Can be removed in minimal version:

Can be removed if even CU is out:

It would be great to have a better Bind notation!

$$tR \sim B_{1,2,3} \sim tm$$

optimistic plan:

Wed: Complete the program.

Thu: Crop & rewrite.

Fri: Make handout.

Sat: Proofread & optimize.

Sun:: print & go.

Super optimistic plan: Also compute ρ_2 ....

$\mathcal{U}_{\gamma\epsilon;\hbar}$ conventions. "consolidate"

$q=e^{\hbar\gamma\epsilon}$, $H=\langle a,x\rangle/([a,x]=\gamma x)$ with

$$A=e^{-\hbar\epsilon a},\quad xA=qAx,\quad S_H(a,A,x)=(-a,A^{-1},-A^{-1}x),$$
$$\Delta_H(a,A,x)=(a_1+a_2,A_1A_2,x_1+A_1x_2)$$

and dual $H^*=\langle b,y\rangle/([b,y]=-\epsilon y)$ with

$$B=e^{-\hbar\gamma b},\quad By=qyB,\quad S_{H^*}(b,B,y)=(-b,B^{-1},-yB^{-1}),$$
$$\Delta_{H^*}(b,B,y)=(b_1+b_2,B_1B_2,y_1B_2+y_2).$$

Pairing by $(a,x)^*=\hbar(b,y)$ ($\Rightarrow\langle B,A\rangle=q$) making $\langle y^lb^i,a^jx^k\rangle=\delta_{ij}\delta_{kl}\hbar^{-(j+k)}j![k]_q!$ so $R=\sum\frac{\hbar^{j+k}y^kb^j\otimes a^jx^k}{j![k]_q!}$. Then $\mathcal{U}=H^{*cop}\otimes H$ with $(\phi f)(\psi g)=\langle\psi_1S^{-1}f_3\rangle\langle\psi_3,f_1\rangle(\phi\psi_2)(f_2g)$ and

$$S(y,b,a,x)=(-B^{-1}y,-b,-a,-A^{-1}x),$$
$$\Delta(y,b,a,x)=(y_1+y_2B_1,b_1+b_2,a_1+a_2,x_1+A_1x_2).$$

With the central $t:=\epsilon a-\gamma b$, $T:=e^{\hbar t}=A^{-1}B$ get

$$[a,y]=-\gamma y,\quad [b,x]=\epsilon x,\quad xy-qyx=(1-TA^2)/\hbar.$$

Cartan: $\theta(y,b,a,x)=(-B^{-1}T^{1/2}x,-b,-a,-A^{-1}T^{-1/2}y)$. (Suggesting that it may be better to redefine $y\to y'=\theta x=A^{-1}T^{-1/2}y$.)

At $\epsilon=0$, $\mathcal{U}_{\hbar;\gamma 0}=\langle t,y,a,x\rangle/([t,\cdot]=0,[a,x]=\gamma x,[a,y]=-\gamma y,[x,y]=(1-T)/\hbar)$ with $\Delta(t,y,a,x)=(t_1+t_2,y_1+T_1y_2,a_1+a_2,x_1+x_2)$ and $\theta(y,b,a,x)=(-T^{-1/2}x,-b,-a,-T^{-1/2}y)$.

Working Hypothesis. $(\hbar,t,y,a,x)$ makes a PBW basis.

Casimir. $\omega=\gamma yx+\epsilon a^2-(t-\gamma\epsilon)a$, satisfies.... Roland in MixOrder.pdf: Centrals are valuable; perhaps we should write everything in CU/QU as $(x\vee y)\cdot$(functions of a)$\cdot$(centrals).

Scaling with deg: $\{\gamma,\epsilon,a,b,x,y\}\to 1$, $\{\hbar\}\to -2$, $\{t\}\to 2$, $\{\omega\}\to 3$.

