

pAHandout

October-03-08
9:03 AM

The Penultimate Alexander Invariant

A Definition of the MVA (From [Ar])

$$c \begin{array}{c} \nearrow b \\ \searrow a \end{array} \rightarrow \begin{array}{c} a \quad b \quad c \\ -1 \quad 1-x \quad y \end{array}$$

$$c \begin{array}{c} \nearrow b \\ \swarrow a \end{array} \rightarrow \begin{array}{c} a \quad b \quad c \\ -y \quad x-1 \quad 1 \end{array}$$

$$A = \frac{(-1)^{\frac{1}{2} \text{wt}(M_{\text{MVA}})}}{w_1(t_1)} \prod_k t_k^{\frac{\mu_1(k) - \mu_2(k)}{2}}$$



Joint with Jana Archibald

Relations by J. Murakami

$$\begin{array}{c} c \quad c \\ \diagup \quad \diagdown \end{array} \quad \begin{array}{c} c \quad c \\ \diagdown \quad \diagup \end{array} \quad \begin{array}{c} c \quad c \\ \diagup \quad \diagup \end{array} \quad \begin{array}{c} c \quad c \\ \diagdown \quad \diagdown \end{array}$$

$$L_c \quad L_c \quad L_0 \quad L_0$$

$$\begin{array}{c} c \quad d \\ \diagup \quad \diagdown \end{array} \quad \begin{array}{c} c \quad d \\ \diagdown \quad \diagup \end{array} \quad \begin{array}{c} c \quad d \\ \diagup \quad \diagup \end{array} \quad \begin{array}{c} c \quad d \\ \diagdown \quad \diagdown \end{array}$$

$$L_{1+} \quad L_{1-} \quad L_{00}$$

$$\begin{array}{c} c \quad d \quad e \\ \diagup \quad \diagdown \quad \diagup \end{array} \quad \begin{array}{c} c \quad d \quad e \\ \diagdown \quad \diagup \quad \diagup \end{array} \quad \begin{array}{c} c \quad d \quad e \\ \diagup \quad \diagup \quad \diagup \end{array} \quad \begin{array}{c} c \quad d \quad e \\ \diagdown \quad \diagdown \quad \diagdown \end{array}$$

$$L_1 \quad L_2 \quad L_3 \quad L_4$$

The Naik-Stanford Double Delta Relation (From [NS])



S. Naik T. Stanford

image not found

$$\partial_{\tau_1}(\Pi_{\tau_1}) \left(\begin{array}{c} \diagup \quad \diagdown \\ i \quad j \quad k \end{array} \right) - \partial_{\tau_2}(\Pi_{\tau_2}) \left(\begin{array}{c} \diagup \quad \diagdown \\ i \quad j \quad k \end{array} \right) + 2(h_1 - h_2)\Pi_{\tau_1}^* \left(\begin{array}{c} \diagdown \quad \diagup \\ i \quad j \quad k \end{array} \right) + 2(h_1 - h_2)\Pi_{\tau_2}^* \left(\begin{array}{c} \diagdown \quad \diagup \\ i \quad j \quad k \end{array} \right) = 0$$

(From [MH])

A Relation by H. Murakami

There's Lots More!

"God created the knots,
all else in topology
is the work of mortals"
Leopold Kronecker (paraphrased)



Visit! Edit!
http://katlas.org

Our Goal

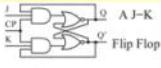
Prove all these relations uniformly, at maximal confidence and minimal brain utilization.

Circuit Algebras

- Have "circuits" with "ends",
- Can be wired arbitrarily.

May have "relations" – de-Morgan, etc.

Example $VT = CA \langle \infty, \infty \rangle / R23 = PA \langle \infty, \infty, \infty \rangle / R23, VR123, MR3$



ReminderS from lin. alg:

IF X is a set,

$\Lambda^k(X)$

$\Lambda^{k+1}(X)$

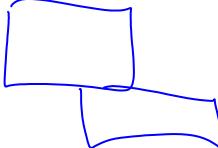
$\Lambda^{k/2}(X)$

if $Y CX^m$,

$i_Y : \Lambda^k(X) \rightarrow \Lambda^{k-m}$
is anti-symmetric in Y .

Def: Alexander half-densities
& compositions

The images of the generators



Q Can you categorify this
weaknesses:

Add:
⇒ we need an
"Alexander invariant
of arbitrary tangles,
well behaved
under tangle
composition", better,
"virtual tangles".

