In[∘]:=
```
PP_ = Identity; $k = 0; γ = γ; ℏ;
```

In[∘]:=
```
Once[<< KnotTheory`];
```

```
Loading KnotTheory` version of September 6, 2014, 13:37:37.2841.
Read more at http://katlas.org/wiki/KnotTheory.
```

The "Speedy" Engine

Internal Utilities

Canonical Form:

In[∘]:=
```
CCF[ℰ_] := ExpandDenominator@ExpandNumerator@Together[
    Expand[ℰ] //. e^x_ e^y_ :> e^(x+y) /. e^x_ :> e^CCF[x]];
CF[ℰ_List] := CF /@ ℰ;
CF[sd_SeriesData] := MapAt[CF, sd, 3];
CF[ℰ_] := Module[
   {vs = Cases[ℰ, (y | b | t | a | x | η | β | τ | α | ξ)_, ∞] ⋃ {y, b, t, a, x, η, β, τ, α, ξ}},
   Total[CoefficientRules[Expand[ℰ], vs] /. (ps_ → c_) :> CCF[c] (Times @@ vs^ps)]
  ];
CF[ℰ_𝔼] := CF /@ ℰ; CF[𝔼_sp___[ℰs___]] := CF /@ 𝔼_sp[ℰs];
```

The Kronecker δ:

In[∘]:=
```
Kδ /: Kδ_{i_,j_} := If[i === j, 1, 0];
```

Equality, multiplication, and degree-adjustment of perturbed Gaussians; $\mathbb{E}[L, Q, P]$ stands for $e^{L+Q} P$:

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];
𝔼[L_, Q_, P_]_$k_ := 𝔼[L, Q, Series[Normal@P, {ϵ, 0, $k}]];
```

Zip and Bind

Variables and their duals:

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

Upper to lower and lower to Upper:

In[∘]:=
```
U2l = {B_i_^p_. :> 𝕖^(-p ħ γ b_i), B^p_. :> 𝕖^(-p ħ γ b), T_i_^p_. :> 𝕖^(-p ħ t_i), T^p_. :> 𝕖^(-p ħ t), 𝒜_i_^p_. :> 𝕖^(p γ α_i), 𝒜^p_. :> 𝕖^(p γ α)};
l2U = {𝕖^(c_. b_i_+d_.) :> B_i^(-c/(ħ γ)) 𝕖^d, 𝕖^(c_. b+d_.) :> B^(-c/(ħ γ)) 𝕖^d,
    𝕖^(c_. t_i_+d_.) :> T_i^(-c/ħ) 𝕖^d, 𝕖^(c_. t+d_.) :> T^(-c/ħ) 𝕖^d,
    𝕖^(c_. α_i_+d_.) :> 𝒜_i^(c/γ) 𝕖^d, 𝕖^(c_. α+d_.) :> 𝒜^(c/γ) 𝕖^d,
    𝕖^ℰ_ :> 𝕖^Expand@ℰ};
```

Derivatives in the presence of exponentiated variables:

In[∘]:=
```
D_b[f_] := ∂_b f - ħ γ B ∂_B f; D_b_i_[f_] := ∂_b_i f - ħ γ B_i ∂_B_i f;
D_t[f_] := ∂_t f - ħ T ∂_T f; D_t_i_[f_] := ∂_t_i f - ħ T_i ∂_T_i f;
D_α[f_] := ∂_α f + γ 𝒜 ∂_𝒜 f; D_α_i_[f_] := ∂_α_i f + γ 𝒜_i ∂_𝒜_i f;
D_v_[f_] := ∂_v f; D_{v_,0}[f_] := f; D_{}[f_] := f; D_{v_,n_Integer}[f_] := D_v[D_{v,n-1}[f]];
D_{l_List,ls___}[f_] := D_{ls}[D_l[f]];
```

Finite Zips:

In[∘]:=
```
collect[sd_SeriesData, ξ_] := MapAt[collect[#, ξ] &, sd, 3];
collect[ℰ_, ξ_] := Collect[ℰ, ξ];
Zip_{}[P_] := P;
Zip_ξs_[Ps_List] := Zip_ξs /@ Ps;
Zip_{ξ_,ξs___}[P_] :=
  (collect[P // Zip_{ξs}, ξ] /. f_. ξ^d_. :> (D_{ξ*,d}[f])) /. ξ* → 0 /.
