```
In[•]:= SetDirectory["C:\\Users\\T15Roland\\Wiskunde\\Bn\\HigherRank"]
        Once[<< KnotTheory`];
```

```
Out[•]= C:\Users\T15Roland\Wiskunde\Bn\HigherRank
```

ParentDirectory: The argument File should be a positive machine–sized integer, a nonempty string or a File specification.

ParentDirectory: The argument File should be a positive machine–sized integer, a nonempty string or a File specification.

ToFileName: A string or list of strings is expected at position 1 in ToFileName[{File, WikiLink, mathematica}].

ToFileName: A string or list of strings is expected at position 1 in ToFileName[{File, QuantumGroups}].

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

SetDelayed: Tag Diff in Diff[K_PD, rut_, ag_, n_, m_] is Protected.

Internal Utilities

```
In[•]:= $k = 1; γ = 1; ħ = 1;
```

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, ζ_] := MapAt[collect[#, ζ] &, sd, 3];
        collect[ℰ_, ζ_] := Collect[ℰ, ζ];
        Zip_{}[P_] := P;
        Zip_{ζs_}[Ps_List] := Zip_{ζs} /@ Ps;
        Zip_{ζ_, ζs___}[P_] :=
          (collect[P // Zip_{ζs}, ζ] /. f_. ζ^d_. :> (D_{ζ*, d}[f])) /. ζ* → 0 /.
           ((ζ* /. {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.

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

```
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[Kδz,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 $\mathbb{E}(L, Q, P) = \text{P}e^{L+Q}$. Such zips regard all of $\text{P}e^{Q}$ as a single"P". Here the z's are b and α and the ζ's are β and a.

```
LZip_ξs_List@𝔼[L_, Q_, P_] :=
  Module[{ξ, z, zs, Zs, c, ys, ηs, lt, zrule, Zrule, ξrule, Q1, EEQ, EQ},
   zs = Table[ξ*, {ξ, ξs}];
   Zs = zs /. {b → B, t → T, α → 𝔄};
   c = L /. Alternatives @@ (ξs ⋃ zs) → 0 /. Alternatives @@ Zs → 1;
   ys = Table[∂ξ (L /. Alternatives @@ zs → 0), {ξ, ξs}];
   ηs = Table[∂z (L /. Alternatives @@ ξs → 0), {z, zs}];
   lt = Inverse@Table[Kδz,ξ* - ∂z,ξ L, {ξ, ξs}, {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))];
   ξrule = Thread[ξs → ξs + ηs.lt];
   Q1 = Q /. (Zrule ⋃ ξrule);
   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_ξs[(EQ @@ zs) (P /. (Zrule ⋃ ξrule))] /.
         Derivative[ps___][EQ][___] :> EEQ[ps] /. _EQ → 1) ]];
```

```
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]];
```

𝔼[Λ]

```
In[•]:= 𝔼_dr_[Λ_] := CF@
          Module[{L, Λ0 = Limit[Λ, ϵ → 0]}, 𝔼_dr[L = Λ0 /. (η | y | ξ | x)_ → 0, Λ0 - L, e^(Λ-Λ0)]_$k /. l2U]
```

“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_}
              }] ]]
```

The Objects

Symmetric Algebra Objects

```
In[•]:= sm_{i_,j_→k_} := 𝔼_{{i,j}→{k}}[b_k (β_i + β_j) + t_k (τ_i + τ_j) + a_k (α_i + α_j) + y_k (η_i + η_j) + x_k (ξ_i + ξ_j)];
        sΔ_{i_→j_,k_} := 𝔼_{{i}→{j,k}}[β_i (b_j + b_k) + τ_i (t_j + t_k) + α_i (a_j + a_k) + η_i (y_j + y_k) + ξ_i (x_j + x_k)];
        sS_{i_} := 𝔼_{{i}→{i}}[-β_i b_i - τ_i t_i - α_i a_i - η_i y_i - ξ_i x_i];
        sϵ_{i_} := 𝔼_{{}→{i}}[0];
        sη_{i_} := 𝔼_{{i}→{}}[0];
```

```
In[•]:= sσ_{i_→j_} := 𝔼_{{i}→{j}}[β_i b_j + τ_i t_j + α_i a_j + η_i y_j + ξ_i x_j];
        sϒ_{i_→j_,k_,l_,m_} := 𝔼_{{i}→{j,k,l,m}}[β_i b_k + τ_i t_k + α_i a_l + η_i y_j + ξ_i x_m];
```

