Cheat Sheet sl_2-Portfolio-2

(the full sl_2 portfolio)

http://drorbn.net/AcademicPensieve/Projects/SL2Portfolio2/
modified 17/6/19, 20:16

Objects. All are of the form $e^{L+Q}P$, where

- L is a quadratic of the form $\sum l_{z\zeta}z\zeta$, where z runs over $\{t_i,\alpha_i\}_{i\in S}$ and ζ over $\{\tau_i,a_i\}_{i\in S}$, with integer coefficients $l_{z\zeta}$.
- Q is a quadratic of the form $\sum q_{z\zeta}z\zeta$, where z runs over $\{x_i,\eta_i\}_{i\in S}$ and ζ over $\{\xi_i,y_i\}_{i\in S}$, with coefficients $q_{z\zeta}$ in the ring R_S of rational functions in $\{T_i,\mathcal{A}_i\}_{i\in S}$.
- $P=\sum\epsilon^k P_k$ is docile ($\deg P_k\le 4k$) in $\{y_i,a_i,x_i,\eta_i,\xi_i\}_{i\in S}$ with coefficients in R_S, and where $\deg(y_i,a_i,x_i,\eta_i,\xi_i)=(1,2,1,1,1)$.

In `QuarksAndDegrees.m`**:** Gradings to remove $\hbar$ and γ.

Q. What becomes of the classical-level automorphism $(y,b,\epsilon)\to(-y,-b,-\epsilon)$?

The "Speedy" Engine

Internal Utilities

Canonical Form:

```
CCF[ℰ_] :=
  PPCCF@ExpandDenominator@
    ExpandNumerator@PPTogether@Together[PPExp[
        Expand[ℰ] //. e^x_ e^y_ :> e^(x+y) /. e^x_ :> e^CCF[x]]];
CF[ℰ_List] := CF /@ ℰ;
CF[sd_SeriesData] := MapAt[CF, sd, 3];
CF[ℰ_] := PPCF@Module[
    {vs = Cases[ℰ, (y | b | t | a | x | η | β | τ | α | ξ)_, ∞] ⋃
       {y, b, t, a, x, η, β, τ, α, ξ}},
    Total[CoefficientRules[Expand[ℰ], vs] /.
      (ps_ → c_) :> CCF[c] (Times @@ vs^ps)]
  ];
CF[ℰ_𝔼] := CF /@ ℰ;
CF[𝔼sp___[ℰs___]] := CF /@ 𝔼sp[ℰs];
```

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;
```

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)};
12U = {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};
```

Derivatives in the presence of exponentiated variables:

```
Db[f_] := ∂b f - ħ γ B ∂B f;  Db_i_[f_] := ∂b_i f - ħ γ B_i ∂B_i f;
Dt[f_] := ∂t f + ħ T ∂T f;  Dt_i_[f_] := ∂t_i f + ħ T_i ∂T_i f;
Dα[f_] := ∂α f + γ 𝒜 ∂𝒜 f;  Dα_i_[f_] := ∂α_i f + γ 𝒜_i ∂𝒜_i f;
Dv_[f_] := ∂v f;  D{v_,0}[f_] := f;  D{}[f_] := f;
D{v_,n_Integer}[f_] := Dv[D{v,n-1}[f]];
D{l_List,ls___}[f_] := D{ls}[Dl[f]];
```

Finite Zips:

```
collect[sd_SeriesData, ζ_] :=
  MapAt[collect[#, ζ] &, sd, 3];
collect[ℰ_, ζ_] := PPCollect@Collect[ℰ, ζ];
Zip{}[P_] := P;
Zipζs_[Ps_List] := Zipζs /@ Ps;
Zip{ζ_,ζs___}[P_] := PPZip[
  (collect[P // Zip{ζs}, ζ] /. f_. ζ^d_. :> (D{ζ*,d}[f])) /.
    ζ* → 0 /. ((ζ* /. {b → B, t → T, α → 𝒜}) → 1)]
```

