CS-SL2Invariant on 180620

June 20, 2018 8:41 PM

Cheat Sheet sl_2-Invariant (the sl_2 portfolio and invariant)

http://drorbn.net/AcademicPensieve/Projects/SL2Invariant/
modified 20/6/18, 20:44

Internal Utilities

Canonical Form:

```
CF[sd_SeriesData] := MapAt[CF, sd, 3];
CF[ℰ_] :=
  PP_CF@ExpandDenominator@
    ExpandNumerator@
     Together[Expand[ℰ] //. e^x_ e^y_ :> e^(x+y) /. e^x_ :> e^CF[x]];
```

The Kronecker δ:

```
Kδ /: Kδ_{i_,j_} := If[i === j, 1, 0];
```

Equality, multiplication, and degree-adjustment of perturbed Gaussians; $\mathbb{E}[L, Q, P]$ stands for $e^{L+Q}P$:

```
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:

```
{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;
```

Finite Zips:

```
collect[sd_SeriesData, ζ_] :=
  MapAt[collect[#, ζ] &, sd, 3];
collect[ℰ_, ζ_] := PP_Collect@Collect[ℰ, ζ];
Zip_{}[P_] := P;
Zip_{ζ_,ζs___}[P_] :=
 PP_Zip[(collect[P // Zip_{ζs}, ζ] /. f_. ζ^d_. :> ∂_{ζ*,d} f) /.
   ζ* → 0]
```

QZip implements the "Q-level zips" on $\mathbb{E}(L, Q, P) = Pe^{L+Q}$. Such zips regard the L variables as scalars.

```
QZip_{ζs_List,simp_}@𝔼[L_, Q_, P_] :=
  PP_QZip@Module[{ζ, z, zs, c, ys, ηs, qt, zrule, Q1, Q2},
    zs = Table[ζ*, {ζ, ζs}];
    c = Q /. Alternatives @@ (ζs ⋃ zs) → 0;
    ys = Table[∂_ζ (Q /. Alternatives @@ zs → 0), {ζ, ζs}];
    ηs = Table[∂_z (Q /. Alternatives @@ ζs → 0), {z, zs}];
    qt = Inverse@Table[Kδ_{z,ζ*} - ∂_{z,ζ}Q, {ζ, ζs}, {z, zs}];
    zrule = Thread[zs → qt.(zs + ys)];
    Q2 = (Q1 = c + ηs.zs /. zrule) /. Alternatives @@ zs → 0;
    simp /@ 𝔼[L, Q2, Det[qt] e^-Q2 Zip_ζs[e^Q1 (P /. zrule)]]];
QZip_{ζs_List} := QZip_{ζs,CF};
```

Upper to lower and lower to Upper:

```
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};
```

LZip implements the "L-level zips" on $\mathbb{E}(L, Q, P) = Pe^{L+Q}$. Such zips regard all of Pe^Q as a single"P". Here the z's are b and α and the ζ's are β and a.

```
LZip_{ζs_List,simp_}@𝔼[L_, Q_, P_] :=
  PP_LZip@Module[{ζ, z, zs, c, ys, ηs, lt, zrule, L1, L2,
     Q1, Q2},
    zs = Table[ζ*, {ζ, ζs}];
    c = L /. Alternatives @@ (ζs ⋃ zs) → 0;
    ys = Table[∂_ζ (L /. Alternatives @@ zs → 0), {ζ, ζs}];
    ηs = Table[∂_z (L /. Alternatives @@ ζs → 0), {z, zs}];
    lt = Inverse@Table[Kδ_{z,ζ*} - ∂_{z,ζ}L, {ζ, ζs}, {z, zs}];
    zrule = Thread[zs → lt.(zs + ys)];
    L2 = (L1 = c + ηs.zs /. zrule) /. Alternatives @@ zs → 0;
    Q2 = (Q1 = Q /. U2l /. zrule) /. Alternatives @@ zs → 0;
    simp /@
     𝔼[L2, Q2, Det[lt] e^(-L2-Q2)
       Zip_ζs[e^(L1+Q1) (P /. U2l /. zrule)]] //. l2U];
LZip_{ζs_List} := LZip_{ζs,CF};
Bind_{}[L_, R_] := L R;
Bind_{is__}[L_𝔼, R_𝔼] := PP_Bind@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_{Flatten@Table[{β_{n@i}, τ_{n@i}, a_{n@i}}, {i, {is}}]} //
   QZip_{Flatten@Table[{ξ_{n@i}, y_{n@i}}, {i, {is}}]}];
B_{l_List}[L_, R_] := Bind_l[L, R];
B_{is___}[L_, R_] := Bind_{{is}}[L, R];
```

"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.

