# Cheat Sheet SL2Portfolio on 180227

February 28, 2018 8:37 AM

## Cheat Sheet $sl_2$-Portfolio (an implementation of the $sl_2$ portfolio)

http://drorbn.net/AcademicPensieve/Projects/SL2Portfolio/
modified February 27, 2018.

**$\mathcal{U}_{\gamma\epsilon;\hbar}$ conventions.** "consolidate"

$q=e^{\hbar\gamma\epsilon}$, $H=\langle a,x\rangle/([a,x]=\gamma x)$ with

$$A=e^{-\hbar\epsilon a},\quad xA=qAx,\quad S_H(a,A,x)=(-a,A^{-1},-A^{-1}x),$$
$$\Delta_H(a,A,x)=(a_1+a_2,A_1A_2,x_1+A_1x_2)$$

and dual $H^*=\langle b,y\rangle/([b,y]=-\epsilon y)$ with

$$B=e^{-\hbar\gamma b},\quad By=qyB,\quad S_{H^*}(b,B,y)=(-b,B^{-1},-yB^{-1}),$$
$$\Delta_{H^*}(b,B,y)=(b_1+b_2,B_1B_2,y_1B_2+y_2).$$

Pairing by $(a,x)^*=\hbar(b,y)$ ($\Rightarrow\langle B,A\rangle=q$) making $\langle y^lb^i,a^jx^k\rangle=\delta_{ij}\delta_{kl}\hbar^{-(j+k)}j![k]_q!$ so $R=\sum\frac{\hbar^{j+k}y^kb^j\otimes a^jx^k}{j![k]_q!}$. Then $\mathcal{U}=H^{*cop}\otimes H$ with $(\phi f)(\psi g)=\langle\psi_1S^{-1}f_3\rangle\langle\psi_3,f_1\rangle(\phi\psi_2)(f_2g)$ and

$$S(y,b,a,x)=(-B^{-1}y,-b,-a,-A^{-1}x),$$
$$\Delta(y,b,a,x)=(y_1+y_2B_1,b_1+b_2,a_1+a_2,x_1+A_1x_2).$$

With the central $t:=\epsilon a-\gamma b$, $T:=e^{\hbar t}=A^{-1/2}B^{1/2}$ get

$$[a,y]=-\gamma y,\quad [b,x]=\epsilon x,\quad xy-qyx=(1-TA^2)/\hbar.$$

Cartan: $\theta(y,b,a,x)=(-B^{-1}T^{1/2}x,-b,-a,-A^{-1}T^{-1/2}y)$. (Suggesting that it may be better to redefine $y\to y'=\theta x=A^{-1}T^{-1/2}y$.)

At $\epsilon=0$, $\mathcal{U}_{\hbar;\gamma 0}=\langle b,y,a,x\rangle/([b,\cdot]=0,[a,x]=\gamma x,[a,y]=-\gamma y,[x,y]=(1-e^{-\hbar\gamma b})/\hbar)$ with $\Delta(b,y,a,x)=(b_1+b_2,y_1+e^{-\hbar\gamma b_1}y_2,a_1+a_2,x_1+x_2)$ and $\theta(y,b,a,x)=(-e^{\hbar\gamma b/2}x,-b,-a,-e^{\hbar\gamma b/2}y)$.

**Working Hypothesis.** $(\hbar,t,y,a,x)$ makes a PBW basis.

**Casimir.** $\omega=\gamma yx+\epsilon a^2-(t-\gamma\epsilon)a$, satisfies.... Roland in MixOrder.pdf: Centrals are valuable; perhaps we should write everything in $CU/QU$ as $(x\vee y)\cdot$(functions of $a$)$\cdot$(centrals).

