1}{\sqrt{2}\cdot\sqrt[4]{2\cdot \mathrm{Pi}\cdot kT}}\cdot\exp\left(-\frac{p^{2}}{4\cdot kT} + \frac{I\cdot p\cdot q}{2\cdot\hbar}\right), \Upsilon_{2} = 0\right\}$$

$$ExampleFreeParticle := \left\{Upsi_{1} = \frac{2^{1/4}\,\mathrm{e}^{-\frac{p^{2}}{4\,kT} + \frac{\mathrm{I}\,p\,q}{2\,\hbar}}}{2\,\left(\pi\,kT\right)^{1/4}}, Upsi_{2} = 0\right\} \tag{3.1.1}$$

Get the hybrid density matrix

$simplify(subs(ExampleFreeParticle, d))$ assuming $p :: \mathbb{R}, q :: \mathbb{R}, \hbar > 0, kT > 0$

$$\begin{bmatrix} \frac{\sqrt{2}\, e^{-\frac{p^2}{2\,kT}}}{2\sqrt{\pi}\,\sqrt{kT}} & 0 \\ 0 & 0 \end{bmatrix} \tag{3.1.2}$$

Example 2: harmonic oscillator classical Boltzmann state

$$ExampleHarmonicOscillator := \left\{ \Upsilon_1 = subs\left(H = \frac{p^2}{2 \cdot m} + \frac{m \cdot \omega^2 \cdot q^2}{2}, \frac{\sqrt{\frac{kT \cdot \omega}{2 \cdot \mathrm{Pi}}}}{H} \cdot \sqrt{1 - \left(1 + \frac{H}{kT}\right) \cdot \exp\left(-\frac{H}{kT}\right)} \right), \Upsilon_2 = 0 \right\} :$$

Solving the special case of a classical harmonic oscillator quadratically coupled to a degenerate two-level quantum system

$$HarmonicOscEq := subs\left(H = \frac{p^2}{2 \cdot m} + \frac{m \cdot \omega^2 \cdot q^2}{2}, f_1 = \frac{\beta \cdot q^2}{2}, f_2 = 0, f_3 = 0, HybridEq \right) :$$

#HarmonicOscSolution:=pdsolve(HarmonicOscEq)

Analytically derived exact solution

Defining the unitary matrix U

$\_, U := LinearAlgebra{:}\text{-}Eigenvectors(\sigma_1)$

$$\_, U := \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \tag{4.1.1}$$

$$U := \frac{Dagger(U)}{\sqrt{2}}$$

$$U := \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{bmatrix} \tag{4.1.2}$$

Check that U diagonalize the coupling

$\lambda := U.\left(\sigma_1\right).Dagger(U)$

$$\lambda := \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{4.1.3}$$

$\lambda := ArrayTools\text{:-}Diagonal(\lambda)$

$$\lambda := \begin{bmatrix} 1 \\ -1 \end{bmatrix} \tag{4.1.4}$$

Introducing notations

$r_1 := \sqrt{p^2 + m \cdot \varkappa_1 \cdot q^2};\ r_2 := \sqrt{p^2 + m \cdot \varkappa_2 \cdot q^2};$

$$r_1 := \sqrt{m\ kappav_1\ q^2 + p^2}$$
$$r_2 := \sqrt{m\ kappav_2\ q^2 + p^2} \tag{4.1.5}$$

$\phi_1 := \arctan\left(\sqrt{m \cdot \varkappa_1} \cdot q, p\right);\ \phi_2 := \arctan\left(\sqrt{m \cdot \varkappa_2} \cdot q, p\right);$

$$\phi_1 := \arctan\left(\sqrt{m\ kappav_1}\ q, p\right)$$
$$\phi_2 := \arctan\left(\sqrt{m\ kappav_2}\ q, p\right) \tag{4.1.6}$$

$\varkappa_1 := m \cdot \omega^2 + \lambda[1] \cdot \beta;\ \varkappa_2 := m \cdot \omega^2 + \lambda[2] \cdot \beta$

$$kappav_1 := m\ \omega^2 + \beta$$
$$kappav_2 := m\ \omega^2 - \beta \tag{4.1.7}$$

$pq2polar1 := \left\{ q = \frac{r_1 \cdot \sin\left(\phi_1 - t \cdot \sqrt{\frac{\varkappa_1}{m}}\right)}{\sqrt{m \cdot \varkappa_1}}, p = r_1 \cdot \cos\left(\phi_1 - t \cdot \sqrt{\frac{\varkappa_1}{m}}\right) \right\}:$