```
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,$k_Integer}, PP_{Boot@$k}@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_}
      }] ]]
```

Booting Up

```
$k = 2; (*ħ=γ=1;*)
Define[
 R_{i,j} = 𝔼[ħ a_j b_i, ħ x_j y_i, e^(Sum_{k=2}^{$k+1} ((1 - e^(γ ϵ ħ))^k (ħ y_i x_j)^k)/(k (1 - e^(k γ ϵ ħ))))]_{$k},
 P_{i,j} = 𝔼[β_i α_j / ħ, η_i ξ_j / ħ,
    1 + If[$k == 0, 0, Normal@P_{{i,j},$k-1}[[3]] -
       (R_{1,2} ~B_{1,2}~ ((P_{{1,j},0})_{$k} (P_{{i,2},$k-1})_{$k}))[[3]]]]]
Define[am_{i,j→k} = 𝔼[(α_i + α_j) a_k, (e^(-γ α_j) ξ_i + ξ_j) x_k, 1]_{$k},
 bm_{i,j→k} = 𝔼[(β_i + β_j) b_k, (η_i + η_j) y_k, e^((e^(-ϵ β_i) - 1) η_j y_k)]_{$k}]
```

Swap

$$\text{Define}\Big[aS_i = \mathbb{E}\Big[-\alpha_i\, a_j,\ -\xi_i\, x_i,\ \text{Sum}\Big[\text{Expand}\Big[\frac{e^{\xi_i x_i}\,(-\hbar\,\gamma\,\epsilon)^k}{2^k\, k!}\,\text{Nest}\Big[\text{Expand}\big[x_i^2\,\partial_{\{x_i,2\}}\#\big]\ \&,\ e^{-\xi_i\, e^{\hbar\,\epsilon\, a_i}\, x_i},\ k\Big]\Big],\ \{k,0,\$k\}\Big]\Big]_{\$k}\sim B_{i,j}\sim am_{i,j\to i},$$

$$\overline{aS}_i = \mathbb{E}\Big[-a_i\,\alpha_i,\ -x_i\,\mathcal{A}_i\,\xi_i,\ 1+\text{If}\big[\$k == 0,\ 0,\ \text{Normal@}\overline{aS}_{\{i\},\$k-1}[[3]] - \big((\overline{aS}_{\{i\},0})_{\$k}\sim B_i\sim aS_i\sim B_i\sim(\overline{aS}_{\{i\},\$k-1})_{\$k}\big)[[3]]\big]\Big]\Big]$$

$$\text{Define}\big[bS_i = R_{i,1}\sim B_1\sim aS_1\sim B_1\sim P_{i,1},$$
$$\overline{bS}_i = R_{i,1}\sim B_1\sim \overline{aS}_1\sim B_1\sim P_{i,1},$$
$$a\Delta_{i\to j,k} = (R_{1,j}\,R_{2,k})\sim B_{1,2}\sim bm_{1,2\to 3}\sim B_3\sim P_{3,i},$$
$$b\Delta_{i\to j,k} = (R_{j,1}\,R_{k,2})\sim B_{1,2}\sim am_{1,2\to 3}\sim B_3\sim P_{i,3}\big]$$

$$\text{Define}\Big[dm_{i,j\to k} = \Big(\mathbb{E}[\beta_i\, b_i + \alpha_j\, a_j,\ \eta_i\, y_i + \xi_j\, x_j,\ 1]\ \big(a\Delta_{i\to 1,2}\sim B_2\sim a\Delta_{2\to 2,3}\sim B_3\sim \overline{aS}_3\big)\ (b\Delta_{j\to -1,-2}\sim B_{-2}\sim b\Delta_{-2\to -2,-3})\Big)\sim B_{-3,-2,-1,1,2,3,i,j}\sim (P_{-1,3}\,P_{-3,1}\,am_{2,j\to k}\,bm_{i,-2\to k}),$$
$$dS_i = \mathbb{E}[\beta_i\, b_1 + \alpha_i\, a_2,\ \eta_i\, y_1 + \xi_i\, x_2,\ 1]\sim B_{1,2}\sim\big(\overline{bS}_1\, aS_2\big)\sim B_{1,2}\sim dm_{2,1\to i},$$
$$d\Delta_{i\to j,k} = (b\Delta_{i\to 3,1}\, a\Delta_{i\to 2,4})\sim B_{1,2,3,4}\sim(dm_{3,4\to k}\, dm_{1,2\to j})\Big]$$

$$\text{Define}\Big[\overline{R}_{i,j} = \text{Expand}\ /@\ R_{i,j}\sim B_j\sim dS_j,$$
$$CC_i = \mathbb{E}\big[0,0,B_i^{1/2}\, e^{-\hbar\,\epsilon\, a_i/2}\big]_{\$k},$$
$$\overline{CC}_i = \mathbb{E}\big[0,0,B_i^{-1/2}\, e^{\hbar\,\epsilon\, a_i/2}\big]_{\$k},$$
$$\text{Kink}_i = \big(R_{1,3}\,\overline{CC}_2\big)\sim B_{1,2}\sim dm_{1,2\to 1}\sim B_{1,3}\sim dm_{1,3\to i},$$
$$\overline{\text{Kink}}_i = \big(\overline{R}_{1,3}\, CC_2\big)\sim B_{1,2}\sim dm_{1,2\to 1}\sim B_{1,3}\sim dm_{1,3\to i}\Big]$$

Note. $t == \epsilon a - \gamma b$ and $b == -t/\gamma + \epsilon a/\gamma$.

$$\text{Define}\Big[b2t_i = \mathbb{E}\big[\alpha_i\, a_i - \beta_i\, t_i/\gamma,\ \xi_i\, x_i + \eta_i\, y_i,\ e^{\epsilon\,\beta_i\, a_i/\gamma}\big]_{\$k},$$
$$t2b_i = \mathbb{E}[\alpha_i\, a_j - \tau_i\,\gamma\, b_j,\ \xi_i\, x_j + \eta_i\, y_j,\ e^{\epsilon\,\tau_i\, a_j}]_{\$k}\Big]$$

$$\text{Timing@Block}\Big[\{\$k = 1\},\ Z = R_{1,5}\,R_{6,2}\,R_{3,7}\,\overline{CC}_4\,\overline{\text{Kink}}_8\,\overline{\text{Kink}}_9\,\overline{\text{Kink}}_{10};\ \text{Do}[Z = Z\sim B_{1,r}\sim dm_{1,r\to 1},\ \{r,2,10\}];\ \text{Simplify}\ /@\ Z\Big]$$

$$\Big\{10.1563,\ \mathbb{E}\Big[0,0,\frac{B_1}{1-B_1+B_1^2} - \frac{1}{(1-B_1+B_1^2)^3}\hbar\, B_1\big(-a_1(-1+B_1-B_1^3+B_1^4) + \gamma\,(B_1 - 2B_1^2 - 2B_1^4 + 2\hbar\, x_1\, y_1 + B_1^3\,(3+2\hbar\, x_1\, y_1))\big)\,\epsilon + O[\epsilon]^2\Big]\Big\}$$

