Hamiltonian Gauge Gravity Surveyor (HiGGS)

Source notebook for the binary file

build

Initialisation

build

Notebook options

build

In[1]:=
```
AppendTo[$Path, NotebookDirectory[]];
MyImport[x_] :=
  Check[ToExpression["<<" <> NotebookDirectory[] <> "mx_cache/" <> x <> ";"],
   Print["not ready yet..."]];
(*
Check[ToExpression["<<"<>NotebookDirectory[]<>"mx_cache/HiGGS_options.mx;"],
  Print["not ready yet..."]];
*)
```

build

Manifold and geometry

build

In[3]:=
```
Get["variations`"];
dimension = 4;                 (* dimension of space-time manifold *)
DefManifold[M4, dimension, IndexRange[{a, z}]];
AddIndices[TangentM4, {a1, b1, c1, d1, e1, f1, g1, h1, i1,
   j1, k1, l1, n1, m1, o1, p1, q1, r1, s1, t1, u1, v1, w1, x1, y1, z1}];
Quiet[DefMetric[-1, G[-a, -c], CD, {",", "∂"}, PrintAs → "γ",
   FlatMetric → True, SymCovDQ → True]];
```

build

Basic functions

build

In[8]:=
```
(*Probably a better place to put this at the top*)
ToNewCanonical[x_] :=
  Module[{temp, printer}, printer = PrintTemporary["Canonicalizing..."];
   (*Beep[];*)
   temp = x;
   temp = temp // ToCanonical;
   temp = temp // ContractMetric;
   temp = temp // ScreenDollarIndices;
   NotebookDelete[printer];
   temp];

(*This constant symbol will parametrise the perturbation*)
DefConstantSymbol[Prt, PrintAs → "ϵ"];
ToOrderRules = {};

EinsteinHilbert = False;

Options[DeclareOrder] = {"IsUnityWithEHTerm" → False, "approximation" → False};
DeclareOrder[tensor_, order_, OptionsPattern[]] := Module[{tmp},
   If[OptionValue["IsUnityWithEHTerm"] == False ||
      (OptionValue["IsUnityWithEHTerm"] == True && EinsteinHilbert == False),
     If[OptionValue["approximation"] == False,
      tmp = MakeRule[{tensor, Evaluate[Prt^order tensor]},
         MetricOn → All, ContractMetrics → True];,
      tmp = MakeRule[{tensor, Evaluate[Prt^order Evaluate[OptionValue[
              "approximation"]]]}, MetricOn → All, ContractMetrics → True];,
      tmp = MakeRule[{tensor, Evaluate[Prt^order Evaluate[OptionValue[
              "approximation"]]]}, MetricOn → All, ContractMetrics → True];];
     ToOrderRules = Join[ToOrderRules, tmp];];];

CacheBuilt[BinaryName_, Symbols_] := DumpSave[NotebookDirectory[] <>
    "bin/build/" <> SymbolName[BinaryName] <> ".mx;", Symbols];
GetOrBuild[BinaryName_] := (BinaryName = False;
   Check[ToExpression["<<" <> NotebookDirectory[] <>
      "bin/build/" <> SymbolName[BinaryName] <> ".mx;"],
    Print["Can't find " <> NotebookDirectory[] <> "bin/build/" <>
      SymbolName[BinaryName] <> ".mx, so building..."];
    BinaryName = True;
   ];);
```

build

Irreducible decomposition of the fields using SO⁺(1,3)

build

Initial definitions

build

In[16]:=
```
SectorNames =
  {"B0p", "B1p", "B1m", "B2p", "A0p", "A0m", "A1p", "A1m", "A2p", "A2m"};
ASectorNames = {"A0p", "A0m", "A1p", "A1m", "A2p", "A2m"};
BSectorNames = {"B0p", "B0m", "B1p", "B1m", "B2p", "B2m"};
DefTensor[R[a, b, -d, -e], M4,
  {Antisymmetric[{a, b}], Antisymmetric[{-d, -e}]}];
DeclareOrder[R[a, b, -d, -e], 1];
DefTensor[T[a, -b, -c], M4, Antisymmetric[{-b, -c}]];
DeclareOrder[T[a, -b, -c], 1];
DefTensor[W[a, b, -d, -e], M4];
DeclareOrder[W[a, b, -d, -e], 1];
DefTensor[RLambda[a, b, -d, -e], M4,
  {Antisymmetric[{a, b}], Antisymmetric[{-d, -e}]}, PrintAs -> "λ"];
DeclareOrder[RLambda[a, b, -d, -e], 1];
DefTensor[TLambda[a, -d, -e], M4, Antisymmetric[{-d, -e}], PrintAs -> "λ"];
DeclareOrder[TLambda[a, -d, -e], 1];
```

build

Basic/Nester forms of R and T

build

In[29]:=
```
(*
(*This is where we get the notation for generating sets of permutations from,
not the documentation!*)
Print[RiemannSymmetry[{-i,-j,-m,-n}]];
*)
DefTensor[R1[-i, -j, -m, -n], M4,
  StrongGenSet[{-i, -j, -m, -n}, GenSet[Cycles[{-i, -j}, {-m, -n}],
    Cycles[{-i, -m}], Cycles[{-j, -n}]]], PrintAs -> "R(1)"];
DeclareOrder[R1[-i, -j, -m, -n], 1];
DefTensor[R2[-i, -j, -m, -n], M4,
  StrongGenSet[{-i, -j, -m, -n}, GenSet[-Cycles[{-i, -m}, {-j, -n}],
    -Cycles[{-i, -j}], -Cycles[{-m, -n}]]], PrintAs -> "R(2)"];
DeclareOrder[R2[-i, -j, -m, -n], 1];
DefTensor[R3[-i, -j, -m, -n], M4,
  Antisymmetric[{-i, -j, -m, -n}], PrintAs -> "R(3)"];
```

```
DeclareOrder[R3[-i, -j, -m, -n], 1];
DefTensor[R4[-i, -j], M4, Symmetric[{-i, -j}], PrintAs -> "R̊⁽⁴⁾"];
DeclareOrder[R4[-i, -j], 1];
DefTensor[R5[-i, -j], M4, Antisymmetric[{-i, -j}], PrintAs -> "R̊⁽⁵⁾"];
DeclareOrder[R5[-i, -j], 1];
DefTensor[R6[], M4, PrintAs -> "R̊⁽⁶⁾"];
DeclareOrder[R6[], 1];
DefTensor[T1[-i, -j, -k], M4, Symmetric[{-i, -j}], PrintAs -> "T⁽¹⁾"];
DeclareOrder[T1[-i, -j, -k], 1];
DefTensor[T2[-i], M4, PrintAs -> "T⁽²⁾"];
DeclareOrder[T2[-i], 1];
DefTensor[T3[-i], M4, PrintAs -> "T⁽³⁾"];
DeclareOrder[T3[-i], 1];
AutomaticRules[R1,
  MakeRule[{R1[a, a1, b, -b], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[R1, MakeRule[{R1[a, b, a1, -b], 0},
   MetricOn → All, ContractMetrics → True]];
(*AutomaticRules[R1,MakeRule[{R1[a,-a,a1,-a1],0},MetricOn→All,
   ContractMetrics→True]];*)(*redundant*)
(*AutomaticRules[R1,MakeRule[{R1[a,a1,-a,-a1],0},MetricOn→All,
   ContractMetrics→True]];*)(*redundant*)
AutomaticRules[R2, MakeRule[{R2[a, b, a1, -b], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[R4, MakeRule[{R4[a, -a], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[T1,
  MakeRule[{T1[a, a1, -a1], 0}, MetricOn → All, ContractMetrics → True]];

AutomaticRules[T1,
  MakeRule[{T1[a, -a, -k], 0}, MetricOn → All, ContractMetrics → True]];

RDefinition = R3[-i, -j, -m, -n] +
   (2/3) (2 R1[-i, -j, -m, -n] +
      R1[-i, -m, -j, -n]) +
   R2[-i, -j, -m, -n] +
   (1/2) (G[-i, -m] (R5[-j, -n] + R4[-j, -n]) +
      G[-j, -n] (R5[-i, -m] + R4[-i, -m]) -
      G[-j, -m] (R5[-i, -n] + R4[-i, -n]) -
      G[-i, -n] (R5[-j, -m] + R4[-j, -m])) -
   (1/12) (G[-i, -m] G[-j, -n] - G[-i, -n] G[-j, -m]) R6[];

TDefinition = (2/3) (T1[-i, -j, -k] - T1[-i, -k, -j]) +
```

```
    (1/3) (G[-i, -j] T2[-k] - G[-i, -k] T2[-j]) +
    epsilonG[-i, -j, -k, -m] T3[m];

RSO13Activate = MakeRule[{R[-i, -j, -m, -n], Evaluate[RDefinition]},
    MetricOn → All, ContractMetrics → True];
TSO13Activate = MakeRule[{T[-i, -j, -k], Evaluate[TDefinition]},
    MetricOn → All, ContractMetrics → True];

StrengthSO13Activate = Join[RSO13Activate, TSO13Activate];
```

build

Basic/Nester forms of Rλ and Tλ

build

```
In[58]:= DefTensor[RLambda1[-i, -j, -m, -n], M4,
   StrongGenSet[{-i, -j, -m, -n}, GenSet[Cycles[{-i, -j}, {-m, -n}],
     Cycles[{-i, -m}], Cycles[{-j, -n}]]], PrintAs -> "(1)Rλ"];
DeclareOrder[RLambda1[-i, -j, -m, -n], 1];
DefTensor[RLambda2[-i, -j, -m, -n], M4,
   StrongGenSet[{-i, -j, -m, -n}, GenSet[-Cycles[{-i, -m}, {-j, -n}],
     -Cycles[{-i, -j}], -Cycles[{-m, -n}]]], PrintAs -> "(2)Rλ"];
DeclareOrder[RLambda2[-i, -j, -m, -n], 1];
DefTensor[RLambda3[-i, -j, -m, -n],
   M4, Antisymmetric[{-i, -j, -m, -n}], PrintAs -> "(3)Rλ"];
DeclareOrder[RLambda3[-i, -j, -m, -n], 1];
DefTensor[RLambda4[-i, -j], M4, Symmetric[{-i, -j}], PrintAs -> "(4)Rλ"];
DeclareOrder[RLambda4[-i, -j], 1];
DefTensor[RLambda5[-i, -j], M4, Antisymmetric[{-i, -j}], PrintAs -> "(5)Rλ"];
DeclareOrder[RLambda5[-i, -j], 1];
DefTensor[RLambda6[], M4, PrintAs -> "(6)Rλ"];
DeclareOrder[RLambda6[], 1];
DefTensor[TLambda1[-i, -j, -k], M4, Symmetric[{-i, -j}], PrintAs -> "(1)Tλ"];
DeclareOrder[TLambda1[-i, -j, -k], 1];
DefTensor[TLambda2[-i], M4, PrintAs -> "(2)Tλ"];
DeclareOrder[TLambda2[-i], 1];
DefTensor[TLambda3[-i], M4, PrintAs -> "(3)Tλ"];
DeclareOrder[TLambda3[-i], 1];
AutomaticRules[RLambda1,
   MakeRule[{RLambda1[a, a1, b, -b], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[RLambda1, MakeRule[{RLambda1[a, b, a1, -b], 0},
    MetricOn → All, ContractMetrics → True]];
(*AutomaticRules[RLambda1,MakeRule[{RLambda1[a,-a,a1,-a1],0},
```

```
    MetricOn→All,ContractMetrics→True]];*)(*redundant*)
(*AutomaticRules[RLambda1,MakeRule[{RLambda1[a,a1,-a,-a1],0},
    MetricOn→All,ContractMetrics→True]];*)(*redundant*)
AutomaticRules[RLambda2, MakeRule[{RLambda2[a, b, a1, -b], 0},
    MetricOn → All, ContractMetrics → True]];
AutomaticRules[RLambda4, MakeRule[{RLambda4[a, -a], 0},
    MetricOn → All, ContractMetrics → True]];
AutomaticRules[TLambda1, MakeRule[{TLambda1[a, a1, -a1], 0},
    MetricOn → All, ContractMetrics → True]];
AutomaticRules[TLambda1, MakeRule[{TLambda1[a, -a, -a1], 0},
    MetricOn → All, ContractMetrics → True]];

RLambdaDefinition = RLambda3[-i, -j, -m, -n] +
    (2/3) (2 RLambda1[-i, -j, -m, -n] +
        RLambda1[-i, -m, -j, -n]) +
    RLambda2[-i, -j, -m, -n] +
    (1/2) (G[-i, -m] (RLambda5[-j, -n] + RLambda4[-j, -n]) +
        G[-j, -n] (RLambda5[-i, -m] + RLambda4[-i, -m]) -
        G[-j, -m] (RLambda5[-i, -n] + RLambda4[-i, -n]) -
        G[-i, -n] (RLambda5[-j, -m] + RLambda4[-j, -m])) -
    (1/12) (G[-i, -m] G[-j, -n] - G[-i, -n] G[-j, -m]) RLambda6[];

TLambdaDefinition = (2/3) (TLambda1[-i, -j, -k] - TLambda1[-i, -k, -j]) +
    (1/3) (G[-i, -j] TLambda2[-k] - G[-i, -k] TLambda2[-j]) +
    epsilonG[-i, -j, -k, -m] TLambda3[m];

RLambdaSO13Activate =
  MakeRule[{RLambda[-i, -j, -m, -n], Evaluate[RLambdaDefinition]},
    MetricOn → All, ContractMetrics → True];
TLambdaSO13Activate = MakeRule[{TLambda[-i, -j, -k], Evaluate[TLambdaDefinition]},
    MetricOn → All, ContractMetrics → True];

StrengthLambdaSO13Activate = Join[RLambdaSO13Activate, TLambdaSO13Activate];
```

build

Basic/Nester forms of σ

build

In[87]:=
```
DefTensor[Spin1[-i, -j, -k], M4, Symmetric[{-i, -j}], PrintAs -> "σ̇⁽¹⁾"];
DeclareOrder[Spin1[-i, -j, -k], 1];
DefTensor[Spin2[-i], M4, PrintAs -> "σ̇⁽²⁾"];
DeclareOrder[Spin2[-i], 1];
DefTensor[Spin3[-i], M4, PrintAs -> "σ̇⁽³⁾"];
DeclareOrder[Spin3[-i], 1];
AutomaticRules[Spin1,
  MakeRule[{Spin1[a, a1, -a1], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[Spin1, MakeRule[{Spin1[a, -a, -a1], 0},
   MetricOn → All, ContractMetrics → True]];

SpinDefinition = (2/3) (Spin1[-i, -j, -k] - Spin1[-i, -k, -j]) +
   (1/3) (G[-i, -j] Spin2[-k] - G[-i, -k] Spin2[-j]) +
   epsilonG[-i, -j, -k, -m] Spin3[m];

DefTensor[STensor[-i, -j, -k], M4, Antisymmetric[{-j, -k}], PrintAs → "σ"];
DeclareOrder[STensor[-i, -j, -k], 1];

SpinSO13Activate = MakeRule[{STensor[-i, -j, -k], Evaluate[SpinDefinition]},
   MetricOn → All, ContractMetrics → True];

StrengthLambdaSO13Activate = Join[RLambdaSO13Activate, TLambdaSO13Activate];
```

build

ORPHAN

build

In[100]:=
```
(*
MyMakeTraceless[expr_,name_]:=Module[{res,TensorRank,NumberFrees,
   TensorContractions,TensorFreeIndices,TensorFreeIndexList},
  res=Evaluate[expr];
  TensorFreeIndices=FindFreeIndices[Evaluate[res]];
  TensorFreeIndexList=
   Developer`ToList[Delete[Map[ToString[#]&,TensorFreeIndices],0]];
  TensorRank=Length[TensorFreeIndexList];
  NumberFrees=Range[TensorRank-2,0,-2];
  TensorContractions=
   AllContractions[expr,FreeMetrics→None,UncontractedIndices→#]&/@NumberFrees;
  TensorContractions=Flatten[TensorContractions];
  Print[TensorContractions];
  (ToExpression["AutomaticRules["<>ToString[name]<>",MakeRule[{"<>ToString[#]<>
       ",0},MetricOn→All,ContractMetrics→True]]"])&/@TensorContractions;];

Print[MyMakeTraceless[R1[-i,-j,-m,-n],"R1"]];
```

```
Print[MyMakeTraceless[R2[-i,-j,-m,-n],"R2"]];
Print[MyMakeTraceless[R3[-i,-j,-m,-n],"R3"]];
Print[MyMakeTraceless[R4[-i,-j],"R4"]];
Print[MyMakeTraceless[R5[-i,-j],"R5"]];
Print[MyMakeTraceless[R6[],"R6"]];
Print[MyMakeTraceless[T1[-i,-j,-k],"T1"]];
Print[MyMakeTraceless[T2[-i],"T2"]];
Print[MyMakeTraceless[T3[-i],"T3"]];

tmp=ToCanonical/@{R1[a,a1,b,-b],R1[a,b,a1,-b],R1[a,-a,a1,-a1],R1[a,a1,-a,-a1]};
Print[tmp];
Quit[];
*)

(*
IrrepGenerators={{{{"R1","R(1)"},{"RLambda1","Rλ(1)"}},
   (1/4) (W[-i,-j,-m,-n]+W[-m,-n,-i,-j]-W[-i,-n,-j,-m]-W[-j,-m,-i,-n])},
  {{{"R2","R(2)"},{"RLambda2","Rλ(2)"}},(1/2) (W[-i,-j,-m,-n]-W[-m,-n,-i,-j])},
  {{{"R3","R(3)"},{"RLambda3","Rλ(3)"}},(1/6) (R[-i,-j,-m,-n]+R[-i,-m,-n,-j]+
      R[-i,-n,-j,-m]+R[-j,-m,-i,-n]+R[-j,-n,-m,-i]+R[-m,-n,-i,-j])},
  {{{"R4","R(4)"},{"RLambda4","Rλ(4)"}},(1/2) (R[-i,k,-j,-k]+R[-j,k,-i,-k])-
    (1/4)G[-i,-j]R[l,k,-l,-k]},
  {{{"R5","R(5)"},{"RLambda5","Rλ(5)"}},(1/2) (R[-i,k,-j,-k]-R[-j,k,-i,-k])},
  {{{"R6","R(6)"},{"RLambda6","Rλ(6)"}},R[l,k,-l,-k]},
  {{{"T1","T(1)"},{"TLambda1","Tλ(1)"}},T[-i,-j,-k]},
  {{{"T2","T(2)"},{"RLambda2","Tλ(2)"}},T[k,-k,-i]},
  {{{"T3","T(3)"},{"TLambda3","Tλ(3)"}},(1/6)epsilonG[-i,-j,-m,-n]T[j,m,n]}};

GenerateIrreps[expr_]:=Module[{NewTensors,shape,ModelIndices,ModelSymmetryTotal,
   ModelSymmetryIndices,ModelSymmetryUsable,IndicesString,SymmetryString},
  NewTensors=Evaluate[expr[[1]]];
  shape=Evaluate[expr[[2]]];
  ModelIndices=FindFreeIndices[shape];
  ModelSymmetryTotal=SymmetryOf[shape];
  ModelSymmetryIndices=
   Map[ToExpression[StringDelete[ToString[#],"●"]]&,ModelSymmetryTotal[[3]]];
  ModelSymmetryUsable=ModelSymmetryTotal[[4]]/.ModelSymmetryIndices;
  IndicesString=ToString[StringTrim[ToString[ModelIndices],("IndexList["|"]")]];
  SymmetryString=ToString[ModelSymmetryUsable];
  (ToExpression["DefTensor["<>ToString[#[[1]]]<>"["<>IndicesString<>"],M4,"<>
       SymmetryString<>",PrintAs->"<>ToString[#[[2]]]<>"]"];)&/@NewTensors;];
```

```
GenerateIrreps/@IrrepGenerators
*)
```

build

Define complete projections {ᴵ𝒫̂}, {ᴹ𝒫̂}

build

In[101]:=
```
DefTensor[PR1[-a, -b, -c, -d, e, f, g, h], M4, PrintAs → "R1𝒫̂"];
DefTensor[PR2[-a, -b, -c, -d, e, f, g, h], M4, PrintAs → "R2𝒫̂"];
DefTensor[PR3[-i, -k, -l, -m, a, b, c, d], M4, PrintAs → "R3𝒫̂"];
DefTensor[PR4[-i, -k, -l, -m, a, b, c, d], M4, PrintAs → "R4𝒫̂"];
DefTensor[PR5[-i, -k, -l, -m, a, b, c, d], M4, PrintAs → "R5𝒫̂"];
DefTensor[PR6[-i, -k, -l, -m, a, b, c, d], M4, PrintAs → "R6𝒫̂"];

ToCanonicalTotal[x_] := ToCanonical[Total[x]];
ToCanonicalParallel[x_] := Module[{Monomials, Ret},
   Monomials = MonomialList[x];
   Ret = Total[ParallelCombine[ToCanonicalTotal, Monomials, List]];
   Ret];

AutomaticRules[PR1, MakeRule[{CD[-x][PR1[-a, -b, -c, -d, e, f, g, h]], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[PR2, MakeRule[{CD[-x][PR2[-a, -b, -c, -d, e, f, g, h]], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[PR3, MakeRule[{CD[-x][PR3[-a, -b, -c, -d, e, f, g, h]], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[PR4, MakeRule[{CD[-x][PR4[-a, -b, -c, -d, e, f, g, h]], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[PR5, MakeRule[{CD[-x][PR5[-a, -b, -c, -d, e, f, g, h]], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[PR6, MakeRule[{CD[-x][PR6[-a, -b, -c, -d, e, f, g, h]], 0},
   MetricOn → All, ContractMetrics → True]];
DefTensor[PW[-i, -k, -l, -m, a, b, c, d], M4, PrintAs → "W𝒫̂"];
DefTensor[PT1[-a, -b, -c, e, f, g], M4, PrintAs → "T1𝒫̂"];
DefTensor[PT2[-a, -b, -c, e, f, g], M4, PrintAs → "T2𝒫̂"];
DefTensor[PT3[-a, -b, -c, e, f, g], M4, PrintAs → "T3𝒫̂"];
AutomaticRules[PT1, MakeRule[
   {CD[-x][PT1[-a, -b, -c, e, f, g]], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[PT2, MakeRule[{CD[-x][PT2[-a, -b, -c, e, f, g]], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[PT3, MakeRule[{CD[-x][PT3[-a, -b, -c, e, f, g]], 0},
   MetricOn → All, ContractMetrics → True]];
```

build

O13ProjectionsToggle

build

In[122]:=
```
GetOrBuild[O13ProjectionsToggle];
If[O13ProjectionsToggle,
  PWActivate =
   MakeRule[{PW[-i, -k, -l, -m, a, b, c, d], G[a, -i] G[b, -k] G[c, -l] G[d, -m] +
      (1/2) (G[b, d] G[a, -i] G[c, -m] G[-k, -l] - G[b, d] G[a, -i] G[c, -l] G[-k, -m] +
         G[b, d] G[a, -k] G[c, -l] G[-i, -m] - G[b, d] G[a, -k] G[c, -m] G[-i, -l]) +
      (1/6) G[a, c] G[b, d] (G[-i, -l] G[-k, -m] - G[-i, -m] G[-k, -l])},
    MetricOn → All, ContractMetrics → True];

  PR1Definition =
   Antisymmetrize[Antisymmetrize[Antisymmetrize[Antisymmetrize[(2/3) G[s, -i]
          G[r, -n] (2 G[p, -j] G[q, -m] + G[p, -m] G[q, -j]) (1/2) (Symmetrize[
             PW[-s, -p, -q, -r, a, b, c, d] + PW[-s, -r, -q, -p, a, b, c, d], {-s, -q}]),
         {-i, -j}], {-m, -n}], {a, b}], {c, d}] /. PWActivate // ToCanonical;
  PR1Activate = MakeRule[{PR1[-i, -j, -m, -n, a, b, c, d], Evaluate[PR1Definition]},
    MetricOn → All, ContractMetrics → True];

  PR2Definition =
   Antisymmetrize[Antisymmetrize[Antisymmetrize[Antisymmetrize[(1/2)
          (PW[-i, -j, -m, -n, a, b, c, d] - PW[-m, -n, -i, -j, a, b, c, d]), {-i, -j}],
        {-m, -n}], {a, b}], {c, d}] /. PWActivate // ToCanonical;
  PR2Activate = MakeRule[{PR2[-i, -j, -m, -n, a, b, c, d], Evaluate[PR2Definition]},
    MetricOn → All, ContractMetrics → True];

  PR3Definition = Antisymmetrize[Antisymmetrize[Antisymmetrize[
       Antisymmetrize[(-1/4) (1/6) epsilonG[-i, -j, -m, -n] epsilonG[a, b, c, d],
        {-i, -j}], {-m, -n}], {a, b}], {c, d}] // ToCanonical;
  PR3Activate = MakeRule[{PR3[-i, -j, -m, -n, a, b, c, d], Evaluate[PR3Definition]},
    MetricOn → All, ContractMetrics → True];

  PR4Definition =
   Antisymmetrize[Antisymmetrize[Antisymmetrize[Antisymmetrize[(1/2)
         (G[-i, -m] G[x, -j] G[y, -n] + G[-j, -n] G[x, -i] G[y, -m] - G[-j, -m] G[x, -i]
            G[y, -n] - G[-i, -n] G[x, -j] G[y, -m]) (Symmetrize[
            G[-x, a] G[-y, c] G[b, d], {-x, -y}] - (1/4) G[-x, -y] G[b, d] G[a, c]),
        {-i, -j}], {-m, -n}], {a, b}], {c, d}] // ToCanonical;
  PR4Activate = MakeRule[{PR4[-i, -j, -m, -n, a, b, c, d], Evaluate[PR4Definition]},
    MetricOn → All, ContractMetrics → True];
```

```
  PR5Definition =
   Antisymmetrize[Antisymmetrize[Antisymmetrize[Antisymmetrize[(1/2)
         (G[-i, -m] G[x, -j] G[y, -n] + G[-j, -n] G[x, -i] G[y, -m] - G[-j, -m] G[x, -i]
            G[y, -n] - G[-i, -n] G[x, -j] G[y, -m])
         Antisymmetrize[G[-x, a] G[-y, c] G[b, d], {-x, -y}], {-i, -j}],
       {-m, -n}], {a, b}], {c, d}] // ToCanonical;
  PR5Activate = MakeRule[{PR5[-i, -j, -m, -n, a, b, c, d], Evaluate[PR5Definition]},
    MetricOn → All, ContractMetrics → True];

  PR6Definition =
   Antisymmetrize[Antisymmetrize[Antisymmetrize[Antisymmetrize[-(1/6)
         G[b, d] G[a, c] (G[-i, -j] G[-m, -n] - G[-i, -m] G[-j, -n]), {-i, -j}],
       {-m, -n}], {a, b}], {c, d}] // ToCanonical;
  PR6Activate = MakeRule[{PR6[-i, -j, -m, -n, a, b, c, d], Evaluate[PR6Definition]},
    MetricOn → All, ContractMetrics → True];

  PT1Definition =
   Antisymmetrize[Antisymmetrize[(4/3) (Symmetrize[G[-i, a] G[-j, b] G[-k, c] +
           (1/3) G[-k, -i] G[a, b] G[c, -j], {-i, -j}] -
         (1/3) G[-i, -j] G[a, b] G[c, -k]), {-j, -k}], {b, c}] // ToCanonical;
  PT1Activate = MakeRule[{PT1[-i, -j, -k, a, b, c], Evaluate[PT1Definition]},
    MetricOn → All, ContractMetrics → True];

  PT2Definition = Antisymmetrize[Antisymmetrize[
      (2/3) G[-i, -j] G[a, b] G[c, -k], {-j, -k}], {b, c}] // ToCanonical;
  PT2Activate = MakeRule[{PT2[-i, -j, -k, a, b, c], Evaluate[PT2Definition]},
    MetricOn → All, ContractMetrics → True];

  PT3Definition = Antisymmetrize[Antisymmetrize[(1/6) epsilonG[-i, -j, -k, -m]
       epsilonG[m, a, b, c], {-j, -k}], {b, c}] // ToCanonical;
  PT3Activate = MakeRule[{PT3[-i, -j, -k, a, b, c], Evaluate[PT3Definition]},
    MetricOn → All, ContractMetrics → True];

  PActivate = Join[PWActivate, PR1Activate, PR2Activate, PR3Activate, PR4Activate,
    PR5Activate, PR6Activate, PT1Activate, PT2Activate, PT3Activate];

  CacheBuilt[O13ProjectionsToggle, {PActivate}];
 ];
(*
DumpSave[NotebookDirectory[]<>"mx_cache/O13Projections.mx",{PActivate}];
Check[ToExpression["<<"<>NotebookDirectory[]<>"mx_cache/O13Projections.mx;"],
 Print["not ready yet..."]];
*)
```

CheckOrthogonalityToggle

```
If[CheckOrthogonalityToggle,
 Print[Style["checking orthogonality", Blue, 16]];
 For[ii = 1, ii < 7, ii++, For[jj = 1, jj < 7, jj++, If[ii ≠ jj, Print[
     ToExpression["PR" <> ToString[ii] <> "[-i,-k,-l,-m,a,b,c,d]PR" <> ToString[jj] <>
         "[-a,-b,-c,-d,e,f,g,h]R[-e,-f,-g,-h]"] /. PActivate // ToCanonical]]]];
 For[ii = 1, ii < 4, ii++, For[jj = 1, jj < 4, jj++, If[ii ≠ jj,
    Print[ToExpression["PT" <> ToString[ii] <> "[-i,-j,-k,a,b,c]PT" <> ToString[jj] <>
         "[-a,-b,-c,e,f,g]T[-e,-f,-g]"] /. PActivate // ToCanonical]]]];

 Print[Style["checking inverse orthogonality", Blue, 16]];

 For[ii = 1, ii < 7, ii++,
   For[jj = 1, jj < 7, jj++, If[ii ≠ jj, Print[ToExpression["PR" <> ToString[ii] <>
          "[a,b,c,d,i,j,k,l]R[-i,-j,-k,-l]PR" <> ToString[jj] <>
          "[-a,-b,-c,-d,e,f,g,h]R[-e,-f,-g,-h]"] /. PActivate // ToCanonical]]]] ×
  For[ii = 1, ii < 4, ii++, For[jj = 1, jj < 4, jj++, If[ii ≠ jj, Print[
      ToExpression["PT" <> ToString[ii] <> "[a,b,c,i,j,k]T[-i,-j,-k]PT" <> ToString[
           jj] <> "[-a,-b,-c,e,f,g]T[-e,-f,-g]"] /. PActivate // ToCanonical]]]];

 Print[Style["checking idempotency", Blue, 16]];

 For[ii = 1, ii < 7, ii++,
   Print[ToExpression["(PR" <> ToString[ii] <> "[-i,-k,-l,-m,a,b,c,d]PR" <>
         ToString[ii] <> "[-a,-b,-c,-d,e,f,g,h]-PR" <>
         ToString[ii] <> "[-i,-k,-l,-m,e,f,g,h])R[-e,-f,-g,-h]"] /.
       PActivate // ToCanonical // FullSimplify]] ×
  For[ii = 1, ii < 4, ii++, Print[ToExpression["(PT" <> ToString[ii] <>
         "[-i,-j,-k,a,b,c]PT" <> ToString[ii] <> "[-a,-b,-c,e,f,g]-PT" <>
         ToString[ii] <> "[-i,-j,-k,e,f,g])T[-e,-f,-g]"] /.
       PActivate // ToCanonical // FullSimplify]];

 Print[Style["checking completeness", Blue, 16]];

 (PR1[-i, -k, -l, -m, a, b, c, d] + PR2[-i, -k, -l, -m, a, b, c, d] +
        PR3[-i, -k, -l, -m, a, b, c, d] + PR4[-i, -k, -l, -m, a, b, c, d] +
        PR5[-i, -k, -l, -m, a, b, c, d] + PR6[-i, -k, -l, -m, a, b, c, d])
      R[-a, -b, -c, -d] /. PActivate // ToCanonical // Simplify ×
    (PT1[-i, -k, -l, a, b, c] + PT2[-i, -k, -l, a, b, c] + PT3[-i, -k, -l, a, b, c])
       T[-a, -b, -c] /. PActivate // ToCanonical // Simplify;

 Print[Style["checking invertability", Blue, 16]];
```

```
For[ii = 1, ii < 7, ii++,
 Print[ToExpression["(PR" <> ToString[ii] <> "[e,f,g,h,-i,-k,-l,-m]-PR" <>
       ToString[ii] <> "[-i,-k,-l,-m,e,f,g,h])R[-e,-f,-g,-h]"] /.
     PActivate // ToCanonical // FullSimplify]];

Quit[];
]
```

Define Ricci, Ricci scalar and torsion contraction

```
(*Define the Ricci \mathcal{R}^a_{\ b}*)
DefTensor[Rc[a, -b], M4, PrintAs → "R"];
DeclareOrder[Rc[a, -b], 1];
(*Define the Ricci scalar \mathcal{R}*)
DefTensor[Rs[], M4, PrintAs → "R"];
DeclareOrder[Rs[], 1];
(*Define the torsion contraction \mathcal{T}^a*)
DefTensor[Tc[-a], M4, PrintAs → "T"];
DeclareOrder[Tc[-a], 1];
(*Rule to expand Ricci*)
ExpandRicci =
  MakeRule[{Rc[a, -b], R[c, a, -c, -b]}, MetricOn → All, ContractMetrics → True];
(*Rule to expand Ricci scalar*)
ExpandRicciScalar =
  MakeRule[{Rs[], R[c, d, -c, -d]}, MetricOn → All, ContractMetrics → True];
(*Rule to expand torsion contraction*)
TorsionExpandContraction =
  MakeRule[{Tc[-a], T[b, -a, -b]}, MetricOn → All, ContractMetrics → True];
(*Total rule to expand contracted field-strength tensors*)
ExpandContractedStrengths =
  Join[ExpandRicci, ExpandRicciScalar, TorsionExpandContraction];

(*Rule to expand Ricci*)
ContractRicci =
  MakeRule[{R[c, a, -c, -b], Rc[a, -b]}, MetricOn → All, ContractMetrics → True];
(*Rule to expand Ricci scalar*)
ContractRicciScalar =
  MakeRule[{R[c, d, -c, -d], Rs[]}, MetricOn → All, ContractMetrics → True];
(*Rule to expand torsion contraction*)
TorsionContractContraction =
  MakeRule[{T[b, -a, -b], Tc[-a]}, MetricOn → All, ContractMetrics → True];
(*Total rule to expand contracted field-strength tensors*)
ContractExpandedStrengths =
  Join[ContractRicci, ContractRicciScalar, TorsionContractContraction];
```

ShowIrrepsToggle

build

In[124]:=
```
(*Irreducible decompositions*)
If[ShowIrrepsToggle,
  AutomaticRules[R,
   MakeRule[{R[c, a, -c, -b], Rc[a, -b]}, MetricOn → All, ContractMetrics → True]];
  AutomaticRules[Rc, MakeRule[{Rc[c, -c], Rs[]},
    MetricOn → All, ContractMetrics → True]];
```

```
AutomaticRules[T, MakeRule[{T[c, -a, -c], Tc[-a]},
  MetricOn → All, ContractMetrics → True]];
PR1[-i, -j, -k, -l, a, b, c, d] R[-a, -b, -c, -d] /. PActivate // ToCanonical //
 ContractMetric;
Print[%];
PR2[-i, -j, -k, -l, a, b, c, d] R[-a, -b, -c, -d] /. PActivate // ToCanonical //
 ContractMetric;
Print[%];
PR3[-i, -j, -k, -l, a, b, c, d] R[-a, -b, -c, -d] /. PActivate // ToCanonical //
 ContractMetric;
Print[%];
PR4[-i, -j, -k, -l, a, b, c, d] R[-a, -b, -c, -d] /. PActivate // ToCanonical //
 ContractMetric;
Print[%];
PR5[-i, -j, -k, -l, a, b, c, d] R[-a, -b, -c, -d] /. PActivate // ToCanonical //
 ContractMetric;
Print[%];
PR6[-i, -j, -k, -l, a, b, c, d] R[-a, -b, -c, -d] /. PActivate // ToCanonical //
 ContractMetric;
Print[%];
PT1[-i, -j, -k, a, b, c] T[-a, -b, -c] /. PActivate // ToCanonical //
 ContractMetric;
Print[%];
PT2[-i, -j, -k, a, b, c] T[-a, -b, -c] /. PActivate // ToCanonical //
 ContractMetric;
Print[%];
PT3[-i, -j, -k, a, b, c] T[-a, -b, -c] /. PActivate // ToCanonical //
 ContractMetric;
Print[%];

tmp = PR1[-i, -j, -k, -l, a, b, c, d] R[-a, -b, -c, -d] /. PActivate /.
   StrengthSO13Activate // ToNewCanonical;
Print[tmp];
tmp = PR2[-i, -j, -k, -l, a, b, c, d] R[-a, -b, -c, -d] /. PActivate /.
   StrengthSO13Activate // ToNewCanonical;
Print[tmp];
tmp = PR3[-i, -j, -k, -l, a, b, c, d] R[-a, -b, -c, -d] /. PActivate /.
   StrengthSO13Activate // ToNewCanonical;
Print[tmp];
tmp = PR4[-i, -j, -k, -l, a, b, c, d] R[-a, -b, -c, -d] /. PActivate /.
   StrengthSO13Activate // ToNewCanonical;
Print[tmp];
tmp = PR5[-i, -j, -k, -l, a, b, c, d] R[-a, -b, -c, -d] /. PActivate /.
   StrengthSO13Activate // ToNewCanonical;
```

```
  Print[tmp];
  tmp = PR6[-i, -j, -k, -l, a, b, c, d] R[-a, -b, -c, -d] /. PActivate /.
      StrengthSO13Activate // ToNewCanonical;
  Print[tmp];
  tmp = PT1[-i, -j, -k, a, b, c] T[-a, -b, -c] /. PActivate /.
      StrengthSO13Activate // ToNewCanonical;
  Print[tmp];
  tmp = PT2[-i, -j, -k, a, b, c] T[-a, -b, -c] /. PActivate /.
      StrengthSO13Activate // ToNewCanonical;
  Print[tmp];
  tmp = PT3[-i, -j, -k, a, b, c] T[-a, -b, -c] /. PActivate /.
      StrengthSO13Activate // ToNewCanonical;
  Print[tmp];

  tmp = PR1[-i, -j, -k, -l, a, b, c, d] RLambda[-a, -b, -c, -d] /. PActivate /.
      StrengthLambdaSO13Activate // ToNewCanonical;
  Print[tmp];
  tmp = PR2[-i, -j, -k, -l, a, b, c, d] RLambda[-a, -b, -c, -d] /. PActivate /.
      StrengthLambdaSO13Activate // ToNewCanonical;
  Print[tmp];
  tmp = PR3[-i, -j, -k, -l, a, b, c, d] RLambda[-a, -b, -c, -d] /. PActivate /.
      StrengthLambdaSO13Activate // ToNewCanonical;
  Print[tmp];
  tmp = PR4[-i, -j, -k, -l, a, b, c, d] RLambda[-a, -b, -c, -d] /. PActivate /.
      StrengthLambdaSO13Activate // ToNewCanonical;
  Print[tmp];
  tmp = PR5[-i, -j, -k, -l, a, b, c, d] RLambda[-a, -b, -c, -d] /. PActivate /.
      StrengthLambdaSO13Activate // ToNewCanonical;
  Print[tmp];
  tmp = PR6[-i, -j, -k, -l, a, b, c, d] RLambda[-a, -b, -c, -d] /. PActivate /.
      StrengthLambdaSO13Activate // ToNewCanonical;
  Print[tmp];
  tmp = PT1[-i, -j, -k, a, b, c] TLambda[-a, -b, -c] /. PActivate /.
      StrengthLambdaSO13Activate // ToNewCanonical;
  Print[tmp];
  tmp = PT2[-i, -j, -k, a, b, c] TLambda[-a, -b, -c] /. PActivate /.
      StrengthLambdaSO13Activate // ToNewCanonical;
  Print[tmp];
  tmp = PT3[-i, -j, -k, a, b, c] TLambda[-a, -b, -c] /. PActivate /.
      StrengthLambdaSO13Activate // ToNewCanonical;
  Print[tmp];

  Quit[];
 ];
```

build

ORPHAN

build

In[125]:=

```
(*
Alphas=DefNiceConstantSymbol[α,#,W]&/@Range[6];
DefNiceConstantSymbol[C,0];
DefNiceConstantSymbol[C,1];
DefNiceConstantSymbol[C,2];
DefNiceConstantSymbol[C,3];

tmp=C0 (R[i,j,-i,-j]R[a,b,-a,-b]-
      4R[a,-i,-a,-j]R[b,j,-b,i]+R[i,j,k,l]R[-k,-l,-i,-j])/.PActivate;
tmp=tmp//ToNewCanonical;
tmp=tmp/.StrengthSO13Activate;
tmp0=tmp//ToNewCanonical;
Print[tmp];

tmp=R[i,j,k,l](-2α6W PR1[-i,-j,-k,-l,a,b,c,d]
      +2α6W PR2[-i,-j,-k,-l,a,b,c,d]
      +α3W PR3[-i,-j,-k,-l,a,b,c,d]
      -α6W PR4[-i,-j,-k,-l,a,b,c,d]
      +α5W PR5[-i,-j,-k,-l,a,b,c,d]
      +α6W PR6[-i,-j,-k,-l,a,b,c,d])R[-a,-b,-c,-d]/.PActivate;
tmp=tmp//ToNewCanonical;
tmp=tmp/.StrengthSO13Activate;
tmp1=tmp//ToNewCanonical;
Print[tmp];

tmp=C1 PW[-i,-j,-k,-l,-p,-q,-u,-v]R[p,q,u,v]PW[i,j,k,l,-a,-b,-c,-d]R[a,b,c,d]+
   C2 PW[-i,-l,-k,-j,-p,-q,-u,-v]R[p,q,u,v]PW[i,j,k,l,-a,-b,-c,-d]R[a,b,c,d]+
   C3 PW[-k,-l,-i,-j,-p,-q,-u,-v]R[p,q,u,v]PW[i,j,k,l,-a,-b,-c,-d]R[a,b,c,d]/.
  PActivate;
tmp=tmp//ToNewCanonical;
tmp=tmp/.StrengthSO13Activate;
tmp2=tmp//ToNewCanonical;
Print[tmp];

Print["herehere"];

tmp=tmp0+tmp1+tmp2//ToNewCanonical;
Print[tmp];
tmp=tmp//CollectTensors;
```

```
Print[tmp];
equations=ToConstantSymbolEquations[tmp==0];
Print[equations]
 sols=Quiet[Solve[equations,{C0,C1,C2,C3}]];
Print[sols]
 sols=Quiet[Solve[equations,{C0,C1,C2,C3,α5W,α3W}]];
Print[sols]
 Quit[];
*)
```

build

ORPHAN

build

```
In[126]:= (*I think this is now not needed*)
(*
(*My couplings for irreps*)
Alphas=DefNiceConstantSymbol[α,#,W]&/@Range[6];
DefNiceConstantSymbol[α,0,W];
(*Added this for extra safety checks with literature*)
Betas=DefNiceConstantSymbol[β,#,W]&/@Range[3];
(*Mike's couplings*)
DefNiceConstantSymbol[α,#,M]&/@Range[6];
DefNiceConstantSymbol[β,#,M]&/@Range[3];
(*Not sure what this is*)
DefNiceConstantSymbol[c,#]&/@Range[3];
(*My couplings for geometric algebra*)
DefNiceConstantSymbol[α,#,G]&/@Range[6];
Betas=DefNiceConstantSymbol[β,#,G]&/@Range[3];
TheoryParameters=Join[Alphas,Betas];
*)
```

build

Multiplier couplings $\{\bar{\alpha}_I\}$, $\{\bar{\beta}_M\}$

build

```
In[127]:= (*My couplings for irrep Lorentz constraints*)
DefConstantSymbol[cAlp1, PrintAs → "ᾱ1"];
DefConstantSymbol[cAlp2, PrintAs → "ᾱ2"];
DefConstantSymbol[cAlp3, PrintAs → "ᾱ3"];
DefConstantSymbol[cAlp4, PrintAs → "ᾱ4"];
DefConstantSymbol[cAlp5, PrintAs → "ᾱ5"];
DefConstantSymbol[cAlp6, PrintAs → "ᾱ6"];

cAlp = {cAlp1, cAlp2, cAlp3, cAlp4, cAlp5, cAlp6};
```

```
(*My couplings for irrep Lorentz constraints*)
DefConstantSymbol[gAlp1, PrintAs → "ά_1"];
DefConstantSymbol[gAlp2, PrintAs → "ά_2"];
DefConstantSymbol[gAlp3, PrintAs → "ά_3"];
DefConstantSymbol[gAlp4, PrintAs → "ά_4"];
DefConstantSymbol[gAlp5, PrintAs → "ά_5"];
DefConstantSymbol[gAlp6, PrintAs → "ά_6"];

gAlp = {gAlp1, gAlp2, gAlp3, gAlp4, gAlp5, gAlp6};

DefConstantSymbol[cAlpParaPara0p, PrintAs → "α̅^∥∥_0+"];
DefConstantSymbol[cAlpParaPara0m, PrintAs → "α̅^∥∥_0-"];
DefConstantSymbol[cAlpParaPara1p, PrintAs → "α̅^∥∥_1+"];
DefConstantSymbol[cAlpParaPara1m, PrintAs → "α̅^∥∥_1-"];
DefConstantSymbol[cAlpParaPara2p, PrintAs → "α̅^∥∥_2+"];
DefConstantSymbol[cAlpParaPara2m, PrintAs → "α̅^∥∥_2-"];

cAlpParaPara = {cAlpParaPara0p, cAlpParaPara0m,
   cAlpParaPara1p, cAlpParaPara1m, cAlpParaPara2p, cAlpParaPara2m};

DefConstantSymbol[cAlpPerpPerp0p, PrintAs → "α̅^⊥⊥_0+"];
DefConstantSymbol[cAlpPerpPerp0m, PrintAs → "α̅^⊥⊥_0-"];
DefConstantSymbol[cAlpPerpPerp1p, PrintAs → "α̅^⊥⊥_1+"];
DefConstantSymbol[cAlpPerpPerp1m, PrintAs → "α̅^⊥⊥_1-"];
DefConstantSymbol[cAlpPerpPerp2p, PrintAs → "α̅^⊥⊥_2+"];
DefConstantSymbol[cAlpPerpPerp2m, PrintAs → "α̅^⊥⊥_2-"];

cAlpPerpPerp = {cAlpPerpPerp0p, cAlpPerpPerp0m,
   cAlpPerpPerp1p, cAlpPerpPerp1m, cAlpPerpPerp2p, cAlpPerpPerp2m};

DefConstantSymbol[cAlpPerpPara0p, PrintAs → "α̅^⊥∥_0+"];
DefConstantSymbol[cAlpPerpPara0m, PrintAs → "α̅^⊥∥_0-"];
DefConstantSymbol[cAlpPerpPara1p, PrintAs → "α̅^⊥∥_1+"];
DefConstantSymbol[cAlpPerpPara1m, PrintAs → "α̅^⊥∥_1-"];
DefConstantSymbol[cAlpPerpPara2p, PrintAs → "α̅^⊥∥_2+"];
DefConstantSymbol[cAlpPerpPara2m, PrintAs → "α̅^⊥∥_2-"];

cAlpPerpPara = {cAlpPerpPara0p, cAlpPerpPara0m,
   cAlpPerpPara1p, cAlpPerpPara1m, cAlpPerpPara2p, cAlpPerpPara2m};

DefConstantSymbol[cAlpParaPerp0p, PrintAs → "α̅^∥⊥_0+"];
DefConstantSymbol[cAlpParaPerp0m, PrintAs → "α̅^∥⊥_0-"];
```

```
DefConstantSymbol[cAlpParaPerp1p, PrintAs → "α̅∥⊥1+"];
DefConstantSymbol[cAlpParaPerp1m, PrintAs → "α̅∥⊥1-"];
DefConstantSymbol[cAlpParaPerp2p, PrintAs → "α̅∥⊥2+"];
DefConstantSymbol[cAlpParaPerp2m, PrintAs → "α̅∥⊥2-"];

cAlpParaPerp = {cAlpParaPerp0p, cAlpParaPerp0m,
   cAlpParaPerp1p, cAlpParaPerp1m, cAlpParaPerp2p, cAlpParaPerp2m};

DefConstantSymbol[cBet1, PrintAs → "β̅1"];
DefConstantSymbol[cBet2, PrintAs → "β̅2"];
DefConstantSymbol[cBet3, PrintAs → "β̅3"];
DefConstantSymbol[cBet4, PrintAs → "β̅4"];
DefConstantSymbol[cBet5, PrintAs → "β̅5"];
DefConstantSymbol[cBet6, PrintAs → "β̅6"];

cBet = {cBet1, cBet2, cBet3};

DefConstantSymbol[gBet1, PrintAs → "β́1"];
DefConstantSymbol[gBet2, PrintAs → "β́2"];
DefConstantSymbol[gBet3, PrintAs → "β́3"];
DefConstantSymbol[gBet4, PrintAs → "β́4"];
DefConstantSymbol[gBet5, PrintAs → "β́5"];
DefConstantSymbol[gBet6, PrintAs → "β́6"];

gBet = {gBet1, gBet2, gBet3};

DefConstantSymbol[cBetParaPara0p, PrintAs → "β̅∥∥0+"];
DefConstantSymbol[cBetParaPara0m, PrintAs → "β̅∥∥0-"];
DefConstantSymbol[cBetParaPara1p, PrintAs → "β̅∥∥1+"];
DefConstantSymbol[cBetParaPara1m, PrintAs → "β̅∥∥1-"];
DefConstantSymbol[cBetParaPara2p, PrintAs → "β̅∥∥2+"];
DefConstantSymbol[cBetParaPara2m, PrintAs → "β̅∥∥2-"];

cBetParaPara = {cBetParaPara0p, cBetParaPara0m,
   cBetParaPara1p, cBetParaPara1m, cBetParaPara2p, cBetParaPara2m};

DefConstantSymbol[cBetPerpPerp0p, PrintAs → "β̅⊥⊥0+"];
DefConstantSymbol[cBetPerpPerp0m, PrintAs → "β̅⊥⊥0-"];
DefConstantSymbol[cBetPerpPerp1p, PrintAs → "β̅⊥⊥1+"];
DefConstantSymbol[cBetPerpPerp1m, PrintAs → "β̅⊥⊥1-"];
DefConstantSymbol[cBetPerpPerp2p, PrintAs → "β̅⊥⊥2+"];
DefConstantSymbol[cBetPerpPerp2m, PrintAs → "β̅⊥⊥2-"];
```

```
cBetPerpPerp = {cBetPerpPerp0p, cBetPerpPerp0m,
   cBetPerpPerp1p, cBetPerpPerp1m, cBetPerpPerp2p, cBetPerpPerp2m};

DefConstantSymbol[cBetPerpPara0p, PrintAs → "β̅⊥∥0+"];
DefConstantSymbol[cBetPerpPara0m, PrintAs → "β̅⊥∥0-"];
DefConstantSymbol[cBetPerpPara1p, PrintAs → "β̅⊥∥1+"];
DefConstantSymbol[cBetPerpPara1m, PrintAs → "β̅⊥∥1-"];
DefConstantSymbol[cBetPerpPara2p, PrintAs → "β̅⊥∥2+"];
DefConstantSymbol[cBetPerpPara2m, PrintAs → "β̅⊥∥2-"];

cBetPerpPara = {cBetPerpPara0p, cBetPerpPara0m,
   cBetPerpPara1p, cBetPerpPara1m, cBetPerpPara2p, cBetPerpPara2m};

DefConstantSymbol[cBetParaPerp0p, PrintAs → "β̅∥⊥0+"];
DefConstantSymbol[cBetParaPerp0m, PrintAs → "β̅∥⊥0-"];
DefConstantSymbol[cBetParaPerp1p, PrintAs → "β̅∥⊥1+"];
DefConstantSymbol[cBetParaPerp1m, PrintAs → "β̅∥⊥1-"];
DefConstantSymbol[cBetParaPerp2p, PrintAs → "β̅∥⊥2+"];
DefConstantSymbol[cBetParaPerp2m, PrintAs → "β̅∥⊥2-"];

cBetParaPerp = {cBetParaPerp0p, cBetParaPerp0m,
   cBetParaPerp1p, cBetParaPerp1m, cBetParaPerp2p, cBetParaPerp2m};
```

build

Quadratic couplings $\hat{\alpha}_0$, $\{\hat{\alpha}_I\}$, $\{\hat{\beta}_M\}$

build

In[211]:=
```
(*Mike's couplings for irrep Lorentz constraints*)
DefConstantSymbol[mAlp0, PrintAs → "α0"];
DefConstantSymbol[mAlp1, PrintAs → "α1"];
DefConstantSymbol[mAlp2, PrintAs → "α2"];
DefConstantSymbol[mAlp3, PrintAs → "α3"];
DefConstantSymbol[mAlp4, PrintAs → "α4"];
DefConstantSymbol[mAlp5, PrintAs → "α5"];
DefConstantSymbol[mAlp6, PrintAs → "α6"];

mAlp = {mAlp1, mAlp2, mAlp3, mAlp4, mAlp5, mAlp6};

(*My couplings for irrep Lorentz constraints*)
DefConstantSymbol[Alp0, PrintAs → "α̂0"];
DefConstantSymbol[Alp1, PrintAs → "α̂1"];
DefConstantSymbol[Alp2, PrintAs → "α̂2"];
DefConstantSymbol[Alp3, PrintAs → "α̂3"];
DefConstantSymbol[Alp4, PrintAs → "α̂4"];
DefConstantSymbol[Alp5, PrintAs → "α̂5"];
DefConstantSymbol[Alp6, PrintAs → "α̂6"];

Alp = {Alp1, Alp2, Alp3, Alp4, Alp5, Alp6};

DefConstantSymbol[mBet1, PrintAs → "β1"];
DefConstantSymbol[mBet2, PrintAs → "β2"];
DefConstantSymbol[mBet3, PrintAs → "β3"];
DefConstantSymbol[mBet4, PrintAs → "β4"];
DefConstantSymbol[mBet5, PrintAs → "β5"];
DefConstantSymbol[mBet6, PrintAs → "β6"];

mBet = {mBet1, mBet2, mBet3};

DefConstantSymbol[Bet1, PrintAs → "β̂1"];
DefConstantSymbol[Bet2, PrintAs → "β̂2"];
DefConstantSymbol[Bet3, PrintAs → "β̂3"];
DefConstantSymbol[Bet4, PrintAs → "β̂4"];
DefConstantSymbol[Bet5, PrintAs → "β̂5"];
DefConstantSymbol[Bet6, PrintAs → "β̂6"];

Bet = {Bet1, Bet2, Bet3};
```

build

ORPHAN

build

Null

build

In[241]:=

```
(*
(*Define dimensionless coupling constants and reduced Planck mass*)
DefNiceConstantSymbol[m,P];
(*
ToExpression[Import[NotebookDirectory[]<>"new_cases_definitions.txt"]];
Theory=SuperTheory;
*)
(*
Print[Style["Will's geometric Lagrangian",Blue,16]]
  WillGLagrangian=gAlp1 Rs[]^2+gAlp2 Rc[-a,-b]Rc[a,b]+
  gAlp3 Rc[-a,-b]Rc[b,a]+(1/2) (gAlp4-gAlp5) R[-a,-b,-c,-d]R[a,b,c,d]+
  gAlp5 R[-a,-b,-c,-d]R[a,c,b,d]+(1/2)gAlp6 R[-a,-b,-c,-d]R[c,d,a,b]+
  mP^2((-1/2) (gBet1+gBet2) T[-a,-b,-c]T[a,b,c]+
     gBet2 T[-a,-b,-c]T[b,a,c]+gBet3 Tc[-a]Tc[a]);
%//ToCanonical;
%/.ExpandContractedStrengths//ToCanonical//NoScalar;
WillGLagrangian=%;
*)
(*
Print[Style["Will's geometric Lagrangian",Blue,16]]
 WillGLLagrangian=gAlp1 RLambda[-k,-l,k,l] Rs[]+
   gAlp2 RLambda[-k,-a,k,-b]Rc[a,b]+gAlp3 RLambda[-k,-a,k,-b]Rc[b,a]+
   (1/2) (gAlp4-gAlp5) RLambda[-a,-b,-c,-d]R[a,b,c,d]+
   gAlp5 RLambda[-a,-b,-c,-d]R[a,c,b,d]+(1/2)gAlp6 RLambda[-a,-b,-c,-d]R[c,d,a,b]+
   mP^2((-1/2) (gBet1+gBet2) TLambda[-a,-b,-c]T[a,b,c]+
      gBet2 TLambda[-a,-b,-c]T[b,a,c]+gBet3 TLambda[k,-a,-k]Tc[a]);
%//ToCanonical;
%/.ExpandContractedStrengths//ToCanonical//NoScalar;
WillGLLagrangian=%;
*)
(**)
Print[Style["Mike's Lagrangian",Blue,16]]
 MikeLagrangian=mAlp1 Rs[]^2+mAlp2 Rc[-a,-b]Rc[a,b]+
   mAlp3 Rc[-a,-b]Rc[b,a]+mAlp4 R[-a,-b,-c,-d]R[a,b,c,d]+
   mAlp5 R[-a,-b,-c,-d]R[a,c,b,d]+mAlp6 R[-a,-b,-c,-d]R[c,d,a,b]+
   mP^2(mBet1 T[-a,-b,-c]T[a,b,c]+mBet2 T[-a,-b,-c]T[b,a,c]+mBet3 Tc[-a]Tc[a]);
%//ToCanonical;
```

```
%/.ExpandContractedStrengths//ToCanonical//NoScalar;
MikeLagrangian=%;
(**)
(*
KNLagrangian=
 mP^2(c1 (1/2)T[-k,-l,-m]T[k,l,m]+c2 (3/4)Antisymmetrize[T[-k,-l,-m],{-k,-l,-m}]
     Antisymmetrize[T[k,l,m],{k,l,m}]+
    c3 (1/2)(T[-k,-l,-m]T[k,l,m]-2Tc[-k]Tc[k]));
%//ToCanonical;
%/.ExpandContractedStrengths//ToCanonical//NoScalar
  KNLagrangian=%;
*)
(**)
Print[Style["Will's Lagrangian",Blue,16]]
 WillLagrangian=
  R[i,k,l,m](Alp1 PR1[-i,-k,-l,-m,a,b,c,d]+Alp2 PR2[-i,-k,-l,-m,a,b,c,d]+
      Alp3 PR3[-i,-k,-l,-m,a,b,c,d]+Alp4 PR4[-i,-k,-l,-m,a,b,c,d]+
      Alp5 PR5[-i,-k,-l,-m,a,b,c,d]+Alp6 PR6[-i,-k,-l,-m,a,b,c,d])R[-a,-b,-c,-d]+
   mP^2 T[i,k,l](Bet1 PT1[-i,-k,-l,a,b,c]+Bet2 PT2[-i,-k,-l,a,b,c]+
      Bet3 PT3[-i,-k,-l,a,b,c])T[-a,-b,-c];
%/.PActivate;
(*
%/.ContractExpandedStrengths;
%//ToNewCanonical
  %/.ContractExpandedStrengths;
*)
%//ToNewCanonical;
WillLagrangian=%;
(*
Print[Style["Will's Lagrangian",Blue,16]]
 WillLLagrangian=
  RLambda[i,k,l,m](cAlp1 PR1[-i,-k,-l,-m,a,b,c,d]+cAlp2 PR2[-i,-k,-l,-m,a,b,c,d]+
      cAlp3 PR3[-i,-k,-l,-m,a,b,c,d]+cAlp4 PR4[-i,-k,-l,-m,a,b,c,d]+
      cAlp5 PR5[-i,-k,-l,-m,a,b,c,d]+cAlp6 PR6[-i,-k,-l,-m,a,b,c,d])
    R[-a,-b,-c,-d]+mP^2 TLambda[i,k,l](cBet1 PT1[-i,-k,-l,a,b,c]+
      cBet2 PT2[-i,-k,-l,a,b,c]+cBet3 PT3[-i,-k,-l,a,b,c])T[-a,-b,-c];
%/.PActivate;
%/.ContractExpandedStrengths;
%//ToNewCanonical
  %/.ContractExpandedStrengths;
WillLLagrangian=%;
*)
Print[WillLagrangian];
```

```
Print[MikeLagrangian];
(*
Print[WillLLagrangian];
*)
(*
DefConstantSymbol[CosCon,PrintAs→"Λ"];
DefConstantSymbol[CGCoup,PrintAs→"α_CG"];
DefConstantSymbol[NormCGCoup,PrintAs→"(1/18)α_CG"];
DefConstantSymbol[GBCoup,PrintAs→"α_GB"];

GaussBonnetTerm=GBCoup(Rs[]Rs[]-4Rc[-a,-b]Rc[b,a]+R[-a,-b,-c,-d]R[c,d,a,b]);

BasicTheory={Alp0→0,Alp1→-2Alp6,Alp2→2Alp6,Alp3→Alp3,Alp4→-Alp6,
  Alp5→Alp5,Alp6→Alp6,Bet1→0,Bet2→Bet2,Bet3→Bet3,cAlp1→0,cAlp2→0,
  cAlp3→0,cAlp4→0,cAlp5→0,cAlp6→0,cBet1→cBet1,cBet2→0,cBet3→0};
PhenomTheory={Alp3→0,Bet3→-Alp6 CosCon/(2 mP^2),Bet2→-2/3};
MultipTheory={cBet1→3/2,Alp5→(7/4) (-(1/3)CGCoup),Alp6→-(1/3)CGCoup};
MultipTheory2={cBet1→0};
GBChoice={GBCoup→0};
NormCG={CGCoup→18NormCGCoup};

Print[Style["Full theory",Blue,16]]
 TotalLagrangian=WillLagrangian+WillLLagrangian+GaussBonnetTerm;
%/.BasicTheory;
%/.PhenomTheory;
%/.MultipTheory2;
(*%/.GBChoice;*)
(*%/.NormCG;*)
%//ToNewCanonical;
%//CollectTensors;
TotalLagrangian=%;
Print[TotalLagrangian];

Print[Style["Bypass theory",Blue,16]]
 TotalLagrangian=WillLagrangian+WillLLagrangian+GaussBonnetTerm;
%/.BasicTheory;
%/.PhenomTheory;
%/.MultipTheory;
%/.GBChoice;
%/.NormCG;
%//ToNewCanonical;
%//CollectTensors;
TotalLagrangian=%;
```

```
Print[TotalLagrangian];

(*Quit[];*)
*)
(**)
Print[Style["Will Mike Coeffs",Blue,16]]
 MikeLagrangian-WillLagrangian//ContractMetric;
%//ToCanonical;
%//Simplify;
%//FullSimplify;
%//CollectTensors
   equations=ToConstantSymbolEquations[%==0]
  Print[equations];
tmp=Solve[equations,{Alp1,Alp2,Alp3,Alp4,Alp5,Alp6,Bet1,Bet2,Bet3}]
  Print[tmp];
tmpx=Solve[equations,{mAlp1,mAlp2,mAlp3,mAlp4,mAlp5,mAlp6,mBet1,mBet2,mBet3}]
  Print[tmpx];

Print[Style["Apply ",Blue,16]]
 DefNiceConstantSymbol[r,#,Y]&/@Range[6];
DefNiceConstantSymbol[t,#,Y]&/@Range[3];
(*equations2=mAlp1==(r4Y-r5Y)/4&&mAlp2==r4Y+r5Y&&mAlp3==r6Y&&mAlp4==(2r1Y+r2Y)/6&&
   mAlp5==(2/3) (r1Y-r2Y)&&mAlp6==(4r1Y+2r2Y-12r3Y+3r4Y-3r5Y)/12&&
   mBet1==(4t1Y+t2Y)/12&&mBet2==(2t1Y-t2Y)/6&&mBet3==-(t1Y-2t3Y)/3;*)
equations2=mAlp1==r6Y&&mAlp2==r4Y+r5Y&&mAlp3==r4Y-r5Y&&
  mAlp4==r1Y/3+r2Y/6&&mAlp5==2r1Y/3-2r2Y/3&&mAlp6==r1Y/3+r2Y/6-r3Y&&
  mBet1==(4t1Y+t2Y)/12&&mBet2==(2t1Y-t2Y)/6&&mBet3==-(t1Y-2t3Y)/3;
equations3=r1Y==0&&t1Y==0&&r3Y-2r4Y==0&&r6Y==0;
equations6=(3/2)mAlp1+(1/4)mAlp2+(1/4)mAlp3+(1/4)mAlp5-(1/2)mAlp6==σ&&
  (3/2)mAlp1+(1/2)mAlp2+(1/2)mAlp3+(3/2)mAlp4-(1/4)mAlp5+(1/2)mAlp6==σ &&
  2mBet1+mBet2+3mBet3==-(4/3)&&-2mBet1+2mBet2==σ λ;
equations7=(3/2)mAlp1+(1/4)mAlp2+(1/4)mAlp3+(1/4)mAlp5-(1/2)mAlp6==-1/3&&
  (3/2)mAlp1+(1/2)mAlp2+(1/2)mAlp3+(3/2)mAlp4-(1/4)mAlp5+(1/2)mAlp6==-1/3 ;
NonvanishingQuantities={r2Y,r1Y-r3Y,2r3Y+r5Y,
  r1Y+r3Y+2r5Y,t2Y,t3Y,r3Y(2r3Y+r5Y) (r3Y+2r5Y)};
(*where last quantity is the ghost condition*)
Print[equations];
Print[equations2];
Print[equations3];
equations4=Join[equations,equations2];
(*
Print[Style["Crv",Blue,16]]
  tmp=Reduce[Join[equations2,equations3,equations7]];
```

```
Print[tmp];
tmp=Eliminate[tmp,{mAlp1,mAlp2,mAlp3,mAlp4,mAlp5,mAlp6,mBet1,mBet2,mBet3}];
Print[tmp];
Quit[];
*)

Print[Style["Apply ",Blue,16]]
 Print[equations4];
equations5=Evaluate[equations2/.tmpx[[1]]];
Print[equations5];
lookat=Solve[equations5,{Alp1,Alp2,Alp3,Alp4,Alp5,Alp6,Bet1,Bet2,Bet3}];
Print[lookat];
looket=Solve[equations5,{r1Y,r2Y,r3Y,r4Y,r5Y,r6Y,t1Y,t2Y,t3Y}];
Print[looket];
lookup=
 Solve[equations2,{mAlp1,mAlp2,mAlp3,mAlp4,mAlp5,mAlp6,mBet1,mBet2,mBet3}];
Print[lookup];
lookop=Solve[equations2,{r1Y,r2Y,r3Y,r4Y,r5Y,r6Y,t1Y,t2Y,t3Y}];
Print[lookop];

Print["direct will"];
(*
tmpl=lookat/.{r1Y→0,t1Y→0,r3Y→2r4Y,r6Y→0};
Print[tmpl];
*)
tmp=Reduce[Join[equations5,equations3]];
Print[tmp];
tmp=Eliminate[tmp,{r1Y,r2Y,r3Y,r4Y,r5Y,r6Y,t1Y,t2Y,t3Y}];
Print[tmp];

ruleos=Quiet[Solve[tmp,{Alp1,Alp2,Alp3,Alp4,Alp5,Alp6,Bet1,Bet2,Bet3}]];
Print[ruleos];
*)





(*
EqsCase1=r1Y==0&&r3Y/2-r4Y==0&&t1Y==0&&t3Y==0;
EqsCase2=r1Y==0&&r3Y/2-r4Y==0&&t1Y==0;
EqsCase3=r1Y==0&&r3Y==0&&r4Y==0&&t1Y+t2Y==0&&t3Y==0;
```

```
EqsCase4=r2Y==0&&r1Y-r3Y==0&&r4Y==0&&t1Y+t2Y==0&&t3Y==0;
EqsCase5=r2Y==0&&r1Y-r3Y==0&&r4Y==0&&t2Y==0&&t1Y+t3Y==0;
EqsCase6=r1Y==0&&2r3Y-r4Y==0&&t1Y+t2Y==0&&t3Y==0;
EqsCase7=r2Y==0&&2r1Y-2r3Y+r4Y==0&&t1Y+t2Y==0&&t3Y==0;

EqsCase8=r2Y==0&&r1Y-r3Y==0&&r4Y==0&&t1Y==0&&t2Y==0;
EqsCase9=r2Y==0&&r1Y-r3Y==0&&r4Y==0&&t1Y==0&&t2Y==0&&t3Y==0;
EqsCase10=r1Y==0&&r2Y==0&&r3Y/2-r4Y==0&&t1Y==0&&t2Y==0&&t3Y==0;
EqsCase11=r1Y==0&&r3Y/2-r4Y==0&&t1Y==0&&t2Y==0&&t3Y==0;
EqsCase12=r1Y==0&&r2Y==0&&r3Y/2-r4Y==0&&t1Y==0&&t3Y==0;
EqsCase13=r2Y==0&&2r1Y-2r3Y+r4Y==0&&t1Y==0&&t2Y==0&&t3Y==0;
EqsCase14=r1Y==0&&r2Y==0&&r3Y/2-r4Y==0&&t1Y==0&&t2Y==0;
EqsCase15=r1Y==0&&r2Y==0&&r3Y/2-r4Y==0&&t1Y==0;
EqsCase16=r1Y==0&&r3Y/2-r4Y==0&&t1Y==0&&t2Y==0;
EqsCase17=r1Y==0&&r2Y==0&&r3Y==0&&r4Y==0&&t1Y+t2Y==0&&t3Y==0;
EqsCase18=r1Y==0&&r2Y==0&&r3Y==0&&r4Y==0&&t2Y==0&&t1Y+t3Y==0;
EqsCase19=r1Y==0&&r2Y==0&&2r3Y-r4Y==0&&t1Y+t2Y==0&&t3Y==0;

GhoCase1=t2Y>0&&r2Y<0&&r3Y(2r3Y+r5Y) (r3Y+2r5Y)<0;
GhoCase2=t2Y>0&&r2Y<0&&r3Y(2r3Y+r5Y) (r3Y+2r5Y)<0;
GhoCase3=r2Y<0&&r5Y<0&&t1Y<0;
GhoCase4=t1Y>0&&r1Y+r5Y<0&&r1Y<0;
GhoCase5=r5Y>0&&2r1Y+r5Y>0&&t1Y>0&&r1Y<0;
GhoCase6=r2Y<0&&2r3Y+r5Y<0&&t1Y<0;
GhoCase7=t1Y>0&&r1Y<0&&2r3Y+r5Y<r1Y

    GhoCase8=r1Y(r1Y+r5Y) (2r1Y+r5Y)<0;
GhoCase9=r1Y(r1Y+r5Y) (2r1Y+r5Y)<0;
GhoCase10=r3Y(2r3Y+r5Y) (r3Y+2r5Y)<0;
GhoCase11=r3Y(2r3Y+r5Y) (r3Y+2r5Y)<0;
GhoCase12=r3Y(2r3Y+r5Y) (r3Y+2r5Y)<0;
GhoCase13=r1Y(r1Y-2r3Y-r5Y) (2r3Y+r5Y)>0;
GhoCase14=r3Y(2r3Y+r5Y) (r3Y+2r5Y)<0;
GhoCase15=r3Y(2r3Y+r5Y) (r3Y+2r5Y)<0;
GhoCase16=r3Y(2r3Y+r5Y) (r3Y+2r5Y)<0;
GhoCase17=r5Y<0;
GhoCase18=r5Y>0;
GhoCase19=r3Y<-r5Y/2;

CasesTot={};
CasesGho={};
For[ii=1,ii<20,ii++,
 Print[ii];
```

```
 tmp=Reduce[Join[equations5,Evaluate[ToExpression["EqsCase"<>ToString[ii]]]]];
 tmp=Eliminate[tmp,{r1Y,r2Y,r3Y,r4Y,r5Y,r6Y,t1Y,t2Y,t3Y}];
 tmp=Quiet[Solve[tmp,{Alp1,Alp2,Alp3,Alp4,Alp5,Alp6,Bet1,Bet2,Bet3}]][[1]];
 tmp=Join[tmp,{Alp0→0,cAlp1→0,cAlp2→0,
    cAlp3→0,cAlp4→0,cAlp5→0,cAlp6→0,cBet1→0,cBet2→0,cBet3→0}];
 Print[tmp];
 CasesTot=Append[CasesTot,tmp];
 tmp2=Evaluate[ToExpression["GhoCase"<>ToString[ii]]]/.looket;
 tmp2=tmp2/.tmp;
 Print[tmp2];
 CasesGho=Append[CasesGho,tmp2];
];

TheoryCaseNew1=CasesTot[[1]];
TheoryCaseNew2=CasesTot[[2]];
TheoryCaseNew3=CasesTot[[3]];
TheoryCaseNew4=CasesTot[[4]];
TheoryCaseNew5=CasesTot[[5]];
TheoryCaseNew6=CasesTot[[6]];
TheoryCaseNew7=CasesTot[[7]];
TheoryCaseNew8=CasesTot[[8]];
TheoryCaseNew9=CasesTot[[9]];
TheoryCaseNew10=CasesTot[[10]];
TheoryCaseNew11=CasesTot[[11]];
TheoryCaseNew12=CasesTot[[12]];
TheoryCaseNew13=CasesTot[[13]];
TheoryCaseNew14=CasesTot[[14]];
TheoryCaseNew15=CasesTot[[15]];
TheoryCaseNew16=CasesTot[[16]];
TheoryCaseNew17=CasesTot[[17]];
TheoryCaseNew18=CasesTot[[18]];
TheoryCaseNew19=CasesTot[[19]];

TheoryCaseGho1=CasesGho[[1]];
TheoryCaseGho2=CasesGho[[2]];
TheoryCaseGho3=CasesGho[[3]];
TheoryCaseGho4=CasesGho[[4]];
TheoryCaseGho5=CasesGho[[5]];
TheoryCaseGho6=CasesGho[[6]];
TheoryCaseGho7=CasesGho[[7]];
TheoryCaseGho8=CasesGho[[8]];
TheoryCaseGho9=CasesGho[[9]];
TheoryCaseGho10=CasesGho[[10]];
TheoryCaseGho11=CasesGho[[11]];
```

```
TheoryCaseGho12=CasesGho[[12]];
TheoryCaseGho13=CasesGho[[13]];
TheoryCaseGho14=CasesGho[[14]];
TheoryCaseGho15=CasesGho[[15]];
TheoryCaseGho16=CasesGho[[16]];
TheoryCaseGho17=CasesGho[[17]];
TheoryCaseGho18=CasesGho[[18]];
TheoryCaseGho19=CasesGho[[19]];

Print["cycling"];
Print[TheoryCaseNew1];
Print[TheoryCaseNew2];
Print[TheoryCaseNew3];
Print[TheoryCaseNew4];
Print[TheoryCaseNew5];
Print[TheoryCaseNew6];
Print[TheoryCaseNew7];
Print[TheoryCaseNew8];
Print[TheoryCaseNew9];
Print[TheoryCaseNew10];
Print[TheoryCaseNew11];
Print[TheoryCaseNew12];
Print[TheoryCaseNew13];
Print[TheoryCaseNew14];
Print[TheoryCaseNew15];
Print[TheoryCaseNew16];
Print[TheoryCaseNew17];
Print[TheoryCaseNew18];
Print[TheoryCaseNew19];
*)
```

```
(*
Print["direct mike"];
tmpl=lookup/.{r1Y→0,t1Y→0,r3Y→2r4Y,r6Y→0};
Print[tmpl];
tmp=Reduce[Join[equations2,equations3,equations6]];
Print[tmp];
tmp=Eliminate[tmp,{r1Y,r2Y,r3Y,r4Y,r5Y,r6Y,t1Y,t2Y,t3Y}];
Print[tmp];
```

```
generals=
 Quiet[Solve[tmp,{mAlp1,mAlp2,mAlp3,mAlp4,mAlp5,mAlp6,mBet1,mBet2,mBet3}]][[1]];
Print[generals];
NonvanishingQuantities=NonvanishingQuantities/.lookop[[1]];
Print[NonvanishingQuantities];
NonvanishingQuantities=NonvanishingQuantities/.generals;
Print[NonvanishingQuantities];

Quit[];
*)
(**)
(*
Print[Style["Comparing geometric",Blue,16]]
   WillGLagrangian-WillLagrangian//ContractMetric;
%//ToCanonical;
%//Simplify;
%//FullSimplify;
%//CollectTensors
   equations=ToConstantSymbolEquations[%==0]
   tmp=Solve[equations,{Alp1,Alp2,Alp3,Alp4,Alp5,Alp6,Bet1,Bet2,Bet3}]
   Print[tmp];
tmp=Solve[equations,{gAlp1,gAlp2,gAlp3,gAlp4,gAlp5,gAlp6,gBet1,gBet2,gBet3}]
  Print[tmp];

Print[Style["Comparing L geometric",Blue,16]]
   WillGLLagrangian-WillLLagrangian//ContractMetric;
%//ToCanonical;
%//Simplify;
%//FullSimplify;
%//CollectTensors
   equations=ToConstantSymbolEquations[%==0]
   tmp=Solve[equations,Join[cAlp,cBet]]
   Print[tmp];
ToGA=tmp[[1]];
tmp=Solve[equations,Join[gAlp,gBet]]
  Print[tmp];

Quit[];

DumpSave[NotebookDirectory[]<>"mx_cache/coordinate_transforms.mx",{ToGA}];
Print["done"];
Quit[];
*)
(*
```

```
MyImport["coordinate_transforms.mx"];
*)
(*
Print[Style["Comparing with KN",Blue,16]]
   KNLagrangian-WillLagrangian//ContractMetric;
%//ToCanonical;
%//Simplify;
%//FullSimplify;
%//CollectTensors
   equations=ToConstantSymbolEquations[%==0]
   KNToWill=Solve[equations,{c1,c2,c3,Alp1,Alp2,Alp3,Alp4,Alp5,Alp6}]
      c1/.KNToWill[[1]]//ToCanonical
       c2+c3/.KNToWill[[1]]//ToCanonical
*)
```

build

ORPHAN

build

```
In[242]:= (*
BPiCPiC={
  {{Un},{Un},{Un},{Un},{Un},{Un}},
  {{A1p,R1p},{Un},{Un},{Un},{Un},{A0m,R0m,A1m,R1m,R2m}},
  {{Un},{Un},{Un},{Un},{Un},{Un}},
  {{A2p,R2p},{Un},{Un},{Un},{Un},{A0m,R0m,A1m,R1m,R2m}}};

PiCPiC={
  {{Un},{Un},{Un},{Un},{Un},{T2m}},
  {{Un},{Un},{Un},{Un},{Un},{Un}},
  {{Un},{Un},{Un},{Un},{Un},{Un}},
  {{Un},{Un},{Un},{Un},{Un},{Un}},
  {{Un},{Un},{Un},{Un},{Un},{Un}},
  {{Un},{Un},{Un},{Un},{Un},{T1p}}};

BPiCParaSLiC={
  {{Un},{Un},{Un},{Un},{Un},{Un}},
  {{R1p},{R1m,R2m},{R0p,R1p,R2p},{R0m,R1m,R2m},{R1p,R2p},{R0m,R1m,R2m}},
  {{Un},{Un},{Un},{Un},{Un},{Un}},
  {{R2p},{R2m},{R1p,R2p},{R1m,R2m},{R0p,R1p,R2p},{R0m,R1m,R2m}}};

PiCParaSLiC={
  {{Dn},{T0m},{T1p},{Dx},{Dn},{T2m}},
  {{Un},{Un},{Un},{Un},{Un},{Un}},
  {{Un},{Un},{Un},{Un},{Un},{Un}},
  {{Un},{Un},{Un},{Un},{Un},{Un}},
```

```
  {{Un},{Un},{Un},{Un},{Un},{Un}},
  {{T2m},{Dn,T1p},{Dx},{T1p},{Dx},{Dn}}};

BPiCPerpSLiC={
  {{Un},{Un},{Un},{Un},{Un},{Un}},
  {{A1p},{A1m,A2m},{A0p,A1p,A2p},{A0m,A1m,A2m},{A1p,A2p},{A0m,A1m,A2m}},
  {{Un},{Un},{Un},{Un},{Un},{Un}},
  {{A2p},{A2m},{A1p,A2p},{A1m,A2m},{A0p,A1p,A2p},{A0m,A1m,A2m}}};

PiCPerpSLiC={
  {{Un},{T0m},{T1p},{Dx},{Dn},{T2m}},
  {{Un},{Un},{Un},{Un},{Un},{Un}},
  {{Un},{Un},{Un},{Un},{Un},{Un}},
  {{Un},{Un},{Un},{Un},{Un},{Un}},
  {{Un},{Un},{Un},{Un},{Un},{Un}},
  {{T2m},{Dn},{Dx},{T1p},{Dx},{Un}}};

ParaSLiCPerpSLiC={
  {{Dn},{T0m},{T1p},{Dx},{Dn},{T2m}},
  {{T0m},{Dn},{Dx},{T1p},{T2m},{Dn,T1p}},
  {{T1p},{Dx},{Dn},{Dx},{T1p},{Dx}},
  {{Dx},{T1p},{Dx},{Dn},{Dx},{T1p}},
  {{Dn},{T2m},{T1p},{Dx},{Dn},{Dx}},
  {{T2m},{Dn},{Dx},{T1p},{Dx},{T1p,Dn}}};
*)
```

build

ORPHAN

build

In[243]:=
```
(*
Theory={Alp0→Alp0,Alp1→0,Alp2→0,Alp3→0,Alp4→0,Alp5→Alp5,Alp6→0,
   Bet1→0,Bet2→0,Bet3→Bet3,cAlp1→0,cAlp2→0,cAlp3→0,cAlp4→0,cAlp5→0,
   cAlp6→0,cBet1→0,cBet2→0,cBet3→0};*)(*This implements Spin 1⁺*)

(*Theory=Join[Theory,{Alp0→0,cAlp1→0,cAlp2→0,cAlp3→0,cAlp4→0,cAlp5→0,cAlp6→0,
    cBet1→cBet1,cBet2→0,cBet3→0}];*)(*This implements Case 16.1.2*)

(*Theory={Alp0→0,Alp1→-2Alp6,Alp2→2Alp6,Alp3→0,Alp4→-Alp6,
   Alp5→Alp5,Alp6→Alp6,Bet1→0,Bet2→Bet2,Bet3→Bet3,cAlp1→0,cAlp2→0,
   cAlp3→0,cAlp4→0,cAlp5→cAlp5,cAlp6→0,cBet1→cBet1,cBet2→0,cBet3→0};*)
(*This implements Case 16.6.2 with NO alp3*)

(*Theory={Alp0→0,Alp1→-2Alp6,Alp2→2Alp6,Alp3→Alp3,Alp4→-Alp6,
   Alp5→Alp5,Alp6→Alp6,Bet1→0,Bet2→Bet2,Bet3→0,cAlp1→0,cAlp2→0,
```

```
    cAlp3→0,cAlp4→0,cAlp5→0,cAlp6→0,cBet1→cBet1,cBet2→0,cBet3→0};*)
(*This implements Case 16.6.2 WITH alp3*)
(*Theory={Alp0→0,Alp1→-2Alp6,Alp2→2Alp6,Alp3→Alp3,Alp4→-Alp6,
    Alp5→Alp5,Alp6→Alp6,Bet1→0,Bet2→Bet2,Bet3→Bet3,cAlp1→0,cAlp2→0,
    cAlp3→0,cAlp4→0,cAlp5→0,cAlp6→0,cBet1→cBet1,cBet2→0,cBet3→0};*)
(*This implements Case 16.6.2 WITH alp3*)

(*Theory=Join[Theory,{Alp0→0,cAlp1→0,cAlp2→0,cAlp3→0,cAlp4→0,cAlp5→cAlp5,
     cAlp6→0,cBet1→0,cBet2→0,cBet3→0}];*)(*This implements Case 16.6.1*)

(*Theory=Join[Theory,{Alp0→0,cAlp1→0,cAlp2→0,cAlp3→0,cAlp4→0,cAlp5→0,cAlp6→0,
     cBet1→0,cBet2→0,cBet3→0}];*)(*This implements Case 16.1.1*)

(*Theory={Alp0→0,Alp1→0,Alp2→0,Alp3→Alp3,Alp4→0,Alp5→0,Alp6→0,
    Bet1→0,Bet2→0,Bet3→Bet3,cAlp1→0,cAlp2→0,cAlp3→0,cAlp4→0,cAlp5→0,
    cAlp6→0,cBet1→0,cBet2→0,cBet3→0};*)(*This implements Case 26*)

(*Theory={Alp0→0,Alp1→0,Alp2→0,Alp3→0,Alp4→0,Alp5→Alp5,Alp6→0,
    Bet1→-2Bet3,Bet2→0,Bet3→Bet3,cAlp1→0,cAlp2→0,cAlp3→0,cAlp4→0,cAlp5→0,
    cAlp6→0,cBet1→0,cBet2→0,cBet3→0};*)(*This implements Case 17*)

(*Theory={Alp0→0,Alp1→0,Alp2→0,Alp3→Alp3,Alp4→0,Alp5→Alp5,Alp6→0,
    Bet1→0,Bet2→0,Bet3→Bet3,cAlp1→0,cAlp2→0,cAlp3→0,cAlp4→0,cAlp5→0,
    cAlp6→0,cBet1→0,cBet2→0,cBet3→0};*)(*This implements Case 28*)

(*Theory={Alp0→0,Alp1→0,Alp2→0,Alp3→Alp3,Alp4→0,Alp5→0,Alp6→0,
    Bet1→Bet1,Bet2→0,Bet3→Bet3,cAlp1→0,cAlp2→0,cAlp3→0,cAlp4→0,cAlp5→0,
    cAlp6→0,cBet1→0,cBet2→0,cBet3→0};*)(*This implements Case 32*)
(*
Theory=TheoryCaseNew19;

(*AlpOriginalConstraintStructure={1,1,0,0,1,1};
BetOriginalConstraintStructure={1,0,0,1,1,0};*)(*When checking Simple 1+*)
(*AlpOriginalConstraintStructure={1,0,0,0,0,1};
BetOriginalConstraintStructure={0,0,1,0,1,0};*)(*When checking Case 16*)
(*AlpOriginalConstraintStructure={1,0,1,1,1,1};
BetOriginalConstraintStructure={1,0,0,1,1,0};*)(*When checking Case 26*)
(*AlpOriginalConstraintStructure={1,1,0,0,1,1};
BetOriginalConstraintStructure={1,0,1,0,0,0};*)(*When checking Case 17*)
(*AlpOriginalConstraintStructure={1,0,0,0,1,1};
BetOriginalConstraintStructure={1,0,0,1,1,0};*)(*When checking Case 28*)
(**)AlpOriginalConstraintStructure={1,0,1,1,1,1};
BetOriginalConstraintStructure={1,0,0,0,0,0};
(**)(*When checking Case 32*)
```

```
AMultiplicities={1,1,3,3,5,5};
Mul=AMultiplicities;

(*Here are the generalised freedom coefficients*)
DefNiceConstantSymbol[ShellPara,ToExpression[#]]&/@ASectorNames;
DefNiceConstantSymbol[ShellOrig,ToExpression[#]]&/@ASectorNames;
DefNiceConstantSymbol[ShellPerp,ToExpression[#]]&/@ASectorNames;
DefNiceConstantSymbol[ShellSing,ToExpression[#]]&/@ASectorNames;
DefNiceConstantSymbol[ShellPrim,ToExpression[#]]&/@ASectorNames;
DefNiceConstantSymbol[ShellPara,ToExpression[#]]&/@BSectorNames;
DefNiceConstantSymbol[ShellOrig,ToExpression[#]]&/@BSectorNames;
DefNiceConstantSymbol[ShellPerp,ToExpression[#]]&/@BSectorNames;
DefNiceConstantSymbol[ShellSing,ToExpression[#]]&/@BSectorNames;
DefNiceConstantSymbol[ShellPrim,ToExpression[#]]&/@BSectorNames;

AlpSwitchBoard={cAlp1==0,cAlp2==0,cAlp3==0,cAlp4==0,cAlp5==0,cAlp6==0};
BetSwitchBoard={cBet1==0,cBet2==0,cBet3==0};
AlpLorentzCases=Subsets[AlpSwitchBoard];
BetLorentzCases=Subsets[BetSwitchBoard];
AlpLorentzCases=Reverse[AlpLorentzCases];
BetLorentzCases=Reverse[BetLorentzCases];
Print[AlpLorentzCases];
Print[BetLorentzCases];
Ls=Table[{Join[ii,jj]},{ii,AlpLorentzCases},{jj,BetLorentzCases}];
Nums=Table[{ToExpression[ToString[ii]<>"."<>ToString[jj]]},{ii,64},{jj,8}];
Combinat=Join[Nums,Ls,3];
targ=Flatten[Combinat,1];

OptUpd[Opts_,x_]:=Flatten[Opts+#&/@x];

AnaMas[conds_]:=Module[{AlpStructureLorentzCase,BetStructureLorentzCase,
    FinalStructureLorentzCase,AllConstraints,DoF,RiemannParts,X5,
    X6,LineW,LineA,B1pCommutators,gAlpLorentzCase,AlpLorentzCase,
    BetLorentzCase,cAlpPerpPerpLorentzCase,cAlpPerpParaLorentzCase,
    cAlpParaPerpLorentzCase,cAlpParaParaLorentzCase,AlpDeterminantsLorentzCase,
    BetDeterminantsLorentzCase,cBetPerpPerpLorentzCase,cBetPerpParaLorentzCase,
    cBetParaPerpLorentzCase,cBetParaParaLorentzCase,DoFSeed,fros,mestr,
    ShellParaFreedomsActivate,ShellOrigFreedomsActivate,ShellPerpFreedomsActivate,
    ShellSingFreedomsActivate,ShellPrimFreedomsActivate,ShellFreedomsActivate},
  AlpLorentzCase=Quiet[Solve[conds[[2]],Join[cAlp,cBet]][[1]]];
  gAlpLorentzCase=Quiet[Solve[conds[[2]]/.ToGA,Join[gAlp,gBet]]];
  cAlpPerpPerpLorentzCase=cAlpPerpPerp/.TocAlp/.AlpLorentzCase;
  cAlpPerpParaLorentzCase=cAlpPerpPara/.TocAlp/.AlpLorentzCase;
  cAlpParaPerpLorentzCase=cAlpParaPerp/.TocAlp/.AlpLorentzCase;
```

```
cAlpParaParaLorentzCase=cAlpParaPara/.TocAlp/.AlpLorentzCase;
cBetPerpPerpLorentzCase=cBetPerpPerp/.TocBet/.AlpLorentzCase;
cBetPerpParaLorentzCase=cBetPerpPara/.TocBet/.AlpLorentzCase;
cBetParaPerpLorentzCase=cBetParaPerp/.TocBet/.AlpLorentzCase;
cBetParaParaLorentzCase=cBetParaPara/.TocBet/.AlpLorentzCase;
ShellParaFreedomsActivate={};
ShellOrigFreedomsActivate={};
ShellPerpFreedomsActivate={};
ShellSingFreedomsActivate={};
ShellPrimFreedomsActivate={};
AlpStructureLorentzCase={};
BetStructureLorentzCase={};
For[ii=1,ii<7,ii++,
 If[cAlpPerpPerpLorentzCase[[ii]]≠0||cAlpPerpParaLorentzCase[[ii]]≠0,
  AlpStructureLorentzCase=Append[AlpStructureLorentzCase,1],
  AlpStructureLorentzCase=Append[AlpStructureLorentzCase,0],
  AlpStructureLorentzCase=Append[AlpStructureLorentzCase,1]]];
For[ii=1,ii<7,ii++,If[cBetPerpPerpLorentzCase[[ii]]≠0||
   cBetParaParaLorentzCase[[ii]]≠0||cBetPerpParaLorentzCase[[ii]]≠0,
  BetStructureLorentzCase=Append[BetStructureLorentzCase,1],
  BetStructureLorentzCase=Append[BetStructureLorentzCase,0],
  BetStructureLorentzCase=Append[BetStructureLorentzCase,1]]];
AlpDeterminantsLorentzCase=cAlpDeterminants/.TocAlp/.AlpLorentzCase;
BetDeterminantsLorentzCase=cBetDeterminants/.TocBet/.AlpLorentzCase;
For[ii=1,ii<7,ii++,
 If[AlpStructureLorentzCase[[ii]]≠0&&AlpDeterminantsLorentzCase[[ii]]==0,
  AlpStructureLorentzCase=ReplacePart[AlpStructureLorentzCase,ii→2]]];
For[ii=3,ii<5,ii++,If[BetStructureLorentzCase[[ii]]≠0&&
   BetDeterminantsLorentzCase[[ii]]==0,
  BetStructureLorentzCase=ReplacePart[BetStructureLorentzCase,ii→2]]];
For[ii=1,ii<7,ii++,If[AlpStructureLorentzCase[[ii]]==0&&
   AlpOriginalConstraintStructure[[ii]]==0,{
   AppendTo[ShellParaFreedomsActivate,Evaluate[
     ToExpression["ShellPara"<>ToString[ASectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellOrigFreedomsActivate,Evaluate[
     ToExpression["ShellOrig"<>ToString[ASectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellPerpFreedomsActivate,Evaluate[
     ToExpression["ShellPerp"<>ToString[ASectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellSingFreedomsActivate,Evaluate[
     ToExpression["ShellSing"<>ToString[ASectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellPrimFreedomsActivate,Evaluate[
     ToExpression["ShellPrim"<>ToString[ASectorNames[[ii]]]<>"->1"]]];
  }]];
For[ii=1,ii<7,ii++,
```

```
 If[AlpStructureLorentzCase[[ii]]==0&&AlpOriginalConstraintStructure[[ii]]==1,{
   AppendTo[ShellParaFreedomsActivate,
    Evaluate[ToExpression["ShellPara"<>ToString[ASectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellOrigFreedomsActivate,Evaluate[
     ToExpression["ShellOrig"<>ToString[ASectorNames[[ii]]]<>"->0"]]];
   AppendTo[ShellPerpFreedomsActivate,Evaluate[
     ToExpression["ShellPerp"<>ToString[ASectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellSingFreedomsActivate,Evaluate[
     ToExpression["ShellSing"<>ToString[ASectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellPrimFreedomsActivate,Evaluate[
     ToExpression["ShellPrim"<>ToString[ASectorNames[[ii]]]<>"->0"]]];
  }]];
For[ii=1,ii<7,ii++,
 If[AlpStructureLorentzCase[[ii]]==1&&AlpOriginalConstraintStructure[[ii]]==0,{
   AppendTo[ShellParaFreedomsActivate,
    Evaluate[ToExpression["ShellPara"<>ToString[ASectorNames[[ii]]]<>"->0"]]];
   AppendTo[ShellOrigFreedomsActivate,Evaluate[
     ToExpression["ShellOrig"<>ToString[ASectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellPerpFreedomsActivate,Evaluate[
     ToExpression["ShellPerp"<>ToString[ASectorNames[[ii]]]<>"->0"]]];
   AppendTo[ShellSingFreedomsActivate,Evaluate[
     ToExpression["ShellSing"<>ToString[ASectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellPrimFreedomsActivate,Evaluate[
     ToExpression["ShellPrim"<>ToString[ASectorNames[[ii]]]<>"->1"]]];
  }]];
For[ii=1,ii<7,ii++,
 If[AlpStructureLorentzCase[[ii]]==1&&AlpOriginalConstraintStructure[[ii]]==1,{
   AppendTo[ShellParaFreedomsActivate,
    Evaluate[ToExpression["ShellPara"<>ToString[ASectorNames[[ii]]]<>"->0"]]];
   AppendTo[ShellOrigFreedomsActivate,Evaluate[
     ToExpression["ShellOrig"<>ToString[ASectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellPerpFreedomsActivate,Evaluate[
     ToExpression["ShellPerp"<>ToString[ASectorNames[[ii]]]<>"->0"]]];
   AppendTo[ShellSingFreedomsActivate,Evaluate[
     ToExpression["ShellSing"<>ToString[ASectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellPrimFreedomsActivate,Evaluate[
     ToExpression["ShellPrim"<>ToString[ASectorNames[[ii]]]<>"->0"]]];
  }]];
For[ii=1,ii<7,ii++,
 If[AlpStructureLorentzCase[[ii]]==2&&AlpOriginalConstraintStructure[[ii]]==0,{
   AppendTo[ShellParaFreedomsActivate,
    Evaluate[ToExpression["ShellPara"<>ToString[ASectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellOrigFreedomsActivate,Evaluate[
     ToExpression["ShellOrig"<>ToString[ASectorNames[[ii]]]<>"->1"]]];
```

```
      AppendTo[ShellPerpFreedomsActivate,Evaluate[
        ToExpression["ShellPerp"<>ToString[ASectorNames[[ii]]]<>"->1"]]];
      AppendTo[ShellSingFreedomsActivate,Evaluate[
        ToExpression["ShellSing"<>ToString[ASectorNames[[ii]]]<>"->0"]]];
      AppendTo[ShellPrimFreedomsActivate,Evaluate[
        ToExpression["ShellPrim"<>ToString[ASectorNames[[ii]]]<>"->1"]]];
     }]];
   For[ii=1,ii<7,ii++,
    If[AlpStructureLorentzCase[[ii]]==2&&AlpOriginalConstraintStructure[[ii]]==1,{
      AppendTo[ShellParaFreedomsActivate,
       Evaluate[ToExpression["ShellPara"<>ToString[ASectorNames[[ii]]]<>"->1"]]];
      AppendTo[ShellOrigFreedomsActivate,Evaluate[
        ToExpression["ShellOrig"<>ToString[ASectorNames[[ii]]]<>"->0"]]];
      AppendTo[ShellPerpFreedomsActivate,Evaluate[
        ToExpression["ShellPerp"<>ToString[ASectorNames[[ii]]]<>"->1"]]];
      AppendTo[ShellSingFreedomsActivate,Evaluate[
        ToExpression["ShellSing"<>ToString[ASectorNames[[ii]]]<>"->1"]]];
      AppendTo[ShellPrimFreedomsActivate,Evaluate[
        ToExpression["ShellPrim"<>ToString[ASectorNames[[ii]]]<>"->0"]]];
     }]];
   For[ii=3,ii<5,ii++,
    If[BetStructureLorentzCase[[ii]]==0&&BetOriginalConstraintStructure[[ii]]==0,{
      AppendTo[ShellParaFreedomsActivate,
       Evaluate[ToExpression["ShellPara"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
      AppendTo[ShellOrigFreedomsActivate,Evaluate[
        ToExpression["ShellOrig"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
      AppendTo[ShellPerpFreedomsActivate,Evaluate[
        ToExpression["ShellPerp"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
      AppendTo[ShellSingFreedomsActivate,Evaluate[
        ToExpression["ShellSing"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
      AppendTo[ShellPrimFreedomsActivate,Evaluate[
        ToExpression["ShellPrim"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
     }]];
   For[ii=3,ii<5,ii++,
    If[BetStructureLorentzCase[[ii]]==0&&BetOriginalConstraintStructure[[ii]]==1,{
      AppendTo[ShellParaFreedomsActivate,
       Evaluate[ToExpression["ShellPara"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
      AppendTo[ShellOrigFreedomsActivate,Evaluate[
        ToExpression["ShellOrig"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
      AppendTo[ShellPerpFreedomsActivate,Evaluate[
        ToExpression["ShellPerp"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
      AppendTo[ShellSingFreedomsActivate,Evaluate[
        ToExpression["ShellSing"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
      AppendTo[ShellPrimFreedomsActivate,Evaluate[
```

```
      ToExpression["ShellPrim"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
   }]];
For[ii=3,ii<5,ii++,
 If[BetStructureLorentzCase[[ii]]==1&&BetOriginalConstraintStructure[[ii]]==0,{
    AppendTo[ShellParaFreedomsActivate,
     Evaluate[ToExpression["ShellPara"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
    AppendTo[ShellOrigFreedomsActivate,Evaluate[
      ToExpression["ShellOrig"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
    AppendTo[ShellPerpFreedomsActivate,Evaluate[
      ToExpression["ShellPerp"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
    AppendTo[ShellSingFreedomsActivate,Evaluate[
      ToExpression["ShellSing"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
    AppendTo[ShellPrimFreedomsActivate,Evaluate[
      ToExpression["ShellPrim"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
   }]];
For[ii=3,ii<5,ii++,
 If[BetStructureLorentzCase[[ii]]==1&&BetOriginalConstraintStructure[[ii]]==1,{
    AppendTo[ShellParaFreedomsActivate,
     Evaluate[ToExpression["ShellPara"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
    AppendTo[ShellOrigFreedomsActivate,Evaluate[
      ToExpression["ShellOrig"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
    AppendTo[ShellPerpFreedomsActivate,Evaluate[
      ToExpression["ShellPerp"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
    AppendTo[ShellSingFreedomsActivate,Evaluate[
      ToExpression["ShellSing"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
    AppendTo[ShellPrimFreedomsActivate,Evaluate[
      ToExpression["ShellPrim"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
   }]];
For[ii=3,ii<5,ii++,
 If[BetStructureLorentzCase[[ii]]==2&&BetOriginalConstraintStructure[[ii]]==0,{
    AppendTo[ShellParaFreedomsActivate,
     Evaluate[ToExpression["ShellPara"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
    AppendTo[ShellOrigFreedomsActivate,Evaluate[
      ToExpression["ShellOrig"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
    AppendTo[ShellPerpFreedomsActivate,Evaluate[
      ToExpression["ShellPerp"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
    AppendTo[ShellSingFreedomsActivate,Evaluate[
      ToExpression["ShellSing"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
    AppendTo[ShellPrimFreedomsActivate,Evaluate[
      ToExpression["ShellPrim"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
   }]];
For[ii=3,ii<5,ii++,
 If[BetStructureLorentzCase[[ii]]==2&&BetOriginalConstraintStructure[[ii]]==1,{
    AppendTo[ShellParaFreedomsActivate,
```

```
    Evaluate[ToExpression["ShellPara"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellOrigFreedomsActivate,Evaluate[
     ToExpression["ShellOrig"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
   AppendTo[ShellPerpFreedomsActivate,Evaluate[
     ToExpression["ShellPerp"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellSingFreedomsActivate,Evaluate[
     ToExpression["ShellSing"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellPrimFreedomsActivate,Evaluate[
     ToExpression["ShellPrim"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
  }]];
Stranges={1,5};
Do[
 If[BetStructureLorentzCase[[ii]]==0&&BetOriginalConstraintStructure[[ii]]==0,{
   AppendTo[ShellParaFreedomsActivate,
    Evaluate[ToExpression["ShellPara"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellOrigFreedomsActivate,Evaluate[
     ToExpression["ShellOrig"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellPerpFreedomsActivate,Evaluate[
     ToExpression["ShellPerp"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellPrimFreedomsActivate,Evaluate[
     ToExpression["ShellPrim"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
  }],{ii,Stranges}];
Do[
 If[BetStructureLorentzCase[[ii]]==0&&BetOriginalConstraintStructure[[ii]]==1,{
   AppendTo[ShellParaFreedomsActivate,
    Evaluate[ToExpression["ShellPara"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellOrigFreedomsActivate,Evaluate[
     ToExpression["ShellOrig"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
   AppendTo[ShellPerpFreedomsActivate,Evaluate[
     ToExpression["ShellPerp"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellPrimFreedomsActivate,Evaluate[
     ToExpression["ShellPrim"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
  }],{ii,Stranges}];
Do[
 If[BetStructureLorentzCase[[ii]]==1&&BetOriginalConstraintStructure[[ii]]==0,{
   AppendTo[ShellParaFreedomsActivate,
    Evaluate[ToExpression["ShellPara"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
   AppendTo[ShellOrigFreedomsActivate,Evaluate[
     ToExpression["ShellOrig"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellPerpFreedomsActivate,Evaluate[
     ToExpression["ShellPerp"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
   AppendTo[ShellPrimFreedomsActivate,Evaluate[
     ToExpression["ShellPrim"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
  }],{ii,Stranges}];
```

```
Do[
 If[BetStructureLorentzCase[[ii]]==1&&BetOriginalConstraintStructure[[ii]]==1,{
   AppendTo[ShellParaFreedomsActivate,
    Evaluate[ToExpression["ShellPara"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
   AppendTo[ShellOrigFreedomsActivate,Evaluate[
     ToExpression["ShellOrig"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellPerpFreedomsActivate,Evaluate[
     ToExpression["ShellPerp"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
   AppendTo[ShellPrimFreedomsActivate,Evaluate[
     ToExpression["ShellPrim"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
  }],{ii,Stranges}];
NStranges={2,6};
Do[
 If[BetStructureLorentzCase[[ii]]==0&&BetOriginalConstraintStructure[[ii]]==0,{
   AppendTo[ShellParaFreedomsActivate,
    Evaluate[ToExpression["ShellPara"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellOrigFreedomsActivate,Evaluate[
     ToExpression["ShellOrig"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellPerpFreedomsActivate,Evaluate[
     ToExpression["ShellPerp"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellPrimFreedomsActivate,Evaluate[
     ToExpression["ShellPrim"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
  }],{ii,NStranges}];
Do[
 If[BetStructureLorentzCase[[ii]]==0&&BetOriginalConstraintStructure[[ii]]==1,{
   AppendTo[ShellParaFreedomsActivate,
    Evaluate[ToExpression["ShellPara"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellOrigFreedomsActivate,Evaluate[
     ToExpression["ShellOrig"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
   AppendTo[ShellPerpFreedomsActivate,Evaluate[
     ToExpression["ShellPerp"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellPrimFreedomsActivate,Evaluate[
     ToExpression["ShellPrim"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
  }],{ii,NStranges}];
Do[
 If[BetStructureLorentzCase[[ii]]==1&&BetOriginalConstraintStructure[[ii]]==0,{
   AppendTo[ShellParaFreedomsActivate,
    Evaluate[ToExpression["ShellPara"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
   AppendTo[ShellOrigFreedomsActivate,Evaluate[
     ToExpression["ShellOrig"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
   AppendTo[ShellPerpFreedomsActivate,Evaluate[
     ToExpression["ShellPerp"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
   AppendTo[ShellPrimFreedomsActivate,Evaluate[
     ToExpression["ShellPrim"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
```

```
    }],{ii,NStranges}];
  Do[
   If[BetStructureLorentzCase[[ii]]==1&&BetOriginalConstraintStructure[[ii]]==1,{
     AppendTo[ShellParaFreedomsActivate,
      Evaluate[ToExpression["ShellPara"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
     AppendTo[ShellOrigFreedomsActivate,Evaluate[
       ToExpression["ShellOrig"<>ToString[BSectorNames[[ii]]]<>"->1"]]];
     AppendTo[ShellPerpFreedomsActivate,Evaluate[
       ToExpression["ShellPerp"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
     AppendTo[ShellPrimFreedomsActivate,Evaluate[
       ToExpression["ShellPrim"<>ToString[BSectorNames[[ii]]]<>"->0"]]];
    }],{ii,NStranges}];
  ShellFreedomsActivate=Join[ShellParaFreedomsActivate,
    ShellOrigFreedomsActivate,ShellPerpFreedomsActivate,
    ShellSingFreedomsActivate,ShellPrimFreedomsActivate];
  Print[Style["Shell freedoms:",Blue,10]];
  Print[ShellParaFreedomsActivate];
  Print[ShellOrigFreedomsActivate];
  Print[ShellPerpFreedomsActivate];
  Print[ShellSingFreedomsActivate];
  ShellFreedomsActivate];

Print[Style["Particular case we are interested in",Red,30]];
ShellFreedomsActivate=AnaMas[targ[[1]]];
Print[Theory];

Quit[];
*)
```