$pq2polar2 := \left\{ q = \frac{r_2 \cdot \sin\left(\phi_2 - t \cdot \sqrt{\frac{\varkappa_2}{m}}\right)}{\sqrt{m \cdot \varkappa_2}}, p = r_2 \cdot \cos\left(\phi_2 - t \cdot \sqrt{\frac{\varkappa_2}{m}}\right) \right\}:$

$Y := \{$
$y_1(1) = eval(F_1(q,p), pq2polar1), y_1(2) = eval(F_1(q,p), pq2polar2),$
$y_2(1) = eval(F_2(q,p), pq2polar1), y_2(2) = eval(F_2(q,p), pq2polar2),$
$\}:$

In the following exact solution, $F_1:$ and $F_2:$ denotes for the initial condition for Υ :
$(subs(t=0, \Upsilon_1) = F_1(q,p), subs(t=0, \Upsilon_2) = F_2(q,p)) :$

Here is the sought exact solution

$ExSolution := \{$

$$\Upsilon_1 = eval\big(y_1(2) + \mathrm{abs}(U[1,1])^2 \cdot (y_1(1) - y_1(2)) + conjugate(U[1,1]) \cdot U[1,2] \cdot (y_2(1) - y_2(2)), Y\big),$$

$$\Upsilon_2 = eval\big(y_2(1) + \mathrm{abs}(U[2,2])^2 \cdot (y_2(2) - y_2(1)) + conjugate(U[1,2]) \cdot U[1,1] \cdot (y_1(1) - y_1(2)), Y\big)$$

$\}:$

Verifying the initial conditions

$simplify(eval(eval(\Upsilon_1, ExSolution), t=0) - F_1(q,p))$

$$0 \tag{4.1.8}$$

$simplify(eval(eval(\Upsilon_2, ExSolution), t=0) - F_2(q,p))$

$$0 \tag{4.1.9}$$

Verifying the exact solution

$simplify(pdetest(ExSolution, HarmonicOscEq))$ assuming $m > 0$

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{4.1.10}$$

Code generation for illustration

Specify the initial condition - the same as in Example 2: harmonic oscillator classical Boltzmann state

$$F_1 := (q,p) \rightarrow \frac{\sqrt{2}\sqrt{\frac{kT\,\omega}{\pi}}\sqrt{1-\left(1+\frac{\frac{p^2}{2\,m}+\frac{m\,\omega^2\,q^2}{2}}{kT}\right)\mathrm{e}^{-\frac{\frac{p^2}{2\,m}+\frac{m\,\omega^2\,q^2}{2}}{kT}}}}{2\left(\frac{p^2}{2\,m}+\frac{m\,\omega^2\,q^2}{2}\right)}:$$

$$F_2 := (q,p) \rightarrow 0:$$

$$ParamsToPlot := \left\{\hbar = 1, m = 1, \omega = 1, \beta = \frac{3}{4}\right\}:$$

$$SolutionToPlot := eval(ExSolution, ParamsToPlot):$$

One more (redundant) check that the obtain solution satisfies the equation

$$pdetest(SolutionToPlot, eval(HarmonicOscEq, ParamsToPlot))$$

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \textbf{(5.1)}$$

$CodeGeneration{:}\text{-}Python\left(eval\left(eval\left(\mathrm{D}_{1,1}, SolutionToPlot\right), ParamsToPlot\right), optimize, resultname = \text{"D11"}\right)$ assuming $q :: \mathbb{R}, p :: \mathbb{R}, t :: \mathbb{R}, kT > 0$

