Pensieve header: A fresh implementation of baby DoPeGDO, the Knot Theory part.

Some Knot Theory

(Alt) In[∘]:=

```
Define[Kink_i = CC_3 R_{1,2} // m_{2,3→2} // m_{2,1→i}, \overline{Kink}_i = CC_3 \overline{R}_{1,2} // m_{1,3→1} // m_{1,2→i}]
```

(Alt) 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]];
```

(Alt) In[∘]:=

```
rot[i_, 0] := 𝔼_{{}→{i}}[1, 0, ϵSeries@0];
rot[i_, n_] := Module[{j},
   rot[i, n] = If[n > 0, rot[i, n - 1] CC_j, rot[i, n + 1] \overline{CC}_j] // m_{i,j→i}];
```

(Alt) In[∘]:=

```
Z[K_] := Z[RVK@K];
Z[rvk_RVK] := (*Z[rvk] =*)
 Module[{todo, n, rots, ζ, done, st, cx, ζ1, i, j, k, k1, k2, k3},
  {todo, rots} = List @@ rvk;
  AppendTo[rots, 0];
  n = Length[todo];
  ζ = 𝔼_{{}→{0}}[1, 0, ϵSeries@0];
  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;
   ζ1 = 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}
     ];
   ζ1 = (rot[k, rots[[i]]] ζ1) // m_{k,i→i}; rots[[i]] = 0;
   ζ1 = (ζ1 rot[k, rots[[i + 1]]]) // m_{i,k→i}; rots[[i + 1]] = 0;
   ζ1 = (rot[k, rots[[j]]] ζ1) // m_{k,j→j}; rots[[j]] = 0;
   ζ1 = (ζ1 rot[k, rots[[j + 1]]]) // m_{j,k→j}; rots[[j + 1]] = 0;
   ζ *= ζ1;
   If[MemberQ[done, i], ζ = ζ // m_{i,i+1→i}; st = st /. st[[i + 2]] → st[[i + 1]]];
   If[MemberQ[done, i - 1], ζ = ζ // m_{st[[i]],i→st[[i]]}; st = st /. st[[i + 1]] → st[[i]]];
   If[MemberQ[done, j], ζ = ζ // m_{j,j+1→j}; st = st /. st[[j + 2]] → st[[j + 1]]];
   If[MemberQ[done, j - 1], ζ = ζ // 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 /@ (ζ (*/. {x_0→x,y_0→y,a_0→a}*))
 ]
```