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

```
QZipζs_List@𝔼[L_, Q_, P_] :=
  PPQZip@Module[{ζ, z, zs, c, ys, ηs, qt, zrule, ζrule, out},
    zs = Table[ζ*, {ζ, ζs}];
    c = CF[Q /. Alternatives @@ (ζs ⋃ zs) → 0];
    ys = CF@Table[∂ζ (Q /. Alternatives @@ zs → 0),
       {ζ, ζs}];
    ηs = CF@Table[∂z (Q /. Alternatives @@ ζs → 0), {z, zs}];
    qt = CF@Inverse@Table[Kδz,ζ* - ∂z,ζ Q, {ζ, ζs}, {z, zs}];
    zrule = Thread[zs → CF[qt.(zs + ys)]];
    ζrule = Thread[ζs → ζs + ηs.qt];
    CF /@ 𝔼[L, c + ηs.qt.ys,
      Det[qt] Zipζs[P /. (zrule ⋃ ζrule)]] ];
```

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.

```
LZipζs_List@𝔼[L_, Q_, P_] :=
  PPLZip@Module[{ζ, z, zs, Zs, c, ys, ηs, lt, zrule,
     Zrule, ζrule, Q1, EEQ, EQ},
    zs = Table[ζ*, {ζ, ζs}];
    Zs = zs /. {b → B, t → T, α → 𝒜};
    c = L /. Alternatives @@ (ζs ⋃ zs) → 0 /.
      Alternatives @@ Zs → 1;
    ys = Table[∂ζ (L /. Alternatives @@ zs → 0), {ζ, ζs}];
    ηs = Table[∂z (L /. Alternatives @@ ζs → 0), {z, zs}];
    lt = Inverse@Table[Kδz,ζ* - ∂z,ζ L, {ζ, ζs}, {z, zs}];
    zrule = Thread[zs → lt.(zs + ys)];
    Zrule = Join[zrule,
      zrule /.
       r_Rule :> ((U = r[[1]] /. {b → B, t → T, α → 𝒜}) →
          (U /. U2l /. r //. 12U))];
    ζrule = Thread[ζs → ζs + ηs.lt];
    Q1 = Q /. (Zrule ⋃ ζrule);
    EEQ[ps___] :=
     EEQ[ps] =
      PP"EEQ"@(CF[e^-Q1 DThread[{zs,{ps}}][e^Q1]] /.
          {Alternatives @@ zs → 0, Alternatives @@ Zs → 1});
    CF@𝔼[c + ηs.lt.ys,
      Q1 /. {Alternatives @@ zs → 0, Alternatives @@ Zs → 1},
      Det[lt]
       (Zipζs[(EQ @@ zs) (P /. (Zrule ⋃ ζrule))] /.
          Derivative[ps___][EQ][___] :> EEQ[ps] /.
         _EQ → 1) ]];
```

```
B{}[L_, R_] := L R;
B{is__}[L_, 𝔼_, R_, 𝔼_] := PPB@Module[{n},
    Times[
       L /. Table[(v : b | B | t | T | a | x | y)i → vn@i,
         {i, {is}}],
       R /. Table[(v : β | τ | α | 𝔄 | ξ | η)i → vn@i, {i, {is}}]
      ] // LZipJoin@@Table[{βn@i, τn@i, an@i}, {i, {is}}] //
     QZipJoin@@Table[{ξn@i, yn@i}, {i, {is}}] ];
Bis___[L_, R_] := B{is}[L, R];
```