**Scaling** with deg: $\{\gamma,\epsilon,a,b,x,y\}\to 1$, $\{\hbar\}\to -2$, $\{t\}\to 2$, $\{\omega\}\to 3$.

---

**Verification** (as in Projects/PPSA/Verification.nb).

remove

restore

qrule = q → e^{γεħ}

```
$p = 8; $k = 2;
(* $k can't be ∞ at least because of Faddeev-Quesne. *)
If[$k == 0, ϵ = 0, ϵ /: ϵ^k_ /; k > $k := 0];
(* $k=0 fails in Series[..{ϵ,...}] *)
SetAttributes[{SS, SST}, HoldAll];
TRule = {T_i_ → e^(ħ t_i), T → e^(ħ t)};
SS[ℰ_] := Block[{ħ, ϵ}, (* Shielded Series *)
   Collect[Normal@Series[ℰ, {ħ, 0, $p}], ħ, Together] ];
SST[ℰ_] :=
  Block[{ħ, ϵ},
   Collect[Normal@Series[ℰ /. TRule, {ħ, 0, $p}], ħ,
    Together] ];
Simp[ℰ_, op_] := Collect[ℰ, _CU | _QU, op];
Simp[ℰ_] :=
  Simp[ℰ, Collect[Normal@Series[#, {ħ, 0, $p}], ħ,
     Expand] &];
SimpT[ℰ_] := Collect[ℰ, _CU | _QU,
   Collect[Normal@Series[# /. TRule, {ħ, 0, $p}], ħ,
     Expand] &];
```

```
DP_{α_→Dx_, β_→Dy_}[P_][λ_] :=
 Total[CoefficientRules[P, {α, β}] /.
   ({m_, n_} → c_) :→ c D[λ, {x, m}, {y, n}]]
```

use 'qrule'

```
DeclareAlgebra[CU, Generators → {y, a, x}, Centrals → {t}];
B[a_CU, y_CU] = -γ y_CU; B[x_CU, a_CU] = -γ x_CU;
B[x_CU, y_CU] = 2 ϵ a_CU - t 1_CU;
(S@CU@y = -y_CU; S@a_CU = -a_CU; S@x_CU = -x_CU;)
S_i_[CU, Centrals] = {t_i → -t_i};
```

```
DeclareAlgebra[QU, Generators → {y, a, x},
  Centrals → {t, T}];
q = SS[e^(γ ϵ ħ)];
B[a_QU, y_QU] = -γ y_QU; B[x_QU, a_QU] = -γ QU@x;
B[x_QU, y_QU] = (q - 1) QU@{y, x} +
   𝕆_QU[{a}, SS[(1 - T e^(-2 ϵ a ħ)) / ħ]];
(S@y_QU = 𝕆_QU[{a, y}, SS[-T^-1 e^(ħ ϵ a) y]]; S@a_QU = -a_QU;
 S@x_QU = 𝕆_QU[{a, x}, SS[-e^(ħ ϵ a) x]];)
S_i_[QU, Centrals] = {t_i → -t_i, T_i → T_i^-1};
```

```
DeclareMorphism[Cθ, CU → CU, {y → -x_CU, a → -a_CU, x → -y_CU},
  {t → -t, T → T^-1}];
DeclareMorphism[Qθ, QU → QU,
  {y → 𝕆_QU[{a, x}, SS[-T^(-1/2) e^(ħ ϵ a) x]], a → -a_QU,
   x → 𝕆_QU[{a, y}, SS[-T^(-1/2) e^(ħ ϵ a) y]]}, {t → -t, T → T^-1}]
```

Can the $A\mathbb{D}$ and $S\mathbb{D}$ formulas be written so as to manifestly see their lowest term in $\epsilon$? This may allow more flexibility with $k. Or perhaps better, these should be written in implicit form and solved by power series.

