Pensieve header: A unified verification program for the sl_2-portfolio project, Uxi version. Continues pensieve://Projects/SL2Portfolio/nb/Verification.pdf.

Also continues pensieve://Projects/PPSA/nb/Verification.pdf and pensieve://2017-06/ and pensieve://2017-08/.

DocileQ

DocileQ

In[•]:=
```
DQ[ℰ_] := (Exponent[Normal@ℰ /.
      {a → a / ϵ, a_i_ → a_i / ϵ, (u : x | y) :↦ ϵ^(-1/2) u, (u : x | y)_i_ :↦ ϵ^(-1/2) u_i}, ϵ, Min] ≥ 0);
```

Initialization / Utilities

It is verification-risky to work with low $E!

TD

In[•]:=
```
$p = 2; $k = 1; $U = QU; $E := {$k, $p};
$trim := {ħ^p_· /; p > $p → 0, ϵ^k_· /; k > $k → 0};
SetAttributes[{SS, SST}, HoldAll];
q_ħ = e^(γ ϵ ħ);
T2t = {T_i_^p_· → e^(p ħ t_i), T^p_· → e^(p ħ t)}; (* "T to lower t" *)
t2T = {e^(c_· t_i + b_·) :↦ T_i^(c/ħ) e^b, e^(c_· t + b_·) :↦ T^(c/ħ) e^b, e^ℰ_ :↦ e^Expand@ℰ}; (* "t to upper T" *)
SS[ℰ_, op_] := Collect[
   Normal@Series[If[$p > 0, ℰ, ℰ /. T2t], {ħ, 0, $p}],
   ħ, op];
SS[ℰ_] := SS[ℰ, Together];
SST[ℰ_, op___] := SS[ℰ /. T2t, op];
Simp[ℰ_, op_] := Collect[ℰ, _CU | _QU, op];
Simp[ℰ_] := Simp[ℰ, SS[#, Expand] &];
SimpT[ℰ_] := Collect[ℰ, _CU | _QU, SST[#, Expand] &];
Kδ /: Kδ_i_,j_ := If[i === j, 1, 0];
```

Differential polynomials (DP):

Utils

In[•]:=
```
DP_(α_→D_x_,β_→D_y_)[P_][λ_] :=
 Total[CoefficientRules[Normal@P, {α, β}] /. ({m_, n_} → c_) :↦ c ∂_{x,m},{y,n} λ]
```

CF

In[•]:=
```
CF[ℰ_] := ExpandDenominator@
   ExpandNumerator@Together[Expand[ℰ] //. e^x_ e^y_ :↦ e^(x+y) /. e^x_ :↦ e^CF[x]];
```

SeriesData

In[•]:=
```
Unprotect[SeriesData];
SeriesData /: CF[sd_SeriesData] := MapAt[CF, sd, 3];
SeriesData /: Expand[sd_SeriesData] := MapAt[Expand, sd, 3];
SeriesData /: Simplify[sd_SeriesData] := MapAt[Simplify, sd, 3];
SeriesData /: Together[sd_SeriesData] := MapAt[Together, sd, 3];
SeriesData /: Collect[sd_SeriesData, specs__] := MapAt[Collect[#, specs] &, sd, 3];
Protect[SeriesData];
```

Self-Pair (SP):

SP

In[•]:=
```
SP{}[P_] := P; SP{ξ_→x_,ps___}[P_] := Expand[P // SP{ps}] /. f_. ξ^d_. :↦ ∂{x,d}f
```

DeclareAlgebra

QLImplementation

In[•]:=
```
Unprotect[NonCommutativeMultiply]; Attributes[NonCommutativeMultiply] = {};
(NCM = NonCommutativeMultiply)[x_] := x;
NCM[x_, y_, z__] := (x ** y) ** z;
0 ** _ = _ ** 0 = 0;
(x_Plus) ** y_ := (# ** y) & /@ x; x_ ** (y_Plus) := (x ** #) & /@ y;
B[x_, x_] = 0; B[x_, y_] := x ** y - y ** x;
B[x_, y_, e_] := B[x, y, e] = B[x, y];
```

QLImplementation

In[•]:=

```
DeclareAlgebra[U_Symbol, opts__Rule] := Module[{gp, sr, g, cp, M, CE, pow, k = 0,
   gs = Generators /. {opts},
   cs = Centrals /. {opts} /. Centrals → {} },
  (#_U = U@#) & /@ gs;
  gp = Alternatives @@ gs; gp = gp | gp_; (* gens *)
  sr = Flatten@Table[{g → ++k, g_i_ → {i, k}}, {g, gs}]; (* sorting → *)
  cp = Alternatives @@ cs; (* cents *)
  SetAttributes[M, HoldRest]; M[0, _] = 0; M[a_, x_] := a x;
  CE[ℰ_] := Collect[ℰ, _U, Expand] /. $trim;
  U_i_[ℰ_] := ℰ /. {t : cp :↦ t_i, u_U :↦ (#_i &) /@ u};
  U_i_[NCM[]] = pow[ℰ_, 0] = U@{} = 1_U = U[];
  B[U@(x_)_i_, U@(y_)_i_] := U_i@B[U@x, U@y];
  B[U@(x_)_i_, U@(y_)_j_] /; i =!= j := 0;
  B[U@y_, U@x_] := CE[-B[U@x, U@y]];
  x_ ** (c_. 1_U) := CE[c x]; (c_. 1_U) ** x_ := CE[c x];
  (a_. U[xx___, x_]) ** (b_. U[y_, yy___]) := If[OrderedQ[{x, y} /. sr],
    CE@M[a b /. $trim, U[xx, x, y, yy]],
    U@xx ** CE@M[a b /. $trim, U@y ** U@x + B[U@x, U@y, $E]] ** U@yy ];
  U@{c_. * (l : gp)^n_, r___} /; FreeQ[c, gp] := CE[c U@Table[l, {n}] ** U@{r}];
  U@{c_. * l : gp, r___} := CE[c U[l] ** U@{r}];
  U@{c_, r___} /; FreeQ[c, gp] := CE[c U@{r}];
  U@{l_Plus, r___} := CE[U@{#, r} & /@ l];
  U@{l_, r___} := U@{Expand[l], r};
  U[ℰ_NonCommutativeMultiply] := U /@ ℰ;
  𝒪_U[specs___, poly_] := Module[{sp, null, vs, us},
    sp = Replace[{specs}, l_List :↦ l_null, {1}];
    vs = Join @@ (First /@ sp);
    us = Join @@ (sp /. l_s_ :↦ (l /. x_i_ :↦ x_s));
    CE[Total[
       CoefficientRules[poly, vs] /. (p_ → c_) :↦ c U@(us^p)
      ]] /. x_null :↦ x];
  𝒪_U[specs___, 𝔼[L_, Q_, P_]] := 𝒪_U[specs, SS@Normal[P ℮^(L+Q)]];
  pow[ℰ_, n_] := pow[ℰ, n - 1] ** ℰ;
  𝒮_U[ℰ_, ss___Rule] := CE@Total[
     CoefficientRules[ℰ, First /@ {ss}] /.
      (p_ → c_) :↦ c NCM @@ MapThread[pow, {Last /@ {ss}, p}]];
  σ_rs___[c_. * u_U] := (c /. (t : cp)_j_ :↦ t_j/.{rs}) U[List @@ (u /. v_j_ :↦ v_j/.{rs})];
  m_j_→k_[c_. * u_U] := CE[((c /. (t : cp)_j → t_k) DeleteCases[u, _j|k]) **
     U@@Cases[u, w_j :↦ w_k] ** U@@Cases[u, _k]];
  U /: c_. * u_U * v_U := CE[c u ** v];
  S_i_[c_. * u_U] := CE[((c /. S_i[U, Centrals]) DeleteCases[u, _i]) **
     U_i[NCM@@Reverse@Cases[u, x_i :↦ S@U@x]]];
  Δ_i_→j_,k_[c_. * u_U] := CE[((c /. Δ_i→j,k[U, Centrals]) DeleteCases[u, _i]) **
     (NCM@@Cases[u, x_i :↦ σ_1→j,2→k@Δ@U@x] /. NCM[] → U[])]; ]
```