Booting Up QU

```
In[•]:= Define[aσ_{i→j} = 𝔼_{{i}→{j}}[a_j α_i + x_j ξ_i], bσ_{i→j} = 𝔼_{{i}→{j}}[b_j β_i + y_j η_i]]
```

Pairing $P : \mathbb{A}\otimes\mathbb{B} \to Q$
$\overline{R}$ is the inverse of R in the algebra $\mathbb{B}\otimes\mathbb{A}$.
NB. We use the co-algebra structure B tensor A^{cop}.

```
In[•]:= Define[P_{i,j} = 𝔼_{{i,j}→{}}[β_j α_i, η_j ξ_i, 1 + O[ϵ]^2],
          R_{i,j} = 𝔼_{{}→{i,j}}[a_j b_i, x_j y_i, 1 + O[ϵ]^2]]
```

```
(*Taken from StandardVersion.nb implementation of PG*)
Define[aΔ_{i→j,k} = 𝔼_{{i}→{j,k}}[a_j α_i + a_k α_i, x_j ξ_i + x_k ξ_i, 1 + (-a_j x_k ξ_i + 1/2 x_j x_k ξ_i^2) ϵ + O[ϵ]^2],
  am_{i,j→k} = 𝔼_{{i,j}→{k}}[a_k α_i + a_k α_j, x_k ξ_i / 𝒜_j + x_k ξ_j, 1 + O[ϵ]^2],
  aS_i = 𝔼_{{i}→{i}}[-a_i α_i, -x_i 𝒜_i ξ_i, 1 + (-a_i x_i 𝒜_i ξ_i - 1/2 x_i^2 𝒜_i^2 ξ_i^2) ϵ + O[ϵ]^2],
  a‾S_i = 𝔼_{{i}→{i}}[-a_i α_i, -x_i 𝒜_i ξ_i, 1 + (x_i 𝒜_i ξ_i - a_i x_i 𝒜_i ξ_i - 1/2 x_i^2 𝒜_i^2 ξ_i^2) ϵ + O[ϵ]^2]]
```

```
In[•]:= Define[bm_{j,k→i} = Module[{i0, i1, j1, k1}, (R_{i0,i1} // aΔ_{i1→j1,k1} // P_{j1,j} // P_{k1,k}) /. i0 → i],
  bΔ_{k→i,j} = Module[{i1, j1, k1, k2}, ((R_{i,i1} R_{j,j1} // am_{i1,j1→k1}) // P_{k1,k2}) /. k2 → k],
  bS_i = Module[{i1, i2}, (R_{i,i1} // aS_{i1} // P_{i1,i2}) /. i2 → i],
  b‾S_i = Module[{i1, i2}, (R_{i,i1} // a‾S_{i1} // P_{i1,i2}) /. i2 → i]]
```

```
In[•]:= Define[R‾_{i,j} = R_{i,j} // bS_i]
```

```
In[•]:= Define[dm_{i,j→k} = Module[{i1, i2, i3, i4}, ((sϒ_{i→i4,i4,i1,i1} // aΔ_{i1→i1,i2} // aΔ_{i2→i2,i3} // aS‾_{i3})
          (sϒ_{j→-i1,-i1,-i4,-i4} // bΔ_{-i1→-i1,-i2} // bΔ_{-i2→-i2,-i3})) // (P_{i1,-i3} P_{i3,-i1} am_{i2,-i4→k} bm_{i4,-i2→k})]]
```

```
In[•]:= dm_{i,j→k}
```

```
Out[•]=
```