Warning, the function names {Re} are not recognized in the target language

```
t3 = math.sqrt(0.1e1 / math.pi * kT)
t4 = q ** 2
t6 = p ** 2
t7 = t4 / 4 + t6
t8 = math.sqrt(4)
t11 = math.atan2(q * t8 / 4, p)
t14 = -t11 + t8 * t / 4
t15 = math.cos(t14)
t16 = t15 ** 2
t17 = t16 * t7
t19 = math.sin(t14)
t20 = t19 ** 2
t21 = t20 * t7
t24 = 0.1e1 / kT
t25 = t24 * (t17 / 2 + 2 * t21)
t26 = 1 + t25
t27 = math.exp(-t25)
t30 = math.sqrt(-t27 * t26 + 1)
t33 = t17 + 4 * t21
t34 = 0.1e1 / t33
t37 = 0.7e1 / 0.4e1 * t4 + t6
t38 = math.sqrt(7)
t39 = t8 * t38
t42 = math.atan2(q * t39 / 4, p)
t46 = -t42 + t8 * t38 * t / 4
t47 = math.cos(t46)
t48 = t47 ** 2
t49 = t48 * t37
t51 = math.sin(t46)
```

```
t52 = t51 ** 2
t53 = t52 * t37
t56 = t24 * (t49 / 2 + 0.2e1 / 0.7e1 * t53)
t57 = 1 + t56
t58 = math.exp(-t56)
t61 = math.sqrt(-t58 * t57 + 1)
t64 = t49 + 0.4e1 / 0.7e1 * t53
t65 = 0.1e1 / t64
t68 = abs(t34 * t30 * t3 + t65 * t61 * t3)
t69 = t68 ** 2
t70 = math.sqrt(2)
t71 = t3 * t70
t77 = t34 * t30 * t71 / 2 + t65 * t61 * t71 / 2
t79 = t33 ** 2
t80 = 0.1e1 / t79
t81 = (t30).conjugate
t82 = t81 * t80
t83 = t16 * q
t86 = t8 * t15 * t7
t87 = 0.1e1 / p
t88 = 0.1e1 / t6
t89 = t88 * t4
t92 = 0.1e1 / (1 + t89 / 4)
t95 = t19 * t92 * t87 * t86
t97 = t20 * q
t99 = t83 / 2 - 0.3e1 / 0.2e1 * t95 + 2 * t97
t104 = 0.1e1 / t30 * t34
t107 = t83 / 4 - 0.3e1 / 0.4e1 * t95 + t97
t111 = t27 * t24
t116 = (t111 * t107 * t26 - t27 * t24 * t107) * t104 * t71 / 4
t117 = t64 ** 2
t118 = 0.1e1 / t117
t119 = (t61).conjugate
t120 = t119 * t118
t121 = t48 * q
t123 = t47 * t37
t129 = t51 / (1 + 0.7e1 / 0.4e1 * t89)
t131 = t129 * t87 * t8 * t38 * t123
t133 = t52 * q
t135 = 0.7e1 / 0.2e1 * t121 + 0.3e1 / 0.14e2 * t131 + 2 * t133
t140 = 0.1e1 / t61 * t65
t143 = 0.7e1 / 0.4e1 * t121 + 0.3e1 / 0.28e2 * t131 + t133
t147 = t58 * t24
t152 = (t147 * t143 * t57 - t58 * t24 * t143) * t140 * t71 / 4
t156 = t16 * p
t158 = t88 * q
t161 = t19 * t92 * t158 * t86
t163 = t20 * p
t171 = t156 + 0.3e1 / 0.4e1 * t161 + 4 * t163
t180 = t48 * p
t184 = t129 * t158 * t39 * t123
t186 = t52 * p
t194 = t180 - 0.3e1 / 0.28e2 * t184 + 0.4e1 / 0.7e1 * t186
t203 = -(2 * t156 + 0.3e1 / 0.2e1 * t161 + 8 * t163) * t82 * t71
/ 2 + (t111 * t171 * t26 - t27 * t24 * t171) * t104 * t71 / 4 -
(2 * t180 - 0.3e1 / 0.14e2 * t184 + 0.8e1 / 0.7e1 * t186) * t120
* t71 / 2 + (t147 * t194 * t57 - t58 * t24 * t194) * t140 * t71 /
4
t217 = Re((-t99 * t82 * t71 / 2 + t116 - t135 * t120 * t71 / 2 +
t152) * q * t77 + t203 * p * t77 + complex(0, 2) * t203 * (t116 -
t99 * t80 * t30 * t71 / 2 + t152 - t135 * t118 * t61 * t71 / 2))
D11 = t69 + t217
```