Dynamical variables

Define variables

```
In[]:= DefTensor[V[-a], M4, PrintAs → "n"];
AutomaticRules[V, MakeRule[{V[-a] V[a], 1}]];
DefTensor[Lapse[], M4, PrintAs → "N"];
DefTensor[Ji[], M4, PrintAs → "J^-1"];
DefTensor[J[], M4];
AutomaticRules[J, MakeRule[{J[] Ji[], 1}, MetricOn → All, ContractMetrics → True]];
CollapseJ1 = MakeRule[{J[] Ji[], 1}, MetricOn → All, ContractMetrics → True];
CollapseJ2 = MakeRule[{J[] Ji[] ^2, Ji[]}, MetricOn → All, ContractMetrics → True];
```

```
CollapseJ3 = MakeRule[{J[]^2 Ji[], J[]}, MetricOn → All, ContractMetrics → True];
CollapseJ = Join[CollapseJ1, CollapseJ2, CollapseJ3];

DefTensor[APi[-a, -b, -c], M4, Antisymmetric[{-a, -b}], PrintAs → "πA"];
DeclareOrder[APi[-a, -b, -c], 1, "IsUnityWithEHTerm" → True];
DefTensor[APiP[-a, -b, -c], M4,
  Antisymmetric[{-a, -b}], PrintAs -> "π̂A", OrthogonalTo → {V[c]}];
DeclareOrder[APiP[-a, -b, -c], 1, "IsUnityWithEHTerm" → True];
DefTensor[BPi[-a, -c], M4, PrintAs → "πb"];
DeclareOrder[BPi[-a, -c], 1];
DefTensor[BPiP[-a, -c], M4, PrintAs → "π̂b", OrthogonalTo → {V[c]}];
DeclareOrder[BPiP[-a, -c], 1];
DefTensor[H[-a, c], M4, PrintAs → "h"];
DefTensor[B[a, -c], M4, PrintAs → "b"];
(*Rule to contract Roman indices*)
AutomaticRules[H,
  MakeRule[{H[-a, i] B[a, -j], G[i, -j]}, MetricOn → All, ContractMetrics → True]];
(*Rule to contract Greek indices*)
AutomaticRules[H,
  MakeRule[{H[-a, i] B[c, -i], G[-a, c]}, MetricOn → All, ContractMetrics → True]];
DefTensor[A[a, c, -d], M4, Antisymmetric[{a, c}]];
DeclareOrder[A[a, c, -d], 1];

DefTensor[G3[-a, -b], M4, Symmetric[{-a, -b}], PrintAs → "γ̂"];
AutomaticRules[G3, MakeRule[
   {G3[-a, -b] G3[b, -d], G3[-a, -d]}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[G3, MakeRule[{G3[-a, a], 3}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[G3,
  MakeRule[{B[a, -b] G3[b, -c] V[-a], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[G3, MakeRule[{CD[-a][G3[-c, b]], 0},
   MetricOn → All, ContractMetrics → True]];

DefTensor[Eps[-a, -b, -c], M4, Antisymmetric[{-a, -b, -c}],
  OrthogonalTo → {V[a], V[b], V[c]}, PrintAs → "ê"];
DeclareOrder[CD[-z][Eps[-a, -b, -c]], 1];
DefTensor[FoliG[-a, -b], M4,
  Symmetric[{-a, -b}], OrthogonalTo → {V[a], V[b]}, PrintAs → "η̂"];
DeclareOrder[CD[-z][FoliG[-a, -b]], 1];
epsilonGVToEps = MakeRule[{V[d] epsilonG[-a, -b, -c, -d], Eps[-a, -b, -c]},
   MetricOn → All, ContractMetrics → True];
EpsToepsilonGV = MakeRule[{Eps[-a, -b, -c], V[d] epsilonG[-a, -b, -c, -d]},
   MetricOn → All, ContractMetrics → True];
GToFoliG = MakeRule[{G[-a, -b], FoliG[-a, -b] + V[-a] V[-b]},
```

```
    MetricOn → All, ContractMetrics → True];
FoliGToG = MakeRule[{FoliG[-a, -b], G[-a, -b] - V[-a] V[-b]},
    MetricOn → All, ContractMetrics → True];

DefTensor[HComp[], M4, PrintAs → "H"];
(*A dummy variable which we will use to construct Poisson brackets*)
```

ADM projections

```
In[∘]:= DefTensor[PPerp[-a, -b], M4, Symmetric[{-a, -b}], PrintAs → "𝒫⊥"];
DefTensor[PPara[-a, -b], M4, Symmetric[{-a, -b}], PrintAs → "𝒫∥"];
PPerpDefinition = V[-a] V[b];
PPerpActivate = MakeRule[{PPerp[-a, b], Evaluate[PPerpDefinition]},
    MetricOn → All, ContractMetrics → True];
PParaDefinition = G[-a, b] - V[-a] V[b];
PParaActivate = MakeRule[{PPara[-a, b], Evaluate[PParaDefinition]},
    MetricOn → All, ContractMetrics → True];
PADMActivate = Join[PPerpActivate, PParaActivate];
```

Automatic rules for converting derivatives to ∇b

In[∘]:=

```
(*Rules for converting all derivatives into
 derivatives of translational gauge fields by chain rule*)
DefTensor[DVDB[-a, -b, c], M4];
DefTensor[DHDB[-a, b, -c, d], M4];
DefTensor[DJDB[-c, d], M4];
DefTensor[DJiDB[-c, d], M4];
DefTensor[DLapseDB[-c, d], M4];

DVDBDefinition = -V[-b] PPara[i, -a] H[-i, c] /. PADMActivate // ToCanonical;
AutomaticRules[DVDB, MakeRule[{DVDB[-a, -b, c], Evaluate[DVDBDefinition]},
   MetricOn → All, ContractMetrics → True]];
DHDBDefinition = -H[-c, b] H[-a, d] // ToCanonical;
AutomaticRules[DHDB, MakeRule[{DHDB[-a, b, -c, d], Evaluate[DHDBDefinition]},
   MetricOn → All, ContractMetrics → True]];
DJDBDefinition = J[] PPara[-c, e] H[-e, d] /. PADMActivate // ToCanonical;
AutomaticRules[DJDB, MakeRule[{DJDB[-c, d], Evaluate[DJDBDefinition]},
   MetricOn → All, ContractMetrics → True]];
DJiDBDefinition = -Ji[] PPara[-c, e] H[-e, d] /. PADMActivate // ToCanonical;
AutomaticRules[DJiDB, MakeRule[{DJiDB[-c, d], Evaluate[DJiDBDefinition]},
   MetricOn → All, ContractMetrics → True]];
DLapseDBDefinition = Lapse[] PPerp[-c, e] H[-e, d] /. PADMActivate // ToCanonical;
AutomaticRules[DLapseDB, MakeRule[{DLapseDB[-c, d], Evaluate[DLapseDBDefinition]},
   MetricOn → All, ContractMetrics → True]];

AutomaticRules[V, MakeRule[{CD[-a][V[-j]],
    Evaluate[-V[-i] PPara[-j, k] H[-k, m] CD[-a][B[i, -m]] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[H, MakeRule[{CD[-a][H[-j, n]], Evaluate[
     -H[-i, n] H[-j, m] CD[-a][B[i, -m]]]}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[J, MakeRule[{CD[-a][J[]], Evaluate[
     J[] H[-k, n] PPara[k, -i] CD[-a][B[i, -n]] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[Ji, MakeRule[{CD[-a][Ji[]],
    Evaluate[-Ji[] H[-k, n] PPara[k, -i] CD[-a][B[i, -n]] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[Lapse, MakeRule[{CD[-a][Lapse[]],
    Evaluate[Lapse[] H[-k, n] PPerp[k, -i] CD[-a][B[i, -n]] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True]];

DeclareOrder[CD[-a][B[i, -m]], 1];
```

Nester form rules

In[•]:=
```
G3HExpand = MakeRule[{G3[n, -m] H[-i, m],
    Evaluate[V[-i] V[j] G3[n, -m] H[-j, m] + PPara[-i, j] H[-j, n] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True];
HG3BExpand = MakeRule[{H[-a, b] G3[-b, c] B[d, -c],
    Evaluate[PPara[-a, b] PPara[-b, d] + V[-a] V[c] H[-c, e] G3[-e, f] B[d, -f] /.
      PADMActivate // ToCanonical]}, MetricOn → All, ContractMetrics → True];
DefTensor[X[k], M4];
AutomaticRules[X,
  MakeRule[{X[-a] V[a], 1}, MetricOn → All, ContractMetrics → True]];
HG3BExpandLazy = MakeRule[{B[d, -b] G3[b, -a] H[-e, a], Evaluate[
     G[d, -e] - V[-e] X[d] /. PADMActivate]}, MetricOn → All, ContractMetrics → True];
UnprocessedX = MakeRule[{X[d], Evaluate[
     V[d] + PPara[d, -c] B[c, -b] G3[b, -e] H[-f, e] V[f] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True];(*seems I never used this below,
and I'd like to know why X didn't cause problems
 with previous velocities,
since it commonly cropps up in brackets
 with the Lapse (but not always)*)
XToV = MakeRule[{X[d], Evaluate[V[d]]}, MetricOn → All, ContractMetrics → True];
HExpandedDefinition =
  G3[-k, j] H[-i, k] + V[-i] V[k] H[-k, j] - V[-i] G3[-k, j] V[l] H[-l, k];
 (*there was a sign error here, since corrected*)
HExpand = MakeRule[{H[-i, j], Evaluate[HExpandedDefinition]},
   MetricOn → All, ContractMetrics → True];

RiemannCartanExpand =
  MakeRule[{R[a, b, -d, -e], H[-d, i] H[-e, j] (CD[-i][A[a, b, -j]] -
       CD[-j][A[a, b, -i]] + A[a, -k, -i] A[k, b, -j] - A[a, -k, -j] A[k, b, -i])},
   MetricOn → All, ContractMetrics → True];
TorsionExpand = MakeRule[{T[a, -b, -c],
    H[-b, i] H[-c, j] (CD[-i][B[a, -j]] - CD[-j][B[a, -i]] + A[a, -k, -i] B[k, -j] -
       A[a, -k, -j] B[k, -i])}, MetricOn → All, ContractMetrics → True];
ExpandStrengths = Join[RiemannCartanExpand, TorsionExpand];
ToTorsion =
  MakeRule[{CD[-s][B[a, -r]], Evaluate[Symmetrize[CD[-s][B[a, -r]], {-s, -r}] -
      Antisymmetrize[A[a, -k, -s] B[k, -r], {-s, -r}] + (1/2) B[b, -s]
       B[c, -r] T[a, -b, -c]]}, MetricOn → All, ContractMetrics → True];
ToRiemannCartan = MakeRule[{CD[-s][A[i, j, -r]],
    Evaluate[Symmetrize[CD[-s][A[i, j, -r]], {-s, -r}] - Antisymmetrize[
       A[i, -m, -s] A[m, j, -r], {-s, -r}] + (1/2) B[k, -s] B[l, -r] R[i, j, -k, -l]]},
```

```
    MetricOn → All, ContractMetrics → True];
ToStrengths = Join[ToTorsion, ToRiemannCartan];

(*would be good to put parallel momenta up here also*)

(*Defining parallel field strengths, i.e. the canonical parts*)
DefTensor[TP[-a, -b, -c], M4, Antisymmetric[{-b, -c}],
  PrintAs → "T̂", OrthogonalTo → {V[b], V[c]}];
DeclareOrder[TP[-a, -b, -c], 1];
DefTensor[RP[-a, -b, -c, -d], M4,
  {Antisymmetric[{-a, -b}], Antisymmetric[{-c, -d}]},
  PrintAs → "R̂", OrthogonalTo → {V[c], V[d]}];
DeclareOrder[RP[-a, -b, -c, -d], 1];
TPToT = MakeRule[{TP[-a, -b, -c], PPara[-b, e] PPara[-c, f] T[-a, -e, -f]},
    MetricOn → All, ContractMetrics → True];
RPToR = MakeRule[{RP[-a, -b, -c, -d], PPara[-c, e] PPara[-d, f] R[-a, -b, -e, -f]},
    MetricOn → All, ContractMetrics → True];
StrengthPToStrength = Join[TPToT, RPToR];

(*Defining parallel field strength multipliers*)
DefTensor[RLambdaP[-a, -b, -c, -d], M4,
  {Antisymmetric[{-a, -b}], Antisymmetric[{-c, -d}]},
  PrintAs → "R̂λ", OrthogonalTo → {V[c], V[d]}];
DeclareOrder[RLambdaP[-a, -b, -c, -d], 1];
DefTensor[TLambdaP[-a, -c, -d], M4,
  Antisymmetric[{-c, -d}], PrintAs → "T̂λ", OrthogonalTo → {V[c], V[d]}];
DeclareOrder[TLambdaP[-a, -c, -d], 1];
TLambdaPToTLambda =
  MakeRule[{TLambdaP[-a, -b, -c], PPara[-b, e] PPara[-c, f] TLambda[-a, -e, -f]},
    MetricOn → All, ContractMetrics → True];
RLambdaPToRLambda = MakeRule[{RLambdaP[-a, -b, -c, -d], PPara[-c, e] PPara[-d, f]
      RLambda[-a, -b, -e, -f]}, MetricOn → All, ContractMetrics → True];
StrengthLambdaPToStrengthLambda = Join[RLambdaPToRLambda, TLambdaPToTLambda];

(*Defining perpendicular field strengths, i.e. the non-canonical parts*)
DefTensor[TPerp[-a, -b], M4, PrintAs → "T*", OrthogonalTo → {V[b]}];
DeclareOrder[TPerp[-a, -b], 1];
DefTensor[RPerp[-a, -b, -c], M4,
  Antisymmetric[{-a, -b}], PrintAs → "R*", OrthogonalTo → {V[c]}];
DeclareOrder[RPerp[-a, -b, -c], 1];
TPerpToT = MakeRule[{TPerp[-a, -b], PPara[-b, f] V[g] T[-a, -f, -g]},
    MetricOn → All, ContractMetrics → True];
RPerpToR = MakeRule[{RPerp[-a, -b, -c], PPara[-c, e] V[f] R[-a, -b, -e, -f]},
```

```
    MetricOn → All, ContractMetrics → True];
StrengthPerpToStrength = Join[TPerpToT, RPerpToR];

(*Defining perpendicular field strength multipliers*)
DefTensor[TLambdaPerp[-a, -b], M4, PrintAs → "T̊λ", OrthogonalTo → {V[b]}];
DeclareOrder[TLambdaPerp[-a, -b], 1];
DefTensor[RLambdaPerp[-a, -b, -c], M4,
  Antisymmetric[{-a, -b}], PrintAs → "R̊λ", OrthogonalTo → {V[c]}];
DeclareOrder[RLambdaPerp[-a, -b, -c], 1];
TLambdaPerpToTLambda =
  MakeRule[{TLambdaPerp[-a, -b], PPara[-b, f] V[g] TLambda[-a, -f, -g]},
   MetricOn → All, ContractMetrics → True];
RLambdaPerpToRLambda = MakeRule[{RLambdaPerp[-a, -b, -c], PPara[-c, e] V[f]
     RLambda[-a, -b, -e, -f]}, MetricOn → All, ContractMetrics → True];
StrengthLambdaPerpToStrengthLambda = Join[RLambdaPerpToRLambda,
   TLambdaPerpToTLambda];

RDecomposeDefinition =
  RP[-a, -b, -c, -d] + 2 Antisymmetrize[V[-d] RPerp[-a, -b, -c], {-c, -d}] /.
        ExpandStrengths /. PADMActivate // ToCanonical //
     CollectTensors // ScreenDollarIndices // CollectTensors;
TDecomposeDefinition = TP[-a, -c, -d] + 2 Antisymmetrize[V[-d] TPerp[-a, -c],
          {-c, -d}] /. ExpandStrengths /. PADMActivate // ToCanonical //
     CollectTensors // ScreenDollarIndices // CollectTensors;
RDecompose = MakeRule[{R[-a, -b, -c, -d], Evaluate[RDecomposeDefinition]},
   MetricOn → All, ContractMetrics → True];
TDecompose = MakeRule[{T[-a, -c, -d], Evaluate[TDecomposeDefinition]},
   MetricOn → All, ContractMetrics → True];
StrengthDecompose = Join[RDecompose, TDecompose];

RLambdaDecomposeDefinition =
  RLambdaP[-a, -b, -c, -d] + 2 Antisymmetrize[V[-d] RLambdaPerp[-a, -b, -c],
          {-c, -d}] /. ExpandStrengths /. PADMActivate // ToCanonical //
     CollectTensors // ScreenDollarIndices // CollectTensors;
TLambdaDecomposeDefinition =
  TLambdaP[-a, -c, -d] + 2 Antisymmetrize[V[-d] TLambdaPerp[-a, -c], {-c, -d}] /.
        ExpandStrengths /. PADMActivate // ToCanonical //
     CollectTensors // ScreenDollarIndices // CollectTensors;
RLambdaDecompose = MakeRule[{RLambda[-a, -b, -c, -d],
    Evaluate[RLambdaDecomposeDefinition]}, MetricOn → All, ContractMetrics → True];
TLambdaDecompose = MakeRule[{TLambda[-a, -c, -d],
    Evaluate[TLambdaDecomposeDefinition]}, MetricOn → All, ContractMetrics → True];
StrengthLambdaDecompose = Join[RLambdaDecompose, TLambdaDecompose];
```

```
(*
TPToT=MakeRule[{TP[-a,-b,-c],T[-a,-i,-j]PPara[i,-b]PPara[j,-c]},
  MetricOn→All,ContractMetrics→True];
RPToR=MakeRule[{RP[-a,-b,-c,-d],R[-a,-b,-i,-j]PPara[i,-c]PPara[j,-d]},
  MetricOn→All,ContractMetrics→True];
StrengthPToStrength=Join[TPToT,RPToR];
*)(*scheduled for decomission*)

CDBCommute = MakeRule[{CD[-s][B[a, -r]],
    Evaluate[CD[-r][B[a, -s]] - 2 Antisymmetrize[A[a, -k, -s] B[k, -r], {-s, -r}] +
      B[b, -s] B[c, -r] T[a, -b, -c]]}, MetricOn → All, ContractMetrics → True];
 (*Might want to write an equivalent version for Riemann
 Cartan curvature*)

DefTensor[DV[-a, -j], M4, OrthogonalTo → {V[j]}, PrintAs → "𝔇n"];
(*DeclareOrder[DV[-a,-j],1];*)
DefTensor[DJ[-a], M4, PrintAs -> "𝔇J"];
(*DeclareOrder[DJ[-a],1];*)

G3VCDBToG3DV = MakeRule[{G3[-l, n] V[-k] CD[-m][B[k, -n]],
    -G3[-l, n] B[j, -n] A[k, -j, -m] V[-k] - G3[-l, n] B[j, -n] DV[-m, -j]},
   MetricOn → All, ContractMetrics → True];

G3HCDBToDJ = MakeRule[{G3[n, -s] H[-k, s] CD[-m][B[k, -n]], Ji[] DJ[-m] -
     V[k] H[-k, a] G3[-a, b] (B[j, -b] DV[-m, -j] + V[-l] A[l, -j, -m] B[j, -b])},
   MetricOn → All, ContractMetrics → True];

(*we want to be able to reverse the v and J derivatives also,
this below just some syntax for that time*)
(*
G3DVToG3VCDB=MakeRule[{G3[-l,n]V[-k]CD[-m][B[k,-n]],
   -G3[-l,n]B[j,-n]A[k,-j,-m]V[-k]-G3[-l,n]B[j,-n]DV[-m,-j]},
  MetricOn→All,ContractMetrics→True];
(*the rules below should of course be generalised beyond simply the
 momenta -- these below now generalise to the field strengths*)
DTP0mDeactivate=MakeRule[{DTP0m[-z],CD[-z][TP0m[]]},
  MetricOn→All,ContractMetrics→True];
DTP1pDeactivate=MakeRule[{DTP1p[-z,-a,-b],
   CD[-z][TP1p[-a,-b]]-A[i,-a,-z]TP1p[-i,-b]-A[i,-b,-z]TP1p[-a,-i]},
  MetricOn→All,ContractMetrics→True];
DTP1mDeactivate=MakeRule[{DTP1m[-z,-a],CD[-z][TP1m[-a]]-A[i,-a,-z]TP1m[-i]},
  MetricOn→All,ContractMetrics→True];
```

```
DTP2mDeactivate=MakeRule[{DTP2m[-z,-a,-b,-c],
   CD[-z][TP2m[-a,-b,-c]]-A[i,-a,-z]TP2m[-i,-b,-c]-A[i,-b,-z]TP2m[-a,-i,-c]-
    A[i,-c,-z]TP2m[-a,-b,-i]},MetricOn→All,ContractMetrics→True];
DRP0pDeactivate=MakeRule[{DRP0p[-z],CD[-z][RP0p[]]},
  MetricOn→All,ContractMetrics→True];
DRP0mDeactivate=MakeRule[{DRP0m[-z],CD[-z][RP0m[]]},
  MetricOn→All,ContractMetrics→True];
DRP1pDeactivate=MakeRule[{DRP1p[-z,-a,-b],
   CD[-z][RP1p[-a,-b]]-A[i,-a,-z]RP1p[-i,-b]-A[i,-b,-z]RP1p[-a,-i]},
  MetricOn→All,ContractMetrics→True];
DRP1mDeactivate=MakeRule[{DRP1m[-z,-a],CD[-z][RP1m[-a]]-A[i,-a,-z]RP1m[-i]},
  MetricOn→All,ContractMetrics→True];
DRP2pDeactivate=MakeRule[{DRP2p[-z,-a,-b],
   CD[-z][RP2p[-a,-b]]-A[i,-a,-z]RP2p[-i,-b]-A[i,-b,-z]RP2p[-a,-i]},
  MetricOn→All,ContractMetrics→True];
DRP2mDeactivate=MakeRule[{DRP2m[-z,-a,-b,-c],
   CD[-z][RP2m[-a,-b,-c]]-A[i,-a,-z]RP2m[-i,-b,-c]-A[i,-b,-z]RP2m[-a,-i,-c]-
    A[i,-c,-z]RP2m[-a,-b,-i]},MetricOn→All,ContractMetrics→True];
DRPDeactivate=Join[DTP0mDeactivate,DTP1pDeactivate,DTP1mDeactivate,
  DTP2mDeactivate,DRP0pDeactivate,DRP0mDeactivate,DRP1pDeactivate,
  DRP1mDeactivate,DRP2pDeactivate,DRP2mDeactivate];
*)

DefTensor[DpJ[-z], M4, PrintAs → "𝔇̂J", OrthogonalTo → {V[z]}];
DeclareOrder[DpJ[-z], 1];
DeclareOrder[DJ[-z], 1,
  "approximation" -> B[w, -z] DpJ[-w] + V[-v] B[v, -z] V[u] H[-u, w] DJ[-w]];
DpJActivate = MakeRule[{G3[-y, z] DJ[-z], G3[-y, z] B[x, -z] DpJ[-x]},
   MetricOn → All, ContractMetrics → True];
DefTensor[DpV[-z, -a], M4, PrintAs → "𝔇̂n", OrthogonalTo → {V[z], V[a]}];
DeclareOrder[DpV[-z, -a], 1];
DeclareOrder[DV[-z, -a], 1,
  "approximation" -> B[w, -z] DpV[-w, -a] + V[-v] B[v, -z] V[u] H[-u, w] DV[-w, -a]];
DpVActivate = MakeRule[{G3[-y, z] DV[-z, -a], Evaluate[
     G3[-y, z] B[x, -z] DpV[-x, -a] + (G[-a, i] - PPara[-a, i]) G3[-y, z] DV[-z, -i] /.
      PADMActivate]}, MetricOn → All, ContractMetrics → True];

DpVExpand = MakeRule[{DpV[-m, -j], Evaluate[
     Symmetrize[DpV[-m, -j], {-m, -j}] - (1/2) V[-i] TP[i, -m, -j] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True];

AVepsilonGToAVEps =
  MakeRule[{A[-e, d, -f] epsilonG[-d, -a, -b, -c] V[e], A[-e, d, -f] V[e]
```

```
    (V[-a] Eps[-d, -b, -c] - V[-b] Eps[-d, -a, -c] + V[-c] Eps[-d, -a, -b])},
   MetricOn → All, ContractMetrics → True];
HEpsToHG3Eps = MakeRule[{Eps[-a, -b, c] H[-c, e], Eps[-a, -b, c] H[-c, f] G3[e, -f]},
   MetricOn → All, ContractMetrics → True];
epsilonGToEpsV = MakeRule[{epsilonG[-a, -b, -c, -d],
    -V[-a] Eps[-b, -c, -d] + V[-b] Eps[-a, -c, -d] - V[-c] Eps[-a, -b, -d] +
     V[-d] Eps[-a, -b, -c]}, MetricOn → All, ContractMetrics → True];
DefTensor[Q[-a, -b], M4, OrthogonalTo → {V[a], V[b]}];
DeclareOrder[Q[-a, -b], 1];
AHEpsExpand = MakeRule[{A[-i, j, -m] Eps[-j, -p, -q] H[-k, m],
    Evaluate[Eps[-i, j, -z] Q[z, -k] Eps[-j, -p, -q] +
       PPerp[-i, a] PPara[-k, b] A[-a, j, -m] Eps[-j, -p, -q] H[-b, m] +
       PPara[-i, a] PPerp[-k, b] A[-a, j, -m] Eps[-j, -p, -q] H[-b, m] +
       PPerp[-i, a] PPerp[-k, b] A[-a, j, -m] Eps[-j, -p, -q] H[-b, m] /.
      PADMActivate]}, MetricOn → All, ContractMetrics → True];
EpsEpsExpand = MakeRule[{Eps[i, a, b] Eps[-i, -c, -d],
    Evaluate[PPara[a, -c] PPara[b, -d] - PPara[a, -d] PPara[b, -c] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True];

DefTensor[CDAInert[-a, -b, -c, -d], M4, Antisymmetric[{-b, -c}]];
DeclareOrder[CDAInert[-a, -b, -c, -d], 1];
CDAToCDAInert = MakeRule[{CD[-a][A[-b, -c, -d]], CDAInert[-a, -b, -c, -d]},
   MetricOn → All, ContractMetrics → True];
CDAInertToCDA = MakeRule[{CDAInert[-a, -b, -c, -d], CD[-a][A[-b, -c, -d]]},
   MetricOn → All, ContractMetrics → True];
AExpandedDefinition = PPara[-a, i] PPara[-b, j] A[-i, -j, -c] +
    PPerp[-a, i] PPara[-b, j] A[-i, -j, -c] -
    PPerp[-b, i] PPara[-a, j] A[-i, -j, -c] /. PADMActivate;
CDAExpandedDefinition = PPara[-a, i] PPara[-b, j] CDAInert[-k, -i, -j, -c] +
    PPerp[-a, i] PPara[-b, j] CDAInert[-k, -i, -j, -c] -
    PPerp[-b, i] PPara[-a, j] CDAInert[-k, -i, -j, -c] /. PADMActivate;
AToAExpanded = MakeRule[{A[-a, -b, -c], Evaluate[AExpandedDefinition]},
   MetricOn → All, ContractMetrics → True];
CDAToCDAExpanded = MakeRule[{CDAInert[-k, -a, -b, -c],
    Evaluate[CDAExpandedDefinition]}, MetricOn → All, ContractMetrics → True];
AExpand = Join[AToAExpanded, CDAToCDAExpanded];
HVCDADefinition = H[-i, m] V[b] CDAInert[-k, i, -b, -c] /. PADMActivate;
HVADefinition = H[-i, m] V[b] A[i, -b, -c] /. PADMActivate;
HG3VCDAToHVCDA = MakeRule[{H[-i, j] G3[-j, m] V[b] CDAInert[-k, i, -b, -c],
    Evaluate[HVCDADefinition]}, MetricOn → All, ContractMetrics → True];
HG3VAToHVA = MakeRule[{H[-i, j] G3[-j, m] V[b] A[i, -b, -c], Evaluate[HVADefinition]},
   MetricOn → All, ContractMetrics → True];
```

Basic form covariance check on $\mathbb{R}^{1,3} \rtimes SO^{+}(1,3)$

In[∘]:=

```
(*Tools for covariance check,
which is useful for emergencies but otherwise commented out*)
(*
DefTensor[CCoord[-a,-b,c],M4,Symmetric[{-a,-b}]]
 DefTensor[FLorentz[-a,-b,-c],M4,PrintAs→"FAILΛ"]
 DefTensor[FCoord[-a,-b,-c],M4,PrintAs→"FAILX"]
 DefTensor[Lorentz[a,-b],M4,PrintAs→"Λ"]
 AutomaticRules[Lorentz,MakeRule[
   {Lorentz[-a,-b]Lorentz[a,-c],G[-b,-c]},MetricOn→All,ContractMetrics→True]];
AutomaticRules[Lorentz,MakeRule[{Lorentz[-b,-a]Lorentz[-c,a],G[-c,-b]},
  MetricOn→All,ContractMetrics→True]];
DefTensor[Coord[a,-b],M4,PrintAs→"X"]
 AutomaticRules[Coord,MakeRule[
   {Coord[-a,-b]Coord[a,-c],G[-b,-c]},MetricOn→All,ContractMetrics→True]];
AutomaticRules[Coord,MakeRule[{Coord[-b,-a]Coord[-c,a],G[-c,-b]},
  MetricOn→All,ContractMetrics→True]];

DefTensor[CDBInert[-a,b,-c],M4];
DefTensor[CDAInert[-a,b,c,-d],M4,Antisymmetric[{b,c}]];
ToCDBInert=
 MakeRule[{CD[-a][B[b,-c]],CDBInert[-a,b,-c]},MetricOn→All,ContractMetrics→True];
ToCDAInert=MakeRule[{CD[-a][A[b,c,-d]],CDAInert[-a,b,c,-d]},
  MetricOn→All,ContractMetrics→True];
ToCDInert=Join[ToCDBInert,ToCDAInert];

GaugeB=MakeRule[{B[b,-c],Lorentz[b,-j]Coord[-c,k]B[j,-k]},
  MetricOn→All,ContractMetrics→True];
GaugeH=MakeRule[{H[-b,c],Lorentz[-b,j]Coord[c,-k]H[-j,k]},
  MetricOn→All,ContractMetrics→True];
GaugeV=MakeRule[{V[b],Lorentz[b,-j]V[j]},MetricOn→All,ContractMetrics→True];
GaugeA=
 MakeRule[{A[b,c,-d],Lorentz[b,-j]Lorentz[c,-k]Coord[-d,l]A[j,k,-l]-Lorentz[c,j]
     Coord[-d,l]CD[-l][Lorentz[b,-j]]},MetricOn→All,ContractMetrics→True];
GaugeMe=Join[GaugeB,GaugeH,GaugeV,GaugeA];

GaugeCDA=MakeRule[{CDAInert[-a,b,c,-d],
   Coord[-a,i]CD[-i][Lorentz[b,-j]Lorentz[c,-k]Coord[-d,l]A[j,k,-l]-Lorentz[c,j]
       Coord[-d,l]CD[-l][Lorentz[b,-j]]]},MetricOn→All,ContractMetrics→True];
GaugeCDB=MakeRule[{CDBInert[-a,b,-c],Coord[-a,i]
    CD[-i][Lorentz[b,-j]Coord[-c,k]B[j,-k]]},MetricOn→All,ContractMetrics→True];
```

```
GaugeMeInert=Join[GaugeCDB,GaugeCDA];

ToCCoord=MakeRule[{CD[-a][Coord[-b,c]],Coord[s,-a]CCoord[-s,-b,c]},
  MetricOn→All,ContractMetrics→True];

DefTensor[Toten[b,-c,d],M4,Symmetric[{b,d}]];
(*SwitchMe=MakeRule[{Lorentz[a,-b]CD[-c][Lorentz[-a,-d]],
    Toten[-b,-c,-d]-Lorentz[-a,-d]CD[-c][Lorentz[a,-b]]},
   MetricOn→All,ContractMetrics→True];*)
(*SwitchMe=MakeRule[{Lorentz[a,-b]CD[-c][Lorentz[-a,-d]],
    -Lorentz[-a,-d]CD[-c][Lorentz[a,-b]]},MetricOn→All,ContractMetrics→True];*)
CommuteMe=MakeRule[{Lorentz[a,-b]CD[-c][Lorentz[-a,-d]],
   Evaluate[Antisymmetrize[Lorentz[a,-b]CD[-c][Lorentz[-a,-d]],{-b,-d}]]},
  MetricOn→All,ContractMetrics→True];
SwitchMe=MakeRule[{Lorentz[a,-b]CD[-c][Lorentz[-a,-d]],
   -Lorentz[-a,-d]CD[-c][Lorentz[a,-b]]},MetricOn→All,ContractMetrics→True];

FlagLorentz=MakeRule[{CD[-a][Lorentz[-b,-c]],FLorentz[-a,-b,-c]},
  MetricOn→All,ContractMetrics→True];
FlagCoord=MakeRule[{CD[-a][Coord[-b,-c]],FCoord[-a,-b,-c]},
  MetricOn→All,ContractMetrics→True];
FlagBroken=Join[FlagLorentz,FlagCoord];

ManRemoveG3=MakeRule[{G3[-b,c],G[-b,c]},MetricOn→All,ContractMetrics→True];

GaugeShift[x_]:=Module[{exp},
  exp=x;
  Print[Style["Manually removing G3",Blue,10]];
  exp=exp/.ManRemoveG3;
  Print[Style["simplifying",Blue,10]];
  exp=exp//ToCanonical//ScreenDollarIndices//ContractMetric//CollectTensors;
  Print[Style["Converting to inert",Blue,10]];
  exp=exp/.ToCDInert;
  Print[Style["simplifying",Blue,10]];
  exp=exp//ToCanonical//ScreenDollarIndices//ContractMetric//CollectTensors;
  Print[Style["transforming gauge",Blue,10]];
  exp=exp/.GaugeMe;
  Print[Style["simplifying",Blue,10]];
  exp=exp//ToCanonical//ScreenDollarIndices//ContractMetric//CollectTensors;
  Print[Style["transforming CD gauge",Blue,10]];
  exp=exp/.GaugeMeInert;
  Print[Style["simplifying",Blue,10]];
  exp=exp//ToCanonical//ScreenDollarIndices//ContractMetric//CollectTensors;
  Print[Style["transforming to coordinate Hessian",Blue,10]];
```

```
  exp=exp/.ToCCoord;
  Print[Style["simplifying",Blue,10]];
  exp=exp//ToCanonical//ScreenDollarIndices//ContractMetric//CollectTensors;
  Print[Style["removing scalar",Blue,10]];
  exp=exp//NoScalar;
  Print[Style["commuting Lorentz gradients",Blue,10]];
  exp=exp/.SwitchMe;
  Print[Style["simplifying",Blue,10]];
  exp=exp//ToCanonical//ScreenDollarIndices//ContractMetric//CollectTensors;
  Print[Style["removing scalar",Blue,10]];
  exp=exp//NoScalar;
  Print[Style["commuting Lorentz gradients",Blue,10]];
  exp=exp/.CommuteMe;
  Print[Style["simplifying",Blue,10]];
  exp=exp//ToCanonical//ScreenDollarIndices//ContractMetric//CollectTensors;
  Print[Style["removing scalar",Blue,10]];
  exp=exp//NoScalar;
  Print[Style["commuting Lorentz gradients",Blue,10]];
  exp=exp/.SwitchMe;
  Print[Style["simplifying",Blue,10]];
  exp=exp//ToCanonical//ScreenDollarIndices//ContractMetric//CollectTensors;
  Print[Style["raising flags",Blue,10]];
  exp=exp/.FlagBroken;
  exp];
*)
```

Irreducible decomposition of the fields using O(3)

Human-readable projections $\{{}^{A}\check{\mathcal{P}}\}$, $\{{}^{E}\check{\mathcal{P}}\}$

In[∘]:=
```
DefTensor[PThreePara[-a, -b, -c, d, e, f],
  M4, {Antisymmetric[{-a, -b}], Antisymmetric[{d, e}]}];
PThreeParaDefinition =
  Antisymmetrize [Antisymmetrize[PPara[-a, d] PPara[-b, e] PPara[-c, f], {-a, -b}],
     {d, e}] /. PADMActivate // ToCanonical;
PThreeParaActivate = MakeRule[{PThreePara[-a, -b, -c, d, e, f],
    Evaluate[PThreeParaDefinition]}, MetricOn → All, ContractMetrics → True];
DefTensor[PThreePerp[-a, -b, -c, d, e, f], M4,
  {Antisymmetric[{-a, -b}], Antisymmetric[{d, e}]}];
PThreePerpDefinition = Antisymmetrize[Antisymmetrize[
     (PPara[-a, d] PPerp[-b, e] + PPerp[-a, d] PPara[-b, e]) PPara[-c, f], {-a, -b}],
```

```
      {d, e}] /. PADMActivate // ToCanonical;
PPerpActivate = MakeRule[{PThreePerp[-a, -b, -c, d, e, f],
    Evaluate[PThreePerpDefinition]}, MetricOn → All, ContractMetrics → True];

DefTensor[PAPerp[-a, -b, d, e, f], M4];
DefTensor[PAPara[-a, -b, -c, d, e, f], M4];
DefTensor[PBPerp[-a, d, e], M4];
DefTensor[PBPara[-a, -b, d, e], M4];

PAPerpDefinition = V[d] PPara[-a, e] G[-b, f] /. PADMActivate // ToCanonical;
PAPerpActivate = MakeRule[{PAPerp[-a, -b, d, e, f], Evaluate[PAPerpDefinition]},
   MetricOn → All, ContractMetrics → True];
PAParaDefinition = PPara[-a, d] PPara[-b, e] G[-c, f] /. PADMActivate // ToCanonical;
PAParaActivate = MakeRule[{PAPara[-a, -b, -c, d, e, f], Evaluate[PAParaDefinition]},
   MetricOn → All, ContractMetrics → True];

PBPerpDefinition = V[d] G[-a, e] /. PADMActivate // ToCanonical;
PBPerpActivate = MakeRule[{PBPerp[-a, d, e], Evaluate[PBPerpDefinition]},
   MetricOn → All, ContractMetrics → True];
PBParaDefinition = PPara[-a, d] G[-b, e] /. PADMActivate // ToCanonical;
PBParaActivate = MakeRule[{PBPara[-a, -b, d, e], Evaluate[PBParaDefinition]},
   MetricOn → All, ContractMetrics → True];

PADMPiActivate =
  Join[PAPerpActivate, PAParaActivate, PBPerpActivate, PBParaActivate];

DefTensor[PA0p[c, d], M4, PrintAs -> "A0+φ̌"];
DefTensor[PA1p[-a, -b, c, d], M4, PrintAs -> "A1+φ̌"];
DefTensor[PA2p[-a, -b, c, d], M4, PrintAs -> "A2+φ̌"];

PA0pDefinition = PPara[c, -k] PPara[d, -l] G[k, l] /. PADMActivate // ToCanonical;
PA1pDefinition = PPara[-a, i] PPara[-b, j] PPara[c, -k] PPara[d, -l]
     Antisymmetrize[G[-i, k] G[-j, l], {-i, -j}] /. PADMActivate // ToCanonical;
PA2pDefinition = PPara[-a, i] PPara[-b, j] PPara[c, -k] PPara[d, -l]
     (Symmetrize[G[-i, k] G[-j, l], {-i, -j}] - (1/3) G[-i, -j] G[k, l]) /.
    PADMActivate // ToCanonical;

DefTensor[PA0m[d, e, f], M4, PrintAs -> "A0-φ̌"];
DefTensor[PA1m[-a, d, e, f], M4, PrintAs -> "A1-φ̌"];
DefTensor[PA2m[-a, -b, -c, d, e, f], M4, PrintAs -> "A2-φ̌"];

PA0mDefinition =
  PPara[-i, d] PPara[-j, e] PPara[-k, f] epsilonG[i, j, k, g] V[-g] /. PADMActivate //
```

```
    ToCanonical;
PA1mDefinition = PPara[-i, d] PPara[-j, f] PPara[k, -a] PPara[-l, e] G[i, j] G[-k, l] /.
     PADMActivate // ToCanonical;
PA2mDefinition = PPara[-a, i] PPara[-b, j] PPara[-c, k] PPara[d, -l]
      PPara[e, -n] PPara[f, -m] (3/4) ((1/3) (2 G[-i, l] G[-j, n] G[-k, m] -
            G[-j, l] G[-k, n] G[-i, m] - G[-k, l] G[-i, n] G[-j, m]) - Antisymmetrize[
         G[-i, -k] G[-j, n] G[l, m], {-i, -j}]) /. PADMActivate // ToCanonical;

PA0pActivate = MakeRule[{PA0p[c, d], Evaluate[PA0pDefinition]},
    MetricOn → All, ContractMetrics → True];
PA1pActivate = MakeRule[{PA1p[-a, -b, c, d], Evaluate[PA1pDefinition]},
    MetricOn → All, ContractMetrics → True];
PA2pActivate = MakeRule[{PA2p[-a, -b, c, d], Evaluate[PA2pDefinition]},
    MetricOn → All, ContractMetrics → True];
PA0mActivate = MakeRule[{PA0m[d, e, f], Evaluate[PA0mDefinition]},
    MetricOn → All, ContractMetrics → True];
PA1mActivate = MakeRule[{PA1m[-a, d, e, f], Evaluate[PA1mDefinition]},
    MetricOn → All, ContractMetrics → True];
PA2mActivate = MakeRule[{PA2m[-a, -b, -c, d, e, f], Evaluate[PA2mDefinition]},
    MetricOn → All, ContractMetrics → True];

DefTensor[PB0p[c, d], M4, PrintAs -> "b0+𝒫̌"];
DefTensor[PB1p[-a, -b, c, d], M4, PrintAs -> "b1+𝒫̌"];
DefTensor[PB2p[-a, -b, c, d], M4, PrintAs -> "b2+𝒫̌"];
DefTensor[PB1m[-a, d], M4, PrintAs -> "b1-𝒫̌"];

PB0pDefinition = PPara[c, -k] PPara[d, -l] G[k, l] /. PADMActivate // ToCanonical;
PB1pDefinition = PPara[-a, i] PPara[-b, j] PPara[c, -k] PPara[d, -l]
      Antisymmetrize[G[-i, k] G[-j, l], {-i, -j}] /. PADMActivate // ToCanonical;
PB2pDefinition = PPara[-a, i] PPara[-b, j] PPara[c, -k] PPara[d, -l]
      (Symmetrize[G[-i, k] G[-j, l], {-i, -j}] - (1/3) G[-i, -j] G[k, l]) /.
     PADMActivate // ToCanonical;
PB1mDefinition = PPara[d, -j] PPara[-a, i] G[-i, j] /. PADMActivate // ToCanonical;

PB0pActivate = MakeRule[{PB0p[c, d], Evaluate[PB0pDefinition]},
    MetricOn → All, ContractMetrics → True];
PB1pActivate = MakeRule[{PB1p[-a, -b, c, d], Evaluate[PB1pDefinition]},
    MetricOn → All, ContractMetrics → True];
PB2pActivate = MakeRule[{PB2p[-a, -b, c, d], Evaluate[PB2pDefinition]},
    MetricOn → All, ContractMetrics → True];
PB1mActivate = MakeRule[{PB1m[-a, d], Evaluate[PB1mDefinition]},
    MetricOn → All, ContractMetrics → True];
```

```
PO3PiActivate =
  Join[PA0pActivate, PA1pActivate, PA2pActivate, PA0mActivate, PA1mActivate,
   PA2mActivate, PB0pActivate, PB1pActivate, PB2pActivate, PB1mActivate];

APiToAPiP = MakeRule[{APi[-i, -j, k] G3[-k, a] B[l, -a], APiP[-i, -j, l]},
   MetricOn → All, ContractMetrics → True];
BPiToBPiP = MakeRule[{BPi[-i, k] G3[-k, a] B[l, -a], BPiP[-i, l]},
   MetricOn → All, ContractMetrics → True];
PiToPiP = Join[APiToAPiP, BPiToBPiP];
CDAPiToCDAPiP = MakeRule[{CD[-z][APi[-i, -j, k]] G3[-k, a] B[l, -a],
    CD[-z][APiP[-i, -j, l]] - APi[-i, -j, k] G3[-k, a] CD[-z][B[l, -a]]},
   MetricOn → All, ContractMetrics → True];
CDBPiToCDBPiP = MakeRule[{CD[-z][BPi[-i, k]] G3[-k, a] B[l, -a],
    CD[-z][BPiP[-i, l]] - BPi[-i, k] G3[-k, a] CD[-z][B[l, -a]]},
   MetricOn → All, ContractMetrics → True];
CDPiToCDPiP = Join[CDAPiToCDAPiP, CDBPiToCDBPiP];
APiToAPiPHard =
  MakeRule[{APi[-i, -j, k] G3[-k, a], APiP[-i, -j, l] PPara[-l, s] H[-s, f] G3[-f, a]},
   MetricOn → All, ContractMetrics → True];
BPiToBPiPHard = MakeRule[{BPi[-i, k] G3[-k, a], BPiP[-i, l] PPara[-l, s]
     H[-s, f] G3[-f, a]}, MetricOn → All, ContractMetrics → True];
PiToPiPHard = Join[APiToAPiPHard, BPiToBPiPHard];
CDAPiToCDAPiPHard = MakeRule[{CD[-z][APi[-i, -j, k]] G3[-k, a],
    Evaluate[CD[-z][APiP[-i, -j, l]] PPara[-l, s] H[-s, f] G3[-f, a] +
       APiP[-i, -j, l] CD[-z][PPara[-l, s] H[-s, f] G3[-f, a]] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True];
CDBPiToCDBPiPHard = MakeRule[{CD[-z][BPi[-i, k]] G3[-k, a],
    Evaluate[CD[-z][BPiP[-i, l]] PPara[-l, s] H[-s, f] G3[-f, a] +
       BPiP[-i, l] CD[-z][PPara[-l, s] H[-s, f] G3[-f, a]] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True];
CDPiToCDPiPHard = Join[CDAPiToCDAPiPHard, CDBPiToCDBPiPHard];
APiPToAPi = MakeRule[{APiP[-i, -j, l], APi[-i, -j, k] G3[-k, a] B[l, -a]},
   MetricOn → All, ContractMetrics → True];
BPiPToBPi = MakeRule[{BPiP[-i, l], BPi[-i, k] G3[-k, a] B[l, -a]},
   MetricOn → All, ContractMetrics → True];
PiPToPi = Join[APiPToAPi, BPiPToBPi];

ActivateGeneralO3Projections[expr_] := Module[{exp, kern}, exp = Evaluate[expr];
   exp = exp // ToCanonical;
   exp = exp /. PActivate;
   exp = exp // ToCanonical;
   exp = exp /. PADMActivate;
   exp = exp // ToCanonical;
   exp = exp /. PADMPiActivate;
```

```
exp = exp // ToCanonical;
exp = exp /. PO3PiActivate;
exp = exp // ToCanonical;
exp = exp /. HG3BExpandLazy;
exp = exp // ContractMetric;
exp = exp // ToCanonical;
exp = exp // CollectTensors;
exp];
```

Complete projections $\{{}^{A}\hat{\mathcal{P}}\}$, $\{{}^{E}\hat{\mathcal{P}}\}$

```
In[∘]:= DefTensor[PB0pT[-n, -m, a, c], M4, PrintAs -> "b0+𝒫̂"];
DefTensor[PB1pT[-n, -m, a, c], M4, PrintAs -> "b1+𝒫̂"];
DefTensor[PB2pT[-n, -m, a, c], M4, PrintAs -> "b2+𝒫̂"];
DefTensor[PB1mT[-n, -m, a, c], M4, PrintAs -> "b1-𝒫̂"];

DefTensor[PA0pT[-n, -m, -o, a, b, c], M4, PrintAs -> "A0+𝒫̂"];
DefTensor[PA1pT[-n, -m, -o, a, b, c], M4, PrintAs -> "A1+𝒫̂"];
DefTensor[PA2pT[-n, -m, -o, a, b, c], M4, PrintAs -> "A2+𝒫̂"];
DefTensor[PA0mT[-n, -m, -o, a, b, c], M4, PrintAs -> "A0-𝒫̂"];
DefTensor[PA1mT[-n, -m, -o, a, b, c], M4, PrintAs -> "A1-𝒫̂"];
DefTensor[PA2mT[-n, -m, -o, a, b, c], M4, PrintAs -> "A2-𝒫̂"];

If[CompleteO3ProjectionsToggle,
  PB0pTDefinition =
   (1/3) PPara[-n, -m] PB0p[e, f] PBPara[-e, -f, a, c] /. PO3PiActivate /.
     PADMPiActivate /. PADMActivate // ToCanonical;
  PB1pTDefinition = PB1p[-n, -m, e, f] PBPara[-e, -f, a, c] /. PO3PiActivate /.
     PADMPiActivate /. PADMActivate // ToCanonical;
  PB2pTDefinition = PB2p[-n, -m, e, f] PBPara[-e, -f, a, c] /. PO3PiActivate /.
     PADMPiActivate /. PADMActivate // ToCanonical;
  PB1mTDefinition = V[-n] PB1m[-m, f] PBPerp[-f, a, c] /. PO3PiActivate /.
     PADMPiActivate /. PADMActivate // ToCanonical;

  PB0pTActivate = MakeRule[{PB0pT[-n, -m, a, c], Evaluate[PB0pTDefinition]},
    MetricOn → All, ContractMetrics → True];
  PB1pTActivate = MakeRule[{PB1pT[-n, -m, a, c], Evaluate[PB1pTDefinition]},
    MetricOn → All, ContractMetrics → True];
  PB2pTActivate = MakeRule[{PB2pT[-n, -m, a, c], Evaluate[PB2pTDefinition]},
    MetricOn → All, ContractMetrics → True];
  PB1mTActivate = MakeRule[{PB1mT[-n, -m, a, c], Evaluate[PB1mTDefinition]},
    MetricOn → All, ContractMetrics → True];
```

```
PA0pTDefinition =
  Antisymmetrize[Antisymmetrize[2 Antisymmetrize[V[-n] (1/3) PPara[-m, -o]
        PA0p[e, f] PAPerp[-e, -f, a, b, c], {-n, -m}], {-n, -m}], {a, b}] /.
     PO3PiActivate /. PADMPiActivate /. PADMActivate // ToCanonical;
PA1pTDefinition = Antisymmetrize[Antisymmetrize[2 Antisymmetrize[
        V[-n] PA1p[-m, -o, e, f] PAPerp[-e, -f, a, b, c], {-n, -m}], {-n, -m}],
      {a, b}] /. PO3PiActivate /. PADMPiActivate /. PADMActivate // ToCanonical;
PA2pTDefinition = Antisymmetrize[Antisymmetrize[2 Antisymmetrize[
        V[-n] PA2p[-m, -o, e, f] PAPerp[-e, -f, a, b, c], {-n, -m}], {-n, -m}],
      {a, b}] /. PO3PiActivate /. PADMPiActivate /. PADMActivate // ToCanonical;
PA0mTDefinition = Antisymmetrize[Antisymmetrize[(-1/6) PA0m[-n, -m, -o]
        PA0m[i, j, k] PAPara[-i, -j, -k, a, b, c], {-n, -m}], {a, b}] /.
     PO3PiActivate /. PADMPiActivate /. PADMActivate // ToCanonical;
PA1mTDefinition = Antisymmetrize[Antisymmetrize[Antisymmetrize[-PPara[-m, -o]
        PA1m[-n, i, j, k] PAPara[-i, -j, -k, a, b, c], {-m, -n}], {-n, -m}],
      {a, b}] /. PO3PiActivate /. PADMPiActivate /. PADMActivate // ToCanonical;
PA2mTDefinition = Antisymmetrize[Antisymmetrize[(4/3) PA2m[-n, -m, -o, d, e, f]
        PAPara[-d, -e, -f, a, b, c], {-n, -m}], {a, b}] /.
     PO3PiActivate /. PADMPiActivate /. PADMActivate // ToCanonical;

PA0pTActivate = MakeRule[{PA0pT[-n, -m, -o, a, b, c], Evaluate[PA0pTDefinition]},
   MetricOn → All, ContractMetrics → True];
PA1pTActivate = MakeRule[{PA1pT[-n, -m, -o, a, b, c], Evaluate[PA1pTDefinition]},
   MetricOn → All, ContractMetrics → True];
PA2pTActivate = MakeRule[{PA2pT[-n, -m, -o, a, b, c], Evaluate[PA2pTDefinition]},
   MetricOn → All, ContractMetrics → True];
PA0mTActivate = MakeRule[{PA0mT[-n, -m, -o, a, b, c], Evaluate[PA0mTDefinition]},
   MetricOn → All, ContractMetrics → True];
PA1mTActivate = MakeRule[{PA1mT[-n, -m, -o, a, b, c], Evaluate[PA1mTDefinition]},
   MetricOn → All, ContractMetrics → True];
PA2mTActivate = MakeRule[{PA2mT[-n, -m, -o, a, b, c], Evaluate[PA2mTDefinition]},
   MetricOn → All, ContractMetrics → True];

NewPO3TActivate =
  Join[PB0pTActivate, PB1pTActivate, PB2pTActivate, PB1mTActivate, PA0pTActivate,
   PA1pTActivate, PA2pTActivate, PA0mTActivate, PA1mTActivate, PA2mTActivate];

tmp =
  (PA0pT[-n, -m, -o, a, b, c] + PA1pT[-n, -m, -o, a, b, c] + PA2pT[-n, -m, -o, a, b, c] +
        PA0mT[-n, -m, -o, a, b, c] + PA1mT[-n, -m, -o, a, b, c] + PA2mT[-n, -m,
         -o, a, b, c]) APi[-a, -b, -e] G3[e, -f] B[-c, f] /. NewPO3TActivate /.
     PO3PiActivate /. PADMPiActivate /. PADMActivate // ToCanonical;
Print[tmp];
```

```
    tmp =
     (PB0pT[-n, -m, a, c] + PB1pT[-n, -m, a, c] + PB2pT[-n, -m, a, c] + PB1mT[-n, -m, a,
           c]) BPi[-a, -e] G3[e, -f] B[-c, f] /. NewPO3TActivate /.
        PO3PiActivate /. PADMPiActivate /. PADMActivate // ToCanonical;
    Print[tmp];

    DumpSave[NotebookDirectory[] <> "mx_cache/O3T.mx", {NewPO3TActivate}];
    Quit[];
   ];
  MyImport["O3T.mx"];
```

Projection normalisations $\{c^{\perp}_{A}\}$. $\{c^{\perp}_{E}\}$

```
In[•]:= DefConstantSymbol[cPerpA0p, PrintAs → "c⊥A0+"];
  DefConstantSymbol[cPerpA0m, PrintAs → "c⊥A0-"];
  DefConstantSymbol[cPerpA1p, PrintAs → "c⊥A1+"];
  DefConstantSymbol[cPerpA1m, PrintAs → "c⊥A1-"];
  DefConstantSymbol[cPerpA2p, PrintAs → "c⊥A2+"];
  DefConstantSymbol[cPerpA2m, PrintAs → "c⊥A2-"];

  DefConstantSymbol[cPerpB0p, PrintAs → "c⊥b0+"];
  DefConstantSymbol[cPerpB0m, PrintAs → "c⊥b0-"];
  DefConstantSymbol[cPerpB1p, PrintAs → "c⊥b1+"];
  DefConstantSymbol[cPerpB1m, PrintAs → "c⊥b1-"];
  DefConstantSymbol[cPerpB2p, PrintAs → "c⊥b2+"];
  DefConstantSymbol[cPerpB2m, PrintAs → "c⊥b2-"];

  If[ProjectionNormalisationsToggle,
    Solutions = {};
    tmp =
     PB0pT[-n, -m, a, c] - cPerpB0p PB0p[g, h] PBPara[-g, -h, -n, -m] PB0p[e, f] PBPara[
           -e, -f, a, c] /. NewPO3TActivate /. PO3PiActivate /.
        PADMPiActivate /. PADMActivate // ToCanonical // CollectTensors;
    Solutions = Join[Solutions, Solve[ToConstantSymbolEquations[tmp == 0],
       cPerpB0p][[1]]];
    tmp = PB1pT[-n, -m, a, c] - cPerpB1p PB1p[-x, -y, g, h] PBPara[-g, -h, -n, -m] PB1p[
           x, y, e, f] PBPara[-e, -f, a, c] /. NewPO3TActivate /. PO3PiActivate /.
        PADMPiActivate /. PADMActivate // ToCanonical // CollectTensors;
    Solutions = Join[Solutions, Solve[ToConstantSymbolEquations[tmp == 0],
       cPerpB1p][[1]]];
    tmp = PB1mT[-n, -m, a, c] - cPerpB1m PB1m[-x, h] PBPerp[-h, -n, -m]
          PB1m[x, f] PBPerp[-f, a, c] /. NewPO3TActivate /. PO3PiActivate /.
        PADMPiActivate /. PADMActivate // ToCanonical // CollectTensors;
```

```
Solutions = Join[Solutions, Solve[ToConstantSymbolEquations[tmp == 0],
     cPerpB1m][[1]]];
tmp = PB2pT[-n, -m, a, c] - cPerpB2p PB2p[-x, -y, g, h] PBPara[-g, -h, -n, -m] PB2p[
           x, y, e, f] PBPara[-e, -f, a, c] /. NewPO3TActivate /. PO3PiActivate /.
      PADMPiActivate /. PADMActivate // ToCanonical // CollectTensors;
Solutions = Join[Solutions, Solve[ToConstantSymbolEquations[tmp == 0],
     cPerpB2p][[1]]];
tmp = Antisymmetrize[Antisymmetrize[PA0pT[-n, -m, -o, a, b, c] - cPerpA0p PA0p[g, h]
            PAPerp[-g, -h, -n, -m, -o] PA0p[e, f] PAPerp[-e, -f, a, b, c],
          {-n, -m}], {a, b}] /. NewPO3TActivate /. PO3PiActivate /.
      PADMPiActivate /. PADMActivate // ToCanonical // CollectTensors;
Solutions = Join[Solutions, Solve[ToConstantSymbolEquations[tmp == 0],
     cPerpA0p][[1]]];
tmp = Antisymmetrize[Antisymmetrize[PA0mT[-n, -m, -o, a, b, c] -
           cPerpA0m PA0m[g, h, i] PAPara[-g, -h, -i, -n, -m, -o]
            PA0m[e, f, j] PAPara[-e, -f, -j, a, b, c], {-n, -m}], {a, b}] /.
       NewPO3TActivate /. PO3PiActivate /. PADMPiActivate /.
     PADMActivate // ToCanonical // CollectTensors;
Solutions = Join[Solutions, Solve[ToConstantSymbolEquations[tmp == 0],
     cPerpA0m][[1]]];
tmp = Antisymmetrize[Antisymmetrize[PA1pT[-n, -m, -o, a, b, c] -
           cPerpA1p PA1p[-x, -y, g, h] PAPerp[-g, -h, -n, -m, -o]
            PA1p[x, y, e, f] PAPerp[-e, -f, a, b, c], {-n, -m}], {a, b}] /.
       NewPO3TActivate /. PO3PiActivate /. PADMPiActivate /.
     PADMActivate // ToCanonical // CollectTensors;
Solutions = Join[Solutions, Solve[ToConstantSymbolEquations[tmp == 0],
     cPerpA1p][[1]]];
tmp = Antisymmetrize[Antisymmetrize[PA1mT[-n, -m, -o, a, b, c] -
           cPerpA1m PA1m[-x, g, h, i] PAPara[-g, -h, -i, -n, -m, -o]
            PA1m[x, e, f, j] PAPara[-e, -f, -j, a, b, c], {-n, -m}], {a, b}] /.
       NewPO3TActivate /. PO3PiActivate /. PADMPiActivate /.
     PADMActivate // ToCanonical // CollectTensors;
Solutions = Join[Solutions, Solve[ToConstantSymbolEquations[tmp == 0],
     cPerpA1m][[1]]];
tmp = Antisymmetrize[Antisymmetrize[PA2pT[-n, -m, -o, a, b, c] -
           cPerpA2p PA2p[-x, -y, g, h] PAPerp[-g, -h, -n, -m, -o]
            PA2p[x, y, e, f] PAPerp[-e, -f, a, b, c], {-n, -m}], {a, b}] /.
       NewPO3TActivate /. PO3PiActivate /. PADMPiActivate /.
     PADMActivate // ToCanonical // CollectTensors;
Solutions = Join[Solutions, Solve[ToConstantSymbolEquations[tmp == 0],
     cPerpA2p][[1]]];
tmp = Antisymmetrize[Antisymmetrize[PA2mT[-n, -m, -o, a, b, c] -
           cPerpA2m PA2m[-x, -y, -z, g, h, i] PAPara[-g, -h, -i, -n, -m, -o]
            PA2m[x, y, z, e, f, j] PAPara[-e, -f, -j, a, b, c], {-n, -m}], {a, b}] /.
```

```
            NewPO3TActivate /. PO3PiActivate /. PADMPiActivate /.
         PADMActivate // ToCanonical // CollectTensors;
    Solutions = Join[Solutions, Solve[ToConstantSymbolEquations[tmp == 0],
         cPerpA2m][[1]]];
    TocPerp = Solutions;

    DumpSave[NotebookDirectory[] <> "mx_cache/O3Differences.mx", {TocPerp}];
    Quit[];
   ];
  MyImport["O3Differences.mx"];

  (*interlude to check some normalisations*)
  If[ProjectionNormalisationsCheckToggle,
    Print[Style["B0p", Blue, 20]];
    tmp =
     PB0p[g, h] PBPara[-g, -h, -n, -m] PB0p[e, f] PBPara[-e, -f, n, m] - (1 / cPerpB0p) /.
            TocPerp /. NewPO3TActivate /. PO3PiActivate /.
          PADMPiActivate /. PADMActivate // ToCanonical // CollectTensors;
    Print[tmp];
    Print[Style["B1p", Blue, 20]];
    tmp =
     PB1p[-x, -y, g, h] PBPara[-g, -h, -n, -m] PB1p[u, v, e, f] PBPara[-e, -f, n, m] -
            (1 / cPerpB1p) Antisymmetrize[Antisymmetrize[PPara[-x, u] PPara[-y, v],
               {-x, -y}], {u, v}] /. TocPerp /. NewPO3TActivate /. PO3PiActivate /.
          PADMPiActivate /. PADMActivate // ToCanonical // CollectTensors;
    Print[tmp];
    Print[Style["B1m", Blue, 20]];
    tmp =
     PB1m[-x, h] PBPerp[-h, -n, -m] PB1m[u, f] PBPerp[-f, n, m] - (1 / cPerpB1m) PPara[
              -x, u] /. TocPerp /. NewPO3TActivate /. PO3PiActivate /.
          PADMPiActivate /. PADMActivate // ToCanonical // CollectTensors;
    Print[tmp];
    Print[Style["B2p", Blue, 20]];
    tmp =
     PB2p[-x, -y, g, h] PBPara[-g, -h, -n, -m] PB2p[u, v, e, f] PBPara[-e, -f, n, m] -
            (1 / cPerpB2p) Symmetrize[Symmetrize[PPara[-x, u] PPara[-y, v],
               {-x, -y}], {u, v}] /. TocPerp /. NewPO3TActivate /. PO3PiActivate /.
          PADMPiActivate /. PADMActivate // ToCanonical // CollectTensors;
    Print[tmp];
    Print[Style["A0p", Red, 20]];
    tmp =
     Antisymmetrize[ PA0p[g, h] PAPerp[-g, -h, -n, -m, -o], {-n, -m}] PA0p[e, f] PAPerp[
              -e, -f, a, b, c] G[n, -a] G[m, -b] G[o, -c] - (1 / cPerpA0p) /. TocPerp /.
```

```
        NewPO3TActivate /. PO3PiActivate /. PADMPiActivate /.
     PADMActivate // ToCanonical // CollectTensors;
Print[tmp];
Print[Style["A0m", Red, 20]];
tmp =
 Antisymmetrize[PA0m[g, h, i] PAPara[-g, -h, -i, -n, -m, -o], {-n, -m}] PA0m[e,
          f, j] PAPara[-e, -f, -j, a, b, c] G[n, -a] G[m, -b] G[o, -c] -
        (1/cPerpA0m) /. TocPerp /. NewPO3TActivate /. PO3PiActivate /.
      PADMPiActivate /. PADMActivate // ToCanonical // CollectTensors;
Print[tmp];
Print[Style["A1p", Red, 20]];
tmp =
 Antisymmetrize[PA1p[-x, -y, g, h] PAPerp[-g, -h, -n, -m, -o], {-n, -m}] PA1p[u, v,
          e, f] PAPerp[-e, -f, a, b, c] G[n, -a] G[m, -b] G[o, -c] - (1/cPerpA1p)
         Antisymmetrize[Antisymmetrize[PPara[-x, u] PPara[-y, v], {-x, -y}],
          {u, v}] /. TocPerp /. NewPO3TActivate /. PO3PiActivate /.
      PADMPiActivate /. PADMActivate // ToCanonical // CollectTensors;
Print[tmp];
Print[Style["A1m", Red, 20]];
tmp =
 Antisymmetrize[PA1m[-x, g, h, i] PAPara[-g, -h, -i, -n, -m, -o], {-n, -m}] PA1m[
          u, e, f, j] PAPara[-e, -f, -j, a, b, c] G[n, -a]
         G[m, -b] G[o, -c] - (1/cPerpA1m) PPara[-x, u] /. TocPerp /.
        NewPO3TActivate /. PO3PiActivate /. PADMPiActivate /.
     PADMActivate // ToCanonical // CollectTensors;
Print[tmp];
Print[Style["A2p", Red, 20]];
tmp =
 Antisymmetrize[PA2p[-x, -y, g, h] PAPerp[-g, -h, -n, -m, -o], {-n, -m}] PA2p[u, v,
          e, f] PAPerp[-e, -f, a, b, c] G[n, -a] G[m, -b] G[o, -c] - (1/cPerpA2p)
         Symmetrize[Symmetrize[PPara[-x, u] PPara[-y, v], {-x, -y}], {u, v}] /.
        TocPerp /. NewPO3TActivate /. PO3PiActivate /. PADMPiActivate /.
     PADMActivate // ToCanonical // CollectTensors;
Print[tmp];
Print[Style["A2m", Red, 20]];
tmp =
 Antisymmetrize[ PA2m[-x, -y, -z, g, h, i] PAPara[-g, -h, -i, -n, -m, -o], {-n, -m}]
         PA2m[u, v, w, e, f, j] PAPara[-e, -f, -j, a, b, c] G[n, -a]
         G[m, -b] G[o, -c] - (1/cPerpA2m) Antisymmetrize[Antisymmetrize[
           PPara[-x, u] PPara[-y, v] PPara[-z, w], {-x, -y}], {u, v}] /. TocPerp /.
        NewPO3TActivate /. PO3PiActivate /. PADMPiActivate /.
     PADMActivate // ToCanonical // CollectTensors;
Print[tmp];
```

```
  Quit[];
 ];
```

Transfer couplings $\{\hat{\alpha}_A^{\perp\perp}\}$, $\{\hat{\beta}_E^{\perp\perp}\}$

```
In[•]:= DefConstantSymbol[BetPerpPerp0p, PrintAs → "β̂0+⊥⊥"];
DefConstantSymbol[BetPerpPerp0m, PrintAs → "β̂0-⊥⊥"];
DefConstantSymbol[BetPerpPerp1p, PrintAs → "β̂1+⊥⊥"];
DefConstantSymbol[BetPerpPerp1m, PrintAs → "β̂1-⊥⊥"];
DefConstantSymbol[BetPerpPerp2p, PrintAs → "β̂2+⊥⊥"];
DefConstantSymbol[BetPerpPerp2m, PrintAs → "β̂2-⊥⊥"];

BetPerpPerp = {BetPerpPerp0p, BetPerpPerp0m,
   BetPerpPerp1p, BetPerpPerp1m, BetPerpPerp2p, BetPerpPerp2m};

DefConstantSymbol[AlpPerpPerp0p, PrintAs → "α̂0+⊥⊥"];
DefConstantSymbol[AlpPerpPerp0m, PrintAs → "α̂0-⊥⊥"];
DefConstantSymbol[AlpPerpPerp1p, PrintAs → "α̂1+⊥⊥"];
DefConstantSymbol[AlpPerpPerp1m, PrintAs → "α̂1-⊥⊥"];
DefConstantSymbol[AlpPerpPerp2p, PrintAs → "α̂2+⊥⊥"];
DefConstantSymbol[AlpPerpPerp2m, PrintAs → "α̂2-⊥⊥"];

AlpPerpPerp = {AlpPerpPerp0p, AlpPerpPerp0m,
   AlpPerpPerp1p, AlpPerpPerp1m, AlpPerpPerp2p, AlpPerpPerp2m};

If[TransferCouplingsPerpPerpToggle,
  TransferCouplingsPerpPerpSolutions = {};
  tmp =
   BetPerpPerp0p PB0p[g, h] PBPara[-g, -h, a, e] - PB0p[x, z] PBPara[-x, -z, i, f] V[g]
           PPara[-f, h] V[-c] PPara[e, -d] (Bet1 PT1[-i, -g, -h, a, c, d] +
             Bet2 PT2[-i, -g, -h, a, c, d] + Bet3 PT3[-i, -g, -h, a, c, d]) /.
         PO3PiActivate /. PActivate /. PADMPiActivate /.
      PADMActivate // ToCanonical // CollectTensors;
  TransferCouplingsPerpPerpSolutions = Join[TransferCouplingsPerpPerpSolutions,
    Solve[ToConstantSymbolEquations[tmp == 0], BetPerpPerp0p][[1]]];
  tmp = BetPerpPerp1p PB1p[-q, -r, g, h] PBPara[-g, -h, a, e] -
          PB1p[-q, -r, x, z] PBPara[-x, -z, i, f] V[g] PPara[-f, h] V[-c]
           PPara[e, -d] (Bet1 PT1[-i, -g, -h, a, c, d] + Bet2 PT2[-i, -g, -h, a, c, d] +
             Bet3 PT3[-i, -g, -h, a, c, d]) /. PO3PiActivate /. PActivate /.
       PADMPiActivate /. PADMActivate // ToCanonical // CollectTensors;
  TransferCouplingsPerpPerpSolutions = Join[TransferCouplingsPerpPerpSolutions,
    Solve[ToConstantSymbolEquations[tmp == 0], BetPerpPerp1p][[1]]];
  tmp = BetPerpPerp1m PB1m[-q, h] PBPerp[-h, a, e] - PB1m[-q, z] PBPerp[-z, i, f]
```

```
            V[g] PPara[-f, h] V[-c] PPara[e, -d] (Bet1 PT1[-i, -g, -h, a, c, d] +
              Bet2 PT2[-i, -g, -h, a, c, d] + Bet3 PT3[-i, -g, -h, a, c, d]) /.
          PO3PiActivate /. PActivate /. PADMPiActivate /.
       PADMActivate // ToCanonical // CollectTensors;
TransferCouplingsPerpPerpSolutions = Join[TransferCouplingsPerpPerpSolutions,
   Solve[ToConstantSymbolEquations[tmp == 0], BetPerpPerp1m][[1]]];
tmp = BetPerpPerp2p PB2p[-q, -r, g, h] PBPara[-g, -h, a, e] -
          PB2p[-q, -r, x, z] PBPara[-x, -z, i, f] V[g] PPara[-f, h] V[-c]
            PPara[e, -d] (Bet1 PT1[-i, -g, -h, a, c, d] + Bet2 PT2[-i, -g, -h, a, c, d] +
              Bet3 PT3[-i, -g, -h, a, c, d]) /. PO3PiActivate /. PActivate /.
        PADMPiActivate /. PADMActivate // ToCanonical // CollectTensors;
TransferCouplingsPerpPerpSolutions = Join[TransferCouplingsPerpPerpSolutions,
   Solve[ToConstantSymbolEquations[tmp == 0], BetPerpPerp2p][[1]]];
tmp = AlpPerpPerp0p PA0p[g, h] Antisymmetrize[PAPerp[-g, -h, a, b, e], {a, b}] -
          PA0p[x, z] PAPerp[-x, -z, i, j, f] V[g] PPara[-f, h] V[-c] PPara[e, -d]
            (Alp1 PR1[-i, -j, -g, -h, a, b, c, d] + Alp2 PR2[-i, -j, -g, -h, a, b, c, d] +
              Alp3 PR3[-i, -j, -g, -h, a, b, c, d] + Alp4
               PR4[-i, -j, -g, -h, a, b, c, d] + Alp5 PR5[-i, -j, -g, -h, a, b, c, d] +
              Alp6 PR6[-i, -j, -g, -h, a, b, c, d]) /. PO3PiActivate /. PActivate /.
        PADMPiActivate /. PADMActivate // ToCanonical // CollectTensors;
TransferCouplingsPerpPerpSolutions = Join[TransferCouplingsPerpPerpSolutions,
   Solve[ToConstantSymbolEquations[tmp == 0], AlpPerpPerp0p][[1]]];
tmp = AlpPerpPerp0m PA0m[g, h, i] Antisymmetrize[PAPara[-g, -h, -i, a, b, e],
             {a, b}] - PA0m[x, y, z] PAPara[-x, -y, -z, i, j, f] V[g] PPara[-f, h]
            V[-c] PPara[e, -d] (Alp1 PR1[-i, -j, -g, -h, a, b, c, d] + Alp2 PR2[-i,
                -j, -g, -h, a, b, c, d] + Alp3 PR3[-i, -j, -g, -h, a, b, c, d] + Alp4
               PR4[-i, -j, -g, -h, a, b, c, d] + Alp5 PR5[-i, -j, -g, -h, a, b, c, d] +
              Alp6 PR6[-i, -j, -g, -h, a, b, c, d]) /. PO3PiActivate /. PActivate /.
        PADMPiActivate /. PADMActivate // ToCanonical // CollectTensors;
TransferCouplingsPerpPerpSolutions = Join[TransferCouplingsPerpPerpSolutions,
   Solve[ToConstantSymbolEquations[tmp == 0], AlpPerpPerp0m][[1]]];
tmp = AlpPerpPerp1p PA1p[-p, -q, g, h] Antisymmetrize[
             PAPerp[-g, -h, a, b, e], {a, b}] - PA1p[-p, -q, x, z]
            PAPerp[-x, -z, i, j, f] V[g] PPara[-f, h] V[-c] PPara[e, -d]
            (Alp1 PR1[-i, -j, -g, -h, a, b, c, d] + Alp2 PR2[-i, -j, -g, -h, a, b, c, d] +
              Alp3 PR3[-i, -j, -g, -h, a, b, c, d] + Alp4 PR4[-i, -j, -g,
                -h, a, b, c, d] + Alp5 PR5[-i, -j, -g, -h, a, b, c, d] +
              Alp6 PR6[-i, -j, -g, -h, a, b, c, d]) /. PO3PiActivate /. PActivate /.
        PADMPiActivate /. PADMActivate // ToCanonical // CollectTensors;
TransferCouplingsPerpPerpSolutions = Join[TransferCouplingsPerpPerpSolutions,
   Solve[ToConstantSymbolEquations[tmp == 0], AlpPerpPerp1p][[1]]];
tmp = AlpPerpPerp1m PA1m[-p, g, h, i] Antisymmetrize[
             PAPara[-g, -h, -i, a, b, e], {a, b}] - PA1m[-p, x, y, z]
```

```
          PAPara[-x, -y, -z, i, j, f] V[g] PPara[-f, h] V[-c] PPara[e, -d]
          (Alp1 PR1[-i, -j, -g, -h, a, b, c, d] + Alp2 PR2[-i, -j, -g, -h, a, b, c, d] +
            Alp3 PR3[-i, -j, -g, -h, a, b, c, d] + Alp4 PR4[-i, -j, -g,
              -h, a, b, c, d] + Alp5 PR5[-i, -j, -g, -h, a, b, c, d] +
            Alp6 PR6[-i, -j, -g, -h, a, b, c, d]) /. PO3PiActivate /. PActivate /.
      PADMPiActivate /. PADMActivate // ToCanonical // CollectTensors;
  TransferCouplingsPerpPerpSolutions = Join[TransferCouplingsPerpPerpSolutions,
    Solve[ToConstantSymbolEquations[tmp == 0], AlpPerpPerp1m][[1]]];
  tmp = AlpPerpPerp2p PA2p[-p, -q, g, h] Antisymmetrize[
           PAPerp[-g, -h, a, b, e], {a, b}] - PA2p[-p, -q, x, z]
          PAPerp[-x, -z, i, j, f] V[g] PPara[-f, h] V[-c] PPara[e, -d]
          (Alp1 PR1[-i, -j, -g, -h, a, b, c, d] + Alp2 PR2[-i, -j, -g, -h, a, b, c, d] +
            Alp3 PR3[-i, -j, -g, -h, a, b, c, d] + Alp4 PR4[-i, -j, -g,
              -h, a, b, c, d] + Alp5 PR5[-i, -j, -g, -h, a, b, c, d] +
            Alp6 PR6[-i, -j, -g, -h, a, b, c, d]) /. PO3PiActivate /. PActivate /.
      PADMPiActivate /. PADMActivate // ToCanonical // CollectTensors;
  TransferCouplingsPerpPerpSolutions = Join[TransferCouplingsPerpPerpSolutions,
    Solve[ToConstantSymbolEquations[tmp == 0], AlpPerpPerp2p][[1]]];
  tmp = AlpPerpPerp2m PA2m[-q, -p, -r, g, h, i] Antisymmetrize[
           PAPara[-g, -h, -i, a, b, e], {a, b}] - PA2m[-q, -p, -r, x, y, z]
          PAPara[-x, -y, -z, i, j, f] V[g] PPara[-f, h] V[-c] PPara[e, -d]
          (Alp1 PR1[-i, -j, -g, -h, a, b, c, d] + Alp2 PR2[-i, -j, -g, -h, a, b, c, d] +
            Alp3 PR3[-i, -j, -g, -h, a, b, c, d] + Alp4
             PR4[-i, -j, -g, -h, a, b, c, d] + Alp5 PR5[-i, -j, -g, -h, a, b, c, d] +
            Alp6 PR6[-i, -j, -g, -h, a, b, c, d]) /. PO3PiActivate /. PActivate /.
      PADMPiActivate /. PADMActivate // ToCanonical // CollectTensors;
  TransferCouplingsPerpPerpSolutions = Join[TransferCouplingsPerpPerpSolutions,
    Solve[ToConstantSymbolEquations[tmp == 0], AlpPerpPerp2m][[1]]];

  DumpSave[NotebookDirectory[] <> "mx_cache/TransferCouplingsPerpPerp.mx",
   {TransferCouplingsPerpPerpSolutions}];
  Quit[];
 ];

(*MyImport["TransferCouplingsPerpPerp.mx"];*)
```

Basic form ϕbJ^P, ϕAJ^P

```
In[•]:= DefTensor[PhiB0p[], M4, PrintAs → "b0+ϕ"];
        DeclareOrder[PhiB0p[], 1];
        DefTensor[PhiB1p[-a, -b], M4, Antisymmetric[{-a, -b}], PrintAs → "b1+ϕ"];

          DeclareOrder[PhiB1p[-a, -b], 1];
```

```
DefTensor[PhiB1m[-a], M4, PrintAs → "b1⁻ϕ"];
DeclareOrder[PhiB1m[-a], 1];
DefTensor[PhiB2p[-a, -b], M4, Symmetric[{-a, -b}], PrintAs → "ϕ"];
DeclareOrder[PhiB2p[-a, -b], 1];
DefTensor[PhiA0p[], M4, PrintAs → "A0⁺ϕ"];
DeclareOrder[PhiA0p[], 1, "IsUnityWithEHTerm" → True];
DefTensor[PhiA0m[], M4, PrintAs → "A0⁻ϕ"];
DeclareOrder[PhiA0m[], 1];
DefTensor[PhiA1p[-a, -b], M4, Antisymmetric[{-a, -b}], PrintAs → "ϕ"];
DeclareOrder[PhiA1p[-a, -b], 1];
DefTensor[PhiA1m[-a], M4, PrintAs → "A1⁻ϕ"];
DeclareOrder[PhiA1m[-a], 1];
DefTensor[PhiA2p[-a, -b], M4, Symmetric[{-a, -b}], PrintAs → "A2⁺ϕ"];
DeclareOrder[PhiA2p[-a, -b], 1];
DefTensor[PhiA2m[-a, -b, -c], M4, Antisymmetric[{-a, -b}], PrintAs → "A2⁻ϕ"];
DeclareOrder[PhiA2m[-a, -b, -c], 1];
AutomaticRules[PhiA2m,
  MakeRule[{PhiA2m[a, -b, -a], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[PhiA2m, MakeRule[{epsilonG[a, b, c, d] PhiA2m[-a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];

DefTensor[BPhi[-a, -c], M4];
DeclareOrder[BPhi[-a, -c], 1];
BPhiDefinition = Ji[] BPi[-i, z] G3[-z, a] B[-k, -a] -
   4 V[g] B[-k, -o] G3[o, -z] H[h, z]
    (Bet1 PT1[-i, -g, -h, a, c, d] + Bet2 PT2[-i, -g, -h, a, c, d] +
      Bet3 PT3[-i, -g, -h, a, c, d]) PPara[-c, m] PPara[-d, n] T[-a, -m, -n] -
   2 V[g] B[-k, -o] G3[o, -z] H[h, z] (cBet1 PT1[-i, -g, -h, a, c, d] +
      cBet2 PT2[-i, -g, -h, a, c, d] + cBet3 PT3[-i, -g, -h, a, c, d])
    PPara[-c, m] PPara[-d, n] TLambda[-a, -m, -n] -
   2 V[g] B[-k, -o] G3[o, -z] H[h, z] (cBet1 PT1[-i, -g, -h, a, c, d] +
      cBet2 PT2[-i, -g, -h, a, c, d] + cBet3 PT3[-i, -g, -h, a, c, d])
    (PPerp[-c, m] PPara[-d, n] TLambda[-a, -m, -n] +
      PPara[-c, m] PPerp[-d, n] TLambda[-a, -m, -n]);
BPhiActivate = MakeRule[{BPhi[-i, -k], Evaluate[BPhiDefinition]},
   MetricOn → All, ContractMetrics → True];

DefTensor[APhi[-a, -b, -c], M4, Antisymmetric[{-a, -b}]];
DeclareOrder[APhi[-a, -b, -c], 1, "IsUnityWithEHTerm" → True];
APhiDefinition = Ji[] APi[-i, -j, z] G3[-z, a] B[-k, -a] +
   2 Alp0 Antisymmetrize[ V[-i] PPara[-j, -k], {-i, -j}] -
   8 V[g] B[-k, -o] G3[o, -z] H[h, z]
    (Alp1 PR1[-i, -j, -g, -h, a, b, c, d] + Alp2 PR2[-i, -j, -g, -h, a, b, c, d] +
```

```
      Alp3 PR3[-i, -j, -g, -h, a, b, c, d] + Alp4 PR4[-i, -j, -g, -h, a, b, c, d] +
      Alp5 PR5[-i, -j, -g, -h, a, b, c, d] + Alp6 PR6[-i, -j, -g, -h, a, b, c, d])
    PPara[-c, m] PPara[-d, n] R[-a, -b, -m, -n] - 4 V[g] B[-k, -o] G3[o, -z] H[h, z]
    (cAlp1 PR1[-i, -j, -g, -h, a, b, c, d] + cAlp2 PR2[-i, -j, -g, -h, a, b, c, d] +
      cAlp3 PR3[-i, -j, -g, -h, a, b, c, d] + cAlp4 PR4[-i, -j, -g, -h, a, b, c, d] +
      cAlp5 PR5[-i, -j, -g, -h, a, b, c, d] + cAlp6 PR6[-i, -j, -g, -h, a, b, c, d])
    PPara[-c, m] PPara[-d, n] RLambda[-a, -b, -m, -n] -
   4 V[g] B[-k, -o] G3[o, -z] H[h, z]
    (cAlp1 PR1[-i, -j, -g, -h, a, b, c, d] + cAlp2 PR2[-i, -j, -g, -h, a, b, c, d] +
      cAlp3 PR3[-i, -j, -g, -h, a, b, c, d] + cAlp4 PR4[-i, -j, -g, -h, a, b, c, d] +
      cAlp5 PR5[-i, -j, -g, -h, a, b, c, d] + cAlp6 PR6[-i, -j, -g, -h, a, b, c, d])
    (PPerp[-c, m] PPara[-d, n] RLambda[-a, -b, -m, -n] +
      PPara[-c, m] PPerp[-d, n] RLambda[-a, -b, -m, -n]);
APhiActivate = MakeRule[{APhi[-i, -j, -k], Evaluate[APhiDefinition]},
   MetricOn → All, ContractMetrics → True];
If[CanonicalPhiToggle,
  PhiB0pDefinition = PB0p[e, f] PBPara[-e, -f, a, c] BPhi[-a, -c] /. BPhiActivate //
    ActivateGeneralO3Projections;
  PhiB1pDefinition = PB1p[-n, -m, e, f] PBPara[-e, -f, a, c] BPhi[-a, -c] /.
     BPhiActivate // ActivateGeneralO3Projections;
  PhiB2pDefinition = PB2p[-n, -m, e, f] PBPara[-e, -f, a, c] BPhi[-a, -c] /.
     BPhiActivate // ActivateGeneralO3Projections;
  PhiB1mDefinition = PB1m[-n, f] PBPerp[-f, a, c] BPhi[-a, -c] /. BPhiActivate //
    ActivateGeneralO3Projections;

  PhiA0pDefinition =
   PA0p[e, f] PAPerp[-e, -f, a, b, c] APhi[-a, -b, -c] /. APhiActivate //
    ActivateGeneralO3Projections;
  PhiA1pDefinition = PA1p[-n, -m, e, f] PAPerp[-e, -f, a, b, c] APhi[-a, -b, -c] /.
     APhiActivate // ActivateGeneralO3Projections;
  PhiA2pDefinition = PA2p[-n, -m, e, f] PAPerp[-e, -f, a, b, c] APhi[-a, -b, -c] /.
     APhiActivate // ActivateGeneralO3Projections;
  PhiA0mDefinition = PA0m[d, e, f] PAPara[-d, -e, -f, a, b, c] APhi[-a, -b, -c] /.
     APhiActivate // ActivateGeneralO3Projections;
  PhiA1mDefinition = PA1m[-n, d, e, f] PAPara[-d, -e, -f, a, b, c] APhi[-a, -b, -c] /.
     APhiActivate // ActivateGeneralO3Projections;
  PhiA2mDefinition = PA2m[-n, -m, -o, d, e, f] PAPara[-d, -e, -f, a, b, c]
      APhi[-a, -b, -c] /. APhiActivate // ActivateGeneralO3Projections;

  PhiB0pActivate = MakeRule[{PhiB0p[], Scalar[Evaluate[PhiB0pDefinition]]},
    MetricOn → All, ContractMetrics → True];
  PhiB1pActivate = MakeRule[{PhiB1p[-n, -m], Evaluate[PhiB1pDefinition]},
    MetricOn → All, ContractMetrics → True];
```

```
    PhiB1mActivate = MakeRule[{PhiB1m[-n], Evaluate[PhiB1mDefinition]},
      MetricOn → All, ContractMetrics → True];
    PhiB2pActivate = MakeRule[{PhiB2p[-n, -m], Evaluate[PhiB2pDefinition]},
      MetricOn → All, ContractMetrics → True];
    PhiA0pActivate = MakeRule[{PhiA0p[], Scalar[Evaluate[PhiA0pDefinition]]},
      MetricOn → All, ContractMetrics → True];
    PhiA0mActivate = MakeRule[{PhiA0m[], Scalar[Evaluate[PhiA0mDefinition]]},
      MetricOn → All, ContractMetrics → True];
    PhiA1pActivate = MakeRule[{PhiA1p[-n, -m], Evaluate[PhiA1pDefinition]},
      MetricOn → All, ContractMetrics → True];
    PhiA1mActivate = MakeRule[{PhiA1m[-n], Evaluate[PhiA1mDefinition]},
      MetricOn → All, ContractMetrics → True];
    PhiA2pActivate = MakeRule[{PhiA2p[-n, -m], Evaluate[PhiA2pDefinition]},
      MetricOn → All, ContractMetrics → True];
    PhiA2mActivate = MakeRule[{PhiA2m[-n, -m, -o], Evaluate[PhiA2mDefinition]},
      MetricOn → All, ContractMetrics → True];

    PhiActivate = Join[PhiB0pActivate, PhiB1pActivate,
      PhiB1mActivate, PhiB2pActivate, PhiA0pActivate, PhiA0mActivate,
      PhiA1pActivate, PhiA1mActivate, PhiA2pActivate, PhiA2mActivate];

    DumpSave[NotebookDirectory[] <> "mx_cache/phiactivate.mx", {PhiActivate}];
    Print["done phiactivate"];
    Quit[];
   ];
  MyImport["phiactivate.mx"];
```

Basic form ¬ ϕbJ^P, ¬ ϕAJ^P

```
In[∘]:= BPhiNonCanonicalDefinition = 4 V[g] B[-k, -o] G3[o, -z]
        H[h, z] (Bet1 PT1[-i, -g, -h, a, c, d] + Bet2 PT2[-i, -g, -h, a, c, d] +
          Bet3 PT3[-i, -g, -h, a, c, d]) (PPerp[-c, m] PPara[-d, n] T[-a, -m, -n] +
          PPara[-c, m] PPerp[-d, n] T[-a, -m, -n]);
  BPhiNonCanonicalActivate = MakeRule[{BPhi[-i, -k],
      Evaluate[BPhiNonCanonicalDefinition]}, MetricOn → All, ContractMetrics → True];

  APhiNonCanonicalDefinition = 8 V[g] B[-k, -o] G3[o, -z] H[h, z]
       (Alp1 PR1[-i, -j, -g, -h, a, b, c, d] + Alp2 PR2[-i, -j, -g, -h, a, b, c, d] +
         Alp3 PR3[-i, -j, -g, -h, a, b, c, d] + Alp4 PR4[-i, -j, -g, -h, a, b, c, d] +
         Alp5 PR5[-i, -j, -g, -h, a, b, c, d] + Alp6 PR6[-i, -j, -g, -h, a, b, c, d])
       (PPerp[-c, m] PPara[-d, n] R[-a, -b, -m, -n] +
         PPara[-c, m] PPerp[-d, n] R[-a, -b, -m, -n]);
  APhiNonCanonicalActivate = MakeRule[{APhi[-i, -j, -k],
```

```
      Evaluate[APhiNonCanonicalDefinition]}, MetricOn → All, ContractMetrics → True];
  If[NonCanonicalPhiToggle,
    PhiNonCanonicalB0pDefinition = PB0p[e, f] PBPara[-e, -f, a, c] BPhi[-a, -c] /.
       BPhiNonCanonicalActivate // ActivateGeneralO3Projections;
    PhiNonCanonicalB1pDefinition = PB1p[-n, -m, e, f] PBPara[-e, -f, a, c]
        BPhi[-a, -c] /. BPhiNonCanonicalActivate // ActivateGeneralO3Projections;
    PhiNonCanonicalB2pDefinition = PB2p[-n, -m, e, f] PBPara[-e, -f, a, c]
        BPhi[-a, -c] /. BPhiNonCanonicalActivate // ActivateGeneralO3Projections;
    PhiNonCanonicalB1mDefinition = PB1m[-n, f] PBPerp[-f, a, c] BPhi[-a, -c] /.
       BPhiNonCanonicalActivate // ActivateGeneralO3Projections;

    PhiNonCanonicalA0pDefinition = PA0p[e, f] PAPerp[-e, -f, a, b, c] APhi[-a, -b, -c] /.
       APhiNonCanonicalActivate // ActivateGeneralO3Projections;
    PhiNonCanonicalA1pDefinition = PA1p[-n, -m, e, f] PAPerp[-e, -f, a, b, c] APhi[
         -a, -b, -c] /. APhiNonCanonicalActivate // ActivateGeneralO3Projections;
    PhiNonCanonicalA2pDefinition = PA2p[-n, -m, e, f] PAPerp[-e, -f, a, b, c] APhi[
         -a, -b, -c] /. APhiNonCanonicalActivate // ActivateGeneralO3Projections;
    PhiNonCanonicalA0mDefinition = PA0m[d, e, f] PAPara[-d, -e, -f, a, b, c] APhi[
         -a, -b, -c] /. APhiNonCanonicalActivate // ActivateGeneralO3Projections;
    PhiNonCanonicalA1mDefinition = PA1m[-n, d, e, f] PAPara[-d, -e, -f, a, b, c] APhi[
         -a, -b, -c] /. APhiNonCanonicalActivate // ActivateGeneralO3Projections;
    PhiNonCanonicalA2mDefinition = PA2m[-n, -m, -o, d, e, f]
        PAPara[-d, -e, -f, a, b, c] APhi[-a, -b, -c] /.
       APhiNonCanonicalActivate // ActivateGeneralO3Projections;

    PhiNonCanonicalB0pActivate =
     MakeRule[{PhiB0p[], Scalar[Evaluate[PhiNonCanonicalB0pDefinition]]},
      MetricOn → All, ContractMetrics → True];
    PhiNonCanonicalB1pActivate = MakeRule[{PhiB1p[-n, -m], Evaluate[
        PhiNonCanonicalB1pDefinition]}, MetricOn → All, ContractMetrics → True];
    PhiNonCanonicalB1mActivate = MakeRule[{PhiB1m[-n], Evaluate[
        PhiNonCanonicalB1mDefinition]}, MetricOn → All, ContractMetrics → True];
    PhiNonCanonicalB2pActivate = MakeRule[{PhiB2p[-n, -m], Evaluate[
        PhiNonCanonicalB2pDefinition]}, MetricOn → All, ContractMetrics → True];
    PhiNonCanonicalA0pActivate = MakeRule[{PhiA0p[],
       Scalar[Evaluate[PhiNonCanonicalA0pDefinition]]},
      MetricOn → All, ContractMetrics → True];
    PhiNonCanonicalA0mActivate = MakeRule[
      {PhiA0m[], Scalar[Evaluate[PhiNonCanonicalA0mDefinition]]},
      MetricOn → All, ContractMetrics → True];
    PhiNonCanonicalA1pActivate = MakeRule[{PhiA1p[-n, -m], Evaluate[
        PhiNonCanonicalA1pDefinition]}, MetricOn → All, ContractMetrics → True];
    PhiNonCanonicalA1mActivate = MakeRule[{PhiA1m[-n], Evaluate[
        PhiNonCanonicalA1mDefinition]}, MetricOn → All, ContractMetrics → True];
```

```
    PhiNonCanonicalA2pActivate = MakeRule[{PhiA2p[-n, -m], Evaluate[
        PhiNonCanonicalA2pDefinition]}, MetricOn → All, ContractMetrics → True];
    PhiNonCanonicalA2mActivate = MakeRule[{PhiA2m[-n, -m, -o], Evaluate[
        PhiNonCanonicalA2mDefinition]}, MetricOn → All, ContractMetrics → True];

    PhiNonCanonicalActivate =
     Join[PhiNonCanonicalB0pActivate, PhiNonCanonicalB1pActivate,
      PhiNonCanonicalB1mActivate, PhiNonCanonicalB2pActivate,
      PhiNonCanonicalA0pActivate, PhiNonCanonicalA0mActivate,
      PhiNonCanonicalA1pActivate, PhiNonCanonicalA1mActivate,
      PhiNonCanonicalA2pActivate, PhiNonCanonicalA2mActivate];

    DumpSave[NotebookDirectory[] <> "mx_cache/phinoncanonicalactivate.mx",
     {PhiNonCanonicalActivate}];
    Print["done phinoncanonicalactivate"];
    Quit[];
  ];
 MyImport["phinoncanonicalactivate.mx"];
```

Define χbJ^P, χAJ^P

```
In[]:= DefTensor[ChiB0p[], M4, PrintAs → "b0⁺χ"];
DeclareOrder[ChiB0p[], 1];
DefTensor[ChiB1p[-a, -b], M4, Antisymmetric[{-a, -b}], PrintAs → "b1⁺χ"];
DeclareOrder[ChiB1p[-a, -b], 1];
DefTensor[ChiB1m[-a], M4, PrintAs → "b1⁻χ"];
DeclareOrder[ChiB1m[-a], 1];
DefTensor[ChiB2p[-a, -b], M4, Symmetric[{-a, -b}], PrintAs → "b2⁺χ"];
DeclareOrder[ChiB2p[-a, -b], 1];
DefTensor[ChiA0p[], M4, PrintAs → "A0⁺χ"];
DeclareOrder[ChiA0p[], 1];
DefTensor[ChiA0m[], M4, PrintAs → "A0⁻χ"];
DeclareOrder[ChiA0m[], 1];
DefTensor[ChiA1p[-a, -b], M4, Antisymmetric[{-a, -b}], PrintAs → "A1⁺χ"];
DeclareOrder[ChiA1p[-a, -b], 1];
DefTensor[ChiA1m[-a], M4, PrintAs → "A1⁻χ"];
DeclareOrder[ChiA1m[-a], 1];
DefTensor[ChiA2p[-a, -b], M4, Symmetric[{-a, -b}], PrintAs → "A2⁺χ"];
DeclareOrder[ChiA2p[-a, -b], 1];
DefTensor[ChiA2m[-a, -b, -c], M4, Antisymmetric[{-a, -b}], PrintAs → "A2⁻χ"];
DeclareOrder[ChiA2m[-a, -b, -c], 1];
AutomaticRules[ChiA2m,
  MakeRule[{ChiA2m[a, -b, -a], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[ChiA2m, MakeRule[{epsilonG[a, b, c, d] ChiA2m[-a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];
```

Basic form $\chi^{\perp}$ bJ^{P}, $\chi^{\perp}$ AJ^{P}

```
In[]:= DefTensor[ChiPerpB0p[], M4, PrintAs → "b0⁺χ⊥"];
DeclareOrder[ChiPerpB0p[], 1];
DefTensor[ChiPerpB1p[-a, -b], M4, Antisymmetric[{-a, -b}], PrintAs → "b1⁺χ⊥"];
DeclareOrder[ChiPerpB1p[-a, -b], 1];
DefTensor[ChiPerpB1m[-a], M4, PrintAs → "b1⁻χ⊥"];
DeclareOrder[ChiPerpB1m[-a], 1];
DefTensor[ChiPerpB2p[-a, -b], M4, Symmetric[{-a, -b}], PrintAs → "b2⁺χ⊥"];
DeclareOrder[ChiPerpB2p[-a, -b], 1];
DefTensor[ChiPerpA0p[], M4, PrintAs → "A0⁺χ⊥"];
DeclareOrder[ChiPerpA0p[], 1];
DefTensor[ChiPerpA0m[], M4, PrintAs → "A0⁻χ⊥"];
DeclareOrder[ChiPerpA0m[], 1];
DefTensor[ChiPerpA1p[-a, -b], M4, Antisymmetric[{-a, -b}], PrintAs → "A1⁺χ⊥"];
DeclareOrder[ChiPerpA1p[-a, -b], 1];
DefTensor[ChiPerpA1m[-a], M4, PrintAs → "A1⁻χ⊥"];
```

```
DeclareOrder[ChiPerpA1m[-a], 1];
DefTensor[ChiPerpA2p[-a, -b], M4, Symmetric[{-a, -b}], PrintAs → "A2⁺χ⊥"];
DeclareOrder[ChiPerpA2p[-a, -b], 1];
DefTensor[ChiPerpA2m[-a, -b, -c], M4, Antisymmetric[{-a, -b}], PrintAs → "A2⁻χ⊥"];
DeclareOrder[ChiPerpA2m[-a, -b, -c], 1];
AutomaticRules[ChiPerpA2m,
  MakeRule[{ChiPerpA2m[a, -b, -a], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[ChiPerpA2m, MakeRule[{epsilonG[a, b, c, d] ChiPerpA2m[-a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];

DefTensor[BChiPerp[-a, -c], M4];
DeclareOrder[BChiPerp[-a, -c], 1];
BChiPerpDefinition = Ji[] BPi[-i, z] G3[-z, a] B[-k, -a] -
   2 V[g] B[-k, -o] G3[o, -z] H[h, z]
    (cBet1 PT1[-i, -g, -h, a, c, d] + cBet2 PT2[-i, -g, -h, a, c, d] +
      cBet3 PT3[-i, -g, -h, a, c, d]) PPara[-c, m] PPara[-d, n] TLambda[-a, -m, -n] -
   2 V[g] B[-k, -o] G3[o, -z] H[h, z] (cBet1 PT1[-i, -g, -h, a, c, d] +
      cBet2 PT2[-i, -g, -h, a, c, d] + cBet3 PT3[-i, -g, -h, a, c, d])
    (PPerp[-c, m] PPara[-d, n] TLambda[-a, -m, -n] +
      PPara[-c, m] PPerp[-d, n] TLambda[-a, -m, -n]);
BChiPerpActivate = MakeRule[{BChiPerp[-i, -k], Evaluate[BChiPerpDefinition]},
   MetricOn → All, ContractMetrics → True];

DefTensor[AChiPerp[-a, -b, -c], M4, Antisymmetric[{-a, -b}]];
DeclareOrder[AChiPerp[-a, -b, -c], 1];
AChiPerpDefinition =
  Ji[] APi[-i, -j, z] G3[-z, a] B[-k, -a] - 4 V[g] B[-k, -o] G3[o, -z] H[h, z]
    (cAlp1 PR1[-i, -j, -g, -h, a, b, c, d] + cAlp2 PR2[-i, -j, -g, -h, a, b, c, d] +
      cAlp3 PR3[-i, -j, -g, -h, a, b, c, d] + cAlp4 PR4[-i, -j, -g, -h, a, b, c, d] +
      cAlp5 PR5[-i, -j, -g, -h, a, b, c, d] + cAlp6 PR6[-i, -j, -g, -h, a, b, c, d])
    PPara[-c, m] PPara[-d, n] RLambda[-a, -b, -m, -n] -
   4 V[g] B[-k, -o] G3[o, -z] H[h, z]
    (cAlp1 PR1[-i, -j, -g, -h, a, b, c, d] + cAlp2 PR2[-i, -j, -g, -h, a, b, c, d] +
      cAlp3 PR3[-i, -j, -g, -h, a, b, c, d] + cAlp4 PR4[-i, -j, -g, -h, a, b, c, d] +
      cAlp5 PR5[-i, -j, -g, -h, a, b, c, d] + cAlp6 PR6[-i, -j, -g, -h, a, b, c, d])
    (PPerp[-c, m] PPara[-d, n] RLambda[-a, -b, -m, -n] +
      PPara[-c, m] PPerp[-d, n] RLambda[-a, -b, -m, -n]);
AChiPerpActivate = MakeRule[{AChiPerp[-i, -j, -k], Evaluate[AChiPerpDefinition]},
   MetricOn → All, ContractMetrics → True];
If[ChiPerpToggle,
  ChiPerpB0pDefinition = PB0p[e, f] PBPara[-e, -f, a, c] BChiPerp[-a, -c] /.
     BChiPerpActivate // ActivateGeneralO3Projections;
  ChiPerpB1pDefinition = PB1p[-n, -m, e, f] PBPara[-e, -f, a, c] BChiPerp[-a, -c] /.
```

```
    BChiPerpActivate // ActivateGeneralO3Projections;
ChiPerpB2pDefinition = PB2p[-n, -m, e, f] PBPara[-e, -f, a, c] BChiPerp[-a, -c] /.
    BChiPerpActivate // ActivateGeneralO3Projections;
ChiPerpB1mDefinition = PB1m[-n, f] PBPerp[-f, a, c] BChiPerp[-a, -c] /.
    BChiPerpActivate // ActivateGeneralO3Projections;

ChiPerpA0pDefinition = PA0p[e, f] PAPerp[-e, -f, a, b, c] AChiPerp[-a, -b, -c] /.
    AChiPerpActivate // ActivateGeneralO3Projections;
ChiPerpA1pDefinition = PA1p[-n, -m, e, f] PAPerp[-e, -f, a, b, c]
     AChiPerp[-a, -b, -c] /. AChiPerpActivate // ActivateGeneralO3Projections;
ChiPerpA2pDefinition = PA2p[-n, -m, e, f] PAPerp[-e, -f, a, b, c]
     AChiPerp[-a, -b, -c] /. AChiPerpActivate // ActivateGeneralO3Projections;
ChiPerpA0mDefinition = PA0m[d, e, f] PAPara[-d, -e, -f, a, b, c]
     AChiPerp[-a, -b, -c] /. AChiPerpActivate // ActivateGeneralO3Projections;
ChiPerpA1mDefinition = PA1m[-n, d, e, f] PAPara[-d, -e, -f, a, b, c]
     AChiPerp[-a, -b, -c] /. AChiPerpActivate // ActivateGeneralO3Projections;
ChiPerpA2mDefinition = PA2m[-n, -m, -o, d, e, f] PAPara[-d, -e, -f, a, b, c]
     AChiPerp[-a, -b, -c] /. AChiPerpActivate // ActivateGeneralO3Projections;

ChiPerpB0pActivate =
  MakeRule[{ChiPerpB0p[], Scalar[Evaluate[ChiPerpB0pDefinition]]},
   MetricOn → All, ContractMetrics → True];
ChiPerpB1pActivate = MakeRule[{ChiPerpB1p[-n, -m],
    Evaluate[ChiPerpB1pDefinition]}, MetricOn → All, ContractMetrics → True];
ChiPerpB1mActivate = MakeRule[{ChiPerpB1m[-n], Evaluate[ChiPerpB1mDefinition]},
   MetricOn → All, ContractMetrics → True];
ChiPerpB2pActivate = MakeRule[{ChiPerpB2p[-n, -m],
    Evaluate[ChiPerpB2pDefinition]}, MetricOn → All, ContractMetrics → True];
ChiPerpA0pActivate = MakeRule[{ChiPerpA0p[], Scalar[
     Evaluate[ChiPerpA0pDefinition]]}, MetricOn → All, ContractMetrics → True];
ChiPerpA0mActivate = MakeRule[{ChiPerpA0m[], Scalar[
     Evaluate[ChiPerpA0mDefinition]]}, MetricOn → All, ContractMetrics → True];
ChiPerpA1pActivate = MakeRule[{ChiPerpA1p[-n, -m],
    Evaluate[ChiPerpA1pDefinition]}, MetricOn → All, ContractMetrics → True];
ChiPerpA1mActivate = MakeRule[{ChiPerpA1m[-n], Evaluate[ChiPerpA1mDefinition]},
   MetricOn → All, ContractMetrics → True];
ChiPerpA2pActivate = MakeRule[{ChiPerpA2p[-n, -m],
    Evaluate[ChiPerpA2pDefinition]}, MetricOn → All, ContractMetrics → True];
ChiPerpA2mActivate = MakeRule[{ChiPerpA2m[-n, -m, -o],
    Evaluate[ChiPerpA2mDefinition]}, MetricOn → All, ContractMetrics → True];

ChiPerpActivate = Join[ChiPerpB0pActivate, ChiPerpB1pActivate, ChiPerpB1mActivate,
   ChiPerpB2pActivate, ChiPerpA0pActivate, ChiPerpA0mActivate, ChiPerpA1pActivate,
   ChiPerpA1mActivate, ChiPerpA2pActivate, ChiPerpA2mActivate];
```

```
    DumpSave[
     NotebookDirectory[] <> "mx_cache/chiperpactivate.mx", {ChiPerpActivate}];
    Print["done chiperpactivate"];
    Quit[];
  ];
MyImport["chiperpactivate.mx"];
```