𝔼 morphisms with domain and range.

```
Bis_List[𝔼d1_→r1_[L1_, Q1_, P1_], 𝔼d2_→r2_[L2_, Q2_, P2_]] :=
  𝔼(d1⋃Complement[d2,is])→(r2⋃Complement[r1,is]) @@
    Bis[𝔼[L1, Q1, P1], 𝔼[L2, Q2, P2]];
𝔼d1_→r1_[L1_, Q1_, P1_] // 𝔼d2_→r2_[L2_, Q2_, P2_] :=
  Br1⋂d2[𝔼d1→r1[L1, Q1, P1], 𝔼d2→r2[L2, Q2, P2]];
𝔼d1_→r1_[L1_, Q1_, P1_] ≡ 𝔼d2_→r2_[L2_, Q2_, P2_] ^:=
  (d1 == d2) ∧ (r1 == r2) ∧ (𝔼[L1, Q1, P1] ≡ 𝔼[L2, Q2, P2]);
𝔼d1_→r1_[L1_, Q1_, P1_] 𝔼d2_→r2_[L2_, Q2_, P2_] ^:=
  𝔼(d1⋃d2)→(r1⋃r2) @@ (𝔼[L1, Q1, P1] × 𝔼[L2, Q2, P2]);
𝔼dr_[L_, Q_, P_]$k_ := 𝔼dr @@ 𝔼[L, Q, P]$k;
𝔼_[ℰ___][i_] := {ℰ}[[i]];
```

𝔼[Λ]

```
𝔼dr_[Λ_] :=
 CF@Module[{L, Λ0 = Limit[Λ, ϵ → 0]},
   𝔼dr[L = Λ0 /. (η | y | ξ | x)_ → 0, Λ0 - L, e^(Λ-Λ0)]$k /. 12U]
```

Exponentials as needed.

Task. Define $\text{Exp}_{m,i,k}[P]$ to compute $e^{\mathbb{O}(P)}$ to ϵ^k in the using the $m_{i,i\to i}$ multiplication, where P is an ϵ-dependent near-docile element, giving the answer in $\mathbb{E}$-form.

Methodology. If $P_0 := P_{\epsilon=0}$ and $e^{\lambda\,\mathbb{O}(P)} = \mathbb{O}\left(e^{\lambda P_0}\,F(\lambda)\right)$, then $F(\lambda = 0) = 1$ and we have:

$$\mathbb{O}\left(e^{\lambda P_0}(P_0\,F(\lambda)+\partial_\lambda F)\right) = \mathbb{O}\left(\partial_\lambda e^{\lambda P_0}\,F(\lambda)\right) =$$

$$\partial_\lambda \mathbb{O}\left(e^{\lambda P_0}\,F(\lambda)\right) = \partial_\lambda e^{\lambda\,\mathbb{O}(P)} = e^{\lambda\,\mathbb{O}(P)}\,\mathbb{O}(P) = \mathbb{O}\left(e^{\lambda P_0}\,F(\lambda)\right)\mathbb{O}(P)$$

This is a linear ODE for F. Setting inductively $F_k = F_{k-1} + \epsilon^k\,\varphi$ we find that $F_0 = 1$ and solve for φ.

```
(* Bug: The first line is valid only if 𝕆(e^P0) == e^𝕆(P0). *)
Expm_,i_,0[P_] := Module[{LQ = Normal@P /. ϵ → 0},
   𝔼[LQ /. (x | y)i → 0, LQ /. (b | a | t)i → 0, 1] ];
Expm_,i_,k_[P_] := Block[{$k = k},
  Module[{P0, λ, φ, φs, F, j, rhs, eqn, pows, at0, atλ},
   P0 = Normal@P /. ϵ → 0;
   F = Normal@Last@Expm,i,k-1[λ P];
   While[
    rhs =
     mi,j→i[
        𝔼{}→{i}[λ P0 /. (x | y)i → 0, λ P0 /. (b | a | t)i → 0,
          F]k sσi→j@𝔼{}→{i}[0, 0, P]k] // Last // Normal;
    eqn = CF[(∂λF) + P0 F - rhs];
    eqn =!= 0, (*do*)
    pows = First /@ CoefficientRules[eqn, {yi, bi, ai, xi}];
    F += Sum[ϵ^k φjs[λ] Times @@ {yi, bi, ai, xi}^js,
      {js, pows}];
    rhs =
     mi,j→i[
        𝔼{}→{i}[λ P0 /. (x | y)i → 0, λ P0 /. (b | a | t)i → 0,
          F]k sσi→j@𝔼{}→{i}[0, 0, P]k] // Last // Normal;
    eqn = CF[(∂λF) + P0 F - rhs];
    φs = Table[φjs[λ], {js, pows}];
    at0 = Table[φjs[0] == 0, {js, pows}];
    atλ = (# == 0) & /@
      (pows /. CoefficientRules[eqn, {yi, bi, ai, xi}]);
    F = F /. DSolve[And @@ (at0 ⋃ atλ), φs, λ][[1]]
   ];
   𝔼{}→{i}[P0 /. (x | y)i → 0, P0 /. (b | a | t)i → 0,
    F + O[ϵ]^(k+1) /. λ → 1]] ]
```