DeclareMorphism

QLImplementation

```
In[•]:= DeclareMorphism[m_, U_ → V_, ongs_List, oncs_List: {}] := (
  Replace[ongs,
   {(g_ → img_) :> (m[U[g]] = img), (g_ :> img_) :> (m[U[g]] := img /. $trim)}, {1}];
  m[1_U] = 1_V;
  m[U[g_i_]] := V_i[m[U@g]];
  m[U[vs__]] := NCM @@ (m /@ U /@ {vs});
  m[ℰ_] := Simp[ℰ /. oncs /. u_U :> m[u]] /. $trim; )
```

Meta-Operations

QLImplementation

```
In[•]:= σ_rs___[ℰ_Plus] := σ_rs /@ ℰ;
m_j_→j_ = Identity; m_j_→k_[0] = 0;
m_j_→k_[ℰ_Plus] := Simp[m_j→k /@ ℰ];
m_is___,i_,j_→k_[ℰ_] := m_j→k@m_is,i→j@ℰ;
S_i_[ℰ_Plus] := Simp[S_i /@ ℰ];
Δ_is___[ℰ_Plus] := Simp[Δ_is /@ ℰ];
```

Implementing CU = $\mathcal{U}(\mathrm{sl}_2^{\gamma\epsilon})$

Verify σ and Δ! Also Generalize Δ to $\Delta_{i,j_1,j_2,\ldots}$.

CU

```
In[•]:= DeclareAlgebra[CU, Generators → {y, a, x}, Centrals → {t}];
B[a_CU, y_CU] = -γ y_CU; B[x_CU, a_CU] = -γ x_CU;
B[x_CU, y_CU] = 2 ϵ a_CU - t 1_CU;
(S@y_CU = -y_CU; S@a_CU = -a_CU; S@x_CU = -x_CU;)
S_i_[CU, Centrals] = {t_i → -t_i};
Δ@y_CU = CU@y_1 + CU@y_2; Δ@a_CU = CU@a_1 + CU@a_2; Δ@x_CU = CU@x_1 + CU@x_2;
Δ_i_→j_,k_[CU, Centrals] = {t_i → t_j + t_k};
```

Implementing QU = $\mathcal{U}_q(\mathrm{sl}_2^{\gamma\epsilon})$

QU

```
In[•]:= DeclareAlgebra[QU, Generators → {y, a, x}, Centrals → {t, T}];
B[a_QU, y_QU] = -γ y_QU; B[x_QU, a_QU] = -γ QU@x;
B[x_QU, y_QU] := SS[q_ħ - 1] QU@{y, x} + 𝕆_QU[{a}, SS[(1 - T ⅇ^(-2 ϵ a ħ)) / ħ]];
(S@y_QU := 𝕆_QU[{a, y}, SS[-T^-1 ⅇ^(ħ ϵ a) y]]; S@a_QU = -a_QU; S@x_QU := 𝕆_QU[{a, x}, SS[-ⅇ^(ħ ϵ a) x]];)
S_i_[QU, Centrals] = {t_i → -t_i, T_i → T_i^-1};
Δ@y_QU := 𝕆_QU[{y_1, a_1}_1, {y_2}_2, SS[y_1 + T_1 ⅇ^(-ħ ϵ a_1) y_2]];
Δ@a_QU = QU@a_1 + QU@a_2; Δ@x_QU := 𝕆_QU[{a_1, x_1}_1, {x_2}_2, SS[x_1 + ⅇ^(-ħ ϵ a_1) x_2]];
Δ_i_→j_,k_[QU, Centrals] = {t_i → t_j + t_k, T_i → T_j T_k};
```

Implementing θ

theta

In[•]:=
```
DeclareMorphism[Cθ, CU → CU, {y → -x_CU, a → -a_CU, x → -y_CU}, {t → -t, T → T^-1}];
DeclareMorphism[Qθ, QU → QU, {y :↦ 𝕆_QU[{a, x}, SS[-T^(-1/2) ℯ^(ħ ϵ a) x]],
  a → -a_QU, x :↦ 𝕆_QU[{a, y}, SS[-T^(-1/2) ℯ^(ħ ϵ a) y]]}, {t → -t, T → T^-1}]
```

The Asymmetric Dequantizator

Following pensieve://People/VanDerVeen/Dequant1.pdf.