Basic form $\chi^{\vDash}$ bJ^P, $\chi^{\vDash}$ AJ^P

```
In[•]:= DefTensor[ChiSingB0p[], M4, PrintAs → "χ⊧b0⁺"];
DeclareOrder[ChiSingB0p[], 1];
DefTensor[ChiSingB1p[-a, -b], M4, Antisymmetric[{-a, -b}], PrintAs → "χ⊧b1⁺"];
DeclareOrder[ChiSingB1p[-a, -b], 1];
DefTensor[ChiSingB1m[-a], M4, PrintAs → "χ⊧b1⁻"];
DeclareOrder[ChiSingB1m[-a], 1];
DefTensor[ChiSingB2p[-a, -b], M4, Symmetric[{-a, -b}], PrintAs → "χ⊧b2⁺"];
DeclareOrder[ChiSingB2p[-a, -b], 1];
DefTensor[ChiSingA0p[], M4, PrintAs → "χ⊧A0⁺"];
DeclareOrder[ChiSingA0p[], 1];
DefTensor[ChiSingA0m[], M4, PrintAs → "χ⊧A0⁻"];
DeclareOrder[ChiSingA0m[], 1];
DefTensor[ChiSingA1p[-a, -b], M4, Antisymmetric[{-a, -b}], PrintAs → "χ⊧A1⁺"];
DeclareOrder[ChiSingA1p[-a, -b], 1];
DefTensor[ChiSingA1m[-a], M4, PrintAs → "χ⊧A1⁻"];
DeclareOrder[ChiSingA1m[-a], 1];
DefTensor[ChiSingA2p[-a, -b], M4, Symmetric[{-a, -b}], PrintAs → "χ⊧A2⁺"];
DeclareOrder[ChiSingA2p[-a, -b], 1];
DefTensor[ChiSingA2m[-a, -b, -c], M4, Antisymmetric[{-a, -b}], PrintAs → "χ⊧A2⁻"];
DeclareOrder[ChiSingA2m[-a, -b, -c], 1];
AutomaticRules[ChiSingA2m,
  MakeRule[{ChiSingA2m[a, -b, -a], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[ChiSingA2m, MakeRule[{epsilonG[a, b, c, d] ChiSingA2m[-a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];

DefTensor[BChiSingExtra[-a, -c], M4];
DeclareOrder[BChiSingExtra[-a, -c], 1];
BChiSingExtraDefinition = 4 V[g] B[-k, -o] G3[o, -z] H[h, z]
    (cBet1 PT1[-i, -g, -h, a, c, d] + cBet2 PT2[-i, -g, -h, a, c, d] +
      cBet3 PT3[-i, -g, -h, a, c, d]) PPara[-c, m] PPara[-d, n] T[-a, -m, -n];
BChiSingExtraActivate = MakeRule[{BChiSingExtra[-i, -k],
     Evaluate[BChiSingExtraDefinition]}, MetricOn → All, ContractMetrics → True];

DefTensor[AChiSingExtra[-a, -b, -c], M4, Antisymmetric[{-a, -b}]];
```

```
DeclareOrder[AChiSingExtra[-a, -b, -c], 1];
AChiSingExtraDefinition = 8 V[g] B[-k, -o] G3[o, -z] H[h, z]
   (cAlp1 PR1[-i, -j, -g, -h, a, b, c, d] + cAlp2 PR2[-i, -j, -g, -h, a, b, c, d] +
     cAlp3 PR3[-i, -j, -g, -h, a, b, c, d] + cAlp4 PR4[-i, -j, -g, -h, a, b, c, d] +
     cAlp5 PR5[-i, -j, -g, -h, a, b, c, d] + cAlp6 PR6[-i, -j, -g, -h, a, b, c, d])
   PPara[-c, m] PPara[-d, n] R[-a, -b, -m, -n];
AChiSingExtraActivate = MakeRule[{AChiSingExtra[-i, -j, -k],
    Evaluate[AChiSingExtraDefinition]}, MetricOn → All, ContractMetrics → True];
If[ChiSingToggle,
  ChiSingExtraB1pDefinition =
   (BetPerpPerp1p / cBetPerpPerp1p) PB1p[-n, -m, e, f] PBPara[-e, -f, a, c]
       BChiSingExtra[-a, -c] /. ToBet /. TocBet /.
     BChiSingExtraActivate // ActivateGeneralO3Projections;
  ChiSingExtraB1mDefinition = (BetPerpPerp1m / cBetPerpPerp1m) PB1m[-n, f]
       PBPerp[-f, a, c] BChiSingExtra[-a, -c] /. ToBet /. TocBet /.
     BChiSingExtraActivate // ActivateGeneralO3Projections;

  ChiSingExtraA0pDefinition =
   (AlpPerpPerp0p / cAlpPerpPerp0p) PA0p[e, f] PAPerp[-e, -f, a, b, c]
       AChiSingExtra[-a, -b, -c] /. ToAlp /. TocAlp /.
     AChiSingExtraActivate // ActivateGeneralO3Projections;
  ChiSingExtraA1pDefinition = (AlpPerpPerp1p / cAlpPerpPerp1p) PA1p[-n, -m, e, f]
       PAPerp[-e, -f, a, b, c] AChiSingExtra[-a, -b, -c] /. ToAlp /. TocAlp /.
     AChiSingExtraActivate // ActivateGeneralO3Projections;
  ChiSingExtraA2pDefinition = (AlpPerpPerp2p / cAlpPerpPerp2p) PA2p[-n, -m, e, f]
       PAPerp[-e, -f, a, b, c] AChiSingExtra[-a, -b, -c] /. ToAlp /. TocAlp /.
     AChiSingExtraActivate // ActivateGeneralO3Projections;
  ChiSingExtraA0mDefinition = (AlpPerpPerp0m / cAlpPerpPerp0m) PA0m[d, e, f]
       PAPara[-d, -e, -f, a, b, c] AChiSingExtra[-a, -b, -c] /. ToAlp /. TocAlp /.
     AChiSingExtraActivate // ActivateGeneralO3Projections;
  ChiSingExtraA1mDefinition = (AlpPerpPerp1m / cAlpPerpPerp1m) PA1m[-n, d, e, f]
       PAPara[-d, -e, -f, a, b, c] AChiSingExtra[-a, -b, -c] /. ToAlp /. TocAlp /.
     AChiSingExtraActivate // ActivateGeneralO3Projections;
  ChiSingExtraA2mDefinition = (AlpPerpPerp2m / cAlpPerpPerp2m) PA2m[-n, -m, -o, d, e,
        f] PAPara[-d, -e, -f, a, b, c] AChiSingExtra[-a, -b, -c] /. ToAlp /. TocAlp /.
     AChiSingExtraActivate // ActivateGeneralO3Projections;

  ChiSingB1pDefinition =
   PhiB1p[-n, -m] + ChiSingExtraB1pDefinition /. PhiActivate // NoScalar //
    ToNewCanonical;
  ChiSingB1mDefinition = PhiB1m[-n] + ChiSingExtraB1mDefinition /. PhiActivate //
     NoScalar // ToNewCanonical;
```

```
  ChiSingA0pDefinition =
   PhiA0p[] + ChiSingExtraA0pDefinition /. PhiActivate // NoScalar //
    ToNewCanonical;
  ChiSingA0mDefinition = PhiA0m[] + ChiSingExtraA0mDefinition /. PhiActivate //
     NoScalar // ToNewCanonical;
  ChiSingA1pDefinition = PhiA1p[-n, -m] + ChiSingExtraA1pDefinition /. PhiActivate //
     NoScalar // ToNewCanonical;
  ChiSingA1mDefinition = PhiA1m[-n] + ChiSingExtraA1mDefinition /. PhiActivate //
     NoScalar // ToNewCanonical;
  ChiSingA2pDefinition = PhiA2p[-n, -m] + ChiSingExtraA2pDefinition /. PhiActivate //
     NoScalar // ToNewCanonical;
  ChiSingA2mDefinition = PhiA2m[-n, -m, -o] + ChiSingExtraA2mDefinition /.
      PhiActivate // NoScalar // ToNewCanonical;

  ChiSingB1pActivate =
   MakeRule[{ChiSingB1p[-n, -m], Evaluate[ChiSingB1pDefinition]},
    MetricOn → All, ContractMetrics → True];
  ChiSingB1mActivate = MakeRule[{ChiSingB1m[-n], Evaluate[ChiSingB1mDefinition]},
    MetricOn → All, ContractMetrics → True];
  ChiSingA0pActivate = MakeRule[{ChiSingA0p[], Scalar[
      Evaluate[ChiSingA0pDefinition]]}, MetricOn → All, ContractMetrics → True];
  ChiSingA0mActivate = MakeRule[{ChiSingA0m[], Scalar[
      Evaluate[ChiSingA0mDefinition]]}, MetricOn → All, ContractMetrics → True];
  ChiSingA1pActivate = MakeRule[{ChiSingA1p[-n, -m],
     Evaluate[ChiSingA1pDefinition]}, MetricOn → All, ContractMetrics → True];
  ChiSingA1mActivate = MakeRule[{ChiSingA1m[-n], Evaluate[ChiSingA1mDefinition]},
    MetricOn → All, ContractMetrics → True];
  ChiSingA2pActivate = MakeRule[{ChiSingA2p[-n, -m],
     Evaluate[ChiSingA2pDefinition]}, MetricOn → All, ContractMetrics → True];
  ChiSingA2mActivate = MakeRule[{ChiSingA2m[-n, -m, -o],
     Evaluate[ChiSingA2mDefinition]}, MetricOn → All, ContractMetrics → True];

  ChiSingActivate = Join[ChiSingB1pActivate, ChiSingB1mActivate,
    ChiSingA0pActivate, ChiSingA0mActivate, ChiSingA1pActivate,
    ChiSingA1mActivate, ChiSingA2pActivate, ChiSingA2mActivate];

  DumpSave[
   NotebookDirectory[] <> "mx_cache/chisingactivate.mx", {ChiSingActivate}];
  Print["done chisingactivate"];
  Quit[];
 ];
MyImport["chisingactivate.mx"];
```

Define ubJ^P, uAJ^P

In[∘]:=
```
DefTensor[UB0p[], M4, PrintAs → "ub0⁺"];
DeclareOrder[UB0p[], 1];
DefTensor[UB1p[-a, -b], M4, Antisymmetric[{-a, -b}],
  PrintAs → "ub1⁺", OrthogonalTo → {V[a], V[b]}];
DeclareOrder[UB1p[-a, -b], 1];
DefTensor[UB1m[-a], M4, PrintAs → "ub1⁻", OrthogonalTo → {V[a]}];
DeclareOrder[UB1m[-a], 1];
DefTensor[UB2p[-a, -b], M4, Symmetric[{-a, -b}],
  PrintAs → "ub2⁺", OrthogonalTo → {V[a], V[b]}];
DeclareOrder[UB2p[-a, -b], 1];
DefTensor[UA0p[], M4, PrintAs → "uA0⁺"];
DeclareOrder[UA0p[], 1];
DefTensor[UA0m[], M4, PrintAs → "uA0⁻"];
DeclareOrder[UA0m[], 1];
DefTensor[UA1p[-a, -b], M4, Antisymmetric[{-a, -b}],
  PrintAs → "uA1⁺", OrthogonalTo → {V[a], V[b]}];
DeclareOrder[UA1p[-a, -b], 1];
DefTensor[UA1m[-a], M4, PrintAs → "uA1⁻", OrthogonalTo → {V[a]}];
DeclareOrder[UA1m[-a], 1];
DefTensor[UA2p[-a, -b], M4, Symmetric[{-a, -b}],
  PrintAs → "uA2⁺", OrthogonalTo → {V[a], V[b]}];
DeclareOrder[UA2p[-a, -b], 1];
DefTensor[UA2m[-a, -b, -c], M4, Antisymmetric[{-a, -b}],
  PrintAs → "uA2⁻", OrthogonalTo → {V[a], V[b], V[c]}];
DeclareOrder[UA2m[-a, -b, -c], 1];
AutomaticRules[UA2m,
  MakeRule[{UA2m[a, -b, -a], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[UA2m, MakeRule[{epsilonG[a, b, c, d] UA2m[-a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[UB2p, MakeRule[{UB2p[a, -a], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[UA2p, MakeRule[{UA2p[a, -a], 0},
   MetricOn → All, ContractMetrics → True]];
```

Basic form ∏̂ bJ^P, ∏̂ AJ^P

In[∘]:=
```
DefTensor[PiPB0p[], M4, PrintAs → "∏̂b0⁺"];
DeclareOrder[PiPB0p[], 1];
DefTensor[PiPB1p[-a, -b], M4, Antisymmetric[{-a, -b}],
  PrintAs → "∏̂b1⁺", OrthogonalTo → {V[a], V[b]}];
```

```
DeclareOrder[PiPB1p[-a, -b], 1];
DefTensor[PiPB1m[-a], M4, PrintAs → "π̂b1⁻", OrthogonalTo → {V[a]}];
DeclareOrder[PiPB1m[-a], 1];
DefTensor[PiPB2p[-a, -b], M4, Symmetric[{-a, -b}],
  PrintAs → "π̂b2⁺", OrthogonalTo → {V[a], V[b]}];
DeclareOrder[PiPB2p[-a, -b], 1];
DefTensor[PiPA0p[], M4, PrintAs → "π̂A0⁺"];
DeclareOrder[PiPA0p[], 1, "IsUnityWithEHTerm" → True];
DefTensor[PiPA0m[], M4, PrintAs → "π̂A0⁻"];
DeclareOrder[PiPA0m[], 1];
DefTensor[PiPA1p[-a, -b], M4, Antisymmetric[{-a, -b}],
  PrintAs → "π̂A1⁺", OrthogonalTo → {V[a], V[b]}];
DeclareOrder[PiPA1p[-a, -b], 1];
DefTensor[PiPA1m[-a], M4, PrintAs → "π̂A1⁻", OrthogonalTo → {V[a]}];
DeclareOrder[PiPA1m[-a], 1];
DefTensor[PiPA2p[-a, -b], M4, Symmetric[{-a, -b}],
  PrintAs → "π̂A2⁺", OrthogonalTo → {V[a], V[b]}];
DeclareOrder[PiPA2p[-a, -b], 1];
DefTensor[PiPA2m[-a, -b, -c], M4, Antisymmetric[{-a, -b}],
  PrintAs → "π̂A2⁻", OrthogonalTo → {V[a], V[b], V[c]}];
DeclareOrder[PiPA2m[-a, -b, -c], 1];
AutomaticRules[PiPA2m,
  MakeRule[{PiPA2m[a, -b, -a], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[PiPA2m, MakeRule[{epsilonG[a, b, c, d] PiPA2m[-a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[PiPB2p, MakeRule[{PiPB2p[a, -a], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[PiPA2p, MakeRule[{PiPA2p[a, -a], 0},
   MetricOn → All, ContractMetrics → True]];

PiPB0pDefinition = PB0p[e, f] PBPara[-e, -f, a, c] BPiP[-a, -c];
PiPB1pDefinition = PB1p[-n, -m, e, f] PBPara[-e, -f, a, c] BPiP[-a, -c];
PiPB2pDefinition = PB2p[-n, -m, e, f] PBPara[-e, -f, a, c] BPiP[-a, -c];
PiPB1mDefinition = PB1m[-n, f] PBPerp[-f, a, c] BPiP[-a, -c];
PiPA0pDefinition = PA0p[e, f] PAPerp[-e, -f, a, b, c] APiP[-a, -b, -c];
PiPA1pDefinition = PA1p[-n, -m, e, f] PAPerp[-e, -f, a, b, c] APiP[-a, -b, -c];
PiPA2pDefinition = PA2p[-n, -m, e, f] PAPerp[-e, -f, a, b, c] APiP[-a, -b, -c];
PiPA0mDefinition = PA0m[d, e, f] PAPara[-d, -e, -f, a, b, c] APiP[-a, -b, -c];
PiPA1mDefinition = PA1m[-n, d, e, f] PAPara[-d, -e, -f, a, b, c] APiP[-a, -b, -c];
PiPA2mDefinition =
  PA2m[-n, -m, -o, d, e, f] PAPara[-d, -e, -f, a, b, c] APiP[-a, -b, -c];

PiPB0pActivate = MakeRule[{PiPB0p[], Scalar[Evaluate[PiPB0pDefinition]]},
```

```
    MetricOn → All, ContractMetrics → True];
PiPB1pActivate = MakeRule[{PiPB1p[-n, -m], Evaluate[PiPB1pDefinition]},
    MetricOn → All, ContractMetrics → True];
PiPB1mActivate = MakeRule[{PiPB1m[-n], Evaluate[PiPB1mDefinition]},
    MetricOn → All, ContractMetrics → True];
PiPB2pActivate = MakeRule[{PiPB2p[-n, -m], Evaluate[PiPB2pDefinition]},
    MetricOn → All, ContractMetrics → True];
PiPA0pActivate = MakeRule[{PiPA0p[], Scalar[Evaluate[PiPA0pDefinition]]},
    MetricOn → All, ContractMetrics → True];
PiPA0mActivate = MakeRule[{PiPA0m[], Scalar[Evaluate[PiPA0mDefinition]]},
    MetricOn → All, ContractMetrics → True];
PiPA1pActivate = MakeRule[{PiPA1p[-n, -m], Evaluate[PiPA1pDefinition]},
    MetricOn → All, ContractMetrics → True];
PiPA1mActivate = MakeRule[{PiPA1m[-n], Evaluate[PiPA1mDefinition]},
    MetricOn → All, ContractMetrics → True];
PiPA2pActivate = MakeRule[{PiPA2p[-n, -m], Evaluate[PiPA2pDefinition]},
    MetricOn → All, ContractMetrics → True];
PiPA2mActivate = MakeRule[{PiPA2m[-n, -m, -o], Evaluate[PiPA2mDefinition]},
    MetricOn → All, ContractMetrics → True];

PiPO3Activate = Join[PiPB0pActivate, PiPB1pActivate,
    PiPB1mActivate, PiPB2pActivate, PiPA0pActivate, PiPA0mActivate,
    PiPA1pActivate, PiPA1mActivate, PiPA2pActivate, PiPA2mActivate];
```

Basic form $\hat{T}\,J^P$, $\hat{R}\,J^P$

```
In[∘]:= (*O(3) decomposition of the canonical parts of field strengths*)
DefTensor[TP0m[], M4, PrintAs → "T̂0⁻"];
DeclareOrder[TP0m[], 1];
DefTensor[TP1p[-a, -b], M4, Antisymmetric[{-a, -b}],
   PrintAs → "T̂1⁺", OrthogonalTo → {V[a], V[b]}];
DeclareOrder[TP1p[-a, -b], 1];
DefTensor[TP1m[-a], M4, PrintAs → "T̂1⁻", OrthogonalTo → {V[a]}];
DeclareOrder[TP1m[-a], 1];
DefTensor[TP2m[-a, -b, -c], M4, Antisymmetric[{-a, -b}],
   PrintAs → "T̂2⁻", OrthogonalTo → {V[a], V[b], V[c]}];
DeclareOrder[TP2m[-a, -b, -c], 1];
DefTensor[RP0p[], M4, PrintAs → "R̂0⁺"];
DeclareOrder[RP0p[], 1];
DefTensor[RP0m[], M4, PrintAs → "R̂0⁻"];
DeclareOrder[RP0m[], 1];
DefTensor[RP1p[-a, -b], M4, Antisymmetric[{-a, -b}],
```

```
  PrintAs → "R̂1⁺", OrthogonalTo → {V[a], V[b]}];
DeclareOrder[RP1p[-a, -b], 1];
DefTensor[RP1m[-a], M4, PrintAs → "R̂1⁻", OrthogonalTo → {V[a]}];
DeclareOrder[RP1m[-a], 1];
DefTensor[RP2p[-a, -b], M4, Symmetric[{-a, -b}],
  PrintAs → "R̂2⁺", OrthogonalTo → {V[a], V[b]}];
DeclareOrder[RP2p[-a, -b], 1];
DefTensor[RP2m[-a, -b, -c], M4, Antisymmetric[{-a, -b}],
  PrintAs → "R̂2⁻", OrthogonalTo → {V[a], V[b], V[c]}];
DeclareOrder[RP2m[-a, -b, -c], 1];
AutomaticRules[TP2m,
  MakeRule[{TP2m[a, -b, -a], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[TP2m, MakeRule[{epsilonG[a, b, c, d] TP2m[-a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[TP2m, MakeRule[{Eps[a, b, c] TP2m[-a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[RP2m, MakeRule[{RP2m[a, -b, -a], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[RP2m, MakeRule[{epsilonG[a, b, c, d] RP2m[-a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[RP2p, MakeRule[{RP2p[a, -a], 0},
   MetricOn → All, ContractMetrics → True]];

(*Projections to break the field strengths up into canonical and non-
 canonical parts*)
DefTensor[PTPerp[-e, -f, a, b, c], M4];
DefTensor[PTPara[-e, -f, -g, a, b, c], M4];
DefTensor[PRPerp[-e, -f, -g, a, b, c, d], M4];
DefTensor[PRPara[-e, -f, -g, -h, a, b, c, d], M4];
PTPerpDefinition = V[a] PPara[-e, b] PPara[-f, c] /. PADMActivate // ToCanonical;
PTPerpActivate = MakeRule[{PTPerp[-e, -f, a, b, c], Evaluate[PTPerpDefinition]},
   MetricOn → All, ContractMetrics → True];
PTParaDefinition = PPara[-e, a] PPara[-f, b] PPara[-g, c] /. PADMActivate //
   ToCanonical;
PTParaActivate = MakeRule[{PTPara[-e, -f, -g, a, b, c], Evaluate[PTParaDefinition]},
   MetricOn → All, ContractMetrics → True];
PRPerpDefinition = V[a] PPara[-e, b] PPara[-f, c] PPara[-g, d] /. PADMActivate //
   ToCanonical;
PRPerpActivate = MakeRule[{PRPerp[-e, -f, -g, a, b, c, d],
    Evaluate[PRPerpDefinition]}, MetricOn → All, ContractMetrics → True];
PRParaDefinition = PPara[-e, a] PPara[-f, b] PPara[-g, c] PPara[-h, d] /.
    PADMActivate // ToCanonical;
PRParaActivate = MakeRule[{PRPara[-e, -f, -g, -h, a, b, c, d],
```

```
    Evaluate[PRParaDefinition]}, MetricOn → All, ContractMetrics → True];
PADMTActivate = Join[PTPerpActivate, PTParaActivate];
PADMRActivate = Join[PRPerpActivate, PRParaActivate];
```

Human-readable projections $\{{}^{A}\breve{\mathcal{P}}\}$, $\{{}^{E}\breve{\mathcal{P}}\}$

```
In[∘]:= (*Projection operators which define the O(3)
 decomposition of the canonical parts of field strengths*)
DefTensor[PT0m[d, e, f], M4, PrintAs -> "^{T0-}𝒫̆"];
DefTensor[PT1p[-a, -b, c, d], M4, PrintAs -> "^{T1+}𝒫̆"];
DefTensor[PT1m[-a, d, e, f], M4, PrintAs -> "^{T1-}𝒫̆"];
DefTensor[PT2m[-a, -b, -c, d, e, f], M4, PrintAs -> "^{T2-}𝒫̆"];
DefTensor[PR0p[e, f, g, h], M4, PrintAs -> "^{R0+}𝒫̆"];
DefTensor[PR0m[e, f, g], M4, PrintAs -> "^{R0-}𝒫̆"];
DefTensor[PR1p[-n, -m, e, f, g, h], M4, PrintAs -> "^{R1+}𝒫̆"];
DefTensor[PR1m[-n, e, f, g], M4, PrintAs -> "^{R1-}𝒫̆"];
DefTensor[PR2p[-n, -m, e, f, g, h], M4, PrintAs -> "^{R2+}𝒫̆"];
DefTensor[PR2m[-n, -m, -o, e, f, g], M4, PrintAs -> "^{R2-}𝒫̆"];
PT0mDefinition =
  PPara[-i, d] PPara[-j, e] PPara[-k, f] epsilonG[i, j, k, g] V[-g] /. PADMActivate //
   ToCanonical;
PT1pDefinition = PPara[-a, i] PPara[-b, j] PPara[c, -k] PPara[d, -l]
     Antisymmetrize[G[-i, k] G[-j, l], {-i, -j}] /. PADMActivate // ToCanonical;
PT1mDefinition = PPara[-i, d] PPara[-j, f] PPara[k, -a] PPara[-l, e] G[i, j] G[-k, l] /.
    PADMActivate // ToCanonical;
PT2mDefinition = PPara[-a, i] PPara[-b, j] PPara[-c, k] PPara[e, -l]
     PPara[f, -n] PPara[d, -m] (3/4) ((1/3) (2 G[-i, l] G[-j, n] G[-k, m] -
          G[-j, l] G[-k, n] G[-i, m] - G[-k, l] G[-i, n] G[-j, m]) - Antisymmetrize[
        G[-i, -k] G[-j, n] G[l, m], {-i, -j}]) /. PADMActivate // ToCanonical;
PR0pDefinition = PPara[-e, -g] PPara[-f, -h] /. PADMActivate // ToCanonical;
PR0mDefinition =
  PPara[-i, -e] PPara[-j, -f] PPara[-k, -g] epsilonG[i, j, k, p] V[-p] /.
    PADMActivate // ToCanonical;
PR1pDefinition = PPara[-e, -g] Antisymmetrize[PPara[-n, -f] PPara[-m, -h],
      {-n, -m}] /. PADMActivate // ToCanonical;
PR1mDefinition = PPara[-e, -g] PPara[-n, -f] /. PADMActivate // ToCanonical;
PR2pDefinition =
  PPara[-e, -g] (Symmetrize[PPara[-n, -f] PPara[-m, -h], {-n, -m}] - (1/3)
        PPara[-n, -m] PPara[-f, -h]) /. PADMActivate // ToCanonical;
PR2mDefinition = PPara[-a, i] PPara[-b, j] PPara[-c, k] PPara[e, -l]
     PPara[f, -n] PPara[d, -m] (3/4) ((1/3) (2 G[-i, l] G[-j, n] G[-k, m] -
          G[-j, l] G[-k, n] G[-i, m] - G[-k, l] G[-i, n] G[-j, m]) - Antisymmetrize[
```

```
      G[-i, -k] G[-j, n] G[l, m], {-i, -j}]) /. PADMActivate // ToCanonical;
PT0mActivate = MakeRule[{PT0m[d, e, f], Evaluate[PT0mDefinition]},
   MetricOn → All, ContractMetrics → True];
PT1pActivate = MakeRule[{PT1p[-a, -b, c, d], Evaluate[PT1pDefinition]},
   MetricOn → All, ContractMetrics → True];
PT1mActivate = MakeRule[{PT1m[-a, d, e, f], Evaluate[PT1mDefinition]},
   MetricOn → All, ContractMetrics → True];
PT2mActivate = MakeRule[{PT2m[-a, -b, -c, d, e, f], Evaluate[PT2mDefinition]},
   MetricOn → All, ContractMetrics → True];
PR0pActivate = MakeRule[{PR0p[-e, -f, -g, -h], Evaluate[PR0pDefinition]},
   MetricOn → All, ContractMetrics → True];
PR0mActivate = MakeRule[{PR0m[-e, -f, -g], Evaluate[PR0mDefinition]},
   MetricOn → All, ContractMetrics → True];
PR1pActivate = MakeRule[{PR1p[-n, -m, -e, -f, -g, -h], Evaluate[PR1pDefinition]},
   MetricOn → All, ContractMetrics → True];
PR1mActivate = MakeRule[{PR1m[-n, -e, -f, -g], Evaluate[PR1mDefinition]},
   MetricOn → All, ContractMetrics → True];
PR2pActivate = MakeRule[{PR2p[-n, -m, -e, -f, -g, -h], Evaluate[PR2pDefinition]},
   MetricOn → All, ContractMetrics → True];
PR2mActivate = MakeRule[{PR2m[-a, -b, -c, d, e, f], Evaluate[PR2mDefinition]},
   MetricOn → All, ContractMetrics → True];

(*These rules then expand those canonical
 field strength O(3) projection operators*)
PO3TActivate = Join[PT0mActivate, PT1pActivate, PT1mActivate, PT2mActivate];
PO3RActivate = Join[PR0pActivate, PR0mActivate,
   PR1pActivate, PR1mActivate, PR2pActivate, PR2mActivate];
```

Projection normalisations $\{c^{\parallel}_{A}\}$. $\{c^{\parallel}_{E}\}$

In[∘]:=

```
DefConstantSymbol[cParaA0p, PrintAs → "$c^{\parallel}_{A0^+}$"];
DefConstantSymbol[cParaA0m, PrintAs → "$c^{\parallel}_{A0^-}$"];
DefConstantSymbol[cParaA1p, PrintAs → "$c^{\parallel}_{A1^+}$"];
DefConstantSymbol[cParaA1m, PrintAs → "$c^{\parallel}_{A1^-}$"];
DefConstantSymbol[cParaA2p, PrintAs → "$c^{\parallel}_{A2^+}$"];
DefConstantSymbol[cParaA2m, PrintAs → "$c^{\parallel}_{A2^-}$"];

DefConstantSymbol[cParaB0p, PrintAs → "$c^{\parallel}_{b0^+}$"];
DefConstantSymbol[cParaB0m, PrintAs → "$c^{\parallel}_{b0^-}$"];
DefConstantSymbol[cParaB1p, PrintAs → "$c^{\parallel}_{b1^+}$"];
DefConstantSymbol[cParaB1m, PrintAs → "$c^{\parallel}_{b1^-}$"];
DefConstantSymbol[cParaB2p, PrintAs → "$c^{\parallel}_{b2^+}$"];
DefConstantSymbol[cParaB2m, PrintAs → "$c^{\parallel}_{b2^-}$"];
```

Transfer couplings $\{\hat{\alpha}_A^{\perp\parallel}\}$, $\{\hat{\beta}_E^{\perp\parallel}\}$

```
In[•]:= DefConstantSymbol[AlpPerpPara0p, PrintAs → "α̂0+⊥∥"];
DefConstantSymbol[AlpPerpPara0m, PrintAs → "α̂0-⊥∥"];
DefConstantSymbol[AlpPerpPara1p, PrintAs → "α̂1+⊥∥"];
DefConstantSymbol[AlpPerpPara1m, PrintAs → "α̂1-⊥∥"];
DefConstantSymbol[AlpPerpPara2p, PrintAs → "α̂2+⊥∥"];
DefConstantSymbol[AlpPerpPara2m, PrintAs → "α̂2-⊥∥"];

AlpPerpPara = {AlpPerpPara0p, AlpPerpPara0m,
   AlpPerpPara1p, AlpPerpPara1m, AlpPerpPara2p, AlpPerpPara2m};

DefConstantSymbol[BetPerpPara0p, PrintAs → "β̂0+⊥∥"];
DefConstantSymbol[BetPerpPara0m, PrintAs → "β̂0-⊥∥"];
DefConstantSymbol[BetPerpPara1p, PrintAs → "β̂1+⊥∥"];
DefConstantSymbol[BetPerpPara1m, PrintAs → "β̂1-⊥∥"];
DefConstantSymbol[BetPerpPara2p, PrintAs → "β̂2+⊥∥"];
DefConstantSymbol[BetPerpPara2m, PrintAs → "β̂2-⊥∥"];

BetPerpPara = {BetPerpPara0p, BetPerpPara0m,
   BetPerpPara1p, BetPerpPara1m, BetPerpPara2p, BetPerpPara2m};

If[TransferCouplingsPerpParaToggle,
  TransferCouplingsPerpParaSolutions = {};
  tmp =
   BetPerpPara0m PT0m[e, f, g] PTPara[-e, -f, -g, a, v, w] - PB0p[x, z] PBPara[-x, -z,
              i, f] V[g] PPara[-f, h] PPara[v, -c] PPara[w, -d]
             (Bet1 PT1[-i, -g, -h, a, c, d] + Bet2 PT2[-i, -g, -h, a, c, d] +
               Bet3 PT3[-i, -g, -h, a, c, d]) /. PO3TActivate /.
          PADMTActivate /. PO3PiActivate /. PActivate /. PADMPiActivate /.
     PADMActivate // ToCanonical // CollectTensors;
  TransferCouplingsPerpParaSolutions = Join[TransferCouplingsPerpParaSolutions,
    Solve[ToConstantSymbolEquations[tmp == 0], BetPerpPara0p][[1]]];
  tmp = BetPerpPara1p PT1p[-n, -m, e, f] PTPerp[-e, -f, a, v, w] -
             PB1p[-q, -r, x, z] PBPara[-x, -z, i, f] V[g] PPara[-f, h] PPara[v, -c]
              PPara[w, -d] (Bet1 PT1[-i, -g, -h, a, c, d] + Bet2 PT2[-i, -g, -h,
                  a, c, d] + Bet3 PT3[-i, -g, -h, a, c, d]) /. PO3TActivate /.
          PADMTActivate /. PO3PiActivate /. PActivate /. PADMPiActivate /.
     PADMActivate // ToCanonical // CollectTensors;
  TransferCouplingsPerpParaSolutions = Join[TransferCouplingsPerpParaSolutions,
    Solve[ToConstantSymbolEquations[tmp == 0], BetPerpPara1p][[1]]];
  tmp = BetPerpPara1m PT1m[-n, e, f, g] PTPara[-e, -f, -g, a, v, w] -
```

```
            PB1m[-q, z] PBPerp[-z, i, f] V[g] PPara[-f, h] PPara[v, -c] PPara[w, -d]
             (Bet1 PT1[-i, -g, -h, a, c, d] + Bet2 PT2[-i, -g, -h, a, c, d] +
               Bet3 PT3[-i, -g, -h, a, c, d]) /. PO3TActivate /.
          PADMTActivate /. PO3PiActivate /. PActivate /. PADMPiActivate /.
      PADMActivate // ToCanonical // CollectTensors;
TransferCouplingsPerpParaSolutions = Join[TransferCouplingsPerpParaSolutions,
   Solve[ToConstantSymbolEquations[tmp == 0], BetPerpPara1m][[1]]];
tmp = BetPerpPara2p PT2m[-n, -m, -o, e, f, g] PTPara[-e, -f, -g, a, v, w] -
           PB2p[-q, -r, x, z] PBPara[-x, -z, i, f] V[g] PPara[-f, h] PPara[v, -c]
            PPara[w, -d] (Bet1 PT1[-i, -g, -h, a, c, d] + Bet2 PT2[-i, -g, -h,
                a, c, d] + Bet3 PT3[-i, -g, -h, a, c, d]) /. PO3TActivate /.
         PADMTActivate.PO3PiActivate /. PActivate /. PADMPiActivate /.
      PADMActivate // ToCanonical // CollectTensors;
TransferCouplingsPerpParaSolutions = Join[TransferCouplingsPerpParaSolutions,
   Solve[ToConstantSymbolEquations[tmp == 0], BetPerpPara2p][[1]]];
tmp = AlpPerpPara0p PR0p[e, f, g, h] Antisymmetrize[PRPara[-e, -f, -g, -h, a, b, v,
               w], {a, b}] - PA0p[x, z] PAPerp[-x, -z, i, j, f] V[g] PPara[-f, h]
             PPara[v, -c] PPara[w, -d] (Alp1 PR1[-i, -j, -g, -h, a, b, c, d] + Alp2
                PR2[-i, -j, -g, -h, a, b, c, d] + Alp3 PR3[-i, -j, -g, -h, a, b, c, d] +
               Alp4 PR4[-i, -j, -g, -h, a, b, c, d] + Alp5 PR5[-i, -j, -g,
                 -h, a, b, c, d] + Alp6 PR6[-i, -j, -g, -h, a, b, c, d]) /.
           PO3TActivate /. PADMTActivate /. PO3PiActivate /. PActivate /.
       PADMPiActivate /. PADMActivate // ToCanonical // CollectTensors;
TransferCouplingsPerpParaSolutions = Join[TransferCouplingsPerpParaSolutions,
   Solve[ToConstantSymbolEquations[tmp == 0], AlpPerpPara0p][[1]]];
tmp = AlpPerpPara0m PR0m[e, f, g] Antisymmetrize[PRPerp[-e, -f, -g, a, b, v, w],
              {a, b}] - PA0m[x, y, z] PAPara[-x, -y, -z, i, j, f] V[g] PPara[-f, h]
             PPara[v, -c] PPara[w, -d] (Alp1 PR1[-i, -j, -g, -h, a, b, c, d] + Alp2
                PR2[-i, -j, -g, -h, a, b, c, d] + Alp3 PR3[-i, -j, -g, -h, a, b, c, d] +
               Alp4 PR4[-i, -j, -g, -h, a, b, c, d] + Alp5 PR5[-i, -j, -g, -h, a,
                 b, c, d] + Alp6 PR6[-i, -j, -g, -h, a, b, c, d]) /. PO3TActivate /.
          PADMTActivate /. PO3PiActivate /. PActivate /. PADMPiActivate /.
      PADMActivate // ToCanonical // CollectTensors;
TransferCouplingsPerpParaSolutions = Join[TransferCouplingsPerpParaSolutions,
   Solve[ToConstantSymbolEquations[tmp == 0], AlpPerpPara0m][[1]]];
tmp = AlpPerpPara1p PR1p[-n, -m, e, f, g, h]
             Antisymmetrize[PRPara[-e, -f, -g, -h, a, b, v, w], {a, b}] -
            PA1p[-p, -q, x, z] PAPerp[-x, -z, i, j, f] V[g] PPara[-f, h]
             PPara[v, -c] PPara[w, -d] (Alp1 PR1[-i, -j, -g, -h, a, b, c, d] + Alp2
                PR2[-i, -j, -g, -h, a, b, c, d] + Alp3 PR3[-i, -j, -g, -h, a, b, c, d] +
               Alp4 PR4[-i, -j, -g, -h, a, b, c, d] + Alp5 PR5[-i, -j, -g, -h, a,
                 b, c, d] + Alp6 PR6[-i, -j, -g, -h, a, b, c, d]) /. PO3TActivate /.
          PADMTActivate /. PO3PiActivate /. PActivate /. PADMPiActivate /.
```

```
      PADMActivate // ToCanonical // CollectTensors;
  TransferCouplingsPerpParaSolutions = Join[TransferCouplingsPerpParaSolutions,
    Solve[ToConstantSymbolEquations[tmp == 0], AlpPerpPara1p][[1]]];
  tmp = AlpPerpPara1m PR1m[-n, e, f, g] Antisymmetrize[PRPerp[-e, -f, -g, a, b, v, w],
              {a, b}] - PA1m[-p, x, y, z] PAPara[-x, -y, -z, i, j, f] V[g] PPara[-f,
              h] PPara[v, -c] PPara[w, -d] (Alp1 PR1[-i, -j, -g, -h, a, b, c, d] + Alp2
                PR2[-i, -j, -g, -h, a, b, c, d] + Alp3 PR3[-i, -j, -g, -h, a, b, c, d] +
               Alp4 PR4[-i, -j, -g, -h, a, b, c, d] + Alp5 PR5[-i, -j, -g, -h, a,
                 b, c, d] + Alp6 PR6[-i, -j, -g, -h, a, b, c, d]) /. PO3TActivate /.
          PADMTActivate /. PO3PiActivate /. PActivate /. PADMPiActivate /.
      PADMActivate // ToCanonical // CollectTensors;
  TransferCouplingsPerpParaSolutions = Join[TransferCouplingsPerpParaSolutions,
    Solve[ToConstantSymbolEquations[tmp == 0], AlpPerpPara1m][[1]]];
  tmp = AlpPerpPara2p PR2p[-n, -m, e, f, g, h]
             Antisymmetrize[PRPara[-e, -f, -g, -h, a, b, v, w], {a, b}] -
            PA2p[-p, -q, x, z] PAPerp[-x, -z, i, j, f] V[g] PPara[-f, h]
             PPara[v, -c] PPara[w, -d] (Alp1 PR1[-i, -j, -g, -h, a, b, c, d] + Alp2
                PR2[-i, -j, -g, -h, a, b, c, d] + Alp3 PR3[-i, -j, -g, -h, a, b, c, d] +
               Alp4 PR4[-i, -j, -g, -h, a, b, c, d] + Alp5 PR5[-i, -j, -g, -h, a,
                 b, c, d] + Alp6 PR6[-i, -j, -g, -h, a, b, c, d]) /. PO3TActivate /.
          PADMTActivate /. PO3PiActivate /. PActivate /. PADMPiActivate /.
      PADMActivate // ToCanonical // CollectTensors;
  TransferCouplingsPerpParaSolutions = Join[TransferCouplingsPerpParaSolutions,
    Solve[ToConstantSymbolEquations[tmp == 0], AlpPerpPara2p][[1]]];
  tmp = AlpPerpPara2m PR2m[-n, -m, -o, e, f, g]
             Antisymmetrize[PRPerp[-e, -f, -g, a, b, v, w], {a, b}]
            - PA2m[-q, -p, -r, x, y, z] PAPara[-x, -y, -z, i, j, f] V[g] PPara[-f, h]
             PPara[v, -c] PPara[w, -d] (Alp1 PR1[-i, -j, -g, -h, a, b, c, d] + Alp2
                PR2[-i, -j, -g, -h, a, b, c, d] + Alp3 PR3[-i, -j, -g, -h, a, b, c, d] +
               Alp4 PR4[-i, -j, -g, -h, a, b, c, d] + Alp5 PR5[-i, -j, -g, -h, a,
                 b, c, d] + Alp6 PR6[-i, -j, -g, -h, a, b, c, d]) /. PO3TActivate /.
          PADMTActivate /. PO3PiActivate /. PActivate /. PADMPiActivate /.
      PADMActivate // ToCanonical // CollectTensors;
  TransferCouplingsPerpParaSolutions = Join[TransferCouplingsPerpParaSolutions,
    Solve[ToConstantSymbolEquations[tmp == 0], AlpPerpPara2m][[1]]];

  DumpSave[NotebookDirectory[] <> "mx_cache/TransferCouplingsPerpPara.mx",
   {TransferCouplingsPerpParaSolutions}];
  Quit[];
 ];

(*MyImport["TransferCouplingsPerpPara.mx"];*)
```

Transfer couplings $\{\hat{\alpha}_{A}^{\parallel\perp}\}$, $\{\hat{\beta}_{E}^{\parallel\perp}\}$

In[ ]:=
```
DefConstantSymbol[AlpParaPerp0p, PrintAs → "α̂^∥⊥_0+"];
DefConstantSymbol[AlpParaPerp0m, PrintAs → "α̂^∥⊥_0-"];
DefConstantSymbol[AlpParaPerp1p, PrintAs → "α̂^∥⊥_1+"];
DefConstantSymbol[AlpParaPerp1m, PrintAs → "α̂^∥⊥_1-"];
DefConstantSymbol[AlpParaPerp2p, PrintAs → "α̂^∥⊥_2+"];
DefConstantSymbol[AlpParaPerp2m, PrintAs → "α̂^∥⊥_2-"];

AlpParaPerp = {AlpParaPerp0p, AlpParaPerp0m,
   AlpParaPerp1p, AlpParaPerp1m, AlpParaPerp2p, AlpParaPerp2m};

DefConstantSymbol[BetParaPerp0p, PrintAs → "β̂^∥⊥_0+"];
DefConstantSymbol[BetParaPerp0m, PrintAs → "β̂^∥⊥_0-"];
DefConstantSymbol[BetParaPerp1p, PrintAs → "β̂^∥⊥_1+"];
DefConstantSymbol[BetParaPerp1m, PrintAs → "β̂^∥⊥_1-"];
DefConstantSymbol[BetParaPerp2p, PrintAs → "β̂^∥⊥_2+"];
DefConstantSymbol[BetParaPerp2m, PrintAs → "β̂^∥⊥_2-"];

BetParaPerp = {BetParaPerp0p, BetParaPerp0m,
   BetParaPerp1p, BetParaPerp1m, BetParaPerp2p, BetParaPerp2m};
```

Transfer couplings $\{\hat{\alpha}^{""}_{A}\}$, $\{\hat{\beta}^{""}_{E}\}$

```
In[•]:= DefConstantSymbol[AlpParaPara0p, PrintAs -> "α̂^{""}_{0+}"];
DefConstantSymbol[AlpParaPara0m, PrintAs -> "α̂^{""}_{0-}"];
DefConstantSymbol[AlpParaPara1p, PrintAs -> "α̂^{""}_{1+}"];
DefConstantSymbol[AlpParaPara1m, PrintAs -> "α̂^{""}_{1-}"];
DefConstantSymbol[AlpParaPara2p, PrintAs -> "α̂^{""}_{2+}"];
DefConstantSymbol[AlpParaPara2m, PrintAs -> "α̂^{""}_{2-}"];

AlpParaPara = {AlpParaPara0p, AlpParaPara0m,
   AlpParaPara1p, AlpParaPara1m, AlpParaPara2p, AlpParaPara2m};

DefConstantSymbol[BetParaPara0p, PrintAs -> "β̂^{""}_{0+}"];
DefConstantSymbol[BetParaPara0m, PrintAs -> "β̂^{""}_{0-}"];
DefConstantSymbol[BetParaPara1p, PrintAs -> "β̂^{""}_{1+}"];
DefConstantSymbol[BetParaPara1m, PrintAs -> "β̂^{""}_{1-}"];
DefConstantSymbol[BetParaPara2p, PrintAs -> "β̂^{""}_{2+}"];
DefConstantSymbol[BetParaPara2m, PrintAs -> "β̂^{""}_{2-}"];

BetParaPara = {BetParaPara0p, BetParaPara0m,
   BetParaPara1p, BetParaPara1m, BetParaPara2p, BetParaPara2m};
```

Transfer solutions calculated by hand

```
In[•]:= AlpDetRelations = {AlpParaPara0p == (Alp4 + Alp6) / 2,
    AlpParaPara0m == (Alp2 + Alp3) / 2,
    AlpParaPara1p == - (Alp2 + Alp5) / 2,
    AlpParaPara1m == (Alp4 + Alp5) / 2,
    AlpParaPara2p == (Alp1 + Alp4) / 2,
    AlpParaPara2m == - (Alp1 + Alp2) / 2,
    AlpPerpPara0p == - (Alp4 - Alp6) / 4,
    AlpPerpPara0m == (Alp2 - Alp3) / 2,
    AlpPerpPara1p == - (Alp2 - Alp5) / 2,
    AlpPerpPara1m == (Alp4 - Alp5) / 2,
    AlpPerpPara2p == (Alp1 - Alp4) / 2,
    AlpPerpPara2m == - (Alp1 - Alp2) / 2,
    AlpParaPerp0p == - (Alp4 - Alp6) / 2,
    AlpParaPerp0m == (Alp2 - Alp3) / 4,
    AlpParaPerp1p == (Alp2 - Alp5) / 4,
    AlpParaPerp1m == (Alp4 - Alp5) / 4,
    AlpParaPerp2p == (Alp1 - Alp4) / 4,
    AlpParaPerp2m == - (Alp1 - Alp2) / 4,
```

```
    AlpPerpPerp0p == (Alp4 + Alp6) / 4,
    AlpPerpPerp0m == (Alp2 + Alp3) / 4,
    AlpPerpPerp1p == (Alp2 + Alp5) / 4,
    AlpPerpPerp1m == (Alp4 + Alp5) / 4,
    AlpPerpPerp2p == (Alp1 + Alp4) / 4,
    AlpPerpPerp2m == - (Alp1 + Alp2) / 4};

BetDetRelations = {BetParaPara0p == 0,
    BetParaPara0m == Bet3 / 6,
    BetParaPara1p == (2 Bet1 + Bet3) / 3,
    BetParaPara1m == (Bet1 + 2 Bet2) / 3,
    BetParaPara2p == 0,
    BetParaPara2m == Bet1,
    BetPerpPara0p == 0,
    BetPerpPara0m == 0,
    BetPerpPara1p == - (Bet1 - Bet3) / 3,
    BetPerpPara1m == - (Bet1 - Bet2) / 3,
    BetPerpPara2p == 0,
    BetPerpPara2m == 0,
    BetParaPerp0p == 0,
    BetParaPerp0m == 0,
    BetParaPerp1p == - (Bet1 - Bet3) / 3,
    BetParaPerp1m == - (Bet1 - Bet2) / 3,
    BetParaPerp2p == 0,
    BetParaPerp2m == 0,
    BetPerpPerp0p == Bet2 / 2,
    BetPerpPerp0m == 0,
    BetPerpPerp1p == (Bet1 + 2 Bet3) / 6,
    BetPerpPerp1m == (2 Bet1 + Bet2) / 6,
    BetPerpPerp2p == Bet1 / 2,
    BetPerpPerp2m == 0};

AlpDeterminants = {AlpParaPara0p AlpPerpPerp0p - AlpParaPerp0p AlpPerpPara0p,
    AlpParaPara0m AlpPerpPerp0m - AlpParaPerp0m AlpPerpPara0m,
    AlpParaPara1p AlpPerpPerp1p - AlpParaPerp1p AlpPerpPara1p,
    AlpParaPara1m AlpPerpPerp1m - AlpParaPerp1m AlpPerpPara1m,
    AlpParaPara2p AlpPerpPerp2p - AlpParaPerp2p AlpPerpPara2p,
    AlpParaPara2m AlpPerpPerp2m - AlpParaPerp2m AlpPerpPara2m};

BetDeterminants = {BetParaPara0p BetPerpPerp0p - BetParaPerp0p BetPerpPara0p,
    BetParaPara0m BetPerpPerp0m - BetParaPerp0m BetPerpPara0m,
    BetParaPara1p BetPerpPerp1p - BetParaPerp1p BetPerpPara1p,
    BetParaPara1m BetPerpPerp1m - BetParaPerp1m BetPerpPara1m,
```

```
    BetParaPara2p BetPerpPerp2p - BetParaPerp2p BetPerpPara2p,
    BetParaPara2m BetPerpPerp2m - BetParaPerp2m BetPerpPara2m};

ToAlp = SolveConstants[AlpDetRelations,
     Join[AlpPerpPara, AlpPerpPerp, AlpParaPara, AlpParaPerp]][[1]];

ToBet = SolveConstants[BetDetRelations,
     Join[BetPerpPara, BetPerpPerp, BetParaPara, BetParaPerp]][[1]];

cAlpDetRelations = {cAlpParaPara0p == (cAlp4 + cAlp6) / 2,
    cAlpParaPara0m == (cAlp2 + cAlp3) / 2,
    cAlpParaPara1p == - (cAlp2 + cAlp5) / 2,
    cAlpParaPara1m == (cAlp4 + cAlp5) / 2,
    cAlpParaPara2p == (cAlp1 + cAlp4) / 2,
    cAlpParaPara2m == - (cAlp1 + cAlp2) / 2,
    cAlpPerpPara0p == - (cAlp4 - cAlp6) / 4,
    cAlpPerpPara0m == (cAlp2 - cAlp3) / 2,
    cAlpPerpPara1p == - (cAlp2 - cAlp5) / 2,
    cAlpPerpPara1m == (cAlp4 - cAlp5) / 2,
    cAlpPerpPara2p == (cAlp1 - cAlp4) / 2,
    cAlpPerpPara2m == - (cAlp1 - cAlp2) / 2,
    cAlpParaPerp0p == - (cAlp4 - cAlp6) / 2,
    cAlpParaPerp0m == (cAlp2 - cAlp3) / 4,
    cAlpParaPerp1p == (cAlp2 - cAlp5) / 4,
    cAlpParaPerp1m == (cAlp4 - cAlp5) / 4,
    cAlpParaPerp2p == (cAlp1 - cAlp4) / 4,
    cAlpParaPerp2m == - (cAlp1 - cAlp2) / 4,
    cAlpPerpPerp0p == (cAlp4 + cAlp6) / 4,
    cAlpPerpPerp0m == (cAlp2 + cAlp3) / 4,
    cAlpPerpPerp1p == (cAlp2 + cAlp5) / 4,
    cAlpPerpPerp1m == (cAlp4 + cAlp5) / 4,
    cAlpPerpPerp2p == (cAlp1 + cAlp4) / 4,
    cAlpPerpPerp2m == - (cAlp1 + cAlp2) / 4};

cBetDetRelations = {cBetParaPara0p == 0,
    cBetParaPara0m == cBet3 / 6,
    cBetParaPara1p == (2 cBet1 + cBet3) / 3,
    cBetParaPara1m == (cBet1 + 2 cBet2) / 3,
    cBetParaPara2p == 0,
    cBetParaPara2m == cBet1,
    cBetPerpPara0p == 0,
    cBetPerpPara0m == 0,
```

```
    cBetPerpPara1p == - (cBet1 - cBet3) / 3,
    cBetPerpPara1m == - (cBet1 - cBet2) / 3,
    cBetPerpPara2p == 0,
    cBetPerpPara2m == 0,
    cBetParaPerp0p == 0,
    cBetParaPerp0m == 0,
    cBetParaPerp1p == - (cBet1 - cBet3) / 3,
    cBetParaPerp1m == - (cBet1 - cBet2) / 3,
    cBetParaPerp2p == 0,
    cBetParaPerp2m == 0,
    cBetPerpPerp0p == cBet2 / 2,
    cBetPerpPerp0m == 0,
    cBetPerpPerp1p == (cBet1 + 2 cBet3) / 6,
    cBetPerpPerp1m == (2 cBet1 + cBet2) / 6,
    cBetPerpPerp2p == cBet1 / 2,
    cBetPerpPerp2m == 0};

cAlpDeterminants = {cAlpParaPara0p cAlpPerpPerp0p - cAlpParaPerp0p cAlpPerpPara0p,
    cAlpParaPara0m cAlpPerpPerp0m - cAlpParaPerp0m cAlpPerpPara0m,
    cAlpParaPara1p cAlpPerpPerp1p - cAlpParaPerp1p cAlpPerpPara1p,
    cAlpParaPara1m cAlpPerpPerp1m - cAlpParaPerp1m cAlpPerpPara1m,
    cAlpParaPara2p cAlpPerpPerp2p - cAlpParaPerp2p cAlpPerpPara2p,
    cAlpParaPara2m cAlpPerpPerp2m - cAlpParaPerp2m cAlpPerpPara2m};

cBetDeterminants = {cBetParaPara0p cBetPerpPerp0p - cBetParaPerp0p cBetPerpPara0p,
    cBetParaPara0m cBetPerpPerp0m - cBetParaPerp0m cBetPerpPara0m,
    cBetParaPara1p cBetPerpPerp1p - cBetParaPerp1p cBetPerpPara1p,
    cBetParaPara1m cBetPerpPerp1m - cBetParaPerp1m cBetPerpPara1m,
    cBetParaPara2p cBetPerpPerp2p - cBetParaPerp2p cBetPerpPara2p,
    cBetParaPara2m cBetPerpPerp2m - cBetParaPerp2m cBetPerpPara2m};

TocAlp = SolveConstants[cAlpDetRelations,
     Join[cAlpPerpPara, cAlpPerpPerp, cAlpParaPara, cAlpParaPerp]][[1]];

TocBet = SolveConstants[cBetDetRelations,
     Join[cBetPerpPara, cBetPerpPerp, cBetParaPara, cBetParaPerp]][[1]];
```

Define $\hat{\lambda}\, J^P$

```
In[•]:= (*O(3) decomposition of the canonical parts of Riemann-Cartan multiplier*)
DefTensor[TLambdaP0m[], M4, PrintAs → "T̂λ0⁻"];
DeclareOrder[TLambdaP0m[], 1];
DefTensor[TLambdaP1p[-a, -b], M4,
  Antisymmetric[{-a, -b}], PrintAs → "T̂λ1⁺", OrthogonalTo → {V[a], V[b]}];
DeclareOrder[TLambdaP1p[-a, -b], 1];
DefTensor[TLambdaP1m[-a], M4, PrintAs → "T̂λ1⁻", OrthogonalTo → {V[a]}];
DeclareOrder[TLambdaP1m[-a], 1];
DefTensor[TLambdaP2m[-a, -b, -c], M4, Antisymmetric[{-a, -b}],
  PrintAs → "T̂λ2⁻", OrthogonalTo → {V[a], V[b], V[c]}];
DeclareOrder[TLambdaP2m[-a, -b, -c], 1];
DefTensor[RLambdaP0p[], M4, PrintAs → "R̂λ0⁺"];
DeclareOrder[RLambdaP0p[], 1];
DefTensor[RLambdaP0m[], M4, PrintAs → "R̂λ0⁻"];
DeclareOrder[RLambdaP0m[], 1];
DefTensor[RLambdaP1p[-a, -b], M4,
  Antisymmetric[{-a, -b}], PrintAs → "R̂λ1⁺", OrthogonalTo → {V[a], V[b]}];
DeclareOrder[RLambdaP1p[-a, -b], 1];
DefTensor[RLambdaP1m[-a], M4, PrintAs → "R̂λ1⁻", OrthogonalTo → {V[a]}];
DeclareOrder[RLambdaP1m[-a], 1];
DefTensor[RLambdaP2p[-a, -b], M4,
  Symmetric[{-a, -b}], PrintAs → "R̂λ2⁺", OrthogonalTo → {V[a], V[b]}];
DeclareOrder[RLambdaP2p[-a, -b], 1];
DefTensor[RLambdaP2m[-a, -b, -c], M4, Antisymmetric[{-a, -b}],
  PrintAs → "R̂λ2⁻", OrthogonalTo → {V[a], V[b], V[c]}];
DeclareOrder[RLambdaP2m[-a, -b, -c], 1];
AutomaticRules[RLambdaP2m,
  MakeRule[{RLambdaP2m[a, -b, -a], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[RLambdaP2m, MakeRule[{epsilonG[a, b, c, d] RLambdaP2m[-a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[RLambdaP2p, MakeRule[{RLambdaP2p[a, -a], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[TLambdaP2m, MakeRule[{TLambdaP2m[a, -b, -a], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[TLambdaP2m, MakeRule[{epsilonG[a, b, c, d] TLambdaP2m[-a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];
```

Define $\overset{*}{T}\,J^P$, $\overset{*}{R}\,J^P$

In[∘]:=
```
(*O(3) decomposition of the non-canonical parts of field strengths*)
DefTensor[TPerp0p[], M4, PrintAs → "T⃰0⁺"];
DeclareOrder[TPerp0p[], 1];
DefTensor[TPerp1p[-a, -b], M4, Antisymmetric[{-a, -b}],
  PrintAs → "T⃰1⁺", OrthogonalTo → {V[a], V[b]}];
DeclareOrder[TPerp1p[-a, -b], 1];
DefTensor[TPerp1m[-a], M4, PrintAs → "T⃰1⁻", OrthogonalTo → {V[a]}];
DeclareOrder[TPerp1m[-a], 1];
DefTensor[TPerp2p[-a, -b], M4,
  Symmetric[{-a, -b}], PrintAs → "T⃰2⁺", OrthogonalTo → {V[a], V[b]}];
DeclareOrder[TPerp2p[-a, -b], 1];
DefTensor[RPerp0p[], M4, PrintAs → "R⃰0⁺"];
DeclareOrder[RPerp0p[], 1];
DefTensor[RPerp0m[], M4, PrintAs → "R⃰0⁻"];
DeclareOrder[RPerp0m[], 1];
DefTensor[RPerp1p[-a, -b], M4, Antisymmetric[{-a, -b}],
  PrintAs → "R⃰1⁺", OrthogonalTo → {V[a], V[b]}];
DeclareOrder[RPerp1p[-a, -b], 1];
DefTensor[RPerp1m[-a], M4, PrintAs → "R⃰1⁻", OrthogonalTo → {V[a]}];
DeclareOrder[RPerp1m[-a], 1];
DefTensor[RPerp2p[-a, -b], M4,
  Symmetric[{-a, -b}], PrintAs → "R⃰2⁺", OrthogonalTo → {V[a], V[b]}];
DeclareOrder[RPerp2p[-a, -b], 1];
DefTensor[RPerp2m[-a, -b, -c], M4, Antisymmetric[{-a, -b}],
  PrintAs → "R⃰2⁻", OrthogonalTo → {V[a], V[b], V[c]}];
DeclareOrder[RPerp2m[-a, -b, -c], 1];
AutomaticRules[TPerp2p,
  MakeRule[{TPerp2p[a, -a], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[RPerp2m, MakeRule[{RPerp2m[a, -b, -a], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[RPerp2m, MakeRule[{epsilonG[a, b, c, d] RPerp2m[-a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[RPerp2p, MakeRule[{RPerp2p[a, -a], 0},
   MetricOn → All, ContractMetrics → True]];
```

Alternative human-readable projections $\{{}^{A}\breve{\mathcal{P}}\}$, $\{{}^{E}\breve{\mathcal{P}}\}$

In[∘]:=
```
(*Projections to break the field strengths up into canonical and non-
 canonical parts*)
DefTensor[PPerpTPerp[-e, a, b], M4];
```

```
DefTensor[PPerpTPara[-e, -f, a, b], M4];
DefTensor[PPerpRPerp[-e, -f, a, b, c], M4];
DefTensor[PPerpRPara[-e, -f, -g, a, b, c], M4];
PPerpTPerpDefinition = V[a] PPara[-e, b] /. PADMActivate // ToCanonical;
PPerpTPerpActivate =
  MakeRule[{PPerpTPerp[-e, a, b], Evaluate[PPerpTPerpDefinition]},
   MetricOn → All, ContractMetrics → True];
PPerpTParaDefinition = PPara[-e, a] PPara[-f, b] /. PADMActivate // ToCanonical;
PPerpTParaActivate =
  MakeRule[{PPerpTPara[-e, -f, a, b], Evaluate[PPerpTParaDefinition]},
   MetricOn → All, ContractMetrics → True];
PPerpRPerpDefinition = V[a] PPara[-e, b] PPara[-f, c] /. PADMActivate // ToCanonical;
PPerpRPerpActivate =
  MakeRule[{PPerpRPerp[-e, -f, a, b, c], Evaluate[PPerpRPerpDefinition]},
   MetricOn → All, ContractMetrics → True];
PPerpRParaDefinition = PPara[-e, a] PPara[-f, b] PPara[-g, c] /. PADMActivate //
   ToCanonical;
PPerpRParaActivate = MakeRule[{PPerpRPara[-e, -f, -g, a, b, c],
    Evaluate[PPerpRParaDefinition]}, MetricOn → All, ContractMetrics → True];
PPerpADMTActivate = Join[PPerpTPerpActivate, PPerpTParaActivate];
PPerpADMRActivate = Join[PPerpRPerpActivate, PPerpRParaActivate];

(*Projection operators which define the O(3)
 decomposition of the canonical parts of field strengths*)
DefTensor[PPerpT0p[e, f], M4, PrintAs -> "T0+𝒫̌"];
DefTensor[PPerpT1p[-a, -b, e, f], M4, PrintAs -> "T1+𝒫̌"];
DefTensor[PPerpT1m[-a, e, f], M4, PrintAs -> "T1-𝒫̌"];
DefTensor[PPerpT2p[-a, -b, e, f], M4, PrintAs -> "T2+𝒫̌"];

DefTensor[PPerpR0p[e, f], M4, PrintAs -> "R0+𝒫̌"];
DefTensor[PPerpR0m[e, f, g], M4, PrintAs -> "R0-𝒫̌"];
DefTensor[PPerpR1p[-n, -m, e, f], M4, PrintAs -> "R1+𝒫̌"];
DefTensor[PPerpR1m[-n, e, f, g], M4, PrintAs -> "R1-𝒫̌"];
DefTensor[PPerpR2p[-n, -m, e, f], M4, PrintAs -> "R2+𝒫̌"];
DefTensor[PPerpR2m[-n, -m, -o, e, f, g], M4, PrintAs -> "R2-𝒫̌"];

PPerpT0pDefinition = PPara[e, f] /. PADMActivate // ToCanonical;
PPerpT1pDefinition =
  Antisymmetrize[PPara[-n, e] PPara[-m, f], {-n, -m}] /. PADMActivate // ToCanonical;
PPerpT1mDefinition = PPara[-n, e] /. PADMActivate // ToCanonical;
PPerpT2pDefinition = (Symmetrize[PPara[-n, e] PPara[-m, f], {-n, -m}] -
      (1/3) PPara[-n, -m] PPara[e, f]) /. PADMActivate // ToCanonical;
```

```
PPerpR0pDefinition = -PPara[e, f] /. PADMActivate // ToCanonical;
PPerpR0mDefinition =
  PPara[-i, e] PPara[-j, f] PPara[-k, g] epsilonG[i, j, k, p] V[-p] /. PADMActivate //
   ToCanonical;
PPerpR1pDefinition = Antisymmetrize[PPara[-n, e] PPara[-m, f], {-n, -m}] /.
    PADMActivate // ToCanonical;
PPerpR1mDefinition = PPara[e, g] PPara[-n, f] /. PADMActivate // ToCanonical;
PPerpR2pDefinition = (Symmetrize[PPara[-n, e] PPara[-m, f], {-n, -m}] -
      (1/3) PPara[-n, -m] PPara[e, f]) /. PADMActivate // ToCanonical;
PPerpR2mDefinition = PPara[-a, i] PPara[-b, j] PPara[-c, k] PPara[e, -l]
     PPara[f, -n] PPara[d, -m] (3/4) ((1/3) (2 G[-i, l] G[-j, n] G[-k, m] -
          G[-j, l] G[-k, n] G[-i, m] - G[-k, l] G[-i, n] G[-j, m]) - Antisymmetrize[
        G[-i, -k] G[-j, n] G[l, m], {-i, -j}]) /. PADMActivate // ToCanonical;

PPerpT0pActivate = MakeRule[{PPerpT0p[e, f], Evaluate[PPerpT0pDefinition]},
   MetricOn → All, ContractMetrics → True];
PPerpT1pActivate = MakeRule[{PPerpT1p[-n, -m, e, f], Evaluate[PPerpT1pDefinition]},
   MetricOn → All, ContractMetrics → True];
PPerpT1mActivate = MakeRule[{PPerpT1m[-n, e], Evaluate[PPerpT1mDefinition]},
   MetricOn → All, ContractMetrics → True];
PPerpT2pActivate = MakeRule[{PPerpT2p[-n, -m, e, f], Evaluate[PPerpT2pDefinition]},
   MetricOn → All, ContractMetrics → True];

PPerpR0pActivate = MakeRule[{PPerpR0p[e, f], Evaluate[PPerpR0pDefinition]},
   MetricOn → All, ContractMetrics → True];
PPerpR0mActivate = MakeRule[{PPerpR0m[e, f, g], Evaluate[PPerpR0mDefinition]},
   MetricOn → All, ContractMetrics → True];
PPerpR1pActivate = MakeRule[{PPerpR1p[-n, -m, e, f], Evaluate[PPerpR1pDefinition]},
   MetricOn → All, ContractMetrics → True];
PPerpR1mActivate = MakeRule[{PPerpR1m[-n, e, f, g], Evaluate[PPerpR1mDefinition]},
   MetricOn → All, ContractMetrics → True];
PPerpR2pActivate = MakeRule[{PPerpR2p[-n, -m, e, f], Evaluate[PPerpR2pDefinition]},
   MetricOn → All, ContractMetrics → True];
PPerpR2mActivate = MakeRule[{PPerpR2m[-a, -b, -c, e, f, d],
    Evaluate[PPerpR2mDefinition]}, MetricOn → All, ContractMetrics → True];

(*These rules then expand those canonical
 field strength O(3) projection operators*)
PPerpO3TActivate = Join[PPerpT0pActivate,
   PPerpT1pActivate, PPerpT1mActivate, PPerpT2pActivate];
PPerpO3RActivate = Join[PPerpR0pActivate, PPerpR0mActivate,
   PPerpR1pActivate, PPerpR1mActivate, PPerpR2pActivate, PPerpR2mActivate];
```

Define $\overset{*}{\lambda}\, J^P$

```
In[•]:= DefTensor[TLambdaPerp0p[], M4, PrintAs → "T*λ0⁺"];
DeclareOrder[TLambdaPerp0p[], 1];
DefTensor[TLambdaPerp1p[-a, -b], M4,
  Antisymmetric[{-a, -b}], PrintAs → "T*λ1⁺", OrthogonalTo → {V[a], V[b]}];
DeclareOrder[TLambdaPerp1p[-a, -b], 1];
DefTensor[TLambdaPerp1m[-a], M4, PrintAs → "T*λ1⁻", OrthogonalTo → {V[a]}];
DeclareOrder[TLambdaPerp1m[-a], 1];
DefTensor[TLambdaPerp2p[-a, -b], M4,
  Symmetric[{-a, -b}], PrintAs → "T*λ2⁺", OrthogonalTo → {V[a], V[b]}];
DeclareOrder[TLambdaPerp2p[-a, -b], 1];
DefTensor[RLambdaPerp0p[], M4, PrintAs → "λ*0⁺"];
DeclareOrder[RLambdaPerp0p[], 1];
DefTensor[RLambdaPerp0m[], M4, PrintAs → "λ*0⁻"];
DeclareOrder[RLambdaPerp0m[], 1];
DefTensor[RLambdaPerp1p[-a, -b], M4,
  Antisymmetric[{-a, -b}], PrintAs → "λ*1⁺", OrthogonalTo → {V[a], V[b]}];
DeclareOrder[RLambdaPerp1p[-a, -b], 1];
DefTensor[RLambdaPerp1m[-a], M4, PrintAs → "λ*1⁻", OrthogonalTo → {V[a]}];
DeclareOrder[RLambdaPerp1m[-a], 1];
DefTensor[RLambdaPerp2p[-a, -b], M4,
  Symmetric[{-a, -b}], PrintAs → "λ*2⁺", OrthogonalTo → {V[a], V[b]}];
DeclareOrder[RLambdaPerp2p[-a, -b], 1];
DefTensor[RLambdaPerp2m[-a, -b, -c], M4, Antisymmetric[{-a, -b}],
  PrintAs → "λ*2⁻", OrthogonalTo → {V[a], V[b], V[c]}];
DeclareOrder[RLambdaPerp2m[-a, -b, -c], 1];
AutomaticRules[RLambdaPerp2m,
  MakeRule[{RLambdaPerp2m[a, -b, -a], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[RLambdaPerp2m,
  MakeRule[{epsilonG[a, b, c, d] RLambdaPerp2m[-a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[RLambdaPerp2p, MakeRule[{RLambdaPerp2p[a, -a], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[TLambdaPerp2p, MakeRule[{TLambdaPerp2p[a, -a], 0},
   MetricOn → All, ContractMetrics → True]];
```

Nester form $\hat{T}$, $\hat{R}$

```
In[•]:= (*These rules then expand the O(3) parts in terms of the canonical parts*)
TP0mDefinition =
  PT0m[e, f, g] PTPara[-e, -f, -g, a, b, c] TP[-a, -b, -c] /. PO3TActivate /.
```

```
    PADMTActivate // ToCanonical;
TP1pDefinition = PT1p[-n, -m, e, f] PTPerp[-e, -f, a, b, c] TP[-a, -b, -c] /.
     PO3TActivate /. PADMTActivate // ToCanonical;
TP1mDefinition = PT1m[-n, e, f, g] PTPara[-e, -f, -g, a, b, c] TP[-a, -b, -c] /.
     PO3TActivate /. PADMTActivate // ToCanonical;
TP2mDefinition = PT2m[-n, -m, -o, e, f, g] PTPara[-e, -f, -g, a, b, c] TP[-a, -b, -c] /.
     PO3TActivate /. PADMTActivate // ToCanonical;

RP0pDefinition =
  PR0p[e, f, g, h] PRPara[-e, -f, -g, -h, a, b, c, d] RP[-a, -b, -c, -d] /.
     PO3RActivate /. PADMRActivate // ToCanonical;
RP0mDefinition = PR0m[e, f, g] PRPerp[-e, -f, -g, a, b, c, d] RP[-a, -b, -c, -d] /.
     PO3RActivate /. PADMRActivate // ToCanonical;
RP1pDefinition = PR1p[-n, -m, e, f, g, h] PRPara[-e, -f, -g, -h, a, b, c, d]
      RP[-a, -b, -c, -d] /. PO3RActivate /. PADMRActivate // ToCanonical;
RP1mDefinition = PR1m[-n, e, f, g] PRPerp[-e, -f, -g, a, b, c, d] RP[-a, -b, -c, -d] /.
     PO3RActivate /. PADMRActivate // ToCanonical;
RP2pDefinition = PR2p[-n, -m, e, f, g, h] PRPara[-e, -f, -g, -h, a, b, c, d]
      RP[-a, -b, -c, -d] /. PO3RActivate /. PADMRActivate // ToCanonical;
RP2mDefinition = PR2m[-n, -m, -o, e, f, g] PRPerp[-e, -f, -g, a, b, c, d]
      RP[-a, -b, -c, -d] /. PO3RActivate /. PADMRActivate // ToCanonical;

TP0mActivate = MakeRule[{TP0m[], Scalar[Evaluate[TP0mDefinition]]},
   MetricOn → All, ContractMetrics → True];
TP1pActivate = MakeRule[{TP1p[-n, -m], Evaluate[TP1pDefinition]},
   MetricOn → All, ContractMetrics → True];
TP1mActivate = MakeRule[{TP1m[-n], Evaluate[TP1mDefinition]},
   MetricOn → All, ContractMetrics → True];
TP2mActivate = MakeRule[{TP2m[-n, -m, -o], Evaluate[TP2mDefinition]},
   MetricOn → All, ContractMetrics → True];

RP0pActivate = MakeRule[{RP0p[], Scalar[Evaluate[RP0pDefinition]]},
   MetricOn → All, ContractMetrics → True];
RP0mActivate = MakeRule[{RP0m[], Scalar[Evaluate[RP0mDefinition]]},
   MetricOn → All, ContractMetrics → True];
RP1pActivate = MakeRule[{RP1p[-n, -m], Evaluate[RP1pDefinition]},
   MetricOn → All, ContractMetrics → True];
RP1mActivate = MakeRule[{RP1m[-n], Evaluate[RP1mDefinition]},
   MetricOn → All, ContractMetrics → True];
RP2pActivate = MakeRule[{RP2p[-n, -m], Evaluate[RP2pDefinition]},
   MetricOn → All, ContractMetrics → True];
RP2mActivate = MakeRule[{RP2m[-n, -m, -o], Evaluate[RP2mDefinition]},
   MetricOn → All, ContractMetrics → True];
```

```
TPO3Activate = Join[TP0mActivate, TP1pActivate, TP1mActivate, TP2mActivate];
RPO3Activate = Join[RP0pActivate, RP0mActivate,
   RP1pActivate, RP1mActivate, RP2pActivate, RP2mActivate];

TPDefinition = V[-a] TP1p[-b, -c] +
      - (1/6) PT0m[-a, -b, -c] TP0m[] +
       Antisymmetrize[-PPara[-a, -b] TP1m[-c], {-b, -c}] +
      (4/3) TP2m[-b, -c, -a] /. PO3TActivate /. PADMActivate // ToCanonical;

DefTensor[RPPara[-a, -b, -c, -d], M4,
  {Antisymmetric[{-a, -b}], Antisymmetric[{-c, -d}]},
  OrthogonalTo → {V[a], V[b], V[c], V[d]}];
DeclareOrder[RPPara[-a, -b, -c, -d], 1];
DefTensor[RPPerp[-a, -b, -c], M4,
  Antisymmetric[{-b, -c}], OrthogonalTo → {V[a], V[b], V[c]}];
DeclareOrder[RPPerp[-a, -b, -c], 1];

RPParaDefinition =
  - (1/6) (PPara[-a, -d] PPara[-b, -c] - PPara[-a, -c] PPara[-b, -d]) RP0p[] -
   (PPara[-b, -d] RP1p[-a, -c] - PPara[-b, -c] RP1p[-a, -d] -
     PPara[-a, -d] RP1p[-b, -c] + PPara[-a, -c] RP1p[-b, -d]) +
   (PPara[-b, -d] RP2p[-a, -c] - PPara[-b, -c] RP2p[-a, -d] -
     PPara[-a, -d] RP2p[-b, -c] + PPara[-a, -c] RP2p[-b, -d]);
RPPerpDefinition = - (1/6) PR0m[-a, -b, -c] RP0m[] +
   Antisymmetrize[-PPara[-a, -b] RP1m[-c], {-b, -c}] + (4/3) RP2m[-b, -c, -a];

RPParaActivate = MakeRule[{RPPara[-a, -b, -c, -d], Evaluate[RPParaDefinition]},
   MetricOn → All, ContractMetrics → True];
RPPerpActivate = MakeRule[{RPPerp[-a, -b, -c], Evaluate[RPPerpDefinition]},
   MetricOn → All, ContractMetrics → True];
RPParaPerpActivate = Join[RPParaActivate, RPPerpActivate];

RPDefinition =
  RPPara[-a, -b, -c, -d] + 2 Antisymmetrize[V[-a] RPPerp[-b, -c, -d], {-a, -b}] /.
      RPParaPerpActivate /. PO3RActivate /. PADMActivate // ToCanonical;

TPDefinition =
  TPDefinition // CollectTensors // ScreenDollarIndices // CollectTensors;
TPDefinition = TPDefinition /. TPO3Activate // CollectTensors //
    ScreenDollarIndices // CollectTensors;
RPDefinition = RPDefinition // CollectTensors // ScreenDollarIndices //
   CollectTensors;
```

```
TPActivate = MakeRule[{TP[-a, -b, -c], Evaluate[TPDefinition]},
   MetricOn → All, ContractMetrics → True];
RPActivate = MakeRule[{RP[-a, -b, -c, -d], Evaluate[RPDefinition]},
   MetricOn → All, ContractMetrics → True];
StrengthPToStrengthPO3 = Join[TPActivate, RPActivate];
```

Nester form $\hat{\lambda}$

```
In[•]:= (*These rules then expand the O(3) parts in terms of the canonical parts*)

TLambdaP0mDefinition =
  PT0m[e, f, g] PTPara[-e, -f, -g, a, b, c] TLambdaP[-a, -b, -c] /. PO3TActivate /.
    PADMTActivate // ToCanonical;
TLambdaP1pDefinition = PT1p[-n, -m, e, f] PTPerp[-e, -f, a, b, c]
      TLambdaP[-a, -b, -c] /. PO3TActivate /. PADMTActivate // ToCanonical;
TLambdaP1mDefinition = PT1m[-n, e, f, g] PTPara[-e, -f, -g, a, b, c]
      TLambdaP[-a, -b, -c] /. PO3TActivate /. PADMTActivate // ToCanonical;
TLambdaP2mDefinition = PT2m[-n, -m, -o, e, f, g] PTPara[-e, -f, -g, a, b, c]
      TLambdaP[-a, -b, -c] /. PO3TActivate /. PADMTActivate // ToCanonical;

RLambdaP0pDefinition =
  PR0p[e, f, g, h] PRPara[-e, -f, -g, -h, a, b, c, d] RLambdaP[-a, -b, -c, -d] /.
     PO3RActivate /. PADMRActivate // ToCanonical;
RLambdaP0mDefinition = PR0m[e, f, g] PRPerp[-e, -f, -g, a, b, c, d]
      RLambdaP[-a, -b, -c, -d] /. PO3RActivate /. PADMRActivate // ToCanonical;
RLambdaP1pDefinition = PR1p[-n, -m, e, f, g, h] PRPara[-e, -f, -g, -h, a, b, c, d]
      RLambdaP[-a, -b, -c, -d] /. PO3RActivate /. PADMRActivate // ToCanonical;
RLambdaP1mDefinition = PR1m[-n, e, f, g] PRPerp[-e, -f, -g, a, b, c, d]
      RLambdaP[-a, -b, -c, -d] /. PO3RActivate /. PADMRActivate // ToCanonical;
RLambdaP2pDefinition = PR2p[-n, -m, e, f, g, h] PRPara[-e, -f, -g, -h, a, b, c, d]
      RLambdaP[-a, -b, -c, -d] /. PO3RActivate /. PADMRActivate // ToCanonical;
RLambdaP2mDefinition = PR2m[-n, -m, -o, e, f, g] PRPerp[-e, -f, -g, a, b, c, d]
      RLambdaP[-a, -b, -c, -d] /. PO3RActivate /. PADMRActivate // ToCanonical;

TLambdaP0mActivate =
  MakeRule[{TLambdaP0m[], Scalar[Evaluate[TLambdaP0mDefinition]]},
   MetricOn → All, ContractMetrics → True];
TLambdaP1pActivate = MakeRule[{TLambdaP1p[-n, -m], Evaluate[TLambdaP1pDefinition]},
   MetricOn → All, ContractMetrics → True];
TLambdaP1mActivate = MakeRule[{TLambdaP1m[-n], Evaluate[TLambdaP1mDefinition]},
   MetricOn → All, ContractMetrics → True];
TLambdaP2mActivate = MakeRule[{TLambdaP2m[-n, -m, -o],
    Evaluate[TLambdaP2mDefinition]}, MetricOn → All, ContractMetrics → True];
```

```
RLambdaP0pActivate =
  MakeRule[{RLambdaP0p[], Scalar[Evaluate[RLambdaP0pDefinition]]},
   MetricOn → All, ContractMetrics → True];
RLambdaP0mActivate = MakeRule[{RLambdaP0m[], Scalar[
     Evaluate[RLambdaP0mDefinition]]}, MetricOn → All, ContractMetrics → True];
RLambdaP1pActivate = MakeRule[{RLambdaP1p[-n, -m], Evaluate[RLambdaP1pDefinition]},
   MetricOn → All, ContractMetrics → True];
RLambdaP1mActivate = MakeRule[{RLambdaP1m[-n], Evaluate[RLambdaP1mDefinition]},
   MetricOn → All, ContractMetrics → True];
RLambdaP2pActivate = MakeRule[{RLambdaP2p[-n, -m], Evaluate[RLambdaP2pDefinition]},
   MetricOn → All, ContractMetrics → True];
RLambdaP2mActivate = MakeRule[{RLambdaP2m[-n, -m, -o],
    Evaluate[RLambdaP2mDefinition]}, MetricOn → All, ContractMetrics → True];

TLambdaPO3Activate = Join[TLambdaP0mActivate,
   TLambdaP1pActivate, TLambdaP1mActivate, TLambdaP2mActivate];
RLambdaPO3Activate = Join[RLambdaP0pActivate, RLambdaP0mActivate,
   RLambdaP1pActivate, RLambdaP1mActivate,
   RLambdaP2pActivate, RLambdaP2mActivate];

TLambdaPDefinition = V[-a] TLambdaP1p[-b, -c] +
      - (1/6) PT0m[-a, -b, -c] TLambdaP0m[] +
      Antisymmetrize[-PPara[-a, -b] TLambdaP1m[-c], {-b, -c}] +
      (4/3) TLambdaP2m[-b, -c, -a] /. PO3TActivate /. PADMActivate // ToCanonical;

DefTensor[RLambdaPPara[-a, -b, -c, -d],
  M4, {Antisymmetric[{-a, -b}], Antisymmetric[{-c, -d}]},
  OrthogonalTo → {V[a], V[b], V[c], V[d]}];
DeclareOrder[RLambdaPPara[-a, -b, -c, -d], 1];
DefTensor[RLambdaPPerp[-a, -b, -c], M4,
  Antisymmetric[{-b, -c}], OrthogonalTo → {V[a], V[b], V[c]}];
DeclareOrder[RLambdaPPerp[-a, -b, -c], 1];

RLambdaPParaDefinition =
  - (1/6) (PPara[-a, -d] PPara[-b, -c] - PPara[-a, -c] PPara[-b, -d]) RLambdaP0p[] -
   (PPara[-b, -d] RLambdaP1p[-a, -c] - PPara[-b, -c] RLambdaP1p[-a, -d] -
     PPara[-a, -d] RLambdaP1p[-b, -c] + PPara[-a, -c] RLambdaP1p[-b, -d]) +
   (PPara[-b, -d] RLambdaP2p[-a, -c] - PPara[-b, -c] RLambdaP2p[-a, -d] -
     PPara[-a, -d] RLambdaP2p[-b, -c] + PPara[-a, -c] RLambdaP2p[-b, -d]);
RLambdaPPerpDefinition = - (1/6) PR0m[-a, -b, -c] RLambdaP0m[] + Antisymmetrize[
    -PPara[-a, -b] RLambdaP1m[-c], {-b, -c}] + (4/3) RLambdaP2m[-b, -c, -a];

RLambdaPParaActivate =
```

```
  MakeRule[{RLambdaPPara[-a, -b, -c, -d], Evaluate[RLambdaPParaDefinition]},
   MetricOn → All, ContractMetrics → True];
RLambdaPPerpActivate = MakeRule[{RLambdaPPerp[-a, -b, -c],
    Evaluate[RLambdaPPerpDefinition]}, MetricOn → All, ContractMetrics → True];
RLambdaPParaPerpActivate = Join[RLambdaPParaActivate, RLambdaPPerpActivate];

RLambdaPDefinition = RLambdaPPara[-a, -b, -c, -d] +
      2 Antisymmetrize[V[-a] RLambdaPPerp[-b, -c, -d], {-a, -b}] /.
     RLambdaPParaPerpActivate /. PO3RActivate /. PADMActivate // ToCanonical;

TLambdaPDefinition =
  TLambdaPDefinition // CollectTensors // ScreenDollarIndices // CollectTensors;
RLambdaPDefinition = RLambdaPDefinition // CollectTensors //
    ScreenDollarIndices // CollectTensors;

TLambdaPActivate = MakeRule[{TLambdaP[-a, -b, -c], Evaluate[TLambdaPDefinition]},
   MetricOn → All, ContractMetrics → True];
RLambdaPActivate = MakeRule[{RLambdaP[-a, -b, -c, -d],
    Evaluate[RLambdaPDefinition]}, MetricOn → All, ContractMetrics → True];
StrengthLambdaPToStrengthLambdaPO3 = Join[TLambdaPActivate, RLambdaPActivate];
```

Nester form $\overset{*}{T}, \overset{*}{R}$

```
In[•]:= (*These rules then expand the O(3) parts in terms of the canonical parts*)
TPerp0pDefinition =
  PPerpT0p[e, f] PPerpTPara[-e, -f, a, b] TPerp[-a, -b] /. PPerpO3TActivate /.
    PPerpADMTActivate // ToCanonical;
TPerp1pDefinition = PPerpT1p[-n, -m, e, f] PPerpTPara[-e, -f, a, b] TPerp[-a, -b] /.
     PPerpO3TActivate /. PPerpADMTActivate // ToCanonical;
TPerp1mDefinition = PPerpT1m[-n, e] PPerpTPerp[-e, a, b] TPerp[-a, -b] /.
     PPerpO3TActivate /. PPerpADMTActivate // ToCanonical;
TPerp2pDefinition = PPerpT2p[-n, -m, e, f] PPerpTPara[-e, -f, a, b] TPerp[-a, -b] /.
     PPerpO3TActivate /. PPerpADMTActivate // ToCanonical;

RPerp0pDefinition =
  PPerpR0p[e, f] PPerpRPerp[-e, -f, a, b, c] RPerp[-a, -b, -c] /. PPerpO3RActivate /.
    PPerpADMRActivate // ToCanonical;
RPerp0mDefinition = PPerpR0m[e, f, g] PPerpRPara[-e, -f, -g, a, b, c]
      RPerp[-a, -b, -c] /. PPerpO3RActivate /. PPerpADMRActivate // ToCanonical;
RPerp1pDefinition = PPerpR1p[-n, -m, e, f] PPerpRPerp[-e, -f, a, b, c]
      RPerp[-a, -b, -c] /. PPerpO3RActivate /. PPerpADMRActivate // ToCanonical;
RPerp1mDefinition = PPerpR1m[-n, e, f, g] PPerpRPara[-e, -f, -g, a, b, c]
      RPerp[-a, -b, -c] /. PPerpO3RActivate /. PPerpADMRActivate // ToCanonical;
```

```
RPerp2pDefinition = PPerpR2p[-n, -m, e, f] PPerpRPerp[-e, -f, a, b, c]
      RPerp[-a, -b, -c] /. PPerpO3RActivate /. PPerpADMRActivate // ToCanonical;
RPerp2mDefinition = PPerpR2m[-n, -m, -o, e, f, g] PPerpRPara[-e, -f, -g, a, b, c]
      RPerp[-a, -b, -c] /. PPerpO3RActivate /. PPerpADMRActivate // ToCanonical;

TPerp0pActivate = MakeRule[{TPerp0p[], Scalar[Evaluate[TPerp0pDefinition]]},
   MetricOn → All, ContractMetrics → True];
TPerp1pActivate = MakeRule[{TPerp1p[-n, -m], Evaluate[TPerp1pDefinition]},
   MetricOn → All, ContractMetrics → True];
TPerp1mActivate = MakeRule[{TPerp1m[-n], Evaluate[TPerp1mDefinition]},
   MetricOn → All, ContractMetrics → True];
TPerp2pActivate = MakeRule[{TPerp2p[-n, -m], Evaluate[TPerp2pDefinition]},
   MetricOn → All, ContractMetrics → True];

RPerp0pActivate = MakeRule[{RPerp0p[], Scalar[Evaluate[RPerp0pDefinition]]},
   MetricOn → All, ContractMetrics → True];
RPerp0mActivate = MakeRule[{RPerp0m[], Scalar[Evaluate[RPerp0mDefinition]]},
   MetricOn → All, ContractMetrics → True];
RPerp1pActivate = MakeRule[{RPerp1p[-n, -m], Evaluate[RPerp1pDefinition]},
   MetricOn → All, ContractMetrics → True];
RPerp1mActivate = MakeRule[{RPerp1m[-n], Evaluate[RPerp1mDefinition]},
   MetricOn → All, ContractMetrics → True];
RPerp2pActivate = MakeRule[{RPerp2p[-n, -m], Evaluate[RPerp2pDefinition]},
   MetricOn → All, ContractMetrics → True];
RPerp2mActivate = MakeRule[{RPerp2m[-n, -m, -o], Evaluate[RPerp2mDefinition]},
   MetricOn → All, ContractMetrics → True];

TPerpO3Activate =
  Join[TPerp0pActivate, TPerp1pActivate, TPerp1mActivate, TPerp2pActivate];
RPerpO3Activate = Join[RPerp0pActivate, RPerp0mActivate,
   RPerp1pActivate, RPerp1mActivate, RPerp2pActivate, RPerp2mActivate];

TPerpDefinition = V[-a] TPerp1m[-b] +
      TPerp1p[-a, -b] +
      TPerp2p[-a, -b] +
      (1/3) PPara[-a, -b] TPerp0p[] /. PPerpO3TActivate /. PADMActivate //
   ToCanonical;

DefTensor[RPerpPerp[-a, -b], M4, OrthogonalTo → {V[a], V[b]}];
DeclareOrder[RPerpPerp[-a, -b], 1];
DefTensor[RPerpPara[-a, -b, -c], M4,
  Antisymmetric[{-a, -b}], OrthogonalTo → {V[a], V[b], V[c]}];
DeclareOrder[RPerpPara[-a, -b, -c], 1];
```

```
RPerpPerpDefinition = RPerp1p[-a, -b] +
      RPerp2p[-a, -b] -
      (1/3) PPara[-a, -b] RPerp0p[] /. PPerpO3RActivate /. PADMActivate //
   ToCanonical;
RPerpParaDefinition = -(1/6) PR0m[-a, -b, -c] RPerp0m[] -
      Antisymmetrize[-PPara[-c, -a] RPerp1m[-b], {-a, -b}] +
      (4/3) RPerp2m[-a, -b, -c] /. PPerpO3RActivate /. PADMActivate // ToCanonical;

RPerpPerpActivate = MakeRule[{RPerpPerp[-a, -b], Evaluate[RPerpPerpDefinition]},
   MetricOn → All, ContractMetrics → True];
RPerpParaActivate = MakeRule[{RPerpPara[-a, -b, -c], Evaluate[RPerpParaDefinition]},
   MetricOn → All, ContractMetrics → True];
RPerpParaPerpActivate = Join[RPerpParaActivate, RPerpPerpActivate];

RPerpDefinition =
  RPerpPara[-a, -b, -c] + 2 Antisymmetrize[V[-a] RPerpPerp[-b, -c], {-a, -b}] /.
      RPerpParaPerpActivate /. PO3RActivate /. PADMActivate // ToCanonical;

TPerpDefinition =
  TPerpDefinition // CollectTensors // ScreenDollarIndices // CollectTensors;
(*
TPerpDefinition=TPerpDefinition/.TPerpO3Activate//NoScalar//ToNewCanonical;
Print[TPerpDefinition];
*)
RPerpDefinition =
  RPerpDefinition // CollectTensors // ScreenDollarIndices // CollectTensors;
(*
RPerpDefinition=RPerpDefinition/.RPerpO3Activate//NoScalar;
RPerpDefinition=RPerpDefinition//ToNewCanonical;
RPerpDefinition=RPerpDefinition//ToCanonical;
Print[RPerpDefinition];
*)

TPerpActivate = MakeRule[{TPerp[-a, -b], Evaluate[TPerpDefinition]},
   MetricOn → All, ContractMetrics → True];
RPerpActivate = MakeRule[{RPerp[-a, -b, -c], Evaluate[RPerpDefinition]},
   MetricOn → All, ContractMetrics → True];
StrengthPerpToStrengthPerpO3 = Join[TPerpActivate, RPerpActivate];
```

Nester form $\overset{*}{\lambda}$

```
In[•]:= (*These rules then expand the O(3) parts in terms of the canonical parts*)
TLambdaPerp0pDefinition =
```

```
  PPerpT0p[e, f] PPerpTPara[-e, -f, a, b] TLambdaPerp[-a, -b] /. PPerpO3TActivate /.
    PPerpADMTActivate // ToCanonical;
TLambdaPerp1pDefinition = PPerpT1p[-n, -m, e, f] PPerpTPara[-e, -f, a, b]
      TLambdaPerp[-a, -b] /. PPerpO3TActivate /. PPerpADMTActivate // ToCanonical;
TLambdaPerp1mDefinition = PPerpT1m[-n, e] PPerpTPerp[-e, a, b] TLambdaPerp[-a, -b] /.
     PPerpO3TActivate /. PPerpADMTActivate // ToCanonical;
TLambdaPerp2pDefinition = PPerpT2p[-n, -m, e, f] PPerpTPara[-e, -f, a, b]
      TLambdaPerp[-a, -b] /. PPerpO3TActivate /. PPerpADMTActivate // ToCanonical;

RLambdaPerp0pDefinition =
  PPerpR0p[e, f] PPerpRPerp[-e, -f, a, b, c] RLambdaPerp[-a, -b, -c] /.
     PPerpO3RActivate /. PPerpADMRActivate // ToCanonical;
RLambdaPerp0mDefinition = PPerpR0m[e, f, g] PPerpRPara[-e, -f, -g, a, b, c]
      RLambdaPerp[-a, -b, -c] /.
     PPerpO3RActivate /. PPerpADMRActivate // ToCanonical;
RLambdaPerp1pDefinition = PPerpR1p[-n, -m, e, f]
      PPerpRPerp[-e, -f, a, b, c] RLambdaPerp[-a, -b, -c] /.
     PPerpO3RActivate /. PPerpADMRActivate // ToCanonical;
RLambdaPerp1mDefinition = PPerpR1m[-n, e, f, g]
      PPerpRPara[-e, -f, -g, a, b, c] RLambdaPerp[-a, -b, -c] /.
     PPerpO3RActivate /. PPerpADMRActivate // ToCanonical;
RLambdaPerp2pDefinition = PPerpR2p[-n, -m, e, f]
      PPerpRPerp[-e, -f, a, b, c] RLambdaPerp[-a, -b, -c] /.
     PPerpO3RActivate /. PPerpADMRActivate // ToCanonical;
RLambdaPerp2mDefinition = PPerpR2m[-n, -m, -o, e, f, g]
      PPerpRPara[-e, -f, -g, a, b, c] RLambdaPerp[-a, -b, -c] /.
     PPerpO3RActivate /. PPerpADMRActivate // ToCanonical;

TLambdaPerp0pActivate =
  MakeRule[{TLambdaPerp0p[], Scalar[Evaluate[TLambdaPerp0pDefinition]]},
   MetricOn → All, ContractMetrics → True];
TLambdaPerp1pActivate = MakeRule[{TLambdaPerp1p[-n, -m],
    Evaluate[TLambdaPerp1pDefinition]}, MetricOn → All, ContractMetrics → True];
TLambdaPerp1mActivate = MakeRule[{TLambdaPerp1m[-n],
    Evaluate[TLambdaPerp1mDefinition]}, MetricOn → All, ContractMetrics → True];
TLambdaPerp2pActivate = MakeRule[{TLambdaPerp2p[-n, -m],
    Evaluate[TLambdaPerp2pDefinition]}, MetricOn → All, ContractMetrics → True];

RLambdaPerp0pActivate =
  MakeRule[{RLambdaPerp0p[], Scalar[Evaluate[RLambdaPerp0pDefinition]]},
   MetricOn → All, ContractMetrics → True];
RLambdaPerp0mActivate = MakeRule[{RLambdaPerp0m[], Scalar[
     Evaluate[RLambdaPerp0mDefinition]]}, MetricOn → All, ContractMetrics → True];
RLambdaPerp1pActivate = MakeRule[{RLambdaPerp1p[-n, -m],
```

```
     Evaluate[RLambdaPerp1pDefinition]}, MetricOn → All, ContractMetrics → True];
RLambdaPerp1mActivate = MakeRule[{RLambdaPerp1m[-n],
     Evaluate[RLambdaPerp1mDefinition]}, MetricOn → All, ContractMetrics → True];
RLambdaPerp2pActivate = MakeRule[{RLambdaPerp2p[-n, -m],
     Evaluate[RLambdaPerp2pDefinition]}, MetricOn → All, ContractMetrics → True];
RLambdaPerp2mActivate = MakeRule[{RLambdaPerp2m[-n, -m, -o],
     Evaluate[RLambdaPerp2mDefinition]}, MetricOn → All, ContractMetrics → True];

TLambdaPerpO3Activate = Join[TLambdaPerp0pActivate,
   TLambdaPerp1pActivate, TLambdaPerp1mActivate, TLambdaPerp2pActivate];
RLambdaPerpO3Activate = Join[RLambdaPerp0pActivate, RLambdaPerp0mActivate,
   RLambdaPerp1pActivate, RLambdaPerp1mActivate,
   RLambdaPerp2pActivate, RLambdaPerp2mActivate];

TLambdaPerpDefinition = V[-a] TLambdaPerp1m[-b] +
      TLambdaPerp1p[-a, -b] +
      TLambdaPerp2p[-a, -b] +
      (1/3) PPara[-a, -b] TLambdaPerp0p[] /. PPerpO3TActivate /. PADMActivate //
   ToCanonical;

DefTensor[RLambdaPerpPerp[-a, -b], M4, OrthogonalTo → {V[a], V[b]}];
DeclareOrder[RLambdaPerpPerp[-a, -b], 1];
DefTensor[RLambdaPerpPara[-a, -b, -c], M4,
  Antisymmetric[{-a, -b}], OrthogonalTo → {V[a], V[b], V[c]}];
DeclareOrder[RLambdaPerpPara[-a, -b, -c], 1];

RLambdaPerpPerpDefinition = RLambdaPerp1p[-a, -b] +
      RLambdaPerp2p[-a, -b] -
      (1/3) PPara[-a, -b] RLambdaPerp0p[] /. PPerpO3RActivate /. PADMActivate //
   ToCanonical;
RLambdaPerpParaDefinition = -(1/6) PR0m[-a, -b, -c] RLambdaPerp0m[] -
      Antisymmetrize[-PPara[-c, -a] RLambdaPerp1m[-b], {-a, -b}] + (4/3)
       RLambdaPerp2m[-a, -b, -c] /. PPerpO3RActivate /. PADMActivate // ToCanonical;

RLambdaPerpPerpActivate =
  MakeRule[{RLambdaPerpPerp[-a, -b], Evaluate[RLambdaPerpPerpDefinition]},
   MetricOn → All, ContractMetrics → True];
RLambdaPerpParaActivate = MakeRule[{RLambdaPerpPara[-a, -b, -c],
     Evaluate[RLambdaPerpParaDefinition]}, MetricOn → All, ContractMetrics → True];
RLambdaPerpParaPerpActivate = Join[RLambdaPerpParaActivate,
   RLambdaPerpPerpActivate];

RLambdaPerpDefinition = RLambdaPerpPara[-a, -b, -c] +
```

```
      2 Antisymmetrize[V[-a] RLambdaPerpPerp[-b, -c], {-a, -b}] /.
     RLambdaPerpParaPerpActivate /. PO3RActivate /. PADMActivate // ToCanonical;

TLambdaPerpDefinition =
  TLambdaPerpDefinition // CollectTensors // ScreenDollarIndices // CollectTensors;

RLambdaPerpDefinition =
  RLambdaPerpDefinition // CollectTensors // ScreenDollarIndices // CollectTensors;
(*
RPerpDefinition=RPerpDefinition/.RPerpO3Activate//NoScalar;
RPerpDefinition=RPerpDefinition//ToNewCanonical;
RPerpDefinition=RPerpDefinition//ToCanonical;
Print[RPerpDefinition];
*)

TLambdaPerpActivate =
  MakeRule[{TLambdaPerp[-a, -b], Evaluate[TLambdaPerpDefinition]},
   MetricOn → All, ContractMetrics → True];
RLambdaPerpActivate = MakeRule[{RLambdaPerp[-a, -b, -c],
    Evaluate[RLambdaPerpDefinition]}, MetricOn → All, ContractMetrics → True];
StrengthLambdaPerpToStrengthLambdaPerpO3 =
  Join[TLambdaPerpActivate, RLambdaPerpActivate];
(*Again used to be Join...*)
```

Nester form $\hat{\pi}\, b$, $\hat{\pi}\, A$

```
In[•]:= BPiPDefinition = ((1/3) PPara[-n, -m] PiPB0p[] +
            PiPB1p[-n, -m] +
            PiPB2p[-n, -m] +
           V[-n] PiPB1m[-m]) /. PO3PiActivate /. PADMActivate // ToNewCanonical;

    APiPDefinition =
      (Antisymmetrize[ 2 Antisymmetrize[V[-n] (1/3) PPara[-m, -o] PiPA0p[], {-n, -m}] +
              2 Antisymmetrize[V[-n] PiPA1p[-m, -o], {-n, -m}] +
              2 Antisymmetrize[V[-n] PiPA2p[-m, -o], {-n, -m}] +
              (-1/6) PA0m[-n, -m, -o] PiPA0m[] +
              Antisymmetrize[-PPara[-m, -o] PiPA1m[-n], {-m, -n}] +
              (4/3) PiPA2m[-n, -m, -o], {-n, -m}]) /. PO3PiActivate /. PADMActivate //
       ToNewCanonical;

    BPiPActivate = MakeRule[{BPiP[-n, -m], Evaluate[BPiPDefinition]},
       MetricOn → All, ContractMetrics → True];
    APiPActivate = MakeRule[{APiP[-n, -m, -o], Evaluate[APiPDefinition]},
       MetricOn → All, ContractMetrics → True];
    PiPToPiPO3 = Join[BPiPActivate, APiPActivate];
```

ORPHAN

```
In[•]:= (*
    DefTensor[TheB0p[],M4,PrintAs→"θb0⁺"];
    DefTensor[TheB1p[-a,-b],M4,Antisymmetric[{-a,-b}],PrintAs→"θb1⁺"];
    DefTensor[TheB1m[-a],M4,PrintAs→"θb1⁻"];
    DefTensor[TheB2p[-a,-b],M4,Symmetric[{-a,-b}],PrintAs→"θb2⁺"];
    DefTensor[TheB2m[-a,-b,-c],M4,Antisymmetric[{-a,-b}],PrintAs→"θb2⁻"];
    DefTensor[TheA0p[],M4,PrintAs→"θA0⁺"];
    DefTensor[TheA0m[],M4,PrintAs→"θA0⁻"];
    DefTensor[TheA1p[-a,-b],M4,Antisymmetric[{-a,-b}],PrintAs→"θA1⁺"];
    DefTensor[TheA1m[-a],M4,PrintAs→"θA1⁻"];
    DefTensor[TheA2p[-a,-b],M4,Symmetric[{-a,-b}],PrintAs→"θA2⁺"];
    DefTensor[TheA2m[-a,-b,-c],M4,Antisymmetric[{-a,-b}],PrintAs→"θA2⁻"];
    AutomaticRules[TheA2m,
     MakeRule[{TheA2m[a,-b,-a],0},MetricOn→All,ContractMetrics→True]];
    AutomaticRules[TheA2m,MakeRule[{epsilonG[a,b,c,d]TheA2m[-a,-b,-c],0},
      MetricOn→All,ContractMetrics→True]];

    TheA0mDefinition=Ji[]PiPA0m[]+16Alp6 RP0m[]/.PADMActivate//ToCanonical;
    TheA1pDefinition=Ji[]PiPA1p[-i,-j]+8Alp6 RP1p[-i,-j]/.PADMActivate//ToCanonical;
    TheA1mDefinition=Ji[]PiPA1m[-i]+16Alp6 RP1m[-i]/.PADMActivate//ToCanonical;
    TheA2pDefinition=Ji[]PiPA2p[-i,-j]+8Alp6 RP2p[-i,-j]/.PADMActivate//ToCanonical;
    TheA2mDefinition=RP2m[-i,-j,-k]/.PADMActivate//ToCanonical;
```

```
TheB1pDefinition=TP1p[-i,-j]/.PADMActivate//ToCanonical;
TheB2mDefinition=TP2m[-i,-j,-k]/.PADMActivate//ToCanonical;

TheA0mActivate=MakeRule[{TheA0m[],Evaluate[TheA0mDefinition]},
  MetricOn→All,ContractMetrics→True];
TheA1pActivate=MakeRule[{TheA1p[-i,-j],Evaluate[TheA1pDefinition]},
  MetricOn→All,ContractMetrics→True];
TheA1mActivate=MakeRule[{TheA1m[-i],Evaluate[TheA1mDefinition]},
  MetricOn→All,ContractMetrics→True];
TheA2pActivate=MakeRule[{TheA2p[-i,-j],Evaluate[TheA2pDefinition]},
  MetricOn→All,ContractMetrics→True];
TheA2mActivate=MakeRule[{TheA2m[-i,-j,-k],Evaluate[TheA2mDefinition]},
  MetricOn→All,ContractMetrics→True];
TheB1pActivate=MakeRule[{TheB1p[-i,-j],Evaluate[TheB1pDefinition]},
  MetricOn→All,ContractMetrics→True];
TheB2mActivate=MakeRule[{TheB2m[-i,-j,-k],Evaluate[TheB2mDefinition]},
  MetricOn→All,ContractMetrics→True];

TheActivate=Join[TheA0mActivate,TheA1pActivate,TheA1mActivate,
  TheA2pActivate,TheA2mActivate,TheB1pActivate,TheB2mActivate];
*)
```

Basic form $\chi^{\parallel}$ bJ^P, $\chi^{\parallel}$ AJ^P

In[◦]:=
```
DefTensor[ChiParaB0m[], M4, PrintAs → "χ"B0⁻""];
DeclareOrder[ChiParaB0m[], 1];
DefTensor[ChiParaB1p[-a, -b], M4, Antisymmetric[{-a, -b}], PrintAs → "χ"B1⁺""];
DeclareOrder[ChiParaB1p[-a, -b], 1];
DefTensor[ChiParaB1m[-a], M4, PrintAs → "χ"B1⁻""];
DeclareOrder[ChiParaB1m[-a], 1];
DefTensor[ChiParaB2m[-a, -b, -c], M4, Antisymmetric[{-a, -b}], PrintAs → "χ"B2⁻""];
DeclareOrder[ChiParaB2m[-a, -b, -c], 1];
DefTensor[ChiParaA0p[], M4, PrintAs → "χ"A0⁺""];
DeclareOrder[ChiParaA0p[], 1];
DefTensor[ChiParaA0m[], M4, PrintAs → "χ"A0⁻""];
DeclareOrder[ChiParaA0m[], 1];
DefTensor[ChiParaA1p[-a, -b], M4, Antisymmetric[{-a, -b}], PrintAs → "χ"A1⁺""];
DeclareOrder[ChiParaA1p[-a, -b], 1];
DefTensor[ChiParaA1m[-a], M4, PrintAs → "χ"A1⁻""];
DeclareOrder[ChiParaA1m[-a], 1];
DefTensor[ChiParaA2p[-a, -b], M4, Symmetric[{-a, -b}], PrintAs → "χ"A2⁺""];
DeclareOrder[ChiParaA2p[-a, -b], 1];
DefTensor[ChiParaA2m[-a, -b, -c], M4, Antisymmetric[{-a, -b}], PrintAs → "χ"A2⁻""];
```

```
DeclareOrder[ChiParaA2m[-a, -b, -c], 1];
AutomaticRules[ChiParaB2m,
  MakeRule[{ChiParaB2m[a, -b, -a], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[ChiParaB2m, MakeRule[{epsilonG[a, b, c, d] ChiParaB2m[-a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[ChiParaA2m, MakeRule[{ChiParaA2m[a, -b, -a], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[ChiParaA2m, MakeRule[{epsilonG[a, b, c, d] ChiParaA2m[-a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];

ChiParaB0mDefinition = TP0m[] /. PADMActivate // ToCanonical;
ChiParaB1pDefinition = TP1p[-i, -j] /. PADMActivate // ToCanonical;
ChiParaB1mDefinition = TP1m[-i] /. PADMActivate // ToCanonical;
ChiParaB2mDefinition = TP2m[-i, -j, -k] /. PADMActivate // ToCanonical;
ChiParaA0pDefinition = RP0p[] /. PADMActivate // ToCanonical;
ChiParaA0mDefinition = RP0m[] /. PADMActivate // ToCanonical;
ChiParaA1pDefinition = RP1p[-i, -j] /. PADMActivate // ToCanonical;
ChiParaA1mDefinition = RP1m[-i] /. PADMActivate // ToCanonical;
ChiParaA2pDefinition = RP2p[-i, -j] /. PADMActivate // ToCanonical;
ChiParaA2mDefinition = RP2m[-i, -j, -k] /. PADMActivate // ToCanonical;

ChiParaB0mActivate = MakeRule[{ChiParaB0m[], Evaluate[ChiParaB0mDefinition]},
   MetricOn → All, ContractMetrics → True];
ChiParaB1pActivate = MakeRule[{ChiParaB1p[-i, -j], Evaluate[ChiParaB1pDefinition]},
   MetricOn → All, ContractMetrics → True];
ChiParaB1mActivate = MakeRule[{ChiParaB1m[-i], Evaluate[ChiParaB1mDefinition]},
   MetricOn → All, ContractMetrics → True];
ChiParaB2mActivate = MakeRule[{ChiParaB2m[-i, -j, -k],
    Evaluate[ChiParaB2mDefinition]}, MetricOn → All, ContractMetrics → True];
ChiParaA0pActivate = MakeRule[{ChiParaA0p[], Evaluate[ChiParaA0pDefinition]},
   MetricOn → All, ContractMetrics → True];
ChiParaA0mActivate = MakeRule[{ChiParaA0m[], Evaluate[ChiParaA0mDefinition]},
   MetricOn → All, ContractMetrics → True];
ChiParaA1pActivate = MakeRule[{ChiParaA1p[-i, -j], Evaluate[ChiParaA1pDefinition]},
   MetricOn → All, ContractMetrics → True];
ChiParaA1mActivate = MakeRule[{ChiParaA1m[-i], Evaluate[ChiParaA1mDefinition]},
   MetricOn → All, ContractMetrics → True];
ChiParaA2pActivate = MakeRule[{ChiParaA2p[-i, -j], Evaluate[ChiParaA2pDefinition]},
   MetricOn → All, ContractMetrics → True];
ChiParaA2mActivate = MakeRule[{ChiParaA2m[-i, -j, -k],
    Evaluate[ChiParaA2mDefinition]}, MetricOn → All, ContractMetrics → True];

ChiParaActivate = Join[ChiParaB0mActivate, ChiParaB1pActivate, ChiParaB1mActivate,
   ChiParaB2mActivate, ChiParaA0pActivate, ChiParaA0mActivate, ChiParaA1pActivate,
```