   ((ξ* /. {b → B, t → T, α → 𝒜}) → 1)
```

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

$$\left\langle P(z_i,\zeta^j)e^{c+\eta^i z_i+y_j\zeta^j+q^i_j z_i\zeta^j}\right\rangle = |\tilde{q}|\left\langle P(z_i,\zeta^j)e^{c+\eta^i z_i}\Big|_{z_i\to\tilde{q}^k_i(z_k+y_k)}\right\rangle$$
$$= |\tilde{q}|e^{c+\eta^i\tilde{q}^k_i y_k}\left\langle P\left(\tilde{q}^k_i(z_k+y_k),\zeta^j+\eta^i\tilde{q}^j_i\right)\right\rangle.$$

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

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 b and α and the ζ's are β and a.

In[∘]:=
```
LZip_{ζs_List}@𝔼[L_, Q_, P_] :=
  Module[{ζ, z, zs, Zs, c, ys, ηs, lt, zrule, Zrule, ζrule, Q1, EEQ, EQ},
   zs = Table[ζ*, {ζ, ζs}];
   Zs = zs /. {b → B, t → T, α → 𝒜};
   c = L /. Alternatives @@ (ζs ⋃ zs) → 0 /. Alternatives @@ Zs → 1;
   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)];
   Zrule = Join[zrule,
     zrule /. r_Rule :→ ((U = r〚1〛 /. {b → B, t → T, α → 𝒜}) → (U /. U2l /. r //. l2U))];
   ζrule = Thread[ζs → ζs + ηs.lt];
   Q1 = Q /. (Zrule ⋃ ζrule);
   EEQ[ps___] := EEQ[ps] =
     (CF[ⅇ^{-Q1} D_{Thread[{zs,{ps}}]}[ⅇ^{Q1}]] /. {Alternatives @@ zs → 0, Alternatives @@ Zs → 1});
   CF@𝔼[c + ηs.lt.ys, Q1 /. {Alternatives @@ zs → 0, Alternatives @@ Zs → 1},
     Det[lt] (Zip_{ζs}[(EQ @@ zs) (P /. (Zrule ⋃ ζrule))] /.
         Derivative[ps___][EQ][___] :→ EEQ[ps] /. _EQ → 1) ]];
```

In[∘]:=
```
B_{}[L_, R_] := L R;
B_{is__}[L_𝔼, R_𝔼] := Module[{n},
   Times[
      L /. Table[(v : b | B | t | T | a | x | y)_i → v_{n@i}, {i, {is}}],
      R /. Table[(v : β | τ | α | 𝒜 | ξ | η)_i → v_{n@i}, {i, {is}}]
    ] // LZip_{Join@@Table[{β_{n@i},τ_{n@i},a_{n@i}},{i,{is}}]} // QZip_{Join@@Table[{ξ_{n@i},y_{n@i}},{i,{is}}]} ];
B_{is___}[L_, R_] := B_{{is}}[L, R];
```

𝔼 morphisms with domain and range.