ADeq

In[•]:=
$$\mathbb{AD}\$f = \gamma \left(\left(\text{Cosh}\left[\hbar \left(a\,\epsilon + \frac{\gamma\,\epsilon}{2} - \frac{t}{2}\right)\right] - \text{Cosh}\left[\hbar \sqrt{\left(\frac{t-\gamma\,\epsilon}{2}\right)^2 + \epsilon\,\omega}\,\right] \right) \Big/ \left(\hbar\, e^{\hbar\,((a+\gamma)\,\epsilon - t/2)}\, \text{Sinh}\left[\frac{\gamma\,\epsilon\,\hbar}{2}\right] \left(a^2\,\epsilon + a\,\gamma\,\epsilon - a\,t - \omega\right) \right) \right);$$

ADeq

In[•]:=
$$\mathbb{AD}\$\omega = \gamma\, CU[y, x] + \epsilon\, CU[a, a] - \left(t - \gamma\,\epsilon\right) CU[a];$$

ADeq

In[•]:=
```
DeclareMorphism[AD, QU → CU,
  {a → a_CU, x → CU@x, y :↦ 𝕊_CU[SS[AD$f], a → a_CU, ω → AD$ω] ** y_CU}]
```

The Symmetric Dequantizator

Following pensieve://People/VanDerVeen/Dequant1.pdf.

SDeq

In[•]:=
$$\mathbb{SD}\$g = \sqrt{\left(\left(2\,\gamma \left(\text{Cosh}\left[\frac{\hbar}{2} \sqrt{t^2 + \gamma^2\,\epsilon^2 + 4\,\epsilon\,\varpi}\,\right] - \text{Cosh}\left[\frac{t - \epsilon\,\gamma - 2\,\epsilon\,a}{2/\hbar}\right] \right) \right) \Big/ \left(\text{Sinh}\left[\frac{\gamma\,\epsilon\,\hbar}{2}\right] \left(t\,\left(2\,a + \gamma\right) - 2\,a\,\left(a + \gamma\right)\,\epsilon + 2\,\varpi\right)\,\hbar \right) \right)};$$

SDeq

In[•]:=
```
SD$f = Simplify[ℯ^(ħ (t/2-ϵ a)) (SD$g /. {a → -a, t → -t})];
```

SDeq

In[•]:=
$$\mathbb{SD}\$\varpi = \gamma\, CU[y, x] + \epsilon\, CU[a, a] - \left(t - \gamma\,\epsilon\right) CU[a] - t\,\gamma\, 1_{CU}/2;$$

SDeq

```
In[•]:= DeclareMorphism[𝕊𝔻, QU → CU, {a → a_CU,
   x :> 𝕊_CU[SS[𝕊𝔻$f], a → a_CU, ϖ → 𝕊𝔻$ϖ] ** x_CU,
   y :> 𝕊_CU[SS[𝕊𝔻$g], a → a_CU, ϖ → 𝕊𝔻$ϖ] ** y_CU }]
```

The representation ρ

rho

```
In[•]:= ρ@y_CU = ρ@y_QU = ({{0, 0}, {ϵ, 0}}); ρ@a_CU = ρ@a_QU = ({{γ, 0}, {0, 0}});
ρ@x_CU = ({{0, γ}, {0, 0}}); ρ@x_QU = ({{0, (1 - ℯ^(-γ ϵ ħ)) / (ϵ ħ)}, {0, 0}});
ρ[ℯ^ℰ_] := MatrixExp[ρ[ℰ]];
ρ[ℰ_] := (ℰ /. T2t /. t → γ ϵ /. (U : CU | QU)[u___] :> Fold[Dot, ({{1, 0}, {0, 1}}), ρ /@ U /@ {u}])
```

Œ and the logoi Λ

Logoi from Pensieve://Talks/Toulouse-1705/DogmaDemo.nb and from Pensieve://Talks/Sydney-1708/ExtraDetails@@.nb.

MultiplyingOEs

```
In[•]:= Œ_U_[s1__, Q1_, P1_] Œ_U_[s2__, Q2_, P2_] ^:= Œ_U[s1, s2, Q1 + Q2, P1 P2];
```

CdsO

```
In[•]:= CU@Œ_CU[specs___, Q_, P_] := 𝒪_CU[specs, SS[ℯ^Q P]];
QU@Œ_QU[specs___, Q_, P_] := 𝒪_QU[specs, SS[ℯ^Q P]];
```

Logos

```
In[•]:= c_Integer_k_Integer := c + O[ϵ]^(k+1);
Λ_U_,k_[{α_, β_}, {x_, x_}] := Œ_U[{x}, (α + β) x, 1_k];
Λ_U_,k_[{ξ_, α_}, {x, a}] := Œ_U[{a, x}, α a + ℯ^(-γ α) ξ x, 1_k];
Λ_U_,k_[{α_, η_}, {a, y}] := Œ_U[{y, a}, α a + ℯ^(-γ α) η y, 1_k];
```

Goal. In either U, compute $F = e^{-\eta y}\, e^{\xi x}\, e^{\eta y}\, e^{-\xi x}$. First compute $G = e^{\xi x}\, y\, e^{-\xi x}$, a finite sum. Now F satisfies the ODE $\partial_\eta F = \partial_\eta\left(e^{-\eta y}\, e^{\eta G}\right) = -yF + FG$ with initial conditions $F(\eta = 0) = 1$. So we set it up and solve:

```
If[$k > 0, With[{U = CU},
  Module[{G, F, fs, bs, e, b, es, sol},
   G = Echo@Simp[Table[ξ^k / k!, {k, 0, $k + 1}].NestList[Simp[B[x_U, #]] &, y_U, $k + 1]];
   fs = Echo@Flatten@Table[f_{1,i,j,k}[η], {l, 0, $k}, {i, 0, l}, {j, 0, l}, {k, 0, l}];
   F = Echo[fs.(bs = fs /. f_{_,i_,j_,k_}[η] :> ϵ^l U@{y^i, a^j, x^k})];
   es = Flatten[
     Table[Coefficient[e, b] == 0, {e, {F - 1_U /. η → 0, F ** G - y_U ** F - ∂_η F}}, {b, bs}]];
   sol = Echo@First[F /. DSolve[es, fs, η]];
   Echo[sol /. {ℯ- → 1, U → Times}];
   Collect[sol /. {ℯ- → 1, U → Times}, ϵ, Simplify]
  ]]]
```