Verification (as in Projects/PPSA/Verification.nb).

```
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);
$p = 2; $k = 1; $U = QU; $E := {$k, $p};
$trim := {ħ^p_. /; p > $p → 0, ϵ^k_. /; k > $k → 0};
SetAttributes[{SS, SST}, HoldAll];   (* T to lower T *)
T2t = {T_i_ → e^(ħ t_i), T → e^(ħ t)}; q_ħ = e^(γ ϵ ħ);
t2T = {e^(c_. t_i_ + b_.) :→ T_i^(c/ħ) e^b, e^(c_. t + b_.) :→ T^(c/ħ) e^b, e^ℰ_ :→ e^Expand@ℰ};
SS[ℰ_, op_] := Collect[   (* t to upper T *)
   Normal@Series[If[$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];
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[ℰ_] := ExpandDenominator@
   ExpandNumerator@
    Together[Expand[ℰ] //. e^x_ e^y_ :→ e^(x+y) /. e^x_ :→ e^CF[x]];
```

```
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];
SP_{}[P_] := P;
SP_{ξ_→x_, ps___}[P_] := Expand[P // SP_{ps}] /. f_. ξ^d_. :→ ∂_{x,d} f
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};
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};
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}]
AD$f = γ (Cosh[ħ (a ϵ + γϵ/2 - t/2)] - Cosh[ħ Sqrt[((t-γϵ)/2)^2 + ϵ ω]]) /
   (ħ e^(ħ ((a+γ) ϵ - t/2)) Sinh[γϵħ/2] (a^2 ϵ + a γ ϵ - a t - ω));
AD$ω = γ CU[y, x] + ϵ CU[a, a] - (t - γ ϵ) CU[a];
DeclareMorphism[AD, QU → CU,
  {a → a_CU, x → CU@x,
   y :→ S_CU[SS[AD$f], a → a_CU, ω → AD$ω] ** y_CU}]
SD$g = Sqrt[(2 γ (Cosh[ħ/2 Sqrt[t^2 + γ^2 ϵ^2 + 4 ϵ ϖ]] - Cosh[(t-ϵγ-2ϵa)/(2/ħ)])) /
   (Sinh[γϵħ/2] (t (2 a + γ) - 2 a (a + γ) ϵ + 2 ϖ) ħ)];
SD$f = Simplify[e^(ħ (t/2 - ϵ a)) (SD$g /. {a → -a, t → -t})];
SD$ϖ = γ CU[y, x] + ϵ CU[a, a] - (t - γ ϵ) CU[a] - t γ 1_CU/2;
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 }]
```

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

Fear Not. If $G = e^{\xi x} y e^{-\xi x}$ then $F = e^{-\eta y} e^{\xi x} e^{\eta y} e^{-\xi x} = e^{-\eta y} e^{\eta G}$ satisfies $\partial_\eta F = -yF + FG$ and $F_{\eta=0} = 1$:

```
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, $k}, {i, 0, l},
       {j, 0, l}, {k, 0, l}];
    F = fs.(bs = fs /. f_l_,i_,j_,k_[η] :→ ϵ^l U@{y^i, a^j, x^k});
    es = Flatten[Table[Coefficient[e, b] == 0,
       {e, {F - 1_U /. η → 0, F ** G - y_U ** F - ∂_η F}},
       {b, bs}]];
    F = F /. DSolve[es, fs, η][[1]];
    Œ_U[{y, a, x},
      ξ x + η y + (U /. {CU → -t η ξ, QU → η ξ (1 - T) / ℏ}),
      F + 𝕆_$k /. {e- → 1, U → Times}
   ] /. {ξ → ξ1, η → η1}]];
Simp[Œ_U_[specs___, Q_, P_]] := Œ_U[specs, CF[Q], CF[P]];
```

```
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}];
   Œ_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]] + 𝕆_k] /.
    {d → ∂_υ,ω q} /. {υ → υ1, ω → ω1}];
```

```
Rord_u_i_,w_j_→k_[Œ_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}];
   Œ_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_u_i_,w_j_→k_[Œ_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};*)
   Œ_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}];
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))]
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}]
```

```
e_q_,k_[x_] := e^(Σ_{j=1}^{k+1} (1 - q)^j x^j / (j (1 - q^j))); e_q_[x_] := e_q,$k[x]
QU[R_i_,j_] := 𝕆_QU[{y_1, a_1}_i, {a_2, x_2}_j,
   SS[e^(ℏ b_1 a_2) e_q_ℏ[ℏ y_1 x_2] /. b_1 → γ^-1 (ϵ a_1 - t_i)]];
QU[R^-1_i_,j_] := S_j@QU[R_i,j];
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_q_ℏ,k[ℏ y_i x_j] /. b_i → γ^-1 (ϵ a_i - t_i)), {ϵ, 0, k}]]
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]];
```