"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[opnisp,$k_Integer, PPBoot@Block[{i, j, k}, opisp,$k = ℰ;
         opnis,$k]];
      SD[opisp, op{is},$k]; SD[opsis__, 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_}
     }] ]]
```

The Objects

Symmetric Algebra Objects

```
smi_,j_→k_ :=
  𝔼{i,j}→{k}[bk (βi + βj) + tk (τi + τj) + ak (αi + αj) +
    yk (ηi + ηj) + xk (ξi + ξj)];
sΔi_→j_,k_ :=
  𝔼{i}→{j,k}[βi (bj + bk) + τi (tj + tk) + αi (aj + ak) +
    ηi (yj + yk) + ξi (xj + xk)];
sSi_ := 𝔼{i}→{i}[-βi bi - τi ti - αi ai - ηi yi - ξi xi];
sϵi_ := 𝔼{}→{i}[0];
sηi_ := 𝔼{i}→{}[0];
```

```
sσ_{i_→j_} := 𝔼_{i}→{j} [β_i b_j + τ_i t_j + α_i a_j + η_i y_j + ξ_i x_j];
sϒ_{i_→j_,k_,l_,m_} := 𝔼_{i}→{j,k,l,m} [β_i b_k + τ_i t_k + α_i a_l + η_i y_j + ξ_i x_m];
```

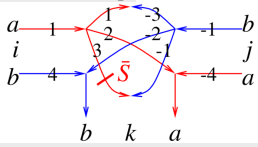
The CU Definitions

```
cΛ = (η_i + (e^{-γ α_i - ϵ β_i} η_j)/(1 + γ ϵ η_j ξ_i)) y_k + (β_i + β_j + Log[1 + γ ϵ η_j ξ_i]/ϵ) b_k +
     (α_i + α_j + Log[1 + γ ϵ η_j ξ_i]/γ) a_k + ((e^{-γ α_j - ϵ β_j} ξ_i)/(1 + γ ϵ η_j ξ_i) + ξ_j) x_k;
Define[cm_{i,j→k} = 𝔼_{i,j}→{k} [cΛ]]
Define[cσ_{i→j} = sσ_{i,j} /. τ_i → 0, cϵ_i = sϵ_i, cη_i = sη_i,
  cΔ_{i→j,k} = sΔ_{i→j,k},
  cS_i = sS_i // sϒ_{i→1,2,3,4} // cm_{4,3→i} // cm_{i,2→i} // cm_{i,1→i}];
```

Booting Up QU