```
PrintProfile[]
```

```
ProfileRoot is root. Profiled time: 136.391
  ( 136)   0.983/ 108.520 above Bind
  ( 126)   0.016/   0.016 above CF
  (  12)       0/   2.688 above Boot[1]
  (  16)   0.079/   6.796 above Boot[2]
  (   5)   0.062/  18.375 above Boot[3]
CF: called 29902 times, time in 73.435/80.232
  ( 28576)   6.797/   6.797 under CF
  (  600)  52.963/  59.760 under LZip
  (  126)   0.016/   0.016 under ProfileRoot
  (  600)  13.659/  13.659 under QZip
  ( 28576)   6.797/   6.797 above CF
Zip: called 1705 times, time in 33.19/140.664
  (  200)   6.192/  27.122 under LZip
  (  200)   2.607/  12.749 under QZip
  ( 1305)  24.391/ 100.790 under Zip
  ( 1705)   6.681/   6.681 above Collect
  ( 1305)  24.391/ 100.790 above Zip
LZip: called 200 times, time in 17.645/104.527
  (  200)  17.645/ 104.530 under Bind
  (  600)  52.963/  59.760 above CF
  (  200)   6.192/  27.122 above Zip
Collect: called 1705 times, time in 6.681/6.681
  ( 1705)   6.681/   6.681 under Zip
QZip: called 200 times, time in 4.016/30.424
  (  200)   4.016/  30.424 under Bind
  (  600)  13.659/  13.659 above CF
  (  200)   2.607/  12.749 above Zip
Bind: called 200 times, time in 1.173/136.124
  ( 136)   0.983/ 108.520 under ProfileRoot
  (  24)   0.032/   2.657 under Boot[1]
  (  24)   0.032/   6.717 under Boot[2]
  (  16)   0.126/  18.234 under Boot[3]
  ( 200)  17.645/ 104.530 above LZip
  ( 200)   4.016/  30.424 above QZip
Boot[3]: called 11 times, time in 0.141/30.643
  (   5)   0.062/  18.375 under ProfileRoot
  (   6)   0.079/  12.268 under Boot[3]
  (  16)   0.126/  18.234 above Bind
  (   6)   0.079/  12.268 above Boot[3]
Boot[2]: called 18 times, time in 0.079/6.843
  (  16)   0.079/   6.796 under ProfileRoot
  (   2)       0/   0.047 under Boot[2]
  (  24)   0.032/   6.717 above Bind
  (   2)       0/   0.047 above Boot[2]
Boot[1]: called 20 times, time in 0.031/3.719
  (  12)       0/   2.688 under ProfileRoot
  (   8)   0.031/   1.031 under Boot[1]
  (  24)   0.032/   2.657 above Bind
  (   2)       0/       0 above Boot[0]
  (   8)   0.031/   1.031 above Boot[1]
Boot[0]: called 2 times, time in 0./0.
  (   2)       0/       0 under Boot[1]
```