$$\mathbb{E}_{\{i,j\}\to\{k\}}\left[a_k\,\alpha_i + a_k\,\alpha_j + b_k\,\beta_i + b_k\,\beta_j,\; y_k\,\eta_i + \frac{y_k\,\eta_j}{\mathcal{A}_i} + \frac{x_k\,\xi_i}{\mathcal{A}_j} + (1 - B_k)\,\eta_j\,\xi_i + x_k\,\xi_j,\right.$$
$$1 + \left(-\frac{y_k\,\beta_i\,\eta_j}{\mathcal{A}_i} + \frac{y_k^2\,\eta_i\,\eta_j}{2\,\mathcal{A}_i} - \frac{x_k\,\beta_j\,\xi_i}{\mathcal{A}_j} + a_k\,B_k\,\eta_j\,\xi_i + \frac{x_k\,y_k\,\eta_j\,\xi_i}{\mathcal{A}_i\,\mathcal{A}_j} -\right.$$
$$\left.\left.\frac{B_k\,y_k\,\eta_j^2\,\xi_i}{\mathcal{A}_i} + \frac{(1 - 3\,B_k)\,x_k\,\eta_j\,\xi_i^2}{2\,\mathcal{A}_j} + \frac{1}{2}\left(-B_k + B_k^2\right)\eta_j^2\,\xi_i^2\right)\epsilon + O[\epsilon]^2\right]$$

```
(*Usual dm tensor
```

$$\mathbb{E}_{\{i,j\}\to\{k\}}\left[a_k\;\alpha_i + a_k\;\alpha_j + b_k\;\beta_i + b_k\;\beta_j, y_k\;\eta_i + \frac{y_k\;\eta_j}{\mathcal{A}_i} + \frac{x_k\;\xi_i}{\mathcal{A}_j} + (1 - B_k)\;\eta_j\;\xi_i + x_k\;\xi_j, 1 + \left(-\frac{y_k\;\beta_i\;\eta_j}{\mathcal{A}_i} - \frac{x_k\;\beta_j\;\xi_i}{\mathcal{A}_j} + \right.\right.$$
$$\left.\left.a_k\;B_k\;\eta_j\;\xi_i + \frac{x_k\;y_k\;\eta_j\;\xi_i}{\mathcal{A}_i\;\mathcal{A}_j} + \frac{(1 - 3\;B_k)\;y_k\;\eta_j^2\;\xi_i}{2\;\mathcal{A}_i} + \frac{(1 - 3\;B_k)\;x_k\;\eta_j\;\xi_i^2}{2\;\mathcal{A}_j} + \frac{1}{4}\left(1 - 4\;B_k + 3\;B_k^2\right)\eta_j^2\;\xi_i^2\right)\epsilon + O[\epsilon]^2\right]$$

```
*)
```

associativity

```
In[•]:= (dm_{1,2→k} // dm_{k,3→k}) ≡ (dm_{2,3→k} // dm_{1,k→k})
```

```
Out[•]=
```

True

R2

```
In[•]:= R_{1,2} R‾_{3,4} // dm_{1,3→1} // dm_{2,4→2}
        R‾_{1,2} R_{3,4} // dm_{1,3→1} // dm_{2,4→2}
```

```
Out[•]=
```

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

```
Out[•]=
```

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

R3

```
In[•]:= (R_{1,2} R_{4,3} R_{5,6} // dm_{1,4→1} // dm_{2,5→5} // dm_{3,6→3}) ≡ (
          R_{1,6} R_{2,3} R_{4,5} // dm_{1,4→1} // dm_{2,5→5} // dm_{3,6→3})
```

```
Out[•]=
```