Task. Define $\mathrm{Exp}_{U_i,k}[\xi, P]$ which computes $e^{\xi\,\mathbb{O}(P)}$ 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 $U@\mathrm{Exp}_{U_i,k}[\xi, P]$ == $\mathbb{S}_U[e^{\xi x},\ x \to \mathbb{O}(P)]$.

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.

```
(* 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_] := 𝔈_U[{y_i, a_i, x_i}_i, Normal@P /. ϵ → 0,
   1 + 0_0];
Exp_{U_i_,k_}[ξ_, P_] :=
 Module[{yax = {y_i, a_i, x_i}, 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 @@ yax^{js});
  rhs =
   Normal@
    Last@m_{i,j→i}[𝔈_U[yax_i, ξ P0, F + 0_k]
      m_{i→j}@𝔈_U[{y_i, a_i, x_i}_i, 0, P + 0_k]];
  at0 = (# == 0) & /@
    Flatten@CoefficientList[F - 1 /. ξ → 0, yax];
  atξ = (# == 0) & /@
    Flatten@CoefficientList[(∂_ξ F) + P0 F - rhs, yax];
  𝔈_U[yax_i, ξ P0, F + 0_k] /.
   DSolve[And @@ (at0 ⋃ atξ), φs, ξ][[1]] ]
```

To do. • Consider renormalizing x and y. • Can everything be done at $\hbar = 1$ defining a filtration by other means? That ought to be possible as the end results depend on t/T and not on $\hbar$. • Bound the degrees of the logoi! • $r = \theta r$? • θ is a global symmetry. Can it be "gauged"? • Global $\eta \to \psi$?

Alternative Algorithms.

```
λ_{alt,k_}[CU] := If[k == 0, 1, Module[{eq, d, b, c, so},
    eq = ρ@e^{ξ x_CU}.ρ@e^{η y_CU} == ρ@e^{d y_CU}.ρ@e^{c (t 1_CU - 2 ϵ a_CU)}.ρ@e^{b x_CU};
    {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}]]];
```

Program (as in Projects/PPSA/Verification.nb).

```
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];
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; )
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];
```

```
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];
  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[])]; ]
```

Asides.

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

$$\frac{1-\mathrm{T}}{\hbar} + 2\,\mathrm{T}\,\epsilon\,a - 2\left(\mathrm{T}\,\epsilon^2\,\hbar\right)a^2 + \frac{4}{3}\,\mathrm{T}\,\epsilon^3\,\hbar^2\,a^3 + O[a]^4$$

