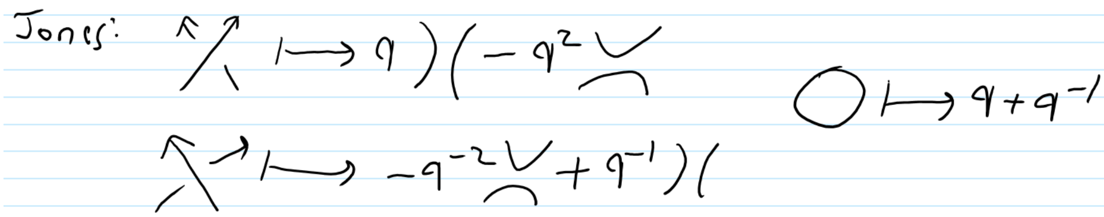
Pensieve header: Computing the Jones polynomial, further details and a faster procedure.

The Jones Polynomial of the Trefoil Knot




```
<< KnotTheory`

K = Knot[3, 1]

PD[K]

t1 = PD[K] /. x_X ⧴ If[PositiveQ[x], Xp @@ x, Xm @@ x]

t2 = Times @@ t1

t3 = t2 /. {
   Xp[i_, j_, k_, l_] ⧴ q * P[i, j] P[k, l] - q^2 * P[i, l] P[j, k],
   Xm[i_, j_, k_, l_] ⧴ -q^(-2) * P[i, j] P[k, l] + 1/q * P[i, l] P[j, k]
  }

t4 = Expand[t3]

SetAttributes[P, Orderless]

t5 = t4 //. P[a_, b_] P[b_, c_] ⧴ P[a, c]

t6 = t5 /. {
   P[i_, i_] ⧴ (q + 1/q),
   P[i_, j_]^2 ⧴ (q + 1/q)
  }

Expand[t6]

Simplify[t6]

Jones[Knot[3, 1]][q]

Simplify[t6/(q + q^-1) /. q → √q]
```

A Jones Polynomial Program

```
J[K_] := Module[{t1, t2, t3, t4, t5, t6, P},
  SetAttributes[P, Orderless];
  t1 = PD[K] /. x_X :> If[PositiveQ[x], Xp @@ x, Xm @@ x];
  t2 = Times @@ t1;
  t3 = t2 /. {
     Xp[i_, j_, k_, l_] :> q * P[i, j] P[k, l] - q^2 * P[i, l] P[j, k],
     Xm[i_, j_, k_, l_] :> -q^(-2) * P[i, j] P[k, l] + 1/q * P[i, l] P[j, k]
    };
  t4 = Expand[t3];
  t5 = t4 //. P[a_, b_] P[b_, c_] :> P[a, c];
  t6 = t5 /. {
     P[i_, i_] :> (q + 1/q),
     P[i_, j_]^2 :> (q + 1/q)
    };
  Simplify[t6/(q + q^-1) /. q → √q]
 ]
```

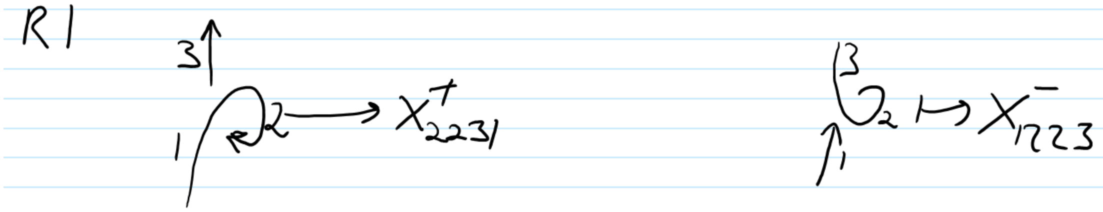
```
J[Knot[8, 17]]
```

```
Jones[Knot[8, 17]][q]
```

```
Timing[
 (K ↦ Expand[J[K]] == Jones[K][q]) /@ AllKnots[{3, 10}]
]
```