In[∘]:=
```
B_{is_List}[𝔼_{d1_→r1_}[L1_, Q1_, P1_], 𝔼_{d2_→r2_}[L2_, Q2_, P2_]] :=
  𝔼_{(d1⋃Complement[d2,is])→(r2⋃Complement[r1,is])} @@ B_{is}[𝔼[L1, Q1, P1], 𝔼[L2, Q2, P2]];
𝔼_{d1_→r1_}[L1_, Q1_, P1_] // 𝔼_{d2_→r2_}[L2_, Q2_, P2_] :=
  B_{r1⋂d2}[𝔼_{d1→r1}[L1, Q1, P1], 𝔼_{d2→r2}[L2, Q2, P2]];
𝔼_{d1_→r1_}[L1_, Q1_, P1_] ≡ 𝔼_{d2_→r2_}[L2_, Q2_, P2_] ^:=
  (d1 == d2) ∧ (r1 == r2) ∧ (𝔼[L1, Q1, P1] ≡ 𝔼[L2, Q2, P2]);
𝔼_{d1_→r1_}[L1_, Q1_, P1_] 𝔼_{d2_→r2_}[L2_, Q2_, P2_] ^:=
  𝔼_{(d1⋃d2)→(r1⋃r2)} @@ (𝔼[L1, Q1, P1] 𝔼[L2, Q2, P2]);
𝔼_{dr_}[L_, Q_, P_]_{$k_} := 𝔼_{dr} @@ 𝔼[L, Q, P]_{$k};
𝔼_[ℰ___][i_] := {ℰ}〚i〛;
R_{i_,j_} := 𝔼_{{}→{i,j}}[0, (y_i - y_j) x_j, 1 + ϵ (x_i y_i - x_j y_i - x_j y_j) +
    ϵ^2 (Sum[r_{If[#===i,0,1]&@k,If[#===i,0,1]&@l} y_k x_l, {k, {i, j}}, {l, {i, j}}] +
      Sum[s_{If[#===i,0,1]&@k,If[#===i,0,1]&@l,If[#===i,0,1]&@m,If[#===i,0,1]&@n} y_k x_l y_m x_n,
       {k, {i, j}}, {l, {i, j}}, {m, {i, j}}, {n, {i, j}}]) + O[ϵ]^3] /.
  {r_{1,0} → 0, r_{1,1} → 1, r_{0,0} → -1, r_{0,1} → 0, s_{0,0,0,0} → 1/2, s_{1,0,1,0} → 0, s_{1,0,0,0} → 1,
   s_{0,0,1,0} → -1, s_{1,1,1,1} → 1/2, s_{1,1,1,0} → 1, s_{1,0,1,1} → -1, s_{0,0,1,1} → 1, s_{0,1,0,1} → 1, s_{0,1,1,0} → -1,
   s_{1,0,0,1} → 1, s_{1,1,0,0} → -1, s_{0,0,0,1} → -1, s_{0,1,0,0} → -1, s_{0,1,1,1} → 1, s_{1,1,0,1} → -1/2}
```

“Define” Code

Define[lhs = rhs, ...] defines the lhs to be rhs, except that rhs is computed only once for each value of $k. Fancy Mathematica not for the faint of heart. Most readers should ignore.

In[∘]:=
```
SetAttributes[Define, HoldAll];
Define[def_, defs__] := (Define[def]; Define[defs];);
Define[op_{is__} = ℰ_] := Module[{SD, ii, jj, kk, isp, nis, nisp, sis}, Block[{i, j, k},
   ReleaseHold[Hold[
      SD[op_{nisp,$k_Integer}, Block[{i, j, k}, op_{isp,$k} = ℰ; op_{nis,$k}]];
      SD[op_{isp}, op_{{is},$k}]; SD[op_{sis__}, op_{{sis}}];
     ] /. {SD → SetDelayed,
      isp → {is} /. {i → i_, j → j_, k → k_},
      nis → {is} /. {i → ii, j → jj, k → kk},
      nisp → {is} /. {i → ii_, j → jj_, k → kk_}
     }] ]]
```

In[∘]:=
```
Define[m_{i,j→k} = 𝔼_{{i,j}→{k}}[0, -ξ_i η_j + (η_i + η_j) y_k + (ξ_i + ξ_j) x_k, 1]]
(*Heisenberg multiplication*)
```

The docile R-matrix and its inverse