Define $\mathbb{D}\hat{\pi}\,\mathrm{bJ}^P$, $\mathbb{D}\hat{\pi}\,\mathrm{AJ}^P$

```
In[*]:= DefTensor[DPiPB0p[-z], M4, PrintAs → "𝔻π̂b0⁺"];
(*DeclareOrder[DPiPB0p[-z],1];*)
DefTensor[DPiPB1p[-z, -a, -b], M4, Antisymmetric[{-a, -b}], PrintAs → "𝔻π̂b1⁺"];
(*DeclareOrder[DPiPB1p[-z,-a,-b],1];*)
DefTensor[DPiPB1m[-z, -a], M4, PrintAs → "𝔻π̂b1⁻"];
(*DeclareOrder[DPiPB1m[-z,-a],1];*)
DefTensor[DPiPB2p[-z, -a, -b], M4, Symmetric[{-a, -b}], PrintAs → "𝔻π̂b2⁺"];
(*DeclareOrder[DPiPB2p[-z,-a,-b],1];*)
AutomaticRules[DPiPB2p,
  MakeRule[{DPiPB2p[-z, a, -a], 0}, MetricOn → All, ContractMetrics → True]];
DefTensor[DPiPA0p[-z], M4, PrintAs → "𝔻π̂A0⁺"];
(*DeclareOrder[DPiPA0p[-z],1];*)
DefTensor[DPiPA0m[-z], M4, PrintAs → "𝔻π̂A0⁻"];
(*DeclareOrder[DPiPA0m[-z],1];*)
DefTensor[DPiPA1p[-z, -a, -b], M4, Antisymmetric[{-a, -b}], PrintAs → "𝔻π̂A1⁺"];
(*DeclareOrder[DPiPA1p[-z,-a,-b],1];*)
DefTensor[DPiPA1m[-z, -a], M4, PrintAs → "𝔻π̂A1⁻"];
(*DeclareOrder[DPiPA1m[-z,-a],1];*)
DefTensor[DPiPA2p[-z, -a, -b], M4, Symmetric[{-a, -b}], PrintAs → "𝔻π̂A2⁺"];
(*DeclareOrder[DPiPA2p[-z,-a,-b],1];*)
DefTensor[DPiPA2m[-z, -a, -b, -c],
  M4, Antisymmetric[{-a, -b}], PrintAs → "𝔻π̂A2⁻"];
(*DeclareOrder[DPiPA2m[-z,-a,-b,-c],1];*)
AutomaticRules[DPiPA2m,
  MakeRule[{DPiPA2m[-z, a, -b, -a], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[DPiPA2m, MakeRule[{epsilonG[a, b, c, d] DPiPA2m[-z, -a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[DPiPA2p, MakeRule[{DPiPA2p[-z, a, -a], 0},
   MetricOn → All, ContractMetrics → True]];
DPiPB0pActivate = MakeRule[{CD[-z][PiPB0p[]], DPiPB0p[-z]},
   MetricOn → All, ContractMetrics → True];
DPiPB1pActivate = MakeRule[{CD[-z][PiPB1p[-a, -b]],
    DPiPB1p[-z, -a, -b] + A[i, -a, -z] PiPB1p[-i, -b] + A[i, -b, -z] PiPB1p[-a, -i]},
   MetricOn → All, ContractMetrics → True];
DPiPB1mActivate = MakeRule[{CD[-z][PiPB1m[-a]], DPiPB1m[-z, -a] +
     A[i, -a, -z] PiPB1m[-i]}, MetricOn → All, ContractMetrics → True];
DPiPB2pActivate = MakeRule[{CD[-z][PiPB2p[-a, -b]],
    DPiPB2p[-z, -a, -b] + A[i, -a, -z] PiPB2p[-i, -b] + A[i, -b, -z] PiPB2p[-a, -i]},
   MetricOn → All, ContractMetrics → True];
DPiPA0pActivate = MakeRule[{CD[-z][PiPA0p[]], DPiPA0p[-z]},
   MetricOn → All, ContractMetrics → True];
```

```
DPiPA0mActivate = MakeRule[{CD[-z][PiPA0m[]], DPiPA0m[-z]},
   MetricOn → All, ContractMetrics → True];
DPiPA1pActivate = MakeRule[{CD[-z][PiPA1p[-a, -b]],
    DPiPA1p[-z, -a, -b] + A[i, -a, -z] PiPA1p[-i, -b] + A[i, -b, -z] PiPA1p[-a, -i]},
   MetricOn → All, ContractMetrics → True];
DPiPA1mActivate = MakeRule[{CD[-z][PiPA1m[-a]], DPiPA1m[-z, -a] +
     A[i, -a, -z] PiPA1m[-i]}, MetricOn → All, ContractMetrics → True];
DPiPA2pActivate = MakeRule[{CD[-z][PiPA2p[-a, -b]],
    DPiPA2p[-z, -a, -b] + A[i, -a, -z] PiPA2p[-i, -b] + A[i, -b, -z] PiPA2p[-a, -i]},
   MetricOn → All, ContractMetrics → True];
DPiPA2mActivate = MakeRule[{CD[-z][PiPA2m[-a, -b, -c]],
    DPiPA2m[-z, -a, -b, -c] + A[i, -a, -z] PiPA2m[-i, -b, -c] +
     A[i, -b, -z] PiPA2m[-a, -i, -c] + A[i, -c, -z] PiPA2m[-a, -b, -i]},
   MetricOn → All, ContractMetrics → True];
DPiPActivate = Join[DPiPB0pActivate, DPiPB1pActivate, DPiPB1mActivate,
   DPiPB2pActivate, DPiPA0pActivate, DPiPA0mActivate, DPiPA1pActivate,
   DPiPA1mActivate, DPiPA2pActivate, DPiPA2mActivate];

(*the rules below should of course be generalised beyond simply the momenta*)
DPiPB0pDeactivate = MakeRule[
   {DPiPB0p[-z], CD[-z][PiPB0p[]]}, MetricOn → All, ContractMetrics → True];
DPiPB1pDeactivate = MakeRule[{DPiPB1p[-z, -a, -b], CD[-z][PiPB1p[-a, -b]] -
     A[i, -a, -z] PiPB1p[-i, -b] - A[i, -b, -z] PiPB1p[-a, -i]},
   MetricOn → All, ContractMetrics → True];
DPiPB1mDeactivate = MakeRule[{DPiPB1m[-z, -a], CD[-z][PiPB1m[-a]] -
     A[i, -a, -z] PiPB1m[-i]}, MetricOn → All, ContractMetrics → True];
DPiPB2pDeactivate = MakeRule[{DPiPB2p[-z, -a, -b], CD[-z][PiPB2p[-a, -b]] -
     A[i, -a, -z] PiPB2p[-i, -b] - A[i, -b, -z] PiPB2p[-a, -i]},
   MetricOn → All, ContractMetrics → True];
DPiPA0pDeactivate = MakeRule[{DPiPA0p[-z], CD[-z][PiPA0p[]]},
   MetricOn → All, ContractMetrics → True];
DPiPA0mDeactivate = MakeRule[{DPiPA0m[-z], CD[-z][PiPA0m[]]},
   MetricOn → All, ContractMetrics → True];
DPiPA1pDeactivate = MakeRule[{DPiPA1p[-z, -a, -b], CD[-z][PiPA1p[-a, -b]] -
     A[i, -a, -z] PiPA1p[-i, -b] - A[i, -b, -z] PiPA1p[-a, -i]},
   MetricOn → All, ContractMetrics → True];
DPiPA1mDeactivate = MakeRule[{DPiPA1m[-z, -a], CD[-z][PiPA1m[-a]] -
     A[i, -a, -z] PiPA1m[-i]}, MetricOn → All, ContractMetrics → True];
DPiPA2pDeactivate = MakeRule[{DPiPA2p[-z, -a, -b], CD[-z][PiPA2p[-a, -b]] -
     A[i, -a, -z] PiPA2p[-i, -b] - A[i, -b, -z] PiPA2p[-a, -i]},
   MetricOn → All, ContractMetrics → True];
DPiPA2mDeactivate = MakeRule[{DPiPA2m[-z, -a, -b, -c],
    CD[-z][PiPA2m[-a, -b, -c]] - A[i, -a, -z] PiPA2m[-i, -b, -c] -
     A[i, -b, -z] PiPA2m[-a, -i, -c] - A[i, -c, -z] PiPA2m[-a, -b, -i]},
```

```
    MetricOn → All, ContractMetrics → True];
DPiPDeactivate = Join[DPiPB0pDeactivate, DPiPB1pDeactivate, DPiPB1mDeactivate,
    DPiPB2pDeactivate, DPiPA0pDeactivate, DPiPA0mDeactivate, DPiPA1pDeactivate,
    DPiPA1mDeactivate, DPiPA2pDeactivate, DPiPA2mDeactivate];
```

Define $\hat{\mathcal{D}}\hat{\pi}$ bJ^P, $\hat{\mathcal{D}}\hat{\pi}$ AJ^P

```
In[∘]:= DefTensor[DpPiPB0p[-z], M4, PrintAs → "𝔇̂π̂b0⁺", OrthogonalTo → {V[z]}];
DeclareOrder[DpPiPB0p[-z], 1];
DeclareOrder[DPiPB0p[-z], 1,
   "approximation" → B[w, -z] DpPiPB0p[-w] + V[-v] B[v, -z] V[u] H[-u, w] DPiPB0p[-w]];
DefTensor[DpPiPB1p[-z, -a, -b], M4, Antisymmetric[{-a, -b}],
   PrintAs → "𝔇̂π̂b1⁺", OrthogonalTo → {V[z], V[a], V[b]}];
DeclareOrder[DpPiPB1p[-z, -a, -b], 1];
DeclareOrder[DPiPB1p[-z, -a, -b], 1, "approximation" →
    B[w, -z] DpPiPB1p[-w, -a, -b] + V[-v] B[v, -z] V[u] H[-u, w] DPiPB1p[-w, -a, -b]];
DefTensor[DpPiPB1m[-z, -a], M4, PrintAs → "𝔇̂π̂b1⁻", OrthogonalTo → {V[z], V[a]}];
DeclareOrder[DpPiPB1m[-z, -a], 1];
DeclareOrder[DPiPB1m[-z, -a], 1, "approximation" →
    B[w, -z] DpPiPB1m[-w, -a] + V[-v] B[v, -z] V[u] H[-u, w] DPiPB1m[-w, -a]];
DefTensor[DpPiPB2p[-z, -a, -b], M4, Symmetric[{-a, -b}],
   PrintAs → "𝔇̂π̂b2⁺", OrthogonalTo → {V[z], V[a], V[b]}];
DeclareOrder[DpPiPB2p[-z, -a, -b], 1];
DeclareOrder[DPiPB2p[-z, -a, -b], 1, "approximation" →
    B[w, -z] DpPiPB2p[-w, -a, -b] + V[-v] B[v, -z] V[u] H[-u, w] DPiPB2p[-w, -a, -b]];
AutomaticRules[DpPiPB2p, MakeRule[{DpPiPB2p[-z, a, -a], 0},
    MetricOn → All, ContractMetrics → True]];
DefTensor[DpPiPA0p[-z], M4, PrintAs → "𝔇̂π̂A0⁺", OrthogonalTo → {V[z]}];
DeclareOrder[DpPiPA0p[-z], 1];
DeclareOrder[DPiPA0p[-z], 1,
   "approximation" → B[w, -z] DpPiPA0p[-w] + V[-v] B[v, -z] V[u] H[-u, w] DPiPA0p[-w]];
DefTensor[DpPiPA0m[-z], M4, PrintAs → "𝔇̂π̂A0⁻", OrthogonalTo → {V[z]}];
DeclareOrder[DpPiPA0m[-z], 1];
DeclareOrder[DPiPA0m[-z], 1,
   "approximation" → B[w, -z] DpPiPA0m[-w] + V[-v] B[v, -z] V[u] H[-u, w] DPiPA0m[-w]];
DefTensor[DpPiPA1p[-z, -a, -b], M4, Antisymmetric[{-a, -b}],
   PrintAs → "𝔇̂π̂A1⁺", OrthogonalTo → {V[z], V[a], V[b]}];
DeclareOrder[DpPiPA1p[-z, -a, -b], 1];
DeclareOrder[DPiPA1p[-z, -a, -b], 1, "approximation" →
    B[w, -z] DpPiPA1p[-w, -a, -b] + V[-v] B[v, -z] V[u] H[-u, w] DPiPA1p[-w, -a, -b]];
DefTensor[DpPiPA1m[-z, -a], M4, PrintAs → "𝔇̂π̂A1⁻", OrthogonalTo → {V[z], V[a]}];
DeclareOrder[DpPiPA1m[-z, -a], 1];
DeclareOrder[DPiPA1m[-z, -a], 1, "approximation" →
```

```
    B[w, -z] DpPiPA1m[-w, -a] + V[-v] B[v, -z] V[u] H[-u, w] DPiPA1m[-w, -a]];
DefTensor[DpPiPA2p[-z, -a, -b], M4, Symmetric[{-a, -b}],
   PrintAs → "𝔇̂π̂A2⁺", OrthogonalTo → {V[z], V[a], V[b]}];
DeclareOrder[DpPiPA2p[-z, -a, -b], 1];
DeclareOrder[DPiPA2p[-z, -a, -b], 1, "approximation" →
    B[w, -z] DpPiPA2p[-w, -a, -b] + V[-v] B[v, -z] V[u] H[-u, w] DPiPA2p[-w, -a, -b]];
DefTensor[DpPiPA2m[-z, -a, -b, -c], M4, Antisymmetric[{-a, -b}],
   PrintAs → "𝔇̂π̂A2⁻", OrthogonalTo → {V[z], V[a], V[b], V[c]}];
DeclareOrder[DpPiPA2m[-z, -a, -b, -c], 1];
DeclareOrder[DPiPA2m[-z, -a, -b, -c],
   1, "approximation" → B[w, -z] DpPiPA2m[-w, -a, -b, -c] +
     V[-v] B[v, -z] V[u] H[-u, w] DPiPA2m[-w, -a, -b, -c]];
AutomaticRules[DpPiPA2m, MakeRule[{DpPiPA2m[-z, a, -b, -a], 0},
    MetricOn → All, ContractMetrics → True]];
AutomaticRules[DpPiPA2m, MakeRule[{epsilonG[a, b, c, d] DpPiPA2m[-z, -a, -b, -c], 0},
    MetricOn → All, ContractMetrics → True]];
AutomaticRules[DpPiPA2p, MakeRule[{DpPiPA2p[-z, a, -a], 0},
    MetricOn → All, ContractMetrics → True]];
DpPiPB0pActivate = MakeRule[{G3[-y, z] DPiPB0p[-z], G3[-y, z] B[x, -z] DpPiPB0p[-x]},
    MetricOn → All, ContractMetrics → True];
DpPiPB1pActivate = MakeRule[{G3[-y, z] DPiPB1p[-z, -a, -b],
     Evaluate[G3[-y, z] B[x, -z] DpPiPB1p[-x, -a, -b] +
         (G[-a, i] G[-b, j] - PPara[-a, i] PPara[-b, j]) G3[-y, z] DPiPB1p[-z, -i, -j] /.
       PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpPiPB1mActivate = MakeRule[{G3[-y, z] DPiPB1m[-z, -a],
     Evaluate[G3[-y, z] B[x, -z] DpPiPB1m[-x, -a] + (G[-a, i] - PPara[-a, i]) G3[-y, z]
         DPiPB1m[-z, -i] /. PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpPiPB2pActivate = MakeRule[{G3[-y, z] DPiPB2p[-z, -a, -b],
     Evaluate[G3[-y, z] B[x, -z] DpPiPB2p[-x, -a, -b] +
         (G[-a, i] G[-b, j] - PPara[-a, i] PPara[-b, j]) G3[-y, z] DPiPB2p[-z, -i, -j] /.
       PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpPiPA0pActivate = MakeRule[{G3[-y, z] DPiPA0p[-z], G3[-y, z] B[x, -z] DpPiPA0p[-x]},
    MetricOn → All, ContractMetrics → True];
DpPiPA0mActivate = MakeRule[{G3[-y, z] DPiPA0m[-z], G3[-y, z] B[x, -z] DpPiPA0m[-x]},
    MetricOn → All, ContractMetrics → True];
DpPiPA1pActivate = MakeRule[{G3[-y, z] DPiPA1p[-z, -a, -b],
     Evaluate[G3[-y, z] B[x, -z] DpPiPA1p[-x, -a, -b] +
         (G[-a, i] G[-b, j] - PPara[-a, i] PPara[-b, j]) G3[-y, z] DPiPA1p[-z, -i, -j] /.
       PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpPiPA1mActivate = MakeRule[{G3[-y, z] DPiPA1m[-z, -a],
     Evaluate[G3[-y, z] B[x, -z] DpPiPA1m[-x, -a] + (G[-a, i] - PPara[-a, i]) G3[-y, z]
         DPiPA1m[-z, -i] /. PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpPiPA2pActivate = MakeRule[{G3[-y, z] DPiPA2p[-z, -a, -b],
```

```
     Evaluate[G3[-y, z] B[x, -z] DpPiPA2p[-x, -a, -b] +
         (G[-a, i] G[-b, j] - PPara[-a, i] PPara[-b, j]) G3[-y, z] DPiPA2p[-z, -i, -j] /.
        PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpPiPA2mActivate = MakeRule[{G3[-y, z] DPiPA2m[-z, -a, -b, -c],
     Evaluate[G3[-y, z] B[x, -z] DpPiPA2m[-x, -a, -b, -c] +
         (G[-a, i] G[-b, j] G[-c, k] - PPara[-a, i] PPara[-b, j] PPara[-c, k])
          G3[-y, z] DPiPA2m[-z, -i, -j, -k] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True];
DpPiPActivate = Join[DpPiPB0pActivate, DpPiPB1pActivate, DpPiPB1mActivate,
   DpPiPB2pActivate, DpPiPA0pActivate, DpPiPA0mActivate, DpPiPA1pActivate,
   DpPiPA1mActivate, DpPiPA2pActivate, DpPiPA2mActivate];

(*again this should be extended over other derivatives,
multiply the above by PPara[-w,v]H[-v,y]*)
DpPiPB0pDeactivate =
  MakeRule[{DpPiPB0p[-w], PPara[-w, v] H[-v, y] G3[-y, z] DPiPB0p[-z]},
   MetricOn → All, ContractMetrics → True];
DpPiPB1pDeactivate = MakeRule[{DpPiPB1p[-w, -a, -b],
     Evaluate[PPara[-w, v] H[-v, y] G3[-y, z] DPiPB1p[-z, -a, -b] -
         PPara[-w, v] H[-v, y] (G[-a, i] G[-b, j] - PPara[-a, i] PPara[-b, j])
          G3[-y, z] DPiPB1p[-z, -i, -j] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True];
DpPiPB1mDeactivate = MakeRule[{DpPiPB1m[-w, -a],
     Evaluate[PPara[-w, v] H[-v, y] G3[-y, z] DPiPB1m[-z, -a] -
         PPara[-w, v] H[-v, y] (G[-a, i] - PPara[-a, i]) G3[-y, z] DPiPB1m[-z, -i] /.
        PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpPiPB2pDeactivate = MakeRule[{DpPiPB2p[-w, -a, -b],
     Evaluate[PPara[-w, v] H[-v, y] G3[-y, z] DPiPB2p[-z, -a, -b] -
         PPara[-w, v] H[-v, y] (G[-a, i] G[-b, j] - PPara[-a, i] PPara[-b, j])
          G3[-y, z] DPiPB2p[-z, -i, -j] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True];
DpPiPA0pDeactivate = MakeRule[{DpPiPA0p[-w], PPara[-w, v] H[-v, y]
      G3[-y, z] DPiPA0p[-z]}, MetricOn → All, ContractMetrics → True];
DpPiPA0mDeactivate = MakeRule[{DpPiPA0m[-w], PPara[-w, v] H[-v, y]
      G3[-y, z] DPiPA0m[-z]}, MetricOn → All, ContractMetrics → True];
DpPiPA1pDeactivate = MakeRule[{DpPiPA1p[-w, -a, -b],
     Evaluate[PPara[-w, v] H[-v, y] G3[-y, z] DPiPA1p[-z, -a, -b] -
         PPara[-w, v] H[-v, y] (G[-a, i] G[-b, j] - PPara[-a, i] PPara[-b, j])
          G3[-y, z] DPiPA1p[-z, -i, -j] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True];
DpPiPA1mDeactivate = MakeRule[{DpPiPA1m[-w, -a],
     Evaluate[PPara[-w, v] H[-v, y] G3[-y, z] DPiPA1m[-z, -a] -
```

```
        PPara[-w, v] H[-v, y] (G[-a, i] - PPara[-a, i]) G3[-y, z] DPiPA1m[-z, -i] /.
       PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpPiPA2pDeactivate = MakeRule[{DpPiPA2p[-w, -a, -b],
     Evaluate[PPara[-w, v] H[-v, y] G3[-y, z] DPiPA2p[-z, -a, -b] -
        PPara[-w, v] H[-v, y] (G[-a, i] G[-b, j] - PPara[-a, i] PPara[-b, j])
         G3[-y, z] DPiPA2p[-z, -i, -j] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True];
DpPiPA2mDeactivate = MakeRule[{DpPiPA2m[-w, -a, -b, -c],
     Evaluate[PPara[-w, v] H[-v, y] G3[-y, z] DPiPA2m[-z, -a, -b, -c] - PPara[-w, v]
         H[-v, y] (G[-a, i] G[-b, j] G[-c, k] - PPara[-a, i] PPara[-b, j] PPara[-c, k])
         G3[-y, z] DPiPA2m[-z, -i, -j, -k] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True];
DpPiPDeactivate = Join[DpPiPB0pDeactivate, DpPiPB1pDeactivate, DpPiPB1mDeactivate,
   DpPiPB2pDeactivate, DpPiPA0pDeactivate, DpPiPA0mDeactivate, DpPiPA1pDeactivate,
   DpPiPA1mDeactivate, DpPiPA2pDeactivate, DpPiPA2mDeactivate];
```

Define $\mathbb{D}\hat{T}\,J^P$, $\mathbb{D}\hat{R}\,J^P$

```
In[•]:= DefTensor[DTP0m[-z], M4, PrintAs → "𝕯T̂0⁻"];
(*DeclareOrder[DTP0m[-z],1];*)
DefTensor[DTP1p[-z, -a, -b], M4, Antisymmetric[{-a, -b}], PrintAs → "𝕯T̂1⁺"];
(*DeclareOrder[DTP1p[-z,-a,-b],1];*)
DefTensor[DTP1m[-z, -a], M4, PrintAs → "𝕯T̂1⁻"];
(*DeclareOrder[DTP1m[-z,-a],1];*)
DefTensor[DTP2m[-z, -a, -b, -c], M4, Antisymmetric[{-a, -b}], PrintAs → "𝕯T̂2⁻"];
(*DeclareOrder[DTP2m[-z,-a,-b,-c],1];*)
AutomaticRules[DTP2m,
  MakeRule[{DTP2m[a, -b, -a], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[DTP2m, MakeRule[{epsilonG[a, b, c, d] DTP2m[-a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];
DefTensor[DRP0p[-z], M4, PrintAs → "𝕯R̂0⁺"];
(*DeclareOrder[DRP0p[-z],1];*)
DefTensor[DRP0m[-z], M4, PrintAs → "𝕯R̂0⁻"];
(*DeclareOrder[DRP0m[-z],1];*)
DefTensor[DRP1p[-z, -a, -b], M4, Antisymmetric[{-a, -b}], PrintAs → "𝕯R̂1⁺"];
(*DeclareOrder[DRP1p[-z,-a,-b],1];*)
DefTensor[DRP1m[-z, -a], M4, PrintAs → "𝕯R̂1⁻"];
(*DeclareOrder[DRP1m[-z,-a],1];*)
DefTensor[DRP2p[-z, -a, -b], M4, Symmetric[{-a, -b}], PrintAs → "𝕯R̂2⁺"];
(*DeclareOrder[DRP2p[-z,-a,-b],1];*)
DefTensor[DRP2m[-z, -a, -b, -c], M4, Antisymmetric[{-a, -b}], PrintAs → "𝕯R̂2⁻"];
(*DeclareOrder[DRP2m[-z,-a,-b,-c],1];*)
AutomaticRules[DRP2m,
```

```
  MakeRule[{DRP2m[-z, a, -b, -a], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[DRP2m, MakeRule[{epsilonG[a, b, c, d] DRP2m[-z, -a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[DRP2p, MakeRule[{DRP2p[-z, a, -a], 0},
   MetricOn → All, ContractMetrics → True]];
DTP0mActivate = MakeRule[{CD[-z][TP0m[]], DTP0m[-z]},
   MetricOn → All, ContractMetrics → True];
DTP1pActivate = MakeRule[{CD[-z][TP1p[-a, -b]],
    DTP1p[-z, -a, -b] + A[i, -a, -z] TP1p[-i, -b] + A[i, -b, -z] TP1p[-a, -i]},
   MetricOn → All, ContractMetrics → True];
DTP1mActivate = MakeRule[{CD[-z][TP1m[-a]], DTP1m[-z, -a] + A[i, -a, -z] TP1m[-i]},
   MetricOn → All, ContractMetrics → True];
DTP2mActivate = MakeRule[{CD[-z][TP2m[-a, -b, -c]], DTP2m[-z, -a, -b, -c] +
     A[i, -a, -z] TP2m[-i, -b, -c] + A[i, -b, -z] TP2m[-a, -i, -c] +
     A[i, -c, -z] TP2m[-a, -b, -i]}, MetricOn → All, ContractMetrics → True];
DRP0pActivate = MakeRule[{CD[-z][RP0p[]], DRP0p[-z]},
   MetricOn → All, ContractMetrics → True];
DRP0mActivate = MakeRule[{CD[-z][RP0m[]], DRP0m[-z]},
   MetricOn → All, ContractMetrics → True];
DRP1pActivate = MakeRule[{CD[-z][RP1p[-a, -b]],
    DRP1p[-z, -a, -b] + A[i, -a, -z] RP1p[-i, -b] + A[i, -b, -z] RP1p[-a, -i]},
   MetricOn → All, ContractMetrics → True];
DRP1mActivate = MakeRule[{CD[-z][RP1m[-a]], DRP1m[-z, -a] + A[i, -a, -z] RP1m[-i]},
   MetricOn → All, ContractMetrics → True];
DRP2pActivate = MakeRule[{CD[-z][RP2p[-a, -b]],
    DRP2p[-z, -a, -b] + A[i, -a, -z] RP2p[-i, -b] + A[i, -b, -z] RP2p[-a, -i]},
   MetricOn → All, ContractMetrics → True];
DRP2mActivate = MakeRule[{CD[-z][RP2m[-a, -b, -c]], DRP2m[-z, -a, -b, -c] +
     A[i, -a, -z] RP2m[-i, -b, -c] + A[i, -b, -z] RP2m[-a, -i, -c] +
     A[i, -c, -z] RP2m[-a, -b, -i]}, MetricOn → All, ContractMetrics → True];
DRPActivate = Join[DTP0mActivate, DTP1pActivate, DTP1mActivate,
   DTP2mActivate, DRP0pActivate, DRP0mActivate, DRP1pActivate,
   DRP1mActivate, DRP2pActivate, DRP2mActivate];

(*the rules below should of course be generalised beyond simply
 the momenta -- these below now generalise to the field strengths*)
DTP0mDeactivate = MakeRule[{DTP0m[-z], CD[-z][TP0m[]]},
   MetricOn → All, ContractMetrics → True];
DTP1pDeactivate = MakeRule[{DTP1p[-z, -a, -b],
    CD[-z][TP1p[-a, -b]] - A[i, -a, -z] TP1p[-i, -b] - A[i, -b, -z] TP1p[-a, -i]},
   MetricOn → All, ContractMetrics → True];
DTP1mDeactivate = MakeRule[{DTP1m[-z, -a], CD[-z][TP1m[-a]] - A[i, -a, -z] TP1m[-i]},
   MetricOn → All, ContractMetrics → True];
DTP2mDeactivate = MakeRule[{DTP2m[-z, -a, -b, -c], CD[-z][TP2m[-a, -b, -c]] -
```

```
      A[i, -a, -z] TP2m[-i, -b, -c] - A[i, -b, -z] TP2m[-a, -i, -c] -
      A[i, -c, -z] TP2m[-a, -b, -i]}, MetricOn → All, ContractMetrics → True];
DRP0pDeactivate = MakeRule[{DRP0p[-z], CD[-z][RP0p[]]},
   MetricOn → All, ContractMetrics → True];
DRP0mDeactivate = MakeRule[{DRP0m[-z], CD[-z][RP0m[]]},
   MetricOn → All, ContractMetrics → True];
DRP1pDeactivate = MakeRule[{DRP1p[-z, -a, -b],
    CD[-z][RP1p[-a, -b]] - A[i, -a, -z] RP1p[-i, -b] - A[i, -b, -z] RP1p[-a, -i]},
   MetricOn → All, ContractMetrics → True];
DRP1mDeactivate = MakeRule[{DRP1m[-z, -a], CD[-z][RP1m[-a]] - A[i, -a, -z] RP1m[-i]},
   MetricOn → All, ContractMetrics → True];
DRP2pDeactivate = MakeRule[{DRP2p[-z, -a, -b],
    CD[-z][RP2p[-a, -b]] - A[i, -a, -z] RP2p[-i, -b] - A[i, -b, -z] RP2p[-a, -i]},
   MetricOn → All, ContractMetrics → True];
DRP2mDeactivate = MakeRule[{DRP2m[-z, -a, -b, -c], CD[-z][RP2m[-a, -b, -c]] -
      A[i, -a, -z] RP2m[-i, -b, -c] - A[i, -b, -z] RP2m[-a, -i, -c] -
      A[i, -c, -z] RP2m[-a, -b, -i]}, MetricOn → All, ContractMetrics → True];
DRPDeactivate = Join[DTP0mDeactivate, DTP1pDeactivate, DTP1mDeactivate,
   DTP2mDeactivate, DRP0pDeactivate, DRP0mDeactivate, DRP1pDeactivate,
   DRP1mDeactivate, DRP2pDeactivate, DRP2mDeactivate];
```

Define $\mathbb{D}\hat{\lambda} J^P$

```
In[◦]:= DefTensor[DTLambdaP0m[-z], M4, PrintAs → "𝔻T̂λ0⁻"];
(*DeclareOrder[DTLambdaP0m[-z],1];*)
DefTensor[DTLambdaP1p[-z, -a, -b],
  M4, Antisymmetric[{-a, -b}], PrintAs → "𝔻T̂λ1⁺"];
(*DeclareOrder[DTLambdaP1p[-z,-a,-b],1];*)
DefTensor[DTLambdaP1m[-z, -a], M4, PrintAs → "𝔻T̂λ1⁻"];
(*DeclareOrder[DTLambdaP1m[-z,-a],1];*)
DefTensor[DTLambdaP2m[-z, -a, -b, -c],
  M4, Antisymmetric[{-a, -b}], PrintAs → "𝔻T̂λ2⁻"];
(*DeclareOrder[DTLambdaP2m[-z,-a,-b,-c],1];*)
AutomaticRules[DTLambdaP2m,
  MakeRule[{DTLambdaP2m[a, -b, -a], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[DTLambdaP2m, MakeRule[{epsilonG[a, b, c, d] DTLambdaP2m[-a, -b, -c],
    0}, MetricOn → All, ContractMetrics → True]];
DefTensor[DRLambdaP0p[-z], M4, PrintAs → "𝔻R̂λ0⁺"];
(*DeclareOrder[DRLambdaP0p[-z],1];*)
DefTensor[DRLambdaP0m[-z], M4, PrintAs → "𝔻R̂λ0⁻"];
(*DeclareOrder[DRLambdaP0m[-z],1];*)
DefTensor[DRLambdaP1p[-z, -a, -b],
  M4, Antisymmetric[{-a, -b}], PrintAs → "𝔻R̂λ1⁺"];
```

```
(*DeclareOrder[DRLambdaP1p[-z,-a,-b],1];*)
DefTensor[DRLambdaP1m[-z, -a], M4, PrintAs → "𝔻R̂λ1⁻"];
(*DeclareOrder[DRLambdaP1m[-z,-a],1];*)
DefTensor[DRLambdaP2p[-z, -a, -b], M4, Symmetric[{-a, -b}], PrintAs → "𝔻R̂λ2⁺"];
(*DeclareOrder[DRLambdaP2p[-z,-a,-b],1];*)
DefTensor[DRLambdaP2m[-z, -a, -b, -c],
  M4, Antisymmetric[{-a, -b}], PrintAs → "𝔻R̂λ2⁻"];
(*DeclareOrder[DRLambdaP2m[-z,-a,-b,-c],1];*)
AutomaticRules[DRLambdaP2m,
  MakeRule[{DRLambdaP2m[-z, a, -b, -a], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[DRLambdaP2m,
  MakeRule[{epsilonG[a, b, c, d] DRLambdaP2m[-z, -a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[DRLambdaP2p, MakeRule[{DRLambdaP2p[-z, a, -a], 0},
   MetricOn → All, ContractMetrics → True]];
DTLambdaP0mActivate = MakeRule[{CD[-z][TLambdaP0m[]], DTLambdaP0m[-z]},
   MetricOn → All, ContractMetrics → True];
DTLambdaP1pActivate = MakeRule[{CD[-z][TLambdaP1p[-a, -b]],
    DTLambdaP1p[-z, -a, -b] + A[i, -a, -z] TLambdaP1p[-i, -b] +
     A[i, -b, -z] TLambdaP1p[-a, -i]}, MetricOn → All, ContractMetrics → True];
DTLambdaP1mActivate = MakeRule[{CD[-z][TLambdaP1m[-a]], DTLambdaP1m[-z, -a] +
     A[i, -a, -z] TLambdaP1m[-i]}, MetricOn → All, ContractMetrics → True];
DTLambdaP2mActivate = MakeRule[{CD[-z][TLambdaP2m[-a, -b, -c]],
    DTLambdaP2m[-z, -a, -b, -c] + A[i, -a, -z] TLambdaP2m[-i, -b, -c] +
     A[i, -b, -z] TLambdaP2m[-a, -i, -c] + A[i, -c, -z] TLambdaP2m[-a, -b, -i]},
   MetricOn → All, ContractMetrics → True];
DRLambdaP0pActivate = MakeRule[{CD[-z][RLambdaP0p[]], DRLambdaP0p[-z]},
   MetricOn → All, ContractMetrics → True];
DRLambdaP0mActivate = MakeRule[{CD[-z][RLambdaP0m[]], DRLambdaP0m[-z]},
   MetricOn → All, ContractMetrics → True];
DRLambdaP1pActivate = MakeRule[{CD[-z][RLambdaP1p[-a, -b]],
    DRLambdaP1p[-z, -a, -b] + A[i, -a, -z] RLambdaP1p[-i, -b] +
     A[i, -b, -z] RLambdaP1p[-a, -i]}, MetricOn → All, ContractMetrics → True];
DRLambdaP1mActivate = MakeRule[{CD[-z][RLambdaP1m[-a]], DRLambdaP1m[-z, -a] +
     A[i, -a, -z] RLambdaP1m[-i]}, MetricOn → All, ContractMetrics → True];
DRLambdaP2pActivate = MakeRule[{CD[-z][RLambdaP2p[-a, -b]],
    DRLambdaP2p[-z, -a, -b] + A[i, -a, -z] RLambdaP2p[-i, -b] +
     A[i, -b, -z] RLambdaP2p[-a, -i]}, MetricOn → All, ContractMetrics → True];
DRLambdaP2mActivate = MakeRule[{CD[-z][RLambdaP2m[-a, -b, -c]],
    DRLambdaP2m[-z, -a, -b, -c] + A[i, -a, -z] RLambdaP2m[-i, -b, -c] +
     A[i, -b, -z] RLambdaP2m[-a, -i, -c] + A[i, -c, -z] RLambdaP2m[-a, -b, -i]},
   MetricOn → All, ContractMetrics → True];
DRLambdaPActivate = Join[DTLambdaP0mActivate, DTLambdaP1pActivate,
```

```
DTLambdaP1mActivate, DTLambdaP2mActivate, DRLambdaP0pActivate,
DRLambdaP0mActivate, DRLambdaP1pActivate, DRLambdaP1mActivate,
DRLambdaP2pActivate, DRLambdaP2mActivate];
```

Define $\mathbb{D}\overset{*}{\lambda}\,J^P$

```
In[◦]:= DefTensor[DTLambdaPerp0p[-z], M4, PrintAs → "𝕕T*λ0⁺"];
(*DeclareOrder[DTLambdaPerp0p[-z],1];*)
DefTensor[DTLambdaPerp1p[-z, -a, -b],
  M4, Antisymmetric[{-a, -b}], PrintAs → "𝕕T*λ1⁺"];
(*DeclareOrder[DTLambdaPerp1p[-z,-a,-b],1];*)
DefTensor[DTLambdaPerp1m[-z, -a], M4, PrintAs → "𝕕T*λ1⁻"];
(*DeclareOrder[DTLambdaPerp1m[-z,-a],1];*)
DefTensor[DTLambdaPerp2p[-z, -a, -b],
  M4, Symmetric[{-a, -b}], PrintAs → "𝕕T*λ2⁺"];
(*DeclareOrder[DTLambdaPerp2p[-z,-a,-b],1];*)
AutomaticRules[DTLambdaPerp2p,
  MakeRule[{DTLambdaPerp2p[-z, a, -a], 0}, MetricOn → All, ContractMetrics → True]];
DefTensor[DRLambdaPerp0p[-z], M4, PrintAs → "𝕕R*λ0⁺"];
(*DeclareOrder[DRLambdaPerp0p[-z],1];*)
DefTensor[DRLambdaPerp0m[-z], M4, PrintAs → "𝕕R*λ0⁻"];
(*DeclareOrder[DRLambdaPerp0m[-z],1];*)
DefTensor[DRLambdaPerp1p[-z, -a, -b],
  M4, Antisymmetric[{-a, -b}], PrintAs → "𝕕R*λ1⁺"];
(*DeclareOrder[DRLambdaPerp1p[-z,-a,-b],1];*)
DefTensor[DRLambdaPerp1m[-z, -a], M4, PrintAs → "𝕕R*λ1⁻"];
(*DeclareOrder[DRLambdaPerp1m[-z,-a],1];*)
DefTensor[DRLambdaPerp2p[-z, -a, -b],
  M4, Symmetric[{-a, -b}], PrintAs → "𝕕R*λ2⁺"];
(*DeclareOrder[DRLambdaPerp2p[-z,-a,-b],1];*)
DefTensor[DRLambdaPerp2m[-z, -a, -b, -c],
  M4, Antisymmetric[{-a, -b}], PrintAs → "𝕕R*λ2⁻"];
(*DeclareOrder[DRLambdaPerp2m[-z,-a,-b,-c],1];*)
AutomaticRules[DRLambdaPerp2m, MakeRule[
   {DRLambdaPerp2m[-z, a, -b, -a], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[DRLambdaPerp2m, MakeRule[
   {epsilonG[a, b, c, d] DRLambdaPerp2m[-z, -a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[DRLambdaPerp2p, MakeRule[{DRLambdaPerp2p[-z, a, -a], 0},
   MetricOn → All, ContractMetrics → True]];
DTLambdaPerp0pActivate = MakeRule[{CD[-z][TLambdaPerp0p[]], DTLambdaPerp0p[-z]},
   MetricOn → All, ContractMetrics → True];
```

```
DTLambdaPerp1pActivate = MakeRule[{CD[-z][TLambdaPerp1p[-a, -b]],
    DTLambdaPerp1p[-z, -a, -b] + A[i, -a, -z] TLambdaPerp1p[-i, -b] +
     A[i, -b, -z] TLambdaPerp1p[-a, -i]}, MetricOn → All, ContractMetrics → True];
DTLambdaPerp1mActivate = MakeRule[{CD[-z][TLambdaPerp1m[-a]],
    DTLambdaPerp1m[-z, -a] + A[i, -a, -z] TLambdaPerp1m[-i]},
   MetricOn → All, ContractMetrics → True];
DTLambdaPerp2pActivate = MakeRule[{CD[-z][TLambdaPerp2p[-a, -b]],
    DTLambdaPerp2p[-z, -a, -b] + A[i, -a, -z] TLambdaPerp2p[-i, -b] +
     A[i, -b, -z] TLambdaPerp2p[-a, -i]}, MetricOn → All, ContractMetrics → True];
DRLambdaPerp0pActivate = MakeRule[{CD[-z][RLambdaPerp0p[]], DRLambdaPerp0p[-z]},
   MetricOn → All, ContractMetrics → True];
DRLambdaPerp0mActivate = MakeRule[{CD[-z][RLambdaPerp0m[]], DRLambdaPerp0m[-z]},
   MetricOn → All, ContractMetrics → True];
DRLambdaPerp1pActivate = MakeRule[{CD[-z][RLambdaPerp1p[-a, -b]],
    DRLambdaPerp1p[-z, -a, -b] + A[i, -a, -z] RLambdaPerp1p[-i, -b] +
     A[i, -b, -z] RLambdaPerp1p[-a, -i]}, MetricOn → All, ContractMetrics → True];
DRLambdaPerp1mActivate = MakeRule[{CD[-z][RLambdaPerp1m[-a]],
    DRLambdaPerp1m[-z, -a] + A[i, -a, -z] RLambdaPerp1m[-i]},
   MetricOn → All, ContractMetrics → True];
DRLambdaPerp2pActivate = MakeRule[{CD[-z][RLambdaPerp2p[-a, -b]],
    DRLambdaPerp2p[-z, -a, -b] + A[i, -a, -z] RLambdaPerp2p[-i, -b] +
     A[i, -b, -z] RLambdaPerp2p[-a, -i]}, MetricOn → All, ContractMetrics → True];
DRLambdaPerp2mActivate = MakeRule[{CD[-z][RLambdaPerp2m[-a, -b, -c]],
    DRLambdaPerp2m[-z, -a, -b, -c] + A[i, -a, -z] RLambdaPerp2m[-i, -b, -c] +
     A[i, -b, -z] RLambdaPerp2m[-a, -i, -c] + A[i, -c, -z] RLambdaPerp2m[-a, -b, -i]},
   MetricOn → All, ContractMetrics → True];
DRLambdaPerpActivate = Join[DTLambdaPerp0pActivate, DTLambdaPerp1pActivate,
   DTLambdaPerp1mActivate, DTLambdaPerp2pActivate, DRLambdaPerp0pActivate,
   DRLambdaPerp0mActivate, DRLambdaPerp1pActivate, DRLambdaPerp1mActivate,
   DRLambdaPerp2pActivate, DRLambdaPerp2mActivate];
```

Define $\mathbb{D}H$

```
In[∘]:= DefTensor[DHComp[-z], M4, PrintAs → "𝔻H"];
        DHCompActivate =
          MakeRule[{CD[-z][HComp[]], DHComp[-z]}, MetricOn → All, ContractMetrics → True];
```

Define $\hat{\mathbb{D}}\hat{T}J^P, \hat{\mathbb{D}}\hat{R}J^P$

```
In[∘]:= DefTensor[DpTP0m[-z], M4, PrintAs → "𝔻̂T̂0⁻", OrthogonalTo → {V[z]}];
        DeclareOrder[DpTP0m[-z], 1];
        DeclareOrder[DTP0m[-z], 1,
          "approximation" → B[w, -z] DpTP0m[-w] + V[-v] B[v, -z] V[u] H[-u, w] DTP0m[-w]];
        DefTensor[DpTP1p[-z, -a, -b], M4, Antisymmetric[{-a, -b}],
```

```
  PrintAs → "𝔇T̂1⁺", OrthogonalTo → {V[z], V[a], V[b]}];
DeclareOrder[DpTP1p[-z, -a, -b], 1];
DeclareOrder[DTP1p[-z, -a, -b], 1, "approximation" →
   B[w, -z] DpTP1p[-w, -a, -b] + V[-v] B[v, -z] V[u] H[-u, w] DTP1p[-w, -a, -b]];
DefTensor[DpTP1m[-z, -a], M4, PrintAs → "𝔇T̂1⁻", OrthogonalTo → {V[z], V[a]}];
DeclareOrder[DpTP1m[-z, -a], 1];
DeclareOrder[DTP1m[-z, -a], 1, "approximation" →
   B[w, -z] DpTP1m[-w, -a] + V[-v] B[v, -z] V[u] H[-u, w] DTP1m[-w, -a]];
DefTensor[DpTP2m[-z, -a, -b, -c], M4, Antisymmetric[{-a, -b}],
  PrintAs → "𝔇T̂2⁻", OrthogonalTo → {V[z], V[a], V[b], V[c]}];
DeclareOrder[DpTP2m[-z, -a, -b, -c], 1];
DeclareOrder[DTP2m[-z, -a, -b, -c],
  1, "approximation" → B[w, -z] DpTP2m[-w, -a, -b, -c] +
    V[-v] B[v, -z] V[u] H[-u, w] DTP2m[-w, -a, -b, -c]];
AutomaticRules[DpTP2m, MakeRule[{DpTP2m[-z, a, -b, -a], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[DpTP2m, MakeRule[{epsilonG[a, b, c, d] DpTP2m[-z, -a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];
DefTensor[DpRP0p[-z], M4, PrintAs → "𝔇R̂0⁺", OrthogonalTo → {V[z]}];
DeclareOrder[DpRP0p[-z], 1];
DeclareOrder[DRP0p[-z], 1,
  "approximation" → B[w, -z] DpRP0p[-w] + V[-v] B[v, -z] V[u] H[-u, w] DRP0p[-w]];
DefTensor[DpRP0m[-z], M4, PrintAs → "𝔇R̂0⁻", OrthogonalTo → {V[z]}];
DeclareOrder[DpRP0m[-z], 1];
DeclareOrder[DRP0m[-z], 1,
  "approximation" → B[w, -z] DpRP0m[-w] + V[-v] B[v, -z] V[u] H[-u, w] DRP0m[-w]];
DefTensor[DpRP1p[-z, -a, -b], M4, Antisymmetric[{-a, -b}],
  PrintAs → "𝔇R̂1⁺", OrthogonalTo → {V[z], V[a], V[b]}];
DeclareOrder[DpRP1p[-z, -a, -b], 1];
DeclareOrder[DRP1p[-z, -a, -b], 1, "approximation" →
   B[w, -z] DpRP1p[-w, -a, -b] + V[-v] B[v, -z] V[u] H[-u, w] DRP1p[-w, -a, -b]];
DefTensor[DpRP1m[-z, -a], M4, PrintAs → "𝔇R̂1⁻", OrthogonalTo → {V[z], V[a]}];
DeclareOrder[DpRP1m[-z, -a], 1];
DeclareOrder[DRP1m[-z, -a], 1, "approximation" →
   B[w, -z] DpRP1m[-w, -a] + V[-v] B[v, -z] V[u] H[-u, w] DRP1m[-w, -a]];
DefTensor[DpRP2p[-z, -a, -b], M4, Symmetric[{-a, -b}],
  PrintAs → "𝔇R̂2⁺", OrthogonalTo → {V[z], V[a], V[b]}];
DeclareOrder[DpRP2p[-z, -a, -b], 1];
DeclareOrder[DRP2p[-z, -a, -b], 1, "approximation" →
   B[w, -z] DpRP2p[-w, -a, -b] + V[-v] B[v, -z] V[u] H[-u, w] DRP2p[-w, -a, -b]];
DefTensor[DpRP2m[-z, -a, -b, -c], M4, Antisymmetric[{-a, -b}],
  PrintAs → "𝔇R̂2⁻", OrthogonalTo → {V[z], V[a], V[b], V[c]}];
DeclareOrder[DpRP2m[-z, -a, -b, -c], 1];
```

```
DeclareOrder[DRP2m[-z, -a, -b, -c],
  1, "approximation" → B[w, -z] DpRP2m[-w, -a, -b, -c] +
    V[-v] B[v, -z] V[u] H[-u, w] DRP2m[-w, -a, -b, -c]];
AutomaticRules[DpRP2m, MakeRule[{DpRP2m[-z, a, -b, -a], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[DpRP2m, MakeRule[{epsilonG[a, b, c, d] DpRP2m[-z, -a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[DpRP2p, MakeRule[{DpRP2p[-z, a, -a], 0},
   MetricOn → All, ContractMetrics → True]];
DpTP0mActivate = MakeRule[{G3[-y, z] DTP0m[-z], G3[-y, z] B[x, -z] DpTP0m[-x]},
   MetricOn → All, ContractMetrics → True];
DpTP1pActivate = MakeRule[{G3[-y, z] DTP1p[-z, -a, -b],
    Evaluate[G3[-y, z] B[x, -z] DpTP1p[-x, -a, -b] +
       (G[-a, i] G[-b, j] - PPara[-a, i] PPara[-b, j]) G3[-y, z] DTP1p[-z, -i, -j] /.
      PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpTP1mActivate = MakeRule[{G3[-y, z] DTP1m[-z, -a],
    Evaluate[G3[-y, z] B[x, -z] DpTP1m[-x, -a] + (G[-a, i] - PPara[-a, i]) G3[-y, z]
        DTP1m[-z, -i] /. PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpTP2mActivate = MakeRule[{G3[-y, z] DTP2m[-z, -a, -b, -c],
    Evaluate[G3[-y, z] B[x, -z] DpTP2m[-x, -a, -b, -c] +
       (G[-a, i] G[-b, j] G[-c, k] - PPara[-a, i] PPara[-b, j] PPara[-c, k])
        G3[-y, z] DTP2m[-z, -i, -j, -k] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True];
DpRP0pActivate = MakeRule[{G3[-y, z] DRP0p[-z], G3[-y, z] B[x, -z] DpRP0p[-x]},
   MetricOn → All, ContractMetrics → True];
DpRP0mActivate = MakeRule[{G3[-y, z] DRP0m[-z], G3[-y, z] B[x, -z] DpRP0m[-x]},
   MetricOn → All, ContractMetrics → True];
DpRP1pActivate = MakeRule[{G3[-y, z] DRP1p[-z, -a, -b],
    Evaluate[G3[-y, z] B[x, -z] DpRP1p[-x, -a, -b] +
       (G[-a, i] G[-b, j] - PPara[-a, i] PPara[-b, j]) G3[-y, z] DRP1p[-z, -i, -j] /.
      PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpRP1mActivate = MakeRule[{G3[-y, z] DRP1m[-z, -a],
    Evaluate[G3[-y, z] B[x, -z] DpRP1m[-x, -a] + (G[-a, i] - PPara[-a, i]) G3[-y, z]
        DRP1m[-z, -i] /. PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpRP2pActivate = MakeRule[{G3[-y, z] DRP2p[-z, -a, -b],
    Evaluate[G3[-y, z] B[x, -z] DpRP2p[-x, -a, -b] +
       (G[-a, i] G[-b, j] - PPara[-a, i] PPara[-b, j]) G3[-y, z] DRP2p[-z, -i, -j] /.
      PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpRP2mActivate = MakeRule[{G3[-y, z] DRP2m[-z, -a, -b, -c],
    Evaluate[G3[-y, z] B[x, -z] DpRP2m[-x, -a, -b, -c] +
       (G[-a, i] G[-b, j] G[-c, k] - PPara[-a, i] PPara[-b, j] PPara[-c, k])
        G3[-y, z] DRP2m[-z, -i, -j, -k] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True];
```

```
DpRPActivate = Join[DpTP0mActivate, DpTP1pActivate, DpTP1mActivate,
   DpTP2mActivate, DpRP0pActivate, DpRP0mActivate, DpRP1pActivate,
   DpRP1mActivate, DpRP2pActivate, DpRP2mActivate];

(*again this should be extended over other derivatives,
multiply the above by PPara[-w,v]H[-v,y]*)
DpTP0mDeactivate = MakeRule[{DpTP0m[-w], PPara[-w, v] H[-v, y] G3[-y, z] DTP0m[-z]},
   MetricOn → All, ContractMetrics → True];
DpTP1pDeactivate = MakeRule[{DpTP1p[-w, -a, -b],
    Evaluate[PPara[-w, v] H[-v, y] G3[-y, z] DTP1p[-z, -a, -b] - PPara[-w, v] H[-v, y]
        (G[-a, i] G[-b, j] - PPara[-a, i] PPara[-b, j]) G3[-y, z] DTP1p[-z, -i, -j] /.
      PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpTP1mDeactivate = MakeRule[{DpTP1m[-w, -a],
    Evaluate[PPara[-w, v] H[-v, y] G3[-y, z] DTP1m[-z, -a] -
       PPara[-w, v] H[-v, y] (G[-a, i] - PPara[-a, i]) G3[-y, z] DTP1m[-z, -i] /.
      PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpTP2mDeactivate = MakeRule[{DpTP2m[-w, -a, -b, -c],
    Evaluate[PPara[-w, v] H[-v, y] G3[-y, z] DTP2m[-z, -a, -b, -c] - PPara[-w, v]
        H[-v, y] (G[-a, i] G[-b, j] G[-c, k] - PPara[-a, i] PPara[-b, j] PPara[-c, k])
        G3[-y, z] DTP2m[-z, -i, -j, -k] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True];
DpRP0pDeactivate = MakeRule[{DpRP0p[-w], PPara[-w, v] H[-v, y] G3[-y, z] DRP0p[-z]},
   MetricOn → All, ContractMetrics → True];
DpRP0mDeactivate = MakeRule[{DpRP0m[-w], PPara[-w, v] H[-v, y] G3[-y, z] DRP0m[-z]},
   MetricOn → All, ContractMetrics → True];
DpRP1pDeactivate = MakeRule[{DpRP1p[-w, -a, -b],
    Evaluate[PPara[-w, v] H[-v, y] G3[-y, z] DRP1p[-z, -a, -b] - PPara[-w, v] H[-v, y]
        (G[-a, i] G[-b, j] - PPara[-a, i] PPara[-b, j]) G3[-y, z] DRP1p[-z, -i, -j] /.
      PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpRP1mDeactivate = MakeRule[{DpRP1m[-w, -a],
    Evaluate[PPara[-w, v] H[-v, y] G3[-y, z] DRP1m[-z, -a] -
       PPara[-w, v] H[-v, y] (G[-a, i] - PPara[-a, i]) G3[-y, z] DRP1m[-z, -i] /.
      PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpRP2pDeactivate = MakeRule[{DpRP2p[-w, -a, -b],
    Evaluate[PPara[-w, v] H[-v, y] G3[-y, z] DRP2p[-z, -a, -b] - PPara[-w, v] H[-v, y]
        (G[-a, i] G[-b, j] - PPara[-a, i] PPara[-b, j]) G3[-y, z] DRP2p[-z, -i, -j] /.
      PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpRP2mDeactivate = MakeRule[{DpRP2m[-w, -a, -b, -c],
    Evaluate[PPara[-w, v] H[-v, y] G3[-y, z] DRP2m[-z, -a, -b, -c] - PPara[-w, v]
        H[-v, y] (G[-a, i] G[-b, j] G[-c, k] - PPara[-a, i] PPara[-b, j] PPara[-c, k])
        G3[-y, z] DRP2m[-z, -i, -j, -k] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True];
```

```
DpRPDeactivate = Join[DpTP0mDeactivate, DpTP1pDeactivate, DpTP1mDeactivate,
   DpTP2mDeactivate, DpRP0pDeactivate, DpRP0mDeactivate, DpRP1pDeactivate,
   DpRP1mDeactivate, DpRP2pDeactivate, DpRP2mDeactivate];
```

Define $\hat{\mathbb{D}}\hat{\lambda}J^{P}$

```
In[]:= DefTensor[DpTLambdaP0m[-z], M4, PrintAs → "𝔻̂T̂λ0⁻", OrthogonalTo → {V[z]}];
DeclareOrder[DpTLambdaP0m[-z], 1];
DeclareOrder[DTLambdaP0m[-z], 1, "approximation" →
   B[w, -z] DpTLambdaP0m[-w] + V[-v] B[v, -z] V[u] H[-u, w] DTLambdaP0m[-w]];
DefTensor[DpTLambdaP1p[-z, -a, -b], M4, Antisymmetric[{-a, -b}],
  PrintAs → "𝔻̂T̂λ1⁺", OrthogonalTo → {V[z], V[a], V[b]}];
DeclareOrder[DpTLambdaP1p[-z, -a, -b], 1];
DeclareOrder[DTLambdaP1p[-z, -a, -b],
  1, "approximation" → B[w, -z] DpTLambdaP1p[-w, -a, -b] +
    V[-v] B[v, -z] V[u] H[-u, w] DTLambdaP1p[-w, -a, -b]];
DefTensor[DpTLambdaP1m[-z, -a], M4, PrintAs → "𝔻̂T̂λ1⁻", OrthogonalTo → {V[z], V[a]}];
DeclareOrder[DpTLambdaP1m[-z, -a], 1];
DeclareOrder[DTLambdaP1m[-z, -a], 1, "approximation" →
   B[w, -z] DpTLambdaP1m[-w, -a] + V[-v] B[v, -z] V[u] H[-u, w] DTLambdaP1m[-w, -a]];
DefTensor[DpTLambdaP2m[-z, -a, -b, -c], M4, Antisymmetric[{-a, -b}],
  PrintAs → "𝔻̂T̂λ2⁻", OrthogonalTo → {V[z], V[a], V[b], V[c]}];
DeclareOrder[DpTLambdaP2m[-z, -a, -b, -c], 1];
DeclareOrder[DTLambdaP2m[-z, -a, -b, -c],
  1, "approximation" → B[w, -z] DpTLambdaP2m[-w, -a, -b, -c] +
    V[-v] B[v, -z] V[u] H[-u, w] DTLambdaP2m[-w, -a, -b, -c]];
AutomaticRules[DpTLambdaP2m, MakeRule[{DpTLambdaP2m[-z, a, -b, -a], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[DpTLambdaP2m, MakeRule[
   {epsilonG[a, b, c, d] DpTLambdaP2m[-z, -a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];
DefTensor[DpRLambdaP0p[-z], M4, PrintAs → "𝔻̂R̂λ0⁺", OrthogonalTo → {V[z]}];
DeclareOrder[DpRLambdaP0p[-z], 1];
DeclareOrder[DRLambdaP0p[-z], 1, "approximation" →
   B[w, -z] DpRLambdaP0p[-w] + V[-v] B[v, -z] V[u] H[-u, w] DRLambdaP0p[-w]];
DefTensor[DpRLambdaP0m[-z], M4, PrintAs → "𝔻̂R̂λ0⁻", OrthogonalTo → {V[z]}];
DeclareOrder[DpRLambdaP0m[-z], 1];
DeclareOrder[DRLambdaP0m[-z], 1, "approximation" →
   B[w, -z] DpRLambdaP0m[-w] + V[-v] B[v, -z] V[u] H[-u, w] DRLambdaP0m[-w]];
DefTensor[DpRLambdaP1p[-z, -a, -b], M4, Antisymmetric[{-a, -b}],
  PrintAs → "𝔻̂R̂λ1⁺", OrthogonalTo → {V[z], V[a], V[b]}];
DeclareOrder[DpRLambdaP1p[-z, -a, -b], 1];
DeclareOrder[DRLambdaP1p[-z, -a, -b],
```

```
  1, "approximation" → B[w, -z] DpRLambdaP1p[-w, -a, -b] +
    V[-v] B[v, -z] V[u] H[-u, w] DRLambdaP1p[-w, -a, -b]];
DefTensor[DpRLambdaP1m[-z, -a], M4, PrintAs → "𝕯R̂λ1⁻", OrthogonalTo → {V[z], V[a]}];
DeclareOrder[DpRLambdaP1m[-z, -a], 1];
DeclareOrder[DRLambdaP1m[-z, -a], 1, "approximation" →
   B[w, -z] DpRLambdaP1m[-w, -a] + V[-v] B[v, -z] V[u] H[-u, w] DRLambdaP1m[-w, -a]];
DefTensor[DpRLambdaP2p[-z, -a, -b], M4, Symmetric[{-a, -b}],
  PrintAs → "𝕯R̂λ2⁺", OrthogonalTo → {V[z], V[a], V[b]}];
DeclareOrder[DpRLambdaP2p[-z, -a, -b], 1];
DeclareOrder[DRLambdaP2p[-z, -a, -b],
  1, "approximation" → B[w, -z] DpRLambdaP2p[-w, -a, -b] +
    V[-v] B[v, -z] V[u] H[-u, w] DRLambdaP2p[-w, -a, -b]];
DefTensor[DpRLambdaP2m[-z, -a, -b, -c], M4, Antisymmetric[{-a, -b}],
  PrintAs → "𝕯R̂λ2⁻", OrthogonalTo → {V[z], V[a], V[b], V[c]}];
DeclareOrder[DpRLambdaP2m[-z, -a, -b, -c], 1];
DeclareOrder[DRLambdaP2m[-z, -a, -b, -c],
  1, "approximation" → B[w, -z] DpRLambdaP2m[-w, -a, -b, -c] +
    V[-v] B[v, -z] V[u] H[-u, w] DRLambdaP2m[-w, -a, -b, -c]];
AutomaticRules[DpRLambdaP2m, MakeRule[{DpRLambdaP2m[-z, a, -b, -a], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[DpRLambdaP2m, MakeRule[
   {epsilonG[a, b, c, d] DpRLambdaP2m[-z, -a, -b, -c], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[DpRLambdaP2p, MakeRule[{DpRLambdaP2p[-z, a, -a], 0},
   MetricOn → All, ContractMetrics → True]];
DpTLambdaP0mActivate = MakeRule[{G3[-y, z] DTLambdaP0m[-z],
    G3[-y, z] B[x, -z] DpTLambdaP0m[-x]}, MetricOn → All, ContractMetrics → True];
DpTLambdaP1pActivate = MakeRule[{G3[-y, z] DTLambdaP1p[-z, -a, -b],
    Evaluate[G3[-y, z] B[x, -z] DpTLambdaP1p[-x, -a, -b] +
        (G[-a, i] G[-b, j] - PPara[-a, i] PPara[-b, j]) G3[-y, z] DTLambdaP1p[-z,
          -i, -j] /. PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpTLambdaP1mActivate = MakeRule[{G3[-y, z] DTLambdaP1m[-z, -a],
    Evaluate[G3[-y, z] B[x, -z] DpTLambdaP1m[-x, -a] +
        (G[-a, i] - PPara[-a, i]) G3[-y, z] DTLambdaP1m[-z, -i] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True];
DpTLambdaP2mActivate = MakeRule[{G3[-y, z] DTLambdaP2m[-z, -a, -b, -c],
    Evaluate[G3[-y, z] B[x, -z] DpTLambdaP2m[-x, -a, -b, -c] +
        (G[-a, i] G[-b, j] G[-c, k] - PPara[-a, i] PPara[-b, j] PPara[-c, k])
         G3[-y, z] DTLambdaP2m[-z, -i, -j, -k] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True];
DpRLambdaP0pActivate = MakeRule[{G3[-y, z] DRLambdaP0p[-z],
    G3[-y, z] B[x, -z] DpRLambdaP0p[-x]}, MetricOn → All, ContractMetrics → True];
DpRLambdaP0mActivate = MakeRule[{G3[-y, z] DRLambdaP0m[-z],
```

```
    G3[-y, z] B[x, -z] DpRLambdaP0m[-x]}, MetricOn → All, ContractMetrics → True];
DpRLambdaP1pActivate = MakeRule[{G3[-y, z] DRLambdaP1p[-z, -a, -b],
    Evaluate[G3[-y, z] B[x, -z] DpRLambdaP1p[-x, -a, -b] +
        (G[-a, i] G[-b, j] - PPara[-a, i] PPara[-b, j]) G3[-y, z] DRLambdaP1p[-z,
          -i, -j] /. PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpRLambdaP1mActivate = MakeRule[{G3[-y, z] DRLambdaP1m[-z, -a],
    Evaluate[G3[-y, z] B[x, -z] DpRLambdaP1m[-x, -a] +
        (G[-a, i] - PPara[-a, i]) G3[-y, z] DRLambdaP1m[-z, -i] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True];
DpRLambdaP2pActivate = MakeRule[{G3[-y, z] DRLambdaP2p[-z, -a, -b],
    Evaluate[G3[-y, z] B[x, -z] DpRLambdaP2p[-x, -a, -b] +
        (G[-a, i] G[-b, j] - PPara[-a, i] PPara[-b, j]) G3[-y, z] DRLambdaP2p[-z,
          -i, -j] /. PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpRLambdaP2mActivate = MakeRule[{G3[-y, z] DRLambdaP2m[-z, -a, -b, -c],
    Evaluate[G3[-y, z] B[x, -z] DpRLambdaP2m[-x, -a, -b, -c] +
        (G[-a, i] G[-b, j] G[-c, k] - PPara[-a, i] PPara[-b, j] PPara[-c, k])
         G3[-y, z] DRLambdaP2m[-z, -i, -j, -k] /. PADMActivate]},
   MetricOn → All, ContractMetrics → True];
DpRLambdaPActivate = Join[DpTLambdaP0mActivate, DpTLambdaP1pActivate,
   DpTLambdaP1mActivate, DpTLambdaP2mActivate, DpRLambdaP0pActivate,
   DpRLambdaP0mActivate, DpRLambdaP1pActivate, DpRLambdaP1mActivate,
   DpRLambdaP2pActivate, DpRLambdaP2mActivate];
```

Define $\hat{\mathbb{D}}\overset{*}{\lambda} J^P$

```
In[•]:= DefTensor[DpTLambdaPerp0p[-z], M4, PrintAs → "𝕯̂Tλ̊0⁺", OrthogonalTo → {V[z]}];
DeclareOrder[DpTLambdaPerp0p[-z], 1];
DeclareOrder[DTLambdaPerp0p[-z], 1, "approximation" →
   B[w, -z] DpTLambdaPerp0p[-w] + V[-v] B[v, -z] V[u] H[-u, w] DTLambdaPerp0p[-w]];
DefTensor[DpTLambdaPerp1p[-z, -a, -b], M4, Antisymmetric[{-a, -b}],
  PrintAs → "𝕯̂Tλ̊1⁺", OrthogonalTo → {V[z], V[a], V[b]}];
DeclareOrder[DpTLambdaPerp1p[-z, -a, -b], 1];
DeclareOrder[DTLambdaPerp1p[-z, -a, -b],
  1, "approximation" → B[w, -z] DpTLambdaPerp1p[-w, -a, -b] +
    V[-v] B[v, -z] V[u] H[-u, w] DTLambdaPerp1p[-w, -a, -b]];
DefTensor[DpTLambdaPerp1m[-z, -a], M4, PrintAs → "𝕯̂Tλ̊1⁻",
  OrthogonalTo → {V[z], V[a]}];
DeclareOrder[DpTLambdaPerp1m[-z, -a], 1];
DeclareOrder[DTLambdaPerp1m[-z, -a],
  1, "approximation" → B[w, -z] DpTLambdaPerp1m[-w, -a] +
    V[-v] B[v, -z] V[u] H[-u, w] DTLambdaPerp1m[-w, -a]];
DefTensor[DpTLambdaPerp2p[-z, -a, -b], M4, Symmetric[{-a, -b}],
```

```
  PrintAs → "𝔇̂T̂λ2⁺", OrthogonalTo → {V[z], V[a], V[b]}];
DeclareOrder[DpTLambdaPerp2p[-z, -a, -b], 1];
DeclareOrder[DTLambdaPerp2p[-z, -a, -b],
  1, "approximation" → B[w, -z] DpTLambdaPerp2p[-w, -a, -b] +
    V[-v] B[v, -z] V[u] H[-u, w] DTLambdaPerp2p[-w, -a, -b]];
AutomaticRules[DpTLambdaPerp2p, MakeRule[{DpTLambdaPerp2p[-z, a, -a], 0},
   MetricOn → All, ContractMetrics → True]];
DefTensor[DpRLambdaPerp0p[-z], M4, PrintAs → "𝔇̂λ̂0⁺", OrthogonalTo → {V[z]}];
DeclareOrder[DpRLambdaPerp0p[-z], 1];
DeclareOrder[DRLambdaPerp0p[-z], 1, "approximation" →
   B[w, -z] DpRLambdaPerp0p[-w] + V[-v] B[v, -z] V[u] H[-u, w] DRLambdaPerp0p[-w]];
DefTensor[DpRLambdaPerp0m[-z], M4, PrintAs → "𝔇̂λ̂0⁻", OrthogonalTo → {V[z]}];
DeclareOrder[DpRLambdaPerp0m[-z], 1];
DeclareOrder[DRLambdaPerp0m[-z], 1, "approximation" →
   B[w, -z] DpRLambdaPerp0m[-w] + V[-v] B[v, -z] V[u] H[-u, w] DRLambdaPerp0m[-w]];
DefTensor[DpRLambdaPerp1p[-z, -a, -b], M4, Antisymmetric[{-a, -b}],
  PrintAs → "𝔇̂λ̂1⁺", OrthogonalTo → {V[z], V[a], V[b]}];
DeclareOrder[DpRLambdaPerp1p[-z, -a, -b], 1];
DeclareOrder[DRLambdaPerp1p[-z, -a, -b],
  1, "approximation" → B[w, -z] DpRLambdaPerp1p[-w, -a, -b] +
    V[-v] B[v, -z] V[u] H[-u, w] DRLambdaPerp1p[-w, -a, -b]];
DefTensor[DpRLambdaPerp1m[-z, -a], M4, PrintAs → "𝔇̂λ̂1⁻",
  OrthogonalTo → {V[z], V[a]}];
DeclareOrder[DpRLambdaPerp1m[-z, -a], 1];
DeclareOrder[DRLambdaPerp1m[-z, -a],
  1, "approximation" → B[w, -z] DpRLambdaPerp1m[-w, -a] +
    V[-v] B[v, -z] V[u] H[-u, w] DRLambdaPerp1m[-w, -a]];
DefTensor[DpRLambdaPerp2p[-z, -a, -b], M4, Symmetric[{-a, -b}],
  PrintAs → "𝔇̂λ̂2⁺", OrthogonalTo → {V[z], V[a], V[b]}];
DeclareOrder[DpRLambdaPerp2p[-z, -a, -b], 1];
DeclareOrder[DRLambdaPerp2p[-z, -a, -b],
  1, "approximation" → B[w, -z] DpRLambdaPerp2p[-w, -a, -b] +
    V[-v] B[v, -z] V[u] H[-u, w] DRLambdaPerp2p[-w, -a, -b]];
DefTensor[DpRLambdaPerp2m[-z, -a, -b, -c], M4, Antisymmetric[{-a, -b}],
  PrintAs → "𝔇̂λ̂2⁻", OrthogonalTo → {V[z], V[a], V[b], V[c]}];
DeclareOrder[DpRLambdaPerp2m[-z, -a, -b, -c], 1];
DeclareOrder[DRLambdaPerp2m[-z, -a, -b, -c], 1,
  "approximation" → B[w, -z] DpRLambdaPerp2m[-w, -a, -b, -c] +
    V[-v] B[v, -z] V[u] H[-u, w] DRLambdaPerp2m[-w, -a, -b, -c]];
AutomaticRules[DpRLambdaPerp2m, MakeRule[{DpRLambdaPerp2m[-z, a, -b, -a], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[DpRLambdaPerp2m, MakeRule[
   {epsilonG[a, b, c, d] DpRLambdaPerp2m[-z, -a, -b, -c], 0},
```

```
    MetricOn → All, ContractMetrics → True]];
AutomaticRules[DpRLambdaPerp2p, MakeRule[{DpRLambdaPerp2p[-z, a, -a], 0},
    MetricOn → All, ContractMetrics → True]];
DpTLambdaPerp0pActivate = MakeRule[{G3[-y, z] DTLambdaPerp0p[-z],
     G3[-y, z] B[x, -z] DpTLambdaPerp0p[-x]}, MetricOn → All, ContractMetrics → True];
DpTLambdaPerp1pActivate = MakeRule[{G3[-y, z] DTLambdaPerp1p[-z, -a, -b],
     Evaluate[G3[-y, z] B[x, -z] DpTLambdaPerp1p[-x, -a, -b] +
        (G[-a, i] G[-b, j] - PPara[-a, i] PPara[-b, j]) G3[-y, z] DTLambdaPerp1p[-z,
          -i, -j] /. PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpTLambdaPerp1mActivate = MakeRule[{G3[-y, z] DTLambdaPerp1m[-z, -a],
     Evaluate[G3[-y, z] B[x, -z] DpTLambdaPerp1m[-x, -a] +
        (G[-a, i] - PPara[-a, i]) G3[-y, z] DTLambdaPerp1m[-z, -i] /. PADMActivate]},
    MetricOn → All, ContractMetrics → True];
DpTLambdaPerp2pActivate = MakeRule[{G3[-y, z] DTLambdaPerp2p[-z, -a, -b],
     Evaluate[G3[-y, z] B[x, -z] DpTLambdaPerp2p[-x, -a, -b] +
        (G[-a, i] G[-b, j] - PPara[-a, i] PPara[-b, j]) G3[-y, z] DTLambdaPerp2p[-z,
          -i, -j] /. PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpRLambdaPerp0pActivate = MakeRule[{G3[-y, z] DRLambdaPerp0p[-z],
     G3[-y, z] B[x, -z] DpRLambdaPerp0p[-x]}, MetricOn → All, ContractMetrics → True];
DpRLambdaPerp0mActivate = MakeRule[{G3[-y, z] DRLambdaPerp0m[-z],
     G3[-y, z] B[x, -z] DpRLambdaPerp0m[-x]}, MetricOn → All, ContractMetrics → True];
DpRLambdaPerp1pActivate = MakeRule[{G3[-y, z] DRLambdaPerp1p[-z, -a, -b],
     Evaluate[G3[-y, z] B[x, -z] DpRLambdaPerp1p[-x, -a, -b] +
        (G[-a, i] G[-b, j] - PPara[-a, i] PPara[-b, j]) G3[-y, z] DRLambdaPerp1p[-z,
          -i, -j] /. PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpRLambdaPerp1mActivate = MakeRule[{G3[-y, z] DRLambdaPerp1m[-z, -a],
     Evaluate[G3[-y, z] B[x, -z] DpRLambdaPerp1m[-x, -a] +
        (G[-a, i] - PPara[-a, i]) G3[-y, z] DRLambdaPerp1m[-z, -i] /. PADMActivate]},
    MetricOn → All, ContractMetrics → True];
DpRLambdaPerp2pActivate = MakeRule[{G3[-y, z] DRLambdaPerp2p[-z, -a, -b],
     Evaluate[G3[-y, z] B[x, -z] DpRLambdaPerp2p[-x, -a, -b] +
        (G[-a, i] G[-b, j] - PPara[-a, i] PPara[-b, j]) G3[-y, z] DRLambdaPerp2p[-z,
          -i, -j] /. PADMActivate]}, MetricOn → All, ContractMetrics → True];
DpRLambdaPerp2mActivate = MakeRule[{G3[-y, z] DRLambdaPerp2m[-z, -a, -b, -c],
     Evaluate[G3[-y, z] B[x, -z] DpRLambdaPerp2m[-x, -a, -b, -c] +
        (G[-a, i] G[-b, j] G[-c, k] - PPara[-a, i] PPara[-b, j] PPara[-c, k])
         G3[-y, z] DRLambdaPerp2m[-z, -i, -j, -k] /. PADMActivate]},
    MetricOn → All, ContractMetrics → True];
DpRLambdaPerpActivate = Join[DpTLambdaPerp0pActivate, DpTLambdaPerp1pActivate,
    DpTLambdaPerp1mActivate, DpTLambdaPerp2pActivate, DpRLambdaPerp0pActivate,
    DpRLambdaPerp0mActivate, DpRLambdaPerp1pActivate, DpRLambdaPerp1mActivate,
    DpRLambdaPerp2pActivate, DpRLambdaPerp2mActivate];
```

Define $\hat{\mathbb{D}}$H

```
In[]:= DefTensor[DpHComp[-z], M4, PrintAs → "𝔻̂H", OrthogonalTo → {V[z]}];
DpHCompActivate = MakeRule[{G3[-y, z] DHComp[-z], G3[-y, z] B[x, -z] DpHComp[-x]},
    MetricOn → All, ContractMetrics → True];

DGrandActivate = Join[DPiPActivate, DRPActivate,
    DRLambdaPActivate, DRLambdaPerpActivate, DHCompActivate];
DpGrandActivate = Join[DpPiPActivate, DpRPActivate, DpRLambdaPActivate,
    DpRLambdaPerpActivate, DpJActivate, DpVActivate, DpHCompActivate];
```

Theory-specific calculations using O(3)

ORPHAN

```
In[]:= (*
MatRules={xPhiB1p→1,xPhiB2p→2,xPhiA0p→3,xPhiA2m→4,xTheA0m→5,xTheA1p→6,
  xTheA1m→7,xTheA2p→8,xTheA2m→9,xTheB1p→10,xTheB2m→11,out→7777,der→9999};
InputMatrix={{{},{},{xTheA1p},{xTheA0m,xTheA1m,xTheA2m},{xTheA1m},
   {out},{xTheA0m,xTheA1m},{xTheA1p,xTheA2p},{out},{out},{der}},
  {{fil},{},{xTheA2p},{xTheA0m,xTheA1m,xTheA2m},{},{out},
   {xTheA1m},{xTheA1p,xTheA2p},{out},{out},{der}},
  {{fil},{fil},{},{xTheB1p},{out},{xTheB1p},{der},{},{xTheB2m},{},{}},
  {{fil},{fil},{fil},{xTheB2m},{xTheB1p},{der},{},{der},{out},{},{con}},
  {{fil},{fil},{fil},{fil},{},{der},{},{xTheB2m},{out},{},{}},
  {{fil},{fil},{fil},{fil},{fil},{},{der},{xTheB1p},{der},{con},{}},
  {{fil},{fil},{fil},{fil},{fil},{fil},{},{der},{xTheB1p},{},{}},
  {{fil},{fil},{fil},{fil},{fil},{fil},{fil},{},{der},{},{}},
  {{fil},{fil},{fil},{fil},{fil},{fil},{fil},{fil},{},{},{}},
  {{fil},{fil},{fil},{fil},{fil},{fil},{fil},{fil},{fil},{},{}},
  {{fil},{fil},{fil},{fil},{fil},{fil},{fil},{fil},{fil},{fil},{}}};

Print[InputMatrix[[1,3]]];


For[ii=1,ii<12,ii++,
 For[jj=ii,jj<12,jj++,InputMatrix[[jj,ii]]=InputMatrix[[ii,jj]]]];
InputMatrix=InputMatrix/.MatRules;

Print[InputMatrix//MatrixForm];

Print["Looking for options"];
```

```
pri=1;
NewThetas={};
For[ii=1,ii<5,ii++,NewThetas=Union[NewThetas,InputMatrix[[pri,ii]]]];
Print[NewThetas];
ExtendedThetas=NewThetas;
Collisions=Map[InputMatrix[[pri,#]]&,NewThetas];
Print[Collisions];
ExtendedThetas=Union[ExtendedThetas,Flatten[Collisions]];
Print[ExtendedThetas];
AllConstraints=Union[{1,2,3,4},ExtendedThetas];
Print[AllConstraints];
Augmentation=Length[ExtendedThetas];
Print[Augmentation];

For[ii=1,ii<Augmentation+5,ii++,
 For[jj=ii,jj<Augmentation+5,jj++,
  {Commutator=InputMatrix[[AllConstraints[[ii]],AllConstraints[[jj]]]],
   Commutator=Complement[Commutator,Intersection[Commutator,ExtendedThetas]],
   If[Commutator≠{},{Print[AllConstraints[[ii]],AllConstraints[[jj]]],
     Print[Commutator]}]
  }]];

*)
```

ORPHAN

This used to be part of ω rules, but was commented out even then.

```
In[◦]:= (*

(*consideration of expansion subroutine in linearisation procedure*)

(*
BPiDefinitionx= G3[-d,c]H[x,-c]PPara[-x,m]((1/3)PPara[-n,-m]PiB0p[]+
        PiB1p[-n,-m]+
        PiB2p[-n,-m]+
        V[-n]PiB1m[-m])/.PO3PiActivate/.PADMActivate//ToCanonical;

APiDefinitionx= G3[-d,c]H[x,-c]PPara[-x,o]
     (Antisymmetrize[2Antisymmetrize[V[-n](1/3)PPara[-m,-o]PiA0p[],{-n,-m}]+
       2Antisymmetrize[V[-n]PiA1p[-m,-o],{-n,-m}]+
       2Antisymmetrize[V[-n]PiA2p[-m,-o],{-n,-m}]+
       (-1/6)PA0m[-n,-m,-o]PiA0m[]+
       Antisymmetrize[-PPara[-m,-o]PiA1m[-n],{-m,-n}]+
```

```
        (4/3) PiA2m[-n,-m,-o],{-n,-m}]) /.PO3PiActivate/.PADMActivate//
  ToCanonical;


(*PiPActivate*)
APiActivatex=MakeRule[{APi[-n,-m,-d],Evaluate[APiDefinitionx]},
  MetricOn→All,ContractMetrics→True];

BPiActivatex=MakeRule[{BPi[-n,-d],Evaluate[BPiDefinitionx]},
  MetricOn→All,ContractMetrics→True];
PiExpansion=Join[APiActivatex,BPiActivatex];

Print["APi expansion into defined O(3) momenta"];
probe=APi[-a,-b,-c]/.PiExpansion;
Print[probe];
Print[
 "APi expansion of defined O(3) momenta into operators and whole momentum"];
probe=probe/.PiPActivate//ScreenDollarIndices//ToCanonical;
probe=probe/.HG3BExpand//ScreenDollarIndices//ToCanonical;
probe=probe//NoScalar//ScreenDollarIndices//ToCanonical;
probe=probe//ContractMetric//ScreenDollarIndices//ToCanonical;
probe=probe//ScreenDollarIndices//ToCanonical;
Print[probe];
Print["BPi expansion into defined O(3) momenta"];
probe=BPi[-a,-b]/.PiExpansion;
Print[probe];
Print[
 "BPi expansion of defined O(3) momenta into operators and whole momentum"];
probe=probe/.PiPActivate//ScreenDollarIndices//ToCanonical;
probe=probe/.HG3BExpand//ScreenDollarIndices//ToCanonical;
probe=probe//NoScalar//ScreenDollarIndices//ToCanonical;
probe=probe//ContractMetric//ScreenDollarIndices//ToCanonical;
probe=probe//ScreenDollarIndices//ToCanonical;
Print[probe];
*)
*)
```

In[•]:=

ORPHAN

This is all the ω material here, one big comment as of Feb.

In[•]:=
```
(*
DefNiceConstantSymbol[ξ,ToExpression[#]]&/@SectorNames;
DefNiceConstantSymbol[ϱ,ToExpression[#]]&/@SectorNames;
DefNiceConstantSymbol[λ,ToExpression[#]]&/@SectorNames;
DefNiceConstantSymbol[ν,ToExpression[#]]&/@SectorNames;
DefNiceConstantSymbol[μ,ToExpression[#]]&/@SectorNames;
DefNiceConstantSymbol[ω,ToExpression[#]]&/@SectorNames;
DefNiceConstantSymbol[κ,ToExpression[#]]&/@SectorNames;
DefNiceConstantSymbol[γ,ToExpression[#]]&/@SectorNames;





Quit[];

Solutions;
AntiSolutions;
TotalSolutions=Join[Solutions,AntiSolutions]
  Print["totalsolutions"]
  Print[TotalSolutions];
Print["cf"]
 Quit[];

Freedoms={};
CriticalCombinations={Bet2,Bet1+2Bet3,2Bet1+Bet2,Bet1,
  Alp4+Alp6,Alp2+Alp3,Alp2+Alp5,Alp4+Alp5,Alp1+Alp4,Alp1+Alp2};
Relevants={};
Simplicities={};
(*These are zero when the Phi can be written purely as a momentum -- whether
  it can also be written as nonphysical+velocity depends on freedoms*)
SimpleCombinations={0,Bet1-Bet3,Bet1-Bet2,0,Alp4-Alp6,
  Alp2-Alp3,Alp2-Alp5,Alp4-Alp5,Alp1-Alp4,Alp1-Alp2};
Masses={Alp0(2Alp0+Bet2),Alp0/2+Bet3,(Alp0/2+Bet3)(Bet1-Alp0),
  (2Alp0+Bet2)(Bet1-Alp0),Alp0(Bet1-Alp0),Bet1-Alp0};

For[ii=1,ii<11,ii++,
  If[Evaluate[CriticalCombinations[[ii]]/.Theory]==0,AppendTo[Freedoms,Evaluate[
     ToExpression["λ"<>ToString[SectorNames[[ii]]]<>"->0"]]],AppendTo[Freedoms,
    Evaluate[ToExpression["λ"<>ToString[SectorNames[[ii]]]<>"->1"]],AppendTo[
    Freedoms,Evaluate[ToExpression["λ"<>ToString[SectorNames[[ii]]]<>"->1"]]]]]
 For[ii=1,ii<11,ii++,If[Evaluate[SimpleCombinations[[ii]]/.Theory]==0,
   AppendTo[Simplicities,
```

```
    Evaluate[ToExpression["νx"<>ToString[SectorNames[[ii]]]<>"->0"]]],
   AppendTo[Simplicities,Evaluate[ToExpression[
      "νx"<>ToString[SectorNames[[ii]]]<>"->1"]]],AppendTo[Simplicities,
    Evaluate[ToExpression["νx"<>ToString[SectorNames[[ii]]]<>"->1"]]]]]

 For[ii=1,ii<11,ii++,If[(Evaluate[SimpleCombinations[[ii]]/.Theory]==0&&
     Evaluate[CriticalCombinations[[ii]]/.Theory]==0),AppendTo[Relevants,
    Evaluate[ToExpression["ν"<>ToString[SectorNames[[ii]]]<>"->0"]]],
   AppendTo[Relevants,Evaluate[ToExpression[
      "ν"<>ToString[SectorNames[[ii]]]<>"->1"]]],AppendTo[Relevants,
    Evaluate[ToExpression["ν"<>ToString[SectorNames[[ii]]]<>"->1"]]]]]


 For[ii=1,ii<11,ii++,Print[Evaluate[SimpleCombinations[[ii]]/.Theory]]]

 Print[Freedoms];
Print[Relevants];
Print[Simplicities];
Print[TotalSolutions];

MomentumListB=DeleteCases[{νB0p PiB0p[],
    νB1p PiB1p[-a,-b],νB1m PiB1m[-a],νB2p PiB2p[-a,-b]}/.Relevants,0];
XMomentumListB=Eps[-x,-y,-z]#&/@MomentumListB;
MomentumListB=Join[MomentumListB,XMomentumListB];
ScalarMomentumListB={};
For[ii=1,ii<Length[MomentumListB]+1,ii++,
 If[Length[Evaluate[FindFreeIndices[Evaluate[MomentumListB[[ii]]]]]]==0,
  AppendTo[ScalarMomentumListB,MomentumListB[[ii]]]]];

MomentumListA=
 DeleteCases[{νA0p PiA0p[],νA0m PiA0m[],νA1p PiA1p[-a,-b],νA1m PiA1m[-a],
    νA2p PiA2p[-a,-b],νA2m PiA2m[-a,-b,-c]}/.Relevants,0];
XMomentumListA=Eps[-x,-y,-z]#&/@MomentumListA;
MomentumListA=Join[MomentumListA,XMomentumListA];
ScalarMomentumListA={};
For[ii=1,ii<Length[MomentumListA]+1,ii++,
 If[Length[Evaluate[FindFreeIndices[Evaluate[MomentumListA[[ii]]]]]]==0,
  AppendTo[ScalarMomentumListA,MomentumListA[[ii]]]]];

NewFreedoms={};
CheckFreedoms={};
Scan[Module[{Av,Bv,Cv,Dv,total,readoff,sector},sector=#;
    Av=ToExpression["ξ"<>ToString[sector]];
```

```
    Bv=ToExpression["c"<>ToString[sector]];
    Cv=ToExpression["λ"<>ToString[sector]];
    Dv=ToExpression["ω"<>ToString[sector]];
    total=Av Bv Cv;
    readoff=Av Bv;
    total=total/.Freedoms;
    total=total/.TotalSolutions;
    readoff=readoff/.TotalSolutions;
    total=total/.Theory;
    AppendTo[NewFreedoms,Evaluate[Dv→total]];
    AppendTo[CheckFreedoms,Evaluate[Dv→readoff]];]&,SectorNames]

 Print[CheckFreedoms];
Print[NewFreedoms];
*)
```

Constraint Structure

```
In[•]:= DefTheory[InputSystem_] :=
  Module[{KeepOnlyObviousZeros, cAlpPerpPerpTheory, cAlpPerpParaTheory,
    cAlpParaPerpTheory, cAlpParaParaTheory, cAlpDetTheory, AlpPerpPerpTheory,
    AlpPerpParaTheory, AlpParaPerpTheory, AlpParaParaTheory, AlpDetTheory,
    cBetPerpPerpTheory, cBetPerpParaTheory, cBetParaPerpTheory,
    cBetParaParaTheory, cBetDetTheory, BetPerpPerpTheory, BetPerpParaTheory,
    BetParaPerpTheory, BetParaParaTheory, BetDetTheory},

   xATP`MakeDefInfo[DefTheory, InputSystem, {"shell freedoms", ""}];

   (*these are rules we can always use to impose the theory*)
   ToTheory = Quiet[Solve[InputSystem, Join[cAlp, cBet, Alp, Bet]][[1]]];

   (*Here are the generalised freedom coefficients*)
   DefNiceConstantSymbol[ShellPara, ToExpression[#]] & /@ ASectorNames;
   DefNiceConstantSymbol[ShellOrig, ToExpression[#]] & /@ ASectorNames;
   DefNiceConstantSymbol[ShellPerp, ToExpression[#]] & /@ ASectorNames;
   DefNiceConstantSymbol[ShellSing, ToExpression[#]] & /@ ASectorNames;
   DefNiceConstantSymbol[ShellPrim, ToExpression[#]] & /@ ASectorNames;
   DefNiceConstantSymbol[ShellPara, ToExpression[#]] & /@ BSectorNames;
   DefNiceConstantSymbol[ShellOrig, ToExpression[#]] & /@ BSectorNames;
   DefNiceConstantSymbol[ShellPerp, ToExpression[#]] & /@ BSectorNames;
   DefNiceConstantSymbol[ShellSing, ToExpression[#]] & /@ BSectorNames;
   DefNiceConstantSymbol[ShellPrim, ToExpression[#]] & /@ BSectorNames;
```

```
(*We don't want our theory-defining rules to have unintended side-effects...
 so we only keep zeros which pop out of the initial rules.*)
KeepOnlyObviousZeros[q_] := If[q == 0, 0, 1, 1];

(*We impose the theory on the coefficients*)
cAlpPerpPerpTheory =
 KeepOnlyObviousZeros /@ (cAlpPerpPerp /. TocAlp /. ToTheory);
cAlpPerpParaTheory = KeepOnlyObviousZeros /@
   (cAlpPerpPara /. TocAlp /. ToTheory);
cAlpParaPerpTheory = KeepOnlyObviousZeros /@
   (cAlpParaPerp /. TocAlp /. ToTheory);
cAlpParaParaTheory = KeepOnlyObviousZeros /@
   (cAlpParaPara /. TocAlp /. ToTheory);
cAlpDetTheory = KeepOnlyObviousZeros /@ (cAlpDeterminants /. TocAlp /. ToTheory);
AlpPerpPerpTheory = KeepOnlyObviousZeros /@ (AlpPerpPerp /. ToAlp /. ToTheory);
AlpPerpParaTheory = KeepOnlyObviousZeros /@ (AlpPerpPara /. ToAlp /. ToTheory);
AlpParaPerpTheory = KeepOnlyObviousZeros /@ (AlpParaPerp /. ToAlp /. ToTheory);
AlpParaParaTheory = KeepOnlyObviousZeros /@ (AlpParaPara /. ToAlp /. ToTheory);
AlpDetTheory = KeepOnlyObviousZeros /@ (AlpDeterminants /. ToAlp /. ToTheory);
cBetPerpPerpTheory =
 KeepOnlyObviousZeros /@ (cBetPerpPerp /. TocBet /. ToTheory);
cBetPerpParaTheory = KeepOnlyObviousZeros /@
   (cBetPerpPara /. TocBet /. ToTheory);
cBetParaPerpTheory = KeepOnlyObviousZeros /@
   (cBetParaPerp /. TocBet /. ToTheory);
cBetParaParaTheory = KeepOnlyObviousZeros /@
   (cBetParaPara /. TocBet /. ToTheory);
cBetDetTheory = KeepOnlyObviousZeros /@ (cBetDeterminants /. TocBet /. ToTheory);
BetPerpPerpTheory = KeepOnlyObviousZeros /@ (BetPerpPerp /. ToBet /. ToTheory);
BetPerpParaTheory = KeepOnlyObviousZeros /@ (BetPerpPara /. ToBet /. ToTheory);
BetParaPerpTheory = KeepOnlyObviousZeros /@ (BetParaPerp /. ToBet /. ToTheory);
BetParaParaTheory = KeepOnlyObviousZeros /@ (BetParaPara /. ToBet /. ToTheory);
BetDetTheory = KeepOnlyObviousZeros /@ (BetDeterminants /. ToBet /. ToTheory);

(*We construct the rule which sends the freedom coefficients to the shell*)
ShellFreedomsActivate = {};
For[ii = 1, ii < 7, ii++,
 If[cAlpPerpPerpTheory[[ii]] cAlpPerpParaTheory[[ii]]
     cAlpParaPerpTheory[[ii]] cAlpParaParaTheory[[ii]] == 0,
   {AppendTo[ShellFreedomsActivate, Evaluate[ToExpression[
       "ShellPara" <> ToString[ASectorNames[[ii]]] <> "->1"]]],
    AppendTo[ShellFreedomsActivate, Evaluate[ToExpression[
       "ShellPerp" <> ToString[ASectorNames[[ii]]] <> "->1"]]],
```

```
          AppendTo[ShellFreedomsActivate, Evaluate[ToExpression[
             "ShellSing" <> ToString[ASectorNames[[ii]]] <> "->1"]]],
          If[AlpPerpPerpTheory[[ii]] == 0,
           AppendTo[ShellFreedomsActivate, Evaluate[
             ToExpression["ShellOrig" <> ToString[ASectorNames[[ii]]] <> "->0"]]],
           AppendTo[ShellFreedomsActivate, Evaluate[ToExpression[
              "ShellOrig" <> ToString[ASectorNames[[ii]]] <> "->1"]]]]},
         If[cAlpDetTheory[[ii]] == 0,
          {AppendTo[ShellFreedomsActivate, Evaluate[
             ToExpression["ShellPara" <> ToString[ASectorNames[[ii]]] <> "->0"]]],
           AppendTo[ShellFreedomsActivate, Evaluate[ToExpression[
              "ShellSing" <> ToString[ASectorNames[[ii]]] <> "->1"]]],
           If[AlpPerpPerpTheory[[ii]] == 0,
            {AppendTo[ShellFreedomsActivate, Evaluate[
               ToExpression["ShellOrig" <> ToString[ASectorNames[[ii]]] <> "->0"]]],
             AppendTo[ShellFreedomsActivate, Evaluate[ToExpression[
                "ShellPerp" <> ToString[ASectorNames[[ii]]] <> "->1"]]]},
            {AppendTo[ShellFreedomsActivate, Evaluate[ToExpression[
                "ShellPerp" <> ToString[ASectorNames[[ii]]] <> "->0"]]],
             AppendTo[ShellFreedomsActivate, Evaluate[ToExpression[
                "ShellOrig" <> ToString[ASectorNames[[ii]]] <> "->1"]]]}]},
          {AppendTo[ShellFreedomsActivate, Evaluate[ToExpression[
              "ShellPara" <> ToString[ASectorNames[[ii]]] <> "->1"]]],
           AppendTo[ShellFreedomsActivate, Evaluate[ToExpression[
              "ShellPerp" <> ToString[ASectorNames[[ii]]] <> "->1"]]],
           If[AlpPerpPerpTheory[[ii]] == 0,
            {AppendTo[ShellFreedomsActivate, Evaluate[
               ToExpression["ShellOrig" <> ToString[ASectorNames[[ii]]] <> "->0"]]],
             AppendTo[ShellFreedomsActivate, Evaluate[ToExpression[
                "ShellSing" <> ToString[ASectorNames[[ii]]] <> "->1"]]]},
            {AppendTo[ShellFreedomsActivate, Evaluate[ToExpression[
                "ShellSing" <> ToString[ASectorNames[[ii]]] <> "->0"]]],
             AppendTo[ShellFreedomsActivate, Evaluate[ToExpression[
                "ShellOrig" <> ToString[ASectorNames[[ii]]] <> "->1"]]]}]}]]];
      ];
```

Calculate $\hat{T}$, $\hat{R}$ shell

```
In[*]:= DefTensor[RPShellPara[-a, -b, -c, -d],
          M4, {Antisymmetric[{-a, -b}], Antisymmetric[{-c, -d}]},
          OrthogonalTo → {V[a], V[b], V[c], V[d]}];
        DefTensor[RPShellPerp[-a, -b, -c], M4, Antisymmetric[{-b, -c}],
          OrthogonalTo → {V[a], V[b], V[c]}];
```

```
DefFieldStrengthShell[ShellFreedomsActivate_] :=
  Module[{TPShellDefinition, RPShellParaDefinition, RPShellPerpDefinition,
    RPShellDefinition, RPShellParaActivate, RPShellPerpActivate,
    RPShellParaPerpActivate, TPShellActivate, RPShellActivate},

   MakeDefInfo[DefFieldStrengthShell,
    StrengthPShellToStrengthPO3, {"field strength shell", ""}];

   TPShellDefinition = ShellParaB1p V[-a] TP1p[-b, -c] +
        - (1/6) ShellParaB0m PT0m[-a, -b, -c] TP0m[] +
         ShellParaB1m Antisymmetrize[-PPara[-a, -b] TP1m[-c], {-b, -c}] +
        (4/3) ShellParaB2m TP2m[-b, -c, -a] /. ShellFreedomsActivate /.
      PO3TActivate /. PADMActivate // ToCanonical;

   RPShellParaDefinition = - (1/6) ShellParaA0p
      (PPara[-a, -d] PPara[-b, -c] - PPara[-a, -c] PPara[-b, -d]) RP0p[] -
     ShellParaA1p (PPara[-b, -d] RP1p[-a, -c] - PPara[-b, -c] RP1p[-a, -d] -
        PPara[-a, -d] RP1p[-b, -c] + PPara[-a, -c] RP1p[-b, -d]) +
     ShellParaA2p (PPara[-b, -d] RP2p[-a, -c] - PPara[-b, -c] RP2p[-a, -d] -
        PPara[-a, -d] RP2p[-b, -c] + PPara[-a, -c] RP2p[-b, -d]);
   RPShellPerpDefinition = - (1/6) ShellParaA0m PR0m[-a, -b, -c] RP0m[] +
     ShellParaA1m Antisymmetrize[-PPara[-a, -b] RP1m[-c], {-b, -c}] +
     (4/3) ShellParaA2m RP2m[-b, -c, -a];

   RPShellParaActivate = MakeRule[{RPShellPara[-a, -b, -c, -d],
      Evaluate[RPShellParaDefinition]}, MetricOn → All, ContractMetrics → True];
   RPShellPerpActivate = MakeRule[{RPShellPerp[-a, -b, -c],
      Evaluate[RPShellPerpDefinition]}, MetricOn → All, ContractMetrics → True];
   RPShellParaPerpActivate = Join[RPShellParaActivate, RPShellPerpActivate];

   RPShellDefinition =
    RPShellPara[-a, -b, -c, -d] + 2 Antisymmetrize[V[-a] RPShellPerp[-b, -c, -d],
          {-a, -b}] /. RPShellParaPerpActivate /.
       ShellFreedomsActivate /. PO3RActivate /. PADMActivate // ToCanonical;

   TPShellDefinition =
    TPShellDefinition // CollectTensors // ScreenDollarIndices // CollectTensors;
   RPShellDefinition = RPShellDefinition // CollectTensors //
      ScreenDollarIndices // CollectTensors;

   TPShellActivate = MakeRule[{TP[-a, -b, -c], Evaluate[TPShellDefinition]},
     MetricOn → All, ContractMetrics → True];
   RPShellActivate = MakeRule[{RP[-a, -b, -c, -d], Evaluate[RPShellDefinition]},
```

```
    MetricOn → All, ContractMetrics → True];
   StrengthPShellToStrengthPO3 = Join[TPShellActivate, RPShellActivate];
  ];
```

Calculate $\hat{\pi}$ bJ^P, $\hat{\pi}$ AJ^P shell

```
In[•]:= DefTensor[PerpBComplement[-i, -k], M4];
DefTensor[OrigBComplement[-i, -k], M4];
DefTensor[SingBComplement[-i, -k], M4];
DefTensor[PerpAComplement[-i, -j, -k], M4, Antisymmetric[{-i, -j}]];
DefTensor[OrigAComplement[-i, -j, -k], M4, Antisymmetric[{-i, -j}]];
DefTensor[SingAComplement[-i, -j, -k], M4, Antisymmetric[{-i, -j}]];

DefMomentaShell[ShellFreedomsActivate_, ToTheory_] :=
  Module[{OrigBComplementDefinition, PerpBComplementDefinition,
    SingBComplementDefinition, OrigAComplementDefinition,
    PerpAComplementDefinition, SingAComplementDefinition,
    PerpBComplementActivate, OrigBComplementActivate, SingBComplementActivate,
    PerpAComplementActivate, OrigAComplementActivate, SingAComplementActivate,
    OnShellBLambdaDefinition, OnShellALambdaDefinition,
    OnShellBLambdaActivate, OnShellALambdaActivate, ComplementActivate, tmp},

   MakeDefInfo[DefFieldStrengthShell,
    StrengthPShellToStrengthPO3, {"field strength shell", ""}];

   OrigBComplementDefinition =
    Evaluate[J[] 4 V[g] B[-k, -o] G3[o, -z] H[h, z] (Bet1 PT1[-i, -g, -h, a, c, d] + Bet2
            PT2[-i, -g, -h, a, c, d] + Bet3 PT3[-i, -g, -h, a, c, d])
         PPara[-c, x] PPara[-d, y] T[-a, -x, -y] + 2 J[] V[g] B[-k, -o]
         G3[o, -z] H[h, z] (cBet1 PT1[-i, -g, -h, a, c, d] + cBet2
            PT2[-i, -g, -h, a, c, d] + cBet3 PT3[-i, -g, -h, a, c, d])
         PPara[-c, m] PPara[-d, n] TLambda[-a, -m, -n] +
        2 J[] V[g] B[-k, -o] G3[o, -z] H[h, z] (cBet1 PT1[-i, -g, -h, a, c, d] +
           cBet2 PT2[-i, -g, -h, a, c, d] + cBet3 PT3[-i, -g, -h, a, c, d])
         (PPerp[-c, m] PPara[-d, n] TLambda[-a, -m, -n] + PPara[-c, m]
            PPerp[-d, n] TLambda[-a, -m, -n]) /. ToTheory /. PActivate /.
      PADMActivate // ToCanonical // ContractMetric // CollectTensors];

   PerpBComplementDefinition =
    Evaluate[2 J[] V[g] B[-k, -o] G3[o, -z] H[h, z] (cBet1 PT1[-i, -g, -h, a, c, d] +
           cBet2 PT2[-i, -g, -h, a, c, d] + cBet3 PT3[-i, -g, -h, a, c, d])
         PPara[-c, m] PPara[-d, n] TLambda[-a, -m, -n] +
        2 J[] V[g] B[-k, -o] G3[o, -z] H[h, z] (cBet1 PT1[-i, -g, -h, a, c, d] +
```

```
          cBet2 PT2[-i, -g, -h, a, c, d] + cBet3 PT3[-i, -g, -h, a, c, d])
        (PPerp[-c, m] PPara[-d, n] TLambda[-a, -m, -n] + PPara[-c, m]
          PPerp[-d, n] TLambda[-a, -m, -n]) /. ToTheory /. PActivate /.
    PADMActivate // ToCanonical // ContractMetric // CollectTensors];

SingBComplementDefinition =
 Evaluate[-J[] 4 V[g] B[-k, -o] G3[o, -z] H[h, z] (cBet1 PT1[-i, -g, -h, a, c, d] +
          cBet2 PT2[-i, -g, -h, a, c, d] + cBet3 PT3[-i, -g, -h, a, c, d])
        PPara[-c, x] PPara[-d, y] T[-a, -x, -y] /. ToTheory /. PActivate /.
    PADMActivate // ToCanonical // ContractMetric // CollectTensors];

OrigAComplementDefinition =
 Evaluate[-2 Alp0 J[] Antisymmetrize[V[-i] PPara[-j, -k], {-i, -j}] +
        J[] 8 V[g] B[-k, -o] G3[o, -z] H[h, z] (Alp1 PR1[-i, -j, -g, -h, a, b,
             c, d] + Alp2 PR2[-i, -j, -g, -h, a, b, c, d] + Alp3 PR3[-i, -j,
             -g, -h, a, b, c, d] + Alp4 PR4[-i, -j, -g, -h, a, b, c, d] + Alp5
            PR5[-i, -j, -g, -h, a, b, c, d] + Alp6 PR6[-i, -j, -g, -h, a, b, c, d])
         PPara[-c, x] PPara[-d, y] R[-a, -b, -x, -y] + 4 J[] V[g] B[-k, -o]
         G3[o, -z] H[h, z] (cAlp1 PR1[-i, -j, -g, -h, a, b, c, d] + cAlp2
            PR2[-i, -j, -g, -h, a, b, c, d] + cAlp3 PR3[-i, -j, -g, -h, a,
             b, c, d] + cAlp4 PR4[-i, -j, -g, -h, a, b, c, d] + cAlp5 PR5[-i,
             -j, -g, -h, a, b, c, d] + cAlp6 PR6[-i, -j, -g, -h, a, b, c, d])
         PPara[-c, m] PPara[-d, n] RLambda[-a, -b, -m, -n] +
        4 J[] V[g] B[-k, -o] G3[o, -z] H[h, z] (cAlp1 PR1[-i, -j, -g, -h, a,
             b, c, d] + cAlp2 PR2[-i, -j, -g, -h, a, b, c, d] + cAlp3 PR3[-i, -j,
             -g, -h, a, b, c, d] + cAlp4 PR4[-i, -j, -g, -h, a, b, c, d] + cAlp5 PR5[
             -i, -j, -g, -h, a, b, c, d] + cAlp6 PR6[-i, -j, -g, -h, a, b, c, d])
        (PPerp[-c, m] PPara[-d, n] RLambda[-a, -b, -m, -n] + PPara[-c, m]
            PPerp[-d, n] RLambda[-a, -b, -m, -n]) /. ToTheory /. PActivate /.
    PADMActivate // ToCanonical // ContractMetric // CollectTensors];

PerpAComplementDefinition =
 Evaluate[4 J[] V[g] B[-k, -o] G3[o, -z] H[h, z] (cAlp1 PR1[-i, -j, -g, -h, a, b, c,
             d] + cAlp2 PR2[-i, -j, -g, -h, a, b, c, d] + cAlp3 PR3[-i, -j, -g,
             -h, a, b, c, d] + cAlp4 PR4[-i, -j, -g, -h, a, b, c, d] + cAlp5 PR5[
             -i, -j, -g, -h, a, b, c, d] + cAlp6 PR6[-i, -j, -g, -h, a, b, c, d])
         PPara[-c, m] PPara[-d, n] RLambda[-a, -b, -m, -n] +
        4 J[] V[g] B[-k, -o] G3[o, -z] H[h, z] (cAlp1 PR1[-i, -j, -g, -h, a,
             b, c, d] + cAlp2 PR2[-i, -j, -g, -h, a, b, c, d] + cAlp3 PR3[-i, -j,
             -g, -h, a, b, c, d] + cAlp4 PR4[-i, -j, -g, -h, a, b, c, d] + cAlp5 PR5[
             -i, -j, -g, -h, a, b, c, d] + cAlp6 PR6[-i, -j, -g, -h, a, b, c, d])
        (PPerp[-c, m] PPara[-d, n] RLambda[-a, -b, -m, -n] + PPara[-c, m]
            PPerp[-d, n] RLambda[-a, -b, -m, -n]) /. ToTheory /. PActivate /.
```

```
      PADMActivate // ToCanonical // ContractMetric // CollectTensors];

Evaluate[SingAComplementDefinition =
  -J[] 8 V[g] B[-k, -o] G3[o, -z] H[h, z] (cAlp1 PR1[-i, -j, -g, -h, a, b, c, d] +
          cAlp2 PR2[-i, -j, -g, -h, a, b, c, d] + cAlp3 PR3[-i, -j, -g, -h, a, b,
            c, d] + cAlp4 PR4[-i, -j, -g, -h, a, b, c, d] + cAlp5 PR5[-i, -j, -g,
            -h, a, b, c, d] + cAlp6 PR6[-i, -j, -g, -h, a, b, c, d]) PPara[-c, x]
        PPara[-d, y] R[-a, -b, -x, -y] /. ToTheory /. PActivate /.
      PADMActivate // ToCanonical // ContractMetric // CollectTensors];

PerpBComplementDefinition =
 PerpBComplementDefinition /. HG3BExpandLazy // ToCanonical //
   ContractMetric // CollectTensors;
OrigBComplementDefinition = OrigBComplementDefinition /. HG3BExpandLazy //
    ToCanonical // ContractMetric // CollectTensors;
SingBComplementDefinition = SingBComplementDefinition /. HG3BExpandLazy //
    ToCanonical // ContractMetric // CollectTensors;
PerpBComplementDefinition = PerpBComplementDefinition /. ExpandStrengths //
    ToCanonical // ContractMetric // CollectTensors;
OrigBComplementDefinition = OrigBComplementDefinition /. ExpandStrengths //
    ToCanonical // ContractMetric // CollectTensors;
SingBComplementDefinition = SingBComplementDefinition /. ExpandStrengths //
    ToCanonical // ContractMetric // CollectTensors;

PerpAComplementDefinition =
 PerpAComplementDefinition /. HG3BExpandLazy // ToCanonical //
   ContractMetric // CollectTensors;
OrigAComplementDefinition = OrigAComplementDefinition /. HG3BExpandLazy //
    ToCanonical // ContractMetric // CollectTensors;
SingAComplementDefinition = SingAComplementDefinition /. HG3BExpandLazy //
    ToCanonical // ContractMetric // CollectTensors;
PerpAComplementDefinition = PerpAComplementDefinition /. ExpandStrengths //
    ToCanonical // ContractMetric // CollectTensors;
OrigAComplementDefinition = OrigAComplementDefinition /. ExpandStrengths //
    ToCanonical // ContractMetric // CollectTensors;
SingAComplementDefinition = SingAComplementDefinition /. ExpandStrengths //
    ToCanonical // ContractMetric // CollectTensors;

PerpBComplementActivate =
 MakeRule[{PerpBComplement[-i, -k], Evaluate[PerpBComplementDefinition]},
  MetricOn → All, ContractMetrics → True];
OrigBComplementActivate = MakeRule[{OrigBComplement[-i, -k], Evaluate[
    OrigBComplementDefinition]}, MetricOn → All, ContractMetrics → True];
```

```
SingBComplementActivate = MakeRule[{SingBComplement[-i, -k], Evaluate[
     SingBComplementDefinition]}, MetricOn → All, ContractMetrics → True];
PerpAComplementActivate = MakeRule[{PerpAComplement[-i, -j, -k], Evaluate[
     PerpAComplementDefinition]}, MetricOn → All, ContractMetrics → True];
OrigAComplementActivate = MakeRule[{OrigAComplement[-i, -j, -k], Evaluate[
     OrigAComplementDefinition]}, MetricOn → All, ContractMetrics → True];
SingAComplementActivate = MakeRule[{SingAComplement[-i, -j, -k], Evaluate[
     SingAComplementDefinition]}, MetricOn → All, ContractMetrics → True];
ComplementActivate = Join[PerpBComplementActivate, OrigBComplementActivate,
   SingBComplementActivate, PerpAComplementActivate,
   OrigAComplementActivate, SingAComplementActivate];

OnShellBLambdaDefinition = (ShellOrigB0p ShellPerpB0p PB0pT[-n, -m, a, c] +
      ShellOrigB1p ShellPerpB1p ShellSingB1p PB1pT[-n, -m, a, c] +
      ShellOrigB2p ShellPerpB2p PB2pT[-n, -m, a, c] +
      ShellOrigB1m ShellPerpB1m ShellSingB1m PB1mT[-n, -m, a, c]) BPiP[-a, -c] +
   ((1 - ShellOrigB0p) PB0pT[-n, -m, i, k] +
      (1 - ShellOrigB1p) PB1pT[-n, -m, i, k] +
      (1 - ShellOrigB2p) PB2pT[-n, -m, i, k] +
      (1 - ShellOrigB1m) PB1mT[-n, -m, i, k]) OrigBComplement[-i, -k] +
   ((1 - ShellPerpB0p) PB0pT[-n, -m, i, k] +
      (1 - ShellPerpB1p) PB1pT[-n, -m, i, k] +
      (1 - ShellPerpB2p) PB2pT[-n, -m, i, k] +
      (1 - ShellPerpB1m) PB1mT[-n, -m, i, k]) PerpBComplement[-i, -k] +
   ((1 - ShellSingB1p) PB1pT[-n, -m, i, k] +
      (1 - ShellSingB1m) PB1mT[-n, -m, i, k]) OrigBComplement[-i, -k] +
   ((1 - ShellSingB1p) (BetPerpPerp1p / cBetPerpPerp1p) PB1pT[-n, -m, i, k] +
      (1 - ShellSingB1m) (BetPerpPerp1m / cBetPerpPerp1m) PB1mT[-n, -m, i, k])
    SingBComplement[-i, -k];
Print[1];
OnShellALambdaDefinition =
  (ShellOrigA0p ShellPerpA0p ShellSingA0p PA0pT[-n, -m, -o, a, b, c] +
      ShellOrigA1p ShellPerpA1p ShellSingA1p PA1pT[-n, -m, -o, a, b, c] +
      ShellOrigA2p ShellPerpA2p ShellSingA2p PA2pT[-n, -m, -o, a, b, c] +
      ShellOrigA0m ShellPerpA0m ShellSingA0m PA0mT[-n, -m, -o, a, b, c] +
      ShellOrigA1m ShellPerpA1m ShellSingA1m PA1mT[-n, -m, -o, a, b, c] +
      ShellOrigA2m ShellPerpA2m ShellSingA2m PA2mT[-n, -m, -o, a, b, c])
    APiP[-a, -b, -c] +
   ((1 - ShellOrigA0p) PA0pT[-n, -m, -o, i, j, k] +
      (1 - ShellOrigA1p) PA1pT[-n, -m, -o, i, j, k] +
      (1 - ShellOrigA2p) PA2pT[-n, -m, -o, i, j, k] +
      (1 - ShellOrigA0m) PA0mT[-n, -m, -o, i, j, k] +
      (1 - ShellOrigA1m) PA1mT[-n, -m, -o, i, j, k] +
```

```
      (1 - ShellOrigA2m) PA2mT[-n, -m, -o, i, j, k])
    OrigAComplement[-i, -j, -k] +
   ((1 - ShellPerpA0p) PA0pT[-n, -m, -o, i, j, k] +
      (1 - ShellPerpA1p) PA1pT[-n, -m, -o, i, j, k] +
      (1 - ShellPerpA2p) PA2pT[-n, -m, -o, i, j, k] +
      (1 - ShellPerpA0m) PA0mT[-n, -m, -o, i, j, k] +
      (1 - ShellPerpA1m) PA1mT[-n, -m, -o, i, j, k] +
      (1 - ShellPerpA2m) PA2mT[-n, -m, -o, i, j, k])
    PerpAComplement[-i, -j, -k] +
   ((1 - ShellSingA0p) PA0pT[-n, -m, -o, i, j, k] +
      (1 - ShellSingA1p) PA1pT[-n, -m, -o, i, j, k] +
      (1 - ShellSingA2p) PA2pT[-n, -m, -o, i, j, k] +
      (1 - ShellSingA0m) PA0mT[-n, -m, -o, i, j, k] +
      (1 - ShellSingA1m) PA1mT[-n, -m, -o, i, j, k] +
      (1 - ShellSingA2m) PA2mT[-n, -m, -o, i, j, k])
    OrigAComplement[-i, -j, -k] +
   ((1 - ShellSingA0p) (AlpPerpPerp0p / cAlpPerpPerp0p) PA0pT[-n, -m, -o, i, j, k] +
      (1 - ShellSingA1p)
       (AlpPerpPerp1p / cAlpPerpPerp1p) PA1pT[-n, -m, -o, i, j, k] +
      (1 - ShellSingA2p) (AlpPerpPerp2p / cAlpPerpPerp2p)
       PA2pT[-n, -m, -o, i, j, k] +
      (1 - ShellSingA0m) (AlpPerpPerp0m / cAlpPerpPerp0m)
       PA0mT[-n, -m, -o, i, j, k] +
      (1 - ShellSingA1m) (AlpPerpPerp1m / cAlpPerpPerp1m)
       PA1mT[-n, -m, -o, i, j, k] +
      (1 - ShellSingA2m) (AlpPerpPerp2m / cAlpPerpPerp2m)
       PA2mT[-n, -m, -o, i, j, k]) SingAComplement[-i, -j, -k];

OnShellALambdaDefinition =
  OnShellALambdaDefinition /. ShellFreedomsActivate /. ToAlp /. TocAlp /.
    ToTheory // ToNewCanonical;
OnShellALambdaDefinition = OnShellALambdaDefinition /. NewPO3TActivate //
   ToNewCanonical;
OnShellALambdaDefinition = OnShellALambdaDefinition /.
    ComplementActivate // ToNewCanonical;

OnShellBLambdaDefinition =
  OnShellBLambdaDefinition /. ShellFreedomsActivate /. ToBet /. TocBet /.
    ToTheory // ToNewCanonical;
OnShellBLambdaDefinition = OnShellBLambdaDefinition /. NewPO3TActivate //
    ToCanonical // ContractMetric;
OnShellBLambdaDefinition = OnShellBLambdaDefinition /. ComplementActivate //
    ToCanonical // ContractMetric;
```

```
  Print[OnShellBLambdaDefinition];
  Print[OnShellALambdaDefinition];
  Print[ToTheory];
  Print[ShellFreedomsActivate];

  OnShellBLambdaActivate =
   MakeRule[{BPiP[-n, -m], Evaluate[OnShellBLambdaDefinition]},
    MetricOn → All, ContractMetrics → True];
  OnShellALambdaActivate = MakeRule[{APiP[-n, -m, -o], Evaluate[
      OnShellALambdaDefinition]}, MetricOn → All, ContractMetrics → True];
  PiPShellToPiPPO3 = Join[OnShellBLambdaActivate, OnShellALambdaActivate];
 ];
```