″ $-t\,\xi\,CU[\,] + 2\,\epsilon\,\xi\,CU[a] - \gamma\,\epsilon\,\xi^2\,CU[x] + CU[y]$

″ $\{f_{0,0,0,0}[\eta],\ f_{1,0,0,0}[\eta],\ f_{1,0,0,1}[\eta],\ f_{1,0,1,0}[\eta],\ f_{1,0,1,1}[\eta],\ f_{1,1,0,0}[\eta],\ f_{1,1,0,1}[\eta],\ f_{1,1,1,0}[\eta],\ f_{1,1,1,1}[\eta]\}$

″ $CU[\,]\,f_{0,0,0,0}[\eta] + \epsilon\,CU[\,]\,f_{1,0,0,0}[\eta] + \epsilon\,CU[x]\,f_{1,0,0,1}[\eta] + \epsilon\,CU[a]\,f_{1,0,1,0}[\eta] + \epsilon\,CU[a, x]\,f_{1,0,1,1}[\eta] + \epsilon\,CU[y]\,f_{1,1,0,0}[\eta] + \epsilon\,CU[y, x]\,f_{1,1,0,1}[\eta] + \epsilon\,CU[y, a]\,f_{1,1,1,0}[\eta] + \epsilon\,CU[y, a, x]\,f_{1,1,1,1}[\eta]$

» $\mathrm{e}^{-t\,\eta\,\xi}\,CU[\,] + \frac{1}{2}\,\mathrm{e}^{-t\,\eta\,\xi}\,t\,\gamma\,\epsilon\,\eta^2\,\xi^2\,CU[\,] + 2\,\mathrm{e}^{-t\,\eta\,\xi}\,\epsilon\,\eta\,\xi\,CU[a] - \mathrm{e}^{-t\,\eta\,\xi}\,\gamma\,\epsilon\,\eta\,\xi^2\,CU[x] - \mathrm{e}^{-t\,\eta\,\xi}\,\gamma\,\epsilon\,\eta^2\,\xi\,CU[y]$

» $1 + 2\,a\,\epsilon\,\eta\,\xi - y\,\gamma\,\epsilon\,\eta^2\,\xi - x\,\gamma\,\epsilon\,\eta\,\xi^2 + \frac{1}{2}\,t\,\gamma\,\epsilon\,\eta^2\,\xi^2$

$1 + \frac{1}{2}\,\epsilon\,\eta\,\xi\,\left(4\,a + \gamma\,\left(-2\,y\,\eta - 2\,x\,\xi + t\,\eta\,\xi\right)\right)$

Logos

In[ ]:=
```
Λ_{U_,kk_}[{ξ1_, η1_}, {x, y}] := Λ_{U,kk}[{ξ1, η1}, {x, y}] =
  Block[{$k = kk, $p = kk}, Module[{ξ, η, G, F, fs, f, bs, e, b, es},
    G = Simp[Table[ξ^k / k!, {k, 0, $k + 1}].NestList[Simp[B[x_U, #]] &, y_U, $k + 1]];
    fs = Flatten@Table[f_{1,i,j,k}[η], {l, 0, $k}, {i, 0, l}, {j, 0, l}, {k, 0, l}];
    F = fs.(bs = fs /. f_{_,i_,j_,k_}[η] :> ϵ^l U@{y^i, a^j, x^k});
    es = Flatten[
      Table[Coefficient[e, b] == 0, {e, {F - 1_U /. η → 0, F ** G - y_U ** F - ∂_η F}}, {b, bs}]];
    F = F /. DSolve[es, fs, η][[1]];
    𝔈_U[{y, a, x},
      ξ x + η y + (U /. {CU → -t η ξ, QU → η ξ (1 - T) / ħ}),
      F + 0_$k /. {ℯ- → 1, U → Times}
    ] /. {ξ → ξ1, η → η1}]];
```

Logos

In[ ]:=
```
Simp[𝔈_U[specs___, Q_, P_]] := 𝔈_U[specs, CF[Q], CF[P]];
```

Logos

In[ ]:=
```
Λ_{U_,k_}[{υ1_, ω1_, δ_}, {u_, w_}] := Simp@Module[{υ, ω, yax, q, p, Q, d},
    {yax, q, p} = List @@ Λ_{U,k}[{υ, ω}, {u, w}];
    𝔈_U[yax, Q = (υ u + ω w + δ u w + d υ ω) / (1 - d δ),
      Expand[(1 - d δ)^-1 ℯ^-Q DP_{υ→D_u, ω→D_w}[p][ℯ^Q]] + 0_k] /. {d → ∂_{υ,ω} q} /. {υ → υ1, ω → ω1}];
```

tSW

tSW

In[•]:=

```
SWxy[U_, kk_] :=
  SWxy[U, kk] = Block[{$U = U, $k = kk, $p = kk}, Module[{G, F, fs, f, bs, e, b, es},
    G = Simp[Table[ξ^k / k!, {k, 0, $k + 1}].NestList[Simp[B[x_U, #]] &, y_U, $k + 1]];
    fs = Flatten@Table[f_{l,i,j,k}[η], {l, 0, $k}, {i, 0, l}, {j, 0, l}, {k, 0, l}];
    F = fs.(bs = fs /. f_{l_,i_,j_,k_}[η] :> ϵ^l U@{y^i, a^j, x^k});
    es = Flatten[
      Table[Coefficient[e, b] == 0, {e, {F - 1_U /. η → 0, F ** G - y_U ** F - ∂_η F}}, {b, bs}]];
    F = F /. DSolve[es, fs, η][[1]];
    𝔼[0,
      ξ x + η y + (U /. {CU → -t η ξ, QU → η ξ (1 - T) / ħ}),
      F + 0_$k /. {e^- → 1, U → Times}
     ] /. (v : η | ξ | t | T | y | a | x) → v_1
    ]];
tSW_{xy_,i_,j_→k_} := SWxy[$U, $k] /. {ξ_1 → ξ_i, η_1 → η_j, (v : t | T | y | a | x)_1 → v_k};
tSW_{xa,i_,j_→k_} := 𝔼[α_j a_k, e^{-γ α_j} ξ_i x_k, 1];
tSW_{ay,i_,j_→k_} := 𝔼[α_i a_k, e^{-γ α_i} η_j y_k, 1];
```