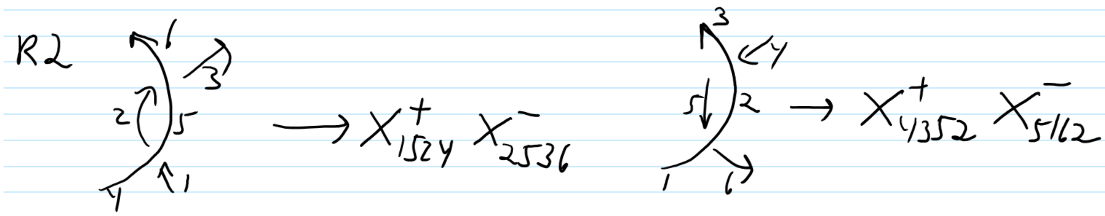
Testing the Reidemeister Moves

```
SetAttributes[P, Orderless];
TJ[t2_] := Module[{t3, t4, t5, t6},
  t3 = t2 /. {
     Xp[i_, j_, k_, l_] :> q * P[i, j] P[k, l] - q^2 * P[i, l] P[j, k],
     Xm[i_, j_, k_, l_] :> -q^(-2) * P[i, j] P[k, l] + 1/q * P[i, l] P[j, k]
    };
  t4 = Expand[t3];
  t5 = t4 //. P[a_, b_] P[b_, c_] :> P[a, c];
  t6 = t5 /. {
     P[i_, i_] :> (q + 1/q),
     P[i_, j_]^2 :> (q + 1/q)
    };
  Simplify[t6]
 ]
```

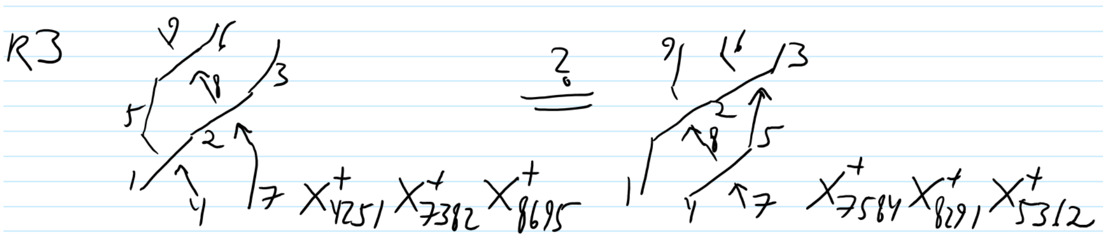



```
Xp[2, 2, 3, 1] // TJ
```

```
Xm[1, 2, 2, 3] // TJ
```



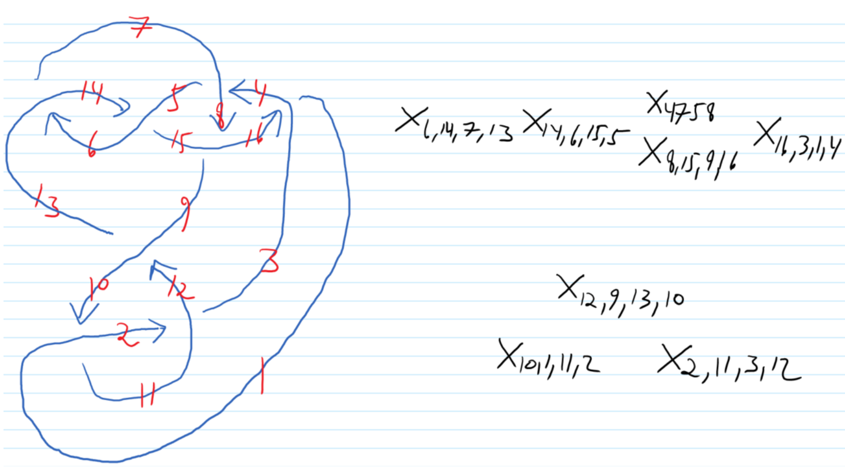

```
TJ /@ {Xp[1, 5, 2, 4] Xm[2, 5, 3, 6], Xp[4, 3, 5, 2] Xm[5, 1, 6, 2]}
```




```
TJ /@ {Xp[4, 2, 5, 1] Xp[7, 3, 8, 2] Xp[8, 6, 9, 5], Xp[7, 5, 8, 4] Xp[8, 2, 9, 1] Xp[5, 3, 6, 2]}
```

```
%[[1]] == %[[2]]
```