Special 2⁻ rules

```
In[∘]:= (*
ManualA2m=MakeRule[{PiPA2m[-a,-b,-c],-16Alp6 J[]RP2m[-a,-b,-c]},
   MetricOn→All,ContractMetrics→True];
 (*It is essential that we update this for Lambdas*)
*)(*this version for case 16*)
(**)
ManualA2m = MakeRule[{PiPA2m[-a, -b, -c], 0}, MetricOn → All, ContractMetrics → True];
(*It is essential that we update this for Lambdas*)
(**)(*this version for case 26*)
(*comment out for simple Spin1⁺*)
```

Linearisation

```
In[•]:= ToOrderCanonical[expr_, order_] := Module[{res},
    Print[Style["To order...", Green, 10]];
    res = expr;
    Switch[order, 0, {
      res = res /. ToOrderRules;
      res = CollectConstants[res, Prt];
      res = res /. {Prt → 0};
     }, 1, {
      res = res /. ToOrderRules;
      res = CollectConstants[res, Prt];
      res = res /.
        {Prt^2 → 0, Prt^3 → 0, Prt^4 → 0, Prt^5 → 0, Prt^6 → 0, Prt^7 → 0, Prt^8 → 0,
         Prt^9 → 0, Prt^10 → 0, Prt^11 → 0, Prt^12 → 0, Prt^13 → 0, Prt^14 → 0};
      res = res /. {Prt → 1};
     }, Infinity, {}];
    res = res // ToNewCanonical;
    res];
```

Nester form $\hat{\pi}\, b, \hat{\pi}\, A, \mathbb{D}\,\hat{\pi}\, b, \mathbb{D}\,\hat{\pi}\, A$

```
In[•]:= Options[ToO3] = {"ToShell" → True, "Order" → Infinity};
  ToO3[x_, OptionsPattern[]] := Module[{res}, res = x;
    Print[Style["To O3...", Blue, 10]];
    (*res=res/.CDPiToCDPiP;*) (*res=res/.CDPiToCDPiPHard;*)
    (*this and the non-Hard line above are new,
    I'm not sure why I didn't need these before?*)res = res // NoScalar /. PiToPiP;
    (*not clear how necessary this is!*)res = res /. PiToPiP;
    res = ToOrderCanonical[res, OptionValue["Order"]];
    If[OptionValue["ToShell"], res = res /. PiPShellToPiPPO3];
    Print[Style["c", Blue, 10]];
    Print[res];
    res = res // ToNewCanonical;
    res = res /. ToStrengths;
    res = ToOrderCanonical[res, OptionValue["Order"]];
    res = res /. StrengthDecompose;
    res = res /. StrengthLambdaDecompose;
    res = res // ToNewCanonical;
    If[OptionValue["ToShell"], res = res /. StrengthPShellToStrengthPO3];
    res = res /. StrengthPToStrengthPO3;
    res = res /. StrengthPerpToStrengthPerpO3;
    res = res /. StrengthLambdaPToStrengthLambdaPO3;
```

```
    res = res /. StrengthLambdaPerpToStrengthLambdaPerpO3;
    res = res // ToNewCanonical;
    res = res /. PiPToPiPO3;
    res = res // ToNewCanonical;
    (**)If[OptionValue["ToShell"], res = res /. ManualA2m];
    (**)res = ToOrderCanonical[res, OptionValue["Order"]];
    res];

DefO3MomentaShell[] :=
  Module[{temp, CDBPiPToCDBPiPO3, CDAPiPToCDAPiPO3, TheoryCDBPiPToCDBPiPO3,
     TheoryBPiPToBPiPO3, TheoryCDAPiPToCDAPiPO3, TheoryAPiPToAPiPO3},

    MakeDefInfo[DefO3MomentaShell,
     undefinedvariable, {"field strength shell", ""}];

    temp = APiP[-a, -b, -c] // ToO3;
    Print[temp];
    TheoryAPiPToAPiPO3 = MakeRule[
      {APiP[-a, -b, -c], Evaluate[temp]}, MetricOn → All, ContractMetrics → True];
    temp = CD[-z][temp] // ToNewCanonical;
    Print[temp];
    TheoryCDAPiPToCDAPiPO3 = MakeRule[{CD[-z][APiP[-a, -b, -c]], Evaluate[temp]},
      MetricOn → All, ContractMetrics → True];
    temp = BPiP[-a, -b] // ToO3;
    Print[temp];
    TheoryBPiPToBPiPO3 = MakeRule[
      {BPiP[-a, -b], Evaluate[temp]}, MetricOn → All, ContractMetrics → True];
    temp = CD[-z][temp] // ToNewCanonical;
    Print[temp];
    TheoryCDBPiPToCDBPiPO3 = MakeRule[{CD[-z][BPiP[-a, -b]], Evaluate[temp]},
      MetricOn → All, ContractMetrics → True];
    TheoryCDPiPToCDPiPO3 = Join[TheoryCDAPiPToCDAPiPO3, TheoryCDBPiPToCDBPiPO3];
    TheoryPiPToPiPO3 = Join[TheoryAPiPToAPiPO3, TheoryBPiPToBPiPO3];

    temp = ToO3[APiP[-a, -b, -c], "ToShell" → False];
    temp = CD[-z][temp] // ToNewCanonical;
    CDAPiPToCDAPiPO3 = MakeRule[{CD[-z][APiP[-a, -b, -c]], Evaluate[temp]},
      MetricOn → All, ContractMetrics → True];
    temp = ToO3[BPiP[-a, -b], "ToShell" → False];
    temp = CD[-z][temp] // ToNewCanonical;
    CDBPiPToCDBPiPO3 = MakeRule[{CD[-z][BPiP[-a, -b]], Evaluate[temp]},
      MetricOn → All, ContractMetrics → True];
    CDPiPToCDPiPO3 = Join[CDAPiPToCDAPiPO3, CDBPiPToCDBPiPO3];
  ];
```

```
In[•]:= DumpSave[NotebookDirectory[] <> "mx_cache/HiGGS.mx", "Global`"];
DumpSave[NotebookDirectory[] <> "HiGGS/HiGGS_global.mx", "Global`"];
(*NotebookEvaluate[NotebookDirectory[]<>"testing_notebook.nb"]*)
Quit[];
```

Nester and basic forms

Nester form general

```
In[•]:= Options[TotalToO3] = {"ToShell" → True, "Order" → Infinity};
TotalToO3[x_, OptionsPattern[]] := Module[{res}, res = x;
   Print[Style["Total to O3...", Blue, 10]];
   Print[res];
   (**)res = res /. CDPiToCDPiP; (**)
   Print[res];
   (**)res = res /. CDPiToCDPiPHard;
   (**)(*this and the non-Hard line above are new,
   I'm not sure why I didn't need these before?*)
   res = res // NoScalar /. PiToPiP;
   (*not clear how necessary this is!*)
   res = res /. PiToPiP;
   res = ToOrderCanonical[res, OptionValue["Order"]];
   If[OptionValue["ToShell"],
    res = res /. TheoryCDPiPToCDPiPO3, res = res /. CDPiPToCDPiPO3];
   res = res // ToNewCanonical;
   If[OptionValue["ToShell"],
    res = res /. TheoryPiPToPiPO3, res = res /. PiPToPiPO3];
   res = res // ToNewCanonical;
   res =
    ToO3[res, "ToShell" → OptionValue["ToShell"], "Order" -> OptionValue["Order"]];
   res = ToOrderCanonical[res, OptionValue["Order"]];
   (*res=res//ToNewCanonical;*)
   res];

CDBToDJDV[x_] := Module[{res}, res = x;
   Print[Style["CDB to DV and DJ...", Blue, 10]];
   res = res /. G3HCDBToDJ;
   res = res // ToNewCanonical;
   res = res /. G3VCDBToG3DV;
   res = res // ToNewCanonical;
   res = res /. CDBCommute;
   res = res // ToNewCanonical;
```

```
    res = res /. G3HCDBToDJ;
    res = res // ToNewCanonical;
    res = res /. G3VCDBToG3DV;
    res = res // ToNewCanonical;
    res = res /. HExpand;
    res = res // ToNewCanonical;
    res = res /. G3HCDBToDJ;
    res = res // ToNewCanonical;
    res = res /. G3VCDBToG3DV;
    res = res // ToNewCanonical;
    res = res /. CDBCommute;
    res = res // ToNewCanonical;
    res = res /. G3HCDBToDJ;
    res = res // ToNewCanonical;
    res = res /. G3VCDBToG3DV;
    res = res // ToNewCanonical;
    res];

CDToD[x_] := Module[{res}, res = x;
    Print[Style["CD to D...", Blue, 10]];
    res = res /. DGrandActivate;
    res = res /. DpGrandActivate;
    res = res /. DpVExpand; (*this is new!*)
    res = res // ToNewCanonical;
    res = res /. epsilonGVToEps;
    res = res /. epsilonGToEpsV;
    res = res // ToNewCanonical;
    res];

CollapseA[x_] := Module[{res}, res = x;
    Print[Style["Collapse A...", Blue, 10]];
    res = res /. CDAToCDAInert;
    res = res /. AExpand;
    res = res /. G3HExpand;
    res = res // ToNewCanonical;
    res = res /. HG3VCDAToHVCDA;
    res = res // ToNewCanonical;
    res = res /. HG3VAToHVA;
    res = res // ToNewCanonical;
    res = res /. G3HExpand;
    res = res // ToNewCanonical;
    res = res /. HExpand;
    res = res // ToNewCanonical;
    res = res /. CDAInertToCDA;
```

```
    res = res // ToNewCanonical;
    res = res /. HG3BExpand;
    (*to deal with the strange combination of A epsilon which cancels*)
    res = res /. G3HExpand;
    res = res /. HEpsToHG3Eps;
    res = res // ToNewCanonical;
    res = res /. AHEpsExpand;
    res = res // ToNewCanonical;
    res = res /. EpsEpsExpand;
    res = res // ToNewCanonical;
    (*finished dealing with this combination*)
    res];

Options[PreSimplify] = {"Hard" → False, "Order" → Infinity};
PreSimplify[x_, OptionsPattern[]] := Module[{res}, res = x;
    Print[Style["Pre-simplifying...", Blue, 10]];
    (*res=res//ToNewCanonical;*)(*should re-test after implementing this*)
    res = ToOrderCanonical[res, OptionValue["Order"]];
    If[OptionValue["Hard"], res = res /. HExpand];
    res = res // ToNewCanonical;
    res = res /. HG3BExpandLazy;
    res = res // ToNewCanonical;
    res = res /. G3HExpand;
    res = ToOrderCanonical[res, OptionValue["Order"]];
    (*res=res//ToNewCanonical;*)
    res];

Options[ToNesterForm] =
   {"ToShell" → True, "Hard" → False, "Order" → Infinity, "GToFoliG" → True};
ToNesterForm[x_, OptionsPattern[]] := Module[{res}, res = x;
    Print[Style["To Nester form...", Blue, 10]];
    res = res /. PhiActivate // NoScalar;
    res = res /. ChiParaActivate // NoScalar;
    res = res /. ChiPerpActivate // NoScalar;
    res = res /. ChiSingActivate // NoScalar;
    If[OptionValue["ToShell"], res = res /. Theory];
    res =
     PreSimplify[res, "Hard" → OptionValue["Hard"], "Order" → OptionValue["Order"]];
    res = TotalToO3[res, "ToShell" → OptionValue["ToShell"],
      "Order" → OptionValue["Order"]];
    res = res // CDToD;
    res = TotalToO3[res,
      "ToShell" → OptionValue["ToShell"], "Order" → OptionValue["Order"]];
    res = res // CDBToDJDV;
```

```
    res = res // CDToD;
    res = TotalToO3[res,
      "ToShell" → OptionValue["ToShell"], "Order" → OptionValue["Order"]];
    res = res // CollapseA;
    If[OptionValue["GToFoliG"], res = res /. GToFoliG];
    res = res // ToNewCanonical;
    res = res /. CollapseJ;
    res = ToOrderCanonical[res, OptionValue["Order"]];
    (*res=res//ToNewCanonical;*)
    res];
```

ORPHAN

```
In[◦]:= (*
  FlipDer=MakeRule[
     {G3[-a,m]DV[-m,-j]PiPB1m[j],G3[-a,m]B[l,-m]H[-j,n]G3[-n,s]DV[-s,-l]PiPB1m[j]-
       G3[-a,m]B[c,-m]V[-i]TP[i,-c,-j]PiPB1m[j]},
     MetricOn→All,ContractMetrics→True];
  NewFlipTorsion=MakeRule[{DpV[-m,-j]RP2m[-b,-c,m],
      Evaluate[DpV[-j,-m]RP2m[-b,-c,m]-V[-i]TP[i,-m,-j]RP2m[-b,-c,m]/.PADMActivate]},
     MetricOn→All,ContractMetrics→True];
  PostNewFlipTorsion=MakeRule[{DpV[-j,-a]PiPA1p[j,-b],
      Evaluate[DpV[-a,-j]PiPA1p[j,-b]-V[-i]TP[i,-j,-a]PiPA1p[j,-b]/.PADMActivate]},
     MetricOn→All,ContractMetrics→True];
  NeoPostNewFlipTorsion=MakeRule[{DpV[-a,-j]PiPA1p[j,-b],
      Evaluate[DpV[-j,-a]PiPA1p[j,-b]-V[-i]TP[i,-a,-j]PiPA1p[j,-b]/.PADMActivate]},
     MetricOn→All,ContractMetrics→True];
  PostPostNewFlipTorsion=MakeRule[{DpV[-j,-a]PiPA2p[j,-b],
      Evaluate[DpV[-a,-j]PiPA2p[j,-b]-V[-i]TP[i,-j,-a]PiPA2p[j,-b]/.PADMActivate]},
     MetricOn→All,ContractMetrics→True];
  NeoPostPostNewFlipTorsion=MakeRule[{DpV[-a,-j]PiPA2p[j,-b],
      Evaluate[DpV[-j,-a]PiPA2p[j,-b]-V[-i]TP[i,-a,-j]PiPA2p[j,-b]/.PADMActivate]},
     MetricOn→All,ContractMetrics→True];
  AFlipTorsion=MakeRule[{DpV[-a,-j]PiPB1m[j],
      Evaluate[DpV[-j,-a]PiPB1m[j]-V[-i]TP[i,-a,-j]PiPB1m[j]/.PADMActivate]},
     MetricOn→All,ContractMetrics→True];
  BFlipTorsion=MakeRule[{DpV[-j,-a]PiPB1m[j],
      Evaluate[DpV[-a,-j]PiPB1m[j]-V[-i]TP[i,-j,-a]PiPB1m[j]/.PADMActivate]},
     MetricOn→All,ContractMetrics→True];
  TotalOfZero=MakeRule[{G3[-k,i]DRP2m[-i,-a,-b,-c]V[c],
      -G3[-k,i]B[j,-i]DpV[-j,c]RP2m[-a,-b,-c]},MetricOn→All,ContractMetrics→True];
  *)(*scheduled for deletion*)
```

Nester form ϕbJ^P, ϕAJ^P

```
Phis = {PhiB0p[], PhiB1p[-i, -j], PhiB1m[-i], PhiB2p[-i, -j], PhiA0p[], PhiA0m[],
   PhiA1p[-i, -j], PhiA1m[-i], PhiA2p[-i, -j], PhiA2m[-i, -j, -k]};
ChiPerps = {ChiPerpB0p[], ChiPerpB1p[-i, -j], ChiPerpB1m[-i],
   ChiPerpB2p[-i, -j], ChiPerpA0p[], ChiPerpA0m[], ChiPerpA1p[-i, -j],
   ChiPerpA1m[-i], ChiPerpA2p[-i, -j], ChiPerpA2m[-i, -j, -k]};
ChiParas = {ChiParaB0m[], ChiParaB1p[-i, -j], ChiParaB1m[-i],
   ChiParaB2m[-i, -j, -k], ChiParaA0p[], ChiParaA0m[], ChiParaA1p[-i, -j],
   ChiParaA1m[-i], ChiParaA2p[-i, -j], ChiParaA2m[-i, -j, -k]};
ChiSings = {ChiSingB1p[-i, -j], ChiSingB1m[-i], ChiSingA0p[], ChiSingA0m[],
   ChiSingA1p[-i, -j], ChiSingA1m[-i], ChiSingA2p[-i, -j], ChiSingA2m[-i, -j, -k]};
Print["commencing phi evals"];
(*
(*Phis=Phis/.PhiNonCanonicalActivate;*)
Phis=(ToNesterForm[#,"ToShell"→False]//CollectTensors)&/@Phis;
Print[Style["Off-shell Phi functions",Red,20]];
Print/@Phis;
Quit[];
*)
(*
Phis=(ToNesterForm[#,"ToShell"→False]//CollectTensors)&/@Phis;
Phis=Phis/.Theory;
Print[Style["Off-shell Phi functions",Red,20]];
Print/@Phis;
*)
(*
Phis=(ToNesterForm[#,"ToShell"→True]//CollectTensors)&/@Phis;
Print[Style["On-shell Phi functions",Red,20]];
Print/@Phis;
Quit[];
*)
(*
ChiPerps=(ToNesterForm[#,"ToShell"→False]//CollectTensors)&/@ChiPerps;
ChiPerps=ChiPerps/.Theory;
Print[Style["Off-shell perpendicular Chi functions",Red,20]];
Print/@ChiPerps;
ChiParas=(ToNesterForm[#,"ToShell"→False]//CollectTensors)&/@ChiParas;
ChiParas=ChiParas/.Theory;
Print[Style["Off-shell parallel Chi functions",Red,20]];
Print/@ChiParas;
ChiSings=(ToNesterForm[#,"ToShell"→False]//CollectTensors)&/@ChiSings;
ChiSings=ChiSings/.Theory;
```

```
Print[Style["Off-shell singular Chi functions",Red,20]];
Print/@ChiSings;
*)

If[phitonesterformphiToggle,
 NesterFormPhiB0pDefinition = ToNesterForm[PhiB0p[], "ToShell" → False];
 Print[NesterFormPhiB0pDefinition];
 ToNesterFormPhiB0p = MakeRule[{PhiB0p[], Evaluate[NesterFormPhiB0pDefinition]},
   MetricOn → All, ContractMetrics → True];
 NesterFormPhiB1pDefinition = ToNesterForm[PhiB1p[-i, -j], "ToShell" → False];
 Print[NesterFormPhiB1pDefinition];
 ToNesterFormPhiB1p =
  MakeRule[{PhiB1p[-i, -j], Evaluate[NesterFormPhiB1pDefinition]},
   MetricOn → All, ContractMetrics → True];
 NesterFormPhiB1mDefinition = ToNesterForm[PhiB1m[-i], "ToShell" → False];
 Print[NesterFormPhiB1mDefinition];
 ToNesterFormPhiB1m = MakeRule[{PhiB1m[-i], Evaluate[NesterFormPhiB1mDefinition]},
   MetricOn → All, ContractMetrics → True];
 NesterFormPhiB2pDefinition = ToNesterForm[PhiB2p[-i, -j], "ToShell" → False];
 Print[NesterFormPhiB2pDefinition];
 ToNesterFormPhiB2p =
  MakeRule[{PhiB2p[-i, -j], Evaluate[NesterFormPhiB2pDefinition]},
   MetricOn → All, ContractMetrics → True];
 NesterFormPhiA0pDefinition = ToNesterForm[PhiA0p[], "ToShell" → False];
 Print[NesterFormPhiA0pDefinition];
 ToNesterFormPhiA0p = MakeRule[{PhiA0p[], Evaluate[NesterFormPhiA0pDefinition]},
   MetricOn → All, ContractMetrics → True];
 NesterFormPhiA0mDefinition = ToNesterForm[PhiA0m[], "ToShell" → False];
 Print[NesterFormPhiA0mDefinition];
 ToNesterFormPhiA0m = MakeRule[{PhiA0m[], Evaluate[NesterFormPhiA0mDefinition]},
   MetricOn → All, ContractMetrics → True];
 NesterFormPhiA1pDefinition = ToNesterForm[PhiA1p[-i, -j], "ToShell" → False];
 Print[NesterFormPhiA1pDefinition];
 ToNesterFormPhiA1p =
  MakeRule[{PhiA1p[-i, -j], Evaluate[NesterFormPhiA1pDefinition]},
   MetricOn → All, ContractMetrics → True];
 NesterFormPhiA1mDefinition = ToNesterForm[PhiA1m[-i], "ToShell" → False];
 Print[NesterFormPhiA1mDefinition];
 ToNesterFormPhiA1m = MakeRule[{PhiA1m[-i], Evaluate[NesterFormPhiA1mDefinition]},
   MetricOn → All, ContractMetrics → True];
 NesterFormPhiA2pDefinition = ToNesterForm[PhiA2p[-i, -j], "ToShell" → False];
 Print[NesterFormPhiA2pDefinition];
 ToNesterFormPhiA2p =
  MakeRule[{PhiA2p[-i, -j], Evaluate[NesterFormPhiA2pDefinition]},
```

```
    MetricOn → All, ContractMetrics → True];
  NesterFormPhiA2mDefinition = ToNesterForm[PhiA2m[-i, -j, -k], "ToShell" → False];
  Print[NesterFormPhiA2mDefinition];
  ToNesterFormPhiA2m =
   MakeRule[{PhiA2m[-i, -j, -k], Evaluate[NesterFormPhiA2mDefinition]},
    MetricOn → All, ContractMetrics → True];

  PhiToNesterFormPhi =
   Join[ToNesterFormPhiB0p, ToNesterFormPhiB1p, ToNesterFormPhiB1m,
    ToNesterFormPhiB2p, ToNesterFormPhiA0p, ToNesterFormPhiA0m, ToNesterFormPhiA1p,
    ToNesterFormPhiA1m, ToNesterFormPhiA2p, ToNesterFormPhiA2m];

  DumpSave[NotebookDirectory[] <> "mx_cache/phitonesterformphi.mx",
   {PhiToNesterFormPhi}];
  Print["done phitonesterformphi"];
  Quit[];
 ]
 MyImport["phitonesterformphi.mx"];
 (*
 TheoryConstraints={PhiB1p[-i,-j],PhiB2p[-i,-j],
   PhiA0p[],PhiA2m[-i,-j,-k],ChiParaA2m[-i,-j,-k],ChiSingB1m[-i]};
 TheoryConstraints=(ToNesterForm[#,"ToShell"→False]/.Theory//CollectTensors)&/@
   TheoryConstraints;
 Print/@TheoryConstraints;
 *)
```

Basic form

```
In[∘]:= ChiActivate = {ρρ → 1};(*dummy version until the secondaries are determined!*)
 Options[ToBasicForm] = {"Hard" → False, "Order" → Infinity};
 ToBasicForm[x_, OptionsPattern[]] := Module[{res}, res = x;
    Print[Style["To basic form...", Blue, 10]];
    res = res /. PhiActivate // NoScalar;
    res = res /. ChiActivate // NoScalar;
    res = res /. ChiParaActivate // NoScalar;
    res = res /. ChiPerpActivate // NoScalar;
    res = res /. ChiSingActivate // NoScalar;
    res = ToOrderCanonical[res, OptionValue["Order"]];
    res = res /. DpRPDeactivate // NoScalar;
    If[OptionValue["Hard"], res = res // ToNewCanonical];
    res = res /. DRPDeactivate // NoScalar;
    If[OptionValue["Hard"], res = res // ToNewCanonical];
    res = res /. RPO3Activate // NoScalar;
    If[OptionValue["Hard"], res = res // ToNewCanonical];
```

```
res = res /. TPO3Activate // NoScalar;
If[OptionValue["Hard"], res = res // ToNewCanonical];
res = res /. StrengthPToStrength // NoScalar;
If[OptionValue["Hard"], res = res // ToNewCanonical];
res = res /. StrengthLambdaPToStrengthLambda // NoScalar;
If[OptionValue["Hard"], res = res // ToNewCanonical];
res = res /. DpPiPDeactivate // NoScalar;
If[OptionValue["Hard"], res = res // ToNewCanonical];
res = res /. DPiPDeactivate // NoScalar;
If[OptionValue["Hard"], res = res // ToNewCanonical];
res = res /. PiPO3Activate // NoScalar;
If[OptionValue["Hard"], res = res // ToNewCanonical];
res = res /. PO3PiActivate // NoScalar;
If[OptionValue["Hard"], res = res // ToNewCanonical];
res = res /. PADMPiActivate // NoScalar;
If[OptionValue["Hard"], res = res // ToNewCanonical];
res = res /. PiPToPi // NoScalar;
If[OptionValue["Hard"], res = res // ToNewCanonical];
res = res /. PhiActivate // NoScalar;
If[OptionValue["Hard"], res = res // ToNewCanonical];
res = res /. Theory // NoScalar;
If[OptionValue["Hard"], res = res // ToNewCanonical];
res = res /. ExpandStrengths // NoScalar;
If[OptionValue["Hard"], res = res // ToNewCanonical];
res = res /. PADMActivate // NoScalar;
If[OptionValue["Hard"], res = res // ToNewCanonical];
res = ToOrderCanonical[res, OptionValue["Order"]];
Print["basic completed"];
res = res // NoScalar;
res = res // ToNewCanonical;
res = res // NoScalar;
res];
```

sPFCs

Form of Hamiltonian

```
(*
Print[Style["Evaluating quadratic part of super-Hamiltonian",Orange,30]]
  temp=J[]((1/16)(ωB0p (PhiB0p[]PhiB0p[])+
          ωB1p  PhiB1p[-a,-b]PhiB1p[a,b]+
          ωB1m  PhiB1m[-a]PhiB1m[a]+
```

```
            ωB2p  PhiB2p[-a,-b]PhiB2p[a,b]+
            (1/4)(ωA0p  PhiA0p[]PhiA0p[]+
               ωA0m  PhiA0m[]PhiA0m[]+
               ωA1p  PhiA1p[-a,-b]PhiA1p[a,b]+
               ωA1m  PhiA1m[-a]PhiA1m[a]+
               ωA2p  PhiA2p[-a,-b]PhiA2p[a,b]+
               ωA2m  PhiA2m[-a,-b,-c]PhiA2m[a,b,c])))/.NewFreedoms/.Theory//
   ToCanonical//CollectTensors;
Print[temp];

HamiltonianFormCase19=temp;

DumpSave[NotebookDirectory[]<>"mx_cache/HamiltonianFormCase19.mx",
 {HamiltonianFormCase19}];
Print["done HamiltonianForm"];
Quit[];
*)
Print[Style["Quadratic part of super-Hamiltonian", Orange, 30]]
MyImport["HamiltonianFormCase1.mx"];
MyImport["HamiltonianFormCase2.mx"];
MyImport["HamiltonianFormCase3.mx"];
MyImport["HamiltonianFormCase4.mx"];
MyImport["HamiltonianFormCase5.mx"];
MyImport["HamiltonianFormCase6.mx"];
MyImport["HamiltonianFormCase7.mx"];
MyImport["HamiltonianFormCase8.mx"];
MyImport["HamiltonianFormCase9.mx"];
MyImport["HamiltonianFormCase10.mx"];
MyImport["HamiltonianFormCase11.mx"];
MyImport["HamiltonianFormCase12.mx"];
MyImport["HamiltonianFormCase13.mx"];
MyImport["HamiltonianFormCase14.mx"];
MyImport["HamiltonianFormCase15.mx"];
MyImport["HamiltonianFormCase16.mx"];
MyImport["HamiltonianFormCase17.mx"];
MyImport["HamiltonianFormCase18.mx"];
MyImport["HamiltonianFormCase19.mx"];

For[ii = 1, ii < 20, ii++,
  Print[ii];
  Print[Evaluate[ToExpression["HamiltonianFormCase" <> ToString[ii]]]];
  Print[Evaluate[ToExpression["TheoryCaseGho" <> ToString[ii]]]];
 ];
```

```
Quit[];
```

```
(*
Print["field parts"];
temp=-(J[]T[i,-m,-n]PPara[m,g]PPara[n,h](Bet1 PT1[-i,-g,-h,a,c,d]+
            Bet2 PT2[-i,-g,-h,a,c,d]+
            Bet3 PT3[-i,-g,-h,a,c,d])PPara[-c,e]PPara[-d,f]T[-a,-e,-f]+
         J[](R[i,j,-m,-n]PPara[m,g]PPara[n,h](Alp1 PR1[-i,-j,-g,-h,a,b,c,d]+
               Alp2 PR2[-i,-j,-g,-h,a,b,c,d]+
               Alp3 PR3[-i,-j,-g,-h,a,b,c,d]+
               Alp4 PR4[-i,-j,-g,-h,a,b,c,d]+
               Alp5 PR5[-i,-j,-g,-h,a,b,c,d]+
               Alp6 PR6[-i,-j,-g,-h,a,b,c,d])-(1/2)Alp0 PPara[a,c]PPara[b,d])
          PPara[-c,e]PPara[-d,f]R[-a,-b,-e,-f])/.PActivate/.
     PADMActivate/.Theory//ToCanonical//CollectTensors;

temp=ToNesterForm[temp,"ToShell"→True,"Hard"→True,"Order"→Infinity];
temp=temp//ToNewCanonical;
temp=temp//CollectTensors;
temp=temp//Simplify;
Print[temp];
*)
Quit[];
```

0^+ super-Hamiltonian

```
(*SERIOUS ERROR FOUND HERE<
 SHELLPRIMSJP IS NOT ENOUGH YOU NEED THE OMEGA NEW FREEDOMS*)
(*
Print[Style["0⁺ linear super-Hamiltonian quadratic part",Red,30]];
temp=J[]((1/16)(ShellPrimB0p (PhiB0p[]PhiB0p[])+
            ShellPrimB1p  PhiB1p[-a,-b]PhiB1p[a,b]+
            ShellPrimB1m  PhiB1m[-a]PhiB1m[a]+
            ShellPrimB2p  PhiB2p[-a,-b]PhiB2p[a,b]+
            (1/4)(ShellPrimA0p  PhiA0p[]PhiA0p[]+
               ShellPrimA0m  PhiA0m[]PhiA0m[]+
               ShellPrimA1p  PhiA1p[-a,-b]PhiA1p[a,b]+
               ShellPrimA1m  PhiA1m[-a]PhiA1m[a]+
               ShellPrimA2p  PhiA2p[-a,-b]PhiA2p[a,b]+
```

```
                    ShellPrimA2m  PhiA2m[-a,-b,-c]PhiA2m[a,b,c])))-
         (J[]T[i,-m,-n]PPara[m,g]PPara[n,h](Bet1 PT1[-i,-g,-h,a,c,d]+
                  Bet2 PT2[-i,-g,-h,a,c,d]+
                  Bet3 PT3[-i,-g,-h,a,c,d])PPara[-c,e]PPara[-d,f]T[-a,-e,-f]+
            J[](R[i,j,-m,-n]PPara[m,g]PPara[n,h](Alp1 PR1[-i,-j,-g,-h,a,b,c,d]+
                    Alp2 PR2[-i,-j,-g,-h,a,b,c,d]+
                    Alp3 PR3[-i,-j,-g,-h,a,b,c,d]+
                    Alp4 PR4[-i,-j,-g,-h,a,b,c,d]+
                    Alp5 PR5[-i,-j,-g,-h,a,b,c,d]+
                    Alp6 PR6[-i,-j,-g,-h,a,b,c,d])-(1/2)Alp0 PPara[a,c]PPara[b,d])
             PPara[-c,e]PPara[-d,f]R[-a,-b,-e,-f])/.PActivate/.
       PADMActivate/.Theory//ToCanonical//CollectTensors;
  Print[temp];
  temp=temp/.ShellFreedomsActivate;
  temp=temp//NoScalar;
  temp=temp/.PhiToNesterFormPhi;
  temp=temp/.Theory;
  temp=ToNesterForm[temp,"ToShell"→True,"Hard"→True,"Order"→1];
  temp=temp//ToNewCanonical;
  temp=temp//CollectTensors;
  temp=temp//Simplify;
  Print[temp];
  *)
  (*
  Print[Style["0⁺ linear super-Hamiltonian derivative part",Red,30]];
  temp=
   -V[k]G3[-b,n](CD[-n][BPiP[-k,j]H[-j,b]]-A[i,-k,-n]BPiP[-i,j]PPara[-j,m]H[-m,b])/.
    PADMActivate;
  temp=ToNesterForm[temp,"ToShell"→True,"Hard"→True,"Order"→1];
  Print[temp];
  *)
```

1⁻ linear super-momentum

```
In[∘]:= (*
  Print[Style["1⁻ linear super-momentum",Red,30]];
  temp=BPiP[-i,r]PPara[-r,p]PPara[-l,q]T[i,-q,-p]+
     (1/2)APiP[-i,-j,r]PPara[-r,p]PPara[-l,q]R[i,j,-q,-p]-
     PPara[-l,k]G3[-b,n](CD[-n][BPiP[-k,j]H[-j,b]]+
        A[i,-k,-n]BPiP[-i,j]PPara[-j,m]H[-m,b])/.PADMActivate;
  temp=ToNesterForm[temp,"ToShell"→True,"Hard"→True,"Order"→1];
  Print[temp];
  *)
```

1^+ angular super-momentum

In[•]:=
```
(*
Print[Style["1⁺ angular super-momentum",Red,30]];
temp=2PPara[-n,k]PPara[-m,l]Antisymmetrize[BPi[-k,a]G3[-a,b]B[-l,-b],{-k,-l}]+
   PPara[-n,k]PPara[-m,l]G3[-b,p](CD[-p][APiP[-k,-l,j]H[-j,b]])+
   PPara[-n,k]PPara[-m,l]G3[-b,p](-2Antisymmetrize[
       A[i,-k,-p]APiP[-i,-l,j] PPara[-j,z]H[-z,b],{-k,-l}])/.PADMActivate;
temp=ToNesterForm[temp,"ToShell"→True,"Hard"→True,"Order"→1];
Print[temp];
*)
```

1^- angular super-momentum

In[•]:=
```
(*
Print[Style["1⁻ angular super-momentum",Red,30]];
temp=2V[k]PPara[-m,l]Antisymmetrize[BPi[-k,a]G3[-a,b]B[-l,-b],{-k,-l}]+
   V[k]PPara[-m,l]G3[-b,p](CD[-p][APiP[-k,-l,j]H[-j,b]])+
   V[k]PPara[-m,l]G3[-b,p](-2Antisymmetrize[
       A[i,-k,-p]APiP[-i,-l,j] PPara[-j,z]H[-z,b],{-k,-l}])/.PADMActivate;
temp=ToNesterForm[temp,"ToShell"→True,"Hard"→True,"Order"→1];
Print[temp];
Print["all done"]
 Quit[];
*)
```

Lagrangian picture

Second-order formalism

```
Print[Style["Linearised field equations", Red, 40]];


DefNiceConstantSymbol[Λ, B];
DefNiceConstantSymbol[ψ, 0];
DefNiceConstantSymbol[ϕ, 0];
DefNiceConstantSymbol[ξ, 0];
DefNiceConstantSymbol[ξ, 1];
DefNiceConstantSymbol[ξ, 2];
DefNiceConstantSymbol[ξ, 3];
DefNiceConstantSymbol[ξ, 4];
DefNiceConstantSymbol[ξ, 5];
```

```
DefNiceConstantSymbol[ξ, 6];
DefNiceConstantSymbol[ζ, 0];
DefNiceConstantSymbol[ζ, 1];
DefNiceConstantSymbol[ζ, 2];
DefNiceConstantSymbol[ζ, 3];
DefNiceConstantSymbol[ζ, 4];
DefNiceConstantSymbol[ζ, 5];
DefNiceConstantSymbol[ζ, 6];
DefNiceConstantSymbol[H, 0];

(*
Bet2=-2/3;
Bet3=0;
*)
ΛB = 0;
ψ0 = 1 / Sqrt[-3 Alp6];
ϕ0 = 0;
H0 = 0;

DefTensor[RRC[i, -m, -n], M4, Antisymmetric[{-m, -n}], PrintAs → "c"];
DeclareOrder[RRC[i, -m, -n], 1];
DefTensor[Contortion[-i, -j, -k], M4, Antisymmetric[{-i, -j}]];

DefTensor[ETensor[-i, -j], M4, PrintAs → "τ"];
DeclareOrder[ETensor[-i, -j], 1];

DefTensor[TKilling[-i], M4, PrintAs → "u"];
AutomaticRules[TKilling,
  MakeRule[{CD[i][TKilling[-j]], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[TKilling, MakeRule[{TKilling[-i] TKilling[i], 1},
   MetricOn → All, ContractMetrics → True]];

IntermediateShell =
  MakeRule[{T1[i, -j, -k], 0}, MetricOn → All, ContractMetrics → True];
ContortionDefinition = -(1 / 2) (T[-i, -j, -k] - T[-j, -i, -k] - T[-k, -i, -j]) /.
   StrengthSO13Activate;
ContortionDefinition = ContortionDefinition /. IntermediateShell;
ContortionDefinition = ContortionDefinition // ToNewCanonical;
Print[ContortionDefinition];
ContortionToTorsion =
  MakeRule[{Contortion[-i, -j, -k], Evaluate[ContortionDefinition]},
   MetricOn → All, ContractMetrics → True];
```

```
RRCToCDB =
  MakeRule[{RRC[i, -j, -k], H[-j, m] H[-k, n] (CD[-m][B[i, -n]] - CD[-n][B[i, -m]])},
   MetricOn → All, ContractMetrics → True];
ADefinition = (1/2) (RRC[-i, -j, -k] - RRC[-k, -i, -j] + RRC[-j, -k, -i]) B[k, -m] +
   Contortion[-i, -j, -k] B[k, -m];
ADefinition = ADefinition /. RRCToCDB;
ADefinition = ADefinition /. ContortionToTorsion;
ADefinition = ADefinition // ToNewCanonical;
NewAActivate = MakeRule[{A[-i, -j, -m], Evaluate[ADefinition]},
   MetricOn → All, ContractMetrics → True];

DefTensor[DetB[], M4, PrintAs → "b"];

DefTensor[ScaleFactor[], M4, PrintAs → "a"];
AutomaticRules[ScaleFactor,
  MakeRule[{CD[-i][ScaleFactor[]], H0 TKilling[-i] ScaleFactor[]^2},
   MetricOn → All, ContractMetrics → True]];

DefTensor[NewtonianPotential[], M4, PrintAs → "Φ"];

DefTensor[BHarmonic[-i, -j], M4, Symmetric[{-i, -j}], PrintAs → "s̄"];
DeclareOrder[BHarmonic[-i, -j], 1];

ScaleFactorToday =
  MakeRule[{ScaleFactor[], 1}, MetricOn → All, ContractMetrics → True];

DefTensor[APiG[-i, -j, k, l], M4,
  {Antisymmetric[{-i, -j}], Antisymmetric[{k, l}]}, PrintAs → "Aπ̃"];
DefTensor[BPiG[-i, k, l], M4, Antisymmetric[{k, l}], PrintAs → "bπ̃"];
DefTensor[Lagrangian[], M4, PrintAs → "L_G"];

DefTensor[T2Faraday[-i, -j], M4, Antisymmetric[{-i, -j}], PrintAs → "F^(2)"];
DefTensor[T3Faraday[-i, -j], M4, Antisymmetric[{-i, -j}], PrintAs → "F^(3)"];

(**)
```

Generalised momenta and the Lagrangian

```
APiGDefinition = -4 DetB[] (2 (Alp1 PR1[-i, -j, -k, -l, a, b, c, d]
          + Alp2 PR2[-i, -j, -k, -l, a, b, c, d]
          + Alp3 PR3[-i, -j, -k, -l, a, b, c, d]
          + Alp4 PR4[-i, -j, -k, -l, a, b, c, d]
          + Alp5 PR5[-i, -j, -k, -l, a, b, c, d]
```

```
            + Alp6 PR6[-i, -j, -k, -l, a, b, c, d]) R[-a, -b, -c, -d] +
          (cAlp1 PR1[-i, -j, -k, -l, a, b, c, d]
            + cAlp2 PR2[-i, -j, -k, -l, a, b, c, d]
            + cAlp3 PR3[-i, -j, -k, -l, a, b, c, d]
            + cAlp4 PR4[-i, -j, -k, -l, a, b, c, d]
            + cAlp5 PR5[-i, -j, -k, -l, a, b, c, d]
            + cAlp6 PR6[-i, -j, -k, -l, a, b, c, d]) RLambda[-a, -b, -c, -d]) /.
      Theory /. PActivate // ToNewCanonical;
APiGDefinition = APiGDefinition /. StrengthLambdaSO13Activate;
APiGActivate = MakeRule[{APiG[-i, -j, -k, -l], Evaluate[APiGDefinition]},
   MetricOn → All, ContractMetrics → True];

BPiGDefinition = -2 DetB[] (2 (Bet1 PT1[-i, -j, -k, a, b, c]
            + Bet2 PT2[-i, -j, -k, a, b, c]
            + Bet3 PT3[-i, -j, -k, a, b, c]) T[-a, -b, -c] +
          (cBet1 PT1[-i, -j, -k, a, b, c]
            + cBet2 PT2[-i, -j, -k, a, b, c]
            + cBet3 PT3[-i, -j, -k, a, b, c]) TLambda[-a, -b, -c]) /. Theory /.
     PActivate // ToNewCanonical;
BPiGDefinition = BPiGDefinition /. StrengthLambdaSO13Activate;
BPiGActivate = MakeRule[{BPiG[-i, -j, -k], Evaluate[BPiGDefinition]},
   MetricOn → All, ContractMetrics → True];

LagrangianDefinition = (T[i, j, k] ((Bet1 PT1[-i, -j, -k, a, b, c]
              + Bet2 PT2[-i, -j, -k, a, b, c]
              + Bet3 PT3[-i, -j, -k, a, b, c]) T[-a, -b, -c] +
            (cBet1 PT1[-i, -j, -k, a, b, c]
              + cBet2 PT2[-i, -j, -k, a, b, c]
              + cBet3 PT3[-i, -j, -k, a, b, c]) TLambda[-a, -b, -c]) +
        R[i, j, k, l] ((Alp1 PR1[-i, -j, -k, -l, a, b, c, d]
              + Alp2 PR2[-i, -j, -k, -l, a, b, c, d]
              + Alp3 PR3[-i, -j, -k, -l, a, b, c, d]
              + Alp4 PR4[-i, -j, -k, -l, a, b, c, d]
              + Alp5 PR5[-i, -j, -k, -l, a, b, c, d]
              + Alp6 PR6[-i, -j, -k, -l, a, b, c, d]) R[-a, -b, -c, -d] +
            (cAlp1 PR1[-i, -j, -k, -l, a, b, c, d]
              + cAlp2 PR2[-i, -j, -k, -l, a, b, c, d]
              + cAlp3 PR3[-i, -j, -k, -l, a, b, c, d]
              + cAlp4 PR4[-i, -j, -k, -l, a, b, c, d]
              + cAlp5 PR5[-i, -j, -k, -l, a, b, c, d]
              + cAlp6 PR6[-i, -j, -k, -l, a, b, c, d]) RLambda[-a, -b, -c, -d])) /.
      Theory /. PActivate // ToNewCanonical;
LagrangianDefinition = LagrangianDefinition /. StrengthLambdaSO13Activate;
```

```
LagrangianActivate = MakeRule[{Lagrangian[], Evaluate[LagrangianDefinition]},
   MetricOn → All, ContractMetrics → True];

FormActivate = Join[APiGActivate, BPiGActivate, LagrangianActivate];
```

Introduction to xPert

```
<< xAct`xPert`;

Unprotect[IndexForm];
IndexForm[LI[x_]] := ColorString[ToString[x], RGBColor[1, 0, 0]];
Protect[IndexForm];

ToBackground = {PlaceholderRule → PlaceholderRuleValue};
ToPerturbation = {PlaceholderRule → PlaceholderRuleValue};
ToInertHarmonic = {PlaceholderRule → PlaceholderRuleValue};

DefTensor[CDBHarmonic[-l, -i, -j], M4, Symmetric[{-i, -j}], PrintAs → "∂s̄"];
ToBackground = Join[ToBackground,
   MakeRule[{CDBHarmonic[-l, -i, -j], 0}, MetricOn → All, ContractMetrics → True]];
DefTensor[CDCDBHarmonic[-l, -m, -i, -j], M4,
  {Symmetric[{-l, -m}], Symmetric[{-i, -j}]}, PrintAs → "∂∂s̄"];
ToBackground = Join[ToBackground, MakeRule[{CDCDBHarmonic[-l, -m, -i, -j], 0},
    MetricOn → All, ContractMetrics → True]];
DefTensor[CDCDCDBHarmonic[-l, -m, -n, -i, -j], M4,
  {Symmetric[{-l, -m, -n}], Symmetric[{-i, -j}]}, PrintAs → "∂∂∂s̄"];
ToBackground = Join[ToBackground, MakeRule[{CDCDCDBHarmonic[-l, -m, -n, -i, -j], 0},
    MetricOn → All, ContractMetrics → True]];
DefTensor[CDCDCDCDBHarmonic[-l, -m, -n, -o, -i, -j], M4,
  {Symmetric[{-l, -m, -n, -o}], Symmetric[{-i, -j}]}, PrintAs → "∂∂∂∂s̄"];
ToBackground = Join[ToBackground,
   MakeRule[{CDCDCDCDBHarmonic[-l, -m, -n, -o, -i, -j], 0},
    MetricOn → All, ContractMetrics → True]];

DefTensor[CDT2[-l, -i], M4, PrintAs → "∂ T^(2)"];
ToBackground = Join[ToBackground,
   MakeRule[{CDT2[-l, -i], 0}, MetricOn → All, ContractMetrics → True]];
DefTensor[CDCDT2[-l, -m, -i], M4, Symmetric[{-l, -m}], PrintAs → "∂∂ T^(2)"];
ToBackground = Join[ToBackground,
   MakeRule[{CDCDT2[-l, -m, -i], 0}, MetricOn → All, ContractMetrics → True]];
DefTensor[CDCDCDT2[-l, -m, -n, -i], M4, Symmetric[{-l, -m, -n}], PrintAs → "∂∂∂ T^(2)"];
ToBackground = Join[ToBackground,
   MakeRule[{CDCDCDT2[-l, -m, -n, -i], 0}, MetricOn → All, ContractMetrics → True]];
```

```
DefTensor[CDT3[-l, -i], M4, OrthogonalTo → {TKilling[l]}, PrintAs → "∂T⁽³⁾"];
ToBackground = Join[ToBackground,
   MakeRule[{CDT3[-l, -i], 0}, MetricOn → All, ContractMetrics → True]];
DefTensor[CDCDT3[-l, -m, -i], M4, Symmetric[{-l, -m}], PrintAs → "∂∂T⁽³⁾"];
ToBackground = Join[ToBackground,
   MakeRule[{CDCDT3[-l, -m, -i], 0}, MetricOn → All, ContractMetrics → True]];
DefTensor[CDCDCDT3[-l, -m, -n, -i], M4, Symmetric[{-l, -m, -n}], PrintAs → "∂∂∂T⁽³⁾"];
ToBackground = Join[ToBackground,
   MakeRule[{CDCDCDT3[-l, -m, -n, -i], 0}, MetricOn → All, ContractMetrics → True]];


(*


(*
(*Derivatives of the Newtonian potential assuming STATIC case*)
(**)
DefTensor[CDT2Pert[-l,-i],M4,OrthogonalTo→{TKilling[l]},PrintAs→"∂δT⁽²⁾"];
DeclareOrder[CDT2Pert[-l,-i],1];
DefTensor[CDCDT2Pert[-l,-m,-i],M4,Symmetric[{-l,-m}],
 OrthogonalTo→{TKilling[l],TKilling[m]},PrintAs→"∂∂δT⁽²⁾"];
DeclareOrder[CDCDT2Pert[-l,-m,-i],1];
DefTensor[CDCDCDT2Pert[-l,-m,-n,-i],M4,Symmetric[{-l,-m,-n}],
 OrthogonalTo→{TKilling[l],TKilling[m],TKilling[n]},PrintAs→"∂∂∂δT⁽²⁾"];
DeclareOrder[CDCDCDT2Pert[-l,-m,-n,-i],1];
DefTensor[CDT3Pert[-l,-i],M4,OrthogonalTo→{TKilling[l]},PrintAs→"∂δT⁽³⁾"];
DeclareOrder[CDT3Pert[-l,-i],1];
DefTensor[CDCDT3Pert[-l,-m,-i],M4,Symmetric[{-l,-m}],
 OrthogonalTo→{TKilling[l],TKilling[m]},PrintAs→"∂∂δT⁽³⁾"];
DeclareOrder[CDCDT3Pert[-l,-m,-i],1];
DefTensor[CDCDCDT3Pert[-l,-m,-n,-i],M4,Symmetric[{-l,-m,-n}],
 OrthogonalTo→{TKilling[l],TKilling[m],TKilling[n]},PrintAs→"∂∂∂δT⁽³⁾"];
DeclareOrder[CDCDCDT3Pert[-l,-m,-n,-i],1];
DefTensor[CDBHarmonic[-l,-i,-j],M4,
 Symmetric[{-i,-j}],OrthogonalTo→{TKilling[l]},PrintAs→"∂s̄"];
DeclareOrder[CDBHarmonic[-l,-i,-j],1];
AutomaticRules[CDBHarmonic,
 MakeRule[{CDBHarmonic[-l,l,-j],0},MetricOn→All,ContractMetrics→True]];
DefTensor[CDCDBHarmonic[-l,-m,-i,-j],M4,{Symmetric[{-l,-m}],Symmetric[{-i,-j}]},
 OrthogonalTo→{TKilling[l],TKilling[m]},PrintAs→"∂∂s̄"];
DeclareOrder[CDCDBHarmonic[-l,-m,-i,-j],1];
AutomaticRules[CDCDBHarmonic,
```

```
 MakeRule[{CDCDBHarmonic[-l,-m,l,-j],0},MetricOn→All,ContractMetrics→True]];
DefTensor[CDCDCDBHarmonic[-l,-m,-n,-i,-j],M4,
 {Symmetric[{-l,-m,-n}],Symmetric[{-i,-j}]},
 OrthogonalTo→{TKilling[l],TKilling[m],TKilling[n]},PrintAs→"∂∂∂s̄"];
DeclareOrder[CDCDCDBHarmonic[-l,-m,-n,-i,-j],1];
AutomaticRules[CDCDCDBHarmonic,
 MakeRule[{CDCDCDBHarmonic[-l,-m,-n,l,-j],0},MetricOn→All,ContractMetrics→True]];
DefTensor[CDCDCDCDBHarmonic[-l,-m,-n,-o,-i,-j],M4,
 {Symmetric[{-l,-m,-n,-o}],Symmetric[{-i,-j}]},
 OrthogonalTo→{TKilling[l],TKilling[m],TKilling[n],TKilling[o]},PrintAs→"∂∂∂∂s̄"];
DeclareOrder[CDCDCDCDBHarmonic[-l,-m,-n,-o,-i,-j],1];
AutomaticRules[CDCDCDCDBHarmonic,MakeRule[
  {CDCDCDCDBHarmonic[-l,-m,-n,-o,l,-j],0},MetricOn→All,ContractMetrics→True]];
(**)

(*Derivatives of the Newtonian potential assuming NON-STATIC case*)
DefTensor[CDT2Pert[-l,-i],M4,OrthogonalTo→{TKilling[l]},PrintAs→"∂δT(2)"];
DeclareOrder[CDT2Pert[-l,-i],1];
DefTensor[CDCDT2Pert[-l,-m,-i],M4,Symmetric[{-l,-m}],
 OrthogonalTo→{TKilling[l],TKilling[m]},PrintAs→"∂∂δT(2)"];
DeclareOrder[CDCDT2Pert[-l,-m,-i],1];
DefTensor[CDCDCDT2Pert[-l,-m,-n,-i],M4,Symmetric[{-l,-m,-n}],
 OrthogonalTo→{TKilling[l],TKilling[m],TKilling[n]},PrintAs→"∂∂∂δT(2)"];
DeclareOrder[CDCDCDT2Pert[-l,-m,-n,-i],1];
DefTensor[CDT3Pert[-l,-i],M4,OrthogonalTo→{TKilling[l]},PrintAs→"∂δT(3)"];
DeclareOrder[CDT3Pert[-l,-i],1];
DefTensor[CDCDT3Pert[-l,-m,-i],M4,Symmetric[{-l,-m}],
 OrthogonalTo→{TKilling[l],TKilling[m]},PrintAs→"∂∂δT(3)"];
DeclareOrder[CDCDT3Pert[-l,-m,-i],1];
DefTensor[CDCDCDT3Pert[-l,-m,-n,-i],M4,Symmetric[{-l,-m,-n}],
 OrthogonalTo→{TKilling[l],TKilling[m],TKilling[n]},PrintAs→"∂∂∂δT(3)"];
DeclareOrder[CDCDCDT3Pert[-l,-m,-n,-i],1];
DefTensor[CDBHarmonic[-l,-i,-j],M4,
 Symmetric[{-i,-j}],OrthogonalTo→{TKilling[l]},PrintAs→"∂s̄"];
DeclareOrder[CDBHarmonic[-l,-i,-j],1];
AutomaticRules[CDBHarmonic,
 MakeRule[{CDBHarmonic[-l,l,-j],0},MetricOn→All,ContractMetrics→True]];
DefTensor[CDCDBHarmonic[-l,-m,-i,-j],M4,{Symmetric[{-l,-m}],Symmetric[{-i,-j}]},
 OrthogonalTo→{TKilling[l],TKilling[m]},PrintAs→"∂∂s̄"];
DeclareOrder[CDCDBHarmonic[-l,-m,-i,-j],1];
AutomaticRules[CDCDBHarmonic,
```

```
 MakeRule[{CDCDBHarmonic[-l,-m,l,-j],0},MetricOn→All,ContractMetrics→True]];
DefTensor[CDCDCDBHarmonic[-l,-m,-n,-i,-j],M4,
 {Symmetric[{-l,-m,-n}],Symmetric[{-i,-j}]},
 OrthogonalTo→{TKilling[l],TKilling[m],TKilling[n]},PrintAs→"∂∂∂s̄"];
DeclareOrder[CDCDCDBHarmonic[-l,-m,-n,-i,-j],1];
AutomaticRules[CDCDCDBHarmonic,
 MakeRule[{CDCDCDBHarmonic[-l,-m,-n,l,-j],0},MetricOn→All,ContractMetrics→True]];
DefTensor[CDCDCDCDBHarmonic[-l,-m,-n,-o,-i,-j],M4,
 {Symmetric[{-l,-m,-n,-o}],Symmetric[{-i,-j}]},
 OrthogonalTo→{TKilling[l],TKilling[m],TKilling[n],TKilling[o]},PrintAs→"∂∂∂∂s̄"];
DeclareOrder[CDCDCDCDBHarmonic[-l,-m,-n,-o,-i,-j],1];
AutomaticRules[CDCDCDCDBHarmonic,MakeRule[
  {CDCDCDCDBHarmonic[-l,-m,-n,-o,l,-j],0},MetricOn→All,ContractMetrics→True]];
*)

ToCDT2Pert=MakeRule[{CD[-l][T2Pert[-i]],CDT2Pert[-l,-i]},
  MetricOn→All,ContractMetrics→True];
ToCDT3Pert=MakeRule[{CD[-l][T3Pert[-i]],CDT3Pert[-l,-i]},
  MetricOn→All,ContractMetrics→True];
ToCDBHarmonic=MakeRule[{CD[-l][BHarmonic[-i,-j]],CDBHarmonic[-l,-i,-j]},
  MetricOn→All,ContractMetrics→True];
ToCDAll=Join[ToCDT2Pert,ToCDT3Pert,ToCDBHarmonic];
ToCDCDT2Pert=MakeRule[{CD[-l][CD[-m][T2Pert[-i]]],CDCDT2Pert[-l,-m,-i]},
  MetricOn→All,ContractMetrics→True];
ToCDCDT2Pert1=MakeRule[{CD[-l][CDT2Pert[-m,-i]],CDCDT2Pert[-l,-m,-i]},
  MetricOn→All,ContractMetrics→True];
ToCDCDT3Pert=MakeRule[{CD[-l][CD[-m][T3Pert[-i]]],CDCDT3Pert[-l,-m,-i]},
  MetricOn→All,ContractMetrics→True];
ToCDCDT3Pert1=MakeRule[{CD[-l][CDT3Pert[-m,-i]],CDCDT3Pert[-l,-m,-i]},
  MetricOn→All,ContractMetrics→True];
ToCDCDBHarmonic=MakeRule[{CD[-l][CD[-m][BHarmonic[-i,-j]]],
   CDCDBHarmonic[-l,-m,-i,-j]},MetricOn→All,ContractMetrics→True];
ToCDCDBHarmonic1=MakeRule[{CD[-l][CDBHarmonic[-m,-i,-j]],
   CDCDBHarmonic[-l,-m,-i,-j]},MetricOn→All,ContractMetrics→True];
ToCDCDAll=Join[ToCDCDT2Pert,ToCDCDT2Pert1,ToCDCDT3Pert,
  ToCDCDT3Pert1,ToCDCDBHarmonic,ToCDCDBHarmonic1];
ToCDCDCDT2Pert=MakeRule[{CD[-l][CD[-m][CD[-n][T2Pert[-i]]]],
   CDCDCDT2Pert[-l,-m,-n,-i]},MetricOn→All,ContractMetrics→True];
ToCDCDCDT2Pert1=MakeRule[{CD[-l][CD[-m][CDT2Pert[-n,-i]]],
   CDCDCDT2Pert[-l,-m,-n,-i]},MetricOn→All,ContractMetrics→True];
ToCDCDCDT2Pert2=MakeRule[{CD[-l][CDCDT2Pert[-m,-n,-i]],CDCDCDT2Pert[-l,-m,-n,-i]},
  MetricOn→All,ContractMetrics→True];
ToCDCDCDT3Pert=MakeRule[{CD[-l][CD[-m][CD[-n][T3Pert[-i]]]],
```

```
    CDCDCDT3Pert[-l,-m,-n,-i]},MetricOn→All,ContractMetrics→True];
ToCDCDCDT3Pert1=MakeRule[{CD[-l][CD[-m][CDT3Pert[-n,-i]]],
    CDCDCDT3Pert[-l,-m,-n,-i]},MetricOn→All,ContractMetrics→True];
ToCDCDCDT3Pert2=MakeRule[{CD[-l][CDCDT3Pert[-m,-n,-i]],CDCDCDT3Pert[-l,-m,-n,-i]},
  MetricOn→All,ContractMetrics→True];
ToCDCDCDBHarmonic=MakeRule[{CD[-l][CD[-m][CD[-n][BHarmonic[-i,-j]]]],
    CDCDCDBHarmonic[-l,-m,-n,-i,-j]},MetricOn→All,ContractMetrics→True];
ToCDCDCDBHarmonic1=MakeRule[{CD[-l][CD[-m][CDBHarmonic[-n,-i,-j]]],
    CDCDCDBHarmonic[-l,-m,-n,-i,-j]},MetricOn→All,ContractMetrics→True];
ToCDCDCDBHarmonic2=MakeRule[{CD[-l][CDCDBHarmonic[-m,-n,-i,-j]],
    CDCDCDBHarmonic[-l,-m,-n,-i,-j]},MetricOn→All,ContractMetrics→True];
ToCDCDCDAll=Join[ToCDCDCDT2Pert,ToCDCDCDT2Pert1,ToCDCDCDT2Pert2,
  ToCDCDCDT3Pert,ToCDCDCDT3Pert1,ToCDCDCDT3Pert2,
  ToCDCDCDBHarmonic,ToCDCDCDBHarmonic1,ToCDCDCDBHarmonic2];
ToCDCDCDCDBHarmonic=MakeRule[{CD[-l][CD[-m][CD[-n][CD[-o][BHarmonic[-i,-j]]]]],
    CDCDCDCDBHarmonic[-l,-m,-n,-o,-i,-j]},MetricOn→All,ContractMetrics→True];
ToCDCDCDCDBHarmonic1=MakeRule[{CD[-l][CD[-m][CD[-n][CDBHarmonic[-o,-i,-j]]]],
    CDCDCDCDBHarmonic[-l,-m,-n,-o,-i,-j]},MetricOn→All,ContractMetrics→True];
ToCDCDCDCDBHarmonic2=MakeRule[{CD[-l][CD[-m][CDCDBHarmonic[-n,-o,-i,-j]]],
    CDCDCDCDBHarmonic[-l,-m,-n,-o,-i,-j]},MetricOn→All,ContractMetrics→True];
ToCDCDCDCDBHarmonic3=MakeRule[{CD[-l][CDCDCDBHarmonic[-m,-n,-o,-i,-j]],
    CDCDCDCDBHarmonic[-l,-m,-n,-o,-i,-j]},MetricOn→All,ContractMetrics→True];
ToCDCDCDCDAll=Join[ToCDCDCDCDBHarmonic,ToCDCDCDCDBHarmonic1,
  ToCDCDCDCDBHarmonic2,ToCDCDCDCDBHarmonic3];

*)

DefTensorPerturbation[CDBHarmonicPtn[LI[order], -l, -i, -j],
  CDBHarmonic[-l, -i, -j], M4, Symmetric[{-i, -j}], PrintAs → "∂s̄"];
AutomaticRules[CDBHarmonicPtn, MakeRule[{CDBHarmonicPtn[LI[1], -l, l, -j], 0},
    MetricOn → All, ContractMetrics → True]];
DefTensorPerturbation[CDCDBHarmonicPtn[LI[order], -l, -m, -i, -j],
  CDCDBHarmonic[-l, -m, -i, -j], M4,
  {Symmetric[{-l, -m}], Symmetric[{-i, -j}]}, PrintAs → "∂∂s̄"];
AutomaticRules[CDCDBHarmonicPtn,
  MakeRule[{CDCDBHarmonicPtn[LI[1], -l, -m, l, -j], 0},
    MetricOn → All, ContractMetrics → True]];
DefTensorPerturbation[CDCDCDBHarmonicPtn[LI[order], -l, ` -m, -n, -i, -j],
  CDCDCDBHarmonic[-l, -m, -n, -i, -j], M4,
  {Symmetric[{-l, -m, -n}], Symmetric[{-i, -j}]}, PrintAs → "∂∂∂s̄"];
AutomaticRules[CDCDCDBHarmonicPtn,
  MakeRule[{CDCDCDBHarmonicPtn[LI[1], -l, -m, -n, l, -j], 0},
    MetricOn → All, ContractMetrics → True]];
```

```
DefTensorPerturbation[CDCDCDCDBHarmonicPtn[LI[order], -l, -m, -n, -o, -i, -j],
  CDCDCDCDBHarmonic[-l, -m, -n, -o, -i, -j], M4,
  {Symmetric[{-l, -m, -n, -o}], Symmetric[{-i, -j}]}, PrintAs → "∂∂∂∂s̄"];
AutomaticRules[CDCDCDCDBHarmonicPtn,
  MakeRule[{CDCDCDCDBHarmonicPtn[LI[1], -l, -m, -n, -o, l, -j], 0},
   MetricOn → All, ContractMetrics → True]];


DefTensorPerturbation[
  CDT2Ptn[LI[order], -l, -i], CDT2[-l, -i], M4, PrintAs → "∂T^(2)"];
DefTensorPerturbation[CDCDT2Ptn[LI[order], -l, -m, -i],
  CDCDT2[-l, -m, -i], M4, Symmetric[{-l, -m}], PrintAs → "∂∂T^(2)"];
DefTensorPerturbation[CDCDCDT2Ptn[LI[order], -l, -m, -n, -i],
  CDCDCDT2[-l, -m, -n, -i], M4, Symmetric[{-l, -m, -n}], PrintAs → "∂∂∂T^(2)"];
DefTensorPerturbation[CDT3Ptn[LI[order], -l, -i],
  CDT3[-l, -i], M4, PrintAs → "∂T^(3)"];
DefTensorPerturbation[CDCDT3Ptn[LI[order], -l, -m, -i],
  CDCDT3[-l, -m, -i], M4, Symmetric[{-l, -m}], PrintAs → "∂∂T^(3)"];
DefTensorPerturbation[CDCDCDT3Ptn[LI[order], -l, -m, -n, -i],
  CDCDCDT3[-l, -m, -n, -i], M4, Symmetric[{-l, -m, -n}], PrintAs → "∂∂∂T^(3)"];
```

Setup perturbative quantities

```
In[•]:= Options[ToMyOrder] = {"ToInert" → False};
ToMyOrder[expr_, order_, OptionsPattern[]] := Module[{res},
   res = expr;
   Print[Style["ExpandPerturbation...", Orange, 10]];
   res = Perturbation[res, order] // ExpandPerturbation;
   (*Print[res//ScreenDollarIndices];*)
   Print[Style["ToPerturbation...", Orange, 10]];
   res = res /. ToPerturbation;
   res = res // ToNewCanonical;
   (*Print[res//ScreenDollarIndices];*)
   Print[Style["ToBackground...", Orange, 10]];
   res = res /. ToBackground;
   res = res // ToNewCanonical;
   (*Print[res//ScreenDollarIndices];*)
   Print[Style["SortCovDs...", Orange, 10]];
   res = res // SortCovDs;
   res = res // ToNewCanonical;
   Print[Style["ToInertHarmonic...", Orange, 10]];
```

```
    If[OptionValue["ToInert"], res = res /. ToInertHarmonic];
    res = res // ToNewCanonical;
    Print[Style["ToInertT...", Orange, 10]];
    If[OptionValue["ToInert"], res = res /. ToInertT];
    res = res // ToNewCanonical;
    Print[Style["ToFaraday...", Orange, 10]];
    If[OptionValue["ToInert"], res = res /. ToFaraday];
    res = res // ToNewCanonical;
    res];

CurrentPrint[expr_] := Module[{res},
    res = expr;
    Print[Style["ScaleFactorToday...", Orange, 10]];
    res = res /. ScaleFactorToday;
    res = res // ToCanonical;
    Print[res];
    res];

DefMetricPerturbation[G, GPtn, ϵ]
AutomaticRules[GPtn,
  MakeRule[{GPtn[LI[1], -b, -c], 0}, MetricOn → All, ContractMetrics → True]];

AutomaticRules[ScaleFactor, MakeRule[{Evaluate[Perturbation[ScaleFactor[]]], 0},
    MetricOn → All, ContractMetrics → True]];
AutomaticRules[TKilling, MakeRule[{Evaluate[Perturbation[TKilling[-i]]], 0},
    MetricOn → All, ContractMetrics → True]];

ToBackground = Join[ToBackground, MakeRule[{H[-i, m], (1 / ScaleFactor[]) G[-i, m]},
     MetricOn → All, ContractMetrics → True]];
DefTensorPerturbation[HPtn[LI[order], -i, m], H[-i, m], M4, PrintAs → "f"];

ToBackground = Join[ToBackground, MakeRule[
     {B[i, -m], ScaleFactor[] G[i, -m]}, MetricOn → All, ContractMetrics → True]];
DefTensorPerturbation[BPtn[LI[order], i, -m], B[i, -m], M4];
AutomaticRules[BPtn, MakeRule[{BPtn[LI[1], -b, -c], -HPtn[LI[1], -b, -c]},
    MetricOn → All, ContractMetrics → True]];

ToBackground = Join[ToBackground,
    MakeRule[{DetB[], ScaleFactor[] ^ 4}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[DetB, MakeRule[{Evaluate[Perturbation[DetB[]]],
     -ScaleFactor[] ^ 3 HPtn[LI[1], i, -i]}, MetricOn → All, ContractMetrics → True]];

ToBackground = Join[ToBackground,
```

```
    MakeRule[{BHarmonic[-i, -j], 0}, MetricOn → All, ContractMetrics → True]];
DefTensorPerturbation[BHarmonicPtn[LI[order], -i, -j],
   BHarmonic[-i, -j], M4, Symmetric[{-i, -j}], PrintAs → "s̄"];
AutomaticRules[BHarmonicPtn, MakeRule[{CD[i][BHarmonicPtn[LI[1], -i, -j]], 0},
    MetricOn → All, ContractMetrics → True]];
AutomaticRules[HPtn, MakeRule[{HPtn[LI[1], -i, -j],
     - (BHarmonicPtn[LI[1], -i, -j] - (1/2) G[-i, -j] BHarmonicPtn[LI[1], k, -k])},
    MetricOn → All, ContractMetrics → True]];

ToCDBHarmonic = MakeRule[{CD[-l][BHarmonicPtn[LI[1], -i, -j]],
     CDBHarmonicPtn[LI[1], -l, -i, -j]}, MetricOn → All, ContractMetrics → True];
ToCDCDBHarmonic = MakeRule[{CD[-l][CD[-m][BHarmonicPtn[LI[1], -i, -j]]],
     CDCDBHarmonicPtn[LI[1], -l, -m, -i, -j]},
    MetricOn → All, ContractMetrics → True];
ToCDCDBHarmonic1 = MakeRule[{CD[-l][CDBHarmonicPtn[LI[1], -m, -i, -j]],
     CDCDBHarmonicPtn[LI[1], -l, -m, -i, -j]},
    MetricOn → All, ContractMetrics → True];
ToCDCDCDBHarmonic = MakeRule[{CD[-l][CD[-m][CD[-n][BHarmonicPtn[LI[1], -i, -j]]]],
     CDCDCDBHarmonicPtn[LI[1], -l, -m, -n, -i, -j]},
    MetricOn → All, ContractMetrics → True];
ToCDCDCDBHarmonic1 = MakeRule[{CD[-l][CD[-m][CDBHarmonicPtn[LI[1], -n, -i, -j]]],
     CDCDCDBHarmonicPtn[LI[1], -l, -m, -n, -i, -j]},
    MetricOn → All, ContractMetrics → True];
ToCDCDCDBHarmonic2 = MakeRule[{CD[-l][CDCDBHarmonicPtn[LI[1], -m, -n, -i, -j]],
     CDCDCDBHarmonicPtn[LI[1], -l, -m, -n, -i, -j]},
    MetricOn → All, ContractMetrics → True];
ToCDCDCDCDBHarmonic = MakeRule[
    {CD[-l][CD[-m][CD[-n][CD[-o][BHarmonicPtn[LI[1], -i, -j]]]]],
     CDCDCDCDBHarmonicPtn[LI[1], -l, -m, -n, -o, -i, -j]},
    MetricOn → All, ContractMetrics → True];
ToCDCDCDCDBHarmonic1 = MakeRule[
    {CD[-l][CD[-m][CD[-n][CDBHarmonicPtn[LI[1], -o, -i, -j]]]],
     CDCDCDCDBHarmonicPtn[LI[1], -l, -m, -n, -o, -i, -j]},
    MetricOn → All, ContractMetrics → True];
ToCDCDCDCDBHarmonic2 = MakeRule[
    {CD[-l][CD[-m][CDCDBHarmonicPtn[LI[1], -n, -o, -i, -j]]],
     CDCDCDCDBHarmonicPtn[LI[1], -l, -m, -n, -o, -i, -j]},
    MetricOn → All, ContractMetrics → True];
ToCDCDCDCDBHarmonic3 = MakeRule[
    {CD[-l][CDCDCDBHarmonicPtn[LI[1], -m, -n, -o, -i, -j]],
     CDCDCDCDBHarmonicPtn[LI[1], -l, -m, -n, -o, -i, -j]},
    MetricOn → All, ContractMetrics → True];
```

```
ToInertHarmonic = Join[ToCDBHarmonic, ToCDCDBHarmonic, ToCDCDBHarmonic1,
   ToCDCDCDBHarmonic, ToCDCDCDBHarmonic1, ToCDCDCDBHarmonic2, ToCDCDCDCDBHarmonic,
   ToCDCDCDCDBHarmonic1, ToCDCDCDCDBHarmonic2, ToCDCDCDCDBHarmonic3];

ToBackground = Join[ToBackground,
   MakeRule[{NewtonianPotential[], 0}, MetricOn → All, ContractMetrics → True]];
DefTensorPerturbation[NewtonianPotentialPtn[LI[order]],
  NewtonianPotential[], M4, PrintAs → "Φ"];
AutomaticRules[NewtonianPotentialPtn,
  MakeRule[{TKilling[i] CD[-i][NewtonianPotentialPtn[LI[1]]], 0},
   MetricOn → All, ContractMetrics → True]];

ToBackground = Join[ToBackground,
   MakeRule[{T1[-i, -j, -k], 0}, MetricOn → All, ContractMetrics → True]];
AutomaticRules[T1, MakeRule[{Perturbation[T1[-i, -j, -k]], 0},
   MetricOn → All, ContractMetrics → True]];

ToBackground = Join[ToBackground,
   MakeRule[{T2[-i], 0 ϕ0 TKilling[-i]}, MetricOn → All, ContractMetrics → True]];
DefTensorPerturbation[T2Ptn[LI[order], -i], T2[-i], M4 , PrintAs -> "T^(2)"];
ToBackground = Join[ToBackground,
   MakeRule[{T3[-i], 0 ψ0 TKilling[-i]}, MetricOn → All, ContractMetrics → True]];
DefTensorPerturbation[T3Ptn[LI[order], -i], T3[-i], M4, PrintAs -> "T^(3)"];

ToCDT2 = MakeRule[{CD[-l][T2Ptn[LI[1], -i]], CDT2Ptn[LI[1], -l, -i]},
   MetricOn → All, ContractMetrics → True];
ToCDT3 = MakeRule[{CD[-l][T3Ptn[LI[1], -i]], CDT3Ptn[LI[1], -l, -i]},
   MetricOn → All, ContractMetrics → True];
ToCDCDT2 = MakeRule[{CD[-l][CD[-m][T2Ptn[LI[1], -i]]],
    CDCDT2Ptn[LI[1], -l, -m, -i]}, MetricOn → All, ContractMetrics → True];
ToCDCDT21 = MakeRule[{CD[-l][CDT2Ptn[LI[1], -m, -i]], CDCDT2Ptn[LI[1], -l, -m, -i]},
   MetricOn → All, ContractMetrics → True];
ToCDCDT3 = MakeRule[{CD[-l][CD[-m][T3Ptn[LI[1], -i]]],
    CDCDT3Ptn[LI[1], -l, -m, -i]}, MetricOn → All, ContractMetrics → True];
ToCDCDT31 = MakeRule[{CD[-l][CDT3Ptn[LI[1], -m, -i]], CDCDT3Ptn[LI[1], -l, -m, -i]},
   MetricOn → All, ContractMetrics → True];
ToCDCDCDT2 = MakeRule[{CD[-l][CD[-m][CD[-n][T2Ptn[LI[1], -i]]]],
    CDCDCDT2Ptn[LI[1], -l, -m, -n, -i]}, MetricOn → All, ContractMetrics → True];
ToCDCDCDT21 = MakeRule[{CD[-l][CD[-m][CDT2Ptn[LI[1], -n, -i]]],
    CDCDCDT2Ptn[LI[1], -l, -m, -n, -i]}, MetricOn → All, ContractMetrics → True];
ToCDCDCDT22 = MakeRule[{CD[-l][CDCDT2Ptn[LI[1], -m, -n, -i]],
    CDCDCDT2Ptn[LI[1], -l, -m, -n, -i]}, MetricOn → All, ContractMetrics → True];
ToCDCDCDT3 = MakeRule[{CD[-l][CD[-m][CD[-n][T3Ptn[LI[1], -i]]]],
```

```
    CDCDCDT3Ptn[LI[1], -l, -m, -n, -i]}, MetricOn → All, ContractMetrics → True];
ToCDCDCDT31 = MakeRule[{CD[-l][CD[-m][CDT3Ptn[LI[1], -n, -i]]],
    CDCDCDT3Ptn[LI[1], -l, -m, -n, -i]}, MetricOn → All, ContractMetrics → True];
ToCDCDCDT32 = MakeRule[{CD[-l][CDCDT3Ptn[LI[1], -m, -n, -i]],
    CDCDCDT3Ptn[LI[1], -l, -m, -n, -i]}, MetricOn → All, ContractMetrics → True];

ToInertT = Join[ToCDT2, ToCDT3, ToCDCDT2, ToCDCDT21, ToCDCDT3, ToCDCDT31, ToCDCDCDT2,
   ToCDCDCDT21, ToCDCDCDT22, ToCDCDCDT3, ToCDCDCDT31, ToCDCDCDT32];

ToBackground = Join[ToBackground,
   MakeRule[{T2Faraday[-i, -j], 0}, MetricOn → All, ContractMetrics → True]];
DefTensorPerturbation[T2FaradayPtn[LI[order], -i, -j], T2Faraday[-i, -j],
  M4 , Antisymmetric[{-i, -j}], PrintAs -> "F̊⁽²⁾"];
ToT2Faraday = MakeRule[{CD[-i][T2Ptn[LI[1], -j]],
    Evaluate[(1/2) T2FaradayPtn[LI[1], -i, -j] +
      Symmetrize[CD[-i][T2Ptn[LI[1], -j]], {-i, -j}]]},
   MetricOn → All, ContractMetrics → True];
ToT2Faraday1 = MakeRule[{CDT2Ptn[LI[1], -l, -i],
    Evaluate[(1/2) T2FaradayPtn[LI[1], -l, -i] + Symmetrize[CDT2Ptn[LI[1], -l, -i],
       {-l, -i}]]}, MetricOn → All, ContractMetrics → True];
ToT2Faraday2 = MakeRule[{CDCDT2Ptn[LI[1], -l, -m, -i],
    Evaluate[(1/2) CD[-m][T2FaradayPtn[LI[1], -l, -i]] +
      Symmetrize[CDCDT2Ptn[LI[1], -l, -m, -i], {-l, -i}]]},
   MetricOn → All, ContractMetrics → True];
ToT2Faraday3 = MakeRule[{CDCDCDT2Ptn[LI[1], -l, -m, -n, -i],
    Evaluate[(1/2) CD[-m][CD[-n][T2FaradayPtn[LI[1], -l, -i]]] +
      Symmetrize[CDCDCDT2Ptn[LI[1], -l, -m, -n, -i], {-l, -i}]]},
   MetricOn → All, ContractMetrics → True];

ToBackground = Join[ToBackground,
   MakeRule[{T3Faraday[-i, -j], 0}, MetricOn → All, ContractMetrics → True]];
DefTensorPerturbation[T3FaradayPtn[LI[order], -i, -j], T3Faraday[-i, -j],
  M4 , Antisymmetric[{-i, -j}], PrintAs -> "F̊⁽³⁾"];
ToT3Faraday = MakeRule[{CD[-i][T3Ptn[LI[1], -j]],
    Evaluate[(1/2) T3FaradayPtn[LI[1], -i, -j] +
      Symmetrize[CD[-i][T3Ptn[LI[1], -j]], {-i, -j}]]},
   MetricOn → All, ContractMetrics → True];
ToT3Faraday1 = MakeRule[{CDT3Ptn[LI[1], -l, -i],
    Evaluate[(1/2) T3FaradayPtn[LI[1], -l, -i] + Symmetrize[CDT3Ptn[LI[1], -l, -i],
       {-l, -i}]]}, MetricOn → All, ContractMetrics → True];
ToT3Faraday2 = MakeRule[{CDCDT3Ptn[LI[1], -l, -m, -i],
    Evaluate[(1/2) CD[-m][T3FaradayPtn[LI[1], -l, -i]] +
```

```
      Symmetrize[CDCDT3Ptn[LI[1], -l, -m, -i], {-l, -i}]]},
   MetricOn → All, ContractMetrics → True];
ToT3Faraday3 = MakeRule[{CDCDCDT3Ptn[LI[1], -l, -m, -n, -i],
     Evaluate[(1/2) CD[-m][CD[-n][T3FaradayPtn[LI[1], -l, -i]]] +
      Symmetrize[CDCDCDT3Ptn[LI[1], -l, -m, -n, -i], {-l, -i}]]},
   MetricOn → All, ContractMetrics → True];

ToFaraday = Join[ToT2Faraday, ToT2Faraday1, ToT2Faraday2,
   ToT2Faraday3, ToT3Faraday, ToT3Faraday1, ToT3Faraday2, ToT3Faraday3];

ToBackground = Join[ToBackground,
   MakeRule[{TLambda1[-i, -j, -k], 0}, MetricOn → All, ContractMetrics → True]];
DefTensorPerturbation[TLambda1Ptn[LI[order], -i, -j, -k],
  TLambda1[-i, -j, -k], M4, Symmetric[{-i, -j}], PrintAs -> "T(1)λ"];
AutomaticRules[TLambda1Ptn, MakeRule[{TLambda1Ptn[LI[1], a, a1, -a1], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[TLambda1Ptn, MakeRule[{TLambda1Ptn[LI[1], a, -a, -k], 0},
   MetricOn → All, ContractMetrics → True]];


(*transition to static Newtonian limit*)
(*
(*AutomaticRules[CDBHarmonicPtn,
  MakeRule[{TKilling[l]CDBHarmonicPtn[LI[1],-l,-i,-j],0},
   MetricOn→All,ContractMetrics→True]];*)
(*AutomaticRules[CDCDBHarmonicPtn,
  MakeRule[{TKilling[l]CDCDBHarmonicPtn[LI[1],-l,-m,-i,-j],0},
   MetricOn→All,ContractMetrics→True]];*)
(*AutomaticRules[CDCDCDBHarmonicPtn,
  MakeRule[{TKilling[l]CDCDCDBHarmonicPtn[LI[1],-l,-m,-n,-i,-j],0},
   MetricOn→All,ContractMetrics→True]];*)
(*AutomaticRules[CDCDCDCDBHarmonicPtn,
  MakeRule[{TKilling[l]CDCDCDCDBHarmonicPtn[LI[1],-l,-m,-n,-o,-i,-j],0},
   MetricOn→All,ContractMetrics→True]];*)


AutomaticRules[BHarmonicPtn,
 MakeRule[{BHarmonicPtn[LI[1],-i,-j],2TKilling[-i]TKilling[-j]
    NewtonianPotentialPtn[LI[1]]},MetricOn→All,ContractMetrics→True]];
AutomaticRules[T2Ptn,MakeRule[{T2Ptn[LI[1],-i],
   ξ0 TKilling[-i]NewtonianPotentialPtn[LI[1]]+ξ1
     CD[-i][NewtonianPotentialPtn[LI[1]]]},MetricOn→All,ContractMetrics→True]];
AutomaticRules[T3Ptn,MakeRule[{T3Ptn[LI[1],-i],
```

```
    ξ0 TKilling[-i]NewtonianPotentialPtn[LI[1]]+ξ1
      CD[-i][NewtonianPotentialPtn[LI[1]]]},MetricOn→All,ContractMetrics→True]];
*)
(*
T3PtnDefinition=ξ0 TKilling[-j]BHarmonicPtn[LI[1],-i,j]+
  ξ1 TKilling[-i]BHarmonicPtn[LI[1],-j,j]+
  ξ2 TKilling[-j]TKilling[-k]BHarmonicPtn[LI[1],j,k]TKilling[-i];
AutomaticRules[T3Ptn,MakeRule[{T3Ptn[LI[1],-i],Evaluate[T3PtnDefinition]},
  MetricOn→All,ContractMetrics→True]];

T2PtnDefinition=ξ0 CD[-j][BHarmonicPtn[LI[1],-i,j]]+
  ξ1 CD[-j][BHarmonicPtn[LI[1],-k,j]]TKilling[k]TKilling[-i]+
  ξ2 CD[-i][BHarmonicPtn[LI[1],-k,k]]+
  ξ3 CD[-i][BHarmonicPtn[LI[1],-k,-j]]TKilling[k]TKilling[j]+
  ξ4 CD[-j][BHarmonicPtn[LI[1],-k,-i]]TKilling[j]TKilling[k]+
  ξ5 CD[-j][BHarmonicPtn[LI[1],-k,k]]TKilling[j]TKilling[-i]+
  ξ6 CD[-j][BHarmonicPtn[LI[1],-k,-l]]
   TKilling[j]TKilling[k]TKilling[l]TKilling[-i];
AutomaticRules[T2Ptn,MakeRule[{T2Ptn[LI[1],-i],Evaluate[T2PtnDefinition]},
  MetricOn→All,ContractMetrics→True]];
*)
```

Pre-calculation of background and first order quantities

```
If[linearisationToggle,
  Print[Style["Rotational gauge field", Orange, 40]];

  tmp = A[i, j, -m];
  tmp = tmp /. NewAActivate;
  ABackgroundDefinition = ToMyOrder[tmp, 0, "ToInert" → True];
  Print[ABackgroundDefinition];
  ToBackground =
   Join[ToBackground, MakeRule[{A[i, j, -m], Evaluate[ABackgroundDefinition]},
     MetricOn → All, ContractMetrics → True]];
  APerturbationDefinition = ToMyOrder[tmp, 1, "ToInert" → True];
  Print[APerturbationDefinition];
  ToPerturbation = Join[ToPerturbation, MakeRule[
     {Evaluate[Perturbation[A[i, j, -m]]], Evaluate[APerturbationDefinition]},
     MetricOn → All, ContractMetrics → True]];

  Print[Style["Riemann-Cartan curvature", Orange, 40]];

  tmp = R[i, j, -k, -l];
  tmp = tmp /. ExpandStrengths;
```

```
RBackgroundDefinition = ToMyOrder[tmp, 0, "ToInert" → True];
Print[RBackgroundDefinition];
ToBackground =
 Join[ToBackground, MakeRule[{R[i, j, -k, -l], Evaluate[RBackgroundDefinition]},
   MetricOn → All, ContractMetrics → True]];
RPerturbationDefinition = ToMyOrder[tmp, 1, "ToInert" → True];
Print[RPerturbationDefinition];
ToPerturbation = Join[ToPerturbation, MakeRule[
   {Evaluate[Perturbation[R[i, j, -k, -l]]], Evaluate[RPerturbationDefinition]},
   MetricOn → All, ContractMetrics → True]];

Print[Style["Torsion", Orange, 40]];

tmp = T[i, -j, -k];
tmp = tmp /. ExpandStrengths;
TBackgroundDefinition = ToMyOrder[tmp, 0, "ToInert" → True];
Print[TBackgroundDefinition];
ToBackground =
 Join[ToBackground, MakeRule[{T[i, -j, -k], Evaluate[TBackgroundDefinition]},
   MetricOn → All, ContractMetrics → True]];
TPerturbationDefinition = ToMyOrder[tmp, 1, "ToInert" → True];
Print[TPerturbationDefinition];
ToPerturbation = Join[ToPerturbation, MakeRule[
   {Evaluate[Perturbation[T[i, -j, -k]]], Evaluate[TPerturbationDefinition]},
   MetricOn → All, ContractMetrics → True]];

Print[Style["Rotational momentum", Orange, 40]];

tmp = APiG[i, j, -k, -l];
tmp = tmp /. FormActivate;
APiGBackgroundDefinition = ToMyOrder[tmp, 0, "ToInert" → True];
Print[APiGBackgroundDefinition];
ToBackground = Join[ToBackground,
  MakeRule[{APiG[i, j, -k, -l], Evaluate[APiGBackgroundDefinition]},
   MetricOn → All, ContractMetrics → True]];
APiGPerturbationDefinition = ToMyOrder[tmp, 1, "ToInert" → True];
Print[APiGPerturbationDefinition];
ToPerturbation = Join[ToPerturbation,
  MakeRule[{Evaluate[Perturbation[APiG[i, j, -k, -l]]], Evaluate[
     APiGPerturbationDefinition]}, MetricOn → All, ContractMetrics → True]];

Print[Style["Translational momentum", Orange, 40]];

tmp = BPiG[i, -j, -k];
```

```
tmp = tmp /. FormActivate;
BPiGBackgroundDefinition = ToMyOrder[tmp, 0, "ToInert" → True];
Print[BPiGBackgroundDefinition];
ToBackground = Join[ToBackground,
  MakeRule[{BPiG[i, -j, -k], Evaluate[BPiGBackgroundDefinition]},
   MetricOn → All, ContractMetrics → True]];
BPiGPerturbationDefinition = ToMyOrder[tmp, 1, "ToInert" → True];
Print[BPiGPerturbationDefinition];
ToPerturbation = Join[ToPerturbation,
  MakeRule[{Evaluate[Perturbation[BPiG[i, -j, -k]]], Evaluate[
     BPiGPerturbationDefinition]}, MetricOn → All, ContractMetrics → True]];

Print[Style["Lagrangian", Orange, 40]];

tmp = Lagrangian[];
tmp = tmp /. FormActivate;
LagrangianBackgroundDefinition = ToMyOrder[tmp, 0, "ToInert" → True];
Print[LagrangianBackgroundDefinition];
ToBackground = Join[ToBackground,
  MakeRule[{Lagrangian[], Evaluate[LagrangianBackgroundDefinition]},
   MetricOn → All, ContractMetrics → True]];
LagrangianPerturbationDefinition = ToMyOrder[tmp, 1, "ToInert" → True];
Print[LagrangianPerturbationDefinition];
ToPerturbation =
 Join[ToPerturbation, MakeRule[{Evaluate[Perturbation[Lagrangian[]]],
    Evaluate[LagrangianPerturbationDefinition]},
   MetricOn → All, ContractMetrics → True]];

Print[Style["Energy-momentum tensor", Orange, 40]];

ETensorDefinition = -CD[-m][BPiG[-i, p, q] H[-p, n] H[-q, m]] +
   A[j, -i, -m] BPiG[-j, p, q] H[-p, n] H[-q, m] + T[p, -k, -i] BPiG[-p, k, r]
    H[-r, n] + (1/2) R[p, q, -k, -i] APiG[-p, -q, k, r] H[-r, n] +
   DetB[] Lagrangian[] H[-i, n] - ΔB H[-i, n] DetB[] // ToNewCanonical;
ETensorActivate = MakeRule[{ETensor[n, -i], Evaluate[ETensorDefinition]},
  MetricOn → All, ContractMetrics → True];

tmp = ETensor[n, -i];
tmp = tmp /. ETensorActivate;
ETensorBackgroundDefinition = ToMyOrder[tmp, 0, "ToInert" → True];
Print[ETensorBackgroundDefinition];
ToBackground = Join[ToBackground,
  MakeRule[{ETensor[n, -i], Evaluate[ETensorBackgroundDefinition]},
   MetricOn → All, ContractMetrics → True]];
```

```
    ETensorPerturbationDefinition = ToMyOrder[tmp, 1, "ToInert" → True];
    Print[ETensorPerturbationDefinition];
    ToPerturbation =
     Join[ToPerturbation, MakeRule[{Evaluate[Perturbation[ETensor[n, -i]]], Evaluate[
         ETensorPerturbationDefinition]}, MetricOn → All, ContractMetrics → True]];

    Print[Style["Spin tensor", Orange, 40]];

    STensorDefinition =
     -CD[-m][APiG[-i, -j, p, q] H[-p, n] H[-q, m]] + A[k, -i, -m] APiG[-k, -j, p, q]
        H[-p, n] H[-q, m] + A[k, -j, -m] APiG[-i, -k, p, q] H[-p, n] H[-q, m] +
       2 Antisymmetrize[BPiG[-i, -j, r] H[-r, n], {-i, -j}] // ToNewCanonical;
    STensorActivate = MakeRule[{STensor[n, -i, -j], Evaluate[STensorDefinition]},
      MetricOn → All, ContractMetrics → True];

    tmp = STensor[n, -i, -j];
    tmp = tmp /. STensorActivate;
    STensorBackgroundDefinition = ToMyOrder[tmp, 0, "ToInert" → True];
    Print[STensorBackgroundDefinition];
    ToBackground = Join[ToBackground,
      MakeRule[{STensor[n, -i, -j], Evaluate[STensorBackgroundDefinition]},
       MetricOn → All, ContractMetrics → True]];
    STensorPerturbationDefinition = ToMyOrder[tmp, 1, "ToInert" → True];
    Print[STensorPerturbationDefinition];
    ToPerturbation = Join[ToPerturbation,
      MakeRule[{Evaluate[Perturbation[STensor[n, -i, -j]]], Evaluate[
         STensorPerturbationDefinition]}, MetricOn → All, ContractMetrics → True]];

    DumpSave[NotebookDirectory[] <> "mx_cache/linearisation.mx",
     {ToBackground, ToPerturbation}];
    Print["done linearisation"];
    Quit[];
   ];
MyImport["linearisation.mx"];
```

Examine field equations

```
(*
ξ0=1/Sqrt[-3Alp6];
ξ1=0;
ξ0=0;
ξ1=Alp5/Alp6-1
*)
```

```
(*
Print[Style["Part 1 spin equation",Orange,20]];

tmp=PT1[-i,-j,-k,-a,b,c]B[a,-m]STensor[m,-b,-c];
Print[tmp];
tmp=tmp/.PActivate;

tmp0=ToMyOrder[tmp,0];
CurrentPrint[tmp0];
tmp1=ToMyOrder[tmp,1];
CurrentPrint[tmp1];
tmp1=ToMyOrder[tmp,1,"ToInertHarmonic"→True];
CurrentPrint[tmp1];

(**)
Print[Style["Difference from part 1 spin equation",Red,20]];

tmp=(2 cBet1 epsilonG[-i,-a,-a1,-b] TKilling[a] (3/(8 cBet1))
     PT1[-j,a1,b,-r,p,q]B[r,-m]STensor[m,-p,-q])/(3 Sqrt[3] Sqrt[-Alp6])-
  CD[-k][PT1[-i,k,-j,-a,b,c]B[a,-m]STensor[m,-b,-c]];
Print[tmp];
tmp=tmp/.PActivate;

tmp0=ToMyOrder[tmp,0];
CurrentPrint[tmp0];
tmp1=ToMyOrder[tmp,1];
CurrentPrint[tmp1];
tmp1=ToMyOrder[tmp,1,"ToInertHarmonic"→True];
CurrentPrint[tmp1];

*)
(**)

MyImport["linearsols.mx"];

Print[Style["Part 2 spin equation", Orange, 40]];

tmp = B[c, -m] STensor[m, -b, -c];
Print[tmp];

tmp0 = ToMyOrder[tmp, 0];
CurrentPrint[tmp0];
tmp1 = ToMyOrder[tmp, 1];
```

```
CurrentPrint[tmp1];
tmp1 = ToMyOrder[tmp, 1, "ToInert" → True];
tmp1 = CurrentPrint[tmp1];
(*
tmp1=ToMyOrder[tmp,1,"ToInertHarmonic"→True];
tmp1=CurrentPrint[tmp1];

Print[Style["Attempting solutions",Orange,20]];
tmp1=tmp1//CollectTensors;
Print[tmp1];
eqs1=tmp1==0//ToConstantSymbolEquations;

Print[Style["Applying solutions",Orange,20]];
tmp1=tmp1/.sols[[1]];
tmp1=tmp1//ToNewCanonical;
tmp1=tmp1//CollectTensors;
Print[tmp1];
*)
Print[Style["Part 3 spin equation", Orange, 20]];

tmp = epsilonG[-i, -a, b, c] B[a, -m] STensor[m, -b, -c];
Print[tmp];

tmp0 = ToMyOrder[tmp, 0];
CurrentPrint[tmp0];
tmp1 = ToMyOrder[tmp, 1];
CurrentPrint[tmp1];
tmp1 = ToMyOrder[tmp, 1, "ToInert" → True];
tmp1 = CurrentPrint[tmp1];

Quit[];

(*
tmp1=ToMyOrder[tmp,1,"ToInertHarmonic"→True];
tmp1=CurrentPrint[tmp1];

Print[Style["Attempting solutions",Orange,20]];
tmp1=tmp1//CollectTensors;
Print[tmp1];
eqs2=tmp1==0//ToConstantSymbolEquations;

Print[Style["Applying solutions",Orange,20]];
tmp1=tmp1/.sols[[1]];
tmp1=tmp1//ToNewCanonical;
```

```
tmp1=tmp1//CollectTensors;
Print[tmp1];

Print[Style["Obtaining solutions",Orange,20]];

eqs=Join[eqs1,eqs2];
Print[eqs];
tkrn=eqs/.sols[[1]];
tkrn=Simplify/@tkrn;
Print[tkrn];
sols=Quiet[Solve[eqs,{ζ1,ζ2,ζ3,ζ4,ζ5,ζ6,ξ1,ξ2,ξ3,ξ4,ξ5,ξ6}]];
Print[sols];
DumpSave[NotebookDirectory[]<>"mx_cache/linearsols.mx",{sols}];
Print["done linearisation"];
*)

(**)
(*
Print[Style["Full stress-energy equation",Orange,20]];

tmp=B[-i,-m]ETensor[m,-j];
Print[tmp];

tmp0=ToMyOrder[tmp,0];
CurrentPrint[tmp0];
tmp1=ToMyOrder[tmp,1];
CurrentPrint[tmp1];
tmp1=ToMyOrder[tmp,1,"ToInertHarmonic"→True];
CurrentPrint[tmp1];
*)
Print[Style["Exclusionary stress-energy equation", Red, 20]];
(**)
tmp = B[-i, -m] ETensor[m, -j] -
   ((2 cBet1 epsilonG[-j, -a, -a1, -b] TKilling[a] (3 / (8 cBet1)) PT1[-i, a1,
         b, -r, p, q] B[r, -m] STensor[m, -p, -q]) / (3 Sqrt[3] Sqrt[-Alp6]) -
     CD[-k][PT1[-j, k, -i, -a, b, c] B[a, -m] STensor[m, -b, -c]]);
Print[tmp];
tmp = tmp /. PActivate;
tmp = Symmetrize[tmp, {-i, -j}];
tmp = tmp // ToNewCanonical;
tmp0 = ToMyOrder[tmp, 0];
CurrentPrint[tmp0];
tmp1 = ToMyOrder[tmp, 1];
```

```
CurrentPrint[tmp1];
Quit[];
(**)
tmp1 = ToMyOrder[tmp, 1, "ToInertHarmonic" → True];
tmp1 = CurrentPrint[tmp1];

Print[Style["Applying solutions", Orange, 20]];
tmp1 = tmp1 /. sols[[1]];
tmp1 = tmp1 // ToNewCanonical;
tmp1 = tmp1 // CollectTensors;
Print[tmp1];
(**)
(**)
(**)

tmp1 = -12 Alp6 CDCDCDCDBHarmonicPtn[xAct`xTensor`LI[1], a, -a, a1, -a1, -i, -j] -
   4 Alp6 CDCDCDCDBHarmonicPtn[xAct`xTensor`LI[1], -i, -j, a, -a, a1, -a1] -
   (2 (Alp5 + 3 Alp5 Alp6 ξ0^2 + 2 Alp6 (-2 + 3 Sqrt[3] Sqrt[-Alp6] ξ0 + 3 Alp6 ξ0^2))
      CDCDBHarmonicPtn[xAct`xTensor`LI[1], a, -a, a1, -a1] G[-i, -j]) / (9 Alp6) +
   4 Alp6 CDCDCDCDBHarmonicPtn[xAct`xTensor`LI[1], a, -a, a1, -a1, b, -b] G[-i, -j] -
   ((Alp5 + 2 Alp6) CDBHarmonicPtn[xAct`xTensor`LI[1], a1, -j, b]
      epsilonG[-i, -a, -a1, -b] TKilling[a]) / (6 Sqrt[3] (-Alp6)^(3/2)) +
   (2 (Alp5 - 4 Alp6) CDCDCDBHarmonicPtn[xAct`xTensor`LI[1], a1, -a1, b, -j, b1]
      epsilonG[-i, -a, -b, -b1] TKilling[a]) / (3 Sqrt[3] Sqrt[-Alp6]) +
   ((Alp5 + 2 Alp6) CDBHarmonicPtn[xAct`xTensor`LI[1], a1, b, -b]
      epsilonG[-i, -j, -a, -a1] TKilling[a]) / (6 Sqrt[3] (-Alp6)^(3/2)) -
   2/3 ξ0 (Sqrt[3] Alp5 Sqrt[-Alp6] ξ0 + 2 Alp6 (-3 + Sqrt[3] Sqrt[-Alp6] ξ0))
    CDCDCDBHarmonicPtn[xAct`xTensor`LI[1], a1, -a1, b, b1, -b1]
    epsilonG[-i, -j, -a, -b] TKilling[a] +
   ((Alp5 + 2 Alp6) ξ0 CDBHarmonicPtn[xAct`xTensor`LI[1], a1, -a, b]
      epsilonG[-i, -j, -a1, -b] TKilling[a]) / (6 Alp6) +
   2/3 (Alp5 + 2 Alp6) ξ0 CDCDCDBHarmonicPtn[xAct`xTensor`LI[1], a1, -a1, b, -a, b1]
    epsilonG[-i, -j, -b, -b1] TKilling[a] +
   ((Alp5 + 2 Alp6) CDBHarmonicPtn[xAct`xTensor`LI[1], a1, -i, b]
      epsilonG[-j, -a, -a1, -b] TKilling[a]) / (6 Sqrt[3] (-Alp6)^(3/2)) -
   (2 (Alp5 - 4 Alp6) CDCDCDBHarmonicPtn[xAct`xTensor`LI[1], a1, -a1, b, -i, b1]
      epsilonG[-j, -a, -b, -b1] TKilling[a]) / (3 Sqrt[3] Sqrt[-Alp6]) +
   (4 (Alp5 + 3 Alp5 Alp6 ξ0^2 + 2 Alp6 (-2 + 3 Sqrt[3] Sqrt[-Alp6] ξ0 + 3 Alp6 ξ0^2))
      CDCDBHarmonicPtn[xAct`xTensor`LI[1], b, -b, -a, -a1] G[-i, -j] TKilling[a]
      TKilling[a1]) / (9 Alp6) + ((Alp5 + 2 Alp6) (-Sqrt[3] Sqrt[-Alp6] + 3 Alp6 ξ0)
      CDBHarmonicPtn[xAct`xTensor`LI[1], b1, -a, -a1] epsilonG[-i, -j, -b, -b1]
      TKilling[a] TKilling[a1] TKilling[b]) / (18 Alp6^2) +
   (2 (Sqrt[3] Alp5 Sqrt[-Alp6] + 4 Sqrt[3] (-Alp6)^(3/2) + 6 Alp6^2 ξ0
```

```
        (-5 + 2 Sqrt[3] Sqrt[-Alp6] ξ0) + 3 Alp5 Alp6 ξ0 (1 + 2 Sqrt[3] Sqrt[-Alp6] ξ0))
      CDCDCDBHarmonicPtn[xAct`xTensor`LI[1], b1, -b1, c, -a, -a1]
      epsilonG[-i, -j, -b, -c] TKilling[a] TKilling[a1] TKilling[b]) / (9 Alp6) -
   (2 (Alp5 + 5 Alp6 + Sqrt[3] Alp5 Sqrt[-Alp6] ξ0 + 7 Sqrt[3] (-Alp6) ^ (3 / 2) ξ0)
      CDCDBHarmonicPtn[xAct`xTensor`LI[1], a1, -a1, -j, -a] TKilling[a]
      TKilling[-i]) / (9 Alp6) + ((Alp5 + 2 Alp6) (-Sqrt[3] Sqrt[-Alp6] + 3 Alp6 ξ0)
      CDBHarmonicPtn[xAct`xTensor`LI[1], b, -a, b1] epsilonG[-j, -a1, -b, -b1]
      TKilling[a] TKilling[a1] TKilling[-i]) / (18 Alp6^2) -
   (2 (Alp5 - Alp6) (1 + Sqrt[3] Sqrt[-Alp6] ξ0) CDCDBHarmonicPtn[
       xAct`xTensor`LI[1], a1, -a1, -i, -a] TKilling[a] TKilling[-j]) / (3 Alp6) +
   ((Alp5 + 2 Alp6) (Sqrt[3] Sqrt[-Alp6] - 3 Alp6 ξ0) CDBHarmonicPtn[
       xAct`xTensor`LI[1], b, -a, b1] epsilonG[-i, -a1, -b, -b1]
      TKilling[a] TKilling[a1] TKilling[-j]) / (18 Alp6^2) -
   (4 (Alp5 (-2 + 3 Alp6 ξ0^2) + Alp6 (-1 + 6 Sqrt[3] Sqrt[-Alp6] ξ0 + 6 Alp6 ξ0^2))
      CDCDBHarmonicPtn[xAct`xTensor`LI[1], a, -a, a1, -a1] TKilling[-i]
      TKilling[-j]) / (9 Alp6) + 4 / 9 (-1 + 7 Sqrt[3] Sqrt[-Alp6] ξ0 +
      12 Alp6 ξ0^2 + (2 Alp5 (-1 + Sqrt[3] Sqrt[-Alp6] ξ0 + 3 Alp6 ξ0^2)) / Alp6)
    CDCDBHarmonicPtn[xAct`xTensor`LI[1], b, -b, -a, -a1] TKilling[a]
    TKilling[a1] TKilling[-i] TKilling[-j];

tmp1 = Symmetrize[tmp1, {-i, -j}];
tmp1 = tmp1 // ToNewCanonical;
tmp1 = tmp1 // CollectTensors;
Print[tmp1];

Print[Style["Applying solutions", Orange, 20]];

(*Alp5→7Alp6/4,*)
(*
krules={ξ0→1/(2Sqrt[-3Alp6])};
*)


ksol = Quiet[Solve[Alp5 + 3 Alp5 Alp6 ξ0^2 +
       2 Alp6 (-2 + 3 Sqrt[3] Sqrt[-Alp6] ξ0 + 3 Alp6 ξ0^2) == 0, ξ0]];
krules = ksol[[1]];
Print[krules];

tmp1 = tmp1 /. krules;
tmp1 = tmp1 // ToNewCanonical;
tmp1 = tmp1 // CollectTensors;
Print[tmp1];
```

```
Print[Style["Applying final solutions", Orange, 20]];

hrules = {Alp5 → 7 Alp6 / 4};
Print[hrules];

tmp1 = tmp1 /. hrules;
tmp1 = tmp1 // ToNewCanonical;
tmp1 = tmp1 // CollectTensors;
Print[tmp1];

Quit[];

Print[Style["Trace of full stress-energy equation", Orange, 20]];



tmp = ETensor[n, -i] B[i, -n];
Print[tmp];

tmp0 = ToMyOrder[tmp, 0];
CurrentPrint[tmp0];
tmp1 = ToMyOrder[tmp, 1];
CurrentPrint[tmp1];
tmp1 = ToMyOrder[tmp, 1, "ToInertHarmonic" → True];
CurrentPrint[tmp1];



Quit[];

Print[Style["Belinfante tensor", Orange, 40]];

tmp = Symmetrize[ETensor[-i, -j] -
      (1 / 2) CD[-k] [2 Symmetrize[STensor[-i, -j, k], {-i, -j}] - STensor[k, -j, -i]],
     {-i, -j}] // ToNewCanonical;
Print[tmp];
tmp = ToMyOrder[tmp, 1];
CurrentPrint[tmp];
```