$$\mathtt{A\mathbb{D}\$f} = \gamma\,\frac{\mathrm{Cosh}\left[\hbar\left(a\,e+\frac{\gamma e}{2}-\frac{t}{2}\right)\right]-\mathrm{Cosh}\left[\hbar\sqrt{\left(\frac{t-\gamma e}{2}\right)^2+e\,\omega}\right]}{\hbar\,e^{\hbar((a+\gamma)e-t/2)}\,\mathrm{Sinh}\left[\frac{\gamma e\hbar}{2}\right]\left(a^2e+a\gamma e-a\,t-\omega\right)};$$

```
A𝔻$ω = γ CU[y, x] + ϵ CU[a, a] - (t - γ ϵ) CU[a];
DeclareMorphism[A𝔻, QU → CU,
 {a → a_CU, x → CU@x,
  y → 𝕊_CU[SS[A𝔻$f] /. e → ϵ, a → a_CU, ω → A𝔻$ω] ** y_CU}]
```

$$\mathtt{S\mathbb{D}\$g} = \sqrt{\frac{2\gamma\left(\mathrm{Cosh}\left[\frac{\hbar}{2}\sqrt{t^2+\gamma^2e^2+4e\varpi}\right]-\mathrm{Cosh}\left[\frac{t-e\gamma-2ea}{2/\hbar}\right]\right)}{\mathrm{Sinh}\left[\frac{\gamma e\hbar}{2}\right]\left(t(2a+\gamma)-2a(a+\gamma)e+2\varpi\right)\hbar}};$$

```
S𝔻$f = Simplify[e^(ħ (t/2 - e a)) (S𝔻$g /. {a → -a, t → -t})];
S𝔻$ϖ = γ CU[y, x] + ϵ CU[a, a] - (t - γ ϵ) CU[a] - t γ 1_CU / 2;
DeclareMorphism[S𝔻, QU → CU, {a → a_CU,
  x → 𝕊_CU[SS[S𝔻$f] /. e → ϵ, a → a_CU, ϖ → S𝔻$ϖ] ** x_CU,
  y → 𝕊_CU[SS[S𝔻$g] /. e → ϵ, a → a_CU, ϖ → S𝔻$ϖ] ** y_CU }]
```

$$\rho@y_{CU}=\rho@y_{QU}=\begin{pmatrix}0&0\\\epsilon&0\end{pmatrix};\ \rho@a_{CU}=\rho@a_{QU}=\begin{pmatrix}\gamma&0\\0&0\end{pmatrix};$$
$$\rho@x_{CU}=\begin{pmatrix}0&\gamma\\0&0\end{pmatrix};\ \rho@x_{QU}=SS@\begin{pmatrix}0&(1-e^{-\gamma\epsilon\hbar})/(\epsilon\hbar)\\0&0\end{pmatrix};$$

```
ρ[e^ℰ_] := MatrixExp[ρ[ℰ]];
ρ[ℰ_] :=
 (ℰ /. {t → γ ϵ, T → e^(ħ γ ϵ)} /.
   (U : CU | QU)[u___] :→ Fold[Dot, {{1, 0}, {0, 1}}, ρ /@ U /@ {u}])
CU@ℭ_CU[specs___, Q_, P_] := 𝕆_CU[specs, SS[e^Q P]];
QU@ℭ_QU[specs___, Q_, P_] := 𝕆_QU[specs, SS[e^Q P]];
Λ_U_[{ξ_, α_}, {x, a}] := ℭ_U[{a, x}, α a + e^(-γ α) ξ x, 1];
Λ_U_[{α_, η_}, {a, y}] := ℭ_U[{y, a}, α a + e^(-γ α) η y, 1];
```