Analyzing a knot suggested by David Vincent




```
K1 = PD[X[6, 14, 7, 13], X[14, 6, 15, 5], X[4, 7, 5, 8], X[8, 15, 9, 16],
   X[16, 3, 1, 4], X[12, 9, 13, 10], X[10, 1, 11, 2], X[2, 11, 3, 12]];
```

```
J1 = J[K1]

Select[AllKnots[{3, 8}], J[#] == J1 &]
```

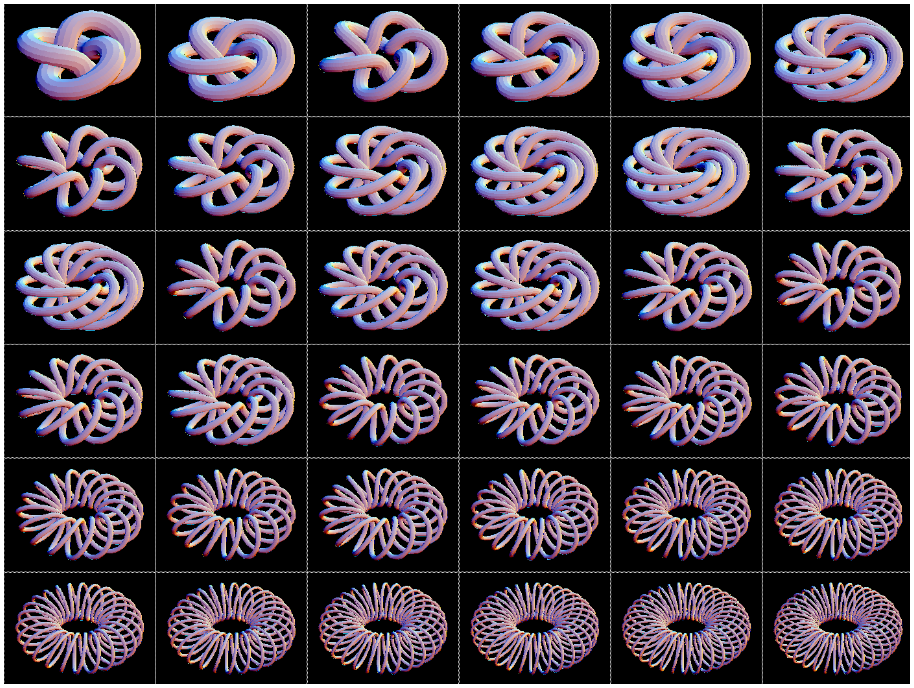
A Faster Jones Program

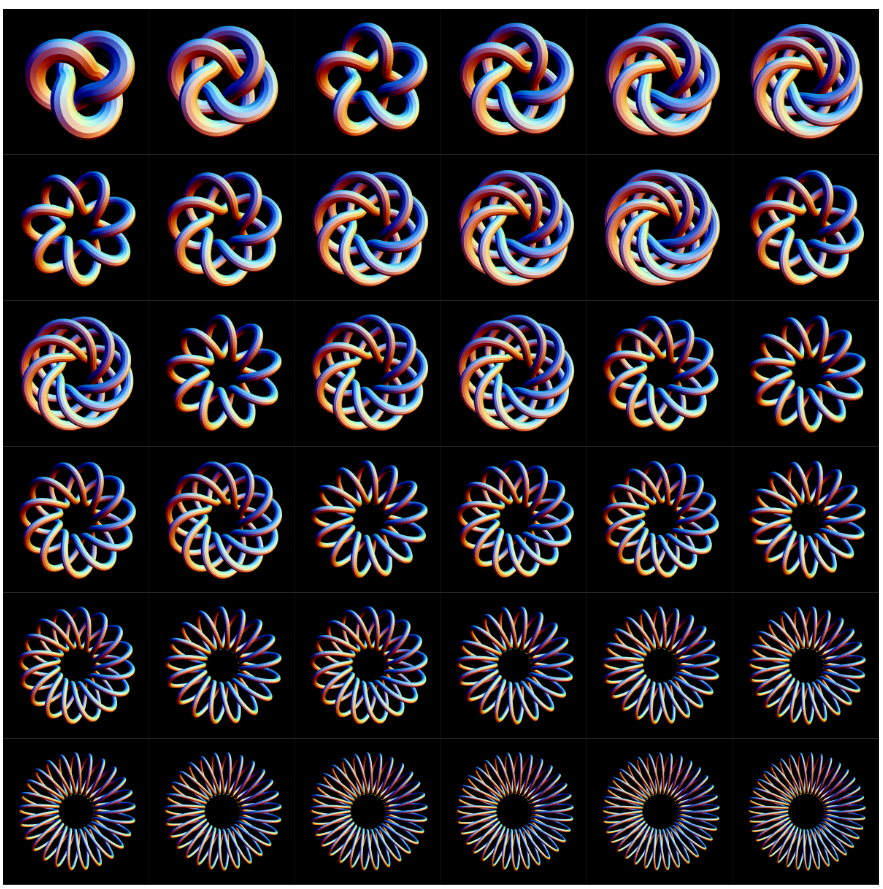
```
FJ[K_] := Module[{todo, touched = {}, x, J = 1},
  SetAttributes[P, Orderless];
  todo = List @@ PD[K] /. x_X :> If[PositiveQ[x], Xp @@ x, Xm @@ x];
  While[Length[todo] > 0,
   x = First@MaximalBy[todo, Length[touched ⋂ (List @@ #)] &];
   J *= x /. {
      Xp[i_, j_, k_, l_] :> q * P[i, j] P[k, l] - q^2 * P[i, l] P[j, k],
      Xm[i_, j_, k_, l_] :> -q^(-2) * P[i, j] P[k, l] + 1/q * P[i, l] P[j, k]
     };