ORPHAN

Definitions and setup

In[•]:=
```
(*
Print[Style["Linearised field equations",Red,40]];

DefTensor[RRC[i,-m,-n],M4,Antisymmetric[{-m,-n}],PrintAs→"c"];
DeclareOrder[RRC[i,-m,-n],1];
DefTensor[Contortion[-i,-j,-k],M4,Antisymmetric[{-i,-j}]];

DefTensor[ETensor[-i,-j],M4,PrintAs→"τ"];
DeclareOrder[ETensor[-i,-j],1];

DefTensor[TKilling[-i],M4,PrintAs→"u"];
AutomaticRules[TKilling,
 MakeRule[{CD[i][TKilling[-j]],0},MetricOn→All,ContractMetrics→True]];
AutomaticRules[TKilling,MakeRule[{TKilling[-i]TKilling[i],1},
  MetricOn→All,ContractMetrics→True]];
DefNiceConstantSymbol[ψ,0];
DefNiceConstantSymbol[ϕ,0];
DefTensor[T2Pert[-i],M4,PrintAs→"δT(2)"];
DeclareOrder[T2Pert[-i],1];
DefTensor[T3Pert[-i],M4,PrintAs→"δT(3)"];
DeclareOrder[T3Pert[-i],1];
IntermediateShell=MakeRule[{T1[i,-j,-k],0},MetricOn→All,ContractMetrics→True];

T2Activate=
 MakeRule[{T2[-i],ϕ0 TKilling[-i]+T2Pert[-i]},MetricOn→All,ContractMetrics→True];
T3Activate=MakeRule[{T3[-i],ψ0 TKilling[-i]+T3Pert[-i]},
  MetricOn→All,ContractMetrics→True];
CosmicBackgroundActivate=Join[T2Activate,T3Activate];

ContortionDefinition=
 -(1/2) (T[-i,-j,-k]-T[-j,-i,-k]-T[-k,-i,-j])/.StrengthSO13Activate;
ContortionDefinition=ContortionDefinition/.IntermediateShell;
ContortionDefinition=ContortionDefinition//ToNewCanonical;
Print[ContortionDefinition];
ContortionToTorsion=
 MakeRule[{Contortion[-i,-j,-k],Evaluate[ContortionDefinition]},
  MetricOn→All,ContractMetrics→True];

RRCToCDB=MakeRule[{RRC[i,-j,-k],H[-j,m]H[-k,n] (CD[-m][B[i,-n]]-CD[-n][B[i,-m]])},
  MetricOn→All,ContractMetrics→True];
ADefinition=(1/2) (RRC[-i,-j,-k]-RRC[-k,-i,-j]+RRC[-j,-k,-i])B[k,-m]+
  Contortion[-i,-j,-k]B[k,-m];
```

```
ADefinition=ADefinition/.RRCToCDB;
ADefinition=ADefinition/.ContortionToTorsion;
ADefinition=ADefinition/.CosmicBackgroundActivate;
ADefinition=ADefinition//ToNewCanonical;
AActivate=
 MakeRule[{A[-i,-j,-m],Evaluate[ADefinition]},MetricOn→All,ContractMetrics→True];

DefTensor[DetB[],M4,PrintAs→"b"];
AutomaticRules[DetB,MakeRule[{CD[-i][DetB[]],DetB[]H[-k,n]CD[-i][B[k,-n]]},
  MetricOn→All,ContractMetrics→True]];

DefNiceConstantSymbol[H,0];
DefTensor[ScaleFactor[],M4,PrintAs→"a"];
AutomaticRules[ScaleFactor,
 MakeRule[{CD[-i][ScaleFactor[]],H0 TKilling[-i]ScaleFactor[]^2},
  MetricOn→All,ContractMetrics→True]];
DefTensor[InverseScaleFactor[],M4,PrintAs→"(a⁻¹)"];
AutomaticRules[InverseScaleFactor,
 MakeRule[{CD[-i][InverseScaleFactor[]],-H0 TKilling[-i]},
  MetricOn→All,ContractMetrics→True]];

DefTensor[BPerSym[-i,-j],M4,Symmetric[{-i,-j}],PrintAs→"s"];
DeclareOrder[BPerSym[-i,-j],1];
DefTensor[BHarmonic[-i,-j],M4,Symmetric[{-i,-j}],PrintAs→"s̄"];
DeclareOrder[BHarmonic[-i,-j],1];
AutomaticRules[BHarmonic,
 MakeRule[{CD[i][BHarmonic[-i,-j]],0},MetricOn→All,ContractMetrics→True]];
ToTraceReverse=MakeRule[
  {BPerSym[-i,-j],BHarmonic[-i,-j]-(1/2)G[-i,-j]BHarmonic[k,-k]},
  MetricOn→All,ContractMetrics→True];

(**)
LineariseH=
 MakeRule[{H[-i,m],G[-i,m]-BPerSym[-i,m]},MetricOn→All,ContractMetrics→True];
LineariseB=MakeRule[{B[i,-m],G[i,-m]+BPerSym[i,-m]},
  MetricOn→All,ContractMetrics→True];
LineariseDetB=MakeRule[{DetB[],1+BPerSym[i,-i]},
  MetricOn→All,ContractMetrics→True];
LineariseAll=Join[LineariseH,LineariseB,LineariseDetB];
(**)
(*uncomment to resurrect the expanding universe*)
(*
LineariseH=MakeRule[{H[-i,m],(1/ScaleFactor[])G[-i,m]-BPerSym[-i,m]},
```

```
  MetricOn→All,ContractMetrics→True];
LineariseB=MakeRule[{B[i,-m],ScaleFactor[]G[i,-m]+BPerSym[i,-m]},
  MetricOn→All,ContractMetrics→True];
LineariseDetB=MakeRule[{DetB[],ScaleFactor[]^4+ScaleFactor[]^3BPerSym[i,-i]},
  MetricOn→All,ContractMetrics→True];
LineariseAll=Join[LineariseH,LineariseB,LineariseDetB];
*)
ScaleFactorToday=MakeRule[{ScaleFactor[],1},MetricOn→All,ContractMetrics→True];
InverseScaleFactorToday=
 MakeRule[{InverseScaleFactor[],1},MetricOn→All,ContractMetrics→True];
TodaysCoordinates=Join[ScaleFactorToday,InverseScaleFactorToday];


DefTensor[APiG[-i,-j,k,l],M4,
 {Antisymmetric[{-i,-j}],Antisymmetric[{k,l}]},PrintAs→"Ãπ"];
DefTensor[BPiG[-i,k,l],M4,Antisymmetric[{k,l}],PrintAs→"b̃π"];
DefTensor[Lagrangian[],M4,PrintAs→"L_G"];

DefTensor[RLambdaEquation[-i,-j],M4,Antisymmetric[{-i,-j}],PrintAs→"∂_Rλ L"];
DefTensor[TLambdaEquation[-i,k,l],M4,Antisymmetric[{k,l}],PrintAs→"∂_Tλ L"];

DefTensor[Faraday2[-i,-j],M4,Antisymmetric[{-i,-j}],PrintAs→"δF^(2)"];
DeclareOrder[Faraday2[-i,-j],1];
DefTensor[Faraday3[-i,-j],M4,Antisymmetric[{-i,-j}],PrintAs→"δF^(3)"];
DeclareOrder[Faraday3[-i,-j],1];

ToFaraday2=MakeRule[{CD[-i][T2Pert[-j]],
   Evaluate[Symmetrize[CD[-i][T2Pert[-j]],{-i,-j}]+(1/2)Faraday2[-i,-j]]},
  MetricOn→All,ContractMetrics→True];
ToFaraday3=MakeRule[{CD[-i][T3Pert[-j]],
   Evaluate[Symmetrize[CD[-i][T3Pert[-j]],{-i,-j}]+(1/2)Faraday3[-i,-j]]},
  MetricOn→All,ContractMetrics→True];
ToFaraday=Join[ToFaraday2,ToFaraday3];
*)
```

Static system derivatives $\partial\bar{s}, \partial\delta \overset{(2)}{T}, \partial\delta \overset{(3)}{T}$

In[∘]:=
```
(*
(*Derivatives of the Newtonian potential assuming STATIC case*)
(**)
DefTensor[CDT2Pert[-l,-i],M4,OrthogonalTo→{TKilling[l]},PrintAs→"∂δT^(2)"];
DeclareOrder[CDT2Pert[-l,-i],1];
```

```
DefTensor[CDCDT2Pert[-l,-m,-i],M4,Symmetric[{-l,-m}],
 OrthogonalTo→{TKilling[l],TKilling[m]},PrintAs→"∂∂δ(2)T"];
DeclareOrder[CDCDT2Pert[-l,-m,-i],1];
DefTensor[CDCDCDT2Pert[-l,-m,-n,-i],M4,Symmetric[{-l,-m,-n}],
 OrthogonalTo→{TKilling[l],TKilling[m],TKilling[n]},PrintAs→"∂∂∂δ(2)T"];
DeclareOrder[CDCDCDT2Pert[-l,-m,-n,-i],1];
DefTensor[CDT3Pert[-l,-i],M4,OrthogonalTo→{TKilling[l]},PrintAs→"∂δ(3)T"];
DeclareOrder[CDT3Pert[-l,-i],1];
DefTensor[CDCDT3Pert[-l,-m,-i],M4,Symmetric[{-l,-m}],
 OrthogonalTo→{TKilling[l],TKilling[m]},PrintAs→"∂∂δ(3)T"];
DeclareOrder[CDCDT3Pert[-l,-m,-i],1];
DefTensor[CDCDCDT3Pert[-l,-m,-n,-i],M4,Symmetric[{-l,-m,-n}],
 OrthogonalTo→{TKilling[l],TKilling[m],TKilling[n]},PrintAs→"∂∂∂δ(3)T"];
DeclareOrder[CDCDCDT3Pert[-l,-m,-n,-i],1];
DefTensor[CDBHarmonic[-l,-i,-j],M4,
 Symmetric[{-i,-j}],OrthogonalTo→{TKilling[l]},PrintAs→"∂s̄"];
DeclareOrder[CDBHarmonic[-l,-i,-j],1];
AutomaticRules[CDBHarmonic,
 MakeRule[{CDBHarmonic[-l,l,-j],0},MetricOn→All,ContractMetrics→True]];
DefTensor[CDCDBHarmonic[-l,-m,-i,-j],M4,{Symmetric[{-l,-m}],Symmetric[{-i,-j}]},
 OrthogonalTo→{TKilling[l],TKilling[m]},PrintAs→"∂∂s̄"];
DeclareOrder[CDCDBHarmonic[-l,-m,-i,-j],1];
AutomaticRules[CDCDBHarmonic,
 MakeRule[{CDCDBHarmonic[-l,-m,l,-j],0},MetricOn→All,ContractMetrics→True]];
DefTensor[CDCDCDBHarmonic[-l,-m,-n,-i,-j],M4,
 {Symmetric[{-l,-m,-n}],Symmetric[{-i,-j}]},
 OrthogonalTo→{TKilling[l],TKilling[m],TKilling[n]},PrintAs→"∂∂∂s̄"];
DeclareOrder[CDCDCDBHarmonic[-l,-m,-n,-i,-j],1];
AutomaticRules[CDCDCDBHarmonic,
 MakeRule[{CDCDCDBHarmonic[-l,-m,-n,l,-j],0},MetricOn→All,ContractMetrics→True]];
DefTensor[CDCDCDCDBHarmonic[-l,-m,-n,-o,-i,-j],M4,
 {Symmetric[{-l,-m,-n,-o}],Symmetric[{-i,-j}]},
 OrthogonalTo→{TKilling[l],TKilling[m],TKilling[n],TKilling[o]},PrintAs→"∂∂∂∂s̄"];
DeclareOrder[CDCDCDCDBHarmonic[-l,-m,-n,-o,-i,-j],1];
AutomaticRules[CDCDCDCDBHarmonic,MakeRule[
  {CDCDCDCDBHarmonic[-l,-m,-n,-o,l,-j],0},MetricOn→All,ContractMetrics→True]];
(**)

(*Derivatives of the Newtonian potential assuming NON-STATIC case*)
(*
DefTensor[CDT2Pert[-l,-i],M4,OrthogonalTo→{TKilling[l]},PrintAs→"∂δ(2)T"];
```

```
DeclareOrder[CDT2Pert[-l,-i],1];
DefTensor[CDCDT2Pert[-l,-m,-i],M4,Symmetric[{-l,-m}],
 OrthogonalTo→{TKilling[l],TKilling[m]},PrintAs→"∂∂δT(2)"];
DeclareOrder[CDCDT2Pert[-l,-m,-i],1];
DefTensor[CDCDCDT2Pert[-l,-m,-n,-i],M4,Symmetric[{-l,-m,-n}],
 OrthogonalTo→{TKilling[l],TKilling[m],TKilling[n]},PrintAs→"∂∂∂δT(2)"];
DeclareOrder[CDCDCDT2Pert[-l,-m,-n,-i],1];
DefTensor[CDT3Pert[-l,-i],M4,OrthogonalTo→{TKilling[l]},PrintAs→"∂δT(3)"];
DeclareOrder[CDT3Pert[-l,-i],1];
DefTensor[CDCDT3Pert[-l,-m,-i],M4,Symmetric[{-l,-m}],
 OrthogonalTo→{TKilling[l],TKilling[m]},PrintAs→"∂∂δT(3)"];
DeclareOrder[CDCDT3Pert[-l,-m,-i],1];
DefTensor[CDCDCDT3Pert[-l,-m,-n,-i],M4,Symmetric[{-l,-m,-n}],
 OrthogonalTo→{TKilling[l],TKilling[m],TKilling[n]},PrintAs→"∂∂∂δT(3)"];
DeclareOrder[CDCDCDT3Pert[-l,-m,-n,-i],1];
DefTensor[CDBHarmonic[-l,-i,-j],M4,
 Symmetric[{-i,-j}],OrthogonalTo→{TKilling[l]},PrintAs→"∂s̄"];
DeclareOrder[CDBHarmonic[-l,-i,-j],1];
AutomaticRules[CDBHarmonic,
 MakeRule[{CDBHarmonic[-l,l,-j],0},MetricOn→All,ContractMetrics→True]];
DefTensor[CDCDBHarmonic[-l,-m,-i,-j],M4,{Symmetric[{-l,-m}],Symmetric[{-i,-j}]},
 OrthogonalTo→{TKilling[l],TKilling[m]},PrintAs→"∂∂s̄"];
DeclareOrder[CDCDBHarmonic[-l,-m,-i,-j],1];
AutomaticRules[CDCDBHarmonic,
 MakeRule[{CDCDBHarmonic[-l,-m,l,-j],0},MetricOn→All,ContractMetrics→True]];
DefTensor[CDCDCDBHarmonic[-l,-m,-n,-i,-j],M4,
 {Symmetric[{-l,-m,-n}],Symmetric[{-i,-j}]},
 OrthogonalTo→{TKilling[l],TKilling[m],TKilling[n]},PrintAs→"∂∂∂s̄"];
DeclareOrder[CDCDCDBHarmonic[-l,-m,-n,-i,-j],1];
AutomaticRules[CDCDCDBHarmonic,
 MakeRule[{CDCDCDBHarmonic[-l,-m,-n,l,-j],0},MetricOn→All,ContractMetrics→True]];
DefTensor[CDCDCDCDBHarmonic[-l,-m,-n,-o,-i,-j],M4,
 {Symmetric[{-l,-m,-n,-o}],Symmetric[{-i,-j}]},
 OrthogonalTo→{TKilling[l],TKilling[m],TKilling[n],TKilling[o]},PrintAs→"∂∂∂∂s̄"];
DeclareOrder[CDCDCDCDBHarmonic[-l,-m,-n,-o,-i,-j],1];
AutomaticRules[CDCDCDCDBHarmonic,MakeRule[
  {CDCDCDCDBHarmonic[-l,-m,-n,-o,l,-j],0},MetricOn→All,ContractMetrics→True]];
*)

ToCDT2Pert=MakeRule[{CD[-l][T2Pert[-i]],CDT2Pert[-l,-i]},
  MetricOn→All,ContractMetrics→True];
ToCDT3Pert=MakeRule[{CD[-l][T3Pert[-i]],CDT3Pert[-l,-i]},
```

```
  MetricOn→All,ContractMetrics→True];
ToCDBHarmonic=MakeRule[{CD[-l][BHarmonic[-i,-j]],CDBHarmonic[-l,-i,-j]},
  MetricOn→All,ContractMetrics→True];
ToCDAll=Join[ToCDT2Pert,ToCDT3Pert,ToCDBHarmonic];
ToCDCDT2Pert=MakeRule[{CD[-l][CD[-m][T2Pert[-i]]],CDCDT2Pert[-l,-m,-i]},
  MetricOn→All,ContractMetrics→True];
ToCDCDT2Pert1=MakeRule[{CD[-l][CDT2Pert[-m,-i]],CDCDT2Pert[-l,-m,-i]},
  MetricOn→All,ContractMetrics→True];
ToCDCDT3Pert=MakeRule[{CD[-l][CD[-m][T3Pert[-i]]],CDCDT3Pert[-l,-m,-i]},
  MetricOn→All,ContractMetrics→True];
ToCDCDT3Pert1=MakeRule[{CD[-l][CDT3Pert[-m,-i]],CDCDT3Pert[-l,-m,-i]},
  MetricOn→All,ContractMetrics→True];
ToCDCDBHarmonic=MakeRule[{CD[-l][CD[-m][BHarmonic[-i,-j]]],
   CDCDBHarmonic[-l,-m,-i,-j]},MetricOn→All,ContractMetrics→True];
ToCDCDBHarmonic1=MakeRule[{CD[-l][CDBHarmonic[-m,-i,-j]],
   CDCDBHarmonic[-l,-m,-i,-j]},MetricOn→All,ContractMetrics→True];
ToCDCDAll=Join[ToCDCDT2Pert,ToCDCDT2Pert1,ToCDCDT3Pert,
  ToCDCDT3Pert1,ToCDCDBHarmonic,ToCDCDBHarmonic1];
ToCDCDCDT2Pert=MakeRule[{CD[-l][CD[-m][CD[-n][T2Pert[-i]]]],
   CDCDCDT2Pert[-l,-m,-n,-i]},MetricOn→All,ContractMetrics→True];
ToCDCDCDT2Pert1=MakeRule[{CD[-l][CD[-m][CDT2Pert[-n,-i]]],
   CDCDCDT2Pert[-l,-m,-n,-i]},MetricOn→All,ContractMetrics→True];
ToCDCDCDT2Pert2=MakeRule[{CD[-l][CDCDT2Pert[-m,-n,-i]],CDCDCDT2Pert[-l,-m,-n,-i]},
  MetricOn→All,ContractMetrics→True];
ToCDCDCDT3Pert=MakeRule[{CD[-l][CD[-m][CD[-n][T3Pert[-i]]]],
   CDCDCDT3Pert[-l,-m,-n,-i]},MetricOn→All,ContractMetrics→True];
ToCDCDCDT3Pert1=MakeRule[{CD[-l][CD[-m][CDT3Pert[-n,-i]]],
   CDCDCDT3Pert[-l,-m,-n,-i]},MetricOn→All,ContractMetrics→True];
ToCDCDCDT3Pert2=MakeRule[{CD[-l][CDCDT3Pert[-m,-n,-i]],CDCDCDT3Pert[-l,-m,-n,-i]},
  MetricOn→All,ContractMetrics→True];
ToCDCDCDBHarmonic=MakeRule[{CD[-l][CD[-m][CD[-n][BHarmonic[-i,-j]]]],
   CDCDCDBHarmonic[-l,-m,-n,-i,-j]},MetricOn→All,ContractMetrics→True];
ToCDCDCDBHarmonic1=MakeRule[{CD[-l][CD[-m][CDBHarmonic[-n,-i,-j]]],
   CDCDCDBHarmonic[-l,-m,-n,-i,-j]},MetricOn→All,ContractMetrics→True];
ToCDCDCDBHarmonic2=MakeRule[{CD[-l][CDCDBHarmonic[-m,-n,-i,-j]],
   CDCDCDBHarmonic[-l,-m,-n,-i,-j]},MetricOn→All,ContractMetrics→True];
ToCDCDCDAll=Join[ToCDCDCDT2Pert,ToCDCDCDT2Pert1,ToCDCDCDT2Pert2,
  ToCDCDCDT3Pert,ToCDCDCDT3Pert1,ToCDCDCDT3Pert2,
  ToCDCDCDBHarmonic,ToCDCDCDBHarmonic1,ToCDCDCDBHarmonic2];
ToCDCDCDCDBHarmonic=MakeRule[{CD[-l][CD[-m][CD[-n][CD[-o][BHarmonic[-i,-j]]]]],
   CDCDCDCDBHarmonic[-l,-m,-n,-o,-i,-j]},MetricOn→All,ContractMetrics→True];
ToCDCDCDCDBHarmonic1=MakeRule[{CD[-l][CD[-m][CD[-n][CDBHarmonic[-o,-i,-j]]]],
   CDCDCDCDBHarmonic[-l,-m,-n,-o,-i,-j]},MetricOn→All,ContractMetrics→True];
ToCDCDCDCDBHarmonic2=MakeRule[{CD[-l][CD[-m][CDCDBHarmonic[-n,-o,-i,-j]]],
```

```
   CDCDCDCDBHarmonic[-l,-m,-n,-o,-i,-j]},MetricOn→All,ContractMetrics→True];
ToCDCDCDCDBHarmonic3=MakeRule[{CD[-l][CDCDCDBHarmonic[-m,-n,-o,-i,-j]],
   CDCDCDCDBHarmonic[-l,-m,-n,-o,-i,-j]},MetricOn→All,ContractMetrics→True];
ToCDCDCDCDAll=Join[ToCDCDCDCDBHarmonic,ToCDCDCDCDBHarmonic1,
  ToCDCDCDCDBHarmonic2,ToCDCDCDCDBHarmonic3];

(*IntermediateShell=Join[IntermediateShell,{ϕ0→0}];*)
(*
KillT2Pert=MakeRule[{T2Pert[-i],0},MetricOn→All,ContractMetrics→True];
IntermediateShell=Join[IntermediateShell,KillT2Pert];
KillCDT2Pert=MakeRule[{CDT2Pert[-l,-i],0},MetricOn→All,ContractMetrics→True];
IntermediateShell=Join[IntermediateShell,KillCDT2Pert];
KillCDCDT2Pert=
 MakeRule[{CDCDT2Pert[-l,-m,-i],0},MetricOn→All,ContractMetrics→True];
IntermediateShell=Join[IntermediateShell,KillCDCDT2Pert];
KillCDCDCDT2Pert=
 MakeRule[{CDCDCDT2Pert[-l,-m,-n,-i],0},MetricOn→All,ContractMetrics→True];
IntermediateShell=Join[IntermediateShell,KillCDCDCDT2Pert];
*)
(*
KillT3Pert=MakeRule[{T3Pert[-i],0},MetricOn→All,ContractMetrics→True];
IntermediateShell=Join[IntermediateShell,KillT3Pert];
KillCDT3Pert=MakeRule[{CDT3Pert[-l,-i],0},MetricOn→All,ContractMetrics→True];
IntermediateShell=Join[IntermediateShell,KillCDT3Pert];
KillCDCDT3Pert=
 MakeRule[{CDCDT3Pert[-l,-m,-i],0},MetricOn→All,ContractMetrics→True];
IntermediateShell=Join[IntermediateShell,KillCDCDT3Pert];
KillCDCDCDT3Pert=
 MakeRule[{CDCDCDT3Pert[-l,-m,-n,-i],0},MetricOn→All,ContractMetrics→True];
IntermediateShell=Join[IntermediateShell,KillCDCDCDT3Pert];
*)
*)
```

Newtonian potential Φ

In[∘]:=
```
(*
(*Derivatives of the Newtonian potential assuming STATIC case*)
(**)
DefTensor[Newt[],M4,PrintAs→"Φ"];
DeclareOrder[Newt[],1];
DefTensor[CDNewt[-l],M4,OrthogonalTo→{TKilling[l]},PrintAs→"∂Φ"];
DeclareOrder[CDNewt[-l],1];
DefTensor[CDCDNewt[-l,-m],M4,Symmetric[{-l,-m}],
```

```
 OrthogonalTo→{TKilling[l],TKilling[m]},PrintAs→"∂∂Φ"];
DeclareOrder[CDCDNewt[-l,-m],1];
DefTensor[CDCDCDNewt[-l,-m,-n],M4,Symmetric[{-l,-m,-n}],
 OrthogonalTo→{TKilling[l],TKilling[m],TKilling[n]},PrintAs→"∂∂∂Φ"];
DeclareOrder[CDCDCDNewt[-l,-m,-n],1];
DefTensor[CDCDCDCDNewt[-l,-m,-n,-o],M4,Symmetric[{-l,-m,-n,-o}],
 OrthogonalTo→{TKilling[l],TKilling[m],TKilling[n],TKilling[o]},PrintAs→"∂∂∂∂Φ"];
DeclareOrder[CDCDCDCDNewt[-l,-m,-n,-o],1];
(**)

(*Derivatives of the Newtonian potential assuming NON-STATIC case*)
(*
DefTensor[Newt[],M4,PrintAs→"Φ"];
DeclareOrder[Newt[],1];
DefTensor[CDNewt[-l],M4,PrintAs→"∂Φ"];
DeclareOrder[CDNewt[-l],1];
DefTensor[CDCDNewt[-l,-m],M4,Symmetric[{-l,-m}],PrintAs→"∂∂Φ"];
DeclareOrder[CDCDNewt[-l,-m],1];
DefTensor[CDCDCDNewt[-l,-m,-n],M4,Symmetric[{-l,-m,-n}],PrintAs→"∂∂∂Φ"];
DeclareOrder[CDCDCDNewt[-l,-m,-n],1];
DefTensor[CDCDCDCDNewt[-l,-m,-n,-o],
 M4,Symmetric[{-l,-m,-n,-o}],PrintAs→"∂∂∂∂Φ"];
DeclareOrder[CDCDCDCDNewt[-l,-m,-n,-o],1];
*)

DefConstantSymbol[Con2,PrintAs→"(2)τ"];
DefConstantSymbol[Con3,PrintAs→"(3)τ"];

ToNewtonian={};

BHarmonicToNewt=MakeRule[{BHarmonic[-i,-j],2Newt[]TKilling[-i]TKilling[-j]},
  MetricOn→All,ContractMetrics→True];
CDBHarmonicToCDNewt=MakeRule[{CDBHarmonic[-l,-i,-j],
   2CDNewt[-l]TKilling[-i]TKilling[-j]},MetricOn→All,ContractMetrics→True];
CDCDBHarmonicToCDCDNewt=MakeRule[{CDCDBHarmonic[-l,-m,-i,-j],
   2CDCDNewt[-l,-m]TKilling[-i]TKilling[-j]},MetricOn→All,ContractMetrics→True];
CDCDCDBHarmonicToCDCDCDNewt=MakeRule[{CDCDCDBHarmonic[-l,-m,-n,-i,-j],
   2CDCDCDNewt[-l,-m,-n]TKilling[-i]TKilling[-j]},
  MetricOn→All,ContractMetrics→True];
CDCDCDCDBHarmonicToCDCDCDCDNewt=MakeRule[{CDCDCDCDBHarmonic[-l,-m,-n,-o,-i,-j],
   2CDCDCDCDNewt[-l,-m,-n,-o]TKilling[-i]TKilling[-j]},
  MetricOn→All,ContractMetrics→True];
ToNewtonian=Join[ToNewtonian,BHarmonicToNewt,CDBHarmonicToCDNewt,
```

```
  CDCDBHarmonicToCDCDNewt,CDCDCDBHarmonicToCDCDCDNewt,
  CDCDCDCDBHarmonicToCDCDCDCDNewt];

T2PertToNewt=MakeRule[{T2Pert[-i],Con2 Newt[]TKilling[-i]},
  MetricOn→All,ContractMetrics→True];
CDT2PertToCDNewt1=MakeRule[{CDT2Pert[-l,-i],Con2 CDNewt[-l]TKilling[-i]},
  MetricOn→All,ContractMetrics→True];
CDT2PertToCDNewt2=MakeRule[{CD[-l][T2Pert[-i]],Con2 CDNewt[-l]TKilling[-i]},
  MetricOn→All,ContractMetrics→True];
CDCDT2PertToCDCDNewt1=MakeRule[{CDCDT2Pert[-l,-m,-i],
   Con2 CDCDNewt[-l,-m]TKilling[-i]},MetricOn→All,ContractMetrics→True];
CDCDT2PertToCDCDNewt2=MakeRule[{CD[-l][CD[-m][T2Pert[-i]]],
   Con2 CDCDNewt[-l,-m]TKilling[-i]},MetricOn→All,ContractMetrics→True];
CDCDT2PertToCDCDNewt3=MakeRule[{CD[-l][CDT2Pert[-m,-i]],
   Con2 CDCDNewt[-l,-m]TKilling[-i]},MetricOn→All,ContractMetrics→True];
CDCDCDT2PertToCDCDCDNewt1=MakeRule[{CDCDCDT2Pert[-l,-m,-n,-i],
   Con2 CDCDCDNewt[-l,-m,-n]TKilling[-i]},MetricOn→All,ContractMetrics→True];
CDCDCDT2PertToCDCDCDNewt2=MakeRule[{CD[-l][CD[-m][CD[-n][T2Pert[-i]]]],
   Con2 CDCDCDNewt[-l,-m,-n]TKilling[-i]},MetricOn→All,ContractMetrics→True];
CDCDCDT2PertToCDCDCDNewt3=MakeRule[{CD[-l][CD[-m][CDT2Pert[-n,-i]]],
   Con2 CDCDCDNewt[-l,-m,-n]TKilling[-i]},MetricOn→All,ContractMetrics→True];
CDCDCDT2PertToCDCDCDNewt4=MakeRule[{CD[-l][CDCDT2Pert[-m,-n,-i]],
   Con2 CDCDCDNewt[-l,-m,-n]TKilling[-i]},MetricOn→All,ContractMetrics→True];
(*
ToNewtonian=Join[ToNewtonian,T2PertToNewt,CDT2PertToCDNewt1,
   CDT2PertToCDNewt2,CDCDT2PertToCDCDNewt1,CDCDT2PertToCDCDNewt2,
   CDCDT2PertToCDCDNewt3,CDCDCDT2PertToCDCDCDNewt1,CDCDCDT2PertToCDCDCDNewt2,
   CDCDCDT2PertToCDCDCDNewt3,CDCDCDT2PertToCDCDCDNewt4];
*)

T3PertToNewt=MakeRule[{T3Pert[-i],Con3 Newt[]TKilling[-i]},
  MetricOn→All,ContractMetrics→True];
CDT3PertToCDNewt1=MakeRule[{CDT3Pert[-l,-i],Con3 CDNewt[-l]TKilling[-i]},
  MetricOn→All,ContractMetrics→True];
CDT3PertToCDNewt2=MakeRule[{CD[-l][T3Pert[-i]],Con3 CDNewt[-l]TKilling[-i]},
  MetricOn→All,ContractMetrics→True];
CDCDT3PertToCDCDNewt1=MakeRule[{CDCDT3Pert[-l,-m,-i],
   Con3 CDCDNewt[-l,-m]TKilling[-i]},MetricOn→All,ContractMetrics→True];
CDCDT3PertToCDCDNewt2=MakeRule[{CD[-l][CD[-m][T3Pert[-i]]],
   Con3 CDCDNewt[-l,-m]TKilling[-i]},MetricOn→All,ContractMetrics→True];
CDCDT3PertToCDCDNewt3=MakeRule[{CD[-l][CDT3Pert[-m,-i]],
   Con3 CDCDNewt[-l,-m]TKilling[-i]},MetricOn→All,ContractMetrics→True];
CDCDCDT3PertToCDCDCDNewt1=MakeRule[{CDCDCDT3Pert[-l,-m,-n,-i],
   Con3 CDCDCDNewt[-l,-m,-n]TKilling[-i]},MetricOn→All,ContractMetrics→True];
```

```
CDCDCDT3PertToCDCDCDNewt2=MakeRule[{CD[-l][CD[-m][CD[-n][T3Pert[-i]]]],
   Con3 CDCDCDNewt[-l,-m,-n]TKilling[-i]},MetricOn→All,ContractMetrics→True];
CDCDCDT3PertToCDCDCDNewt3=MakeRule[{CD[-l][CD[-m][CDT3Pert[-n,-i]]],
   Con3 CDCDCDNewt[-l,-m,-n]TKilling[-i]},MetricOn→All,ContractMetrics→True];
CDCDCDT3PertToCDCDCDNewt4=MakeRule[{CD[-l][CDCDT3Pert[-m,-n,-i]],
   Con3 CDCDCDNewt[-l,-m,-n]TKilling[-i]},MetricOn→All,ContractMetrics→True];
(*
ToNewtonian=Join[ToNewtonian,T3PertToNewt,CDT3PertToCDNewt1,
   CDT3PertToCDNewt2,CDCDT3PertToCDCDNewt1,CDCDT3PertToCDCDNewt2,
   CDCDT3PertToCDCDNewt3,CDCDCDT3PertToCDCDCDNewt1,CDCDCDT3PertToCDCDCDNewt2,
   CDCDCDT3PertToCDCDCDNewt3,CDCDCDT3PertToCDCDCDNewt4];
*)

CDNewtToCDNewt=
 MakeRule[{CD[-l][Newt[]],CDNewt[-l]},MetricOn→All,ContractMetrics→True];
CDCDNewtToCDCDNewt1=MakeRule[{CD[-l][CD[-m][Newt[]]],CDCDNewt[-l,-m]},
  MetricOn→All,ContractMetrics→True];
CDCDNewtToCDCDNewt2=MakeRule[{CD[-l][CDNewt[-m]],CDCDNewt[-l,-m]},
  MetricOn→All,ContractMetrics→True];
CDCDCDNewtToCDCDCDNewt1=MakeRule[{CD[-l][CD[-m][CD[-n][Newt[]]]],
   CDCDCDNewt[-l,-m,-n]},MetricOn→All,ContractMetrics→True];
CDCDCDNewtToCDCDCDNewt2=MakeRule[{CD[-l][CD[-m][CDNewt[-n]]],
   CDCDCDNewt[-l,-m,-n]},MetricOn→All,ContractMetrics→True];
CDCDCDNewtToCDCDCDNewt3=MakeRule[{CD[-l][CDCDNewt[-n,-m]],CDCDCDNewt[-l,-m,-n]},
  MetricOn→All,ContractMetrics→True];
CDCDCDCDNewtToCDCDCDCDNewt1=MakeRule[{CD[-l][CD[-m][CD[-n][CD[-o][Newt[]]]]],
   CDCDCDCDNewt[-l,-m,-n,-o]},MetricOn→All,ContractMetrics→True];
CDCDCDCDNewtToCDCDCDCDNewt2=MakeRule[{CD[-l][CD[-m][CD[-n][CDNewt[-o]]]],
   CDCDCDCDNewt[-l,-m,-n,-o]},MetricOn→All,ContractMetrics→True];
CDCDCDCDNewtToCDCDCDCDNewt3=MakeRule[{CD[-l][CD[-m][CDCDNewt[-o,-n]]],
   CDCDCDCDNewt[-l,-m,-n,-o]},MetricOn→All,ContractMetrics→True];
CDCDCDCDNewtToCDCDCDCDNewt4=MakeRule[{CD[-l][CDCDCDNewt[-o,-n,-m]],
   CDCDCDCDNewt[-l,-m,-n,-o]},MetricOn→All,ContractMetrics→True];
ToNewtonian=Join[ToNewtonian,CDNewtToCDNewt,CDCDNewtToCDCDNewt1,
  CDCDNewtToCDCDNewt2,CDCDCDNewtToCDCDCDNewt1,CDCDCDNewtToCDCDCDNewt2,
  CDCDCDNewtToCDCDCDNewt3,CDCDCDCDNewtToCDCDCDCDNewt1,CDCDCDCDNewtToCDCDCDCDNewt2,
  CDCDCDCDNewtToCDCDCDCDNewt3,CDCDCDCDNewtToCDCDCDCDNewt4];
*)
```

Constants Ansatz

```
(*
(**)ToCoast={Bet2→-2/3,ψ0→1/Sqrt[-3Alp6],ϕ0→0,Bet3→0,H0→0,Con2→0,Con3→0};
(**)(*CS1*)
```

```
(*ToCoast={Bet2→-2/3,ψ0→-1/Sqrt[-3Alp6],ϕ0→0,Bet3→0,H0→0,Con2→0,Con3→0};*)
(*CS2*)


(*ToCoast={Bet2→-2/3,ψ0→-1/Sqrt[-3Alp6],ϕ0→0,Bet3→0,H0→0,Con2→0,Con3→0};*)
(*Version to use if we want no emergent dark energy*)
(*ToCoast=
  {Bet2→-2/3,ψ0→1/Sqrt[-3Alp6],ϕ0→3Sqrt[-Bet3/(2Alp6)],H0→0,Con2→0,Con3→0};*)
(*Version to use if we want emergent dark energy NEW SIGNS*)
(*ToCoast=
  {Bet2→-2/3,ψ0→-1/Sqrt[-3Alp6],ϕ0→-3Sqrt[-Bet3/(2Alp6)],H0→0,Con2→0,Con3→0};*)
(*Version to use if we want emergent dark energy*)

DefNiceConstantSymbol[ζ,0];
DefNiceConstantSymbol[ζ,1];
T2PertDefinition=ζ0 TKilling[-l]Newt[]+ζ1 CDNewt[-l];
ToCoastT2PertActivate=MakeRule[
  {T2Pert[-l],Evaluate[T2PertDefinition]},MetricOn→All,ContractMetrics→True];
CDT2PertDefinition=ζ0 TKilling[-l]CDNewt[-i]+ζ1 CDCDNewt[-i,-l];
ToCoastCDT2PertActivate=MakeRule[{CDT2Pert[-i,-l],Evaluate[CDT2PertDefinition]},
  MetricOn→All,ContractMetrics→True];
CDCDT2PertDefinition=ζ0 TKilling[-l]CDCDNewt[-j,-i]+ζ1 CDCDCDNewt[-j,-i,-l];
ToCoastCDCDT2PertActivate=
 MakeRule[{CDCDT2Pert[-j,-i,-l],Evaluate[CDCDT2PertDefinition]},
  MetricOn→All,ContractMetrics→True];
CDCDCDT2PertDefinition=ζ0 TKilling[-l]CDCDCDNewt[-k,-j,-i]+
  ζ1 CDCDCDCDNewt[-k,-j,-i,-l];
ToCoastCDCDCDT2PertActivate=MakeRule[{CDCDCDT2Pert[-k,-j,-i,-l],
   Evaluate[CDCDCDT2PertDefinition]},MetricOn→All,ContractMetrics→True];

DefNiceConstantSymbol[ξ,0];
DefNiceConstantSymbol[ξ,1];
T3PertDefinition=ξ0 TKilling[-l]Newt[]+ξ1 CDNewt[-l];
ToCoastT3PertActivate=MakeRule[
  {T3Pert[-l],Evaluate[T3PertDefinition]},MetricOn→All,ContractMetrics→True];
CDT3PertDefinition=ξ0 TKilling[-l]CDNewt[-i]+ξ1 CDCDNewt[-i,-l];
ToCoastCDT3PertActivate=MakeRule[{CDT3Pert[-i,-l],Evaluate[CDT3PertDefinition]},
  MetricOn→All,ContractMetrics→True];
CDCDT3PertDefinition=ξ0 TKilling[-l]CDCDNewt[-j,-i]+ξ1 CDCDCDNewt[-j,-i,-l];
ToCoastCDCDT3PertActivate=
 MakeRule[{CDCDT3Pert[-j,-i,-l],Evaluate[CDCDT3PertDefinition]},
  MetricOn→All,ContractMetrics→True];
CDCDCDT3PertDefinition=ξ0 TKilling[-l]CDCDCDNewt[-k,-j,-i]+
```

```
  ξ1 CDCDCDCDNewt[-k,-j,-i,-l];
ToCoastCDCDCDT3PertActivate=MakeRule[{CDCDCDT3Pert[-k,-j,-i,-l],
   Evaluate[CDCDCDT3PertDefinition]},MetricOn→All,ContractMetrics→True];

DefNiceConstantSymbol[χ,0];
DefNiceConstantSymbol[χ,1];
RLambda5Definition=Antisymmetrize[χ0 TKilling[-i]CDNewt[-j]+
   χ1 epsilonG[-i,-j,k,l]TKilling[-k]CDNewt[-l],{-i,-j}];
ToCoastRLambda5Activate=MakeRule[{RLambda5[-i,-j],Evaluate[RLambda5Definition]},
  MetricOn→All,ContractMetrics→True];
(*
ToCoastRLambda5Activate=
  MakeRule[{RLambda5[-i,-j],0},MetricOn→All,ContractMetrics→True];
*)
ToCoast=Join[ToCoast,ToCoastRLambda5Activate,
  ToCoastT2PertActivate,ToCoastCDT2PertActivate,ToCoastCDCDT2PertActivate,
  ToCoastCDCDCDT2PertActivate,ToCoastT3PertActivate,ToCoastCDT3PertActivate,
  ToCoastCDCDT3PertActivate,ToCoastCDCDCDT3PertActivate];


(*ToSecondCoast={ζ0→0,ξ0→-(Sqrt[-Alp6]/(Sqrt[3] Alp6)),ξ1→0,
   χ0→0,χ1→-((2 Sqrt[-Alp6] (-Alp5+Alp6))/(Sqrt[3] cAlp5 Alp6))-
     (2 Sqrt[-Alp6] ζ1)/(Sqrt[3] cAlp5)};*)(*CS1*)
ToSecondCoast={ζ0→0,ξ0→-(Sqrt[-Alp6]/(Sqrt[3] Alp6)),ξ1→0,χ0→0,cAlp5→0};(*CS1*)
*)
```

Simplification processes

```
In[∘]:= (*
ThroughInert[expr_]:=Module[{res},
  Print[Style["ThroughInert",Green,10]];
  res=expr;
  res=res/.ToCDCDCDCDAll;
  res=res//ToNewCanonical;
  res=res/.ToCDCDCDAll;
  res=res//ToNewCanonical;
  res=res/.ToCDCDAll;
  res=res//ToNewCanonical;
  res=res/.ToCDAll;
  res=res//ToNewCanonical;
  res];
```

```
ToToday[expr_]:=Module[{res},
  res=expr;
  res=res/.TodaysCoordinates;
  res=res//ToNewCanonical;
  res=res//CollectTensors;
  res];

StaticUpToOrder[expr_,order_]:=Module[{res},
  Print[Style["StaticUpToOrder",Green,20]];
  res=expr;
  Print["TSO13Activate"];
  res=res/.TSO13Activate;
  res=res/.IntermediateShell;
  res=res/.CosmicBackgroundActivate;
  res=res//ToNewCanonical;
  Print["ExpandStrengths"];
  res=res/.ExpandStrengths;
  res=res//ToNewCanonical;
  Print["StrengthLambdaSO13Activate"];
  res=res/.StrengthLambdaSO13Activate;
  res=res//ToNewCanonical;
  Print["AActivate"];
  res=res/.AActivate;
  res=res//ToNewCanonical;
  Print["LineariseAll"];
  res=res/.LineariseAll;
  res=res//ToNewCanonical;
  res=ToOrderCanonical[res,order];
  Print["SortCovDs"];
  res=res//SortCovDs;
  res=res//ToNewCanonical;
  Print["ThroughInert"];
  res=res//ThroughInert;
  res=res//ToNewCanonical;
  Print["ToTraceReverse"];
  res=res/.ToTraceReverse;
  res=res//ToNewCanonical;
  Print["ThroughInert"];
  res=res//ThroughInert;
  res=res//ToNewCanonical;
  Print["ThroughInert"];
  res=res//ThroughInert;
  res=res//ToNewCanonical;
  (*
```

```
  Print[Style["Full",Green,15]];
  Print[res];
  Print[Style["ToToday",Green,15]];
  Print[ToToday[res]];
  *)
  res];

NewtonianUpToOrder[expr_,order_]:=Module[{res},
  Print[Style["NewtonianUpToOrder",Green,20]];
  res=expr;
  res=StaticUpToOrder[res,order];
  (**)
  Print["ToCoast"];
  res=res/.ToCoast;
  res=res//ToNewCanonical;
  (**)
  (**)
  Print["ToSecondCoast"];
  res=res/.ToSecondCoast;
  res=res//ToNewCanonical;
  (**)
  Print["ToNewtonian"];
  res=res/.ToNewtonian;
  res=res//ToNewCanonical;
  (*
  Print[Style["Full",Green,15]];
  Print[res];
  *)
  Print[Style["ToToday",Green,15]];
  Print[ToToday[res]];
  res];
*)
```

Q-de Sitter space

```
(*
(*
factor=
 epsilonG[-i,-j,-k,-l]epsilonG[i,j,k,m]TKilling[l]TKilling[-m]//ToNewCanonical;
Print[factor];
DefNiceConstantSymbol[ϕ,1];
oldsys={Bet2→-2/3,ψ0→-1/Sqrt[-3Alp6],ϕ0→-3Sqrt[-Bet3/(2Alp6)],H0→0};
spin2=-4 ϕ0 (Bet2-2 Alp6 ψ0^2)/.oldsys//ToCanonical;
```

```
Print[spin2];
spin3=
 16/3 (27 H0^2 Alp6+9 Bet3+18 H0 Alp6 ϕ0+2 Alp6 ϕ0^2) ψ0/.oldsys//ToCanonical;
Print[spin3];
erg3=(-3 H0^2 Alp6 ψ0^2-8/3 H0 ϕ0 (Bet2+Alp6 ψ0^2)-
     2/9 (Bet2 ϕ0^2+(-9 Bet3+2 Alp6 ϕ0^2) ψ0^2))/.oldsys//ToCanonical;
Print[erg3];
erg4=(4/9) (-Bet2 ϕ0^2+27 H0^2 Alp6 ψ0^2+(9 Bet3+4 Alp6 ϕ0^2) ψ0^2-
      3 H0 ϕ0 (Bet2-8 Alp6 ψ0^2))/.oldsys//ToCanonical;
Print[erg4];
equations=spin2==0&&spin3==0&&erg1==0&&erg2==0;
solutions=Quiet[Solve[equations,{H0,ϕ0}]];
Print[solutions];
equations=spin2==0&&spin3==0&&erg3==0&&erg4==0;
solutions=Quiet[Solve[equations,{H0,ϕ0}]];
Print[solutions];
equations=erg3==0&&erg4==0;
solutions=Quiet[Solve[equations,{H0,ϕ0}]];
Print[solutions];
Quit[];
*)
(*
spin2=-4 (1+Sqrt[3] Con3 Sqrt[-Alp6]) CDNewt[-b]+4/3 (-Alp5+Alp6)
   (-2 Con2+3 Sqrt[2] Sqrt[-(Bet3/Alp6)]) CDCDNewt[a,-a]TKilling[-b]+
  4 Sqrt[2] (1-2 Sqrt[3] Con3 Sqrt[-Alp6]) Sqrt[-(Bet3/Alp6)]Newt[] TKilling[-b]-
  4 cAlp5 CD[-a][RLambda5[-b,a]];
spin3=8 Sqrt[2] (Sqrt[3] Sqrt[-Alp6]+3 Con3 Alp6) Sqrt[-(Bet3/Alp6)]CDNewt[-i]+
  ((8 (Alp5-4 Alp6))/(Sqrt[3] Sqrt[-Alp6])-8 Con3 (Alp5+2 Alp6))
   CDCDNewt[a,-a] TKilling[-i]+
  (16 (3 Bet3+2 Sqrt[2] Con2 Alp6 Sqrt[-(Bet3/Alp6)]) Newt[] TKilling[-i])/
   (Sqrt[3] Sqrt[-Alp6])+4 cAlp5 epsilonG[-i,-a,-b,-c] CD[c][RLambda5[a,b]];

DefNiceConstantSymbol[χ,0];
DefNiceConstantSymbol[χ,1];
RLambda5Definition=Antisymmetrize[χ0 TKilling[-i]CDNewt[-j]+
   χ1 epsilonG[-i,-j,k,l]TKilling[-k]CDNewt[-l],{-i,-j}];
RLambda5Activate=MakeRule[{RLambda5[-i,-j],Evaluate[RLambda5Definition]},
  MetricOn→All,ContractMetrics→True];

spin2=spin2/.RLambda5Activate;
spin2=NewtonianUpToOrder[spin2,1];
spin2=spin2//CollectTensors;
```

```
spin3=spin3/.RLambda5Activate;
spin3=NewtonianUpToOrder[spin3,1];
spin3=spin3//CollectTensors;

Print["system of equations"];
spin2=spin2==0//ToConstantSymbolEquations;
Print[spin2];
spin3=spin3==0//ToConstantSymbolEquations;
Print[spin3];
equations=Join[spin2,spin3];
Print[equations];
solutions=Quiet[Solve[equations,{Con2,Con3,χ0,χ1}]];
Print[solutions];
Quit[];
*)
(*
spin2=-4 (1+Sqrt[3] Sqrt[-Alp6] ξ0) CDNewt[-b]+
  2/3 (4 Alp5 ζ0-4 Alp6 ζ0-3 cAlp5 χ0) CDCDNewt[a,-a] TKilling[-b];
spin3=-24 Alp6 ξ1 CDCDCDNewt[-i,a,-a]+
  8/3 ((Sqrt[3] Alp5)/Sqrt[-Alp6]+4 Sqrt[3] Sqrt[-Alp6]+2 Sqrt[3] Sqrt[-Alp6] ζ1-
     3 Alp5 ξ0-6 Alp6 ξ0-3 cAlp5 χ1) CDCDNewt[a,-a]TKilling[-i];


spin2=spin2//CollectTensors;
spin3=spin3//CollectTensors;
Print["system of equations"];
spin2=spin2==0//ToConstantSymbolEquations;
Print[spin2];
spin3=spin3==0//ToConstantSymbolEquations;
Print[spin3];
equations=Join[spin2,spin3];
Print[equations];
solutions=Quiet[Solve[equations,{ζ0,ζ1,ξ0,ξ1,χ0,χ1}]];
Print[solutions];
Quit[];
*)
(*
multiplier=
 1/6 (Sqrt[3]/Sqrt[-Alp6]-3 ζ0) CDNewt[a]epsilonG[-i,-j,-a,-a1] TKilling[a1]-
  1/3 ζ0 CDNewt[-j] TKilling[-i]+1/3 ζ0 CDNewt[-i] TKilling[-j];
spin2=(4-4 Sqrt[3] Sqrt[-Alp6] ξ0)CDNewt[-b]+
  (-((8 Alp5 ζ0)/3)+(8 Alp6 ζ0)/3+2 cAlp5 χ0) CDCDNewt[a,-a]TKilling[-b];
spin3=-24 Alp6 ξ1 CDCDCDNewt[-i,a,-a]+
```

```
  8/3 (-4 Sqrt[3] Sqrt[-Alp6]+(Sqrt[3] Alp5 Alp6)/(-Alp6)^(3/2)-2 Sqrt[3]
      Sqrt[-Alp6] ξ1-3 Alp5 ξ0-6 Alp6 ξ0-3 cAlp5 χ1) CDCDNewt[a,-a]TKilling[-i];

multiplier=multiplier//CollectTensors;
spin2=spin2//CollectTensors;
spin3=spin3//CollectTensors;
Print["system of equations"];
multiplier=multiplier==0//ToConstantSymbolEquations;
Print[multiplier];
spin2=spin2==0//ToConstantSymbolEquations;
Print[spin2];
spin3=spin3==0//ToConstantSymbolEquations;
Print[spin3];
equations=Join[multiplier,spin2,spin3];
Print[equations];
solutions=Quiet[Solve[equations,{ζ0,ζ1,ξ0,ξ1,χ0,χ1}]];
Print[solutions];
Quit[];
*)
(*
spin2=(4+4 Sqrt[3] Sqrt[-Alp6] ζ0)CDNewt[-b]+
   (4 Sqrt[2] Alp5 Sqrt[-(Bet3/Alp6)]-4 Sqrt[2] Alp6 Sqrt[-(Bet3/Alp6)]-
      (8 Alp5 ζ0)/3+(8 Alp6 ζ0)/3+2 cAlp5 χ0) CDCDNewt[a,-a] TKilling[-b]+
   4 Sqrt[2] Sqrt[-(Bet3/Alp6)] (-1+2 Sqrt[3] Sqrt[-Alp6] ζ0) Newt[] TKilling[-b]

    spin3=-24 Alp6 ξ1 CDCDCDNewt[-i,a,-a]+
   8 Sqrt[2] Sqrt[-(Bet3/Alp6)] (Sqrt[3] Sqrt[-Alp6]+3 Alp6 ξ0) CDNewt[-i]+
   8/3 ((Sqrt[3] Alp5)/Sqrt[-Alp6]+4 Sqrt[3] Sqrt[-Alp6]+2 Sqrt[3] Sqrt[-Alp6] ζ1-
      3 Alp5 ξ0-6 Alp6 ξ0-9 Sqrt[2] Alp6 Sqrt[-(Bet3/Alp6)] ξ1-3 cAlp5 χ1)
    CDCDNewt[a,-a] TKilling[-i]+(16 (3 Bet3+2 Sqrt[2] Alp6 Sqrt[-(Bet3/Alp6)] ξ0)
       Newt[] TKilling[-i])/(Sqrt[3] Sqrt[-Alp6])

    DefConstantSymbol[α5,PrintAs→"α̂₅"];
DefConstantSymbol[aα6,PrintAs→"|α̂₆|"];
DefConstantSymbol[β3,PrintAs→"β̂₃"];
DefConstantSymbol[ϵ];

ToPertl={Bet3→β3 ϵ^4,Alp6→-aα6 ϵ^2,Alp5→α5 ϵ^2};

spin2=spin2/.ToPertl//ToCanonical;
spin2=spin2//CollectTensors;
Print[spin2]
 spin3=spin3/.ToPertl//ToCanonical;
```

```
spin3=spin3//CollectTensors;
Print[spin3]
 Print["system of equations"];
spin2=spin2==0//ToConstantSymbolEquations;
spin3=spin3==0//ToConstantSymbolEquations;
equations=Join[spin2,spin3];
Print[equations];
Print["break and power expand"]
 equations=BreakScalars/@equations;
equations=PowerExpand/@equations;
Print[equations];
Print["series in eps"];
equations=(Series[#,{ϵ,0,5}])&/@equations;
Print[equations];

(*
solutions=Quiet[Solve[equations,{ζ0,ζ1,ξ0,ξ1,χ0,χ1}]];
Print[solutions];
*)
Quit[];
*)
*)
```

Components to order

```
(*
DefTensor[AUpTo0[-i,-j,-k],M4];

DefTensor[AUpTo1[-i,-j,-k],M4];
DeclareOrder[AUpTo1[-i,-j,-k],1];

DefTensor[RUpTo0[-i,-j,-k,-l],M4];

DefTensor[RUpTo1[-i,-j,-k,-l],M4];
DeclareOrder[RUpTo1[-i,-j,-k,-l],1];

DefTensor[TUpTo0[-i,-j,-k],M4];

DefTensor[TUpTo1[-i,-j,-k],M4];
DeclareOrder[TUpTo1[-i,-j,-k],1];

DefTensor[DetBHHUpTo0[-i,-j,-k,-l],M4];

DefTensor[DetBHHUpTo1[-i,-j,-k,-l],M4];
```

```
DeclareOrder[DetBHHUpTo1[-i,-j,-k,-l],1];

DefTensor[CDDetBHHUpTo0[-j,n,-k],M4];

DefTensor[CDDetBHHUpTo1[-j,n,-k],M4];
DeclareOrder[CDDetBHHUpTo1[-j,n,-k],1];

DefTensor[CDRUpTo0[-m,-i,-j,-k,-l],M4];

DefTensor[CDRUpTo1[-m,-i,-j,-k,-l],M4];
DeclareOrder[CDRUpTo1[-m,-i,-j,-k,-l],1];

DefTensor[CDTUpTo0[-m,-i,-j,-k],M4];

DefTensor[CDTUpTo1[-m,-i,-j,-k],M4];
DeclareOrder[CDTUpTo1[-m,-i,-j,-k],1];

DefTensor[DetBHUpTo0[-i,n],M4];

DefTensor[DetBHUpTo1[-i,n],M4];
DeclareOrder[DetBHUpTo1[-i,n],1];

If[componentstoorderToggle,
  AUpTo0Definition=StaticUpToOrder[A[-i,-j,-k],0];
  AUpTo0Activate=MakeRule[{AUpTo0[-i,-j,-k],Evaluate[AUpTo0Definition]},
    MetricOn→All,ContractMetrics→True];
  AUpTo1Definition=StaticUpToOrder[A[-i,-j,-k],1];
  AUpTo1Definition=AUpTo1Definition-AUpTo0Definition//ToNewCanonical;
  AUpTo1Activate=MakeRule[{AUpTo1[-i,-j,-k],Evaluate[AUpTo1Definition]},
    MetricOn→All,ContractMetrics→True];
  RUpTo0Definition=StaticUpToOrder[R[-i,-j,-k,-l],0];
  RUpTo0Activate=MakeRule[{RUpTo0[-i,-j,-k,-l],Evaluate[RUpTo0Definition]},
    MetricOn→All,ContractMetrics→True];
  RUpTo1Definition=StaticUpToOrder[R[-i,-j,-k,-l],1];
  RUpTo1Definition=RUpTo1Definition-RUpTo0Definition//ToNewCanonical;
  RUpTo1Activate=MakeRule[{RUpTo1[-i,-j,-k,-l],Evaluate[RUpTo1Definition]},
    MetricOn→All,ContractMetrics→True];
  TUpTo0Definition=StaticUpToOrder[T[-i,-j,-k],0];
  TUpTo0Activate=MakeRule[{TUpTo0[-i,-j,-k],Evaluate[TUpTo0Definition]},
    MetricOn→All,ContractMetrics→True];
  TUpTo1Definition=StaticUpToOrder[T[-i,-j,-k],1];
  TUpTo1Definition=TUpTo1Definition-TUpTo0Definition//ToNewCanonical;
  TUpTo1Activate=MakeRule[{TUpTo1[-i,-j,-k],Evaluate[TUpTo1Definition]},
    MetricOn→All,ContractMetrics→True];
```

```
DetBHHUpTo0Definition=StaticUpToOrder[DetB[]H[-j,n]H[-k,m],0];
DetBHHUpTo0Activate=
 MakeRule[{DetBHHUpTo0[-j,n,-k,m],Evaluate[DetBHHUpTo0Definition]},
  MetricOn→All,ContractMetrics→True];
DetBHHUpTo1Definition=StaticUpToOrder[DetB[]H[-j,n]H[-k,m],1];
DetBHHUpTo1Definition=
 DetBHHUpTo1Definition-DetBHHUpTo0Definition//ToNewCanonical;
DetBHHUpTo1Activate=MakeRule[{DetBHHUpTo1[-j,n,-k,m],
   Evaluate[DetBHHUpTo1Definition]},MetricOn→All,ContractMetrics→True];
CDDetBHHUpTo0Definition=StaticUpToOrder[CD[-m][DetB[]H[-j,n]H[-k,m]],0];
CDDetBHHUpTo0Activate=
 MakeRule[{CDDetBHHUpTo0[-j,n,-k],Evaluate[CDDetBHHUpTo0Definition]},
  MetricOn→All,ContractMetrics→True];
CDDetBHHUpTo1Definition=StaticUpToOrder[CD[-m][DetB[]H[-j,n]H[-k,m]],1];
CDDetBHHUpTo1Definition=
 CDDetBHHUpTo1Definition-CDDetBHHUpTo0Definition//ToNewCanonical;
CDDetBHHUpTo1Activate=MakeRule[{CDDetBHHUpTo1[-j,n,-k],
   Evaluate[CDDetBHHUpTo1Definition]},MetricOn→All,ContractMetrics→True];
CDRUpTo0Definition=StaticUpToOrder[CD[-m][R[-i,-j,-k,-l]],0];
CDRUpTo0Activate=MakeRule[{CDRUpTo0[-m,-i,-j,-k,-l],
   Evaluate[CDRUpTo0Definition]},MetricOn→All,ContractMetrics→True];
CDRUpTo1Definition=StaticUpToOrder[CD[-m][R[-i,-j,-k,-l]],1];
CDRUpTo1Definition=CDRUpTo1Definition-CDRUpTo0Definition//ToNewCanonical;
CDRUpTo1Activate=MakeRule[{CDRUpTo1[-m,-i,-j,-k,-l],
   Evaluate[CDRUpTo1Definition]},MetricOn→All,ContractMetrics→True];
CDTUpTo0Definition=StaticUpToOrder[CD[-m][T[-i,-j,-k]],0];
CDTUpTo0Activate=MakeRule[{CDTUpTo0[-m,-i,-j,-k],Evaluate[CDTUpTo0Definition]},
  MetricOn→All,ContractMetrics→True];
CDTUpTo1Definition=StaticUpToOrder[CD[-m][T[-i,-j,-k]],1];
CDTUpTo1Definition=CDTUpTo1Definition-CDTUpTo0Definition//ToNewCanonical;
CDTUpTo1Activate=MakeRule[{CDTUpTo1[-m,-i,-j,-k],Evaluate[CDTUpTo1Definition]},
  MetricOn→All,ContractMetrics→True];
DetBHUpTo0Definition=StaticUpToOrder[DetB[]H[-i,n],0];
DetBHUpTo0Activate=MakeRule[{DetBHUpTo0[-i,n],Evaluate[DetBHUpTo0Definition]},
  MetricOn→All,ContractMetrics→True];
DetBHUpTo1Definition=StaticUpToOrder[DetB[]H[-i,n],1];
DetBHUpTo1Definition=
 DetBHUpTo1Definition-DetBHUpTo0Definition//ToNewCanonical;
DetBHUpTo1Activate=MakeRule[{DetBHUpTo1[-i,n],Evaluate[DetBHUpTo1Definition]},
  MetricOn→All,ContractMetrics→True];

HeavyActivate=Join[AUpTo0Activate,AUpTo1Activate,RUpTo0Activate,RUpTo1Activate,
  TUpTo0Activate,TUpTo1Activate,CDDetBHHUpTo0Activate,DetBHHUpTo0Activate,
  DetBHHUpTo1Activate,CDDetBHHUpTo1Activate,CDRUpTo0Activate,CDRUpTo1Activate,
```

```
        CDTUpTo0Activate,CDTUpTo1Activate,DetBHUpTo0Activate,DetBHUpTo1Activate];

    DumpSave[NotebookDirectory[]<>"mx_cache/componentstoorder.mx",{HeavyActivate}];
    Print["componentstoorder done"];
    Quit[];
  ]
  MyImport["componentstoorder.mx"];
*)
```

Soldering and Welding

```
In[•]:= (*
WeldParts[left_,middle_,right_,order_]:=Module[{res},
  Print[Style["Welding partial expressions",Orange,20]];
  res=middle;
  res=res/.Theory/.PActivate;
  res=res//ToNewCanonical;
  res=left res/.HeavyActivate;
  res=res//ToNewCanonical;
  res=res right/.HeavyActivate;
  res=res//ToNewCanonical;
  res=NewtonianUpToOrder[res,order];
  res];

AddToResult[expr_,total_]:=Module[{res},
  Print[Style["Soldering to total",Orange,20]];
  res=total+expr;
  res=res//ToNewCanonical;
  Print[res];
  res];
*)
```

Generalised momenta formula

```
(*
BPiGDefinition=2DetB[](2(Bet1 PT1[-i,-j,-k,a,b,c]
          +Bet2 PT2[-i,-j,-k,a,b,c]
          +Bet3 PT3[-i,-j,-k,a,b,c])T[-a,-b,-c]+
       (cBet1 PT1[-i,-j,-k,a,b,c]
          +cBet2 PT2[-i,-j,-k,a,b,c]
          +cBet3 PT3[-i,-j,-k,a,b,c])TLambda[-a,-b,-c])/.Theory//ToNewCanonical;
BPiGActivate=MakeRule[{BPiG[-i,-j,-k],Evaluate[BPiGDefinition]},
  MetricOn→All,ContractMetrics→True];
```

```
APiGDefinition=4DetB[](2(Alp1 PR1[-i,-j,-k,-l,a,b,c,d]
          +Alp2 PR2[-i,-j,-k,-l,a,b,c,d]
          +Alp3 PR3[-i,-j,-k,-l,a,b,c,d]
          +Alp4 PR4[-i,-j,-k,-l,a,b,c,d]
          +Alp5 PR5[-i,-j,-k,-l,a,b,c,d]
          +Alp6 PR6[-i,-j,-k,-l,a,b,c,d])R[-a,-b,-c,-d]+
       (cAlp1 PR1[-i,-j,-k,-l,a,b,c,d]
          +cAlp2 PR2[-i,-j,-k,-l,a,b,c,d]
          +cAlp3 PR3[-i,-j,-k,-l,a,b,c,d]
          +cAlp4 PR4[-i,-j,-k,-l,a,b,c,d]
          +cAlp5 PR5[-i,-j,-k,-l,a,b,c,d]
          +cAlp6 PR6[-i,-j,-k,-l,a,b,c,d])RLambda[-a,-b,-c,-d])/.Theory//
  ToNewCanonical;
APiGActivate=MakeRule[{APiG[-i,-j,-k,-l],Evaluate[APiGDefinition]},
  MetricOn→All,ContractMetrics→True];

LagrangianDefinition=(T[i,j,k]((Bet1 PT1[-i,-j,-k,a,b,c]
            +Bet2 PT2[-i,-j,-k,a,b,c]
            +Bet3 PT3[-i,-j,-k,a,b,c])T[-a,-b,-c]+
         (cBet1 PT1[-i,-j,-k,a,b,c]
            +cBet2 PT2[-i,-j,-k,a,b,c]
            +cBet3 PT3[-i,-j,-k,a,b,c])TLambda[-a,-b,-c])+
      R[i,j,k,l]((Alp1 PR1[-i,-j,-k,-l,a,b,c,d]
            +Alp2 PR2[-i,-j,-k,-l,a,b,c,d]
            +Alp3 PR3[-i,-j,-k,-l,a,b,c,d]
            +Alp4 PR4[-i,-j,-k,-l,a,b,c,d]
            +Alp5 PR5[-i,-j,-k,-l,a,b,c,d]
            +Alp6 PR6[-i,-j,-k,-l,a,b,c,d])R[-a,-b,-c,-d]+
         (cAlp1 PR1[-i,-j,-k,-l,a,b,c,d]
            +cAlp2 PR2[-i,-j,-k,-l,a,b,c,d]
            +cAlp3 PR3[-i,-j,-k,-l,a,b,c,d]
            +cAlp4 PR4[-i,-j,-k,-l,a,b,c,d]
            +cAlp5 PR5[-i,-j,-k,-l,a,b,c,d]
            +cAlp6 PR6[-i,-j,-k,-l,a,b,c,d])RLambda[-a,-b,-c,-d]))/.Theory//
  ToNewCanonical;
LagrangianActivate=MakeRule[{Lagrangian[],Evaluate[LagrangianDefinition]},
  MetricOn→All,ContractMetrics→True];
*)
```

Constructing $\tau^{v}{}_{i}$ equation

```
(*
ETensorDefinition=0;
```

```
(*derivative part of derivative of translational momentum*)

tmp=-2(2(Bet1 PT1[-i,j,k,a,b,c]
       +Bet2 PT2[-i,j,k,a,b,c]
       +Bet3 PT3[-i,j,k,a,b,c]));
tmp=WeldParts[CDDetBHHUpTo0[-j,n,-k],tmp,TUpTo0[-a,-b,-c],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

tmp=-2(2(Bet1 PT1[-i,j,k,a,b,c]
       +Bet2 PT2[-i,j,k,a,b,c]
       +Bet3 PT3[-i,j,k,a,b,c]));
tmp=WeldParts[CDDetBHHUpTo0[-j,n,-k],tmp,TUpTo1[-a,-b,-c],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

tmp=-2((cBet1 PT1[-i,j,k,a,b,c]
      +cBet2 PT2[-i,j,k,a,b,c]
      +cBet3 PT3[-i,j,k,a,b,c]));
tmp=WeldParts[CDDetBHHUpTo0[-j,n,-k],tmp,TLambda[-a,-b,-c],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

tmp=-2(2(Bet1 PT1[-i,j,k,a,b,c]
       +Bet2 PT2[-i,j,k,a,b,c]
       +Bet3 PT3[-i,j,k,a,b,c]));
tmp=WeldParts[CDDetBHHUpTo1[-j,n,-k],tmp,TUpTo0[-a,-b,-c],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

tmp=-2(2(Bet1 PT1[-i,j,k,a,b,c]
       +Bet2 PT2[-i,j,k,a,b,c]
       +Bet3 PT3[-i,j,k,a,b,c]));
tmp=WeldParts[DetBHHUpTo0[-j,n,-k,m],tmp,CDTUpTo0[-m,-a,-b,-c],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

tmp=-2(2(Bet1 PT1[-i,j,k,a,b,c]
       +Bet2 PT2[-i,j,k,a,b,c]
       +Bet3 PT3[-i,j,k,a,b,c]));
tmp=WeldParts[DetBHHUpTo0[-j,n,-k,m],tmp,CDTUpTo1[-m,-a,-b,-c],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

tmp=-2((cBet1 PT1[-i,j,k,a,b,c]
      +cBet2 PT2[-i,j,k,a,b,c]
      +cBet3 PT3[-i,j,k,a,b,c]));
tmp=WeldParts[DetBHHUpTo0[-j,n,-k,m],tmp,CD[-m][TLambda[-a,-b,-c]],1];
```

```
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

tmp=-2(2(Bet1 PT1[-i,j,k,a,b,c]
       +Bet2 PT2[-i,j,k,a,b,c]
       +Bet3 PT3[-i,j,k,a,b,c]));
tmp=WeldParts[DetBHHUpTo1[-j,n,-k,m],tmp,CDTUpTo0[-m,-a,-b,-c],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

(*connection part of covariant derivative of translational momentum*)

tmp=2(2(Bet1 PT1[-u,j,k,a,b,c]
       +Bet2 PT2[-u,j,k,a,b,c]
       +Bet3 PT3[-u,j,k,a,b,c]));
tmp=WeldParts[AUpTo0[u,-i,-m]DetBHHUpTo0[-j,n,-k,m],tmp,TUpTo0[-a,-b,-c],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

tmp=2(2(Bet1 PT1[-u,j,k,a,b,c]
       +Bet2 PT2[-u,j,k,a,b,c]
       +Bet3 PT3[-u,j,k,a,b,c]));
tmp=WeldParts[AUpTo0[u,-i,-m]DetBHHUpTo0[-j,n,-k,m],tmp,TUpTo1[-a,-b,-c],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

tmp=2((cBet1 PT1[-u,j,k,a,b,c]
      +cBet2 PT2[-u,j,k,a,b,c]
      +cBet3 PT3[-u,j,k,a,b,c]));
tmp=WeldParts[AUpTo0[u,-i,-m]DetBHHUpTo0[-j,n,-k,m],tmp,TLambda[-a,-b,-c],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

tmp=2(2(Bet1 PT1[-u,j,k,a,b,c]
       +Bet2 PT2[-u,j,k,a,b,c]
       +Bet3 PT3[-u,j,k,a,b,c]));
tmp=WeldParts[AUpTo0[u,-i,-m]DetBHHUpTo1[-j,n,-k,m],tmp,TUpTo0[-a,-b,-c],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

tmp=2(2(Bet1 PT1[-u,j,k,a,b,c]
       +Bet2 PT2[-u,j,k,a,b,c]
       +Bet3 PT3[-u,j,k,a,b,c]));
tmp=WeldParts[AUpTo1[u,-i,-m]DetBHHUpTo0[-j,n,-k,m],tmp,TUpTo0[-a,-b,-c],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

(*Lagrangian*)

tmp=-((Bet1 PT1[-p,-q,-k,a,b,c]
```

```
      +Bet2 PT2[-p,-q,-k,a,b,c]
      +Bet3 PT3[-p,-q,-k,a,b,c]));
tmp=WeldParts[DetB[]T[p,q,k],tmp,T[-a,-b,-c]H[-i,n],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

tmp=-((cBet1 PT1[-p,-q,-k,a,b,c]
      +cBet2 PT2[-p,-q,-k,a,b,c]
      +cBet3 PT3[-p,-q,-k,a,b,c]));
tmp=WeldParts[DetB[]T[p,q,k],tmp,TLambda[-a,-b,-c]H[-i,n],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

tmp=-(Alp1 PR1[-p,-q,-k,-l,a,b,c,d]
     +Alp2 PR2[-p,-q,-k,-l,a,b,c,d]
     +Alp3 PR3[-p,-q,-k,-l,a,b,c,d]
     +Alp4 PR4[-p,-q,-k,-l,a,b,c,d]
     +Alp5 PR5[-p,-q,-k,-l,a,b,c,d]
     +Alp6 PR6[-p,-q,-k,-l,a,b,c,d]);
tmp=WeldParts[DetBHUpTo0[-i,n]RUpTo0[p,q,k,l],tmp,RUpTo0[-a,-b,-c,-d],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

tmp=-(Alp1 PR1[-p,-q,-k,-l,a,b,c,d]
     +Alp2 PR2[-p,-q,-k,-l,a,b,c,d]
     +Alp3 PR3[-p,-q,-k,-l,a,b,c,d]
     +Alp4 PR4[-p,-q,-k,-l,a,b,c,d]
     +Alp5 PR5[-p,-q,-k,-l,a,b,c,d]
     +Alp6 PR6[-p,-q,-k,-l,a,b,c,d]);
tmp=WeldParts[DetBHUpTo0[-i,n]RUpTo0[p,q,k,l],tmp,RUpTo1[-a,-b,-c,-d],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

tmp=-(Alp1 PR1[-p,-q,-k,-l,a,b,c,d]
     +Alp2 PR2[-p,-q,-k,-l,a,b,c,d]
     +Alp3 PR3[-p,-q,-k,-l,a,b,c,d]
     +Alp4 PR4[-p,-q,-k,-l,a,b,c,d]
     +Alp5 PR5[-p,-q,-k,-l,a,b,c,d]
     +Alp6 PR6[-p,-q,-k,-l,a,b,c,d]);
tmp=WeldParts[DetBHUpTo0[-i,n]RUpTo1[p,q,k,l],tmp,RUpTo0[-a,-b,-c,-d],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

tmp=-(Alp1 PR1[-p,-q,-k,-l,a,b,c,d]
     +Alp2 PR2[-p,-q,-k,-l,a,b,c,d]
     +Alp3 PR3[-p,-q,-k,-l,a,b,c,d]
     +Alp4 PR4[-p,-q,-k,-l,a,b,c,d]
     +Alp5 PR5[-p,-q,-k,-l,a,b,c,d]
```

```
    +Alp6 PR6[-p,-q,-k,-l,a,b,c,d]);
tmp=WeldParts[DetBHUpTo1[-i,n]RUpTo0[p,q,k,l],tmp,RUpTo0[-a,-b,-c,-d],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

tmp=-(cAlp1 PR1[-p,-q,-k,-l,a,b,c,d]
    +cAlp2 PR2[-p,-q,-k,-l,a,b,c,d]
    +cAlp3 PR3[-p,-q,-k,-l,a,b,c,d]
    +cAlp4 PR4[-p,-q,-k,-l,a,b,c,d]
    +cAlp5 PR5[-p,-q,-k,-l,a,b,c,d]
    +cAlp6 PR6[-p,-q,-k,-l,a,b,c,d]);
tmp=WeldParts[DetBHUpTo0[-i,n]RUpTo0[p,q,k,l],tmp,RLambda[-a,-b,-c,-d],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

(*torsion and translational momentum*)

tmp=2(2(Bet1 PT1[-p,k,l,a,b,c]
      +Bet2 PT2[-p,k,l,a,b,c]
      +Bet3 PT3[-p,k,l,a,b,c]));
tmp=WeldParts[DetB[]T[p,-k,-i],tmp,T[-a,-b,-c]H[-l,n],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

tmp=2((cBet1 PT1[-p,k,l,a,b,c]
     +cBet2 PT2[-p,k,l,a,b,c]
     +cBet3 PT3[-p,k,l,a,b,c]));
tmp=WeldParts[DetB[]T[p,-k,-i],tmp,TLambda[-a,-b,-c]H[-l,n],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

(*Riemann--Cartan curvature and rotational momentum*)

tmp=(1/2)8(Alp1 PR1[-p,-q,k,l,a,b,c,d]
    +Alp2 PR2[-p,-q,k,l,a,b,c,d]
    +Alp3 PR3[-p,-q,k,l,a,b,c,d]
    +Alp4 PR4[-p,-q,k,l,a,b,c,d]
    +Alp5 PR5[-p,-q,k,l,a,b,c,d]
    +Alp6 PR6[-p,-q,k,l,a,b,c,d]);
tmp=WeldParts[DetBHUpTo0[-l,n]RUpTo0[p,q,-k,-i],tmp,RUpTo0[-a,-b,-c,-d],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

tmp=(1/2)8(Alp1 PR1[-p,-q,k,l,a,b,c,d]
    +Alp2 PR2[-p,-q,k,l,a,b,c,d]
    +Alp3 PR3[-p,-q,k,l,a,b,c,d]
    +Alp4 PR4[-p,-q,k,l,a,b,c,d]
    +Alp5 PR5[-p,-q,k,l,a,b,c,d]
```

```
    +Alp6 PR6[-p,-q,k,l,a,b,c,d]);
tmp=WeldParts[DetBHUpTo0[-l,n]RUpTo0[p,q,-k,-i],tmp,RUpTo1[-a,-b,-c,-d],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

tmp=(1/2)8(Alp1 PR1[-p,-q,k,l,a,b,c,d]
    +Alp2 PR2[-p,-q,k,l,a,b,c,d]
    +Alp3 PR3[-p,-q,k,l,a,b,c,d]
    +Alp4 PR4[-p,-q,k,l,a,b,c,d]
    +Alp5 PR5[-p,-q,k,l,a,b,c,d]
    +Alp6 PR6[-p,-q,k,l,a,b,c,d]);
tmp=WeldParts[DetBHUpTo0[-l,n]RUpTo1[p,q,-k,-i],tmp,RUpTo0[-a,-b,-c,-d],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

tmp=(1/2)8(Alp1 PR1[-p,-q,k,l,a,b,c,d]
    +Alp2 PR2[-p,-q,k,l,a,b,c,d]
    +Alp3 PR3[-p,-q,k,l,a,b,c,d]
    +Alp4 PR4[-p,-q,k,l,a,b,c,d]
    +Alp5 PR5[-p,-q,k,l,a,b,c,d]
    +Alp6 PR6[-p,-q,k,l,a,b,c,d]);
tmp=WeldParts[DetBHUpTo1[-l,n]RUpTo0[p,q,-k,-i],tmp,RUpTo0[-a,-b,-c,-d],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

tmp=(1/2)4(cAlp1 PR1[-p,-q,k,l,a,b,c,d]
    +cAlp2 PR2[-p,-q,k,l,a,b,c,d]
    +cAlp3 PR3[-p,-q,k,l,a,b,c,d]
    +cAlp4 PR4[-p,-q,k,l,a,b,c,d]
    +cAlp5 PR5[-p,-q,k,l,a,b,c,d]
    +cAlp6 PR6[-p,-q,k,l,a,b,c,d]);
tmp=WeldParts[DetBHUpTo0[-l,n]RUpTo0[p,q,-k,-i],tmp,RLambda[-a,-b,-c,-d],1];
ETensorDefinition=AddToResult[ETensorDefinition,tmp];

DumpSave[NotebookDirectory[]<>"mx_cache/etensor.mx",{ETensorDefinition}];
Print["etensor done"];
Quit[];
*)
(*
MyImport["etensor.mx"];
ETensorActivate=MakeRule[{ETensor[-i,n],Evaluate[ETensorDefinition]},
  MetricOn→All,ContractMetrics→True];
*)
```

Constructing $\sigma^{v}{}_{ij}$ equation

```
(*
STensorDefinition=0;

(*derivative part of derivative of translational momentum*)
(**)
tmp=-4(2(Alp1 PR1[-i,-j,k,l,a,b,c,d]
       +Alp2 PR2[-i,-j,k,l,a,b,c,d]
       +Alp3 PR3[-i,-j,k,l,a,b,c,d]
       +Alp4 PR4[-i,-j,k,l,a,b,c,d]
       +Alp5 PR5[-i,-j,k,l,a,b,c,d]
       +Alp6 PR6[-i,-j,k,l,a,b,c,d]));
tmp=WeldParts[CDDetBHHUpTo0[-k,n,-l],tmp,RUpTo0[-a,-b,-c,-d],1];
STensorDefinition=AddToResult[STensorDefinition,tmp];

tmp=-4(2(Alp1 PR1[-i,-j,k,l,a,b,c,d]
       +Alp2 PR2[-i,-j,k,l,a,b,c,d]
       +Alp3 PR3[-i,-j,k,l,a,b,c,d]
       +Alp4 PR4[-i,-j,k,l,a,b,c,d]
       +Alp5 PR5[-i,-j,k,l,a,b,c,d]
       +Alp6 PR6[-i,-j,k,l,a,b,c,d]));
tmp=WeldParts[CDDetBHHUpTo0[-k,n,-l],tmp,RUpTo1[-a,-b,-c,-d],1];
STensorDefinition=AddToResult[STensorDefinition,tmp];

tmp=-4((cAlp1 PR1[-i,-j,k,l,a,b,c,d]
      +cAlp2 PR2[-i,-j,k,l,a,b,c,d]
      +cAlp3 PR3[-i,-j,k,l,a,b,c,d]
      +cAlp4 PR4[-i,-j,k,l,a,b,c,d]
      +cAlp5 PR5[-i,-j,k,l,a,b,c,d]
      +cAlp6 PR6[-i,-j,k,l,a,b,c,d]));
tmp=WeldParts[CDDetBHHUpTo0[-k,n,-l],tmp,RLambda[-a,-b,-c,-d],1];
STensorDefinition=AddToResult[STensorDefinition,tmp];

tmp=-4(2(Alp1 PR1[-i,-j,k,l,a,b,c,d]
       +Alp2 PR2[-i,-j,k,l,a,b,c,d]
       +Alp3 PR3[-i,-j,k,l,a,b,c,d]
       +Alp4 PR4[-i,-j,k,l,a,b,c,d]
       +Alp5 PR5[-i,-j,k,l,a,b,c,d]
       +Alp6 PR6[-i,-j,k,l,a,b,c,d]));
tmp=WeldParts[CDDetBHHUpTo1[-k,n,-l],tmp,RUpTo0[-a,-b,-c,-d],1];
STensorDefinition=AddToResult[STensorDefinition,tmp];

tmp=-4(2(Alp1 PR1[-i,-j,k,l,a,b,c,d]
       +Alp2 PR2[-i,-j,k,l,a,b,c,d]
```

```
        +Alp3 PR3[-i,-j,k,l,a,b,c,d]
        +Alp4 PR4[-i,-j,k,l,a,b,c,d]
        +Alp5 PR5[-i,-j,k,l,a,b,c,d]
        +Alp6 PR6[-i,-j,k,l,a,b,c,d]));
tmp=WeldParts[DetBHHUpTo0[-k,n,-l,m],tmp,CDRUpTo0[-m,-a,-b,-c,-d],1];
STensorDefinition=AddToResult[STensorDefinition,tmp];

tmp=-4(2(Alp1 PR1[-i,-j,k,l,a,b,c,d]
        +Alp2 PR2[-i,-j,k,l,a,b,c,d]
        +Alp3 PR3[-i,-j,k,l,a,b,c,d]
        +Alp4 PR4[-i,-j,k,l,a,b,c,d]
        +Alp5 PR5[-i,-j,k,l,a,b,c,d]
        +Alp6 PR6[-i,-j,k,l,a,b,c,d]));
tmp=WeldParts[DetBHHUpTo0[-k,n,-l,m],tmp,CDRUpTo1[-m,-a,-b,-c,-d],1];
STensorDefinition=AddToResult[STensorDefinition,tmp];

tmp=-4((cAlp1 PR1[-i,-j,k,l,a,b,c,d]
       +cAlp2 PR2[-i,-j,k,l,a,b,c,d]
       +cAlp3 PR3[-i,-j,k,l,a,b,c,d]
       +cAlp4 PR4[-i,-j,k,l,a,b,c,d]
       +cAlp5 PR5[-i,-j,k,l,a,b,c,d]
       +cAlp6 PR6[-i,-j,k,l,a,b,c,d]));
tmp=WeldParts[DetBHHUpTo0[-k,n,-l,m],tmp,CD[-m][RLambda[-a,-b,-c,-d]],1];
STensorDefinition=AddToResult[STensorDefinition,tmp];

tmp=-4(2(Alp1 PR1[-i,-j,k,l,a,b,c,d]
        +Alp2 PR2[-i,-j,k,l,a,b,c,d]
        +Alp3 PR3[-i,-j,k,l,a,b,c,d]
        +Alp4 PR4[-i,-j,k,l,a,b,c,d]
        +Alp5 PR5[-i,-j,k,l,a,b,c,d]
        +Alp6 PR6[-i,-j,k,l,a,b,c,d]));
tmp=WeldParts[DetBHHUpTo1[-k,n,-l,m],tmp,CDRUpTo0[-m,-a,-b,-c,-d],1];
STensorDefinition=AddToResult[STensorDefinition,tmp];
(**)
(*connection part of covariant derivative of translational momentum first index*)

tmp=4(2(Alp1 PR1[-u,-j,k,l,a,b,c,d]
        +Alp2 PR2[-u,-j,k,l,a,b,c,d]
        +Alp3 PR3[-u,-j,k,l,a,b,c,d]
        +Alp4 PR4[-u,-j,k,l,a,b,c,d]
        +Alp5 PR5[-u,-j,k,l,a,b,c,d]
        +Alp6 PR6[-u,-j,k,l,a,b,c,d]));
tmp=WeldParts[AUpTo0[u,-i,-m]DetBHHUpTo0[-k,n,-l,m],tmp,RUpTo0[-a,-b,-c,-d],1];
```

```
STensorDefinition=AddToResult[STensorDefinition,tmp];

tmp=4(2(Alp1 PR1[-u,-j,k,l,a,b,c,d]
       +Alp2 PR2[-u,-j,k,l,a,b,c,d]
       +Alp3 PR3[-u,-j,k,l,a,b,c,d]
       +Alp4 PR4[-u,-j,k,l,a,b,c,d]
       +Alp5 PR5[-u,-j,k,l,a,b,c,d]
       +Alp6 PR6[-u,-j,k,l,a,b,c,d]));
tmp=WeldParts[AUpTo0[u,-i,-m]DetBHHUpTo0[-k,n,-l,m],tmp,RUpTo1[-a,-b,-c,-d],1];
STensorDefinition=AddToResult[STensorDefinition,tmp];

tmp=4((cAlp1 PR1[-u,-j,k,l,a,b,c,d]
      +cAlp2 PR2[-u,-j,k,l,a,b,c,d]
      +cAlp3 PR3[-u,-j,k,l,a,b,c,d]
      +cAlp4 PR4[-u,-j,k,l,a,b,c,d]
      +cAlp5 PR5[-u,-j,k,l,a,b,c,d]
      +cAlp6 PR6[-u,-j,k,l,a,b,c,d]));
tmp=WeldParts[AUpTo0[u,-i,-m]DetBHHUpTo0[-k,n,-l,m],tmp,RLambda[-a,-b,-c,-d],1];
STensorDefinition=AddToResult[STensorDefinition,tmp];

tmp=4(2(Alp1 PR1[-u,-j,k,l,a,b,c,d]
       +Alp2 PR2[-u,-j,k,l,a,b,c,d]
       +Alp3 PR3[-u,-j,k,l,a,b,c,d]
       +Alp4 PR4[-u,-j,k,l,a,b,c,d]
       +Alp5 PR5[-u,-j,k,l,a,b,c,d]
       +Alp6 PR6[-u,-j,k,l,a,b,c,d]));
tmp=WeldParts[AUpTo0[u,-i,-m]DetBHHUpTo1[-k,n,-l,m],tmp,RUpTo0[-a,-b,-c,-d],1];
STensorDefinition=AddToResult[STensorDefinition,tmp];

tmp=4(2(Alp1 PR1[-u,-j,k,l,a,b,c,d]
       +Alp2 PR2[-u,-j,k,l,a,b,c,d]
       +Alp3 PR3[-u,-j,k,l,a,b,c,d]
       +Alp4 PR4[-u,-j,k,l,a,b,c,d]
       +Alp5 PR5[-u,-j,k,l,a,b,c,d]
       +Alp6 PR6[-u,-j,k,l,a,b,c,d]));
tmp=WeldParts[AUpTo1[u,-i,-m]DetBHHUpTo0[-k,n,-l,m],tmp,RUpTo0[-a,-b,-c,-d],1];
STensorDefinition=AddToResult[STensorDefinition,tmp];

(*connection part of covariant derivative
 of translational momentum second index*)

tmp=4(2(Alp1 PR1[-i,-u,k,l,a,b,c,d]
       +Alp2 PR2[-i,-u,k,l,a,b,c,d]
```

```
        +Alp3 PR3[-i,-u,k,l,a,b,c,d]
        +Alp4 PR4[-i,-u,k,l,a,b,c,d]
        +Alp5 PR5[-i,-u,k,l,a,b,c,d]
        +Alp6 PR6[-i,-u,k,l,a,b,c,d]));
tmp=WeldParts[AUpTo0[u,-j,-m]DetBHHUpTo0[-k,n,-l,m],tmp,RUpTo0[-a,-b,-c,-d],1];
STensorDefinition=AddToResult[STensorDefinition,tmp];

tmp=4(2(Alp1 PR1[-i,-u,k,l,a,b,c,d]
        +Alp2 PR2[-i,-u,k,l,a,b,c,d]
        +Alp3 PR3[-i,-u,k,l,a,b,c,d]
        +Alp4 PR4[-i,-u,k,l,a,b,c,d]
        +Alp5 PR5[-i,-u,k,l,a,b,c,d]
        +Alp6 PR6[-i,-u,k,l,a,b,c,d]));
tmp=WeldParts[AUpTo0[u,-j,-m]DetBHHUpTo0[-k,n,-l,m],tmp,RUpTo1[-a,-b,-c,-d],1];
STensorDefinition=AddToResult[STensorDefinition,tmp];

tmp=4((cAlp1 PR1[-i,-u,k,l,a,b,c,d]
      +cAlp2 PR2[-i,-u,k,l,a,b,c,d]
      +cAlp3 PR3[-i,-u,k,l,a,b,c,d]
      +cAlp4 PR4[-i,-u,k,l,a,b,c,d]
      +cAlp5 PR5[-i,-u,k,l,a,b,c,d]
      +cAlp6 PR6[-i,-u,k,l,a,b,c,d]));
tmp=WeldParts[AUpTo0[u,-j,-m]DetBHHUpTo0[-k,n,-l,m],tmp,RLambda[-a,-b,-c,-d],1];
STensorDefinition=AddToResult[STensorDefinition,tmp];

tmp=4(2(Alp1 PR1[-i,-u,k,l,a,b,c,d]
        +Alp2 PR2[-i,-u,k,l,a,b,c,d]
        +Alp3 PR3[-i,-u,k,l,a,b,c,d]
        +Alp4 PR4[-i,-u,k,l,a,b,c,d]
        +Alp5 PR5[-i,-u,k,l,a,b,c,d]
        +Alp6 PR6[-i,-u,k,l,a,b,c,d]));
tmp=WeldParts[AUpTo0[u,-j,-m]DetBHHUpTo1[-k,n,-l,m],tmp,RUpTo0[-a,-b,-c,-d],1];
STensorDefinition=AddToResult[STensorDefinition,tmp];

tmp=4(2(Alp1 PR1[-i,-u,k,l,a,b,c,d]
        +Alp2 PR2[-i,-u,k,l,a,b,c,d]
        +Alp3 PR3[-i,-u,k,l,a,b,c,d]
        +Alp4 PR4[-i,-u,k,l,a,b,c,d]
        +Alp5 PR5[-i,-u,k,l,a,b,c,d]
        +Alp6 PR6[-i,-u,k,l,a,b,c,d]));
tmp=WeldParts[AUpTo1[u,-j,-m]DetBHHUpTo0[-k,n,-l,m],tmp,RUpTo0[-a,-b,-c,-d],1];
STensorDefinition=AddToResult[STensorDefinition,tmp];
```

```
(**)
(*skew-symmetrised translational momentum*)

tmp=2Antisymmetrize[2(2(Bet1 PT1[-i,-j,n,a,b,c]
        +Bet2 PT2[-i,-j,n,a,b,c]
        +Bet3 PT3[-i,-j,n,a,b,c])),{-i,-j}];
tmp=WeldParts[DetB[],tmp,T[-a,-b,-c],1];
STensorDefinition=AddToResult[STensorDefinition,tmp];

tmp=2Antisymmetrize[2((cBet1 PT1[-i,-j,n,a,b,c]
       +cBet2 PT2[-i,-j,n,a,b,c]
       +cBet3 PT3[-i,-j,n,a,b,c])),{-i,-j}];
tmp=WeldParts[DetB[],tmp,TLambda[-a,-b,-c],1];
STensorDefinition=AddToResult[STensorDefinition,tmp];

DumpSave[NotebookDirectory[]<>"mx_cache/stensor.mx",{STensorDefinition}];
Print["stensor done"];
Quit[];
*)
(*
MyImport["stensor.mx"];
STensorActivate=MakeRule[{STensor[n,-i,-j],Evaluate[STensorDefinition]},
  MetricOn→All,ContractMetrics→True];
*)
```

Constructing $\lambda^{ij}{}_{kl}$ equation

```
(*
RLambdaEquationDefinition=0;

tmp=Antisymmetrize[R[-i,l,-j,-l],{-i,-j}];
tmp=WeldParts[1,tmp,1,1];
RLambdaEquationDefinition=AddToResult[RLambdaEquationDefinition,tmp];
(*
DumpSave[NotebookDirectory[]<>"mx_cache/rlambdaequation.mx",
 {RLambdaEquationDefinition}];
Print["rlambdaequation done"];
Quit[];
(**)
(**)
MyImport["rlambdaequation.mx"];
*)
RLambdaEquationActivate=
 MakeRule[{RLambdaEquation[-i,-j],Evaluate[RLambdaEquationDefinition]},
  MetricOn→All,ContractMetrics→True];
*)
```

Examination of field equations

```
(*
HeavierActivate=Join[ETensorActivate,STensorActivate,RLambdaEquationActivate];

Print[Style["Linearised equations",Red,40]];
(**)
Print[Style["Riemann-Cartan multiplier equation",Red,20]];

tmp=RLambdaEquation[-i,-j];
Print[tmp];
tmp=tmp/.PActivate;
tmp=tmp/.HeavierActivate;
NewtonianUpToOrder[tmp,1];
NewtonianUpToOrder[tmp,0];

(**)

Print[Style["Part 1 spin equation",Red,20]];
```

```
tmp=PT1[-i,-j,-k,-a,b,c]B[a,-m]STensor[m,-b,-c];
Print[tmp];
tmp=tmp/.PActivate;
tmp=tmp/.HeavierActivate;
NewtonianUpToOrder[tmp,1];
NewtonianUpToOrder[tmp,0];

Print[Style["Difference from part 1 spin equation",Red,20]];

tmp=(2 cBet1 epsilonG[-i,-a,-a1,-b] TKilling[a] (3/(8 cBet1))
      PT1[-j,a1,b,-r,p,q]B[r,-m]STensor[m,-p,-q])/(3 Sqrt[3] Sqrt[-Alp6])-
  CD[-k][PT1[-i,k,-j,-a,b,c]B[a,-m]STensor[m,-b,-c]];
Print[tmp];
tmp=tmp/.PActivate;
tmp=tmp/.HeavierActivate;
diff=NewtonianUpToOrder[tmp,1];
NewtonianUpToOrder[tmp,0];

(**)

Print[Style["Part 2 spin equation",Red,20]];

tmp=B[c,-m]STensor[m,-b,-c];
Print[tmp];
tmp=tmp/.HeavierActivate;
NewtonianUpToOrder[tmp,1];
NewtonianUpToOrder[tmp,0];

Print[Style["Part 3 spin equation",Red,20]];

tmp=epsilonG[-i,-a,b,c]B[a,-m]STensor[m,-b,-c];
Print[tmp];
tmp=tmp/.HeavierActivate;
NewtonianUpToOrder[tmp,1];
NewtonianUpToOrder[tmp,0];

(*Quit[];*)

Print[Style["Full stress-energy equation",Red,20]];

tmp=B[-i,-m]ETensor[m,-j];
Print[tmp];
tmp=tmp/.HeavierActivate;
```

```
NewtonianUpToOrder[tmp,1];
NewtonianUpToOrder[tmp,0];

Print[Style["Exclusionary stress-energy equation",Red,20]];

tmp=B[-i,-m]ETensor[m,-j]-diff;
Print[tmp];
tmp=tmp/.HeavierActivate;
NewtonianUpToOrder[tmp,1];
NewtonianUpToOrder[tmp,0];

Print[Style["Trace of full stress-energy equation",Red,20]];

tmp=ETensor[n,-i]B[i,-n];
Print[tmp];
tmp=tmp/.HeavierActivate;
NewtonianUpToOrder[tmp,1];
NewtonianUpToOrder[tmp,0];

Quit[];

Print[Style["Full Belinfante equation",Red,20]];

tmp=
 ETensor[-i,-j]-(1/2)CD[-k][STensor[-i,-j,k]+STensor[-j,-i,k]-STensor[k,-i,-j]];
Print[tmp];
tmp=tmp/.HeavierActivate;
NewtonianUpToOrder[tmp,1];
NewtonianUpToOrder[tmp,0];

Print[Style["Symmetric Belinfante equation",Red,20]];

tmp=Symmetrize[ETensor[-i,-j]-
    (1/2)CD[-k][STensor[-i,-j,k]+STensor[-j,-i,k]-STensor[k,-i,-j]],{-i,-j}];
Print[tmp];
tmp=tmp/.HeavierActivate;
NewtonianUpToOrder[tmp,1];
NewtonianUpToOrder[tmp,0];

Quit[];

Print[Style["Difference",Red,20]];
```

```
tmp=Symmetrize[
  -(1/2)CD[-k][STensor[-i,-j,k]+STensor[-j,-i,k]-STensor[k,-i,-j]],{-i,-j}];
Print[tmp];
tmp=tmp/.HeavierActivate;
tmp1=UpToOrder[tmp,1];
Print["removing final multiplier"];
tmp1=tmp1/.FinalMultiplier
   tmp1=UpToOrder[tmp1,1];
tmp1=UpToOrder[tmp,0];
Print["removing final multiplier"];
tmp1=tmp1/.FinalMultiplier
   tmp1=UpToOrder[tmp1,0];


Print[Style["Riemann-Cartan multiplier equation",Red,20]];

tmp=RLambdaEquation[-i,-j];
Print[tmp];
tmp=tmp/.HeavierActivate;
tmp=tmp//FourthOrder;
tmp=tmp/.IntermediateShell;
tmp=tmp//FourthOrder;
Print[tmp];

Print[Style["Derivative of Riemann-Cartan multiplier equation",Red,20]];

tmp=CD[k][RLambdaEquation[-i,-k]];
Print[tmp];
tmp=tmp/.HeavierActivate;
tmp=tmp//FourthOrder;
tmp=tmp/.IntermediateShell;
tmp=tmp//FourthOrder;
Print[tmp];

Print[Style["Skew Derivative of Riemann-Cartan multiplier equation",Red,20]];

tmp=epsilonG[-i,l,j,k]CD[-l][RLambdaEquation[-j,-k]];
Print[tmp];
tmp=tmp/.HeavierActivate;
tmp=tmp//FourthOrder;
tmp=tmp/.IntermediateShell;
tmp=tmp//FourthOrder;
Print[tmp];
```

```
Print[Style["Torsion multiplier equation",Red,20]];

tmp=TLambdaEquation[-i,-j,-k];
Print[tmp];
tmp=tmp/.HeavierActivate;
tmp=tmp//FourthOrder;
tmp=tmp/.IntermediateShell;
tmp=tmp//FourthOrder;
Print[tmp];
*)
```