Reorderings with Rord

Rord

In[•]:=

```
Rord_{u_i_,w_j_→k_}[𝔄_U[L___, {l___, u_i_, w_j_, r___}_s_, R___, Q_, P_]] :=
  Simp@Module[{υ, ω, δ, Λ1, yax, q, p, kk = P[[5]], δ1 = ∂_{u_i,w_j} Q},
    {yax, q, p} = Echo[List @@ If[δ1 === 0, Λ_{U,kk}[{υ, ω}, {u, w}],
        Λ_{U,kk}[{υ, ω, δ}, {u, w}]] /. {y → y_k, a → a_k, x → x_k, t → t_s, T → T_s}];
    𝔄_U[L, {l, Sequence @@ yax, r}_s, R, q + (Q /. u_i | w_j → 0), e^{-q} DP_{u_i→D_υ,w_j→D_ω}[P][p e^q]] /.
     {υ → ∂_{u_i} Q /. w_j → 0, ω → ∂_{w_j} Q /. u_i → 0, δ → δ1}];
```

Rord

In[•]:=

```
Rord_{u_i_,w_j_→k_}[𝔄_U[L___, {l___, u_i_, w_j_, r___}_s_, R___, Q_, P_]] :=
  Simp@Module[{υ, ω, δ, Λ1, yax, q, p, n, kk = P[[5]], δ1 = ∂_{u_i,w_j} Q},
    {yax, q, p} = List @@ If[δ1 === 0, Λ_{U,kk}[{υ, ω}, {u, w}], Λ_{U,kk}[{υ, ω, δ}, {u, w}]] /.
      {y → y_n, a → a_n, x → x_n, t → t_s, T → T_s};
    (*Echo@{{{u_i,υ},{w_j,ω}},P,p e^q};*)
    𝔄_U[L, {l, Sequence @@ yax, r}_s, R, q + (Q /. u_i | w_j → 0), e^{-q} SP_{u_i→υ,w_j→ω}[P p e^q]] /.
     {n → k, υ → ∂_{u_i} Q /. w_j → 0, ω → ∂_{w_j} Q /. u_i → 0, δ → δ1}];
```

Canonical ordering with Cord

Cord

In[•]:=
```
Cord[Œ_U_[L___, {l___, u_i_, w_j_, r___}_s_, R___, Q_, P_]] /;
   OrderedQ[{w, u} /. {y → 1, a → 2, x → 3}] :=
  (*Echo@{u_i,w_j};*) Cord[Rord_{u_i,w_j→Unique[]}[Œ_U[L, {l, u_i, w_j, r}_s, R, Q, P]]];
Cord[Œ_U_[specs__, Q_, P_]] := Œ_U[Sequence @@ Sort@{specs}, Q, P] /.
  Flatten[{specs} /. {yax__}_s_ :> ({yax} /. u_i_ :> (u_i → u_s))]
```

Stitching Œ's.

StitchingOEs

In[•]:=
```
m_j_→k_[Œ_U_[specs__, Q_, P_]] := Cord[Œ_U[Sequence @@ Append[DeleteCases[{specs}, {__}_j|k],
      Flatten[{Cases[{specs}, {us__}_j :> {us}], Cases[{specs}, {us__}_k :> {us}]}]_k],
    Q, P] /. {t_j → t_k, T_j → T_k}]
```

In[•]:=
```
Œ_U_[sp1__, Q1_, P1_] ≡ Œ_U_[sp2__, Q2_, P2_] :=
 Sort[{sp1}] == Sort[{sp2}] ∧ Simplify[Q1 == Q2] ∧ Simplify[Normal[P1 - P2] == 0]
```

R in QU.

The Faddeev-Quesne formula:

Faddeev

In[•]:=
```
e_q_,k_[x_] := e^(Sum[(1 - q)^j x^j / (j (1 - q^j)), {j, 1, k + 1}]); e_q_[x_] := e_q,$k[x]
```

R

In[•]:=
```
QU[R_i_,j_] := O_QU[{y_1, a_1}_i, {a_2, x_2}_j, SS[e^(ħ b_1 a_2) e_qħ[ħ y_1 x_2] /. b_1 → γ^-1 (ϵ a_1 - t_i)]];
QU[R_i_^-1,j_] := S_j@QU[R_i,j];
```

R in $Œ_{QU}$.

RinOE

In[•]:=
```
Œ_QU,k_[R_i_,j_] := Œ_QU[{y_i, a_i, x_i}_i, {y_j, a_j, x_j}_j, -ħ γ^-1 t_i a_j + ħ y_i x_j,
  Series[e^(ħ γ^-1 t_i a_j - ħ y_i x_j) (e^(ħ b_i a_j) e_qħ,k[ħ y_i x_j] /. b_i → γ^-1 (ϵ a_i - t_i)), {ϵ, 0, k}]]
```

The morphism $Œ_{U,k}$.

MorphismOE

In[•]:=
```
Œ_U_,k_[a_ * b_] := Œ_U,k[a] Œ_U,k[b];
Œ_U_,k_[m_is__[a_]] := m_is[Œ_U,k[a]];
```

Exponentials as needed.

Exp

Task. Define $\mathrm{Exp}_{U_i,k}[\xi, P]$ which computes $e^{\xi\,\mathfrak{O}(P)}$ to ϵ^k in the algebra U_i, where ξ is a scalar, X is x_i or y_i,

and P is an ϵ-dependent near-docile element, giving the answer in Œ-form. Should satisfy $U@\text{Exp}_{U_i,k}[\xi, P] == \mathbb{S}_U\left[e^{\xi x},\ x \to \mathbb{O}(P)\right]$.