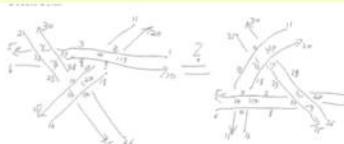
Dror Bar-Natan: Talks: Sandbjerg-0810: The Penultimate Alexander Invariant
We Mean Business

```

We Mean Business

1 (* WP : Wedge Product *)
2 WSort[expr_] := Expand[expr /. v_W :> Signature[v]*Sort[w]];
3 WP[{_, _}] := WP[{_, 0}] = 0;
4 WP[{a_, b_}] := WSORT[Distribute[a ** b] /.
5 {c1_.., w1_W} :> (c2_.. * w2_W) :> c1 c2 Join[w1, w2]];
6
7 (* IM: Interior Multiplication *)
8 IM[{_, expr_}] := expr;
9 IM[{i_, w_M}] := If[FreeQ[w, i], 0,
10 -(-i)`Position[w, i] [[{1,1}]]`DeleteCases[w, i] ];
11 IM[{im_.., i_, w_M}] := IM[{im}, IM[i, w]];
12 IM[{ims_List, expr_}] := expr /. v_W :> IM[ims, v]
13
14 (* pA on Crossings *)
15 pA[{i1_.., k1_.., j1_.., l1_..}] := AND[{t[i1]==t[k1]==t[j1]==t[l1]}, {i,1}, W[i,k], l1,
16 W[i,l1] + t[i1-1]W[j1,k1] - t[j1]W[i1,k1] + W[i,j1] + t[l1]W[j1,k1]];
17 pA[{i1_.., k1_.., j1_.., l1_..}] := AND[{t[i1]==t[k1]==t[j1]==t[l1]}, {i,j}, W[k,l1],
18 t[j1]W[i1,k1] - t[j1]W[i1,l1] + W[j,k1] + {t[i1-1]W[j1,k1] + W[k,l1]}];
19
20 (* Variable Equivalences *)
21 ReductionRules[Times[_]] = {};
22 ReductionRules[Equal[{a_.., b_..}]] := (# > a) & /@ b;
23 ReductionRules[eqq_Times] := Join @@ (ReductionRules /@ List@@eqgs)
24
25 (* AH: Alexander-Half-Densities *)
26 AHD[{eqs_, is_, os_, p_}] := AHD[eqs, is, os, Expand[-p]];
27 AHD /: Reduce[AHD[{eqs_, is_, os_, p_}], q_] :=
28 AHD[{eqs, Sort[is], WSort[q_]}, WSORT[q_]] /.
29 ReductionRules[eqq];
30 AHD /: AHD[{eqg1_, is1_, os1_, p1_}, AHD[{eqg2_, is2_, os2_, p2_}]] := Module[{glued =
31 (glued = Intersection[Union[is1, is2], List@@Union[os1, os2]]], Reduce[AHD[

32 eqg1*eqg2 //. eq1_Equal<=>eq2_Equal /;
33 Intersection[List@@eq1, List@@eq2] != {} :> Union[eq1, eq2],
34 Complement[Union[is1, is2], glued], IM[glued, WP[os1, os2]],
35 IM[glued, WP[p1, p2]]},
36 IM[glued, WP[p1, p2]]
37 ]];
38
39 (* pA on Circuit Diagrams *)
40 pA[cd_CircuitDiagram, eqgs_] := pA[cd, {}], AHD[Times[eqs], O, W[]];
41 pA[cd_CircuitDiagram, done_, ahd_AHD] := Module[{_
42 {pos = First[Ordering[Length[Complement[List @@ #, done]]] & /@ cd],
43 pa[Delete[cd, pos], Union[done, List @@ cd[[pos]]], ahd*pA[cd[[pos]]]]},
44 _];
45 AHCircuitDiagram[] := ahd AHD /. abd
```



```

class Timing:
    def __init__(self, p=100):
        self.N1 = [1] * p
        self.N2 = [1] * p
        self.N3 = [1] * p
        self.N4 = [1] * p
        self.N5 = [1] * p
        self.N6 = [1] * p
        self.N7 = [1] * p
        self.N8 = [1] * p
        self.N9 = [1] * p
        self.N10 = [1] * p
        self.N11 = [1] * p
        self.N12 = [1] * p
        self.N13 = [1] * p
        self.N14 = [1] * p
        self.N15 = [1] * p
        self.N16 = [1] * p
        self.N17 = [1] * p
        self.N18 = [1] * p
        self.N19 = [1] * p
        self.N20 = [1] * p
        self.N21 = [1] * p
        self.N22 = [1] * p
        self.N23 = [1] * p
        self.N24 = [1] * p
        self.N25 = [1] * p
        self.N26 = [1] * p
        self.N27 = [1] * p
        self.N28 = [1] * p
        self.N29 = [1] * p
        self.N30 = [1] * p
        self.N31 = [1] * p
        self.N32 = [1] * p
        self.N33 = [1] * p
        self.N34 = [1] * p
        self.N35 = [1] * p
        self.N36 = [1] * p
        self.N37 = [1] * p
        self.N38 = [1] * p
        self.N39 = [1] * p
        self.N40 = [1] * p
        self.N41 = [1] * p
        self.N42 = [1] * p
        self.N43 = [1] * p