```
Define[aσ_{i→j} = 𝔼_{i}→{j} [a_j α_i + x_j ξ_i],
 bσ_{i→j} = 𝔼_{i}→{j} [b_j β_i + y_j η_i]]
Define[am_{i,j→k} = 𝔼_{i,j}→{k} [(α_i + α_j) a_k + (𝒜_j^{-1} ξ_i + ξ_j) x_k],
 bm_{i,j→k} = 𝔼_{i,j}→{k} [(β_i + β_j) b_k + (η_i + e^{-ϵ β_i} η_j) y_k]]
Define[R_{i,j} = 𝔼_{}→{i,j} [ħ a_j b_i + Sum_{k=1}^{$k+1} (1 - e^{γ ϵ ħ})^k (ħ y_i x_j)^k / (k (1 - e^{k γ ϵ ħ}))],
 R̄_{i,j} = CF@𝔼_{}→{i,j} [-ħ a_j b_i, -ħ x_j y_i / B_i,
    1 + If[$k == 0, 0, (R̄_{{i,j},$k-1})_$k [3] -
       ((R̄_{{i,j},0})_$k R_{1,2} (R̄_{{3,4},$k-1})_$k) // (bm_{i,1→i} am_{j,2→j}) //
         (bm_{i,3→i} am_{j,4→j})) [3]]],
 P_{i,j} = 𝔼_{i,j}→{} [β_i α_j / ħ, η_i ξ_j / ħ,
   1 + If[$k == 0, 0, (P_{{i,j},$k-1})_$k [3] -
      (R_{1,2} // ((P_{{1,j},0})_$k (P_{{i,2},$k-1})_$k)) [3]]]
Define[aS_i = (aσ_{i→2} R̄_{1,i}) // P_{1,2},
 āS_i = 𝔼_{i}→{i} [-a_i α_i, -x_i 𝒜_i ξ_i,
   1 + If[$k == 0, 0, (āS_{{i},$k-1})_$k [3] -
      ((āS_{{i},0})_$k // aS_i // (āS_{{i},$k-1})_$k) [3]]]]
Define[bS_i = bσ_{i→1} R_{i,2} // aS_2 // P_{1,2},
 b̄S_i = bσ_{i→1} R_{i,2} // āS_2 // P_{1,2},
 aΔ_{i→j,k} = (R_{1,j} R_{2,k}) // bm_{1,2→3} // P_{3,i},
 bΔ_{i→j,k} = (R_{j,1} R_{k,2}) // am_{1,2→3} // P_{i,3}]
```

The Drinfel'd double:




```
Define[
 dm_{i,j→k} =
  ((sϒ_{i→4,4,1,1} // aΔ_{1→1,2} // aΔ_{2→2,3} // āS_3)
     (sϒ_{j→-1,-1,-4,-4} // bΔ_{-1→-1,-2} // bΔ_{-2→-2,-3})) //
   (P_{-1,3} P_{-3,1} am_{2,-4→k} bm_{4,-2→k})]
Define[dσ_{i→j} = aσ_{i→j} bσ_{i→j},
 dϵ_i = sϵ_i, dη_i = sη_i,
 dS_i = sϒ_{i→1,1,2,2} // (b̄S_1 aS_2) // dm_{2,1→i},
 d̄S_i = sϒ_{i→1,1,2,2} // (bS_1 āS_2) // dm_{2,1→i},
 dΔ_{i→j,k} = (bΔ_{i→3,1} aΔ_{i→2,4}) // (dm_{3,4→k} dm_{1,2→j})]
Define[C_i = 𝔼_{}→{i} [0, 0, B_i^{1/2} e^{-ħ ϵ a_i/2}]_$k,
 C̄_i = 𝔼_{}→{i} [0, 0, B_i^{-1/2} e^{ħ ϵ a_i/2}]_$k,
 Kink_i = (R_{1,3} C̄_2) // dm_{1,2→1} // dm_{1,3→i},
 K̄ink_i = (R̄_{1,3} C_2) // dm_{1,2→1} // dm_{1,3→i}]
```

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

```
Define[b2t_i = 𝔼_{i}→{i} [α_i a_i + β_i (ϵ a_i - t_i)/γ + ξ_i x_i + η_i y_i],
 t2b_i = 𝔼_{i}→{i} [α_i a_i + τ_i (ϵ a_i - γ b_i) + ξ_i x_i + η_i y_i]]
```

The Knot Tensors