Methodology. If $P_0 := P_{\epsilon=0}$ and $e^{\xi\,\mathbb{O}(P)} = \mathbb{O}\left(e^{\xi P_0} F(\xi)\right)$, then $F(\xi=0)=1$ and we have:
$\mathbb{O}\left(e^{\xi P_0}(P_0 F(\xi) + \partial_\xi F\right) = \mathbb{O}\left(\partial_\xi e^{\xi P_0} F(\xi)\right) = \partial_\xi \mathbb{O}\left(e^{\xi P_0} F(\xi)\right) = \partial_\xi e^{\xi\,\mathbb{O}(P)} = e^{\xi\,\mathbb{O}(P)}\,\mathbb{O}(P) = \mathbb{O}\left(e^{\xi P_0} F(\xi)\right)\mathbb{O}(P)$.
This is an ODE for F. Setting inductively $F_k = F_{k-1} + \epsilon^k \varphi$ we find that $F_0 = 1$ and solve for φ.

Exp

```
(* Bug: The first line is valid only if 𝕆(e^P0) == e^𝕆(P0). *)
(* Bug: ξ must be a symbol. *)
Exp_{U_i_,0}[ξ_, P_] := Module[{LQ = Normal@P /. ϵ → 0},
   𝔼[ξ LQ /. (x | y)_i → 0, ξ LQ /. (t | a)_i → 0, 1]];
Exp_{U_i_,k_}[ξ_, P_] := Block[{$U = U, $k = k},
  Module[{P0, φ, φs, F, j, rhs, at0, atξ},
   P0 = Normal@P /. ϵ → 0;
   φs =
    Flatten@Table[φ_{j1,j2,j3}[ξ], {j2, 0, k}, {j1, 0, 2 k + 1 - j2}, {j3, 0, 2 k + 1 - j2 - j1}];
   F = Normal@Last@Exp_{U_i,k-1}[ξ, P] + ϵ^k φs. (φs /. φ_{js__}[ξ] :> Times @@ {y_i, a_i, x_i}^{js});
   rhs = Normal@
     Last@m_{i,j→i}[𝔼[ξ P0 /. (x | y)_i → 0, ξ P0 /. (t | a)_i → 0, F + 0_k] m_{i→j}@𝔼[0, 0, P + 0_k]];
   at0 = (# == 0) & /@ Flatten@CoefficientList[F - 1 /. ξ → 0, {y_i, a_i, x_i}];
   atξ = (# == 0) & /@ Flatten@CoefficientList[(∂_ξ F) + P0 F - rhs, {y_i, a_i, x_i}];
   𝔼[ξ P0 /. (x | y)_i → 0, ξ P0 /. (t | a)_i → 0, F + 0_k] /.
    DSolve[And @@ (at0 ⋃ atξ), φs, ξ][[1]]]]
```

Zip and Bind

E

In[ ]:=
```
𝔼 /: 𝔼[L1_, Q1_, P1_] ≡ 𝔼[L2_, Q2_, P2_] :=
  CF[L1 == L2] ∧ CF[Q1 == Q2] ∧ CF[Normal[P1 - P2] == 0];
𝔼 /: 𝔼[L1_, Q1_, P1_] 𝔼[L2_, Q2_, P2_] := 𝔼[L1 + L2, Q1 + Q2, P1 * P2];
```

Zip

In[ ]:=
```
{t*, y*, a*, x*, z*} = {τ, η, α, ξ, ζ};
{τ*, η*, α*, ξ*, ζ*} = {t, y, a, x, z}; (u_i_)* := (u*)_i;
```

Zip

In[ ]:=
```
Zip_{}[P_] := P; Zip_{ξ_, ξs___}[P_] := (Expand[P // Zip_{ξs}] /. f_. ξ^d_. :> ∂_{ξ*,d} f) /. ξ* → 0
```

QZip implements the "Q-level zips" on $\mathbb{E}(L, Q, P) = \text{P}e^{L+Q}$. Such zips regard the L variables as scalars.

Zip

```
In[•]:= 𝔼 /: QZip_{ξs_List}@𝔼[L_, Q_, P_] := Module[{ξ, z, zs, c, ys, ηs, qt, zrule, Q1, Q2},
    zs = Table[ξ*, {ξ, ξs}];
    c = Q /. Alternatives @@ (ξs ⋃ zs) → 0;
    ys = Table[∂_ξ (Q /. Alternatives @@ zs → 0), {ξ, ξs}];
    ηs = Table[∂_z (Q /. Alternatives @@ ξs → 0), {z, zs}];
    qt = Inverse@Table[Kδ_{z,ξ*} - ∂_{z,ξ} Q, {ξ, ξs}, {z, zs}];
    zrule = Thread[zs → qt.(zs + ys)];
    Q2 = (Q1 = c + ηs.zs /. zrule) /. Alternatives @@ zs → 0;
    CF /@ 𝔼[L, Q2, Det[qt] ℮^-Q2 Zip_ξs[℮^Q1 (P /. zrule)]] ];
```

LZip implements the "L-level zips" on $\mathbb{E}(L, Q, P) = \mathrm{P}e^{L+Q}$. Such zips regard all of $\mathrm{P}e^{Q}$ as a single"P". Here the z's are t and α and the ζ's are τ and a.

Zip

```
In[•]:= 𝔼 /: LZip_{ξs_List}@𝔼[L_, Q_, P_] := Module[{ξ, z, zs, c, ys, ηs, lt, zrule, L1, L2, Q1, Q2},
    zs = Table[ξ*, {ξ, ξs}];
    c = L /. Alternatives @@ (ξs ⋃ zs) → 0;
    ys = Table[∂_ξ (L /. Alternatives @@ zs → 0), {ξ, ξs}];
    ηs = Table[∂_z (L /. Alternatives @@ ξs → 0), {z, zs}];
    lt = Inverse@Table[Kδ_{z,ξ*} - ∂_{z,ξ} L, {ξ, ξs}, {z, zs}];
    zrule = Thread[zs → lt.(zs + ys)];
    L2 = (L1 = c + ηs.zs /. zrule) /. Alternatives @@ zs → 0;
    Q2 = (Q1 = Q /. T2t /. zrule) /. Alternatives @@ zs → 0;
    CF /@ 𝔼[L2, Q2, Det[lt] ℮^-L2-Q2 Zip_ξs[℮^L1+Q1 (P /. T2t /. zrule)]] //. t2T ];
```