$\mathcal{GDO}$-Categories. Given $\mathfrak{g}$ with basis $B=\{x,y,\dots\}$, consider the following diagram:

$$\begin{tikzcd}
\mathbb{Q}=\hat{\mathcal{U}}_{(q)}\left(\bigoplus_0\mathfrak{g}\right) \arrow[r, "Z"] & \hat{\mathcal{U}}_{(q)}\left(\mathfrak{g}\right) \arrow[r, shift left, "\Delta"] & \hat{\mathcal{U}}_{(q)}\left(\bigoplus_2\mathfrak{g}\right) \arrow[l, shift left, "m"] \\
\hat{\mathcal{S}}\left(\emptyset\right) \arrow[u] \arrow[r, "Z"] & \hat{\mathcal{S}}\left(B\right) \arrow[u, bend left, "\mathbb{O}(xy\ldots:\,\cdot)"] \arrow[u, bend right, "\mathbb{O}(yx\ldots:\,\cdot)"'] \arrow[r, shift left, "\Delta"] \arrow[loop below, "SW_{xy}"] & \hat{\mathcal{S}}\left(B_1,B_2\right) \arrow[l, shift left, "m"] \arrow[u, "\mathbb{O}(y_1x_1\ldots\otimes y_2x_2\ldots:\,\cdot)"']
\end{tikzcd}$$

Hence Z, SW_{xy}, m, Δ, (and likewise S and θ) are morphisms in the *completion* of the monoidal category $\mathcal{F}$ whose objects are finite sets B and whose morphism are $\operatorname{mor}_{\mathcal{F}}(B,B'):=\operatorname{Hom}_{\mathbb{Q}}\left(\mathcal{S}(B)\to\mathcal{S}(B')\right)=\mathcal{S}\left(B^*,B'\right)$ (by convention, $x^*=\xi$, $y^*=\eta$, etc.). Ergo we need to *consolidate* (at least parts of) said completion.

Aside. "Consolidate" means "give a finite name to an infinite object, and figure out how to sufficiently manipulate such finite names". E.g., solving $f''=-f$ we encounter and set $\sum\frac{(-1)^kx^{2k}}{(2k)!}\rightsquigarrow\cos x$, $\sum\frac{(-1)^kx^{2k+1}}{(2k+1)!}\rightsquigarrow\sin x$, and then $\cos^2x+\sin^2x=1$ and $\sin(x+y)=\sin x\cos y+\cos x\sin y$.

Example.

Example. In $QU/(\epsilon^2=0)$ using the *yax* order over $\mathbb{Q}[\![\hbar]\!]$, with $T=\mathbb{e}^{\hbar t}$, $\bar{T}=T^{-1}$, $\mathcal{A}=\mathbb{e}^{\gamma\alpha}$, and $\bar{\mathcal{A}}=\mathcal{A}^{-1}$,

$$R_{ij}=\mathbb{e}^{\hbar(y_ix_j-t_ia_j/\gamma)}\left(1+\epsilon\hbar\left(a_ia_j/\gamma-\gamma\hbar^2y_i^2x_j^2/4\right)\right)\in\mathcal{S}(B_i,B_j),$$

$$m=\mathbb{e}^{(\alpha_1+\alpha_2)a+\eta_2\xi_1(1-T)/\hbar+(\xi_1\bar{\mathcal{A}}_2+\xi_2)x+(\eta_1+\eta_2\bar{\mathcal{A}}_1)y}\left(1+\epsilon\lambda_m\right)\in\mathcal{S}(B_1^*,B_2^*,B),$$

with $\lambda_m=2a\eta_2\xi_1T+\frac{1}{4}\gamma\eta_2^2\xi_1^2\left(3T^2-4T+1\right)/\hbar-\frac{1}{2}\gamma\eta_2\xi_1^2(3T-1)x\bar{\mathcal{A}}_2-\frac{1}{2}\gamma\eta_2^2\xi_1(3T-1)y\bar{\mathcal{A}}_1+\gamma\eta_2\xi_1xy\hbar\bar{\mathcal{A}}_1\bar{\mathcal{A}}_2$,

$$\Delta=\mathbb{e}^{\tau(t_1+t_1)+\eta(y_1+T_1y_2)+\alpha(a_1+a_2)+\xi(x_1+x_2)}\left(1+\epsilon\lambda_\Delta\right)\in\mathcal{S}(B^*,B_1,B_2),$$