**Fear Not.** If $G=e^{\xi x}ye^{-\xi x}$ then $F=e^{-\eta y}e^{\xi x}e^{\eta y}e^{-\xi x}=e^{-\eta y}e^{\eta G}$ satisfies $\partial_\eta F=-yF+FG$ and $F_{\eta=0}=1$:

```
Λ𝒰_[{ξ1_, η1_}, {x, y}] :=
  Λ𝒰[{ξ1, η1}, {x, y}] =
   Module[{ξ, η, G, F, fs, f, bs, e, b, es},
    G = Simp[Table[ξ^k/k!, {k, 0, $k + 1}].
       NestList[Simp[B[x𝒰, #]] &, y𝒰, $k + 1]];
    fs = Flatten@Table[f_{l,i,j,k}[η], {l, 0, $k}, {i, 0, l},
       {j, 0, l}, {k, 0, l}];
    F = fs.(bs = fs /. f_{l_,i_,j_,k_}[η] ↦ ϵ^l 𝒰@{y^i, a^j, x^k});
    es = Flatten[Table[Coefficient[e, b] == 0,
       {e, {F - 1𝒰 /. η → 0, F ** G - y𝒰 ** F - ∂_η F}}, {b, bs}]];
    F = F /. DSolve[es, fs, η][[1]];
    𝔼𝒰[{y, a, x},
     ξ x + η y + (𝒰 /. {CU → -t η ξ, QU → η ξ (1 - T) / ℏ}),
     F /. {ϵ- → 1, 𝒰 → Times}
    ] /. {ξ → ξ1, η → η1}];
Simp[𝔼𝒰_[specs___, Q_, P_]] :=
  𝔼𝒰[specs, ExpandNumerator@Together[Q],
   Collect[P, ϵ, ExpandNumerator@*Together]];
Λ𝒰_[{υ1_, ω1_, δ_}, {u_, w_}] :=
 Simp@Module[{υ, ω, yax, q, p, Q, d},
   {yax, q, p} = List @@ Λ𝒰[{υ, ω}, {u, w}];
   𝔼𝒰[yax, Q = (υ u + ω w + δ u w + d υ ω) / (1 - d δ),
      Expand[(1 - d δ)^-1 e^-Q DP_{υ→D_u, ω→D_w}[p][e^Q]]] /.
     {d → ∂_{υ,ω} q} /. {υ → υ1, ω → ω1}]
Rord_{u_i_, w_j_ → k_}[𝔼𝒰_[L___, {l___, u_i_, w_j_, r___}_s_,
   R___, Q_, P_]] :=
 Simp@Module[{υ, ω, δ, Λ1, yax, q, p, δ1 = ∂_{u_i, w_j} Q},
   {yax, q, p} =
    List @@ If[δ1 === 0, Λ𝒰[{υ, ω}, {u, w}],
       Λ𝒰[{υ, ω, δ}, {u, w}]] /.
     {y → y_k, a → a_k, x → x_k, t → t_s, T → T_s};
   𝔼𝒰[L, {l, Sequence @@ yax, r}_s, R, q + (Q /. u_i | w_j → 0),
     e^-q DP_{u_i→D_υ, w_j→D_ω}[P][p e^q]] /.
    {υ → ∂_{u_i} Q /. w_j → 0, ω → ∂_{w_j} Q /. u_i → 0, δ → δ1}]
```

```
e_{q_,k_}[x_] := e^(Sum[(1 - q)^j x^j / (j (1 - q^j)), {j, 1, k}]); e_{q_}[x_] := e_{q,$k}[x]
QU[R_{i_,j_}] := 𝕆_QU[{y1, a1}_i, {a2, x2}_j,
   SS[e^(ℏ b1 a2) e_q[ℏ y1 x2] /. b1 → γ^-1 (ϵ a1 - t_i)]];
QU[R^-1_{i_,j_}] := S_j@QU[R_{i,j}];
𝔼_QU[R_{i_,j_}] := 𝔼_QU[{y_i, a_i, x_i}_i, {y_j, a_j, x_j}_j,
  -ℏ γ^-1 t_i a_j + ℏ y_i x_j,
  ~~Normal@~~
   ~~Series~~[e^(ℏ γ^-1 t_i a_j - ℏ y_i x_j)
     (e^(ℏ b_i a_j) e_q[ℏ y_i x_j] /. b_i → γ^-1 (ϵ a_i - t_i)), ~~{ϵ, 0, $k}~~]]
```

use e, q→1

$RS_{e\to\epsilon}$

---

**To do.** • Consider renormalizing $x$ and $y$. • Implement variable swaps. • Implement $m_{ij\to k}$. • Implement $\mathbb{E}$, $R\mathbb{E}$, and the casts CU and QU. • Reconsider the expansion of $T$ and $q$ in the hope of improving speed. • Can everything be done at $\hbar = 1$ defining a filtration by other means? That ought to be possible as the end results depend on $t/T$ and not on $\hbar$. • Bound the degrees of the logoi!