   J = Expand[J] //. P[a_, b_] P[b_, c_] :> P[a, c] /. {
      P[i_, i_] :> (q + 1/q),
      P[i_, j_]^2 :> (q + 1/q)
     };
   todo = DeleteCases[todo, x];
  ];
  Simplify[J/(q + q^-1) /. q → Sqrt[q]]
 ]

FJ[Knot[3, 1]]

Timing[
 (K ⟼ Expand[FJ[K]] == Jones[K][q]) /@ AllKnots[{3, 10}]
]
```

Some Torus Knot Computations





? TorusKnots

```
TorusKnots[100]

LaunchKernels[];
DistributeDefinitions /@ {"KnotTheory`", "Global`"}

WaitAll[{
  ParallelSubmit[
   t0 = TimeUsed[];
   Do[
    {t1, J1} = Timing@J[K];
    Print["Slow Jones: ", {K, Length@PD[K], TimeUsed[] - t0, t1, J1}],
    {K, TorusKnots[100]}
   ]
  ],
  ParallelSubmit[
   t0 = TimeUsed[];
   Do[
    {t1, J1} = Timing@FJ[K];
    Print["Fast Jones: ", {K, Length@PD[K], TimeUsed[] - t0, t1, J1}],
    {K, TorusKnots[100]}
   ]
  ]
 }]
```