$CodeGeneration{:-}Python\Big(eval\Big(eval\Big(\mathrm{D}_{1,\,2}, SolutionToPlot\Big), ParamsToPlot\Big), optimize, resultname = \text{"D12"}\Big)$ assuming $q :: \mathbb{R}, p :: \mathbb{R}, t :: \mathbb{R}, kT > 0$

```
t1 = math.sqrt(2)
t4 = math.sqrt(0.1e1 / math.pi * kT)
t5 = t4 * t1
t6 = q ** 2
t8 = p ** 2
t9 = t6 / 4 + t8
t10 = math.sqrt(4)
t13 = math.atan2(q * t10 / 4, p)
t16 = -t13 + t10 * t / 4
t17 = math.cos(t16)
t18 = t17 ** 2
t19 = t18 * t9
t21 = math.sin(t16)
t22 = t21 ** 2
t23 = t22 * t9
t26 = 0.1e1 / kT
t27 = t26 * (t19 / 2 + 2 * t23)
t28 = 1 + t27
t29 = math.exp(-t27)
t32 = math.sqrt(-t29 * t28 + 1)
t34 = t19 + 4 * t23
t35 = 0.1e1 / t34
t37 = t35 * t32 * t5
t39 = 0.7e1 / 0.4e1 * t6 + t8
t40 = math.sqrt(7)
t41 = t10 * t40
t44 = math.atan2(q * t41 / 4, p)
t48 = -t44 + t10 * t40 * t / 4
t49 = math.cos(t48)
t50 = t49 ** 2
t51 = t50 * t39
t53 = math.sin(t48)
t54 = t53 ** 2
t55 = t54 * t39
t58 = t26 * (t51 / 2 + 0.2e1 / 0.7e1 * t55)
t59 = 1 + t58
t60 = math.exp(-t58)
t63 = math.sqrt(-t60 * t59 + 1)
t65 = t51 + 0.4e1 / 0.7e1 * t55
t66 = 0.1e1 / t65
t68 = t66 * t63 * t5
t70 = t37 / 2 + t68 / 2
t73 = (t68 / 2 - t37 / 2).conjugate
t77 = 0.1e1 / t32 * t35
t78 = t18 * q
t81 = t10 * t17 * t9
t82 = 0.1e1 / p
t83 = 0.1e1 / t8
t84 = t83 * t6
t87 = 0.1e1 / (1 + t84 / 4)
t90 = t21 * t87 * t82 * t81
t92 = t22 * q
t93 = t78 / 4 - 0.3e1 / 0.4e1 * t90 + t92
t97 = t29 * t26
t102 = (-t29 * t26 * t93 + t97 * t93 * t28) * t77 * t5 / 4
t103 = t34 ** 2
t104 = 0.1e1 / t103
```

```
t105 = t104 * t32
t109 = t78 / 2 - 0.3e1 / 0.2e1 * t90 + 2 * t92
t114 = 0.1e1 / t63 * t66
t115 = t50 * q
t117 = t49 * t39
t123 = t53 / (1 + 0.7e1 / 0.4e1 * t84)
t125 = t123 * t82 * t10 * t40 * t117
t127 = t54 * q
t128 = 0.7e1 / 0.4e1 * t115 + 0.3e1 / 0.28e2 * t125 + t127
t132 = t60 * t26
t137 = (t132 * t128 * t59 - t60 * t26 * t128) * t114 * t5 / 4
t138 = t65 ** 2
t139 = 0.1e1 / t138
t140 = t139 * t63
t144 = 0.7e1 / 0.2e1 * t115 + 0.3e1 / 0.14e2 * t125 + 2 * t127
t148 = t102 - t109 * t105 * t5 / 2 + t137 - t144 * t140 * t5 / 2
t149 = (t63).conjugate
t150 = t149 * t139
t151 = t50 * p
t154 = t83 * q
t156 = t123 * t154 * t41 * t117
t158 = t54 * p
t160 = 2 * t151 - 0.3e1 / 0.14e2 * t156 + 0.8e1 / 0.7e1 * t158
t166 = t151 - 0.3e1 / 0.28e2 * t156 + 0.4e1 / 0.7e1 * t158
t174 = (t132 * t166 * t59 - t60 * t26 * t166) * t114 * t5 / 4
t175 = (t32).conjugate
t176 = t175 * t104
t177 = t18 * p
t181 = t21 * t87 * t154 * t81
t183 = t22 * p
t185 = 2 * t177 + 0.3e1 / 0.2e1 * t181 + 8 * t183
t191 = t177 + 0.3e1 / 0.4e1 * t181 + 4 * t183
t199 = (-t29 * t26 * t191 + t97 * t191 * t28) * t77 * t5 / 4
t200 = -t160 * t150 * t5 / 2 + t174 + t185 * t176 * t5 / 2 - t199
t208 = -t144 * t150 * t5 / 2 + t137 + t109 * t176 * t5 / 2 - t102
t215 = t199 - t185 * t105 * t5 / 2 + t174 - t160 * t140 * t5 / 2
D12 = 2 * t73 * t70 + complex(0, 1) * (t200 * t148 - t215 * t208)
+ (t200 * p + t208 * q) * t70 / 2 + (t215 * p + t148 * q) * t73 /
2
```

CodeGeneration:-*Python*$\left(eval\left(eval\left(\mathrm{D}_{2,2}, SolutionToPlot\right), ParamsToPlot\right), optimize, resultname = \text{"D22"}\right)$ assuming $q :: \mathbb{R}, p :: \mathbb{R}, t :: \mathbb{R}, kT > 0$