**Aside.**

```
Series[(1 - T e^(-2 e a h)) / h, {a, 0, 3}]
```

$$\frac{1-T}{h} + 2\,e\,T\,a - 2\,(e^2 h T)\,a^2 + \frac{4}{3}\,e^3 h^2 T\,a^3 + O[a]^4$$

**Program** (as in Projects/PPSA/Verification.nb).

```
Unprotect[NonCommutativeMultiply];
Attributes[NonCommutativeMultiply] = {};
(NCM = NonCommutativeMultiply)[x_] := x;
NCM[x_, y_, z__] := (x ** y) ** z;
0 ** _ = _ ** 0 = 0;
(x_Plus) ** y_ := (# ** y) & /@ x;
x_ ** (y_Plus) := (x ** #) & /@ y;
B[x_, x_] = 0; B[x_, y_] := x ** y - y ** x;
DeclareAlgebra[U_Symbol, opts__Rule] :=
 Module[{gp, sr, g, cp, CE, pow, k = 0,
   gs = Generators /. {opts}, cs = Centrals /. {opts} },
  (#_U = U@#) & /@ gs;
  gp = Alternatives @@ gs; gp = gp | gp_; (* gens *)
  sr = Flatten@Table[{g → ++k, g_i_ → {i, k}}, {g, gs}];
  (* sorting → *)
  cp = Alternatives @@ cs; (* cents *)
  CE[ℰ_] := Collect[ℰ, _U,
    (Expand[#] /. ħ^d_ /; d > $p :> 0) &];
  U_i_[ℰ_] :=
   ℰ /. {t : cp :> t_i, u_U :> Replace[u, x_ :> x_i, 1]};
  U_i_[NCM[]] = pow[ℰ_, 0] = U@{} = 1_U = U[];
  B[U@(x_)_i_, U@(y_)_i_] :=
   B[U@x_i, U@y_i] = U_i@B[U@x, U@y];
  B[U@(x_)_i_, U@(y_)_j_] /; i =!= j := 0;
  B[U@y_, U@x_] := CE[-B[U@x, U@y]];
  x_ ** 1_U := x; 1_U ** x_ := x;
  (a_. * x_U) ** (b_. * y_U) :=
   If[a b === 0, 0, CE[a b (x ** y)]];
  U[xx___, x_] ** U[y_, yy___] :=
   If[OrderedQ[{x, y} /. sr], U[xx, x, y, yy],
    U@xx ** (U@y ** U@x + B[U@x, U@y]) ** U@yy];
  U@{c_. * (l : gp)^n_, r___} /; FreeQ[c, gp] :=
   CE[c U@Table[l, {n}] ** U@{r}];
  U@{c_. * l : gp, r___} := CE[c U[l] ** U@{r}];
  U@{c_, r___} /; FreeQ[c, gp] := CE[c U@{r}];
  U@{l_Plus, r___} := CE[U@{#, r} & /@ l];
  U@{l_, r___} := U@{Expand[l], r};
  U[ℰ_NonCommutativeMultiply] := U /@ ℰ;
  𝕆_U[specs___, poly_] := Module[{sp, null, vs, us},
    sp = Replace[{specs}, l_List :> l_null, {1}];
    vs = Join @@ (First /@ sp);
    us = Join @@ (sp /. l_s_ :> (l /. x_i_ :> x_s));
    CE[Total[
       CoefficientRules[poly, vs] /. (p_ → c_) :> c U@(us^p)
      ]] /. x_null :> x];
  pow[ℰ_, n_] := pow[ℰ, n - 1] ** ℰ;
  𝕊_U[ℰ_, ss___Rule] := CE@Total[
     CoefficientRules[ℰ, First /@ {ss}] /.
      (p_ → c_) :>
       c NCM @@ MapThread[pow, {Last /@ {ss}, p}]];
  m_j_→k_[c_. * u_U] :=
   CE[(~~c /. ((#_j → #_k) & /@ cs)~~) c /. (t:cp)_j :> t_k
     DeleteCases[u, _{j|k}] ** U @@ Cases[u, w_j :> w_k] **
      U @@ Cases[u, _k]];
  S_i_[c_. * u_U] :=
   CE[(c /. S_i[U, Centrals])
     DeleteCases[u, _i] **
      U_i[NCM @@ Reverse@Cases[u, x_i :> S@U@x]]] ]
```