Bind

```
In[•]:= Bind_{}[L_, R_] := L R;
Bind_{is__}[L_𝔼, R_𝔼] := Module[{n},
    Times[
       L /. Table[(v : T | t | a | x | y)_i → v_{n@i}, {i, {is}}],
       R /. Table[(v : τ | α | ξ | η)_i → v_{n@i}, {i, {is}}]
      ] // LZip_{Flatten@Table[{τ_{n@i}, a_{n@i}}, {i, {is}}]} // QZip_{Flatten@Table[{ξ_{n@i}, y_{n@i}}, {i, {is}}]} ];
B_{l_List} := Bind_l; B_{is___} := Bind_{is};
Bind[ℰ_𝔼] := ℰ;
Bind[Ls__, ξs_List, R_] := Bind_ξs[Bind[Ls], R];
```

Tensorial Representations

t1

```
In[•]:= tη = t1 = 𝔼[0, 0, 1 + 0_$k];
```

tm

```
In[•]:= tm_{i_,j_→k_} := Module[{tk},
    𝔼[(τ_i + τ_j) t_k + α_i a_k + α_j a_k, η_i y_k + ξ_j x_k, 1]
     (tSW_{xy,i,j→tk} /. {t_tk → t_k, T_tk → T_k, y_tk → ℮^-γ α_i y_k, a_tk → a_k, x_tk → ℮^-γ α_j x_k}) ];
m_{j_→k_}[ℰ_𝔼] := ℰ ~B_{j,k}~ tm_{j,k→k};
```

In[•]:=
```
tm_{1,2→3}
```

Out[•]=
$$\mathbb{E}\Big[a_3\,\alpha_1+a_3\,\alpha_2+t_3\,(\tau_1+\tau_2),\ y_3\,\eta_1+e^{-\gamma\,\alpha_1}\,y_3\,\eta_2+e^{-\gamma\,\alpha_2}\,x_3\,\xi_1+\frac{\left(1-T_3\right)\,\eta_2\,\xi_1}{\hbar}+x_3\,\xi_2,$$
$$1+\frac{1}{4\,\hbar}\,\eta_2\,\xi_1\,\left(8\,\hbar\,a_3\,T_3+4\,e^{-\gamma\,\alpha_1-\gamma\,\alpha_2}\,\gamma\,\hbar^2\,x_3\,y_3+2\,e^{-\gamma\,\alpha_1}\,\gamma\,\hbar\,y_3\,\eta_2-6\,e^{-\gamma\,\alpha_1}\,\gamma\,\hbar\,T_3\,y_3\,\eta_2+\right.$$
$$\left.2\,e^{-\gamma\,\alpha_2}\,\gamma\,\hbar\,x_3\,\xi_1-6\,e^{-\gamma\,\alpha_2}\,\gamma\,\hbar\,T_3\,x_3\,\xi_1+\gamma\,\eta_2\,\xi_1-4\,\gamma\,T_3\,\eta_2\,\xi_1+3\,\gamma\,T_3^2\,\eta_2\,\xi_1\right)\,\epsilon+0[\epsilon]^2\Big]$$

tS

In[•]:=
```
S[U_, kk_] := S[U, kk] = Module[{OE},
    OE = m_{3,2,1→1}[Exp_{QU_1,$k}[η, S_1[QU[y_1]]] /. QU → Times]
       Exp_{QU_2,$k}[α, S_2[QU[a_2]]] /. QU → Times] Exp_{QU_3,$k}[ξ, S_3[QU[x_3]]] /. QU → Times]];
    𝔼[-t_1 τ_1 + OE⟦1⟧, OE⟦2⟧, OE⟦3⟧] /. {η → η_1, α → α_1, ξ → ξ_1}];
tS_i_ := S[$U, $k] /. {(v : τ | η | α | ξ)_1 → v_i, (v : t | T | y | a | x)_1 → v_i};
```

In[•]:=
```
tS_1
```

Out[•]=
$$\mathbb{E}\Big[-a_1\,\alpha_1-t_1\,\tau_1,\ \frac{1}{\hbar\,T_1}\left(-e^{\gamma\,\alpha_1}\,\hbar\,y_1\,\eta_1-e^{\gamma\,\alpha_1}\,\hbar\,T_1\,x_1\,\xi_1+e^{\gamma\,\alpha_1}\,\eta_1\,\xi_1-e^{\gamma\,\alpha_1}\,T_1\,\eta_1\,\xi_1\right),$$
$$1+\frac{1}{4\,\hbar\,T_1^2}\,\left(4\,e^{\gamma\,\alpha_1}\,\gamma\,\hbar^2\,T_1\,y_1\,\eta_1-4\,e^{\gamma\,\alpha_1}\,\hbar^2\,a_1\,T_1\,y_1\,\eta_1-2\,e^{2\,\gamma\,\alpha_1}\,\gamma\,\hbar^2\,y_1^2\,\eta_1^2-4\,e^{\gamma\,\alpha_1}\,\hbar^2\,a_1\,T_1^2\,x_1\,\xi_1-\right.$$
$$4\,e^{\gamma\,\alpha_1}\,\gamma\,\hbar\,T_1\,\eta_1\,\xi_1+8\,e^{\gamma\,\alpha_1}\,\hbar\,a_1\,T_1\,\eta_1\,\xi_1+4\,e^{\gamma\,\alpha_1}\,\gamma\,\hbar\,T_1^2\,\eta_1\,\xi_1-4\,e^{2\,\gamma\,\alpha_1}\,\gamma\,\hbar^2\,T_1\,x_1\,y_1\,\eta_1\,\xi_1+$$
$$6\,e^{2\,\gamma\,\alpha_1}\,\gamma\,\hbar\,y_1\,\eta_1^2\,\xi_1-2\,e^{2\,\gamma\,\alpha_1}\,\gamma\,\hbar\,T_1\,y_1\,\eta_1^2\,\xi_1-2\,e^{2\,\gamma\,\alpha_1}\,\gamma\,\hbar^2\,T_1^2\,x_1^2\,\xi_1^2+6\,e^{2\,\gamma\,\alpha_1}\,\gamma\,\hbar\,T_1\,x_1\,\eta_1\,\xi_1^2-$$
$$\left.2\,e^{2\,\gamma\,\alpha_1}\,\gamma\,\hbar\,T_1^2\,x_1\,\eta_1\,\xi_1^2-3\,e^{2\,\gamma\,\alpha_1}\,\gamma\,\eta_1^2\,\xi_1^2+4\,e^{2\,\gamma\,\alpha_1}\,\gamma\,T_1\,\eta_1^2\,\xi_1^2-e^{2\,\gamma\,\alpha_1}\,\gamma\,T_1^2\,\eta_1^2\,\xi_1^2\right)\,\epsilon+0[\epsilon]^2\Big]$$