```
Define[kR_{i,j} = R_{i,j} // (b2t_i b2t_j) /. t_{i|j} → t,
 k̄R_{i,j} = R̄_{i,j} // (b2t_i b2t_j) /. {t_{i|j} → t, T_{i|j} → T},
 km_{i,j→k} = (t2b_i t2b_j) // dm_{i,j→k} //
   b2t_k /. {t_k → t, T_k → T, τ_{i|j} → 0},
 kC_i = C_i // b2t_i /. T_i → T,
 k̄C_i = C̄_i // b2t_i /. T_i → T,
 kKink_i = Kink_i // b2t_i /. {t_i → t, T_i → T},
 k̄Kink_i = K̄ink_i // b2t_i /. {t_i → t, T_i → T}]
```

Benchmarking in QU

```
PrintProfile[]
```

```
ProfileRoot is root. Profiled time: 69.421
  (     1)   0.016/ 34.828 above Z
  (   157)   0.436/ 27.406 above B
  (    37)   0.092/  7.092 above Boot
  (   147)   0.032/  0.095 above CF
  (     1)      0/     0 above RVK
CF: called 12203 times, time in 23.25/54.255
  (  1047)   0.560/  0.998 under EEQ
  (     4)   0.016/  0.032 under Z
  (    47)   0.031/  0.063 under Boot
  (  1347)   6.298/ 17.397 under LZip
  (   147)   0.032/  0.095 under ProfileRoot
  (  9611)  16.313/ 35.670 under QZip
  ( 36119)   9.796/ 31.005 above CCF
Together: called 36119 times, time in 16.351/21.209
  ( 36119)  16.351/ 21.209 under CCF
  ( 36119)   4.858/  4.858 above Exp
CCF: called 36119 times, time in 9.796/31.005
  ( 36119)   9.796/ 31.005 under CF
  ( 36119)  16.351/ 21.209 above Together
Zip: called 2675 times, time in 8.631/37.554
  (   294)   0.878/  5.924 under LZip
  (   294)   0.890/  4.392 under QZip
  (  2087)   6.863/ 27.238 under Zip
  (  2675)   1.685/  1.685 above Collect
  (  2087)   6.863/ 27.238 above Zip
Exp: called 36119 times, time in 4.858/4.858
  ( 36119)   4.858/  4.858 under Together
LZip: called 294 times, time in 2.187/26.878
  (   294)   2.187/ 26.878 under B
  (  1047)   0.372/  1.370 above EEQ
  (  1347)   6.298/ 17.397 above CF
  (   294)   0.878/  5.924 above Zip
Collect: called 2675 times, time in 1.685/1.685
  (  2675)   1.685/  1.685 under Zip
QZip: called 294 times, time in 1.311/41.373
  (   294)   1.311/ 41.373 under B
  (  9611)  16.313/ 35.670 above CF
  (   294)   0.890/  4.392 above Zip
B: called 294 times, time in 0.671/68.922
  (    72)   0.093/ 34.702 under Z
  (    65)   0.142/  6.814 under Boot
  (   157)   0.436/ 27.406 under ProfileRoot
  (   294)   2.187/ 26.878 above LZip
  (   294)   1.311/ 41.373 above QZip
EEQ: called 1047 times, time in 0.372/1.37
  (  1047)   0.372/  1.370 under LZip
  (  1047)   0.560/  0.998 above CF
Boot: called 59 times, time in 0.293/10.934
  (     3)   0.015/  0.078 under Z
  (    19)   0.186/  3.764 under Boot
  (    37)   0.092/  7.092 under ProfileRoot
  (    65)   0.142/  6.814 above B
  (    19)   0.186/  3.764 above Boot
  (    47)   0.031/  0.063 above CF
Z: called 1 times, time in 0.016/34.828
  (     1)   0.016/ 34.828 under ProfileRoot
  (    72)   0.093/ 34.702 above B
  (     3)   0.015/  0.078 above Boot
  (     4)   0.016/  0.032 above CF
RVK: called 1 times, time in 0./0.
  (     1)      0/     0 under ProfileRoot
```