ORPHAN

```
(*



tmp=tmp/.HeavyActivate;
Print["trying again"];
tmp=UpToOrder[tmp,1];
Print[tmp];

Quit[];

tmp=RUpTo0[i,j,k,l] tmp/.HeavyActivate;
Print["trying again again"];
tmp=tmp//ToNewCanonical;
Print[tmp];




Quit[];
```

```
tmp3=UpToOrder[R[-i,-j,-k,-l],0];
tmp4=UpToOrder[R[-a,-b,-c,-d],1];
DefTensor[R[i,j,k,l],M4]
  tmp5=4DetB[]RUpTo0[-i,-j,-k,-l](2(Alp1 PR1[-i,-j,-k,-l,a,b,c,d]
          +Alp2 PR2[-i,-j,-k,-l,a,b,c,d]
          +Alp3 PR3[-i,-j,-k,-l,a,b,c,d]
          +Alp4 PR4[-i,-j,-k,-l,a,b,c,d]
          +Alp5 PR5[-i,-j,-k,-l,a,b,c,d]
          +Alp6 PR6[-i,-j,-k,-l,a,b,c,d])RUpTo1[-a,-b,-c,-d]+
       (cAlp1 PR1[-i,-j,-k,-l,a,b,c,d]
          +cAlp2 PR2[-i,-j,-k,-l,a,b,c,d]
          +cAlp3 PR3[-i,-j,-k,-l,a,b,c,d]
          +cAlp4 PR4[-i,-j,-k,-l,a,b,c,d]
          +cAlp5 PR5[-i,-j,-k,-l,a,b,c,d]
          +cAlp6 PR6[-i,-j,-k,-l,a,b,c,d])RLambda[-a,-b,-c,-d])/.Theory/.
   PActivate//ToNewCanonical;
tmp5=UpToOrder[tmp5,1];
tmp6=UpToOrder[tmp3 tmp5,1];

Quit[];




Quit[];

APiGDefinition=4DetB[](2(Alp1 PR1[-i,-j,-k,-l,a,b,c,d]
          +Alp2 PR2[-i,-j,-k,-l,a,b,c,d]
          +Alp3 PR3[-i,-j,-k,-l,a,b,c,d]
          +Alp4 PR4[-i,-j,-k,-l,a,b,c,d]
          +Alp5 PR5[-i,-j,-k,-l,a,b,c,d]
          +Alp6 PR6[-i,-j,-k,-l,a,b,c,d])R[-a,-b,-c,-d]+
       (cAlp1 PR1[-i,-j,-k,-l,a,b,c,d]
          +cAlp2 PR2[-i,-j,-k,-l,a,b,c,d]
          +cAlp3 PR3[-i,-j,-k,-l,a,b,c,d]
          +cAlp4 PR4[-i,-j,-k,-l,a,b,c,d]
          +cAlp5 PR5[-i,-j,-k,-l,a,b,c,d]
          +cAlp6 PR6[-i,-j,-k,-l,a,b,c,d])RLambda[-a,-b,-c,-d])/.Theory//
  ToNewCanonical;
APiGActivate=MakeRule[{APiG[-i,-j,-k,-l],Evaluate[APiGDefinition]},
  MetricOn→All,ContractMetrics→True];
```

```
(*
HarmonicLinearise[expr_]:=Module[{res},
  Print[Style["Trying harmonic linearise",Green,20]];
  res=expr;
  Print["HeavyActivate"];
  res=res/.HeavyActivate;
  res=res//ToNewCanonical;
  Print["TSO13Activate"];
  res=res/.TSO13Activate;
  res=res/.CosmicBackgroundActivate;
  res=res//ToNewCanonical;
  Print["ExpandStrengths"];
  res=res/.ExpandStrengths;
  res=res//ToNewCanonical;
  Print["StrengthLambdaSO13Activate"];
  res=res/.StrengthLambdaSO13Activate;
  res=res//ToNewCanonical;
  Print["AActivate"];
  res=res/.AActivate;
  res=res//ScreenDollarIndices;
  Print[res];
  res=res//ToNewCanonical;
  res=ToOrderCanonical[res,1];
  res=res/.StrongLinear;
  res=res//ToNewCanonical;
  res=res/.ToTraceReverse;
  res=res//ToNewCanonical;
  res=res//SortCovDs;
  res=res//ToNewCanonical;
  Print[res];
  res];

DumpSave[NotebookDirectory[]<>"mx_cache/pigs.mx",{HeavyActivate}];
Print["pigs done"];
(*Quit[];*)
(**)
MyImport["pigs.mx"];
*)
```

```
BPiGDefinition=2DetB[](2(Bet1 PT1[-i,-j,-k,a,b,c]
          +Bet2 PT2[-i,-j,-k,a,b,c]
          +Bet3 PT3[-i,-j,-k,a,b,c])T[-a,-b,-c]+
       (cBet1 PT1[-i,-j,-k,a,b,c]
          +cBet2 PT2[-i,-j,-k,a,b,c]
          +cBet3 PT3[-i,-j,-k,a,b,c])TLambda[-a,-b,-c])/.Theory//ToNewCanonical;
BPiGActivate=MakeRule[{BPiG[-i,-j,-k],Evaluate[BPiGDefinition]},
  MetricOn→All,ContractMetrics→True];

APiGDefinition=4DetB[](2(Alp1 PR1[-i,-j,-k,-l,a,b,c,d]
          +Alp2 PR2[-i,-j,-k,-l,a,b,c,d]
          +Alp3 PR3[-i,-j,-k,-l,a,b,c,d]
          +Alp4 PR4[-i,-j,-k,-l,a,b,c,d]
          +Alp5 PR5[-i,-j,-k,-l,a,b,c,d]
          +Alp6 PR6[-i,-j,-k,-l,a,b,c,d])R[-a,-b,-c,-d]+
       (cAlp1 PR1[-i,-j,-k,-l,a,b,c,d]
          +cAlp2 PR2[-i,-j,-k,-l,a,b,c,d]
          +cAlp3 PR3[-i,-j,-k,-l,a,b,c,d]
          +cAlp4 PR4[-i,-j,-k,-l,a,b,c,d]
          +cAlp5 PR5[-i,-j,-k,-l,a,b,c,d]
          +cAlp6 PR6[-i,-j,-k,-l,a,b,c,d])RLambda[-a,-b,-c,-d])/.Theory//
  ToNewCanonical;
APiGActivate=MakeRule[{APiG[-i,-j,-k,-l],Evaluate[APiGDefinition]},
  MetricOn→All,ContractMetrics→True];

LagrangianDefinition=(T[i,j,k]((Bet1 PT1[-i,-j,-k,a,b,c]
            +Bet2 PT2[-i,-j,-k,a,b,c]
            +Bet3 PT3[-i,-j,-k,a,b,c])T[-a,-b,-c]+
         (cBet1 PT1[-i,-j,-k,a,b,c]
            +cBet2 PT2[-i,-j,-k,a,b,c]
            +cBet3 PT3[-i,-j,-k,a,b,c])TLambda[-a,-b,-c])+
     R[i,j,k,l]((Alp1 PR1[-i,-j,-k,-l,a,b,c,d]
            +Alp2 PR2[-i,-j,-k,-l,a,b,c,d]
            +Alp3 PR3[-i,-j,-k,-l,a,b,c,d]
            +Alp4 PR4[-i,-j,-k,-l,a,b,c,d]
            +Alp5 PR5[-i,-j,-k,-l,a,b,c,d]
            +Alp6 PR6[-i,-j,-k,-l,a,b,c,d])R[-a,-b,-c,-d]+
         (cAlp1 PR1[-i,-j,-k,-l,a,b,c,d]
            +cAlp2 PR2[-i,-j,-k,-l,a,b,c,d]
            +cAlp3 PR3[-i,-j,-k,-l,a,b,c,d]
            +cAlp4 PR4[-i,-j,-k,-l,a,b,c,d]
            +cAlp5 PR5[-i,-j,-k,-l,a,b,c,d]
```

```
            +cAlp6 PR6[-i,-j,-k,-l,a,b,c,d])RLambda[-a,-b,-c,-d]))/.Theory//
  ToNewCanonical;
LagrangianActivate=MakeRule[{Lagrangian[],Evaluate[LagrangianDefinition]},
  MetricOn→All,ContractMetrics→True];

TLambdaEquationDefinition=2DetB[]((cBet1 PT1[-i,-j,-k,a,b,c]
        +cBet2 PT2[-i,-j,-k,a,b,c]
        +cBet3 PT3[-i,-j,-k,a,b,c])T[-a,-b,-c])/.Theory//ToNewCanonical;
TLambdaEquationActivate=
 MakeRule[{TLambdaEquation[-i,-j,-k],Evaluate[TLambdaEquationDefinition]},
  MetricOn→All,ContractMetrics→True];

RLambdaEquationDefinition=
 DetB[] cAlp5 Antisymmetrize[R[l,-i,-l,-j],{-i,-j}]/.Theory//ToNewCanonical;
RLambdaEquationActivate=MakeRule[{RLambdaEquation[-i,-j],
   Evaluate[RLambdaEquationDefinition]},MetricOn→All,ContractMetrics→True];

HeavyActivate=Join[BPiGActivate,APiGActivate,
  LagrangianActivate,TLambdaEquationActivate,RLambdaEquationActivate];

tmp3=UpToOrder[R[-a,-b,-c,-d],0];
Print[tmp3];
Quit[];
tmp4=UpToOrder[R[-a,-b,-c,-d],1];
tmp5=4DetB[](2(Alp1 PR1[-i,-j,-k,-l,a,b,c,d]
          +Alp2 PR2[-i,-j,-k,-l,a,b,c,d]
          +Alp3 PR3[-i,-j,-k,-l,a,b,c,d]
          +Alp4 PR4[-i,-j,-k,-l,a,b,c,d]
          +Alp5 PR5[-i,-j,-k,-l,a,b,c,d]
          +Alp6 PR6[-i,-j,-k,-l,a,b,c,d])tmp4+
       (cAlp1 PR1[-i,-j,-k,-l,a,b,c,d]
          +cAlp2 PR2[-i,-j,-k,-l,a,b,c,d]
          +cAlp3 PR3[-i,-j,-k,-l,a,b,c,d]
          +cAlp4 PR4[-i,-j,-k,-l,a,b,c,d]
          +cAlp5 PR5[-i,-j,-k,-l,a,b,c,d]
          +cAlp6 PR6[-i,-j,-k,-l,a,b,c,d])RLambda[-a,-b,-c,-d])/.Theory/.
   PActivate//ToNewCanonical;
tmp5=UpToOrder[tmp5,1];


Quit[];
```

```
tmp1=-CD[-m][H[-j,n]H[-k,m]BPiG[-i,j,k]]+
   A[l,-i,-m]H[-j,n]H[-k,m]BPiG[-l,j,k]//HarmonicLinearise;
tmp2=T[p,-k,-i]BPiG[-p,k,l]H[-l,n]//HarmonicLinearise;
tmp3=(1/2)R[p,q,-k,-i]//HarmonicLinearise;

tmp4=DetB[]Lagrangian[]H[-i,n]//HarmonicLinearise;

tmp1=
 -CD[-m][H[-l,n]H[-k,m]APiG[-i,-j,l,k]]+A[p,-i,-m]H[-l,n]H[-k,m]APiG[-p,-j,l,k]+
   A[p,-j,-m]H[-l,n]H[-k,m]APiG[-i,-p,l,k]//HarmonicLinearise;
tmp2=2Antisymmetrize[BPiG[-i,-j,k]H[-k,n],{-i,-j}]//HarmonicLinearise;

Quit[];


HarmonicLinearise[expr_]:=Module[{res},
  Print[Style["Trying harmonic linearise",Green,20]];
  res=expr;
  Print["HeavyActivate"];
  res=res/.HeavyActivate;
  res=res//ToNewCanonical;
  Print["TSO13Activate"];
  res=res/.TSO13Activate;
  res=res/.CosmicBackgroundActivate;
  res=res//ToNewCanonical;
  Print["ExpandStrengths"];
  res=res/.ExpandStrengths;
  res=res//ToNewCanonical;
  Print["StrengthLambdaSO13Activate"];
  res=res/.StrengthLambdaSO13Activate;
  res=res//ToNewCanonical;
  Print["AActivate"];
  res=res/.AActivate;
  res=res//ScreenDollarIndices;
  Print[res];
  res=res//ToNewCanonical;
  res=ToOrderCanonical[res,1];
  res=res/.StrongLinear;
  res=res//ToNewCanonical;
  res=res/.ToTraceReverse;
  res=res//ToNewCanonical;
  res=res//SortCovDs;
  res=res//ToNewCanonical;
  Print[res];
```

```
  res];

FourthOrder[expr_]:=Module[{res},
  res=expr;
  res=ReplaceDummies[res,IndexList[a1,b1,c1,d1,e1,f1,g1]];
  res=SeparateMetric[G][res,IndexList[a1,b1,c1,d1,e1,f1,g1]];
  res=res//SortCovDs;
  res=res//ToCanonical;
  res=res//ToNewCanonical;
  res=res//ToNewCanonical;
  res=res//CollectTensors;
  res=res//SortCovDs;
  res=res//ToCanonical;
  res=res//ToNewCanonical;
  res=res//CollectTensors;
  res=ReplaceDummies[res,IndexList[a1,b1,c1,d1,e1,f1,g1]];
  res=SeparateMetric[G][res,IndexList[a1,b1,c1,d1,e1,f1,g1]];
  res=res//SortCovDs;
  res=res//ToCanonical;
  res=res//ToNewCanonical;
  res=res//ToNewCanonical;
  res=res//CollectTensors;
  res=res//SortCovDs;
  res=res//ToCanonical;
  res=res//ToNewCanonical;
  res=res//CollectTensors;
  res=res/.ToFaraday;
  res=res//ToCanonical;
  res=res//ToNewCanonical;
  res=res//CollectTensors;
  res];



TEquation=-CD[-m][H[-j,n]H[-k,m]BPiG[-i,j,k]]/.HeavyActivate;
TEquation=TEquation//HarmonicLinearise;
Print[TEquation];
ETensorActivate=
 MakeRule[{ETensor[n,-i],Evaluate[TEquation]},MetricOn→All,ContractMetrics→True];

REquation=-CD[-m][H[-l,n]H[-k,m]APiG[-i,-j,l,k]]+
   2Antisymmetrize[BPiG[-i,-j,k]H[-k,n],{-i,-j}]/.HeavyActivate;
REquation=REquation//HarmonicLinearise;
Print[REquation];
```

```
STensorActivate=MakeRule[
  {STensor[n,-i,-j],Evaluate[REquation]},MetricOn→All,ContractMetrics→True];

(*IntermediateShell=
  MakeRule[{T1[i,-j,-k],0},MetricOn→All,ContractMetrics→True];*)
IntermediateShell=MakeRule[{T1[i,-j,-k],T1[i,-j,-k]},
  MetricOn→All,ContractMetrics→True];
TidySpin=MakeRule[{CD[-a][Faraday2[-b,a]],
   Evaluate[-(1/(8/3 (Alp5-Alp6)))(4 Bet2 T2[-b]+4 cAlp5 CD[-a][RLambda5[-b,a]]+
       8/3 (Alp5-Alp6) CD[-b1][CD[-a][T1[a,b1,-b]]]-
       8/3 (Alp5-Alp6) CD[-b1][CD[-a][T1[-b,a,b1]]]+Spin1[-b])]},
  MetricOn→All,ContractMetrics→True];
(*
4 Bet2 T2[-b]+8/3 (Alp5-Alp6) CD[-a][Faraday2[-b,a]]+
 4 cAlp5 CD[-a][RLambda5[-b,a]]+8/3 (Alp5-Alp6) CD[-b1][CD[-a][T1[a,b1,-b]]]-
 8/3 (Alp5-Alp6) CD[-b1][CD[-a][T1[-b,a,b1]]]
*)
HeavierActivate=Join[STensorActivate,ETensorActivate,HeavyActivate];

DumpSave[NotebookDirectory[]<>"mx_cache/sources.mx",{HeavierActivate}];
Print["sources done"];
(*Quit[];*)
(**)
MyImport["sources.mx"];

Print[Style["Linearised equations",Red,40]];

Print[Style["Full spin equation",Red,20]];

tmp=STensor[i,-j,-k];
Print[tmp];
tmp=tmp/.HeavierActivate;
tmp=tmp//FourthOrder;
Print[tmp];

Print[Style["Symmetric part of spin equation",Red,20]];

tmp=Symmetrize[STensor[-i,-j,-k],{-i,-j}]+
  (1/6) (G[-k,-i]STensor[a,-a,-j]+G[-k,-j]STensor[a,-a,-i])-
  (1/3)G[-i,-j]STensor[a,-a,-k];
Print[tmp];
tmp=tmp/.HeavierActivate;
tmp=tmp//FourthOrder;
```

```
Print[tmp];

Print[Style["Part 1 spin equation",Red,20]];

tmp=PT1[-i,-j,-k,a,b,c]STensor[-a,-b,-c];
Print[tmp];
tmp=tmp/.PActivate;
tmp=tmp/.HeavierActivate;
tmp=tmp//FourthOrder;
tmp=tmp/.IntermediateShell;
tmp=tmp//FourthOrder;
tmp=tmp/.TidySpin;
tmp=tmp//FourthOrder;
tmp=tmp/.TidySpin;
tmp=tmp//FourthOrder;
Print[tmp];

Print[Style["Part 2 spin equation",Red,20]];

tmp=STensor[c,-b,-c];
Print[tmp];
tmp=tmp/.PActivate;
tmp=tmp/.HeavierActivate;
tmp=tmp//FourthOrder;
tmp=tmp/.IntermediateShell;
tmp=tmp//FourthOrder;
Print[tmp];

Print[Style["Part 3 spin equation",Red,20]];

tmp=epsilonG[-i,a,b,c]STensor[-a,-b,-c];
Print[tmp];
tmp=tmp/.PActivate;
tmp=tmp/.HeavierActivate;
tmp=tmp//FourthOrder;
tmp=tmp/.IntermediateShell;
tmp=tmp//FourthOrder;
Print[tmp];

Print[Style["Full stress-energy equation",Red,20]];

tmp=ETensor[-i,-j];
Print[tmp];
tmp=tmp/.HeavierActivate;
```

```
tmp=tmp//FourthOrder;
tmp=tmp/.IntermediateShell;
tmp=tmp//FourthOrder;
Print[tmp];

Print[Style["Full Belinfante equation",Red,20]];

tmp=
 ETensor[-i,-j]-(1/2)CD[-k][STensor[-i,-j,k]+STensor[-j,-i,k]-STensor[k,-i,-j]];
Print[tmp];
tmp=tmp/.HeavierActivate;
tmp=tmp//FourthOrder;
tmp=tmp/.IntermediateShell;
tmp=tmp//FourthOrder;
Print[tmp];

Print[Style["Symmetric Belinfante equation",Red,20]];

tmp=Symmetrize[ETensor[-i,-j]-
   (1/2)CD[-k][STensor[-i,-j,k]+STensor[-j,-i,k]-STensor[k,-i,-j]],{-i,-j}];
Print[tmp];
tmp=tmp/.HeavierActivate;
tmp=tmp//FourthOrder;
tmp=tmp/.IntermediateShell;
tmp=tmp//FourthOrder;
Print[tmp];

Print[Style["Riemann-Cartan multiplier equation",Red,20]];

tmp=RLambdaEquation[-i,-j];
Print[tmp];
tmp=tmp/.HeavierActivate;
tmp=tmp//FourthOrder;
tmp=tmp/.IntermediateShell;
tmp=tmp//FourthOrder;
Print[tmp];

Print[Style["Derivative of Riemann-Cartan multiplier equation",Red,20]];

tmp=CD[k][RLambdaEquation[-i,-k]];
Print[tmp];
tmp=tmp/.HeavierActivate;
tmp=tmp//FourthOrder;
```

```
tmp=tmp/.IntermediateShell;
tmp=tmp//FourthOrder;
Print[tmp];

Print[Style["Skew Derivative of Riemann-Cartan multiplier equation",Red,20]];

tmp=epsilonG[-i,l,j,k]CD[-l][RLambdaEquation[-j,-k]];
Print[tmp];
tmp=tmp/.HeavierActivate;
tmp=tmp//FourthOrder;
tmp=tmp/.IntermediateShell;
tmp=tmp//FourthOrder;
Print[tmp];

Print[Style["Torsion multiplier equation",Red,20]];

tmp=TLambdaEquation[-i,-j,-k];
Print[tmp];
tmp=tmp/.HeavierActivate;
tmp=tmp//FourthOrder;
tmp=tmp/.IntermediateShell;
tmp=tmp//FourthOrder;
Print[tmp];

(*
Print[Style["Another derivative of Riemann-Cartan multiplier equation",Red,20]];

tmp=CD[i][RLambdaEquation[-i,-j,-k,-l]];
Print[tmp];
tmp=tmp/.HeavierActivate;
tmp=tmp//FourthOrder;
tmp=tmp/.IntermediateShell;
tmp=tmp//FourthOrder;
Print[tmp];

Print[Style["symmetric part of the same",Red,20]];

tmp=Symmetrize[CD[i][RLambdaEquation[-i,-j,-k,-l]],{-j,-k}];
Print[tmp];
tmp=tmp/.HeavierActivate;
tmp=tmp//FourthOrder;
tmp=tmp/.IntermediateShell;
tmp=tmp//FourthOrder;
Print[tmp];
```

```
Print[Style["epsilon part of the same",Red,20]];

tmp=epsilonG[-n,j,k,l]CD[i][RLambdaEquation[-i,-j,-k,-l]];
Print[tmp];
tmp=tmp/.HeavierActivate;
tmp=tmp//FourthOrder;
tmp=tmp/.IntermediateShell;
tmp=tmp//FourthOrder;
Print[tmp];
*)




Quit[];

Print[Style["skew partial spin-torsion",Red,20]];

tmp=CD[-l][STensor[-i,-j,l]]/.SourceActivate//ToNewCanonical;
tmp=FourthOrder[tmp];
tmp=tmp//ToNewCanonical;
tmp=tmp//CollectTensors;
tmp=tmp//SortCovDs;
tmp=tmp//ToCanonical;
tmp=tmp//ToNewCanonical;
tmp=tmp//CollectTensors;
Print[tmp];

Print[Style["symm partial spin-torsion",Red,20]];

tmp=Symmetrize[CD[-l][STensor[l,-i,-j]],{-i,-j}]/.SourceActivate//ToNewCanonical;
tmp=FourthOrder[tmp];
tmp=tmp//ToNewCanonical;
tmp=tmp//CollectTensors;
tmp=tmp//SortCovDs;
tmp=tmp//ToCanonical;
tmp=tmp//ToNewCanonical;
tmp=tmp//CollectTensors;
Print[tmp];
com1=tmp;

Print[Style["symm stress-energy",Red,20]];
```

```
tmp=Symmetrize[ETensor[-i,-j],{-i,-j}]/.SourceActivate//ToNewCanonical;
tmp=FourthOrder[tmp];
tmp=tmp//ToNewCanonical;
tmp=tmp//CollectTensors;
tmp=tmp//SortCovDs;
tmp=tmp//ToCanonical;
tmp=tmp//ToNewCanonical;
tmp=tmp//CollectTensors;
Print[tmp];
com2=tmp;
```

```
Print[Style["Symm Weyl",Red,20]];

tmp=WTensor[-i,-j]/.SourceActivate//ToNewCanonical;
tmp=FourthOrder[tmp];
tmp=tmp//ToNewCanonical;
tmp=tmp//CollectTensors;
tmp=tmp//SortCovDs;
tmp=tmp//ToCanonical;
tmp=tmp//ToNewCanonical;
tmp=tmp//CollectTensors;
Print[tmp];

Print[Style["Riemann--Cartan constraint",Red,20]];

tmp=SLambdaTensor[-i,-j,-k,-l]/.SourceActivate//ToNewCanonical;
tmp=FourthOrder[tmp];
tmp=tmp//ToNewCanonical;
tmp=tmp//CollectTensors;
tmp=tmp//SortCovDs;
tmp=tmp//ToCanonical;
tmp=tmp//ToNewCanonical;
tmp=tmp//CollectTensors;
Print[tmp];
```

```
Print[Style["skew partial spin-torsion",Red,20]];

tmp=CD[-l][STensor[-i,-j,l]]/.SourceActivate//ToNewCanonical;
tmp=FourthOrder[tmp];
tmp=tmp//ToNewCanonical;
tmp=tmp//CollectTensors;
tmp=tmp//SortCovDs;
tmp=tmp//ToCanonical;
tmp=tmp//ToNewCanonical;
tmp=tmp//CollectTensors;
Print[tmp];

Print[Style["symm partial spin-torsion",Red,20]];

tmp=Symmetrize[CD[-l][STensor[l,-i,-j]],{-i,-j}]/.SourceActivate//ToNewCanonical;
tmp=FourthOrder[tmp];
tmp=tmp//ToNewCanonical;
tmp=tmp//CollectTensors;
tmp=tmp//SortCovDs;
tmp=tmp//ToCanonical;
tmp=tmp//ToNewCanonical;
tmp=tmp//CollectTensors;
Print[tmp];
com1=tmp;

Print[Style["symm stress-energy",Red,20]];

tmp=Symmetrize[ETensor[-i,-j],{-i,-j}]/.SourceActivate//ToNewCanonical;
tmp=FourthOrder[tmp];
tmp=tmp//ToNewCanonical;
tmp=tmp//CollectTensors;
tmp=tmp//SortCovDs;
tmp=tmp//ToCanonical;
tmp=tmp//ToNewCanonical;
tmp=tmp//CollectTensors;
Print[tmp];
com2=tmp;

Print[Style["combined stress-energy",Red,20]];

tmp=com1+com2;
tmp=tmp//ToNewCanonical;
Print[tmp];
```

```
Print[Style["trace of combined stress-energy",Red,20]];

tmp=G[i,j](com1+com2);
tmp=FourthOrder[tmp];
tmp=tmp//ToNewCanonical;
tmp=tmp//CollectTensors;
tmp=tmp//SortCovDs;
tmp=tmp//ToCanonical;
tmp=tmp//ToNewCanonical;
tmp=tmp//CollectTensors;
tmp=tmp//ToNewCanonical;
Print[tmp];




Quit[];

(*quick examination of the unusual identity*)
(*
DefTensor[Mult[a,b,-d,-e],M4,
 {Antisymmetric[{a,b}],Antisymmetric[{-d,-e}]},PrintAs→"λ"];
BarredAConstants=DefNiceConstantSymbol[α,#,X]&/@Range[6];

DefTensor[FTorsion[-a,-b,-c],M4,Antisymmetric[{-b,-c}]];
DefTensor[FRiemannCartan[-a,-b,-c,-d],M4,
 {Antisymmetric[{-a,-b}],Antisymmetric[{-c,-d}]}];
DefTensor[FRiemannCartanMult[-a,-b,-c,-d],M4,
 {Antisymmetric[{-a,-b}],Antisymmetric[{-c,-d}]}];
DefTensor[ShellR[-a,-b,-c,-d],M4,
 {Antisymmetric[{-a,-b}],Antisymmetric[{-c,-d}]}];
FTorsionDefinition=(Bet1 PT1[-i,-j,-k,a,b,c]
      +Bet2 PT2[-i,-j,-k,a,b,c]
      +Bet3 PT3[-i,-j,-k,a,b,c])T[-a,-b,-c]/.PActivate//ToNewCanonical;
FRiemannCartanDefinition=(Alp1 PR1[-i,-j,-k,-l,a,b,c,d]
       +Alp2 PR2[-i,-j,-k,-l,a,b,c,d]
       +Alp3 PR3[-i,-j,-k,-l,a,b,c,d]
       +Alp4 PR4[-i,-j,-k,-l,a,b,c,d]
       +Alp5 PR5[-i,-j,-k,-l,a,b,c,d]
       +Alp6 PR6[-i,-j,-k,-l,a,b,c,d])R[-a,-b,-c,-d]+
    (α1X PR1[-i,-j,-k,-l,a,b,c,d]
       +α2X PR2[-i,-j,-k,-l,a,b,c,d]
       +α3X PR3[-i,-j,-k,-l,a,b,c,d]
```

```
        +α4X PR4[-i,-j,-k,-l,a,b,c,d]
        +α5X PR5[-i,-j,-k,-l,a,b,c,d]
        +α6X PR6[-i,-j,-k,-l,a,b,c,d])Mult[-a,-b,-c,-d]/.PActivate//
  ToNewCanonical;
FRiemannCartanMultDefinition=(α1X PR1[-i,-j,-k,-l,a,b,c,d]
       +α2X PR2[-i,-j,-k,-l,a,b,c,d]
       +α3X PR3[-i,-j,-k,-l,a,b,c,d]
       +α4X PR4[-i,-j,-k,-l,a,b,c,d]
       +α5X PR5[-i,-j,-k,-l,a,b,c,d]
       +α6X PR6[-i,-j,-k,-l,a,b,c,d])Mult[-a,-b,-c,-d]/.PActivate//ToNewCanonical;
ShellRDefinition=((1-α1X) PR1[-i,-j,-k,-l,a,b,c,d]
       +(1-α2X) PR2[-i,-j,-k,-l,a,b,c,d]
       +(1-α3X) PR3[-i,-j,-k,-l,a,b,c,d]
       +(1-α4X) PR4[-i,-j,-k,-l,a,b,c,d]
       +(1-α5X) PR5[-i,-j,-k,-l,a,b,c,d]
       +(1-α6X) PR6[-i,-j,-k,-l,a,b,c,d])R[-a,-b,-c,-d]/.PActivate//
  ToNewCanonical;
FTorsionActivate=MakeRule[{FTorsion[-i,-j,-k],Evaluate[FTorsionDefinition]},
  MetricOn→All,ContractMetrics→True];
FRiemannCartanActivate=MakeRule[{FRiemannCartan[-i,-j,-k,-l],
   Evaluate[FRiemannCartanDefinition]},MetricOn→All,ContractMetrics→True];
FRiemannCartanMultActivate=MakeRule[{FRiemannCartanMult[-i,-j,-k,-l],
   Evaluate[FRiemannCartanMultDefinition]},MetricOn→All,ContractMetrics→True];
ShellRActivate=MakeRule[{ShellR[-i,-j,-k,-l],Evaluate[ShellRDefinition]},
  MetricOn→All,ContractMetrics→True];
```

```
tmp=T[-j,-p,-q]FTorsion[-i,p,q]-2T[p,-k,-i]FTorsion[-p,k,-j]/.FTorsionActivate//
   ToNewCanonical//CollectTensors;
Print[tmp];
tmpa=
 R[p,q,-k,-i]FRiemannCartan[-p,-q,k,-j]-R[p,q,-k,-j]FRiemannCartan[-p,-q,k,-i]/.
     FRiemannCartanActivate//ToNewCanonical//CollectTensors//ToCanonical;
Print[tmpa];
tmpb=
 R[k,-i,-p,-q]FRiemannCartan[-k,-j,p,q]-R[k,-j,-p,-q]FRiemannCartan[-k,-i,p,q]/.
     FRiemannCartanActivate//ToNewCanonical//CollectTensors//ToCanonical;
Print[tmpb];
tmpc=tmpa+tmpb//ToNewCanonical//CollectTensors//ToCanonical;
Print[tmpc];
Print["looking at difference of equations"]
```

```
  tmp=
 (ShellR[p,q,-k,-i]FRiemannCartanMult[-p,-q,k,-j])/.FRiemannCartanMultActivate//
    ToNewCanonical//CollectTensors//ToCanonical;
(*
tmp=(ShellR[p,q,-k,-i]FRiemannCartanMult[-p,-q,k,-j]-
         ShellR[p,q,-k,-j]FRiemannCartanMult[-p,-q,k,-i]+
         ShellR[k,-i,-p,-q]FRiemannCartanMult[-k,-j,p,q]-
         ShellR[k,-j,-p,-q]FRiemannCartanMult[-k,-i,p,q])/.
      FRiemannCartanMultActivate//ToNewCanonical//CollectTensors//ToCanonical;
*)
Print[tmp];
tmp=tmp/.ShellRActivate//ToNewCanonical//CollectTensors//ToCanonical;
Print[tmp];
tmp=tmp//CollectTensors;
Print[tmp];
*)
*)
```

ORPHAN

```
(*
(*
DefTensor[R1[-a,-b,-c,-d],M4,
 {Antisymmetric[{-a,-b}],Antisymmetric[{-c,-d}]},PrintAs→"R1"];
DefTensor[R2[-a,-b,-c,-d],M4,
 {Antisymmetric[{-a,-b}],Antisymmetric[{-c,-d}]},PrintAs→"R2"];
DefTensor[R3[-a,-b,-c,-d],M4,
 {Antisymmetric[{-a,-b}],Antisymmetric[{-c,-d}]},PrintAs→"R3"];
DefTensor[R4[-a,-b,-c,-d],M4,
 {Antisymmetric[{-a,-b}],Antisymmetric[{-c,-d}]},PrintAs→"R4"];
DefTensor[R5[-a,-b,-c,-d],M4,
 {Antisymmetric[{-a,-b}],Antisymmetric[{-c,-d}]},PrintAs→"R5"];
DefTensor[R6[-a,-b,-c,-d],M4,
 {Antisymmetric[{-a,-b}],Antisymmetric[{-c,-d}]},PrintAs→"R6"];

DefTensor[T1[-a,-b,-c],M4,Antisymmetric[{-b,-c}],PrintAs→"T1"];
DefTensor[T2[-a,-b,-c],M4,Antisymmetric[{-b,-c}],PrintAs→"T2"];
DefTensor[T3[-a,-b,-c],M4,Antisymmetric[{-b,-c}],PrintAs→"T3"];

BarredAConstants=DefNiceConstantSymbol[α,#,X]&/@Range[6];
BarredAConstants=DefNiceConstantSymbol[β,#,X]&/@Range[3];
*)
(*
DefTensor[FirstTerm[-a,-b,-c,-d],M4,
```

```
 {Antisymmetric[{-a,-b}],Antisymmetric[{-c,-d}]},PrintAs→"FT"];
FirstTermDefinition=(α1X PR1[-i,-j,-k,-l,a,b,c,d]
        +α2X PR2[-i,-j,-k,-l,a,b,c,d]
        +α3X PR3[-i,-j,-k,-l,a,b,c,d]
        +α4X PR4[-i,-j,-k,-l,a,b,c,d]
        +α5X PR5[-i,-j,-k,-l,a,b,c,d]
        +α6X PR6[-i,-j,-k,-l,a,b,c,d])PPara[-c,x]PPara[-d,y]R[-a,-b,-x,-y]/.
     PADMActivate/.PActivate//ToNesterForm;
FirstTermActivate=MakeRule[{FirstTerm[-i,-j,-k,-l],Evaluate[FirstTermDefinition]},
  MetricOn→All,ContractMetrics→True];
*)
(*
DefTensor[FirstTerm[-a,-b,-c,-d],M4,
 {Antisymmetric[{-a,-b}],Antisymmetric[{-c,-d}]},PrintAs→"FT"];
FirstTermDefinition=(α1X PR1[-i,-j,-k,-l,a,b,c,d]
        +α2X PR2[-i,-j,-k,-l,a,b,c,d]
        +α3X PR3[-i,-j,-k,-l,a,b,c,d]
        +α4X PR4[-i,-j,-k,-l,a,b,c,d]
        +α5X PR5[-i,-j,-k,-l,a,b,c,d]
        +α6X PR6[-i,-j,-k,-l,a,b,c,d])PPara[-c,x]PPara[-d,y]R[-a,-b,-x,-y]/.
     PADMActivate/.PActivate//ToNesterForm;
FirstTermActivate=MakeRule[{FirstTerm[-i,-j,-k,-l],Evaluate[FirstTermDefinition]},
  MetricOn→All,ContractMetrics→True];

DefTensor[FirstTermTorsion[-a,-c,-d],M4,Antisymmetric[{-c,-d}],PrintAs→"FTT"];
FirstTermTorsionDefinition=(β1X PT1[-i,-k,-l,a,c,d]
        +β2X PT2[-i,-k,-l,a,c,d]
        +β3X PT3[-i,-k,-l,a,c,d])PPara[-c,x]PPara[-d,y]T[-a,-x,-y]/.PADMActivate/.
    PActivate//ToNesterForm;
FirstTermTorsionActivate=MakeRule[{FirstTermTorsion[-i,-k,-l],
    Evaluate[FirstTermTorsionDefinition]},MetricOn→All,ContractMetrics→True];

DefTensor[SecondTerm[-a,-b,-c,-d],M4,
 {Antisymmetric[{-a,-b}],Antisymmetric[{-c,-d}]},PrintAs→"ST"];
SecondTermDefinition=(α1X PR1[-i,-j,-k,-l,a,b,c,d]
        +α2X PR2[-i,-j,-k,-l,a,b,c,d]
        +α3X PR3[-i,-j,-k,-l,a,b,c,d]
        +α4X PR4[-i,-j,-k,-l,a,b,c,d]
        +α5X PR5[-i,-j,-k,-l,a,b,c,d]
        +α6X PR6[-i,-j,-k,-l,a,b,c,d])V[-c]V[x]PPara[-d,y]R[-a,-b,-x,-y]/.
     PADMActivate/.PActivate//ToNesterForm;
SecondTermActivate=MakeRule[{SecondTerm[-i,-j,-k,-l],
    Evaluate[SecondTermDefinition]},MetricOn→All,ContractMetrics→True];
```

```
DefTensor[SecondTermTorsion[-a,-c,-d],M4,Antisymmetric[{-c,-d}],PrintAs→"STT"];
SecondTermTorsionDefinition=(β1X PT1[-i,-k,-l,a,c,d]
        +β2X PT2[-i,-k,-l,a,c,d]
        +β3X PT3[-i,-k,-l,a,c,d])V[-c]V[x]PPara[-d,y]T[-a,-x,-y]/.PADMActivate/.
   PActivate//ToNesterForm;
SecondTermTorsionActivate=MakeRule[{SecondTermTorsion[-i,-k,-l],
   Evaluate[SecondTermTorsionDefinition]},MetricOn→All,ContractMetrics→True];
*)
(*
R1Definition=PR1[-i,-j,-k,-l,a,b,c,d]R[-a,-b,-c,-d]/.PActivate//ToNesterForm;
R1Activate=MakeRule[{R1[-i,-j,-k,-l],Evaluate[R1Definition]},
  MetricOn→All,ContractMetrics→True];
R2Definition=PR2[-i,-j,-k,-l,a,b,c,d]R[-a,-b,-c,-d]/.PActivate//ToNesterForm;
R2Activate=MakeRule[{R2[-i,-j,-k,-l],Evaluate[R2Definition]},
  MetricOn→All,ContractMetrics→True];
R3Definition=PR3[-i,-j,-k,-l,a,b,c,d]R[-a,-b,-c,-d]/.PActivate//ToNesterForm;
R3Activate=MakeRule[{R3[-i,-j,-k,-l],Evaluate[R3Definition]},
  MetricOn→All,ContractMetrics→True];
R4Definition=PR4[-i,-j,-k,-l,a,b,c,d]R[-a,-b,-c,-d]/.PActivate//ToNesterForm;
R4Activate=MakeRule[{R4[-i,-j,-k,-l],Evaluate[R4Definition]},
  MetricOn→All,ContractMetrics→True];
R5Definition=PR5[-i,-j,-k,-l,a,b,c,d]R[-a,-b,-c,-d]/.PActivate//ToNesterForm;
R5Activate=MakeRule[{R5[-i,-j,-k,-l],Evaluate[R5Definition]},
  MetricOn→All,ContractMetrics→True];
R6Definition=PR6[-i,-j,-k,-l,a,b,c,d]R[-a,-b,-c,-d]/.PActivate//ToNesterForm;
R6Activate=MakeRule[{R6[-i,-j,-k,-l],Evaluate[R6Definition]},
  MetricOn→All,ContractMetrics→True];
RNActivate=Join[R1Activate,R2Activate,R3Activate,
  R4Activate,R5Activate,R6Activate];

T1Definition=PT1[-i,-j,-k,a,b,c]T[-a,-b,-c]/.PActivate//ToNesterForm;
T1Activate=MakeRule[{T1[-i,-j,-k],Evaluate[T1Definition]},
  MetricOn→All,ContractMetrics→True];
T2Definition=PT2[-i,-j,-k,a,b,c]T[-a,-b,-c]/.PActivate//ToNesterForm;
T2Activate=MakeRule[{T2[-i,-j,-k],Evaluate[T2Definition]},
  MetricOn→All,ContractMetrics→True];
T3Definition=PT3[-i,-j,-k,a,b,c]T[-a,-b,-c]/.PActivate//ToNesterForm;
T3Activate=MakeRule[{T3[-i,-j,-k],Evaluate[T3Definition]},
  MetricOn→All,ContractMetrics→True];
TNActivate=Join[T1Activate,T2Activate,T3Activate];

DumpSave[NotebookDirectory[]<>"mx_cache/Adjunct.mx",{RNActivate,TNActivate}];
```

```
Print["done"];
Quit[];
*)
MyImport["Adjunct.mx"];
(*
AdjunctToNester[x_]:=Module[{res},res=x;
  Print[Style["Expanding adjunct...",Blue,10]];
  res=res/.PADMActivate/.RNActivate/.TNActivate/.FirstTermActivate/.
      SecondTermActivate/.FirstTermTorsionActivate/.SecondTermTorsionActivate;
  res=res/.PO3RActivate/.PPerpO3RActivate/.PADMRActivate/.
     PPerpADMRActivate//ToCanonical;
  res=res/.PO3TActivate/.PPerpO3TActivate/.PADMTActivate/.
     PPerpADMTActivate//ToCanonical;
  res=res//ToNewCanonical;
  res=res/.FoliGToG;
  res=res//ToNewCanonical;
  res=res/.GToFoliG;
  res=res//ToNewCanonical;
  res];



AdjunctRiemannCartanDisplay[part_]:=Module[{temp1},
  Print[Style["Riemann-Cartan display irreps:",Orange,20]];
  Print[Style["Parallel 0⁺:",Blue,20]];
  temp1=PR0p[e,f,g,h]PRPara[-e,-f,-g,-h,a,b,c,d]
     PPara[-c,x]PPara[-d,y]part//AdjunctToNester;
  Print[temp1];
  Print[Style["Parallel 0⁻:",Blue,20]];
  temp1=PR0m[e,f,g]PRPerp[-e,-f,-g,a,b,c,d]
     PPara[-c,x]PPara[-d,y]part//AdjunctToNester;
  Print[temp1];
  Print[Style["Parallel 1⁺:",Blue,20]];
  temp1=PR1p[-n,-m,e,f,g,h]PRPara[-e,-f,-g,-h,a,b,c,d]
     PPara[-c,x]PPara[-d,y]part//AdjunctToNester;
  Print[temp1];
  Print[Style["Parallel 1⁻:",Blue,20]];
  temp1=PR1m[-n,e,f,g]PRPerp[-e,-f,-g,a,b,c,d]
     PPara[-c,x]PPara[-d,y]part//AdjunctToNester;
  Print[temp1];
  Print[Style["Parallel 2⁺:",Blue,20]];
  temp1=PR2p[-n,-m,e,f,g,h]PRPara[-e,-f,-g,-h,a,b,c,d]
     PPara[-c,x]PPara[-d,y]part//AdjunctToNester;
  Print[temp1];
```

```
  Print[Style["Parallel 2⁻:",Blue,20]];
  temp1=PR2m[-n,-m,-o,e,f,g]PRPerp[-e,-f,-g,a,b,c,d]
     PPara[-c,x]PPara[-d,y]part//AdjunctToNester;
  Print[temp1];
  Print[Style["Perpendicular 0⁺:",Red,20]];
  temp1=
   PPerpR0p[e,f]PPerpRPerp[-e,-f,a,b,c]PPara[-c,x]V[y]part//AdjunctToNester;
  Print[temp1];
  Print[Style["Perpendicular 0⁻:",Red,20]];
  temp1=
   PPerpR0m[e,f,g]PPerpRPara[-e,-f,-g,a,b,c]PPara[-c,x]V[y]part//AdjunctToNester;
  Print[temp1];
  Print[Style["Perpendicular 1⁺:",Red,20]];
  temp1=PPerpR1p[-n,-m,e,f]
     PPerpRPerp[-e,-f,a,b,c]PPara[-c,x]V[y]part//AdjunctToNester;
  Print[temp1];
  Print[Style["Perpendicular 1⁻:",Red,20]];
  temp1=PPerpR1m[-n,e,f,g]
     PPerpRPara[-e,-f,-g,a,b,c]PPara[-c,x]V[y]part//AdjunctToNester;
  Print[temp1];
  Print[Style["Perpendicular 2⁺:",Red,20]];
  temp1=PPerpR2p[-n,-m,e,f]
     PPerpRPerp[-e,-f,a,b,c]PPara[-c,x]V[y]part//AdjunctToNester;
  Print[temp1];
  Print[Style["Perpendicular 2⁻:",Red,20]];
  temp1=PPerpR2m[-n,-m,-o,e,f,g]
     PPerpRPara[-e,-f,-g,a,b,c]PPara[-c,x]V[y]part//AdjunctToNester;
  Print[temp1];
 ];

AdjunctTorsionDisplay[part_]:=Module[{temp1},
  Print[Style["Riemann-Cartan display irreps:",Orange,20]];
  Print[Style["Parallel 0⁻:",Blue,20]];
  temp1=
   PT0m[e,f,g]PTPara[-e,-f,-g,a,b,c]PPara[-b,x]PPara[-c,y]part//AdjunctToNester;
  Print[temp1];
  Print[Style["Parallel 1⁺:",Blue,20]];
  temp1=
   PT1p[-n,-m,e,f]PTPerp[-e,-f,a,b,c]PPara[-b,x]PPara[-c,y]part//AdjunctToNester;
  Print[temp1];
  Print[Style["Parallel 1⁻:",Blue,20]];
  temp1=PT1m[-n,e,f,g]PTPara[-e,-f,-g,a,b,c]
     PPara[-b,x]PPara[-c,y]part//AdjunctToNester;
  Print[temp1];
```

```
  Print[Style["Parallel 2⁻:",Blue,20]];
  temp1=PT2m[-n,-m,-o,e,f,g]
     PTPara[-e,-f,-g,a,b,c]PPara[-b,x]PPara[-c,y]part//AdjunctToNester;
  Print[temp1];
  Print[Style["Perpendicular 0⁺:",Red,20]];
  temp1=PPerpT0p[e,f]PPerpTPara[-e,-f,a,b]PPara[-b,x]V[y]part//AdjunctToNester;
  Print[temp1];
  Print[Style["Perpendicular 1⁺:",Red,20]];
  temp1=
   PPerpT1p[-n,-m,e,f]PPerpTPara[-e,-f,a,b]PPara[-b,x]V[y]part//AdjunctToNester;
  Print[temp1];
  Print[Style["Perpendicular 1⁻:",Red,20]];
  temp1=PPerpT1m[-n,e]PPerpTPerp[-e,a,b]PPara[-b,x]V[y]part//AdjunctToNester;
  Print[temp1];
  Print[Style["Perpendicular 2⁺:",Red,20]];
  temp1=
   PPerpT2p[-n,-m,e,f]PPerpTPara[-e,-f,a,b]PPara[-b,x]V[y]part//AdjunctToNester;
  Print[temp1];
 ];
*)
(*
Print[Style["PART 1",Red,40]];
AdjunctRiemannCartanDisplay[R1[-a,-b,-x,-y]];
Print[Style["PART 2",Red,40]];
AdjunctRiemannCartanDisplay[R2[-a,-b,-x,-y]];
Print[Style["PART 3",Red,40]];
AdjunctRiemannCartanDisplay[R3[-a,-b,-x,-y]];
Print[Style["PART 4",Red,40]];
AdjunctRiemannCartanDisplay[R4[-a,-b,-x,-y]];
Print[Style["PART 5",Red,40]];
AdjunctRiemannCartanDisplay[R5[-a,-b,-x,-y]];
Print[Style["PART 6",Red,40]];
AdjunctRiemannCartanDisplay[R6[-a,-b,-x,-y]];

Quit[];
*)
(*
Print[Style["FIRST TERM",Red,40]];
AdjunctTorsionDisplay[FirstTermTorsion[-a,-x,-y]];


Print[Style["SECOND TERM",Red,40]];
AdjunctTorsionDisplay[SecondTermTorsion[-a,-x,-y]];
```

```
Quit[];
*)
(*
Print[Style["PART 1",Red,40]];
AdjunctTorsionDisplay[T1[-a,-x,-y]];
Print[Style["PART 2",Red,40]];
AdjunctTorsionDisplay[T2[-a,-x,-y]];
Print[Style["PART 3",Red,40]];
AdjunctTorsionDisplay[T3[-a,-x,-y]];
*)
*)
```

Exact solutions

Brinkmann gauge

In[•]:=
```
(*
DefConstantSymbol[ϕ];
DefConstantSymbol[θ];

DefTensor[Vt[-a],M4,PrintAs→"(e_t)"];
DefTensor[Vx[-a],M4,PrintAs→"(e_x)"];
DefTensor[Vy[-a],M4,PrintAs→"(e_y)"];
DefTensor[Vz[-a],M4,PrintAs→"(e_z)"];

AutomaticRules[Vt,MakeRule[{Vt[-a]V[a],1},MetricOn→All,ContractMetrics→True]];
AutomaticRules[Vt,MakeRule[{Vt[-a]Vt[a],1},MetricOn→All,ContractMetrics→True]];
AutomaticRules[Vt,MakeRule[{Vt[-a]Vx[a],0},MetricOn→All,ContractMetrics→True]];
AutomaticRules[Vt,MakeRule[{Vt[-a]Vy[a],0},MetricOn→All,ContractMetrics→True]];
AutomaticRules[Vt,MakeRule[{Vt[-a]Vz[a],0},MetricOn→All,ContractMetrics→True]];

AutomaticRules[Vx,MakeRule[{Vx[-a]V[a],0},MetricOn→All,ContractMetrics→True]];
AutomaticRules[Vx,MakeRule[{Vx[-a]Vx[a],-1},MetricOn→All,ContractMetrics→True]];
AutomaticRules[Vx,MakeRule[{Vx[-a]Vy[a],0},MetricOn→All,ContractMetrics→True]];
AutomaticRules[Vx,MakeRule[{Vx[-a]Vz[a],0},MetricOn→All,ContractMetrics→True]];

AutomaticRules[Vy,MakeRule[{Vy[-a]V[a],0},MetricOn→All,ContractMetrics→True]];
AutomaticRules[Vy,MakeRule[{Vy[-a]Vy[a],-1},MetricOn→All,ContractMetrics→True]];
AutomaticRules[Vy,MakeRule[{Vy[-a]Vz[a],0},MetricOn→All,ContractMetrics→True]];

AutomaticRules[Vz,MakeRule[{Vz[-a]V[a],0},MetricOn→All,ContractMetrics→True]];
AutomaticRules[Vz,MakeRule[{Vz[-a]Vz[a],-1},MetricOn→All,ContractMetrics→True]];

Tycho=MakeRule[{Vt[-a],V[-a]},MetricOn→All,ContractMetrics→True];
*)
(*
epsilonG[-e,-f,-g,-h]epsilonG[-a,-b,-c,-d]//ToNewCanonical;
Print[%];
*)
```

Riemann-Cartan waves (2 D.o.F)

In[•]:=
```
(*
ADotB=Cos[ϕ] (Vt[-l]Vx[-k]-Vt[-k]Vx[-l])+
  Cos[ϕ] (Vz[-l]Vx[-k]-Vz[-k]Vx[-l])+
  Sin[ϕ] (Vt[-l]Vy[-k]-Vt[-k]Vy[-l])+
  Sin[ϕ] (Vz[-l]Vy[-k]-Vz[-k]Vy[-l]);

CDotB=Cos[ϕ] (Vt[-j]Vx[-i]-Vt[-i]Vx[-j])+
```

```
  Cos[ϕ] (Vz[-j]Vx[-i]-Vz[-i]Vx[-j]) +
  Sin[ϕ] (Vt[-j]Vy[-i]-Vt[-i]Vy[-j]) +
  Sin[ϕ] (Vz[-j]Vy[-i]-Vz[-i]Vy[-j]);

CDotA=CDotA;

BDotB=0;

AWedgeBDotCWedgeB=-G[-k,-e]G[-l,-f]
   (Cos[ϕ]Vt[-g]Vx[-h]+Cos[ϕ]Vz[-g]Vx[-h]+Sin[ϕ]Vt[-g]Vy[-h]+Sin[ϕ]Vz[-g]Vy[-h])
   epsilonG[e,f,g,h]epsilonG[a,b,c,d]G[-i,-a]G[-j,-b] (Cos[ϕ]Vt[-c]Vx[-d]+
     Cos[ϕ]Vz[-c]Vx[-d]+Sin[ϕ]Vt[-c]Vy[-d]+Sin[ϕ]Vz[-c]Vy[-d]) //ToNewCanonical;

CWedgeADotBWedgeB=
 -G[-i,-e]G[-j,-f]G[-k,-g]G[-l,-h]epsilonG[e,f,g,h]epsilonG[a,b,c,d]
   (Cos[ϕ]Vt[-a]Vx[-b]+Cos[ϕ]Vz[-a]Vx[-b]+Sin[ϕ]Vt[-a]Vy[-b]+Sin[ϕ]Vz[-a]Vy[-b])
   (Cos[ϕ]Vt[-c]Vx[-d]+Cos[ϕ]Vz[-c]Vx[-d]+Sin[ϕ]Vt[-c]Vy[-d]+Sin[ϕ]Vz[-c]Vy[-d]) //
  ToNewCanonical;

temp=ADotB CDotB-(1/2)CDotA BDotB+
   AWedgeBDotCWedgeB-(1/2)CWedgeADotBWedgeB//ToNewCanonical;
temp=temp//TrigReduce//ToNewCanonical;

Print[temp];
RC2DoF=
 MakeRule[{R[-i,-j,-k,-l],Evaluate[temp]},MetricOn→All,ContractMetrics→True];

Print["manual expansion"];
DefTensor[xV[-i],M4];
DefTensor[yV[-i],M4];
xVExpand=MakeRule[{xV[-i],Vt[-i]+Vz[-i]},MetricOn→All,ContractMetrics→True];
yVExpand=
 MakeRule[{yV[-i],Cos[ϕ]Vx[-i]+Sin[ϕ]Vy[-i]},MetricOn→All,ContractMetrics→True];
VExpand=Join[xVExpand,yVExpand];

temp=2(yV[-i]xV[-j]-xV[-i]yV[-j]) (yV[-l]xV[-k]-xV[-l]yV[-k])-
   2(G[-i,-k]xV[-j]xV[-l]-G[-j,-k]xV[-i]xV[-l]-
      G[-i,-l]xV[-j]xV[-k]+G[-j,-l]xV[-i]xV[-k])+
   (xV[-k]xV[-j]yV[-l]yV[-i]-
     xV[-l]xV[-j]yV[-k]yV[-i]-
     xV[-k]xV[-l]yV[-j]yV[-i]+
     xV[-l]xV[-k]yV[-j]yV[-i]-
     xV[-k]xV[-i]yV[-l]yV[-j]+
```

```
      xV[-l]xV[-i]yV[-k]yV[-j]+
      xV[-k]xV[-l]yV[-i]yV[-j]-
      xV[-l]xV[-k]yV[-i]yV[-j]-
      xV[-j]xV[-i]yV[-k]yV[-l]+
      xV[-j]xV[-i]yV[-l]yV[-k]+
      xV[-l]xV[-i]yV[-k]yV[-j]-
      xV[-k]xV[-i]yV[-l]yV[-j]+
      xV[-i]xV[-j]yV[-k]yV[-l]-
      xV[-i]xV[-j]yV[-l]yV[-k]-
      xV[-l]xV[-j]yV[-k]yV[-i]+
      xV[-k]xV[-j]yV[-l]yV[-i])/.VExpand;
temp=temp//ToNewCanonical;
temp=temp//TrigReduce;
Print[temp];
RC2DoFManual=
 MakeRule[{R[-i,-j,-k,-l],Evaluate[temp]},MetricOn→All,ContractMetrics→True];
*)
```

Riemann-Cartan waves (1 D.o.F)

In[∘]:=
```
(*
temp=(Vt[-i]G[-j,z]-Vt[-j]G[-i,z]+Vz[-i]G[-j,z]-Vz[-j]G[-i,z])
    (Vt[-k]G[-l,-z]-Vt[-l]G[-k,-z]+Vz[-k]G[-l,-z]-Vz[-l]G[-k,-z])//ToNewCanonical;

Print[temp];
RC1DoF=
 MakeRule[{R[-i,-j,-k,-l],Evaluate[temp]},MetricOn→All,ContractMetrics→True];
*)
```

Torsion waves (2 D.o.F)

In[∘]:=
```
(*
temp=2Cos[θ](Vx[-i](Vx[-k](Vt[-j]+Vz[-j])-Vx[-j](Vt[-k]+Vz[-k]))-
      Vy[-i](Vy[-k](Vt[-j]+Vz[-j])-Vy[-j](Vt[-k]+Vz[-k])))+
  2Sin[θ](Vx[-i]Vx[a]-Vy[-i]Vy[a])(Vt[b]+Vz[b])
   (-epsilonG[-j,-k,-a,-b]epsilonG[e,f,g,h])(Vt[-e]Vx[-f]Vy[-g]Vz[-h]);

Print[temp];
T2DoF=MakeRule[{T[-i,-j,-k],Evaluate[temp]},MetricOn→All,ContractMetrics→True];
*)
```

Torsion waves (1 D.o.F)

In[•]:=
```
(*
temp=G[-i,a](Vt[b]+Vz[b])
  (-epsilonG[-j,-k,-a,-b]epsilonG[-e,-f,-g,-h])(Vt[e]Vx[f]Vy[g]Vz[h]);

Print[temp];
T1DoF=MakeRule[{T[-i,-j,-k],Evaluate[temp]},MetricOn→All,ContractMetrics→True];
*)
```

FLRW spacetime

In[•]:=
```
(*
DefConstantSymbol[ScF,PrintAs→"a"];
DefConstantSymbol[Hubble,PrintAs→"H"];
DefConstantSymbol[PsiF,PrintAs→"ψ"];
DefConstantSymbol[DPsiF,PrintAs→"ψ̇"];
DefConstantSymbol[PhiF,PrintAs→"ϕ"];
DefConstantSymbol[DPhiF,PrintAs→"ϕ̇"];
DefTensor[HVal[-j,n],M4];
HValDefinition=(1/ScF) (G[-j,n]-V[-j]V[n])+V[-j]V[n];
HValActivate=MakeRule[{HVal[-j,n],Evaluate[HValDefinition]},
  MetricOn→All,ContractMetrics→True];
DefTensor[BVal[i,-m],M4];
BValDefinition=ScF(G[i,-m]-V[i]V[-m])+V[i]V[-m];
BValActivate=MakeRule[{BVal[i,-m],Evaluate[BValDefinition]},
  MetricOn→All,ContractMetrics→True];
DefTensor[AVal[i,j,-m],M4,Antisymmetric[{i,j}]];
AValDefinition=
 ScF V[k](PhiF (G[j,-m]G[i,-k]-G[i,-m]G[j,-k])/2-(1/2)PsiF epsilonG[-m,-k,i,j]);
AValActivate=MakeRule[{AVal[i,j,-m],Evaluate[AValDefinition]},
  MetricOn→All,ContractMetrics→True];
DefTensor[DAVal[-m,i,j,-n],M4,Antisymmetric[{i,j}]];
DAValDefinition=
 ScF V[-m]V[k]((Hubble PhiF+DPhiF) (G[j,-n]G[i,-k]-G[i,-n]G[j,-k])/2-
    (1/2) (Hubble PsiF+DPsiF) epsilonG[-n,-k,i,j]);
DAValActivate=MakeRule[{DAVal[-m,i,j,-n],Evaluate[DAValDefinition]},
  MetricOn→All,ContractMetrics→True];

ValActivate=Join[HValActivate,BValActivate,AValActivate,DAValActivate];

temp=(HVal[-k,m]HVal[-l,n]-HVal[-l,m]HVal[-k,n])
    (DAVal[-m,i,j,-n]+AVal[i,-z,-m]AVal[z,j,-n])/.ValActivate//ToNewCanonical;

Print[temp];
RCCosmic=
 MakeRule[{R[i,j,-k,-l],Evaluate[temp]},MetricOn→All,ContractMetrics→True];
*)
```

Nester form $\hat{T}$, $\hat{R}$ displayed

In[•]:=
```
(*
RiemannCartanDisplay[rule_]:=Module[{temp1},
```

```
Print[Style["Riemann-Cartan display irreps:",Orange,20]];
Print[Style["Parallel 0⁺:",Blue,20]];
temp1=RP0p[]/.RPO3Activate/.StrengthPToStrength/.PADMActivate//NoScalar;
temp1=temp1/.rule;
temp1=temp1/.Tycho//ToCanonical//ToNewCanonical;
Print[temp1];
Print[Style["Parallel 0⁻:",Blue,20]];
temp1=RP0m[]/.RPO3Activate/.StrengthPToStrength/.PADMActivate//NoScalar;
temp1=temp1/.rule;
temp1=temp1/.Tycho//ToCanonical//ToNewCanonical;
Print[temp1];
Print[Style["Parallel 1⁺:",Blue,20]];
temp1=RP1p[-i,-j]/.RPO3Activate/.StrengthPToStrength/.PADMActivate//NoScalar;
temp1=temp1/.rule;
temp1=temp1/.Tycho//ToCanonical//ToNewCanonical;
Print[temp1];
Print[Style["Parallel 1⁻:",Blue,20]];
temp1=RP1m[-i]/.RPO3Activate/.StrengthPToStrength/.PADMActivate//NoScalar;
temp1=temp1/.rule;
temp1=temp1/.Tycho//ToCanonical//ToNewCanonical;
Print[temp1];
Print[Style["Parallel 2⁺:",Blue,20]];
temp1=RP2p[-i,-j]/.RPO3Activate/.StrengthPToStrength/.PADMActivate//NoScalar;
temp1=temp1/.rule;
temp1=temp1/.Tycho//ToCanonical//ToNewCanonical;
Print[temp1];
Print[Style["Parallel 2⁻:",Blue,20]];
temp1=
 RP2m[-i,-j,-k]/.RPO3Activate/.StrengthPToStrength/.PADMActivate//NoScalar;
temp1=temp1/.rule;
temp1=temp1/.Tycho//ToCanonical//ToNewCanonical;
Print[temp1];
Print[Style["Perpendicular 0⁺:",Red,20]];
temp1=
 RPerp0p[]/.RPerpO3Activate/.StrengthPerpToStrength/.PADMActivate//NoScalar;
temp1=temp1/.rule;
temp1=temp1/.Tycho//ToCanonical//ToNewCanonical;
Print[temp1];
Print[Style["Perpendicular 0⁻:",Red,20]];
temp1=
 RPerp0m[]/.RPerpO3Activate/.StrengthPerpToStrength/.PADMActivate//NoScalar;
temp1=temp1/.rule;
temp1=temp1/.Tycho//ToCanonical//ToNewCanonical;
Print[temp1];
```

```
  Print[Style["Perpendicular 1⁺:",Red,20]];
  temp1=RPerp1p[-i,-j]/.RPerpO3Activate/.StrengthPerpToStrength/.PADMActivate//
    NoScalar;
  temp1=temp1/.rule;
  temp1=temp1/.Tycho//ToCanonical//ToNewCanonical;
  Print[temp1];
  Print[Style["Perpendicular 1⁻:",Red,20]];
  temp1=
   RPerp1m[-i]/.RPerpO3Activate/.StrengthPerpToStrength/.PADMActivate//NoScalar;
  temp1=temp1/.rule;
  temp1=temp1/.Tycho//ToCanonical//ToNewCanonical;
  Print[temp1];
  Print[Style["Perpendicular 2⁺:",Red,20]];
  temp1=RPerp2p[-i,-j]/.RPerpO3Activate/.StrengthPerpToStrength/.PADMActivate//
    NoScalar;
  temp1=temp1/.rule;
  temp1=temp1/.Tycho//ToCanonical//ToNewCanonical;
  Print[temp1];
  Print[Style["Perpendicular 2⁻:",Red,20]];
  temp1=RPerp2m[-i,-j,-k]/.RPerpO3Activate/.StrengthPerpToStrength/.
     PADMActivate//NoScalar;
  temp1=temp1/.rule;
  temp1=temp1/.Tycho//ToCanonical//ToNewCanonical;
  Print[temp1];
 ];

TorsionDisplay[rule_]:=Module[{temp1},
  Print[Style["Torsion display irreps:",Orange,20]];
  Print[Style["Parallel 0⁻:",Blue,20]];
  temp1=TP0m[]/.TPO3Activate/.StrengthPToStrength/.PADMActivate//NoScalar;
  temp1=temp1/.rule;
  temp1=temp1/.Tycho//ToCanonical//ToNewCanonical;
  Print[temp1];
  Print[Style["Parallel 1⁺:",Blue,20]];
  temp1=TP1p[-i,-j]/.TPO3Activate/.StrengthPToStrength/.PADMActivate//NoScalar;
  temp1=temp1/.rule;
  temp1=temp1/.Tycho//ToCanonical//ToNewCanonical;
  Print[temp1];
  Print[Style["Parallel 1⁻:",Blue,20]];
  temp1=TP1m[-i]/.TPO3Activate/.StrengthPToStrength/.PADMActivate//NoScalar;
  temp1=temp1/.rule;
  temp1=temp1/.Tycho//ToCanonical//ToNewCanonical;
  Print[temp1];
  Print[Style["Parallel 2⁻:",Blue,20]];
```

```
   temp1=
    TP2m[-i,-j,-k]/.TPO3Activate/.StrengthPToStrength/.PADMActivate//NoScalar;
   temp1=temp1/.rule;
   temp1=temp1/.Tycho//ToCanonical//ToNewCanonical;
   Print[temp1];
   Print[Style["Perpendicular 0⁺:",Red,20]];
   temp1=
    TPerp0p[]/.TPerpO3Activate/.StrengthPerpToStrength/.PADMActivate//NoScalar;
   temp1=temp1/.rule;
   temp1=temp1/.Tycho//ToCanonical//ToNewCanonical;
   Print[temp1];
   Print[Style["Perpendicular 1⁺:",Red,20]];
   temp1=TPerp1p[-i,-j]/.TPerpO3Activate/.StrengthPerpToStrength/.PADMActivate//
     NoScalar;
   temp1=temp1/.rule;
   temp1=temp1/.Tycho//ToCanonical//ToNewCanonical;
   Print[temp1];
   Print[Style["Perpendicular 1⁻:",Red,20]];
   temp1=
    TPerp1m[-i]/.TPerpO3Activate/.StrengthPerpToStrength/.PADMActivate//NoScalar;
   temp1=temp1/.rule;
   temp1=temp1/.Tycho//ToCanonical//ToNewCanonical;
   Print[temp1];
   Print[Style["Perpendicular 2⁺:",Red,20]];
   temp1=TPerp2p[-i,-j]/.TPerpO3Activate/.StrengthPerpToStrength/.PADMActivate//
     NoScalar;
   temp1=temp1/.rule;
   temp1=temp1/.Tycho//ToCanonical//ToNewCanonical;
   Print[temp1];
  ];


RiemannCartanDisplay[RC2DoF];
RiemannCartanDisplay[RC2DoFManual];
Quit[];
*)
(*
RiemannCartanDisplay[RC1DoF];
TorsionDisplay[T2DoF];
TorsionDisplay[T1DoF];
TorsionDisplay[T1DoF];
*)
(*
TorsionDisplay[T1DoF];
```

```
RiemannCartanDisplay[RCCosmic];

Quit[];
*)
```

Poisson bracket function

```
In[•]:= DefTensor[KX[-a, -b, -c], M4];
DefTensor[KKX[-a, -b, -c, -d], M4, Antisymmetric[{-b, -c}]];
DefTensor[KXP[-a, -b, -c], M4];
DefTensor[KKXP[-a, -b, -c, -d], M4, Antisymmetric[{-b, -c}]];
InertDerB = MakeRule[{CD[-a][B[-b, -c]], KX[-a, -b, -c]},
   MetricOn → All, ContractMetrics → True];
InertDerA = MakeRule[{CD[-a][A[-b, -c, -d]], KKX[-a, -b, -c, -d]},
   MetricOn → All, ContractMetrics → True];
InertDerBP = MakeRule[{CD[-a][BPi[-b, -c]], KXP[-a, -b, -c]},
   MetricOn → All, ContractMetrics → True];
InertDerAP = MakeRule[{CD[-a][APi[-b, -c, -d]], KKXP[-a, -b, -c, -d]},
   MetricOn → All, ContractMetrics → True];
InertDer = Join[InertDerB, InertDerA, InertDerBP, InertDerAP];
InertDerRevB = MakeRule[{KX[-a, -b, -c], CD[-a][B[-b, -c]]},
   MetricOn → All, ContractMetrics → True];
InertDerRevA = MakeRule[{KKX[-a, -b, -c, -d], CD[-a][A[-b, -c, -d]]},
   MetricOn → All, ContractMetrics → True];
InertDerRevBP = MakeRule[{KXP[-a, -b, -c], CD[-a][BPi[-b, -c]]},
   MetricOn → All, ContractMetrics → True];
InertDerRevAP = MakeRule[{KKXP[-a, -b, -c, -d], CD[-a][APi[-b, -c, -d]]},
   MetricOn → All, ContractMetrics → True];
InertDerRev = Join[InertDerRevB, InertDerRevA, InertDerRevBP, InertDerRevAP];

Derivative3B = MakeRule[{CD[-a][B[b, -c]], G3[-a, d] CD[-d][B[b, -c]]},
   MetricOn → All, ContractMetrics → True];
Derivative3A = MakeRule[{CD[-a][A[b, e, -c]], G3[-a, d] CD[-d][A[b, e, -c]]},
   MetricOn → All, ContractMetrics → True];
Derivative3 = Join[Derivative3B, Derivative3A];

ForceAToZeroExplicit =
  MakeRule[{A[i, j, -k], 0}, MetricOn → All, ContractMetrics → True];
 (*This is included for the surface terms, use very carefully!*)
ForceQToZero = MakeRule[{Q[i, j], 0}, MetricOn → All, ContractMetrics → True];
(*This is included for the surface terms, use very carefully!*)
ForceAToZero = Join[ForceAToZeroExplicit, ForceQToZero];
```

```
DefTensor[DummyGradient[-z], M4, PrintAs → "𝔇̂", OrthogonalTo → {V[z]}];
DefTensor[DummyHessian[-z, -w], M4, PrintAs → "𝔇̂𝔇̂", OrthogonalTo → {V[z], V[w]}];
DefTensor[DummyGradientGreek[-z], M4, PrintAs → "𝔇"];
DefTensor[DummyHessianGreek[-z, -w], M4, PrintAs → "𝔇𝔇"];
DummyGradientGreekActivate =
  MakeRule[{DummyGradientGreek[-b], DummyGradient[-i] B[i, -a] G3[a, -b]},
   MetricOn → All, ContractMetrics → True];
DummyHessianGreekActivate = MakeRule[{DummyHessianGreek[-b, -c],
    DummyHessian[-i, -j] B[i, -a] G3[a, -b] B[j, -d] G3[d, -c]},
   MetricOn → All, ContractMetrics → True];
DummyDerivativeGreekActivate = Join[DummyGradientGreekActivate,
   DummyHessianGreekActivate];

ManualCovariantDerivative[ind_, expr_, greeks_, dummy_] :=
  Module[{res, Inds, UpperInds, LowerInds},
   Inds = Complement[FindFreeIndices[expr], greeks];
   LowerInds = Select[Inds, (Quiet[#[[1]]] == -1) &];
   UpperInds = Complement[Inds, LowerInds];
   res = CD[ind][expr];
   Scan[(res = res - A[dummy, #, ind] ReplaceIndex[Evaluate[expr], # → -dummy]) &,
    LowerInds];
   Scan[(res = res + A[#, -dummy, ind] ReplaceIndex[Evaluate[expr], # → dummy]) &,
    UpperInds];
   res = res // ToNewCanonical;
   res];

Options[PoissonBracket] =
  {"ToShell" → True, "Hard" → False, "Surficial" → False, "Order" → Infinity,
   "GToFoliG" → True, "PreTruncate" → False, "NesterForm" → True};
PoissonBracket[f1x_, f2x_, OptionsPattern[]] :=
  Module[{sur, sur1, sur2, res, ris, f1, f2, f1a, f2a, f1b, f2b, nf1,
    nf2, NonVanishing, final, failtrue, BracketForm, BracketAnsatzFull,
    BracketAnsatz, BracketSolution, AnsatzSolutions, difference, ret,
    test, Variationalf1B, Variationalf2B, Variationalf1A, Variationalf2A,
    Variationalf1BPi, Variationalf2BPi, Variationalf1APi, Variationalf2APi,
    Partialf1B, Partialf2B, Partialf1A, Partialf2A, Partialf1BPi, Partialf2BPi,
    Partialf1APi, Partialf2APi, Partialf1DBz, Partialf2DBz, Partialf1DAz,
    Partialf2DAz, Partialf1DBPiz, Partialf2DBPiz, Partialf1DAPiz,
    Partialf2DAPiz, Partialf1DBv, Partialf2DBv, Partialf1DAv, Partialf2DAv,
    Partialf1DBPiv, Partialf2DBPiv, Partialf1DAPiv, Partialf2DAPiv,
    BarPartialf1B, BarPartialf2B, BarPartialf1A, BarPartialf2A, BarPartialf1BPi,
    BarPartialf2BPi, BarPartialf1APi, BarPartialf2APi, BarVariationalf1B,
    BarVariationalf2B, BarVariationalf1A, BarVariationalf2A, BarVariationalf1BPi,
```

```
 BarVariationalf2BPi, BarVariationalf1APi, BarVariationalf2APi,
 DeltaDelta, DDeltaDelta, DeltaDDelta, DDeltaDDelta, return, fieldversion,
 momentafail, ras, D0Term, D1Term, D2Term, D0TermPrimitive, SecondIndices},
Print[Style["Evaluating local Poisson between:", Purple, 20]];
Print[f1x];
Print[f2x];
Print[Style["Expanding constraints...", Orange, 10]];
f1 = ToBasicForm[f1x, "Hard" → True];
f1 = f1 // NoScalar;
If[OptionValue["PreTruncate"], f1 = ToOrderCanonical[f1, 1]];
Print[f1 // ScreenDollarIndices];
f2 = ToBasicForm[f2x, "Hard" → True];
f2 = f2 // NoScalar;
If[OptionValue["PreTruncate"], f2 = ToOrderCanonical[f2, 1]];
Print[f2 // ScreenDollarIndices];
nf1 = Length[FindFreeIndices[f1]];
nf2 = Length[FindFreeIndices[f2]];
f1a = ReplaceDummies[f1];
f2a = ReplaceDummies[f2];
BracketForm = f1x f2x // ToCanonical;
f1a = f1a /. Derivative3;
f2a = f2a /. Derivative3;
f1b = f1a /. InertDer;
f1b = f1b // NoScalar;
f2b = f2a /. InertDer;
f2b = f2b // NoScalar;
Print[Style["Taking variational derivatives...", Orange, 10]];
Variationalf1B = VarAction[f1a, B[q, -r]] + DVDB[-x, -q, r] VarAction[f1a, V[-x]] +
  DHDB[-x, y, -q, r] VarAction[f1a, H[-x, y]] + DJDB[-q, r] VarAction[f1a, J[]] +
  DLapseDB[-q, r] VarAction[f1a, Lapse[]] + DJiDB[-q, r] VarAction[f1a, Ji[]];
Variationalf2B = VarAction[f2a, B[q, -r]] + DVDB[-x, -q, r] VarAction[f2a, V[-x]] +
  DHDB[-x, y, -q, r] VarAction[f2a, H[-x, y]] + DJDB[-q, r] VarAction[f2a, J[]] +
  DLapseDB[-q, r] VarAction[f2a, Lapse[]] + DJiDB[-q, r] VarAction[f2a, Ji[]];
Variationalf1A = VarAction[f1a, A[q, r, -s]];
Variationalf2A = VarAction[f2a, A[q, r, -s]];
Variationalf1BPi = VarAction[f1a, BPi[-q, r]];
Variationalf2BPi = VarAction[f2a, BPi[-q, r]];
Variationalf1APi = VarAction[f1a, APi[-q, -r, s]];
Variationalf2APi = VarAction[f2a, APi[-q, -r, s]];
Print[Style["Taking partial derivatives...", Orange, 10]];
Partialf1B = VarAction[f1b, B[q, -r]] + DVDB[-x, -q, r] VarAction[f1b, V[-x]] +
  DHDB[-x, y, -q, r] VarAction[f1b, H[-x, y]] + DJDB[-q, r] VarAction[f1b, J[]] +
  DLapseDB[-q, r] VarAction[f1b, Lapse[]] + DJiDB[-q, r] VarAction[f1b, Ji[]];
Partialf2B = VarAction[f2b, B[q, -r]] + DVDB[-x, -q, r] VarAction[f2b, V[-x]] +
```

```
    DHDB[-x, y, -q, r] VarAction[f2b, H[-x, y]] + DJDB[-q, r] VarAction[f2b, J[]] +
    DLapseDB[-q, r] VarAction[f2b, Lapse[]] + DJiDB[-q, r] VarAction[f2b, Ji[]];
  Partialf1A = VarAction[f1b, A[q, r, -s]];
  Partialf2A = VarAction[f2b, A[q, r, -s]];
  Partialf1BPi = VarAction[f1b, BPi[-q, r]];
  Partialf2BPi = VarAction[f2b, BPi[-q, r]];
  Partialf1APi = VarAction[f1b, APi[-q, -r, s]];
  Partialf2APi = VarAction[f2b, APi[-q, -r, s]];
  Partialf1DBz = VarAction[f1b, KX[-z, q, -r]];
  Partialf2DBz = VarAction[f2b, KX[-z, q, -r]];
  Partialf1DAz = VarAction[f1b, KKX[-z, q, r, -s]];
  Partialf2DAz = VarAction[f2b, KKX[-z, q, r, -s]];
  Partialf1DBPiz = VarAction[f1b, KXP[-z, -q, r]];
  Partialf2DBPiz = VarAction[f2b, KXP[-z, -q, r]];
  Partialf1DAPiz = VarAction[f1b, KKXP[-z, -q, -r, s]];
  Partialf2DAPiz = VarAction[f2b, KKXP[-z, -q, -r, s]];
  Partialf1DBv = VarAction[f1b, KX[-v, q, -r]];
  Partialf2DBv = VarAction[f2b, KX[-v, q, -r]];
  Partialf1DAv = VarAction[f1b, KKX[-v, q, r, -s]];
  Partialf2DAv = VarAction[f2b, KKX[-v, q, r, -s]];
  Partialf1DBPiv = VarAction[f1b, KXP[-v, -q, r]];
  Partialf2DBPiv = VarAction[f2b, KXP[-v, -q, r]];
  Partialf1DAPiv = VarAction[f1b, KKXP[-v, -q, -r, s]];
  Partialf2DAPiv = VarAction[f2b, KKXP[-v, -q, -r, s]];
  If[OptionValue["Surficial"], {
    Print[Style["Finding barred derivatives...", Orange, 10]];
    BarPartialf1B =
     Partialf1B - ReplaceIndex[Evaluate[Partialf1DBz], -q → -w] A[w, -q, -z];
    BarPartialf2B = Partialf2B - ReplaceIndex[Evaluate[Partialf2DBz], -q → -w]
       A[w, -q, -z];
    BarPartialf1A = Partialf1A - ReplaceIndex[Evaluate[Partialf1DAz], -q → -w]
       A[w, -q, -z] - ReplaceIndex[Evaluate[Partialf1DAz], -r → -w] A[w, -r, -z];
    BarPartialf2A = Partialf2A - ReplaceIndex[Evaluate[Partialf2DAz], -q → -w]
       A[w, -q, -z] - ReplaceIndex[Evaluate[Partialf2DAz], -r → -w] A[w, -r, -z];
    BarPartialf1BPi = Partialf1BPi + ReplaceIndex[
        Evaluate[Partialf1DBPiz], q → w] A[q, -w, -z];
    BarPartialf2BPi = Partialf2BPi + ReplaceIndex[
        Evaluate[Partialf2DBPiz], q → w] A[q, -w, -z];
    BarPartialf1APi = Partialf1APi + ReplaceIndex[Evaluate[Partialf1DAPiz], q → w]
       A[q, -w, -z] + ReplaceIndex[Evaluate[Partialf1DAPiz], r → w] A[r, -w, -z];
    BarPartialf2APi = Partialf2APi + ReplaceIndex[Evaluate[Partialf2DAPiz], q → w]
       A[q, -w, -z] + ReplaceIndex[Evaluate[Partialf2DAPiz], r → w] A[r, -w, -z];
    BarVariationalf1B = BarPartialf1B - ManualCovariantDerivative[
       -z, Partialf1DBz, IndexList[z, r], w];
```

```
BarVariationalf2B = BarPartialf2B - ManualCovariantDerivative[
    -z, Partialf2DBz, IndexList[z, r], w];
BarVariationalf1A = BarPartialf1A - ManualCovariantDerivative[
    -z, Partialf1DAz, IndexList[z, s], w];
BarVariationalf2A = BarPartialf2A - ManualCovariantDerivative[
    -z, Partialf2DAz, IndexList[z, s], w];
BarVariationalf1BPi = BarPartialf1BPi - ManualCovariantDerivative[
    -z, Partialf1DBPiz, IndexList[z, -r], w];
BarVariationalf2BPi = BarPartialf2BPi - ManualCovariantDerivative[
    -z, Partialf2DBPiz, IndexList[z, -r], w];
BarVariationalf1APi = BarPartialf1APi - ManualCovariantDerivative[
    -z, Partialf1DAPiz, IndexList[z, -s], w];
BarVariationalf2APi = BarPartialf2APi - ManualCovariantDerivative[
    -z, Partialf2DAPiz, IndexList[z, -s], w];
Print[Style["Finding kernel coefficients...", Orange, 10]];
D0Term = BarPartialf1B BarVariationalf2BPi +
   2 BarPartialf1A BarVariationalf2APi -
   BarPartialf1BPi BarVariationalf2B -
   2 BarPartialf1APi BarVariationalf2A +
   ReplaceIndex[Evaluate[Partialf1DBz], z → t]
    ManualCovariantDerivative[-t, BarVariationalf2BPi, IndexList[-r], u] +
   2 ReplaceIndex[Evaluate[Partialf1DAz], z → t]
    ManualCovariantDerivative[-t, BarVariationalf2APi, IndexList[-s], u] -
   ReplaceIndex[Evaluate[Partialf1DBPiz], z → t]
    ManualCovariantDerivative[-t, BarVariationalf2B, IndexList[r], u] -
   2 ReplaceIndex[Evaluate[Partialf1DAPiz], z → t]
    ManualCovariantDerivative[-t, BarVariationalf2A, IndexList[s], u];
D1Term = (Partialf1DBPiz BarVariationalf2B +
      2 Partialf1DAPiz BarVariationalf2A -
      Partialf1DBz BarVariationalf2BPi -
      2 Partialf1DAz BarVariationalf2APi +
      BarPartialf1BPi Partialf2DBz +
      2 BarPartialf1APi Partialf2DAz -
      BarPartialf1B Partialf2DBPiz -
      2 BarPartialf1A Partialf2DAPiz +
      ReplaceIndex[Evaluate[Partialf1DBPiz], z → w]
       ManualCovariantDerivative[-w, Partialf2DBz, IndexList[z, r], u] +
      2 ReplaceIndex[Evaluate[Partialf1DAPiz], z → w]
       ManualCovariantDerivative[-w, Partialf2DAz, IndexList[z, s], u] -
      ReplaceIndex[Evaluate[Partialf1DBz], z → w]
       ManualCovariantDerivative[-w, Partialf2DBPiz, IndexList[z, -r], u] -
      2 ReplaceIndex[Evaluate[Partialf1DAz], z → w] ManualCovariantDerivative[
        -w, Partialf2DAPiz, IndexList[z, -s], u]) CD[-z][HComp[]];
```

```
  D2Term = Partialf1DBz ReplaceIndex[Evaluate[Partialf2DBPiz], z → w] +
    2 Partialf1DAz ReplaceIndex[Evaluate[Partialf2DAPiz], z → w] -
    Partialf1DBPiz ReplaceIndex[Evaluate[Partialf2DBz], z → w] -
    2 Partialf1DAPiz ReplaceIndex[Evaluate[Partialf2DAz], z → w];
  Print[Style["Putting into list form and then Nester form...", Orange, 10]];
  res = {D0Term, D1Term, D2Term};
  res = res /. InertDerRev;
  res = res /. Derivative3;
  res = ToNesterForm[#, "ToShell" → OptionValue["ToShell"],
      "Hard" → OptionValue["Hard"], "Order" → OptionValue["Order"],
      "GToFoliG" → OptionValue["GToFoliG"]] & /@ res;
  res = CollectTensors /@ res;
  Print[res];
 }, {
  Print[Style["Finding old kernel coefficients...", Orange, 10]];
  DeltaDelta =
   Variationalf1B Variationalf2BPi + 2 Variationalf1A Variationalf2APi -
    Variationalf1BPi Variationalf2B - 2 Variationalf1APi Variationalf2A;
  DDeltaDelta = -Partialf1DBv Variationalf2BPi - 2 Partialf1DAv Variationalf2APi +
    Partialf1DBPiv Variationalf2B + 2 Partialf1DAPiv Variationalf2A;
  DeltaDDelta = -Variationalf1B Partialf2DBPiv - 2 Variationalf1A Partialf2DAPiv +
    Variationalf1BPi Partialf2DBv + 2 Variationalf1APi Partialf2DAv;
  DDeltaDDelta = Partialf1DBz Partialf2DBPiv + 2 Partialf1DAz Partialf2DAPiv -
    Partialf1DBPiz Partialf2DBv - 2 Partialf1DAPiz Partialf2DAv;
  Print[Style["Putting into list form and then Nester form...", Orange, 10]];
  res = {DeltaDelta, DDeltaDelta, DeltaDDelta, DDeltaDDelta};
  res = res /. InertDerRev;
  res = res /. Derivative3;
  Print[ScreenDollarIndices /@ res];
  If[OptionValue["NesterForm"], {
    res = ToNesterForm[#, "ToShell" → OptionValue["ToShell"],
         "Hard" → OptionValue["Hard"], "Order" → OptionValue["Order"],
         "GToFoliG" → OptionValue["GToFoliG"]] & /@ res;
   }, {
    res = ToBasicForm[res,
       "Hard" -> OptionValue["Hard"], "Order" → OptionValue["Order"]];
   }];
  res = CollectTensors /@ res;
  Print[res];
 }];
res];
```

Poisson bracket testing cell

```
(*
tmp=PoissonBracket[PhiB1m[-q1],
  -V[k]G3[m,-n](CD[-m][BPi[-k,n]]-A[w,-k,-m]BPi[-w,n]),
  "ToShell"→True,"Hard"→True,"Surficial"→False,"Order"→Infinity];
Print[tmp];
Quit[];
*)
(*
tmp=PoissonBracket[PhiB1p[-i,-j],PhiA1p[-l,-m],
  "ToShell"→True,"Hard"→True,"Surficial"→False,"Order"→0];
Print[tmp];
Quit[];
*)
(*
tmp=PoissonBracket[PhiA2m[-q1,-p1,-v1],
  -V[k]G3[m,-n](CD[-m][BPi[-k,n]]-A[w,-k,-m]BPi[-w,n]),
  "ToShell"→True,"Hard"→True,"Surficial"→False,"Order"→1];
Print[tmp];
Quit[];
*)
(*
Print[Style["Here is trial eval",Red,30]];
Temp={0,0,1/2 H[-v1,v] Ji[] PiPA1p[-p1,-q1]+1/4 H[-q1,v] Ji[] PiPA1p[-p1,-v1]-
   1/4 H[-p1,v] Ji[] PiPA1p[-q1,-v1]+3/4 H[-q1,v] Ji[] PiPA2p[-p1,-v1]-
   3/4 H[-p1,v] Ji[] PiPA2p[-q1,-v1]+1/2 Alp6 FoliG[-q1,-v1] H[-p1,v] RP0p[]-
   1/2 Alp6 FoliG[-p1,-v1] H[-q1,v] RP0p[]+
   6 Alp6 FoliG[-q1,-v1] H[a,v] RP1p[-p1,-a]-
   8 Alp6 H[-v1,v] RP1p[-p1,-q1]-4 Alp6 H[-q1,v] RP1p[-p1,-v1]-
   6 Alp6 FoliG[-p1,-v1] H[a,v] RP1p[-q1,-a]+4 Alp6 H[-p1,v] RP1p[-q1,-v1]+
   6 Alp6 FoliG[-q1,-v1] H[a,v] RP2p[-p1,-a]-12 Alp6 H[-q1,v] RP2p[-p1,-v1]-
   6 Alp6 FoliG[-p1,-v1] H[a,v] RP2p[-q1,-a]+12 Alp6 H[-p1,v] RP2p[-q1,-v1]-
   3/8 APi[-q1,-a,a1] FoliG[-p1,-v1] G3[v,-a1] Ji[] V[a]+
   3/8 APi[-p1,-a,a1] FoliG[-q1,-v1] G3[v,-a1] Ji[] V[a]+
   1/4 APi[-q1,-v1,a] G3[v,-a] Ji[] V[-p1]+
   1/24 Eps[-q1,-v1,-a] H[a,v] Ji[] PiPA0m[] V[-p1]+
   1/8 H[-v1,v] Ji[] PiPA1m[-q1] V[-p1]-1/8 H[-q1,v] Ji[] PiPA1m[-v1] V[-p1]+
   16/3 Alp6 H[a,v] RP2m[-q1,-v1,-a] V[-p1]+
   1/4 H[a,v] Ji[] PiPA1p[-q1,-v1] V[-a] V[-p1]+
   3/4 H[a,v] Ji[] PiPA2p[-q1,-v1] V[-a] V[-p1]-1/2 Alp6 FoliG[-q1,-v1]
    H[a,v] RP0p[] V[-a] V[-p1]-4 Alp6 H[a,v] RP1p[-q1,-v1] V[-a] V[-p1]-
   12 Alp6 H[a,v] RP2p[-q1,-v1] V[-a] V[-p1]-1/4 APi[-p1,-v1,a] G3[v,-a]
    Ji[] V[-q1]-1/24 Eps[-p1,-v1,-a] H[a,v] Ji[] PiPA0m[] V[-q1]-
```

```
1/8 H[-v1,v] Ji[] PiPA1m[-p1] V[-q1]+1/8 H[-p1,v] Ji[] PiPA1m[-v1] V[-q1]-
16/3 Alp6 H[a,v] RP2m[-p1,-v1,-a] V[-q1]-
1/4 H[a,v] Ji[] PiPA1p[-p1,-v1] V[-a] V[-q1]-
3/4 H[a,v] Ji[] PiPA2p[-p1,-v1] V[-a] V[-q1]+
1/2 Alp6 FoliG[-p1,-v1] H[a,v] RP0p[] V[-a] V[-q1]+
4 Alp6 H[a,v] RP1p[-p1,-v1] V[-a] V[-q1]+
12 Alp6 H[a,v] RP2p[-p1,-v1] V[-a] V[-q1]-1/2 APi[-p1,-q1,a] G3[v,-a]
 Ji[] V[-v1]-1/12 Eps[-p1,-q1,-a] H[a,v] Ji[] PiPA0m[] V[-v1]-
1/4 H[-q1,v] Ji[] PiPA1m[-p1] V[-v1]+1/4 H[-p1,v] Ji[] PiPA1m[-q1] V[-v1]-
32/3 Alp6 H[a,v] RP2m[-p1,-q1,-a] V[-v1]-
1/2 H[a,v] Ji[] PiPA1p[-p1,-q1] V[-a] V[-v1]+8 Alp6 H[a,v] RP1p[-p1,-q1]
 V[-a] V[-v1]-3/8 H[a,v] Ji[] PiPA1m[-q1] V[-a] V[-p1] V[-v1]-
3/4 APi[-q1,-a,a1] G3[v,-a1] Ji[] V[a] V[-p1] V[-v1]+
3/8 H[a,v] Ji[] PiPA1m[-p1] V[-a] V[-q1] V[-v1]+
3/4 APi[-p1,-a,a1] G3[v,-a1] Ji[] V[a] V[-q1] V[-v1],0};
GradTemp=CD[-u][Evaluate[Temp[[3]]]];
GradTemp=ToNesterForm[GradTemp,"ToShell"→True,"Hard"→True,"Order"→1];
Quit[];
*)
```

Commutators

ϕ-ϕ

Commutator ansätze

In[∘]:=
```
(*
DefNiceConstantSymbol[γ,#]&/@Range[20];
PPMGuessList=ToExpression["γ"<>ToString[#]]&/@Range[20];
Print[PPMGuessList];
PPMGuessMat=Table[0,{i,10},{j,10}];
PPMGuessMat[[2,5]]=γ1 Ji[]^2 PiPA1p[-a,-b]+γ2 Ji[]RP1p[-a,-b];
temp=PA2m[-e,-f,-g,i,j,k]Antisymmetrize[Antisymmetrize[
    PPara[-k,-a]epsilonG[-b,-i,-j,-m]V[m](γ1 Ji[]^2PiPA0m[]+γ2 Ji[] RP0m[])+
     PPara[-k,-i]epsilonG[-j,-a,-b,-m]V[m](γ3 Ji[]^2PiPA0m[]+γ4 Ji[] RP0m[])+
     PPara[-k,-i]PPara[-j,-a](γ5 Ji[]^2 PiPA1m[-b]+γ5 Ji[] RP1m[-b])+
     PPara[-k,-a]PPara[-b,-i](γ7 Ji[]^2 PiPA1m[-j]+γ8 Ji[] RP1m[-j])+
     PPara[-i,-a]PPara[-b,-j](γ9 Ji[]^2 PiPA1m[-k]+γ10 Ji[] RP1m[-k])+
     PPara[-i,-a](γ11 Ji[]^2PiPA2m[-b,-j,-k]+γ12 Ji[] RP2m[-b,-j,-k])+
     PPara[-k,-a](γ13 Ji[]^2PiPA2m[-i,-j,-b]+γ14 Ji[] RP2m[-i,-j,-b])+
     PPara[-k,-i](γ15 Ji[]^2PiPA2m[-a,-b,-j]+γ16 Ji[] RP2m[-a,-b,-j])+
```

```
      PPara[-k,-i](γ17 Ji[]^2PiPA2m[-j,-a,-b]+γ18 Ji[] RP2m[-j,-a,-b])+
      PPara[-k,-a](γ19 Ji[]^2PiPA2m[-b,-i,-j]+γ20 Ji[] RP2m[-b,-i,-j]),{-a,-b}],
    {-i,-j}];
temp=temp/.PO3PiActivate;
temp=temp/.PADMActivate;
temp=temp//ToFoli;
Print[temp];
PPMGuessMat[[2,10]]=temp;
temp=PA2m[-e,-f,-g,i,j,k]
   (Symmetrize[PPara[-a,c]PPara[-b,d]-(1/3)PPara[-a,-b]PPara[c,d],{-a,-b}])
   Antisymmetrize[
    PPara[-k,-c]epsilonG[-d,-i,-j,-m]V[m](γ1 Ji[]^2PiPA0m[]+γ2 Ji[] RP0m[])+
     PPara[-k,-i]epsilonG[-j,-c,-d,-m]V[m](γ3 Ji[]^2PiPA0m[]+γ4 Ji[] RP0m[])+
     PPara[-k,-i]PPara[-j,-c](γ5 Ji[]^2 PiPA1m[-d]+γ5 Ji[] RP1m[-d])+
     PPara[-k,-c]PPara[-d,-i](γ7 Ji[]^2 PiPA1m[-j]+γ8 Ji[] RP1m[-j])+
     PPara[-i,-c]PPara[-d,-j](γ9 Ji[]^2 PiPA1m[-k]+γ10 Ji[] RP1m[-k])+
     PPara[-i,-c](γ11 Ji[]^2PiPA2m[-d,-j,-k]+γ12 Ji[] RP2m[-d,-j,-k])+
     PPara[-k,-c](γ13 Ji[]^2PiPA2m[-i,-j,-d]+γ14 Ji[] RP2m[-i,-j,-d])+
     PPara[-k,-i](γ15 Ji[]^2PiPA2m[-c,-d,-j]+γ16 Ji[] RP2m[-c,-d,-j])+
     PPara[-k,-i](γ17 Ji[]^2PiPA2m[-j,-c,-d]+γ18 Ji[] RP2m[-j,-c,-d])+
     PPara[-k,-c](γ19 Ji[]^2PiPA2m[-d,-i,-j]+γ20 Ji[] RP2m[-d,-i,-j]),{-i,-j}];
temp=temp/.PO3PiActivate;
temp=temp/.PADMActivate;
temp=temp//ToFoli;
Print[temp];
PPMGuessMat[[4,10]]=temp;
temp=Ji[]PA2m[-a,-b,-c,p,q,r]PA2m[-e,-f,-g,u,v,w]
   Antisymmetrize[Antisymmetrize[γ1 PPara[-p,-u]PPara[-v,-q]TP1p[-r,-w]+
      γ2 PPara[-p,-r]PPara[-w,-q]TP1p[-u,-v]+
      γ3 PPara[-r,-u]PPara[-v,-w]TP1p[-p,-q]+
      γ4 PPara[-r,-u]PPara[-v,-p]TP1p[-q,-w]+
      γ5 PPara[-w,-u]PPara[-v,-p]TP1p[-q,-r]+
      γ6 PPara[-r,-p]PPara[-q,-u]TP1p[-v,-w]+
      γ7 PPara[-w,-p]PPara[-q,-u]TP1p[-v,-r]+
      γ8 PPara[-r,-w]PPara[-q,-u]TP1p[-v,-p],{-p,-q}],{-u,-v}];
temp=temp/.PO3PiActivate;
temp=temp/.PADMActivate;
temp=temp//ToFoli;
Print[temp];
PPMGuessMat[[10,10]]=temp;
*)
```

Commutators

In[∘]:=
```
(*PoissonBracket[PhiB1p[-a,-b],PhiA0p[]];*)
(*PoissonBracket[PhiB1p[-a,-b],PhiA2m[-e,-f,-g]];*)
(*PoissonBracket[PhiB2p[-a,-b],PhiA0p[]];*)
(*PoissonBracket[PhiB2p[-a,-b],PhiA2m[-e,-f,-g]];*)
(*PoissonBracket[PhiA0p[],PhiA2m[-e,-f,-g]];*)
(*PoissonBracket[PhiA2m[-a,-b,-c],PhiA2m[-e,-f,-g]];*)
```

ϕ-Auto

Commutator ansätze

In[∘]:=
```
(*Watch this space!*)
```

Commutators

```
In[•]:= (*PoissonBracket[PhiB1p[-i,-j],TheA0m[]];*)
(*PoissonBracket[PhiB2p[-i,-j],TheA0m[]];*)
(*PoissonBracket[PhiA0p[],TheA0m[]];*)
(*PoissonBracket[PhiA2m[-i,-j,-k],TheA0m[]];*)
(*PoissonBracket[TheA1p[-i,-j],TheA0m[]];*)
(*PoissonBracket[TheA1m[-i],TheA0m[]];*)
(*PoissonBracket[TheA2p[-i,-j],TheA0m[]];*)
(*PoissonBracket[TheA2m[-i,-j,-k],TheA0m[]];*)
(*PoissonBracket[TheB1p[-i,-j],TheA0m[]];*)
(*PoissonBracket[TheB2m[-i,-j,-k],TheA0m[]];*)

(*PoissonBracket[PhiB1p[-i,-j],TheA1m[-l]];*)
(*PoissonBracket[PhiB2p[-i,-j],TheA1m[-l]];*)
(*PoissonBracket[PhiA0p[],TheA1m[-l]];*)
(*PoissonBracket[PhiA2m[-i,-j,-k],TheA1m[-l]];*)
(*PoissonBracket[TheA1p[-i,-j],TheA1m[-l],1];*) (*BAD*)
(*PoissonBracket[TheA1m[-i],TheA1m[-l],0];*)
(*PoissonBracket[TheA2p[-i,-j],TheA1m[-l],1];*) (*BAD*)
(*PoissonBracket[TheA2m[-i,-j,-k],TheA1m[-l],0];*)
(*PoissonBracket[TheB1p[-i,-j],TheA1m[-l],0];*)
(*PoissonBracket[TheB2m[-i,-j,-k],TheA1m[-l],0];*)

(*PoissonBracket[PhiB1p[-i,-j],TheA1p[-l,-m],1];*)
(*PoissonBracket[PhiB2p[-i,-j],TheA1p[-l,-m],1];*)
(*PoissonBracket[PhiA0p[],TheA1p[-l,-m],0];*)
(*tempx=ToNesterForm[PA2m[-i,-j,-k,-l,-m,-n]/.PO3PiActivate,"Hard"→False]*)

(*PoissonBracket[PhiB1p[-i,-j],TheA1m[-l],1];*)
```

$\phi-\chi^{\|}$

Commutator ansätze

```
In[•]:= (*
DefNiceConstantSymbol[γ,#]&/@Range[20];
PPMGuessList=ToExpression["γ"<>ToString[#]]&/@Range[20];
*)
(*
tmp=PA2m[-l,-m,-n,p,q,r]Antisymmetrize[
   Antisymmetrize[PPara[-r,-i]epsilonG[-j,-p,-q,-s]V[s]γ1 Ji[] RP0m[]+
```

```
      PPara[-r,-p]epsilonG[-q,-i,-j,-s]V[s]γ2 Ji[] RP0m[]+
      PPara[-r,-p]PPara[-q,-i]γ3 Ji[] RP1m[-j]+
      PPara[-r,-i]PPara[-j,-p]γ4 Ji[] RP1m[-q]+
      PPara[-p,-i]PPara[-j,-q]γ5 Ji[] RP1m[-r]+
      PPara[-p,-i]γ6 Ji[] RP2m[-j,-q,-r]+
      PPara[-r,-i]γ7 Ji[] RP2m[-p,-q,-j]+
      PPara[-r,-p]γ8 Ji[] RP2m[-i,-j,-q]+
      PPara[-r,-p]γ9 Ji[] RP2m[-q,-i,-j]+
      PPara[-r,-i]γ10 Ji[] RP2m[-j,-p,-q],{-i,-j}],{-p,-q}];
tmp=tmp/.PO3PiActivate;
tmp=tmp/.PADMActivate;
tmp=ToNesterForm[tmp,1];
Print[tmp];
PhiB1pSiCLorentzParaA2m=tmp;
*)
(*
tmp=PA2m[-l,-m,-n,p,q,r]
  (Symmetrize[PPara[-i,w]PPara[-j,v]-(1/3)PPara[-i,-j]PPara[w,v],{-i,-j}])
  Antisymmetrize[PPara[-r,-w]epsilonG[-v,-p,-q,-s]V[s]γ1 Ji[] RP0m[]+
    PPara[-r,-p]epsilonG[-q,-w,-v,-s]V[s]γ2 Ji[] RP0m[]+
    PPara[-r,-p]PPara[-q,-w]γ3 Ji[] RP1m[-v]+
    PPara[-r,-w]PPara[-v,-p]γ4 Ji[] RP1m[-q]+
    PPara[-p,-w]PPara[-v,-q]γ5 Ji[] RP1m[-r]+
    PPara[-p,-w]γ6 Ji[] RP2m[-v,-q,-r]+
    PPara[-r,-w]γ7 Ji[] RP2m[-p,-q,-v]+
    PPara[-r,-p]γ8 Ji[] RP2m[-w,-v,-q]+
    PPara[-r,-p]γ9 Ji[] RP2m[-q,-w,-v]+
    PPara[-r,-w]γ10 Ji[] RP2m[-v,-p,-q],{-p,-q}];
tmp=tmp/.PO3PiActivate;
tmp=tmp/.PADMActivate;
tmp=ToNesterForm[tmp,1];
Print[tmp];
PhiB2pSiCLorentzParaA2m=tmp;
*)
(*
tmp=Ji[]PA2m[-i,-j,-k,p,q,r]PA2m[-l,-m,-n,u,v,w]
  Antisymmetrize[Antisymmetrize[γ1 PPara[-p,-u]PPara[-v,-q]TP1p[-r,-w]+
      γ2 PPara[-p,-r]PPara[-w,-q]TP1p[-u,-v]+
      γ3 PPara[-r,-u]PPara[-v,-w]TP1p[-p,-q]+
      γ4 PPara[-r,-u]PPara[-v,-p]TP1p[-q,-w]+
      γ5 PPara[-w,-u]PPara[-v,-p]TP1p[-q,-r]+
      γ6 PPara[-r,-p]PPara[-q,-u]TP1p[-v,-w]+
      γ7 PPara[-w,-p]PPara[-q,-u]TP1p[-v,-r]+
      γ8 PPara[-r,-w]PPara[-q,-u]TP1p[-v,-p]+
```

```
    γ9 PPara[-p,-u]PPara[-v,-q]TP1p[-r,-w]+
    γ10 PPara[-p,-r]PPara[-w,-q]DpV[-u,-v]+
    γ11 PPara[-r,-u]PPara[-v,-w]DpV[-p,-q]+
    γ12 PPara[-r,-u]PPara[-v,-p]DpV[-q,-w]+
    γ13 PPara[-w,-u]PPara[-v,-p]DpV[-q,-r]+
    γ14 PPara[-r,-p]PPara[-q,-u]DpV[-v,-w]+
    γ15 PPara[-w,-p]PPara[-q,-u]DpV[-v,-r]+
    γ16 PPara[-r,-w]PPara[-q,-u]DpV[-v,-p],{-p,-q}],{-u,-v}];
tmp=tmp/.PO3PiActivate;
tmp=tmp/.PADMActivate;
tmp=ToNesterForm[tmp,1];
Print[tmp];
PhiA2mSiCLorentzParaA2m=tmp;
*)
```

Commutators

```
In[∘]:= (*PoissonBracket[PhiB1p[-i,-j],SiCLorentzParaA0p[],"Hard"→True];*)
(*PoissonBracket[PhiB2p[-i,-j],SiCLorentzParaA0p[],1];*)
(*PoissonBracket[PhiA0p[],SiCLorentzParaA0p[],1];*)
(*PoissonBracket[PhiA2m[-i,-j,-k],SiCLorentzParaA0p[],1];*)
(*PoissonBracket[PhiB1p[-i,-j],SiCLorentzParaA0m[],1];*)
(*PoissonBracket[PhiB2p[-i,-j],SiCLorentzParaA0m[],1];*)
(*PoissonBracket[PhiA0p[],SiCLorentzParaA0m[],1];*)
(*PoissonBracket[PhiA2m[-i,-j,-k],SiCLorentzParaA0m[],1];*)
(*PoissonBracket[PhiB1p[-i,-j],SiCLorentzParaA1p[-l,-m],"Hard"→True];*)
(*PoissonBracket[PhiB1p[-i,-j],
  SiCLorentzParaA1p[-l,-m],"Hard"→True,"ToShell"→False];*)
(*PoissonBracket[PhiB2p[-i,-j],SiCLorentzParaA1p[-l,-m],1];*)
(*PoissonBracket[PhiA0p[],SiCLorentzParaA1p[-l,-m],1];*)
(*PoissonBracket[PhiA2m[-i,-j,-k],SiCLorentzParaA1p[-l,-m],1];*)(*ERROR*)
(*PoissonBracket[PhiB1p[-i,-j],SiCLorentzParaA1m[-l],1];*)
(*PoissonBracket[PhiB2p[-i,-j],SiCLorentzParaA1m[-l],1];*)
(*PoissonBracket[PhiA0p[],SiCLorentzParaA1m[-l],1];*)(*ERROR*)
(*PoissonBracket[PhiA2m[-i,-j,-k],SiCLorentzParaA1m[-l],1];*)
(*PoissonBracket[PhiB1p[-i,-j],SiCLorentzParaA2p[-l,-m],1];*)
(*PoissonBracket[PhiB2p[-i,-j],SiCLorentzParaA2p[-l,-m],1];*)
(*PoissonBracket[PhiA0p[],SiCLorentzParaA2p[-l,-m],1];*)
(*PoissonBracket[PhiA2m[-i,-j,-k],SiCLorentzParaA2p[-l,-m],1];*)(*ERROR*)
(*
tmp=PoissonBracket[PhiB1p[-i,-j],SiCLorentzParaA2m[-l,-m,-n],1][[1]];
tmp=tmp-PhiB1pSiCLorentzParaA2m//CollectTensors;
Print[tmp];
CommutatorGuessList=ToExpression["γ"<>ToString[#]]&/@Range[10];
```

```
Print[CommutatorGuessList];
eqs=ToConstantSymbolEquations[tmp==0];
Print[eqs];
sol=Solve[eqs];
Print[sol];
*)
(*
tmp=PoissonBracket[PhiB2p[-i,-j],SiCLorentzParaA2m[-l,-m,-n],1][[1]];
tmp=tmp-PhiB2pSiCLorentzParaA2m//CollectTensors;
Print[tmp];
CommutatorGuessList=ToExpression["γ"<>ToString[#]]&/@Range[10];
Print[CommutatorGuessList];
eqs=ToConstantSymbolEquations[tmp==0];
Print[eqs];
sol=Solve[eqs];
Print[sol];
*)
(*PoissonBracket[PhiA0p[],SiCLorentzParaA2m[-l,-m,-n],1];*)
(*
tmp=PoissonBracket[PhiA2m[-i,-j,-k],SiCLorentzParaA2m[-l,-m,-n],1][[1]];
tmp=tmp-PhiA2mSiCLorentzParaA2m//CollectTensors;
Print[tmp];
CommutatorGuessList=ToExpression["γ"<>ToString[#]]&/@Range[16];
Print[CommutatorGuessList];
eqs=ToConstantSymbolEquations[tmp==0];
Print[eqs];
sol=Solve[eqs];
Print[sol];

Quit[];
*)

(*PoissonBracket[PhiA0p[],
  SiCLorentzParaB2m[-l,-m,-n],"Hard"→True,"ToShell"→False];*)
(*PoissonBracket[PhiA2m[-i,-j,-k],SiCLorentzParaB2m[-l,-m,-n],
  "Hard"→True,"ToShell"→False];*)
```

$\phi - \chi^{\perp}$

Commutator ansätze

In[∘]:=

```
DefNiceConstantSymbol[γ, #] & /@ Range[20];
```

```
PPMGuessList = ToExpression["γ" <> ToString[#]] & /@ Range[40];

(*
tmp=Ji[]PA2m[-i,-j,-k,p,q,r]Antisymmetrize[γ1 Eps[-p,-q,-u]DpV[u,-r]+
    γ2 Eps[-p,-q,-u]DpV[-r,u]+
    γ3 Eps[-p,-r,-u]DpV[u,-q]+
    γ4 Eps[-p,-r,-u]DpV[-q,u]+
    γ5 PPara[-p,-r]Eps[-q,-u,-v]TP1p[u,v]+
    γ6 Eps[-p,-q,-u]TP1p[u,-r]+
    γ7 Eps[-p,-r,-u]TP1p[u,-q],{-p,-q}];
tmp=tmp/.PO3PiActivate;
tmp=tmp/.PADMActivate;
tmp=ToNesterForm[tmp,"Hard"→True,"ToShell"→False];
Print[tmp];
PhiA2mSiCLorentzPerpA0m=tmp;
*)
(*
tmp=Ji[]PA2m[-i,-j,-k,p,q,r]Antisymmetrize[γ1 PPara[-p,-l]TP1p[-q,-r]+
    γ2 PPara[-p,-r]TP1p[-q,-l]+
    γ3 PPara[-l,-r]TP1p[-p,-q],{-p,-q}];
tmp=tmp/.PO3PiActivate;
tmp=tmp/.PADMActivate;
tmp=ToNesterForm[tmp,"Hard"→True,"ToShell"→False];
Print[tmp];
PhiA2mSiCLorentzPerpA1m=tmp;
*)
(*
tmp=PA2m[-l,-m,-n,p,q,r]Antisymmetrize[
   Antisymmetrize[PPara[-r,-i]epsilonG[-j,-p,-q,-s]V[s]γ1 Ji[]Ji[] PiPA0m[]+
     PPara[-r,-p]epsilonG[-q,-i,-j,-s]V[s]γ2 Ji[]Ji[] PiPA0m[]+
     PPara[-r,-p]PPara[-q,-i]γ3 Ji[]Ji[] PiPA1m[-j]+
     PPara[-r,-i]PPara[-j,-p]γ4 Ji[]Ji[] PiPA1m[-q]+
     PPara[-p,-i]PPara[-j,-q]γ5 Ji[]Ji[] PiPA1m[-r]+
     PPara[-p,-i]γ6 Ji[]Ji[] PiPA2m[-j,-q,-r]+
     PPara[-r,-i]γ7 Ji[]Ji[] PiPA2m[-p,-q,-j]+
     PPara[-r,-p]γ8 Ji[] Ji[]PiPA2m[-i,-j,-q]+
     PPara[-r,-p]γ9 Ji[] Ji[]PiPA2m[-q,-i,-j]+
     PPara[-r,-i]γ10 Ji[]Ji[] PiPA2m[-j,-p,-q],{-i,-j}],{-p,-q}];
tmp=tmp/.PO3PiActivate;
tmp=tmp/.PADMActivate;
tmp=ToNesterForm[tmp,"Hard"→True,"ToShell"→False];
Print[tmp];
PhiB1pSiCLorentzPerpA2m=tmp;
*)
```

```
(*
tmp=PA2m[-l,-m,-n,p,q,r]
  (Symmetrize[PPara[-i,w]PPara[-j,v]-(1/3)PPara[-i,-j]PPara[w,v],{-i,-j}])
  Antisymmetrize[PPara[-r,-w]epsilonG[-v,-p,-q,-s]V[s]γ1 Ji[]Ji[] PiPA0m[]+
    PPara[-r,-p]epsilonG[-q,-w,-v,-s]V[s]γ2 Ji[]Ji[] PiPA0m[]+
    PPara[-r,-p]PPara[-q,-w]γ3 Ji[] Ji[]PiPA1m[-v]+
    PPara[-r,-w]PPara[-v,-p]γ4 Ji[]Ji[] PiPA1m[-q]+
    PPara[-p,-w]PPara[-v,-q]γ5 Ji[]Ji[] PiPA1m[-r]+
    PPara[-p,-w]γ6 Ji[] Ji[]PiPA2m[-v,-q,-r]+
    PPara[-r,-w]γ7 Ji[]Ji[] PiPA2m[-p,-q,-v]+
    PPara[-r,-p]γ8 Ji[] Ji[]PiPA2m[-w,-v,-q]+
    PPara[-r,-p]γ9 Ji[]Ji[] PiPA2m[-q,-w,-v]+
    PPara[-r,-w]γ10 Ji[] Ji[]PiPA2m[-v,-p,-q],{-p,-q}];
tmp=tmp/.PO3PiActivate;
tmp=tmp/.PADMActivate;
tmp=ToNesterForm[tmp,"Hard"→True,"ToShell"→False];
Print[tmp];
PhiB2pSiCLorentzPerpA2m=tmp;
*)
```