True

```
In[•]:= Define[C_i = 𝔼_{}→{i}[0, 0, B_i^{1/2} e^{-ħ ϵ a_i/2}]_$k,
         C‾_i := 𝔼_{}→{i}[0, 0, B_i^{-1/2} e^{ħ ϵ a_i/2}]_$k,
         Kink_i = Module[{i1, i2, i3}, (R_{i1,i3} C‾_{i2}) // dm_{i1,i2→i1} // dm_{i1,i3→i}],
         Kink‾_i = Module[{i1, i2, i3}, (R‾_{i1,i3} C_{i2}) // dm_{i1,i2→i1} // dm_{i1,i3→i}]]
```

```
RVK::usage =
  "RVK[xs, rots] represents a Rotational Virtual Knot with a list of n Xp/Xm crossings
    xs and a length 2n list of rotation numbers rots. Crossing
    sites are indexed 1 through 2n, and rots[[k]] is the rotation
    between site k-1 and site k. RVK is also a casting operator
    converting to the RVK presentation from other knot presentations.";
```

```
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@Replace[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}),
             _Xp | _Xm :> {}
            }), {1}],
     Cases[front, k | -k] /. {k, -k} :> --rots[[k + 1]];
    ]];
   RVK[xs, rots] ];
RVK[K_] := RVK[PD[K]];
```

```
rot[i_, 0] := 𝔼{}→{i}[0, 0, 1];
rot[i_, n_] := rot[i, n, $k];
rot[i_, n_, k_] := Module[{j},
   rot[i, n, k] = If[n > 0, rot[i, n - 1] C_j, rot[i, n + 1] C̄_j] // dm_{i,j→i}];
```

```
Width[pd_PD] :=
 Max[Length /@ FoldList[Complement[#1 ⋃ #2, #1 ⋂ #2] &, {}, List @@ List @@@ pd]]
```

```
ThinPosition[K_] := Module[{todo, done, pd, c},
   todo = List @@ PD@K; done = {}; pd = PD[];
   While[todo =!= {},
    AppendTo[pd, c = RandomChoice@MaximalBy[todo, Length[done ⋂ List @@ #] &]];
    todo = DeleteCases[todo, c];
    done = done ⋃ List @@ c];
   pd ];
ThinPosition[K_, n_] := First@MinimalBy[Table[ThinPosition[K], n], Width];
```

```
In[•]:= Z[K_] := Z[RVK@EchoFunction[Width]@ThinPosition[K, 100]];
Z[rvk_RVK] := Monitor[ Module[{ξ, done, st, c, χ, i, j, k},
    ξ = 1; done = {}; st = Range[2 Length[rvk[[1]]]]; $M = {};
    Do[AppendTo[$M, c];
     {i, j} = List @@ c;
     χ = (c /. {_Xp :> R_{i,j} Kink_0, _Xm :> R̄_{i,j} Kink_0}) // dm_{j,0→j};
     Do[χ = (rot[0, rvk[[2, k]]] χ) // dm_{0,k→k}, {k, {i, j}}];
     ξ *= χ;
     Do[
      If[MemberQ[done, k + 1], ξ = ξ // dm_{k,k+1→k}; st = st /. k + 1 → k];
      If[MemberQ[done, k - 1], ξ = ξ // dm_{st[[k-1]],k→st[[k-1]]}; st = st /. k → st[[k - 1]],
      {k, {i, j}}];
     done = done ⋃ {i, j},
     {c, rvk[[1]]}
    ];
    CF /@ (ξ /. {x_1 → x, y_1 → y, a_1 → a, B_1 → B})
   ], {Length@$M, $M}]
ρ_1[K_] := Coefficient[Alexander[K][B]^3 (-B)/(B - 1)^2 Z[K][[3]] /. {a → -1, x → 0}, ϵ] // Factor
```

Some knots

```
In[•]:= ρ_1@Knot[3, 1]
ρ_1@Mirror@Knot[3, 1]
```

Get: ParentDirectory[File] in $Path is not a string.

KnotTheory: Loading precomputed data in PD4Knots`.

» 4

Out[•]=

$$\frac{1 + B^2}{B}$$

» 4

Out[•]=

$$-\frac{1 + B^2}{B}$$

```
In[•]:= ρ_1@Knot[4, 1]
```

» 4

Out[•]=

$$0$$

```
In[•]:= ρ_1@Knot[5, 2]
```

» 4

Out[•]=

$$\frac{5 - 4\,B + 5\,B^2}{B}$$

The value of T should be:

In[•]:=
```
dm1,2→1[𝔼{}→{1,2}[0, 0, B1 (1 + ϵ a1) x2 - B2 (1 + ϵ a2) x1]]
dm1,2→1[𝔼{}→{1,2}[0, 0, B1 (1 + ϵ a1) y2 - B2 (1 + ϵ a2) y1]]
```

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

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