tDelta

In[•]:=
```
Δ[U_, kk_] := Δ[U, kk] = Module[{OE},
    OE = Block[{$k = kk, $p = kk + 1},
      m_{1,3,5→1}@m_{2,4,6→2}@Times[ (* Warning: wrong unless $p≥$k+1! *)
           ReplacePart[1 → 0]@Exp_{QU_1,$k}[η, Δ_{1→1,2}[QU[y_1]] /. QU → Times],
           ReplacePart[2 → 0]@Exp_{QU_3,$k}[α, Δ_{3→3,4}[QU[a_3]] /. QU → Times],
           ReplacePart[1 → 0]@Exp_{QU_5,$k}[ξ, Δ_{5→5,6}[QU[x_5]] /. QU → Times]
         ] /. {η → η_1, α → α_1, ξ → ξ_1}];
    𝔼[τ_1 (t_1 + t_2) + α_1 (a_1 + a_2), OE⟦2⟧, OE⟦3⟧]];
tΔ_{i_→j_,k_} :=
  Δ[$U, $k] /. {(v : τ | η | α | ξ)_1 → v_i, (v : t | T | y | a | x)_1 → v_j, (v : t | T | y | a | x)_2 → v_k};
```

In[•]:=
```
tΔ_{1→1,2}
```

Out[•]=
$$\mathbb{E}\Big[(a_1+a_2)\,\alpha_1+\left(t_1+t_2\right)\,\tau_1,\ y_1\,\eta_1+T_1\,y_2\,\eta_1+x_1\,\xi_1+x_2\,\xi_1,$$
$$1+\frac{1}{2}\,\left(-2\,\hbar\,a_1\,T_1\,y_2\,\eta_1+\gamma\,\hbar\,T_1\,y_1\,y_2\,\eta_1^2-2\,\hbar\,a_1\,x_2\,\xi_1+\gamma\,\hbar\,x_1\,x_2\,\xi_1^2\right)\,\epsilon+0[\epsilon]^2\Big]$$

tR

```
In[•]:= R[QU, kk_] := R[QU, kk] = Module[{OE},
          OE = Simplify /@ 𝔼QU,kk@R1,2;
          𝔼[-ħ a2 t1/γ, ħ x2 y1, Last@OE]];
        tRi_,j_ := R[$U, $k] /. {(v : t | T | y | a | x)1 → vi, (v : t | T | y | a | x)2 → vj};
        tR̄i_,j_ := tR̄i,j = tRi,j ~ Bj ~ tSj;
```

```
In[•]:= {tR1,2, tR̄1,2}
```

```
Out[•]= {𝔼[-ħ a2 t1/γ, ħ x2 y1, 1 + (ħ a1 a2/γ - 1/4 γ ħ^3 x2^2 y1^2) ϵ + O[ϵ]^2], 𝔼[ħ a2 t1/γ, -ħ x2 y1/T1,
           1 - 1/(4 (γ T1^2)) (ħ (4 a1 T1 (a2 T1 + γ ħ x2 y1) + γ ħ x2 y1 (4 a2 T1 + 3 γ ħ x2 y1))) ϵ + O[ϵ]^2]}
```

tC is the counterclockwise spinner; $\overline{\text{tC}}$ is its inverse.

tC

```
In[•]:= tCi_ := 𝔼[0, 0, Ti^(1/2) ℯ^(-ϵ ai ħ) + 0$k];
        tC̄i_ := 𝔼[0, 0, Ti^(-1/2) ℯ^(ϵ ai ħ) + 0$k];
```

```
In[•]:= Block[{$k = 3}, {tC1, tC̄2}]
```

```
Out[•]= {𝔼[-t1/2, 0, 1 + a1 ϵ + 1/2 a1^2 ϵ^2 + 1/6 a1^3 ϵ^3 + O[ϵ]^4], 𝔼[t2/2, 0, 1 - a2 ϵ + 1/2 a2^2 ϵ^2 - 1/6 a2^3 ϵ^3 + O[ϵ]^4]}
```

tKink

```
In[•]:= Kink[QU, kk_] := Kink[QU, kk] = Block[{$k = kk}, (tR1,3 tC̄2) ~ B1,2 ~ tm1,2→1 ~ B1,3 ~ tm1,3→1];
        tKinki_ := Kink[$U, $k] /. {(v : t | T | y | a | x)1 → vi};
        K̄ink[QU, kk_] := K̄ink[QU, kk] = Block[{$k = kk}, (tR̄1,3 tC2) ~ B1,2 ~ tm1,2→1 ~ B1,3 ~ tm1,3→1];
        tK̄inki_ := K̄ink[$U, $k] /. {(v : t | T | y | a | x)1 → vi}
```

Alternative Algorithms

AltLogos

```
In[•]:= λalt,k_[CU] := If[k == 0, 1, Module[{eq, d, b, c, so},
          eq = ρ@ℯ^(ξ xcu) . ρ@ℯ^(η ycu) == ρ@ℯ^(d ycu) . ρ@ℯ^(c (t 1cu - 2 ϵ acu)) . ρ@ℯ^(b xcu);
          {so} = Solve[Thread[Flatten /@ eq], {d, b, c}] /. C@1 → 0;
          Series[ℯ^(-η y-ξ x+η ξ t+c t+d y-2 ϵ c a+b x) /. so, {ϵ, 0, k}]]];
```

Asides

Aside

```
In[•]:= Series[(1 - T ℯ^(-2 ϵ a ħ)) / ħ, {a, 0, 3}]
```

Aside

```
Out[•]= (1 - T)/ħ + 2 T ϵ a - 2 (T ϵ^2 ħ) a^2 + 4/3 T ϵ^3 ħ^2 a^3 + O[a]^4
```