        self.N44 = [1] * p
        self.N45 = [1] * p
        self.N46 = [1] * p
        self.N47 = [1] * p
        self.N48 = [1] * p
        self.N49 = [1] * p
        self.N50 = [1] * p
        self.N51 = [1] * p
        self.N52 = [1] * p
        self.N53 = [1] * p
        self.N54 = [1] * p
        self.N55 = [1] * p
        self.N56 = [1] * p
        self.N57 = [1] * p
        self.N58 = [1] * p
        self.N59 = [1] * p
        self.N60 = [1] * p
        self.N61 = [1] * p
        self.N62 = [1] * p
        self.N63 = [1] * p
        self.N64 = [1] * p
        self.N65 = [1] * p
        self.N66 = [1] * p
        self.N67 = [1] * p
        self.N68 = [1] * p
        self.N69 = [1] * p
        self.N70 = [1] * p
        self.N71 = [1] * p
        self.N72 = [1] * p
        self.N73 = [1] * p
        self.N74 = [1] * p
        self.N75 = [1] * p
        self.N76 = [1] * p
        self.N77 = [1] * p
        self.N78 = [1] * p
        self.N79 = [1] * p
        self.N80 = [1] * p
        self.N81 = [1] * p
        self.N82 = [1] * p
        self.N83 = [1] * p
        self.N84 = [1] * p
        self.N85 = [1] * p
        self.N86 = [1] * p
        self.N87 = [1] * p
        self.N88 = [1] * p
        self.N89 = [1] * p
        self.N90 = [1] * p
        self.N91 = [1] * p
        self.N92 = [1] * p
        self.N93 = [1] * p
        self.N94 = [1] * p
        self.N95 = [1] * p
        self.N96 = [1] * p
        self.N97 = [1] * p
        self.N98 = [1] * p
        self.N99 = [1] * p
        self.N100 = [1] * p

    def calculate_time(self, N1, N2, N3, N4, N5, N6, N7, N8, N9, N10, N11, N12, N13, N14, N15, N16, N17, N18, N19, N20, N21, N22, N23, N24, N25, N26, N27, N28, N29, N30, N31, N32, N33, N34, N35, N36, N37, N38, N39, N40, N41, N42, N43, N44, N45, N46, N47, N48, N49, N50, N51, N52, N53, N54, N55, N56, N57, N58, N59, N60, N61, N62, N63, N64, N65, N66, N67, N68, N69, N70, N71, N72, N73, N74, N75, N76, N77, N78, N79, N80, N81, N82, N83, N84, N85, N86, N87, N88, N89, N90, N91, N92, N93, N94, N95, N96, N97, N98, N99, N100):
        start_time = time.time()
        N1 = N1 + N2
        N2 = N3 + N4
        N3 = N5 + N6
        N4 = N7 + N8
        N5 = N9 + N10
        N6 = N11 + N12
        N7 = N13 + N14
        N8 = N15 + N16
        N9 = N17 + N18
        N10 = N19 + N20
        N11 = N21 + N22
        N12 = N23 + N24
        N13 = N25 + N26
        N14 = N27 + N28
        N15 = N29 + N30
        N16 = N31 + N32
        N17 = N33 + N34
        N18 = N35 + N36
        N19 = N37 + N38
        N20 = N39 + N40
        N21 = N41 + N42
        N22 = N43 + N44
        N23 = N45 + N46
        N24 = N47 + N48
        N25 = N49 + N50
        N26 = N51 + N52
        N27 = N53 + N54
        N28 = N55 + N56
        N29 = N57 + N58
        N30 = N59 + N60
        N31 = N61 + N62
        N32 = N63 + N64
        N33 = N65 + N66
        N34 = N67 + N68
        N35 = N69 + N70
        N36 = N71 + N72
        N37 = N73 + N74
        N38 = N75 + N76
        N39 = N77 + N78
        N40 = N79 + N80
        N41 = N81 + N82
        N42 = N83 + N84
        N43 = N85 + N86
        N44 = N87 + N88
        N45 = N89 + N80
        N46 = N91 + N92
        N47 = N93 + N94
        N48 = N95 + N96
        N49 = N97 + N98
        N50 = N99 + N100
        end_time = time.time()
        total_time = end_time - start_time
        return total_time