In[∘]:=
```
Once[RRules = {}; RiRules = {}; CCRules = {β → 0, δ → 0};]
```

In[•]:=
```
R_{i_,j_} :=
  𝔼_{{}→{i,j}}[0, (y_i - y_j) (T - 1) x_j, 1 + ϵ (u x_i y_i + v x_j y_i + w x_i y_j + z x_j y_j) + O[ϵ]^2] /. RRules
R̄_{i_,j_} := 𝔼_{{}→{i,j}}[0, (y_i - y_j) (T^-1 - 1) x_j,
    1 + ϵ (a x_i y_i + b x_j y_i + c x_i y_j + d x_j y_j) + O[ϵ]^2] /. RiRules
CC_{i_} := 𝔼_{{}→{i}}[0, 0, √T + (α + β x_i y_i) ϵ + O[ϵ]^2] /. CCRules
CC̄_{i_} := 𝔼_{{}→{i}}[0, 0, (√T)^-1 + (γ + δ x_i y_i) ϵ + O[ϵ]^2] /. CCRules
Kink_{i_} := CC_3 R_{1,2} // m_{2,3→2} // m_{2,1→i}
Kink̄_{i_} := CC_3 R̄_{1,2} // m_{1,3→1} // m_{1,2→i}
```

In[•]:=
```
Coefficient[(R_{1,2} R_{4,3} R_{5,6} // m_{1,4→1} // m_{2,5→2} // m_{3,6→3})⟦3⟧ -
    (R_{2,3} R_{4,5} R_{1,6} // m_{1,4→1} // m_{2,5→2} // m_{3,6→3})⟦3⟧ // Normal, ϵ]
```

Out[•]= $-(u - T u + v)\, x_2 y_1 + (u - T u + v + w - T w)\, x_2 y_1 - v\, x_3 y_1 + (T u - T^2 u + 2 v - T v + z - T z)\, x_3 y_1 + w\, x_1 y_2 - (2 w - T w)\, x_1 y_2 + (T u + z)\, x_2 y_2 - (T u + w - 2 T w + T^2 w + z)\, x_2 y_2 - (T v + z - T z)\, x_3 y_2 + (T v + T w - T^2 w + z - T z)\, x_3 y_2 + w\, x_1 y_3 - T w\, x_1 y_3 + T w\, x_2 y_3 - (2 T w - T^2 w)\, x_2 y_3$

In[•]:=
```
Solve[Thread[CoefficientRules[Coefficient[(R_{1,2} R_{4,3} R_{5,6} // m_{1,4→1} // m_{2,5→2} // m_{3,6→3})⟦3⟧ -
        (R_{2,3} R_{4,5} R_{1,6} // m_{1,4→1} // m_{2,5→2} // m_{3,6→3})⟦3⟧ // Normal, ϵ],
      {x_1, x_2, x_3, y_1, y_2, y_3}]⟦;; , 2⟧ == 0], {u, v, w, z}]
```

Solve: Equations may not give solutions for all "solve" variables.

Out[•]= $\{\{w \to 0,\ z \to -T u - v\}\}$

In[•]:=
```
RRules = {w → 0, z → -T u - v}; R_{1,2} // CF
```

Out[•]= $\mathbb{E}_{\{\}\to\{1,2\}}\left[0,\ (-1 + T)\, x_2 y_1 + (1 - T)\, x_2 y_2,\ 1 + (u\, x_1 y_1 + v\, x_2 y_1 + (-T u - v)\, x_2 y_2)\, \epsilon + O[\epsilon]^2\right]$

In[•]:=
```
Solve[Thread[CoefficientRules[Coefficient[(R_{1,2} R̄_{3,4} // m_{1,3→1} // m_{2,4→2})⟦3⟧ // Normal, ϵ],
      {x_1, x_2, y_1, y_2}]⟦;; , 2⟧ == 0], {a, b, c, d}]
```

Out[•]= $\left\{\left\{a \to -u,\ b \to -\frac{v}{T^2},\ c \to 0,\ d \to -\frac{-T u - v}{T^2}\right\}\right\}$

In[•]:=
```
RiRules = {a → -u, b → -v/T^2, c → 0, d → -(-T u - v)/T^2}; R̄_{1,2} // CF