Warning, the function names {Re} are not recognized in the target language

```
t3 = math.sqrt(0.1e1 / math.pi * kT)
t4 = q ** 2
t6 = p ** 2
t7 = 0.7e1 / 0.4e1 * t4 + t6
t8 = math.sqrt(7)
t9 = math.sqrt(4)
t10 = t9 * t8
t13 = math.atan2(q * t10 / 4, p)
t17 = -t13 + t9 * t8 * t / 4
t18 = math.cos(t17)
t19 = t18 ** 2
t20 = t19 * t7
t22 = math.sin(t17)
t23 = t22 ** 2
t24 = t23 * t7
t27 = 0.1e1 / kT
```

```
t28 = t27 * (t20 / 2 + 0.2e1 / 0.7e1 * t24)
t29 = 1 + t28
t30 = math.exp(-t28)
t33 = math.sqrt(-t30 * t29 + 1)
t36 = t20 + 0.4e1 / 0.7e1 * t24
t37 = 0.1e1 / t36
t40 = t4 / 4 + t6
t43 = math.atan2(q * t9 / 4, p)
t46 = -t43 + t9 * t / 4
t47 = math.cos(t46)
t48 = t47 ** 2
t49 = t48 * t40
t51 = math.sin(t46)
t52 = t51 ** 2
t53 = t52 * t40
t56 = t27 * (t49 / 2 + 2 * t53)
t57 = 1 + t56
t58 = math.exp(-t56)
t61 = math.sqrt(-t58 * t57 + 1)
t64 = t49 + 4 * t53
t65 = 0.1e1 / t64
t68 = abs(t37 * t33 * t3 - t65 * t61 * t3)
t69 = t68 ** 2
t70 = math.sqrt(2)
t71 = t3 * t70
t77 = t37 * t33 * t71 / 2 - t65 * t61 * t71 / 2
t79 = t36 ** 2
t80 = 0.1e1 / t79
t81 = (t33).conjugate
t82 = t81 * t80
t83 = t19 * q
t85 = t18 * t7
t87 = 0.1e1 / p
t89 = 0.1e1 / t6
t90 = t89 * t4
t94 = t22 / (1 + 0.7e1 / 0.4e1 * t90)
t96 = t94 * t87 * t9 * t8 * t85
t98 = t23 * q
t100 = 0.7e1 / 0.2e1 * t83 + 0.3e1 / 0.14e2 * t96 + 2 * t98
t105 = 0.1e1 / t33 * t37
t108 = 0.7e1 / 0.4e1 * t83 + 0.3e1 / 0.28e2 * t96 + t98
t112 = t30 * t27
t117 = (t112 * t108 * t29 - t30 * t27 * t108) * t105 * t71 / 4
t118 = t64 ** 2
t119 = 0.1e1 / t118
t120 = (t61).conjugate
t121 = t120 * t119
t122 = t48 * q
t125 = t9 * t47 * t40
t128 = 0.1e1 / (1 + t90 / 4)
t131 = t51 * t128 * t87 * t125
t133 = t52 * q
t135 = t122 / 2 - 0.3e1 / 0.2e1 * t131 + 2 * t133
t140 = 0.1e1 / t61 * t65
t143 = t122 / 4 - 0.3e1 / 0.4e1 * t131 + t133
t147 = t58 * t27
t152 = (t147 * t143 * t57 - t58 * t27 * t143) * t140 * t71 / 4
t156 = t19 * p
t159 = t89 * q
t161 = t94 * t159 * t10 * t85
t163 = t23 * p
t171 = t156 - 0.3e1 / 0.28e2 * t161 + 0.4e1 / 0.7e1 * t163
```

```
t180 = t48 * p
t184 = t51 * t128 * t159 * t125
t186 = t52 * p
t194 = t180 + 0.3e1 / 0.4e1 * t184 + 4 * t186
t203 = -(2 * t156 - 0.3e1 / 0.14e2 * t161 + 0.8e1 / 0.7e1 * t163)
* t82 * t71 / 2 + (t112 * t171 * t29 - t30 * t27 * t171) * t105 *
t71 / 4 + (2 * t180 + 0.3e1 / 0.2e1 * t184 + 8 * t186) * t121 *
t71 / 2 - (t147 * t194 * t57 - t58 * t27 * t194) * t140 * t71 / 4
t217 = Re((-t100 * t82 * t71 / 2 + t117 + t135 * t121 * t71 / 2 -
t152) * q * t77 + t203 * p * t77 + complex(0, 2) * t203 * (t117 -
t100 * t80 * t33 * t71 / 2 - t152 + t135 * t119 * t61 * t71 / 2))
D22 = t69 + t217
```