emancipate!

```
DeclareMorphism[m_, U_ → V_, ongs_List, oncs_List: {}] := (
  Replace[ongs, (g_ → img_) :> (m[U[g]] = img), {1}];
  m[1_U] = 1_V;
  m[U[g_i_]] := V_i[m[U@g]];
  m[U[vs__]] := NCM @@ (m /@ U /@ {vs});
  m[ℰ_] := Simp[ℰ /. oncs /. u_U :> m[u]]; )
m_j_→j_ = Identity;
m_j_→k_[ℰ_Plus] := Simp[m_j→k /@ ℰ];
m_is___,i_,j_→k_[ℰ_] := m_j→k@m_is,i→j@ℰ;
S_i_[ℰ_Plus] := Simp[S_i /@ ℰ];
```

---

**Alternative Algorithms.**

```
Simp[Œ_U_[specs___, Q_, P_]] :=
  Œ_U[specs, ExpandNumerator@Together[Q],
   Collect[P, ϵ, ExpandNumerator@*Together]];
Λ_U_[{υ1_, ω1_, δ_}, {u_, w_}] :=
 Simp@Module[{υ, ω, yax, q, p, Q, d},
   {yax, q, p} = List @@ Λ_U[{υ, ω}, {u, w}];
   Œ_U[yax, Q = (υ u + ω w + δ u w + d υ ω) / (1 - d δ),
     Expand[(1 - d δ)^-1 e^-Q DP_{υ→D_u, ω→D_w}[p][e^Q]]] /.
    {d → ∂_{υ,ω}q} /. {υ → υ1, ω → ω1}]
```

**(Proposed) Agenda.** Using Århus-like techniques, construct a map $Z\colon \mathcal{T}_{vous} \to \mathcal{A}_{vous}$, where $\mathcal{T}_{vous}$ is the space of VOUS-tangles: Virtual tangles with only Over or Under strands, some labeled as Surgery strands, with a non-singular linking matrix between the surgery strands, modulo acyclic Reidemeister 2 moves and Kirby slide relations, and where $\mathcal{A}_{vous}$ is some space of arrow diagrams modulo appropriate relations. The construction will either fix the definitions of $\mathcal{T}_{vous}$ and $\mathcal{A}_{vous}$ or will allow some flexibility that will be fixed so that the following will hold true:

1. $\mathcal{T}_{vous}$ should have a clearer topological interpretation, perhaps in terms of Heegaard diagrams.
2. $\mathcal{A}_{vous}$ should pair with some kind of Lie bialgebras.
3. $\mathcal{A}_{vous}$ should be the associated graded of $\mathcal{T}_{vous}$ and $Z$ should be an expansion.
4. Ordinary tangles $\mathcal{T}_{ord}$ and ordinary virtual tangles $\mathcal{T}_{v\text{-}ord}$ should map into $\mathcal{T}_{vous}$, and when viewed on $\mathcal{T}_{(v\text{-})ord}$, the invariant $Z$ should explain the Drinfel'd double construction.

It may be better to first construct a $Z$ and only later worry about the numbered properties. Yet property 4 has stand-alone topological content which may be very interesting: $\mathcal{T}_{vous}$ is a space with an $R3$-free presentation and which contains $\mathcal{T}_{(v\text{-})ord}$, at least nearly faithfully. What does it mean? To what extent does it make $R3$ superfluous in knot theory?
As for constructing $Z$, the first step should be a $Z\colon \mathcal{T}_{vou} \to \mathcal{A}_{vou}$ (no surgery), which would have a prescribed behaviour on strand-doubling.