```

Out[•]= $\mathbb{E}_{\{\}\to\{1,2\}}\left[0,\ \frac{(1 - T)\, x_2 y_1}{T} + \frac{(-1 + T)\, x_2 y_2}{T},\ 1 + \left(-u\, x_1 y_1 - \frac{v\, x_2 y_1}{T^2} + \frac{(T u + v)\, x_2 y_2}{T^2}\right) \epsilon + O[\epsilon]^2\right]$

In[•]:=
```
CC_1 CC̄_2 // m_{1,2→1} (*Requirements on CC to nail it down*)
Kink_1 ≡ (CC̄_3 R_{1,2} // m_{1,3→1} // m_{1,2→1})
```

Out[•]= $\mathbb{E}_{\{\}\to\{1\}}\left[0,\ 0,\ 1 + \left(\frac{\alpha}{\sqrt{T}} + \sqrt{T}\, \gamma\right) \epsilon + O[\epsilon]^2\right]$

Out[•]= $\frac{-v\, \epsilon + \sqrt{T}\, \alpha\, \epsilon - T^{3/2}\, \gamma\, \epsilon}{T^{3/2}} == 0$

In[∘]:= $\text{Solve}\left[\left\{-v + \sqrt{T}\,\alpha - T^{3/2}\gamma == 0,\ \left(\frac{\alpha}{\sqrt{T}} + \sqrt{T}\,\gamma\right) == 0\right\}, \{\alpha, \gamma\}\right]$

Out[∘]= $\left\{\left\{\alpha \to \frac{v}{2\sqrt{T}},\ \gamma \to -\frac{v}{2\,T^{3/2}}\right\}\right\}$

In[∘]:= $\text{CCRules} = \left\{\beta \to 0,\ \delta \to 0,\ \alpha \to \frac{v}{2\sqrt{T}},\ \gamma \to -\frac{v}{2\,T^{3/2}}\right\};\ CC_1$

Out[∘]= $\mathbb{E}_{\{\}\to\{1\}}\left[0,\ 0,\ \sqrt{T} + \frac{v\,\epsilon}{2\sqrt{T}} + O[\epsilon]^2\right]$

OC fails

In[∘]:= $(R_{1,2}\,R_{4,3}\ //\ m_{1,4\to1}) \equiv (R_{1,3}\,R_{4,2}\ //\ m_{1,4\to1})\ //\ \text{Simplify}$

Out[∘]= $(-1 + T)\,u\,\epsilon\,(x_2 - x_3)\,y_1 == 0$

R2 braid-like

In[∘]:= $R_{1,2}\,\overline{R}_{3,4}\ //\ m_{1,3\to1}\ //\ m_{2,4\to2}$
$\overline{R}_{1,2}\,R_{3,4}\ //\ m_{1,3\to1}\ //\ m_{2,4\to2}$

Out[∘]= $\mathbb{E}_{\{\}\to\{1,2\}}\left[0,\ 0,\ 1 + O[\epsilon]^2\right]$

Out[∘]= $\mathbb{E}_{\{\}\to\{1,2\}}\left[0,\ 0,\ 1 + O[\epsilon]^2\right]$

‘naked’ R2 cyclic (!!):

In[∘]:= $R_{3,2}\,\overline{R}_{1,4}\ //\ m_{1,3\to1}\ //\ m_{2,4\to2}$
$R_{1,4}\,\overline{R}_{3,2}\ //\ m_{1,3\to1}\ //\ m_{2,4\to2}$

Out[∘]= $\mathbb{E}_{\{\}\to\{1,2\}}\left[0,\ 0,\ 1 + O[\epsilon]^2\right]$

Out[∘]= $\mathbb{E}_{\{\}\to\{1,2\}}\left[0,\ 0,\ 1 + O[\epsilon]^2\right]$

Proper R2 cyclic:

In[∘]:= $\left(CC_3\,R_{5,2}\,\overline{R}_{1,4}\ //\ m_{1,3\to1}\ //\ m_{1,5\to1}\ //\ m_{2,4\to2}\right) \equiv CC_1\,\mathbb{E}_{\{\}\to\{2\}}\left[0,\ 0,\ 1 + O[\epsilon]^2\right]$