```

spur. Equal 48 (last 78 road)
stage Two

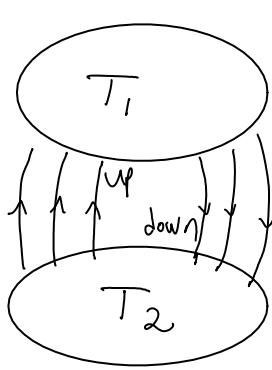
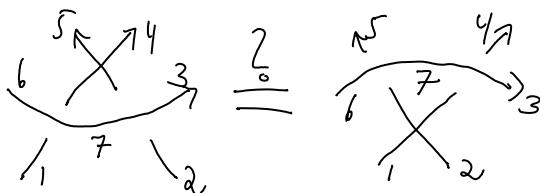
Comments online 2. $W[i_1, i_2, \dots]$ represents $i_1 \wedge i_2 \wedge \dots$. To sort it we Sort its arguments and multiply by the Signature of the permutation used. 3. The wedge product of 0 with anything is 0. 4-5. The wedge product of two things involves applying the Distributive law, joining all pairs of W's, and WSorting the result. 8. Inner multiplying by an empty list of indices does nothing. 9-10. Inner multiplying a single index yields 0 if that index is not present, otherwise it's a sign and the index is deleted. 11-12. Afterwards it's simple recursion. 15-18. For the crossings x_p and x_m it is straightforward to determine the incoming strands, the outgoing ones, and the variable equivalences. The associated half-densities are just as in the formulas. 21-23. The technicalities of imposing variable equivalences are annoying. 26. That's all we need from the definition of a tensor product. 27-28. Straightforward simplifications. 29. (The circuit algebra) product of two Alexander Half Densities: 30. The glued strands are the intersection of the ins and the outs. 32-33. Merging the variable equivalences is tricky but natural. 34-35. Removing the glued strands from the ins and outs. 36 The Key Point. The wedge product of the half-densities, inner with the glued strands. 40-45. A quick implementation of a "thin scanning" algorithm for multiple products. The key line is 42, where we select the next crossing we multiply in to be the crossing with the fewest "loose strands".

Mathematica over Xing Comments

Mathematica commutes
commute.

References

- [Ar] J. Archibald, *The Weight System of the Multivariable Alexander Polynomial*, arXiv:0710.4885.
 - [MH] H. Murakami, *A Weight System Derived from the Multivariable Conway Potential Function*, Jour. of the London Math. Soc. **59** (1999) 698–714, arXiv:math/9903108.
 - [MJ] J. Murakami, *A State Model for the Multi-Variable Alexander Polynomial*, Pac. Jour. of Math. **157-1** (1993) 109–135.
 - [NS] S. Naik and T. Stanford, *A Move on Diagrams that Generates S-Equivalence of Knots*, Jour. of Knot Theory and its Ramifications **12-5** (2003) 717–724, arXiv:math/9911005.
 - [Va] A. Vaintrob, *Melvin-Morton Conjecture and Primitive Feynman Diagrams*, Inter. J. Math. **8** (1997) 537–553, arXiv:alg/9605028.



T_1^{int}	d	u	T_2^{int}	
T_1^{int}	0	0	0	m_1 rows
down	1	1	0	l rows
up	0	0	1	l rows
T_2^{int}	0	0	1	m_2 rows