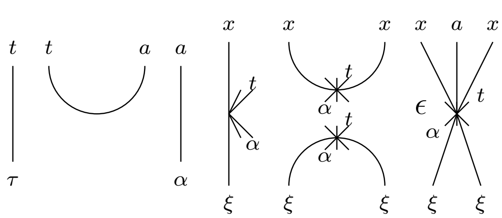
with $\lambda_\Delta=-a_1\eta T_1y_2\hbar-a_1\xi x_2\hbar+\frac{1}{2}\gamma\eta^2T_1y_1y_2\hbar+\frac{1}{2}\gamma\xi^2x_1x_2\hbar$, and

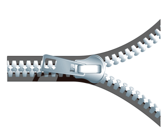
$$S=\mathbb{e}^{-\tau t-\alpha a-\eta\xi(1-\bar{T})\mathcal{A}/\hbar-\bar{T}\eta y\mathcal{A}-\xi x\mathcal{A}}\left(1+\epsilon\lambda_S\right)\in\mathcal{S}(B^*,B),$$

with $\lambda_S=2\bar{T}\mathcal{A}a\eta\xi-\bar{T}\mathcal{A}a\eta y\hbar-a\xi x\hbar\mathcal{A}-\frac{1}{4}\gamma\eta^2\xi^2\left(1-4\bar{T}+3\bar{T}^2\right)\mathcal{A}^2/\hbar-\frac{1}{2}\gamma\eta^2y^2\hbar\bar{T}^2\mathcal{A}^2-\frac{1}{2}\gamma\eta^2\xi\bar{T}(1-3\bar{T})y\mathcal{A}^2+\gamma\eta\xi(1-\bar{T})\mathcal{A}-\frac{1}{2}\gamma\eta\xi^2(1-3\bar{T})x\mathcal{A}^2-\gamma\eta\xi xy\hbar\bar{T}\mathcal{A}^2+\gamma\eta y\hbar\bar{T}\mathcal{A}-\frac{1}{2}\gamma\xi x^2\hbar\mathcal{A}^2$.

Problem. Compute the likes of $m/\!\!/\Delta=\left(m|_{b\to\partial_\beta}\Delta\right)_{\beta=0}$ and $(R_{12}R_{34})/\!\!/m_2^{13}=\left((R_{12}R_{34})|_{b\to\partial_\beta}m_2^{13}\right)_{\beta=0}$.

A generic morphism:





The Zipping Issue.

The Contraction Theorem. If P has a finite ζ-degree and the y's and the q's are "small",

$$\left\langle P(z_i,\zeta^j)\right\rangle_{(\zeta_i)}=P\left(z_i,\overleftrightarrow{\partial}_{z_j}\right)\Big|_{z_i=0},$$

$$\left\langle P(z_i,\zeta^j)\mathbb{e}^{\eta^iz_i+y_j\zeta^j}\right\rangle_{(\zeta_i)}=\left\langle P(z_i+y_i,\zeta^j)\mathbb{e}^{\eta^i(z_i+y_i)}\right\rangle_{(\zeta_i)},$$

(proof: replace $y_j\to\hbar y_j$ and test at $\hbar=0$ and at $\partial_\hbar$), and

$$\left\langle P(z_i,\zeta^j)\mathbb{e}^{c+\eta^iz_i+y_j\zeta^j+q^i_jz_i\zeta^j}\right\rangle_{(\zeta_i)}=\det(\tilde{q})\left\langle P(\tilde{q}^k_i(z_k+y_k),\zeta^j)\mathbb{e}^{c+\eta^i\tilde{q}^k_i(z_k+y_k)}\right\rangle_{(\zeta_i)}$$