$\left(\overline{CC}_4\,R_{1,6}\,\overline{R}_{3,2}\ //\ m_{1,3\to1}\ //\ m_{2,4\to2}\ //\ m_{2,6\to2}\right) \equiv \overline{CC}_2\,\mathbb{E}_{\{\}\to\{1\}}\left[0,\ 0,\ 1 + O[\epsilon]^2\right]$

Out[∘]= True

Out[∘]= True

R3:

In[∘]:= $(R_{1,2}\,R_{4,3}\,R_{5,6}\ //\ m_{1,4\to1}\ //\ m_{2,5\to2}\ //\ m_{3,6\to3}) \equiv (R_{2,3}\,R_{4,5}\,R_{1,6}\ //\ m_{1,4\to1}\ //\ m_{2,5\to2}\ //\ m_{3,6\to3})$

Out[∘]= True

Pairwise equality of the four kinks:

In[∘]:=
```
Kink₁ ≡ (CC̄₃ R₁,₂ // m₁,₃→₁ // m₁,₂→₁)
Kink̄₁ ≡ (CC̄₃ R̄₁,₂ // m₂,₃→₂ // m₂,₁→₁)
```

Out[∘]= True

Out[∘]= True

Trefoils?:

In[∘]:=
```
Kink̄₈ Kink̄₉ Kink̄₁₀ CC̄₇ R₁,₄ R₅,₂ R₃,₆ // m₁,₂→₁ // m₁,₃→₁ // m₁,₇→₁ // m₁,₄→₁ // m₁,₅→₁ // m₁,₆→₁ //
    m₁,₈→₁ // m₁,₉→₁ // m₁,₁₀→₁
Kink₈ Kink₉ Kink₁₀ CC₇ R₄,₁ R₂,₅ R₆,₃ // m₁,₂→₁ // m₁,₃→₁ // m₁,₇→₁ // m₁,₄→₁ // m₁,₅→₁ // m₁,₆→₁ //
    m₁,₈→₁ // m₁,₉→₁ // m₁,₁₀→₁
```

Out[∘]= $\mathbb{E}_{\{\}\to\{1\}}\left[0,\ 0,\ \frac{T}{1-T+T^2}+\frac{\left(v-T^2 v\right)\epsilon}{1-2T+3T^2-2T^3+T^4}+O[\epsilon]^2\right]$

Out[∘]= $\mathbb{E}_{\{\}\to\{1\}}\left[0,\ 0,\ \frac{T}{1-T+T^2}+\frac{\left(v-T^2 v\right)\epsilon}{1-2T+3T^2-2T^3+T^4}+O[\epsilon]^2\right]$

Unfortunately? This timid invariant does not see the difference between the mirror trefoils. Perhaps it is actually determined by Alexander as Dror feared.

In[∘]:=
```
Kink₈ Kink₉ Kink₁₀ CC₄ R̄₁,₅ R̄₆,₂ R̄₃,₇ // m₁,₂→₁ // m₁,₃→₁ // m₁,₄→₁ // m₁,₅→₁ // m₁,₆→₁ // m₁,₇→₁ //
    m₁,₈→₁ // m₁,₉→₁ // m₁,₁₀→₁
```

Out[∘]= $\mathbb{E}_{\{\}\to\{1\}}\left[0,\ 0,\ \frac{T}{1-T+T^2}+\frac{\left(v-T^2 v\right)\epsilon}{1-2T+3T^2-2T^3+T^4}+O[\epsilon]^2\right]$

In[∘]:=
```
RVK[pd_PD] := Module[{n, xs, x, rots, front = {0}, k},
   n = Length@pd; rots = Table[0, {2 n}];
   xs = Cases[pd, x_X :> { Xp[x[[4]], x[[1]]]   PositiveQ@x
                           Xm[x[[2]], x[[1]]]   True        ];
   For[k = 0, k < 2 n, ++k, If[k == 0 ∨ FreeQ[front, -k],
     front = Flatten[front /. k → (xs /. {
            Xp[k + 1, l_] | Xm[l_, k + 1] :> {l, k + 1, 1 - l},
            Xp[l_, k + 1] | Xm[k + 1, l_] :> (++rots[[l]]; {1 - l, k + 1, l})
          })],
     Cases[front, k | -k] /. {k, -k} :> --rots[[k + 1]];
    ]];
   RVK[xs, rots] ];
RVK[K_] := RVK[PD[K]];
```

In[∘]:=
```
rot[i_, 0] := 𝔼{}→{i}[0, 0, 1];
rot[i_, n_] := Module[{j},
   rot[i, n] = If[n > 0, rot[i, n - 1] CCj, rot[i, n + 1] CC̄j] // mi,j→i];
```

In[∘]:=
```
Z[K_] := Z[RVK@K];
Z[rvk_RVK] := (*Z[rvk] =*)
 Module[{todo, n, rots, ξ, done, st, cx, ξ1, i, j, k, k1, k2, k3},
  {todo, rots} = List @@ rvk;
  AppendTo[rots, 0];
  n = Length[todo];
  ξ = 𝔼_{{}→{0}}[0, 0, 1];
  done = {0};
  st = Range[0, 2 n + 1];
  While[{} =!= ($M = todo),
   {cx} = MaximalBy[todo, Length[done ⋂ {#[[1]], #[[2]], #[[1]] - 1, #[[2]] - 1}] &, 1];
   {i, j} = List @@ cx;
   ξ1 = Switch[Head[cx],
      Xp, (R_{i,j} \overline{Kink_k}) // m_{j,k→j},
      Xm, (\overline{R}_{i,j} Kink_k) // m_{j,k→j}
     ];
   ξ1 = (rot[k, rots[[i]]] ξ1) // m_{k,i→i}; rots[[i]] = 0;
   ξ1 = (ξ1 rot[k, rots[[i + 1]]]) // m_{i,k→i}; rots[[i + 1]] = 0;
   ξ1 = (rot[k, rots[[j]]] ξ1) // m_{k,j→j}; rots[[j]] = 0;
   ξ1 = (ξ1 rot[k, rots[[j + 1]]]) // m_{j,k→j}; rots[[j + 1]] = 0;
   ξ *= ξ1;
   If[MemberQ[done, i], ξ = ξ // m_{i,i+1→i}; st = st /. st[[i + 2]] → st[[i + 1]]];
   If[MemberQ[done, i - 1], ξ = ξ // m_{st[[i]],i→st[[i]]}; st = st /. st[[i + 1]] → st[[i]]];
   If[MemberQ[done, j], ξ = ξ // m_{j,j+1→j}; st = st /. st[[j + 2]] → st[[j + 1]]];
   If[MemberQ[done, j - 1], ξ = ξ // m_{st[[j]],j→st[[j]]}; st = st /. st[[j + 1]] → st[[j]]];
   done = done ⋃ {i - 1, i, j - 1, j};
   todo = DeleteCases[todo, cx]
  ];
  CF /@ (ξ (*/. {x_0→x,y_0→y,a_0→a}*))
 ]
```

In[∘]:=
```
TimidBit[K_] := Module[{Alex = Alexander[K][T]},
  Alex^2 T^2 (v (-1 + T) (1 + T))^-1 Coefficient[Z[K][[3]], ϵ] // Factor]
```

In[∘]:=
```
TimidBit /@ AllKnots[{3, 5}]
```

Out[∘]= $\left\{-1,\ 1,\ -\frac{2 - T + 2 T^2}{T},\ -2\right\}$

In[∘]:=
```
(*Two knots with equal Alexander, Timid bit also agrees*)
Alexander[Knot[6, 1]] == Alexander[Knot[9, 46]]
TimidBit[Knot[6, 1]] == TimidBit[Knot[9, 46]]
```

Out[∘]= True

Out[∘]= True

The General Timid invariant seems equivalent to Alexander but how explicitly? At least they perform equally on the Rolfsen table.

In[∘]:=
```
AllKnots[{3, 10}] // Length
Alexander /@ AllKnots[{3, 10}] // Union // Length
{Alexander[#], TimidBit[#]} & /@ AllKnots[{3, 10}] // Union // Length
```

Out[∘]= 249

Out[∘]= 211

Out[∘]= 211