where $\tilde{q}$ is the inverse matrix of $1-q$: $(\delta^i_j-q^i_j)\tilde{q}^j_k=\delta^i_k$ (proof: replace $q^i_j\to\hbar q^i_j$ and test at $\hbar=0$ and at $\partial_\hbar$).

```
𝔼 /: 𝔼[L1_, Q1_, P1_] ≡ 𝔼[L2_, Q2_, P2_] :=
  Simplify[L1 == L2] ∧ Simplify[Q1 == Q2] ∧
   Simplify[Normal[P1 - P2] == 0];
𝔼 /: 𝔼[L1_, Q1_, P1_] 𝔼[L2_, Q2_, P2_] :=
  𝔼[L1 + L2, Q1 + Q2, P1 ** P2];
{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_{}[P_] := P;
Zip_{ζ_,ζs___}[P_] :=
 (Expand[P // Zip_{ζs}] /. f_. ζ^d_. :> ∂_{ζ*,d} f) /. ζ* → 0
𝔼 /: QZip_{ζs_List}@𝔼[L_, Q_, P_] :=
  Module[{ζ, z, zs, c, ys, ηs, qt, zrule, Q1, Q2},
   zs = Table[ζ*, {ζ, ζs}];
   c = Q /. Alternatives @@ (ζs ⋃ zs) → 0;
   ys = Table[∂_ζ (Q /. Alternatives @@ zs → 0), {ζ, ζs}];
   ηs = Table[∂_z (Q /. Alternatives @@ ζs → 0), {z, zs}];
   qt = Inverse@Table[Kδ_{z,ζ*} - ∂_{z,ζ} Q, {ζ, ζs}, {z, zs}];
   zrule = Thread[zs → qt.(zs + ys)];
   Q2 = (Q1 = c + ηs.zs /. zrule) /. Alternatives @@ zs → 0;
   Simplify /@
    𝔼[L, Q2, Det[qt] e^(-Q2) Zip_{ζs}[e^Q1 (P /. zrule)]] ];
𝔼 /: LZip_{ζs_List}@𝔼[L_, Q_, P_] :=
  Module[{ζ, z, zs, c, ys, ηs, lt, zrule, L1, L2, Q1, Q2},
   zs = Table[ζ*, {ζ, ζs}];
   c = L /. Alternatives @@ (ζs ⋃ zs) → 0;
   ys = Table[∂_ζ (L /. Alternatives @@ zs → 0), {ζ, ζs}];
   ηs = Table[∂_z (L /. Alternatives @@ ζs → 0), {z, zs}];
   lt = Inverse@Table[Kδ_{z,ζ*} - ∂_{z,ζ} L, {ζ, ζs}, {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;
   Simplify /@
    𝔼[L2, Q2, Det[lt] e^(-L2-Q2)
      Zip_{ζs}[e^(L1+Q1) (P /. T2t /. zrule)]] //. t2T ];
Bind_{is___Integer}[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}}]} ];
Bind[ℰ_𝔼] := ℰ;
Bind[Ls__, ζs_List, R_] := Bind_{ζs}[Bind[Ls], R];
m[U_, kk_]_{i_,j_→k_} := m[U, kk]_{i,j→k} = Module[{OE},
    OE = Simplify /@
      m_{i,j→k}@Œ_U[{y_i, a_i, x_i}_i, {y_j, a_j, x_j}_j,
        η_i y_i + α_i a_i + ξ_i x_i + η_j y_j + α_j a_j + ξ_j x_j, 1 + O_kk];
    𝔼[t_k (τ_i + τ_j) + (OE⟦2⟧ /. (ξ | η)_{i|j} → 0),
     OE⟦2⟧ /. a_k → 0, OE⟦3⟧] ];
tm_{i_,j_→k_} := m[$U, $k]_{i,j→k};
```

```
R[U_, kk_]i_,j_ := R[U, kk]i,j→k = Module[{OE},
    OE = Simplify /@ ŒU,kk@Ri,j;
    𝔼[-(ℏ aj ti)/γ, ℏ xj yi, Last@OE] ];
tRi_,j_ := R[$U, $k]i,j;
```