Commutators

```
In[]:= (*PoissonBracket[PhiB1p[-i,-j],
  SiCLorentzPerpA0p[],"Hard"→True,"ToShell"→False];*)
(*PoissonBracket[PhiB2p[-i,-j],SiCLorentzPerpA0p[],
  "Hard"→True,"ToShell"→False];*)
(*PoissonBracket[PhiA2m[-i,-j,-k],SiCLorentzPerpA0p[],
  "Hard"→True,"ToShell"→False];*)

(*PoissonBracket[PhiB1p[-i,-j],
  SiCLorentzPerpA0m[],"Hard"→True,"ToShell"→False];*)
(*PoissonBracket[PhiB2p[-i,-j],SiCLorentzPerpA0m[],
  "Hard"→True,"ToShell"→False];*)
(*PoissonBracket[PhiA0p[],SiCLorentzPerpA0m[],"Hard"→True,"ToShell"→False];*)
(*
tmp=PoissonBracket[PhiA2m[-i,-j,-k],
   SiCLorentzPerpA0m[],"Hard"→True,"ToShell"→False][[1]];
tmp=tmp-PhiA2mSiCLorentzPerpA0m//CollectTensors;
Print[tmp];
eqs=ToConstantSymbolEquations[tmp==0];
Print[eqs];
sol=Solve[eqs];
Print[sol];
*)
```

```
(*PoissonBracket[PhiB1p[-i,-j],
  SiCLorentzPerpA1p[-l,-m],"Hard"→True,"ToShell"→False];*)
(*PoissonBracket[PhiB2p[-i,-j],SiCLorentzPerpA1p[-l,-m],
  "Hard"→True,"ToShell"→False];*)
(*PoissonBracket[PhiA0p[],SiCLorentzPerpA1p[-l,-m],
  "Hard"→True,"ToShell"→False];*)
(*PoissonBracket[PhiA2m[-i,-j,-k],SiCLorentzPerpA1p[-l,-m],
  "Hard"→True,"ToShell"→False];*)(*FAILED VIA COMPLEXITY ONLY*)

(*PoissonBracket[PhiB1p[-i,-j],
  SiCLorentzPerpA1m[-l],"Hard"→True,"ToShell"→False];*)
(*PoissonBracket[PhiB2p[-i,-j],SiCLorentzPerpA1m[-l],
  "Hard"→True,"ToShell"→False];*)
(*PoissonBracket[PhiA0p[],SiCLorentzPerpA1m[-l],"Hard"→True,"ToShell"→False];*)
(*FAILED*)
(*
tmp=PoissonBracket[PhiA2m[-i,-j,-k],
   SiCLorentzPerpA1m[-l],"Hard"→True,"ToShell"→False][[1]];
tmp=tmp-PhiA2mSiCLorentzPerpA1m//CollectTensors;
Print[tmp];
eqs=ToConstantSymbolEquations[tmp==0];
Print[eqs];
sol=Solve[eqs];
Print[sol];
*)

(*PoissonBracket[PhiB1p[-i,-j],
  SiCLorentzPerpA2p[-l,-m],"Hard"→True,"ToShell"→False];*)
(*PoissonBracket[PhiB2p[-i,-j],SiCLorentzPerpA2p[-l,-m],
  "Hard"→True,"ToShell"→False];*)
(*PoissonBracket[PhiA0p[],SiCLorentzPerpA2p[-l,-m],
  "Hard"→True,"ToShell"→False];*)
(*PoissonBracket[PhiA2m[-i,-j,-k],SiCLorentzPerpA2p[-l,-m],
  "Hard"→True,"ToShell"→False];*)(*FAILED*)
(*
tmp=PoissonBracket[PhiB1p[-i,-j],
   SiCLorentzPerpA2m[-l,-m,-n],"Hard"→True,"ToShell"→False][[1]];
tmp=tmp-PhiB1pSiCLorentzPerpA2m//CollectTensors;
Print[tmp];
eqs=ToConstantSymbolEquations[tmp==0];
Print[eqs];
sol=Solve[eqs];
Print[sol];
```

```
*)
(*
tmp=PoissonBracket[PhiB2p[-i,-j],
   SiCLorentzPerpA2m[-l,-m,-n],"Hard"→True,"ToShell"→False][[1]];
Print[tmp];
tmp=tmp-PhiB2pSiCLorentzPerpA2m//CollectTensors;
Print[tmp];
eqs=ToConstantSymbolEquations[tmp==0];
Print[eqs];
sol=Solve[eqs];
Print[sol];
*)
(*PoissonBracket[PhiA0p[],
  SiCLorentzPerpA2m[-l,-m,-n],"Hard"→True,"ToShell"→False];*)
(*PoissonBracket[PhiA0p[],SiCLorentzPerpB1m[-l],"Hard"→True,"ToShell"→False];*)
(*PoissonBracket[PhiA2m[-i,-j,-k],
  SiCLorentzPerpB1m[-l],"Hard"→True,"ToShell"→False];*)
```

$\chi^{\parallel}$-$\chi^{\perp}$

Commutator ansätze

In[∘]:=

```
DefNiceConstantSymbol[γ, #] & /@ Range[80];
PPMGuessList = ToExpression["γ" <> ToString[#]] & /@ Range[80];
(*
tmp=Ji[]PA2m[-i,-j,-k,p,q,r]Antisymmetrize[γ1 Eps[-p,-q,-u]DpV[u,-r]+
    γ2 Eps[-p,-q,-u]DpV[-r,u]+
    γ3 Eps[-p,-r,-u]DpV[u,-q]+
    γ4 Eps[-p,-r,-u]DpV[-q,u]+
    γ5 PPara[-p,-r]Eps[-q,-u,-v]TP1p[u,v]+
    γ6 Eps[-p,-q,-u]TP1p[u,-r]+
    γ7 Eps[-p,-r,-u]TP1p[u,-q],{-p,-q}];
tmp=tmp/.PO3PiActivate;
tmp=tmp/.PADMActivate;
tmp=ToNesterForm[tmp,"Hard"→True,"ToShell"→False];
Print[tmp];
SiCLorentzParaA2mSiCLorentzPerpA0m=tmp;
*)
(*
tmp=Ji[]PA2m[-i,-j,-k,p,q,r]Antisymmetrize[γ1 PPara[-p,-l]TP1p[-q,-r]+
    γ2 PPara[-p,-r]TP1p[-q,-l]+
```

```
    γ3 PPara[-l,-r]TP1p[-p,-q],{-p,-q}];
tmp=tmp/.PO3PiActivate;
tmp=tmp/.PADMActivate;
tmp=ToNesterForm[tmp,"Hard"→True,"ToShell"→False];
Print[tmp];
SiCLorentzParaA2mSiCLorentzPerpA1m=tmp;
*)
(*
tmp=Ji[] (Symmetrize[PPara[-i,w]PPara[-j,v]-(1/3)PPara[-i,-j]PPara[w,v],{-i,-j}])
  (Symmetrize[PPara[-l,p]PPara[-m,q]-(1/3)PPara[-l,-m]PPara[p,q],{-l,-m}])
  (γ1 PPara[-w,-p](Symmetrize[
       PPara[-q,x]PPara[-v,y]-(1/3)PPara[-q,-v]PPara[x,y],{-q,-v}])DpV[-x,-y]+
    γ2 PPara[-w,-p]PPara[-q,-v]DpV[x,-x]);
tmp=tmp/.PO3PiActivate;
tmp=tmp/.PADMActivate;
tmp=ToNesterForm[tmp,"Hard"→True,"ToShell"→False];
Print[tmp];
SiCLorentzParaA2pSiCLorentzPerpA2p=tmp;
*)
(*
tmp=Ji[]PA2m[-l,-m,-n,p,q,r]Antisymmetrize[γ1 Eps[-p,-q,-u]DpV[u,-r]+
    γ2 Eps[-p,-q,-u]DpV[-r,u]+
    γ3 Eps[-p,-r,-u]DpV[u,-q]+
    γ4 Eps[-p,-r,-u]DpV[-q,u]+
    γ5 PPara[-p,-r]Eps[-q,-u,-v]TP1p[u,v]+
    γ6 Eps[-p,-q,-u]TP1p[u,-r]+
    γ7 Eps[-p,-r,-u]TP1p[u,-q],{-p,-q}];
tmp=tmp/.PO3PiActivate;
tmp=tmp/.PADMActivate;
tmp=ToNesterForm[tmp,"Hard"→True,"ToShell"→False];
Print[tmp];
SiCLorentzParaA0mSiCLorentzPerpA2m=tmp;
*)
(*
tmp=Ji[]PA2m[-l,-m,-n,p,q,r]Antisymmetrize[γ1 PPara[-p,-i]TP1p[-q,-r]+
    γ2 PPara[-p,-r]TP1p[-q,-i]+
    γ3 PPara[-i,-r]TP1p[-p,-q],{-p,-q}];
tmp=tmp/.PO3PiActivate;
tmp=tmp/.PADMActivate;
tmp=ToNesterForm[tmp,"Hard"→True,"ToShell"→False];
Print[tmp];
SiCLorentzParaA1mSiCLorentzPerpA2m=tmp;
*)
```

```
(*
tmp=Ji[]PA2m[-i,-j,-k,p,q,r]PA2m[-l,-m,-n,u,v,w]
  Antisymmetrize[Antisymmetrize[γ1 PPara[-p,-u]PPara[-v,-q]TP1p[-r,-w]+
     γ2 PPara[-p,-r]PPara[-w,-q]TP1p[-u,-v]+
     γ3 PPara[-r,-u]PPara[-v,-w]TP1p[-p,-q]+
     γ4 PPara[-r,-u]PPara[-v,-p]TP1p[-q,-w]+
     γ5 PPara[-w,-u]PPara[-v,-p]TP1p[-q,-r]+
     γ6 PPara[-r,-p]PPara[-q,-u]TP1p[-v,-w]+
     γ7 PPara[-w,-p]PPara[-q,-u]TP1p[-v,-r]+
     γ8 PPara[-r,-w]PPara[-q,-u]TP1p[-v,-p]+
     γ9 PPara[-p,-u]PPara[-v,-q]TP1p[-r,-w]+
     γ10 PPara[-p,-r]PPara[-w,-q]DpV[-u,-v]+
     γ11 PPara[-r,-u]PPara[-v,-w]DpV[-p,-q]+
     γ12 PPara[-r,-u]PPara[-v,-p]DpV[-q,-w]+
     γ13 PPara[-w,-u]PPara[-v,-p]DpV[-q,-r]+
     γ14 PPara[-r,-p]PPara[-q,-u]DpV[-v,-w]+
     γ15 PPara[-w,-p]PPara[-q,-u]DpV[-v,-r]+
     γ16 PPara[-r,-w]PPara[-q,-u]DpV[-v,-p],{-p,-q}],{-u,-v}];
tmp=tmp/.PO3PiActivate;
tmp=tmp/.PADMActivate;
tmp=ToNesterForm[tmp,"Hard"→True,"ToShell"→False];
Print[tmp];
SiCLorentzParaA2mSiCLorentzPerpA2m=tmp;
*)
(*
tmp=Ji[]PA2m[-i,-j,-k,p,q,r]PA2m[-l,-m,-n,u,v,w]
  Antisymmetrize[Antisymmetrize[γ1 PPara[-p,-u]PPara[-v,-q]PPara[-r,-w]+
     γ2 PPara[-p,-r]PPara[-w,-q]PPara[-u,-v]+
     γ3 PPara[-r,-u]PPara[-v,-w]PPara[-p,-q]+
     γ4 PPara[-r,-u]PPara[-v,-p]PPara[-q,-w]+
     γ5 PPara[-w,-u]PPara[-v,-p]PPara[-q,-r]+
     γ6 PPara[-r,-p]PPara[-q,-u]PPara[-v,-w]+
     γ7 PPara[-w,-p]PPara[-q,-u]PPara[-v,-r]+
     γ8 PPara[-r,-w]PPara[-q,-u]PPara[-v,-p]+
     γ9 PPara[-p,-u]PPara[-v,-q]PPara[-r,-w],{-p,-q}],{-u,-v}];
tmp=tmp/.PO3PiActivate;
tmp=tmp/.PADMActivate;
tmp=ToNesterForm[tmp,"Hard"→True,"ToShell"→False];
Print[tmp];
SiCLorentzParaB2mSiCLorentzPerpA2m=tmp;
*)
```

Commutators

```
In[]:= (*PoissonBracket[SiCLorentzParaA0p[],
         SiCLorentzPerpA0p[],"Hard"→True,"ToShell"→False];*)
       (*PoissonBracket[SiCLorentzParaA0m[],SiCLorentzPerpA0p[],
         "Hard"→True,"ToShell"→False];*)
       (*PoissonBracket[SiCLorentzParaA1p[-i,-j],SiCLorentzPerpA0p[],
         "Hard"→True,"ToShell"→False];*)
       (*PoissonBracket[SiCLorentzParaA1m[-i],SiCLorentzPerpA0p[],
         "Hard"→True,"ToShell"→False];*)(*SURFICIAL*)
       (*PoissonBracket[SiCLorentzParaA2p[-i,-j],SiCLorentzPerpA0p[],
         "Hard"→True,"ToShell"→False];*)
       (*PoissonBracket[SiCLorentzParaA2m[-i,-j,-k],
         SiCLorentzPerpA0p[],"Hard"→True,"ToShell"→False];*)

       (*PoissonBracket[SiCLorentzParaA0p[],
         SiCLorentzPerpA0m[],"Hard"→True,"ToShell"→False];*)
       (*PoissonBracket[SiCLorentzParaA0m[],SiCLorentzPerpA0m[],
         "Hard"→True,"ToShell"→False];*)
       (*PoissonBracket[SiCLorentzParaA1p[-i,-j],SiCLorentzPerpA0m[],
         "Hard"→True,"ToShell"→False];*)(*SURFICIAL*)
       (*PoissonBracket[SiCLorentzParaA1m[-i],SiCLorentzPerpA0m[],
         "Hard"→True,"ToShell"→False];*)
       (*PoissonBracket[SiCLorentzParaA2p[-i,-j],SiCLorentzPerpA0m[],
         "Hard"→True,"ToShell"→False];*)
       (*
       tmp=PoissonBracket[SiCLorentzParaA2m[-i,-j,-k],
          SiCLorentzPerpA0m[],"Hard"→True,"ToShell"→False][[1]];
       Print[tmp];
       tmp=tmp-SiCLorentzParaA2mSiCLorentzPerpA0m//CollectTensors;
       Print[tmp];
       eqs=ToConstantSymbolEquations[tmp==0];
       Print[eqs];
       sol=Solve[eqs];
       Print[sol];
       *)

       (*PoissonBracket[SiCLorentzParaA0p[],
         SiCLorentzPerpA1p[-l,-m],"Hard"→True,"ToShell"→False];*)
       (*PoissonBracket[SiCLorentzParaA0m[],SiCLorentzPerpA1p[-l,-m],
         "Hard"→True,"ToShell"→False];*)(*FAILED*)
       (*PoissonBracket[SiCLorentzParaA1p[-i,-j],
         SiCLorentzPerpA1p[-l,-m],"Hard"→True,"ToShell"→False];*)
       (*PoissonBracket[SiCLorentzParaA1m[-i],SiCLorentzPerpA1p[-l,-m],
         "Hard"→True,"ToShell"→False];*)(*FAILED*)
```

```
(*PoissonBracket[SiCLorentzParaA2p[-i,-j],
  SiCLorentzPerpA1p[-l,-m],"Hard"→True,"ToShell"→False];*)
(*PoissonBracket[SiCLorentzParaA2m[-i,-j,-k],SiCLorentzPerpA1p[-l,-m],
  "Hard"→True,"ToShell"→False];*)(*FAILED*)

(*PoissonBracket[SiCLorentzParaA0p[],
  SiCLorentzPerpA1m[-l],"Hard"→True,"ToShell"→False];*)(*FAILED*)
(*PoissonBracket[SiCLorentzParaA0m[],SiCLorentzPerpA1m[-l],
  "Hard"→True,"ToShell"→False];*)
(*PoissonBracket[SiCLorentzParaA1p[-i,-j],SiCLorentzPerpA1m[-l],
  "Hard"→True,"ToShell"→False];*)(*FAILED*)
(*PoissonBracket[SiCLorentzParaA1m[-i],SiCLorentzPerpA1m[-l],
  "Hard"→True,"ToShell"→False];*)
(*PoissonBracket[SiCLorentzParaA2p[-i,-j],SiCLorentzPerpA1m[-l],
  "Hard"→True,"ToShell"→False];*)(*FAILED*)
(*
tmp=PoissonBracket[SiCLorentzParaA2m[-i,-j,-k],
   SiCLorentzPerpA1m[-l],"Hard"→True,"ToShell"→False][[1]];
Print[tmp];
tmp=tmp-SiCLorentzParaA2mSiCLorentzPerpA1m//CollectTensors;
Print[tmp];
eqs=ToConstantSymbolEquations[tmp==0];
Print[eqs];
sol=Solve[eqs];
Print[sol];
*)

(*PoissonBracket[SiCLorentzParaA0p[],
  SiCLorentzPerpA2p[-l,-m],"Hard"→True,"ToShell"→False];*)
(*PoissonBracket[SiCLorentzParaA0m[],SiCLorentzPerpA2p[-l,-m],
  "Hard"→True,"ToShell"→False];*)
(*PoissonBracket[SiCLorentzParaA1p[-i,-j],
  SiCLorentzPerpA2p[-l,-m],"Hard"→True,"ToShell"→False];*)
(*PoissonBracket[SiCLorentzParaA1m[-i],SiCLorentzPerpA2p[-l,-m],
  "Hard"→True,"ToShell"→False];*)(*FAILED*)
(*
tmp=PoissonBracket[SiCLorentzParaA2p[-i,-j],
   SiCLorentzPerpA2p[-l,-m],"Hard"→True,"ToShell"→False][[1]];
Print[tmp];
tmp=tmp-SiCLorentzParaA2pSiCLorentzPerpA2p//CollectTensors;
Print[tmp];
eqs=ToConstantSymbolEquations[tmp==0];
Print[eqs];
sol=Solve[eqs];
```

```
Print[sol];
*)
(*PoissonBracket[SiCLorentzParaA2m[-i,-j,-k],
  SiCLorentzPerpA2p[-l,-m],"Hard"→True,"ToShell"→False];*)(*FAILED*)

(*PoissonBracket[SiCLorentzParaA0p[],
  SiCLorentzPerpA2m[-l,-m,-n],"Hard"→True,"ToShell"→False];*)
(*
tmp=PoissonBracket[SiCLorentzParaA0m[],
   SiCLorentzPerpA2m[-l,-m,-n],"Hard"→True,"ToShell"→False][[1]];
Print[tmp];
tmp=tmp-SiCLorentzParaA0mSiCLorentzPerpA2m//CollectTensors;
Print[tmp];
eqs=ToConstantSymbolEquations[tmp==0];
Print[eqs];
sol=Solve[eqs];
Print[sol];
*)
(*PoissonBracket[SiCLorentzParaA1p[-i,-j],
  SiCLorentzPerpA2m[-l,-m,-n],"Hard"→True,"ToShell"→False];*)(*FAILED*)
(*
tmp=PoissonBracket[SiCLorentzParaA1m[-i],
   SiCLorentzPerpA2m[-l,-m,-n],"Hard"→True,"ToShell"→False][[1]];
Print[tmp];
tmp=tmp-SiCLorentzParaA1mSiCLorentzPerpA2m//CollectTensors;
Print[tmp];
eqs=ToConstantSymbolEquations[tmp==0];
Print[eqs];
sol=Solve[eqs];
Print[sol];
*)
(*PoissonBracket[SiCLorentzParaA2p[-i,-j],
  SiCLorentzPerpA2m[-l,-m,-n],"Hard"→True,"ToShell"→False];*)(*FAILED*)
(*
tmp=PoissonBracket[SiCLorentzParaA2m[-i,-j,-k],
   SiCLorentzPerpA2m[-l,-m,-n],"Hard"→True,"ToShell"→False][[1]];
Print[tmp];
tmp=tmp-SiCLorentzParaA2mSiCLorentzPerpA2m//CollectTensors;
Print[tmp];
eqs=ToConstantSymbolEquations[tmp==0];
Print[eqs];
sol=Solve[eqs];
Print[sol];
*)
```

```
(*PoissonBracket[SiCLorentzParaB2m[-i,-j,-k],
  SiCLorentzPerpB1p[-l,-m],"Hard"→True,"ToShell"→False];*)(*FAILED*)
(*PoissonBracket[SiCLorentzParaB2m[-i,-j,-k],SiCLorentzPerpB2p[-l,-m],
  "Hard"→True,"ToShell"→False];*)(*FAILED*)
(*PoissonBracket[SiCLorentzParaB2m[-i,-j,-k],
  SiCLorentzPerpA0p[],"Hard"→True,"ToShell"→False];*)
(*
tmp=PoissonBracket[SiCLorentzParaB2m[-i,-j,-k],
   SiCLorentzPerpA2m[-l,-m,-n],"Hard"→True,"ToShell"→False][[1]];
Print[tmp];
tmp=tmp-SiCLorentzParaB2mSiCLorentzPerpA2m//CollectTensors;
Print[tmp];
eqs=ToConstantSymbolEquations[tmp==0];
Print[eqs];
sol=Solve[eqs];
Print[sol];
*)
```

$\chi^{\perp}$-$\chi^{\perp}$

Commutator ansätze

Commutators

In[∘]:=
```
(*PoissonBracket[SiCLorentzPerpB1p[-i,-j],
  SiCLorentzPerpB1m[-l],"Hard"→True,"ToShell"→False];*)
(*PoissonBracket[SiCLorentzPerpB2p[-i,-j],
  SiCLorentzPerpB1m[-l],"Hard"→True,"ToShell"→False];*)
```

Simple spin – 1^+ case

Commutator ansätze

In[∘]:=
```
DefNiceConstantSymbol[γ, #] & /@ Range[20];
(*
tmp=PA2m[-l,-m,-n,p,q,r]
  (Symmetrize[PPara[-i,x]PPara[-j,y]-(1/3)PPara[-i,-j]PPara[x,y],{-i,-j}])
  Antisymmetrize[
   PPara[-r,-p]PPara[-q,-x]γ4 Ji[]^2 PiPA1m[-y]+
    PPara[-r,-x]PPara[-y,-p]γ5 Ji[]^2 PiPA1m[-q]+
    PPara[-p,-x]PPara[-y,-q]γ6 Ji[]^2 PiPA1m[-r],{-p,-q}];
tmp=tmp/.PO3PiActivate;
tmp=tmp/.PADMActivate;
tmp=ToNesterForm[tmp,"Hard"→True,"ToShell"→False];
Print[tmp];
PhiB2pPhiA2m=tmp;
*)
```

Commutators

In[∘]:=
```
(*PoissonBracket[PhiB0p[],PhiB0p[],"ToShell"→True,"Hard"→False,"Surficial"→True];
(**)
(**)PoissonBracket[PhiB0p[],PhiB1m[-l],
 "ToShell"→True,"Hard"→False,"Surficial"→True];
(**)
(**)PoissonBracket[PhiB0p[],PhiB2p[-l,-m],
 "ToShell"→True,"Hard"→False,"Surficial"→True];
(**)
(**)PoissonBracket[PhiB0p[],PhiA0p[],
 "ToShell"→True,"Hard"→False,"Surficial"→True];
(**)
(**)PoissonBracket[PhiB0p[],PhiA0m[],
 "ToShell"→True,"Hard"→False,"Surficial"→True];
(**)
(**)PoissonBracket[PhiB0p[],PhiA2p[-l,-m],
 "ToShell"→True,"Hard"→False,"Surficial"→True];
(**)
(**)PoissonBracket[PhiB0p[],PhiA2m[-l,-m,-n],
 "ToShell"→True,"Hard"→False,"Surficial"→True];*)

(*PoissonBracket[PhiB1m[-i],PhiB1m[-l],
 "ToShell"→True,"Hard"→False,"Surficial"→True];
```

```
(**)
(**)PoissonBracket[PhiB1m[-i],PhiB2p[-l,-m],
 "ToShell"→True,"Hard"→False,"Surficial"→True];
(**)
(**)PoissonBracket[PhiB1m[-i],PhiA0p[],
 "ToShell"→True,"Hard"→False,"Surficial"→True];
(**)
(**)PoissonBracket[PhiB1m[-i],PhiA0m[],
 "ToShell"→True,"Hard"→False,"Surficial"→True];
(**)
(**)PoissonBracket[PhiB1m[-i],PhiA2p[-l,-m],
 "ToShell"→True,"Hard"→False,"Surficial"→True];
(**)
(**)PoissonBracket[PhiB1m[-i],PhiA2m[-l,-m,-n],
 "ToShell"→True,"Hard"→False,"Surficial"→True];*)

(*PoissonBracket[PhiB2p[-i,-j],PhiB2p[-l,-m],
 "ToShell"→True,"Hard"→False,"Surficial"→True];
(**)
(**)PoissonBracket[PhiB2p[-i,-j],PhiA0p[],
 "ToShell"→True,"Hard"→False,"Surficial"→True];
(**)
(**)PoissonBracket[PhiB2p[-i,-j],PhiA0m[],
 "ToShell"→True,"Hard"→False,"Surficial"→True];
(**)
(**)PoissonBracket[PhiB2p[-i,-j],PhiA2p[-l,-m],
 "ToShell"→True,"Hard"→False,"Surficial"→True];*)
(*
tmp=PoissonBracket[PhiB2p[-i,-j],PhiA2m[-l,-m,-n],
  "ToShell"→True,"Hard"→False,"Surficial"→True];
Print[tmp];
tmp2=tmp[[1]]-PhiB2pPhiA2m//CollectTensors;
Print[tmp2];
eqs=ToConstantSymbolEquations[tmp2==0];
Print[eqs];
sol=Solve[eqs];
Print[sol];
*)

(*PoissonBracket[PhiA0p[],PhiA0p[],"ToShell"→True,"Hard"→False,"Surficial"→True];
(**)
(**)PoissonBracket[PhiA0p[],PhiA0m[],
 "ToShell"→True,"Hard"→False,"Surficial"→True];
(**)
```

```
(**)PoissonBracket[PhiA0p[],PhiA2p[-l,-m],
 "ToShell"→True,"Hard"→False,"Surficial"→True];
(**)
(**)PoissonBracket[PhiA0p[],PhiA2m[-l,-m,-n],
 "ToShell"→True,"Hard"→False,"Surficial"→True];
(**)

(**)PoissonBracket[PhiA0m[],PhiA0m[],
 "ToShell"→True,"Hard"→False,"Surficial"→True];
(**)
(**)PoissonBracket[PhiA0m[],PhiA2p[-l,-m],
 "ToShell"→True,"Hard"→False,"Surficial"→True];
(**)
(**)PoissonBracket[PhiA0m[],PhiA2m[-l,-m,-n],
 "ToShell"→True,"Hard"→False,"Surficial"→True];
(**)

(**)PoissonBracket[PhiA2p[-i,-j],PhiA2p[-l,-m],
 "ToShell"→True,"Hard"→False,"Surficial"→True];
(**)
(**)PoissonBracket[PhiA2p[-i,-j],PhiA2m[-l,-m,-n],
 "ToShell"→True,"Hard"→False,"Surficial"→True];
(**)

(**)PoissonBracket[PhiA2m[-i,-j,-k],PhiA2m[-l,-m,-n],
 "ToShell"→True,"Hard"→False,"Surficial"→True];*)

(*Quit[];*)
```

Case 16.1.2

Commutator ansätze

In[∘]:=
```
(*
tmp=Ji[]PA2m[-i,-j,-k,p,q,r]Antisymmetrize[γ1 Ji[]PPara[-p,-l]PiPA1p[-q,-r]+
    γ2 Ji[]PPara[-p,-r]PiPA1p[-q,-l]+
    γ3 Ji[] PPara[-l,-r]PiPA1p[-p,-q]+
    γ4 Ji[]PPara[-p,-l]PiPA2p[-q,-r]+
    γ5 Ji[]PPara[-p,-r]PiPA2p[-q,-l]+
    γ6 PPara[-p,-l]RP1p[-q,-r]+
    γ7 PPara[-p,-r]RP1p[-q,-l]+
    γ8 PPara[-l,-r]RP1p[-p,-q]+
    γ9 PPara[-p,-l]RP2p[-q,-r]+
```

```
    γ10 PPara[-p,-r]RP2p[-q,-l],{-p,-q}];
tmp=tmp/.PO3PiActivate;
tmp=tmp/.PADMActivate;
tmp=ToNesterForm[tmp,"Hard"→True,"ToShell"→False];
Print[tmp];
PhiA2mChiSingB1m=tmp;
*)
(*
tmp=Ji[]PA2m[-i,-j,-k,p,q,r]Antisymmetrize[γ1 PPara[-p,-l]TP1p[-q,-r]+
    γ2 PPara[-p,-r]TP1p[-q,-l]+
    γ3  PPara[-l,-r]TP1p[-p,-q]+
    γ4 PPara[-p,-l]DpV[-q,-r]+
    γ5 PPara[-p,-r]DpV[-q,-l]+
    γ6 PPara[-l,-r]DpV[-p,-q]+
    γ7 PPara[-p,-r]PPara[-q,-l]DpV[-x,x],{-p,-q}];
tmp=tmp/.PO3PiActivate;
tmp=tmp/.PADMActivate;
tmp=ToNesterForm[tmp,"Hard"→True,"ToShell"→False];
Print[tmp];
ChiParaB2mChiSingB1m=tmp;
*)
(*
tmp=Ji[]PA2m[-i,-j,-k,p,q,r]PA2m[-l,-m,-n,u,v,w]
  Antisymmetrize[Antisymmetrize[γ1 PPara[-p,-u]PPara[-v,-q]PPara[-r,-w]+
     γ2 PPara[-p,-r]PPara[-w,-q]PPara[-u,-v]+
     γ3 PPara[-r,-u]PPara[-v,-w]PPara[-p,-q]+
     γ4 PPara[-r,-u]PPara[-v,-p]PPara[-q,-w]+
     γ5 PPara[-w,-u]PPara[-v,-p]PPara[-q,-r]+
     γ6 PPara[-r,-p]PPara[-q,-u]PPara[-v,-w]+
     γ7 PPara[-w,-p]PPara[-q,-u]PPara[-v,-r]+
     γ8 PPara[-r,-w]PPara[-q,-u]PPara[-v,-p]+
     γ9 PPara[-p,-u]PPara[-v,-q]PPara[-r,-w],{-p,-q}],{-u,-v}];
tmp=tmp/.PO3PiActivate;
tmp=tmp/.PADMActivate;
tmp=ToNesterForm[tmp,"Hard"→True,"ToShell"→False];
Print[tmp];
PhiA2mChiParaB2m=tmp;
*)
(*
tmp=PA2m[-l,-m,-n,p,q,r]Antisymmetrize[Antisymmetrize[
     PPara[-r,-p]PPara[-q,-i]γ1 Ji[]Ji[] DpJ[-j]+
      (*PPara[-r,-i]PPara[-j,-p]γ2 Ji[]Ji[] DpJ[-q]+*)
      PPara[-p,-i]PPara[-j,-q]γ3 Ji[]Ji[] DpJ[-r]+
      PPara[-r,-p]PPara[-q,-i]γ4 Ji[] TP1m[-j]+
```

```
     (*PPara[-r,-i]PPara[-j,-p]γ5 Ji[] TP1m[-q]+*)
     PPara[-p,-i]PPara[-j,-q]γ6 Ji[] TP1m[-r]+
     PPara[-p,-i]γ7 Ji[] DpV[-j,-q]V[-r]+
     PPara[-p,-i]γ8 Ji[] DpV[-q,-j]V[-r]+
     PPara[-p,-i]γ9 Ji[] DpV[-j,-r]V[-q]+
     PPara[-p,-i]γ10 Ji[] DpV[-r,-j]V[-q]+
     (*PPara[-p,-i]γ11 Ji[] DpV[-r,-q]V[-j]+*)
     PPara[-p,-i]γ12 Ji[] DpV[-q,-r]V[-j]+
     (*PPara[-r,-i]γ13 Ji[] DpV[-p,-q]V[-j]+*)
     PPara[-r,-i]γ14 Ji[] DpV[-p,-j]V[-q]+
     PPara[-r,-i]γ15 Ji[] DpV[-j,-p]V[-q]+
     PPara[-r,-p]γ16 Ji[] DpV[-i,-j]V[-q]
    (*PPara[-r,-p]γ17 Ji[] DpV[-i,-q]V[-j]+*)
    (*PPara[-r,-p]γ18 Ji[] DpV[-q,-i]V[-j]+*)
    (*PPara[-p,-i]γ19 Ji[] TP1p[-j,-q]V[-r]+*)
    (*PPara[-p,-i]γ20 Ji[] TP1p[-j,-r]V[-q]+*)
    (*PPara[-p,-i]γ21 Ji[] TP1p[-r,-q]V[-j]+*)
    (*PPara[-r,-i]γ17 Ji[] TP1p[-p,-q]V[-j]+*)
    (*PPara[-r,-i]γ18 Ji[] TP1p[-p,-j]V[-q]+*)
    (*PPara[-r,-p]γ19 Ji[]TP1p[-i,-j]V[-q]+*)
    (*PPara[-r,-p]γ20 Ji[] TP1p[-i,-q]V[-j]*),{-i,-j}],{-p,-q}]+
Antisymmetrize[Antisymmetrize[
  (*PPara[-l,-i]γ21 Ji[] DpV[-j,-m]V[-n]+*)
  PPara[-l,-i]γ22 Ji[] DpV[-m,-j]V[-n]+
    (*PPara[-l,-i]γ23 Ji[] DpV[-j,-n]V[-m]+*)
   PPara[-l,-i]γ24 Ji[] DpV[-n,-j]V[-m]+
    (*PPara[-l,-i]γ25 Ji[] DpV[-n,-m]V[-j]+*)
   PPara[-l,-i]γ26 Ji[] DpV[-m,-n]V[-j]+
    (*PPara[-n,-i]γ27 Ji[] DpV[-l,-m]V[-j]+*)
    (*PPara[-n,-i]γ28 Ji[] DpV[-l,-j]V[-m]+*)
   PPara[-n,-i]γ29 Ji[] DpV[-j,-l]V[-m]+
    (*PPara[-n,-l]γ30 Ji[] DpV[-i,-j]V[-m]+*)
    (*PPara[-n,-l]γ31 Ji[] DpV[-i,-m]V[-j]+*)
   PPara[-n,-l]γ32 Ji[] DpV[-m,-i]V[-j]+
    (*PPara[-m,-i]PPara[-j,-n]γ40 Ji[] DpV[-x,x]V[-l]+*)
   PPara[-l,-i]PPara[-j,-n]γ41 Ji[] DpV[-x,x]V[-m]+
   PPara[-m,-i]PPara[-j,-l]γ42 Ji[] DpV[-x,x]V[-n]+
   PPara[-l,-i]γ33 Ji[] TP1p[-j,-m]V[-n]+
   PPara[-l,-i]γ34 Ji[] TP1p[-j,-n]V[-m]+
   PPara[-l,-i]γ35 Ji[] TP1p[-n,-m]V[-j]+
   PPara[-n,-i]γ36 Ji[] TP1p[-l,-m]V[-j]+
   PPara[-n,-i]γ37 Ji[] TP1p[-l,-j]V[-m]+
   PPara[-n,-l]γ38 Ji[]TP1p[-i,-j]V[-m]+
   PPara[-n,-l]γ39 Ji[] TP1p[-i,-m]V[-j],{-i,-j}],{-l,-m}];
```

```
tmp=tmp/.PO3PiActivate;
tmp=tmp/.PADMActivate;
tmp=ToNesterForm[tmp,"Hard"→True,"ToShell"→False];
Print[tmp];
PhiB1pChiParaB2m=tmp;
*)
(*
tmp=PA2m[-l,-m,-n,p,q,r]Antisymmetrize[Antisymmetrize[
    PPara[-r,-p]PPara[-q,-i]γ4 Ji[] DpHComp[-j]+
     PPara[-r,-i]PPara[-j,-p]γ5 Ji[] DpHComp[-q]+
     PPara[-p,-i]PPara[-j,-q]γ6 Ji[] DpHComp[-r],{-i,-j}],{-p,-q}];
tmp=tmp/.PO3PiActivate;
tmp=tmp/.PADMActivate;
tmp=ToNesterForm[tmp,"Hard"→True,"ToShell"→False];
Print[tmp];
PhiB1pChiParaB2mPart2=tmp;
*)
(*
tmp=PA2m[-l,-m,-n,p,q,r]
   (Symmetrize[PPara[-i,x]PPara[-j,y]-(1/3)PPara[-i,-j]PPara[x,y],{-i,-j}])
   Antisymmetrize[
    PPara[-r,-p]PPara[-q,-x]γ1 Ji[]Ji[] DpJ[-y]+
     PPara[-r,-x]PPara[-y,-p]γ2 Ji[]Ji[] DpJ[-q]+
     PPara[-p,-x]PPara[-y,-q]γ3 Ji[]Ji[] DpJ[-r]+
     PPara[-r,-p]PPara[-q,-x]γ4 Ji[] TP1m[-y]+
     PPara[-r,-x]PPara[-y,-p]γ5 Ji[] TP1m[-q]+
     PPara[-p,-x]PPara[-y,-q]γ6 Ji[] TP1m[-r]+
     PPara[-p,-x]γ7 Ji[] DpV[-y,-q]V[-r]+
     PPara[-p,-x]γ8 Ji[] DpV[-q,-y]V[-r]+
     PPara[-p,-x]γ9 Ji[] DpV[-y,-r]V[-q]+
     PPara[-p,-x]γ10 Ji[] DpV[-r,-y]V[-q]+
     PPara[-p,-x]γ11 Ji[] DpV[-r,-q]V[-y]+
     PPara[-p,-x]γ12 Ji[] DpV[-q,-r]V[-y]+
     PPara[-r,-x]γ13 Ji[] DpV[-p,-q]V[-y]+
     PPara[-r,-x]γ14 Ji[] DpV[-p,-y]V[-q]+
     PPara[-r,-x]γ15 Ji[] DpV[-y,-p]V[-q]+
     PPara[-r,-p]γ16 Ji[] DpV[-x,-y]V[-q]+
     PPara[-r,-p]γ17 Ji[] DpV[-x,-q]V[-y]+
     PPara[-r,-p]γ18 Ji[] DpV[-q,-x]V[-y]+
     PPara[-p,-x]γ19 Ji[] TP1p[-y,-q]V[-r]+
     PPara[-p,-x]γ20 Ji[] TP1p[-y,-r]V[-q]+
     PPara[-p,-x]γ21 Ji[] TP1p[-r,-q]V[-y]+
     PPara[-r,-x]γ17 Ji[] TP1p[-p,-q]V[-y]+
     PPara[-r,-x]γ18 Ji[] TP1p[-p,-y]V[-q]+
```

```
      PPara[-r,-p]γ19 Ji[]TP1p[-x,-y]V[-q]+
      PPara[-r,-p]γ20 Ji[] TP1p[-x,-q]V[-y]+
      Eps[-p,-q,-x]γ21 PPara[-y,-r]Ji[]TP0m[]+
      Eps[-p,-r,-x]γ22 PPara[-y,-q]Ji[]TP0m[],{-p,-q}]+
   (Symmetrize[PPara[-i,x]PPara[-j,y]-(1/3)PPara[-i,-j]PPara[x,y],{-i,-j}])
    Antisymmetrize[
     PPara[-l,-x]γ23 Ji[] DpV[-y,-m]V[-n]+
      PPara[-l,-x]γ24 Ji[] DpV[-m,-y]V[-n]+
      PPara[-l,-x]γ25 Ji[] DpV[-y,-n]V[-m]+
      PPara[-l,-x]γ26 Ji[] DpV[-n,-y]V[-m]+
      PPara[-l,-x]γ27 Ji[] DpV[-n,-m]V[-y]+
      PPara[-l,-x]γ28 Ji[] DpV[-m,-n]V[-y]+
      PPara[-n,-x]γ29 Ji[] DpV[-l,-m]V[-y]+
      PPara[-n,-x]γ30 Ji[] DpV[-l,-y]V[-m]+
      PPara[-n,-x]γ31 Ji[] DpV[-y,-l]V[-m]+
      PPara[-n,-l]γ32 Ji[] DpV[-x,-y]V[-m]+
      PPara[-n,-l]γ33 Ji[] DpV[-x,-m]V[-y]+
      PPara[-n,-l]γ34 Ji[] DpV[-m,-x]V[-y]+
      PPara[-m,-x]PPara[-y,-n]γ35 Ji[] DpV[-w,w]V[-l]+
      PPara[-l,-x]PPara[-y,-n]γ36 Ji[] DpV[-w,w]V[-m]+
      PPara[-m,-x]PPara[-y,-l]γ37 Ji[] DpV[-w,w]V[-n]+
      PPara[-l,-x]γ38 Ji[] TP1p[-y,-m]V[-n]+
      PPara[-l,-x]γ39 Ji[] TP1p[-y,-n]V[-m]+
      PPara[-l,-x]γ40 Ji[] TP1p[-n,-m]V[-y]+
      PPara[-n,-x]γ41 Ji[] TP1p[-l,-m]V[-y]+
      PPara[-n,-x]γ42 Ji[] TP1p[-l,-y]V[-m]+
      PPara[-n,-l]γ43 Ji[]TP1p[-x,-y]V[-m]+
      PPara[-n,-l]γ44 Ji[] TP1p[-x,-m]V[-y],{-l,-m}];
tmp=tmp/.PO3PiActivate;
tmp=tmp/.PADMActivate;
tmp=ToNesterForm[tmp,"Hard"→True,"ToShell"→False];
Print[tmp];
PhiB2pChiParaB2m=tmp;
*)
(*
tmp=PA2m[-l,-m,-n,p,q,r]
   (Symmetrize[PPara[-i,x]PPara[-j,y]-(1/3)PPara[-i,-j]PPara[x,y],{-i,-j}])
   Antisymmetrize[
    PPara[-r,-p]PPara[-q,-x]γ4 Ji[] DpHComp[-y]+
     PPara[-r,-x]PPara[-y,-p]γ5 Ji[] DpHComp[-q]+
     PPara[-p,-x]PPara[-y,-q]γ6 Ji[] DpHComp[-r],{-p,-q}];
tmp=tmp/.PO3PiActivate;
tmp=tmp/.PADMActivate;
```

```
tmp=ToNesterForm[tmp,"Hard"→True,"ToShell"→False];
Print[tmp];
PhiB2pChiParaB2mPart2=tmp;
*)
(*
tmp=PA2m[-l,-m,-n,-i,-j,k] DpHComp[-k];
tmp=tmp/.PO3PiActivate;
tmp=tmp/.PADMActivate;
tmp=ToNesterForm[tmp,"Hard"→True,"ToShell"→False];
Print[tmp];
DefTensor[QQ[-i,-j,-l],M4,Antisymmetric[{-i,-j}],OrthogonalTo→{V[i],V[j],V[l]}];
Print["contracting"];
tmp1=MakeContractionAnsatz[PPara[-a,-b]QQ[-c,-d,-e],
  IndexList[-l,-m,-n,-i,-j],{Antisymmetric[{-l,-m}],Antisymmetric[{-i,-j}]}];
Print["done con"];
QQActivate=
 MakeRule[{QQ[-i,-j,-l],PPara[-l,-i]DpHComp[-j]-PPara[-l,-j]DpHComp[-i]},
  MetricOn→All,ContractMetrics→True];
Print["acting"];
tmp1=tmp1/.QQActivate//ToNewCanonical;
tmp1=tmp1/.PADMActivate//ToNewCanonical;
tmp1=ToNesterForm[tmp1,"Hard"→True,"ToShell"→False];
Print[tmp1];

tmp2=tmp-tmp1//CollectTensors;
Print[tmp2];
eqs=ToConstantSymbolEquations[tmp2==0];
Print[eqs];
sol=Solve[eqs];
Print[sol];
Quit[];
*)
```

Commutators

```
(*PoissonBracket[PhiB1p[-i,-j],PhiB1p[-l,-m],
  "ToShell"→True,"Hard"→False,"Surficial"→False];*)
(*PoissonBracket[PhiB1p[-i,-j],PhiB2p[-l,-m]];*)
(*PoissonBracket[PhiB2p[-i,-j],PhiB2p[-l,-m]];*)
(*
EvalPhiB1pChiParaB2m=PoissonBracket[PhiB1p[-i,-j],
  ChiParaB2m[-l,-m,-n],"ToShell"→True,"Hard"→False,"Surficial"→True];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"EvalPhiB1pChiParaB2m"<>".mx",
 {EvalPhiB1pChiParaB2m}];
```

```
Print["finished printing"];
Quit[];
*)
(*
MyImport["ChiB1mSimple1p.mx"];
Print[tmp];
tmp2=tmp[[1]]-PhiB1pChiParaB2m//CollectTensors;
Print[tmp2];
eqs=ToConstantSymbolEquations[tmp2==0];
Print[eqs];
sol=Solve[eqs];
Print[sol];
tmp3=tmp[[2]]-PhiB1pChiParaB2mPart2//CollectTensors;
Print[tmp3];
eqs=ToConstantSymbolEquations[tmp3==0];
Print[eqs];
sol=Solve[eqs];
Print[sol];
*)
(*
tmp=PoissonBracket[PhiB2p[-i,-j],ChiParaB2m[-l,-m,-n],
  "ToShell"→True,"Hard"→False,"Surficial"→True];
Print[tmp];
tmp2=tmp[[1]]-PhiB2pChiParaB2m//CollectTensors;
Print[tmp2];
eqs=ToConstantSymbolEquations[tmp2==0];
Print[eqs];
sol=Solve[eqs];
Print[sol];
tmp3=tmp[[2]]-PhiB2pChiParaB2mPart2//CollectTensors;
Print[tmp3];
eqs=ToConstantSymbolEquations[tmp3==0];
Print[eqs];
sol=Solve[eqs];
Print[sol];
*)
(*EvalPhiB1pChiSingB1m=PoissonBracket[PhiB1p[-i,-j],
  ChiSingB1m[-l],"ToShell"→True,"Hard"→False,"Surficial"→True];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"EvalPhiB1pChiSingB1m"<>".mx",
 {EvalPhiB1pChiSingB1m}];
Print["finished printing"];
MyImport["EvalPhiB1pChiSingB1m.mx"];
*)
(*PoissonBracket[PhiB2p[-i,-j],ChiSingB1m[-l],
```

```
  "ToShell"→False,"Hard"→False,"Surficial"→True];*)
(*PoissonBracket[PhiA0p[],ChiSingB1m[-l]];*)
(*
tmp=PoissonBracket[PhiA2m[-i,-j,-k],ChiParaB2m[-l,-m,-n],
   "Hard"→False,"ToShell"→False,"Surficial"→True][[1]];
Print[tmp];
tmp=tmp-PhiA2mChiParaB2m//CollectTensors;
Print[tmp];
eqs=ToConstantSymbolEquations[tmp==0];
Print[eqs];
sol=Solve[eqs];
Print[sol];
*)
(*
tmp=PoissonBracket[PhiA2m[-i,-j,-k],
   ChiSingB1m[-l],"Hard"→False,"ToShell"→True][[1]];
Print[tmp];
tmp=tmp-PhiA2mChiSingB1m//CollectTensors;
Print[tmp];
eqs=ToConstantSymbolEquations[tmp==0];
Print[eqs];
sol=Solve[eqs];
Print[sol];
*)
(*
tmp=PoissonBracket[ChiParaB2m[-i,-j,-k],
   ChiSingB1m[-l],"Hard"→False,"ToShell"→True][[1]];
Print[tmp];
tmp=tmp-ChiParaB2mChiSingB1m//CollectTensors;
Print[tmp];
eqs=ToConstantSymbolEquations[tmp==0];
Print[eqs];
sol=Solve[eqs];
Print[sol];
*)
(*PoissonBracket[ChiSingB1m[-i],ChiSingB1m[-l],"Hard"→False,"ToShell"→True];*)
(*PoissonBracket[PhiB1p[-i,-j],PhiA0p[],"Hard"→False,"ToShell"→True];*)
(*PoissonBracket[PhiB2p[-i,-j],PhiA0p[],"Hard"→False,"ToShell"→True];*)
(*PoissonBracket[PhiB2p[-i,-j],PhiA2m[-l,-m,-n],"Hard"→False,"ToShell"→True];*)
(*PoissonBracket[PhiB1p[-i,-j],PhiA2m[-l,-m,-n],"Hard"→False,"ToShell"→True];*)
(*Quit[];*)
```

Generalised velocity $\dot{\psi}$

Inert commutators

In[∘]:=
```
(*copies of all the field strength tensors*)
DefTensor[RD[a, b, -d, -e, -x, -y, -z], M4,
  {Antisymmetric[{a, b}], Antisymmetric[{-d, -e}]}, PrintAs → "{ψ,R}δδ"];
DefTensor[RDS1[a, b, -d, -e, -x, -y, -z, v], M4,
  {Antisymmetric[{a, b}], Antisymmetric[{-d, -e}]}, PrintAs → "{ψ,R}∂δδ"];
DefTensor[RDS2[a, b, -d, -e, -x, -y, -z, v], M4,
  {Antisymmetric[{a, b}], Antisymmetric[{-d, -e}]}, PrintAs → "{ψ,R}δ∂δ"];
DefTensor[RDS3[a, b, -d, -e, -x, -y, -z, v, w], M4,
  {Antisymmetric[{a, b}], Antisymmetric[{-d, -e}]}, PrintAs → "{ψ,R}∂δ∂δ"];

DefTensor[TD[a, -b, -c, -x, -y, -z],
  M4, Antisymmetric[{-b, -c}], PrintAs → "{ψ,T}δδ"];
DefTensor[TDS1[a, -b, -c, -x, -y, -z, v], M4,
  Antisymmetric[{-b, -c}], PrintAs → "{ψ,T}∂δδ"];
DefTensor[TDS2[a, -b, -c, -x, -y, -z, v], M4,
  Antisymmetric[{-b, -c}], PrintAs → "{ψ,T}δ∂δ"];
DefTensor[TDS3[a, -b, -c, -x, -y, -z, v, w], M4,
  Antisymmetric[{-b, -c}], PrintAs → "{ψ,T}∂δ∂δ"];

(*copies of all the constraint functions*)
DefTensor[PhiDB0p[-x, -y, -z], M4, PrintAs → "{ψ,ϕb0+}δδ"];
DefTensor[PhiDS1B0p[-x, -y, -z, v], M4, PrintAs → "{ψ,ϕb0+}∂δδ"];
DefTensor[PhiDS2B0p[-x, -y, -z, v], M4, PrintAs → "{ψ,ϕb0+}δ∂δ"];
DefTensor[PhiDS3B0p[-x, -y, -z, v, w], M4, PrintAs → "{ψ,ϕb0+}∂δ∂δ"];

DefTensor[PhiDB1p[-a, -b, -x, -y, -z],
  M4, Antisymmetric[{-a, -b}], PrintAs → "{ψ,ϕb1+}δδ"];
DefTensor[PhiDS1B1p[-a, -b, -x, -y, -z, v], M4,
  Antisymmetric[{-a, -b}], PrintAs → "{ψ,ϕb1+}∂δδ"];
DefTensor[PhiDS2B1p[-a, -b, -x, -y, -z, v], M4,
  Antisymmetric[{-a, -b}], PrintAs → "{ψ,ϕb1+}δ∂δ"];
DefTensor[PhiDS3B1p[-a, -b, -x, -y, -z, v, w], M4,
  Antisymmetric[{-a, -b}], PrintAs → "{ψ,ϕb1+}∂δ∂δ"];

DefTensor[PhiDB1m[-a, -x, -y, -z], M4, PrintAs → "{ψ,ϕb1-}δδ"];
DefTensor[PhiDS1B1m[-a, -x, -y, -z, v], M4, PrintAs → "{ψ,ϕb1-}∂δδ"];
DefTensor[PhiDS2B1m[-a, -x, -y, -z, v], M4, PrintAs → "{ψ,ϕb1-}δ∂δ"];
DefTensor[PhiDS3B1m[-a, -x, -y, -z, v, w], M4, PrintAs → "{ψ,ϕb1-}∂δ∂δ"];
```

```
DefTensor[PhiDB2p[-a, -b, -x, -y, -z],
  M4, Symmetric[{-a, -b}], PrintAs → "{ψ,ϕb2⁺}δδ"];
DefTensor[PhiDS1B2p[-a, -b, -x, -y, -z, v], M4,
  Symmetric[{-a, -b}], PrintAs → "{ψ,ϕb2⁺}∂δδ"];
DefTensor[PhiDS2B2p[-a, -b, -x, -y, -z, v], M4,
  Symmetric[{-a, -b}], PrintAs → "{ψ,ϕb2⁺}δ∂δ"];
DefTensor[PhiDS3B2p[-a, -b, -x, -y, -z, v, w], M4,
  Symmetric[{-a, -b}], PrintAs → "{ψ,ϕb2⁺}∂δ∂δ"];

DefTensor[PhiDA0p[-x, -y, -z], M4, PrintAs → "{ψ,ϕA0⁺}δδ"];
DefTensor[PhiDS1A0p[-x, -y, -z, v], M4, PrintAs → "{ψ,ϕA0⁺}∂δδ"];
DefTensor[PhiDS2A0p[-x, -y, -z, v], M4, PrintAs → "{ψ,ϕA0⁺}δ∂δ"];
DefTensor[PhiDS3A0p[-x, -y, -z, v, w], M4, PrintAs → "{ψ,ϕA0⁺}∂δ∂δ"];

DefTensor[PhiDA0m[-x, -y, -z], M4, PrintAs → "{ψ,ϕA0⁻}δδ"];
DefTensor[PhiDS1A0m[-x, -y, -z, v], M4, PrintAs → "{ψ,ϕA0⁻}∂δδ"];
DefTensor[PhiDS2A0m[-x, -y, -z, v], M4, PrintAs → "{ψ,ϕA0⁻}δ∂δ"];
DefTensor[PhiDS3A0m[-x, -y, -z, v, w], M4, PrintAs → "{ψ,ϕA0⁻}∂δ∂δ"];

DefTensor[PhiDA1p[-a, -b, -x, -y, -z],
  M4, Antisymmetric[{-a, -b}], PrintAs → "{ψ,ϕA1⁺}δδ"];
DefTensor[PhiDS1A1p[-a, -b, -x, -y, -z, v], M4,
  Antisymmetric[{-a, -b}], PrintAs → "{ψ,ϕA1⁺}∂δδ"];
DefTensor[PhiDS2A1p[-a, -b, -x, -y, -z, v], M4,
  Antisymmetric[{-a, -b}], PrintAs → "{ψ,ϕA1⁺}δ∂δ"];
DefTensor[PhiDS3A1p[-a, -b, -x, -y, -z, v, w], M4,
  Antisymmetric[{-a, -b}], PrintAs → "{ψ,ϕA1⁺}∂δ∂δ"];

DefTensor[PhiDA1m[-a, -x, -y, -z], M4, PrintAs → "{ψ,ϕA1⁻}δδ"];
DefTensor[PhiDS1A1m[-a, -x, -y, -z, v], M4, PrintAs → "{ψ,ϕA1⁻}∂δδ"];
DefTensor[PhiDS2A1m[-a, -x, -y, -z, v], M4, PrintAs → "{ψ,ϕA1⁻}δ∂δ"];
DefTensor[PhiDS3A1m[-a, -x, -y, -z, v, w], M4, PrintAs → "{ψ,ϕA1⁻}∂δ∂δ"];

DefTensor[PhiDA2p[-a, -b, -x, -y, -z],
  M4, Symmetric[{-a, -b}], PrintAs → "{ψ,ϕA2⁺}δδ"];
DefTensor[PhiDS1A2p[-a, -b, -x, -y, -z, v], M4,
  Symmetric[{-a, -b}], PrintAs → "{ψ,ϕA2⁺}∂δδ"];
DefTensor[PhiDS2A2p[-a, -b, -x, -y, -z, v], M4,
  Symmetric[{-a, -b}], PrintAs → "{ψ,ϕA2⁺}δ∂δ"];
DefTensor[PhiDS3A2p[-a, -b, -x, -y, -z, v, w], M4,
  Symmetric[{-a, -b}], PrintAs → "{ψ,ϕA2⁺}∂δ∂δ"];

DefTensor[PhiDA2m[-a, -b, -c, -x, -y, -z],
```

```
  M4, Antisymmetric[{-a, -b}], PrintAs → "{ψ,ϕA2⁻}δδ"];
DefTensor[PhiDS1A2m[-a, -b, -c, -x, -y, -z, v], M4,
  Antisymmetric[{-a, -b}], PrintAs → "{ψ,ϕA2⁻}∂δδ"];
DefTensor[PhiDS2A2m[-a, -b, -c, -x, -y, -z, v], M4,
  Antisymmetric[{-a, -b}], PrintAs → "{ψ,ϕA2⁻}δ∂δ"];
DefTensor[PhiDS3A2m[-a, -b, -c, -x, -y, -z, v, w], M4,
  Antisymmetric[{-a, -b}], PrintAs → "{ψ,ϕA2⁻}∂δ∂δ"];

(*Apparently A2m is the only sector which
 requires extra attention beyond symmetry declarations*)
AutomaticRules[PhiDA2m, MakeRule[{PhiDA2m[a, -b, -a, -x, -y, -z], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[PhiDS1A2m, MakeRule[{PhiDS1A2m[a, -b, -a, -x, -y, -z, v], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[PhiDS2A2m, MakeRule[{PhiDS2A2m[a, -b, -a, -x, -y, -z, v], 0},
   MetricOn → All, ContractMetrics → True]];
AutomaticRules[PhiDS3A2m, MakeRule[{PhiDS3A2m[a, -b, -a, -x, -y, -z, v, w], 0},
   MetricOn → All, ContractMetrics → True]];

(*This part set up to deal with final surface term*)
DefTensor[QD[-a, -y, -z], M4, PrintAs → "{ψ,-nᵛDαπᵥᵅ}δδ"];
DefTensor[QDS1[-a, -y, -z, v], M4, PrintAs → "{ψ,-nᵛDαπᵥᵅ}∂δδ"];
DefTensor[QDS2[-a, -y, -z, v], M4, PrintAs → "{ψ,-nᵛDαπᵥᵅ}δ∂δ"];
DefTensor[QDS3[-a, -y, -z, v, w], M4, PrintAs → "{ψ,-nᵛDαπᵥᵅ}∂δ∂δ"];

(*This part to deal with the measure*)
DefTensor[JD[-a, -y, -z], M4, PrintAs → "{ψ,J}δδ"];
DefTensor[JDS1[-a, -y, -z, v], M4, PrintAs → "{ψ,J}∂δδ"];
DefTensor[JDS2[-a, -y, -z, v], M4, PrintAs → "{ψ,J}δ∂δ"];
DefTensor[JDS3[-a, -y, -z, v, w], M4, PrintAs → "{ψ,J}∂δ∂δ"];

(*This part to deal with the lapse*)
DefTensor[LapseD[-a, -y, -z], M4, PrintAs → "{ψ,N}δδ"];
DefTensor[LapseDS1[-a, -y, -z, v], M4, PrintAs → "{ψ,N}∂δδ"];
DefTensor[LapseDS2[-a, -y, -z, v], M4, PrintAs → "{ψ,N}δ∂δ"];
DefTensor[LapseDS3[-a, -y, -z, v, w], M4, PrintAs → "{ψ,N}∂δ∂δ"];
```

Placeholder vectors

```
In[•]:= (*The placeholder vectors*)
DefTensor[S1[-a], M4, PrintAs → "σ̊¹"];
DefTensor[S2[-a], M4, PrintAs → "σ̊²"];
DefTensor[S3[-a], M4, PrintAs → "σ̊³"];

StripPlaceholderVectors[Psi_, expr_] :=
  Module[{Temp, GradTemp, PsiFreeIndices, PsiFreeIndexList, PhiFreeIndexList,
    PsiFreeIndexListLength, PhiFreeIndexListString, PlaceholderVectors,
    DeltaList, PlaceholderBracketRules, PlaceholdersToDifferentiate,
    return, FreeConstraint, PlaceholderBracketActivate, ii},
   PsiFreeIndices = FindFreeIndices[Psi];
   PsiFreeIndexList =
    Developer`ToList[Delete[Map[ToString[#] &, PsiFreeIndices], 0]];
   PsiFreeIndexListLength = Length[PsiFreeIndexList];
   PlaceholderVectors = {"S1[-k]", "S2[-k]", "S3[-k]"};
   PlaceholdersToDifferentiate = {};
   For[ii = 1, ii < PsiFreeIndexListLength + 1, ii++, PlaceholdersToDifferentiate =
     Append[PlaceholdersToDifferentiate, ToExpression[StringReplace[
        PlaceholderVectors[[ii]], {"-k" → PsiFreeIndexList[[ii]]}]]]];
   Print[PlaceholdersToDifferentiate];
   Temp = expr;
   For[ii = 1, ii < PsiFreeIndexListLength + 1, ii++,
    Temp = VarAction[Temp, PlaceholdersToDifferentiate[[ii]]]];
   Temp = Temp // ToNewCanonical;
   Print[Temp];
   Temp];
```

Velocity components

```
(*
(*PBs on constraint functions*)
ConstraintHamiltonianBilinear=
 2(1/16) (ωB0p ( Lapse[]J[]PhiB0p[]PhiDB0p[-x,-y,-z]-
            CD[-v][Lapse[]J[]PhiB0p[]PhiDS1B0p[-x,-y,-z,v]]+
            CD[-v][Lapse[]J[]PhiB0p[]]PhiDS2B0p[-x,-y,-z,v]-
            CD[-w][CD[-v][Lapse[]J[]PhiB0p[]]PhiDS3B0p[-x,-y,-z,v,w]])+
        ωB1p ( Lapse[]J[]PhiB1p[-a,-b]PhiDB1p[a,b,-x,-y,-z]-
            CD[-v][Lapse[]J[]PhiB1p[-a,-b]PhiDS1B1p[a,b,-x,-y,-z,v]]+
            CD[-v][Lapse[]J[]PhiB1p[-a,-b]]PhiDS2B1p[a,b,-x,-y,-z,v]-
            CD[-w][CD[-v][Lapse[]J[]PhiB1p[-a,-b]]PhiDS3B1p[a,b,-x,-y,-z,v,w]])+
        ωB1m ( Lapse[]J[]PhiB1m[-a]PhiDB1m[a,-x,-y,-z]-
            CD[-v][Lapse[]J[]PhiB1m[-a]PhiDS1B1m[a,-x,-y,-z,v]]+
            CD[-v][Lapse[]J[]PhiB1m[-a]]PhiDS2B1m[a,-x,-y,-z,v]-
```

```
CD[-w][CD[-v][Lapse[]J[]PhiB1m[-a]]PhiDS3B1m[a,-x,-y,-z,v,w]]) +
    ωB2p ( Lapse[]J[]PhiB2p[-a,-b]PhiDB2p[a,b,-x,-y,-z]-
        CD[-v][Lapse[]J[]PhiB2p[-a,-b]PhiDS1B2p[a,b,-x,-y,-z,v]] +
        CD[-v][Lapse[]J[]PhiB2p[-a,-b]]PhiDS2B2p[a,b,-x,-y,-z,v]-
        CD[-w][CD[-v][Lapse[]J[]PhiB2p[-a,-b]]PhiDS3B2p[a,b,-x,-y,-z,v,w]]) +
    (1/4) (ωA0p ( Lapse[]J[]PhiA0p[]PhiDA0p[-x,-y,-z]-
            CD[-v][Lapse[]J[]PhiA0p[]PhiDS1A0p[-x,-y,-z,v]] +
            CD[-v][Lapse[]J[]PhiA0p[]]PhiDS2A0p[-x,-y,-z,v]-
            CD[-w][CD[-v][Lapse[]J[]PhiA0p[]]PhiDS3A0p[-x,-y,-z,v,w]]) +
        ωA0m ( Lapse[]J[]PhiA0m[]PhiDA0m[-x,-y,-z]-
            CD[-v][Lapse[]J[]PhiA0m[]PhiDS1A0m[-x,-y,-z,v]] +
            CD[-v][Lapse[]J[]PhiA0m[]]PhiDS2A0m[-x,-y,-z,v]-
            CD[-w][CD[-v][Lapse[]J[]PhiA0m[]]PhiDS3A0m[-x,-y,-z,v,w]]) +
        ωA1p ( Lapse[]J[]PhiA1p[-a,-b]PhiDA1p[a,b,-x,-y,-z]-
            CD[-v][Lapse[]J[]PhiA1p[-a,-b]PhiDS1A1p[a,b,-x,-y,-z,v]] +
            CD[-v][Lapse[]J[]PhiA1p[-a,-b]]PhiDS2A1p[a,b,-x,-y,-z,v]-
            CD[-w][
            CD[-v][Lapse[]J[]PhiA1p[-a,-b]]PhiDS3A1p[a,b,-x,-y,-z,v,w]]) +
        ωA1m ( Lapse[]J[]PhiA1m[-a]PhiDA1m[a,-x,-y,-z]-
            CD[-v][Lapse[]J[]PhiA1m[-a]PhiDS1A1m[a,-x,-y,-z,v]] +
            CD[-v][Lapse[]J[]PhiA1m[-a]]PhiDS2A1m[a,-x,-y,-z,v]-
            CD[-w][CD[-v][Lapse[]J[]PhiA1m[-a]]PhiDS3A1m[a,-x,-y,-z,v,w]]) +
        ωA2p ( Lapse[]J[]PhiA2p[-a,-b]PhiDA2p[a,b,-x,-y,-z]-
            CD[-v][Lapse[]J[]PhiA2p[-a,-b]PhiDS1A2p[a,b,-x,-y,-z,v]] +
            CD[-v][Lapse[]J[]PhiA2p[-a,-b]]PhiDS2A2p[a,b,-x,-y,-z,v]-
            CD[-w][
            CD[-v][Lapse[]J[]PhiA2p[-a,-b]]PhiDS3A2p[a,b,-x,-y,-z,v,w]]) +
        ωA2m ( Lapse[]J[]PhiA2m[-a,-b,-c]PhiDA2m[a,b,c,-x,-y,-z]-
            CD[-v][Lapse[]J[]PhiA2m[-a,-b,-c]PhiDS1A2m[a,b,c,-x,-y,-z,v]] +
            CD[-v][Lapse[]J[]PhiA2m[-a,-b,-c]]PhiDS2A2m[a,b,c,-x,-y,-z,v]-
            CD[-w][CD[-v][Lapse[]J[]PhiA2m[-a,-b,-c]]
             PhiDS3A2m[a,b,c,-x,-y,-z,v,w]]))) /.
  NewFreedoms/.Theory//ToCanonical//CollectTensors;

(*PBs for field strength tensors and ADM projectors,
remember PBs vanish on main field strength projectors as only functions of G*)
LagrangianHamiltonianBilinear=
 -2( Lapse[]J[]T[i,-m,-n]PPara[m,g]PPara[n,h](Bet1 PT1[-i,-g,-h,a,c,d] +
        Bet2 PT2[-i,-g,-h,a,c,d] +
        Bet3 PT3[-i,-g,-h,a,c,d])TD[-a,-c,-d,-x,-y,-z]-
      CD[-v][
      Lapse[]J[]T[i,-m,-n]PPara[m,g]PPara[n,h](Bet1 PT1[-i,-g,-h,a,c,d] +
         Bet2 PT2[-i,-g,-h,a,c,d] +
```

```
            Bet3 PT3[-i,-g,-h,a,c,d]) TDS1[-a,-c,-d,-x,-y,-z,v]] +
        CD[-v][
         Lapse[] J[] T[i,-m,-n] PPara[m,g] PPara[n,h] (Bet1 PT1[-i,-g,-h,a,c,d] +
             Bet2 PT2[-i,-g,-h,a,c,d] +
             Bet3 PT3[-i,-g,-h,a,c,d])] TDS2[-a,-c,-d,-x,-y,-z,v] -
        CD[-w][CD[-v][
           Lapse[] J[] T[i,-m,-n] PPara[m,g] PPara[n,h] (Bet1 PT1[-i,-g,-h,a,c,d] +
               Bet2 PT2[-i,-g,-h,a,c,d] +
               Bet3 PT3[-i,-g,-h,a,c,d])] TDS3[-a,-c,-d,-x,-y,-z,v,w]] +
        Lapse[] J[]
        (R[i,j,-m,-n] PPara[m,g] PPara[n,h] (Alp1 PR1[-i,-j,-g,-h,a,b,c,d] +
               Alp2 PR2[-i,-j,-g,-h,a,b,c,d] +
               Alp3 PR3[-i,-j,-g,-h,a,b,c,d] +
               Alp4 PR4[-i,-j,-g,-h,a,b,c,d] +
               Alp5 PR5[-i,-j,-g,-h,a,b,c,d] +
               Alp6 PR6[-i,-j,-g,-h,a,b,c,d]) - (1/4) Alp0 PPara[a,c] PPara[b,d])
        RD[-a,-b,-c,-d,-x,-y,-z] -
        CD[-v][Lapse[] J[] (R[i,j,-m,-n] PPara[m,g] PPara[n,h]
              (Alp1 PR1[-i,-j,-g,-h,a,b,c,d] +
                 Alp2 PR2[-i,-j,-g,-h,a,b,c,d] +
                 Alp3 PR3[-i,-j,-g,-h,a,b,c,d] +
                 Alp4 PR4[-i,-j,-g,-h,a,b,c,d] +
                 Alp5 PR5[-i,-j,-g,-h,a,b,c,d] +
                 Alp6 PR6[-i,-j,-g,-h,a,b,c,d]) - (1/4) Alp0 PPara[a,c] PPara[b,d])
         RDS1[-a,-b,-c,-d,-x,-y,-z,v]] +
        CD[-v][Lapse[] J[] (R[i,j,-m,-n] PPara[m,g] PPara[n,h]
               (Alp1 PR1[-i,-j,-g,-h,a,b,c,d] +
                  Alp2 PR2[-i,-j,-g,-h,a,b,c,d] +
                  Alp3 PR3[-i,-j,-g,-h,a,b,c,d] +
                  Alp4 PR4[-i,-j,-g,-h,a,b,c,d] +
                  Alp5 PR5[-i,-j,-g,-h,a,b,c,d] +
                  Alp6 PR6[-i,-j,-g,-h,a,b,c,d]) - (1/4) Alp0 PPara[a,c] PPara[b,d])]
        RDS2[-a,-b,-c,-d,-x,-y,-z,v] -
        CD[-w][CD[-v][Lapse[] J[] (R[i,j,-m,-n] PPara[m,g] PPara[
                  n,h] (Alp1 PR1[-i,-j,-g,-h,a,b,c,d] +
                    Alp2 PR2[-i,-j,-g,-h,a,b,c,d] +
                    Alp3 PR3[-i,-j,-g,-h,a,b,c,d] +
                    Alp4 PR4[-i,-j,-g,-h,a,b,c,d] +
                    Alp5 PR5[-i,-j,-g,-h,a,b,c,d] +
                    Alp6 PR6[-i,-j,-g,-h,a,b,c,d]) -
               (1/4) Alp0 PPara[a,c] PPara[b,d])] RDS3[-a,-b,-c,-d,-x,-y,-z,v,w]]) /.
    NewFreedoms /. Theory // ToCanonical // CollectTensors;
```

```
LagrangianHamiltonianBilinearMultiplier=
 -( Lapse[]J[]TLambda[i,-m,-n]PPara[m,g]PPara[n,h](cBet1 PT1[-i,-g,-h,a,c,d]+
          cBet2 PT2[-i,-g,-h,a,c,d]+
          cBet3 PT3[-i,-g,-h,a,c,d])TD[-a,-c,-d,-x,-y,-z]-
        CD[-v][Lapse[]J[]TLambda[i,-m,-n]
         PPara[m,g]PPara[n,h](cBet1 PT1[-i,-g,-h,a,c,d]+
           cBet2 PT2[-i,-g,-h,a,c,d]+
           cBet3 PT3[-i,-g,-h,a,c,d])TDS1[-a,-c,-d,-x,-y,-z,v]]+
        CD[-v][Lapse[]J[]TLambda[i,-m,-n]
          PPara[m,g]PPara[n,h](cBet1 PT1[-i,-g,-h,a,c,d]+
            cBet2 PT2[-i,-g,-h,a,c,d]+
            cBet3 PT3[-i,-g,-h,a,c,d])]TDS2[-a,-c,-d,-x,-y,-z,v]-
        CD[-w][CD[-v][Lapse[]J[]TLambda[i,-m,-n]
           PPara[m,g]PPara[n,h](cBet1 PT1[-i,-g,-h,a,c,d]+
             cBet2 PT2[-i,-g,-h,a,c,d]+
             cBet3 PT3[-i,-g,-h,a,c,d])]TDS3[-a,-c,-d,-x,-y,-z,v,w]]+
        Lapse[]J[]RLambda[i,j,-m,-n]
        PPara[m,g]PPara[n,h](cAlp1 PR1[-i,-j,-g,-h,a,b,c,d]+
          cAlp2 PR2[-i,-j,-g,-h,a,b,c,d]+
          cAlp3 PR3[-i,-j,-g,-h,a,b,c,d]+
          cAlp4 PR4[-i,-j,-g,-h,a,b,c,d]+
          cAlp5 PR5[-i,-j,-g,-h,a,b,c,d]+
          cAlp6 PR6[-i,-j,-g,-h,a,b,c,d])RD[-a,-b,-c,-d,-x,-y,-z]-
        CD[-v][Lapse[]J[]RLambda[i,j,-m,-n]
         PPara[m,g]PPara[n,h](cAlp1 PR1[-i,-j,-g,-h,a,b,c,d]+
           cAlp2 PR2[-i,-j,-g,-h,a,b,c,d]+
           cAlp3 PR3[-i,-j,-g,-h,a,b,c,d]+
           cAlp4 PR4[-i,-j,-g,-h,a,b,c,d]+
           cAlp5 PR5[-i,-j,-g,-h,a,b,c,d]+
           cAlp6 PR6[-i,-j,-g,-h,a,b,c,d])RDS1[-a,-b,-c,-d,-x,-y,-z,v]]+
        CD[-v][Lapse[]J[]RLambda[i,j,-m,-n]
          PPara[m,g]PPara[n,h](cAlp1 PR1[-i,-j,-g,-h,a,b,c,d]+
            cAlp2 PR2[-i,-j,-g,-h,a,b,c,d]+
            cAlp3 PR3[-i,-j,-g,-h,a,b,c,d]+
            cAlp4 PR4[-i,-j,-g,-h,a,b,c,d]+
            cAlp5 PR5[-i,-j,-g,-h,a,b,c,d]+
            cAlp6 PR6[-i,-j,-g,-h,a,b,c,d])]RDS2[-a,-b,-c,-d,-x,-y,-z,v]-
        CD[-w][CD[-v][Lapse[]J[]RLambda[i,j,-m,-n]
           PPara[m,g]PPara[n,h](cAlp1 PR1[-i,-j,-g,-h,a,b,c,d]+
             cAlp2 PR2[-i,-j,-g,-h,a,b,c,d]+
             cAlp3 PR3[-i,-j,-g,-h,a,b,c,d]+
             cAlp4 PR4[-i,-j,-g,-h,a,b,c,d]+
             cAlp5 PR5[-i,-j,-g,-h,a,b,c,d]+
```

```
                cAlp6 PR6[-i,-j,-g,-h,a,b,c,d])]RDS3[-a,-b,-c,-d,-x,-y,-z,v,w]])/.
      NewFreedoms/.Theory//ToCanonical//CollectTensors;

(*PB on the measure factor in front of constraint and Lagrangian parts*)
ConstraintLagrangianMeasure1=((1/16) (ωB0p (PhiB0p[]PhiB0p[])+
        ωB1p PhiB1p[-a,-b]PhiB1p[a,b]+
        ωB1m PhiB1m[-a]PhiB1m[a]+
        ωB2p PhiB2p[-a,-b]PhiB2p[a,b]+
        (1/4) (ωA0p PhiA0p[]PhiA0p[]+
           ωA0m PhiA0m[]PhiA0m[]+
           ωA1p PhiA1p[-a,-b]PhiA1p[a,b]+
           ωA1m PhiA1m[-a]PhiA1m[a]+
           ωA2p PhiA2p[-a,-b]PhiA2p[a,b]+
           ωA2m PhiA2m[-a,-b,-c]PhiA2m[a,b,c]))-
    (T[i,-m,-n]PPara[m,g]PPara[n,h](Bet1 PT1[-i,-g,-h,a,c,d]+
          Bet2 PT2[-i,-g,-h,a,c,d]+
          Bet3 PT3[-i,-g,-h,a,c,d])PPara[-c,p]PPara[-d,q]T[-a,-p,-q]+
       TLambda[i,-m,-n]PPara[m,g]PPara[n,h](cBet1 PT1[-i,-g,-h,a,c,d]+
          cBet2 PT2[-i,-g,-h,a,c,d]+
          cBet3 PT3[-i,-g,-h,a,c,d])PPara[-c,p]PPara[-d,q]T[-a,-p,-q]+
       (R[i,j,-m,-n]PPara[m,g]PPara[n,h](Alp1 PR1[-i,-j,-g,-h,a,b,c,d]+
             Alp2 PR2[-i,-j,-g,-h,a,b,c,d]+
             Alp3 PR3[-i,-j,-g,-h,a,b,c,d]+
             Alp4 PR4[-i,-j,-g,-h,a,b,c,d]+
             Alp5 PR5[-i,-j,-g,-h,a,b,c,d]+
             Alp6 PR6[-i,-j,-g,-h,a,b,c,d])-(1/2)Alp0 PPara[a,c]PPara[b,d])
        PPara[-c,p]PPara[-d,q]R[-a,-b,-p,-q]+
       RLambda[i,j,-m,-n]PPara[m,g]PPara[n,h](cAlp1 PR1[-i,-j,-g,-h,a,b,c,d]+
          cAlp2 PR2[-i,-j,-g,-h,a,b,c,d]+
          cAlp3 PR3[-i,-j,-g,-h,a,b,c,d]+
          cAlp4 PR4[-i,-j,-g,-h,a,b,c,d]+
          cAlp5 PR5[-i,-j,-g,-h,a,b,c,d]+
          cAlp6 PR6[-i,-j,-g,-h,a,b,c,d])PPara[-c,p]PPara[-d,q]R[-a,-b,-p,-q]))
  JD[-x,-y,-z];

ConstraintLagrangianMeasure2=- CD[-v][((1/16) (ωB0p (PhiB0p[]PhiB0p[])+
          ωB1p PhiB1p[-a,-b]PhiB1p[a,b]+
          ωB1m PhiB1m[-a]PhiB1m[a]+
          ωB2p PhiB2p[-a,-b]PhiB2p[a,b]+
          (1/4) (ωA0p PhiA0p[]PhiA0p[]+
             ωA0m PhiA0m[]PhiA0m[]+
             ωA1p PhiA1p[-a,-b]PhiA1p[a,b]+
             ωA1m PhiA1m[-a]PhiA1m[a]+
```

```
            ωA2p PhiA2p[-a,-b]PhiA2p[a,b]+
            ωA2m PhiA2m[-a,-b,-c]PhiA2m[a,b,c]))-
      (T[i,-m,-n]PPara[m,g]PPara[n,h](Bet1 PT1[-i,-g,-h,a,c,d]+
           Bet2 PT2[-i,-g,-h,a,c,d]+
           Bet3 PT3[-i,-g,-h,a,c,d])PPara[-c,p]PPara[-d,q]T[-a,-p,-q]+
        TLambda[i,-m,-n]PPara[m,g]PPara[n,h](cBet1 PT1[-i,-g,-h,a,c,d]+
           cBet2 PT2[-i,-g,-h,a,c,d]+
           cBet3 PT3[-i,-g,-h,a,c,d])PPara[-c,p]PPara[-d,q]T[-a,-p,-q]+
        (R[i,j,-m,-n]PPara[m,g]PPara[n,h](Alp1 PR1[-i,-j,-g,-h,a,b,c,d]+
              Alp2 PR2[-i,-j,-g,-h,a,b,c,d]+
              Alp3 PR3[-i,-j,-g,-h,a,b,c,d]+
              Alp4 PR4[-i,-j,-g,-h,a,b,c,d]+
              Alp5 PR5[-i,-j,-g,-h,a,b,c,d]+
              Alp6 PR6[-i,-j,-g,-h,a,b,c,d])-(1/2)Alp0 PPara[a,c]PPara[b,d])
         PPara[-c,p]PPara[-d,q]R[-a,-b,-p,-q]+
        RLambda[i,j,-m,-n]PPara[m,g]PPara[n,h](cAlp1 PR1[-i,-j,-g,-h,a,b,c,d]+
           cAlp2 PR2[-i,-j,-g,-h,a,b,c,d]+
           cAlp3 PR3[-i,-j,-g,-h,a,b,c,d]+
           cAlp4 PR4[-i,-j,-g,-h,a,b,c,d]+
           cAlp5 PR5[-i,-j,-g,-h,a,b,c,d]+
           cAlp6 PR6[-i,-j,-g,-h,a,b,c,d])PPara[-c,p]PPara[-d,q]R[-a,-b,-p,-q]))
    JDS1[-x,-y,-z,v]];

ConstraintLagrangianMeasure3= CD[-v][((1/16)(ωB0p (PhiB0p[]PhiB0p[])+
        ωB1p PhiB1p[-a,-b]PhiB1p[a,b]+
        ωB1m PhiB1m[-a]PhiB1m[a]+
        ωB2p PhiB2p[-a,-b]PhiB2p[a,b]+
        (1/4)(ωA0p PhiA0p[]PhiA0p[]+
           ωA0m PhiA0m[]PhiA0m[]+
           ωA1p PhiA1p[-a,-b]PhiA1p[a,b]+
           ωA1m PhiA1m[-a]PhiA1m[a]+
           ωA2p PhiA2p[-a,-b]PhiA2p[a,b]+
           ωA2m PhiA2m[-a,-b,-c]PhiA2m[a,b,c]))-
     (T[i,-m,-n]PPara[m,g]PPara[n,h](Bet1 PT1[-i,-g,-h,a,c,d]+
          Bet2 PT2[-i,-g,-h,a,c,d]+
          Bet3 PT3[-i,-g,-h,a,c,d])PPara[-c,p]PPara[-d,q]T[-a,-p,-q]+
       TLambda[i,-m,-n]PPara[m,g]PPara[n,h](cBet1 PT1[-i,-g,-h,a,c,d]+
          cBet2 PT2[-i,-g,-h,a,c,d]+
          cBet3 PT3[-i,-g,-h,a,c,d])PPara[-c,p]PPara[-d,q]T[-a,-p,-q]+
       (R[i,j,-m,-n]PPara[m,g]PPara[n,h](Alp1 PR1[-i,-j,-g,-h,a,b,c,d]+
             Alp2 PR2[-i,-j,-g,-h,a,b,c,d]+
             Alp3 PR3[-i,-j,-g,-h,a,b,c,d]+
             Alp4 PR4[-i,-j,-g,-h,a,b,c,d]+
```

```
                Alp5 PR5[-i,-j,-g,-h,a,b,c,d]+
                Alp6 PR6[-i,-j,-g,-h,a,b,c,d])-(1/2)Alp0 PPara[a,c]PPara[b,d])
           PPara[-c,p]PPara[-d,q]R[-a,-b,-p,-q]+
          RLambda[i,j,-m,-n]PPara[m,g]PPara[n,h](cAlp1 PR1[-i,-j,-g,-h,a,b,c,d]+
             cAlp2 PR2[-i,-j,-g,-h,a,b,c,d]+
             cAlp3 PR3[-i,-j,-g,-h,a,b,c,d]+
             cAlp4 PR4[-i,-j,-g,-h,a,b,c,d]+
             cAlp5 PR5[-i,-j,-g,-h,a,b,c,d]+
             cAlp6 PR6[-i,-j,-g,-h,a,b,c,d])PPara[-c,p]PPara[-d,q]R[-a,-b,-p,-q]))]
  JDS2[-x,-y,-z,v];

ConstraintLagrangianMeasure4=- CD[-w][CD[-v][((1/16)(ωB0p (PhiB0p[]PhiB0p[])+
          ωB1p PhiB1p[-a,-b]PhiB1p[a,b]+
          ωB1m PhiB1m[-a]PhiB1m[a]+
          ωB2p PhiB2p[-a,-b]PhiB2p[a,b]+
          (1/4)(ωA0p PhiA0p[]PhiA0p[]+
             ωA0m PhiA0m[]PhiA0m[]+
             ωA1p PhiA1p[-a,-b]PhiA1p[a,b]+
             ωA1m PhiA1m[-a]PhiA1m[a]+
             ωA2p PhiA2p[-a,-b]PhiA2p[a,b]+
             ωA2m PhiA2m[-a,-b,-c]PhiA2m[a,b,c]))-
      (T[i,-m,-n]PPara[m,g]PPara[n,h](Bet1 PT1[-i,-g,-h,a,c,d]+
           Bet2 PT2[-i,-g,-h,a,c,d]+
           Bet3 PT3[-i,-g,-h,a,c,d])PPara[-c,p]PPara[-d,q]T[-a,-p,-q]+
        TLambda[i,-m,-n]PPara[m,g]PPara[n,h](cBet1 PT1[-i,-g,-h,a,c,d]+
           cBet2 PT2[-i,-g,-h,a,c,d]+
           cBet3 PT3[-i,-g,-h,a,c,d])PPara[-c,p]PPara[-d,q]T[-a,-p,-q]+
        (R[i,j,-m,-n]PPara[m,g]PPara[n,h](Alp1 PR1[-i,-j,-g,-h,a,b,c,d]+
              Alp2 PR2[-i,-j,-g,-h,a,b,c,d]+
              Alp3 PR3[-i,-j,-g,-h,a,b,c,d]+
              Alp4 PR4[-i,-j,-g,-h,a,b,c,d]+
              Alp5 PR5[-i,-j,-g,-h,a,b,c,d]+
              Alp6 PR6[-i,-j,-g,-h,a,b,c,d])-(1/2)Alp0 PPara[a,c]PPara[b,d])
          PPara[-c,p]PPara[-d,q]R[-a,-b,-p,-q]+
        RLambda[i,j,-m,-n]PPara[m,g]PPara[n,h](cAlp1 PR1[-i,-j,-g,-h,a,b,c,d]+
           cAlp2 PR2[-i,-j,-g,-h,a,b,c,d]+
           cAlp3 PR3[-i,-j,-g,-h,a,b,c,d]+
           cAlp4 PR4[-i,-j,-g,-h,a,b,c,d]+
           cAlp5 PR5[-i,-j,-g,-h,a,b,c,d]+
           cAlp6 PR6[-i,-j,-g,-h,a,b,c,d])
          PPara[-c,p]PPara[-d,q]R[-a,-b,-p,-q]))]JDS3[-x,-y,-z,v,w]];

ConstraintLagrangianMeasure1=
```

```
 ConstraintLagrangianMeasure1/.NewFreedoms/.Theory//ToCanonical//CollectTensors;
ConstraintLagrangianMeasure2=
 ConstraintLagrangianMeasure2/.NewFreedoms/.Theory//ToCanonical//CollectTensors;
ConstraintLagrangianMeasure3=
 ConstraintLagrangianMeasure3/.NewFreedoms/.Theory//ToCanonical//CollectTensors;
ConstraintLagrangianMeasure4=
 ConstraintLagrangianMeasure4/.NewFreedoms/.Theory//ToCanonical//CollectTensors;

(*PB on final surface term*)
SurfaceHamiltonian= Lapse[]QD[-x,-y,-z]-
   CD[-v][Lapse[]QDS1[-x,-y,-z,v]]+
   CD[-v][Lapse[]]QDS2[-x,-y,-z,v]-
   CD[-j][CD[-v][Lapse[]]QDS3[-x,-y,-z,v,j]]-
  ( V[k]G3[m,-n](CD[-m][BPi[-k,n]]-A[w,-k,-m]BPi[-w,n])LapseD[-x,-y,-z]-
    CD[-v][
     V[k]G3[m,-n](CD[-m][BPi[-k,n]]-A[w,-k,-m]BPi[-w,n])LapseDS1[-x,-y,-z,v]]+
    CD[-v][V[k]G3[m,-n](CD[-m][BPi[-k,n]]-A[w,-k,-m]BPi[-w,n])]
     LapseDS2[-x,-y,-z,v]-
    CD[-j][CD[-v][V[k]G3[m,-n](CD[-m][BPi[-k,n]]-A[w,-k,-m]BPi[-w,n])]
      LapseDS3[-x,-y,-z,v,j]]);

ConstraintHamiltonianBilinear=
 ConstraintHamiltonianBilinear/.NewFreedoms//ToNewCanonical;
ConstraintHamiltonianBilinear=ConstraintHamiltonianBilinear//ToNewCanonical;
Print["ConstraintHamiltonianBilinear"];
ConstraintHamiltonianBilinear=
 ConstraintHamiltonianBilinear/.PActivate//ToNewCanonical;
Print["ConstraintHamiltonianBilinear"];
ConstraintHamiltonianBilinear=
 ConstraintHamiltonianBilinear/.PADMActivate//ToNewCanonical;
Print[ConstraintHamiltonianBilinear];

LagrangianHamiltonianBilinear=
 LagrangianHamiltonianBilinear/.NewFreedoms/.Theory//ToNewCanonical;
LagrangianHamiltonianBilinear=LagrangianHamiltonianBilinear//ToNewCanonical;
Print["LagrangianHamiltonianBilinear"];
LagrangianHamiltonianBilinear=
 LagrangianHamiltonianBilinear/.PActivate//ToNewCanonical;
Print["LagrangianHamiltonianBilinear"];
LagrangianHamiltonianBilinear=
 LagrangianHamiltonianBilinear/.PADMActivate//ToNewCanonical;
Print[LagrangianHamiltonianBilinear];
```

```
LagrangianHamiltonianBilinearMultiplier=
 LagrangianHamiltonianBilinearMultiplier/.NewFreedoms/.Theory//ToNewCanonical;
LagrangianHamiltonianBilinearMultiplier=
 LagrangianHamiltonianBilinearMultiplier//ToNewCanonical;
Print["LagrangianHamiltonianBilinearMultiplier"];
LagrangianHamiltonianBilinearMultiplier=
 LagrangianHamiltonianBilinearMultiplier/.PActivate//ToNewCanonical;
Print["LagrangianHamiltonianBilinearMultiplier"];
LagrangianHamiltonianBilinearMultiplier=
 LagrangianHamiltonianBilinearMultiplier/.PADMActivate//ToNewCanonical;
Print[LagrangianHamiltonianBilinearMultiplier];

ConstraintLagrangianMeasure1=
 ConstraintLagrangianMeasure1/.NewFreedoms/.Theory//ToNewCanonical;
ConstraintLagrangianMeasure1=ConstraintLagrangianMeasure1//ToNewCanonical;
Print["ConstraintLagrangianMeasure1"];
ConstraintLagrangianMeasure1=
 ConstraintLagrangianMeasure1/.PActivate//ToNewCanonical;
Print["ConstraintLagrangianMeasure1"];
ConstraintLagrangianMeasure1=
 ConstraintLagrangianMeasure1/.PADMActivate//ToNewCanonical;
Print[ConstraintLagrangianMeasure1];

ConstraintLagrangianMeasure2=
 ConstraintLagrangianMeasure2/.NewFreedoms/.Theory//ToNewCanonical;
ConstraintLagrangianMeasure2=ConstraintLagrangianMeasure2//ToNewCanonical;
Print["ConstraintLagrangianMeasure2"];
ConstraintLagrangianMeasure2=
 ConstraintLagrangianMeasure2/.PActivate//ToNewCanonical;
Print["ConstraintLagrangianMeasure2"];
ConstraintLagrangianMeasure2=
 ConstraintLagrangianMeasure2/.PADMActivate//ToNewCanonical;
Print[ConstraintLagrangianMeasure1];

ConstraintLagrangianMeasure3=
 ConstraintLagrangianMeasure3/.NewFreedoms/.Theory//ToNewCanonical;
ConstraintLagrangianMeasure3=ConstraintLagrangianMeasure3//ToNewCanonical;
Print["ConstraintLagrangianMeasure3"];
ConstraintLagrangianMeasure3=
 ConstraintLagrangianMeasure3/.PActivate//ToNewCanonical;
Print["ConstraintLagrangianMeasure3"];
ConstraintLagrangianMeasure3=
 ConstraintLagrangianMeasure3/.PADMActivate//ToNewCanonical;
Print[ConstraintLagrangianMeasure3];
```

```
ConstraintLagrangianMeasure4=
 ConstraintLagrangianMeasure4/.NewFreedoms/.Theory//ToNewCanonical;
ConstraintLagrangianMeasure4=ToOrderCanonical[ConstraintLagrangianMeasure4,1];
ConstraintLagrangianMeasure4=ConstraintLagrangianMeasure4//ToNewCanonical;
Print["ConstraintLagrangianMeasure4"];
ConstraintLagrangianMeasure4=
 ConstraintLagrangianMeasure4/.PActivate//ToNewCanonical;
Print["ConstraintLagrangianMeasure4"];
ConstraintLagrangianMeasure4=
 ConstraintLagrangianMeasure4/.PADMActivate//ToNewCanonical;
Print[ConstraintLagrangianMeasure4];

SurfaceHamiltonian=SurfaceHamiltonian/.NewFreedoms/.Theory//ToNewCanonical;
SurfaceHamiltonian=SurfaceHamiltonian//ToNewCanonical;
Print["trying pactivate"];
SurfaceHamiltonian=SurfaceHamiltonian/.PActivate//ToNewCanonical;
Print["trying pADM"];
SurfaceHamiltonian=SurfaceHamiltonian/.PADMActivate//ToNewCanonical;
Print[SurfaceHamiltonian];

Print["total"];

SuperHamiltonian=
 ConstraintHamiltonianBilinear+
   LagrangianHamiltonianBilinear+
   LagrangianHamiltonianBilinearMultiplier+
   ConstraintLagrangianMeasure1+
   ConstraintLagrangianMeasure2+
   ConstraintLagrangianMeasure3+
   ConstraintLagrangianMeasure4+
   SurfaceHamiltonian//ToNewCanonical;

Print[SuperHamiltonian];

DumpSave[NotebookDirectory[]<>"mx_cache/superhamiltonian.mx",{SuperHamiltonian}];
Print["done superhamiltonian"];
Quit[];
*)
MyImport["superhamiltonian.mx"];
SuperHamiltonian =
  ReplaceDummies[SuperHamiltonian, IndexList[l, n, m, p, q, r, s, t, u, v, w]];
Print[SuperHamiltonian];
SuperHamiltonian = SuperHamiltonian S1[x] S2[y] S3[z] // ToNewCanonical;
```

ORPHAN

```
(*
(*Super-Hamiltonian*)
DefTensor[Ham[],M4];
HamDefinition=J[] ((1/16) (ωB0p (PhiB0p[]PhiB0p[]) +
        ωB1p PhiB1p[-a,-b]PhiB1p[a,b]+
        ωB1m PhiB1m[-a]PhiB1m[a]+
        ωB2p PhiB2p[-a,-b]PhiB2p[a,b]+
        (1/4) (ωA0p PhiA0p[]PhiA0p[]+
           ωA0m PhiA0m[]PhiA0m[]+
           ωA1p PhiA1p[-a,-b]PhiA1p[a,b]+
           ωA1m PhiA1m[-a]PhiA1m[a]+
           ωA2p PhiA2p[-a,-b]PhiA2p[a,b]+
           ωA2m PhiA2m[-a,-b,-c]PhiA2m[a,b,c])) -
     (T[i,-m,-n]PPara[m,g]PPara[n,h] (Bet1 PT1[-i,-g,-h,a,c,d]+
          Bet2 PT2[-i,-g,-h,a,c,d]+
          Bet3 PT3[-i,-g,-h,a,c,d]) PPara[-c,p]PPara[-d,q]T[-a,-p,-q]+
       R[i,j,-m,-n]PPara[m,g]PPara[n,h] (Alp1 PR1[-i,-j,-g,-h,a,b,c,d]+
          Alp2 PR2[-i,-j,-g,-h,a,b,c,d]+
          Alp3 PR3[-i,-j,-g,-h,a,b,c,d]+
          Alp4 PR4[-i,-j,-g,-h,a,b,c,d]+
          Alp5 PR5[-i,-j,-g,-h,a,b,c,d]+
          Alp6 PR6[-i,-j,-g,-h,a,b,c,d]) PPara[-c,p]PPara[-d,q]R[-a,-b,-p,-q])) -
  V[k]G3[m,-n] (CD[-m][BPi[-k,n]]-A[w,-k,-m]BPi[-w,n]);
HamActivate=
 MakeRule[{Ham[],Evaluate[HamDefinition]},MetricOn→All,ContractMetrics→True];
Ham0p=Ham[]/.HamActivate/.TotalSolutions//ToCanonical//CollectTensors;


Ham0p=Ham0p/.NewFreedoms//ToCanonical;
Ham0p=Ham0p//ToCanonical//ScreenDollarIndices//CollectTensors;
Hamx0p=Hamx0p/.NewFreedoms//ToCanonical;
Hamx0p=Hamx0p//ToCanonical//ScreenDollarIndices//CollectTensors;
*)

(*
Print["Constraint:"];
Print[ConstraintHamiltonianBilinear];
Print["Lagrangian:"];
Print[LagrangianHamiltonianBilinear];
Print["Measure:"];
Print[ConstraintLagrangianMeasure];
```

```
Print["Surface:"];
Print[ SurfaceHamiltonian];

Quit[];
*)
(*
DefTensor[BPiInert[-a,-b],M4,PrintAs→"πb"];
DefTensor[APiInert[-a,-b,-c],M4,Antisymmetric[{-a,-b}],PrintAs→"πA"];
*)
(*OPEN*)
(*
DefTensor[Hamx[],M4];
HamxDefinition=J[]((1/16)(ωB0p (PhiB0p[]PhiB0p[])+
        ωB1p PhiB1p[-a,-b]PhiB1p[a,b]+
        ωB1m PhiB1m[-a]PhiB1m[a]+
        ωB2p PhiB2p[-a,-b]PhiB2p[a,b]+
        (1/4)(ωA0p PhiA0p[]PhiA0p[]+
           ωA0m PhiA0m[]PhiA0m[]+
           ωA1p PhiA1p[-a,-b]PhiA1p[a,b]+
           ωA1m PhiA1m[-a]PhiA1m[a]+
           ωA2p PhiA2p[-a,-b]PhiA2p[a,b]+
           ωA2m PhiA2m[-a,-b,-c]PhiA2m[a,b,c]))-
     (T[i,-m,-n]PPara[m,g]PPara[n,h](Bet1 PT1[-i,-g,-h,a,c,d]+
          Bet2 PT2[-i,-g,-h,a,c,d]+
          Bet3 PT3[-i,-g,-h,a,c,d])PPara[-c,p]PPara[-d,q]T[-a,-p,-q]+
       R[i,j,-m,-n]PPara[m,g]PPara[n,h](Alp1 PR1[-i,-j,-g,-h,a,b,c,d]+
          Alp2 PR2[-i,-j,-g,-h,a,b,c,d]+
          Alp3 PR3[-i,-j,-g,-h,a,b,c,d]+
          Alp4 PR4[-i,-j,-g,-h,a,b,c,d]+
          Alp5 PR5[-i,-j,-g,-h,a,b,c,d]+
          Alp6 PR6[-i,-j,-g,-h,a,b,c,d])PPara[-c,p]PPara[-d,q]R[-a,-b,-p,-q]))-
  V[k]G3[m,-n]Dx[-m][BPiInert[-k,n]];
HamxActivate=
 MakeRule[{Hamx[],Evaluate[HamxDefinition]},MetricOn→All,ContractMetrics→True];
Hamx0p=Hamx[]/.HamxActivate/.TotalSolutions//ToCanonical//CollectTensors;
*)(*CLOSE*)
(*Super-Angular Momentum and Super-Momentum*)
(*OPEN*)
(*
DefTensor[Mom[-a],M4];
MomDefinition=
 B[a1,-a](BPiP[-i,e]T[i,-a1,-e]+(1/2)APi[-i,-j,e]R[i,j,-a1,-e])-B[k,-a]
    (CD[-b][G3[b,-e]BPi[-k,e]]-A[z,-k,-b]G3[b,-e]BPi[-z,e])/.ExpandStrengths;
```

```
MomActivate=MakeRule[{Mom[-a],Evaluate[MomDefinition]},
  MetricOn→All,ContractMetrics→True];
Mom1m=Mom[-a]/.MomActivate//ToNewCanonical;

DefTensor[Momx[-a],M4];
MomxDefinition=G3[c,-a]B[a1,-c]PPara[-a1,b1]
    (BPiP[-i,e]PPara[-e,k]T[i,-b1,-k]+(1/2)APiP[-i,-j,e]PPara[-e,k]R[i,j,-b1,-k])-
   B[k,-a]G3[b,-e]Dx[-b][BPi[-k,e]]/.ExpandStrengths;
MomxActivate=MakeRule[{Momx[-a],Evaluate[MomxDefinition]},
  MetricOn→All,ContractMetrics→True];
Momx1m=Momx[-a]/.MomxActivate//ToNewCanonical;

DefTensor[Rot[-a,-b],M4,Antisymmetric[{-a,-b}]];
RotDefinition=
 2Antisymmetrize[BPi[-a,c]G3[-c,z]B[-b,-z]+APi[-d,-a,c]G3[-c,z]A[d,-b,-z],
    {-a,-b}]+CD[-c][G3[c,-z]APi[-a,-b,z]];
RotActivate=MakeRule[{Rot[-a,-b],Evaluate[RotDefinition]},
  MetricOn→All,ContractMetrics→True];
Rot1p=PPara[-a,c]PPara[-b,d]Rot[-c,-d]/.PADMActivate/.RotActivate//
  ToNewCanonical;
Rot1m=PPerp[-a,c]PPara[-b,d]Rot[-c,-d]/.PADMActivate/.RotActivate//
  ToNewCanonical;

DefTensor[Rotx[-a,-b],M4,Antisymmetric[{-a,-b}]];
RotxDefinition=2Antisymmetrize[BPi[-a,c]G3[-c,z]B[-b,-z],{-a,-b}]+
  G3[c,-z]Dx[-c][APiInert[-a,-b,z]];
RotxActivate=MakeRule[{Rotx[-a,-b],Evaluate[RotxDefinition]},
  MetricOn→All,ContractMetrics→True];
Rotx1p=PPara[-a,c]PPara[-b,d]Rotx[-c,-d]/.PADMActivate/.RotxActivate//
  ToNewCanonical;
Rotx1m=V[c]PPara[-b,d]Rotx[-c,-d]/.PADMActivate/.RotxActivate//ToNewCanonical;
*)(*CLOSE*)
```

Generalised velocity function

In[◦]:=
```
EH0 = 0;
If[EinsteinHilbert, EH0 = 1];
Print[EH0];
```

```
CanonicalVelocity[Psi_, superhamiltonian_, order_] :=
```

```
Block[{Temp, GradTemp, PsiFreeIndices, PsiFreeIndexList, PhiFreeIndexList,
   PsiFreeIndexListLength, PhiFreeIndexListString, PlaceholderVectors, DeltaList,
   PlaceholderBracketRules, return, FreeConstraint, PlaceholderBracketActivate,
   ii}, Print[Style["Calculating self-consistency for:", Red, 20]];
 Print[Psi];
 Print[Style["Stripping indices...", Blue, 16]];
 PsiFreeIndices = FindFreeIndices[Psi];
 PsiFreeIndexList =
  Developer`ToList[Delete[Map[ToString[#] &, PsiFreeIndices], 0]];
 PsiFreeIndexListLength = Length[PsiFreeIndexList];
 PlaceholderVectors = {"S1[x1]", "S2[y1]", "S3[z1]"};
 DeltaList = {"G[x1,-k]", "G[y1,-k]", "G[z1,-k]"};
 PlaceholderBracketRules = {};
 For[ii = 1, ii < PsiFreeIndexListLength + 1, ii++, PlaceholderBracketRules =
   Append[PlaceholderBracketRules, PlaceholderVectors[[ii]] →
     StringReplace[DeltaList[[ii]], {"-k" → PsiFreeIndexList[[ii]]}]]];
 Print[PlaceholderBracketRules];
 PlaceholderBracketActivate = {};
 (**)
 Print[Style["Riemann bracket...", Blue, 20, Italic]];
 Temp = PoissonBracket[Psi, PPara[-i, e] PPara[-j, f] R[-g, -h, -e, -f],
   "ToShell" → True, "Hard" → True, "Surficial" → False,
   "Order" → EH0, "GToFoliG" → False, "NesterForm" → False];
 Print[Style["Rule for coefficient of δ(x-x₁)δ(x-x₂):", Red, 16]];
 Print[Evaluate[
   ToExpression[StringReplace["RD[-g,-h,-i,-j,-x1,-y1,-z1]S1[x1]S2[y1]S3[z1]",
     PlaceholderBracketRules]]]];
 PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
   MakeRule[{Evaluate[ToExpression[
       StringReplace["RD[-g,-h,-i,-j,-x1,-y1,-z1]S1[x1]S2[y1]S3[z1]",
        PlaceholderBracketRules]]],
     Evaluate[Temp[[1]]]}, MetricOn → All, ContractMetrics → True]];
 Print[Style["Rule for coefficient of ∂δ(x-x₁)δ(x-x₂):", Red, 16]];
 Print[Evaluate[ToExpression[
    StringReplace["RDS1[-g,-h,-i,-j,-x1,-y1,-z1,z]S1[x1]S2[y1]S3[z1]",
     PlaceholderBracketRules]]]];
 PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
   MakeRule[{Evaluate[ToExpression[
       StringReplace["RDS1[-g,-h,-i,-j,-x1,-y1,-z1,v]S1[x1]S2[y1]S3[z1]",
        PlaceholderBracketRules]]],
     Evaluate[Temp[[2]]]}, MetricOn → All, ContractMetrics → True]];
 Print[Style["Rule for coefficient of δ(x-x₁)∂δ(x-x₂):", Red, 16]];
 Print[Evaluate[ToExpression[
    StringReplace["RDS2[-g,-h,-i,-j,-x1,-y1,-z1,z]S1[x1]S2[y1]S3[z1]",
```

```
    PlaceholderBracketRules]]]];
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
  MakeRule[{Evaluate[ToExpression[
      StringReplace["RDS2[-g,-h,-i,-j,-x1,-y1,-z1,v]S1[x1]S2[y1]S3[z1]",
       PlaceholderBracketRules]]],
    Evaluate[Temp[[3]]]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for coefficient of ∂δ(x-x₁)∂δ(x-x₂):", Red, 16]];
Print[Evaluate[ToExpression[
   StringReplace["RDS3[-g,-h,-i,-j,-x1,-y1,-z1,v,z]S1[x1]S2[y1]S3[z1]",
    PlaceholderBracketRules]]]];
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
  MakeRule[{Evaluate[ToExpression[
      StringReplace["RDS3[-g,-h,-i,-j,-x1,-y1,-z1,v,z]S1[x1]S2[y1]S3[z1]",
       PlaceholderBracketRules]]],
    Evaluate[Temp[[4]]]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for ∂ coefficient of ∂δ(x-x₁)δ(x-x₂):", Red, 16]];
Print[Evaluate[ToExpression[
   StringReplace["CD[-u][RDS1[-g,-h,-i,-j,-x1,-y1,-z1,z]]S1[x1]S2[y1]S3[z1]",
    PlaceholderBracketRules]]]];
GradTemp = CD[-u][Evaluate[Temp[[2]]]];
GradTemp = ToBasicForm[GradTemp, "Hard" → True, "Order" → EH0];
Print[GradTemp];
(*GradTemp=ToNesterForm[GradTemp,
   "ToShell"→True,"Hard"→True,"Order"→EH0,"GToFoliG"→False];*)
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
  MakeRule[{Evaluate[ToExpression[StringReplace[
       "CD[-u][RDS1[-g,-h,-i,-j,-x1,-y1,-z1,v]]S1[x1]S2[y1]S3[z1]",
       PlaceholderBracketRules]]],
    Evaluate[GradTemp]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for ∂ coefficient of δ(x-x₁)∂δ(x-x₂):", Red, 16]];
Print[Evaluate[ToExpression[
   StringReplace["CD[-u][RDS2[-g,-h,-i,-j,-x1,-y1,-z1,z]]S1[x1]S2[y1]S3[z1]",
    PlaceholderBracketRules]]]];
GradTemp = CD[-u][Evaluate[Temp[[3]]]];
GradTemp = ToBasicForm[GradTemp, "Hard" → True, "Order" → EH0];
Print[GradTemp];
(*GradTemp=ToNesterForm[GradTemp,
   "ToShell"→True,"Hard"→True,"Order"→EH0,"GToFoliG"→False];*)
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
  MakeRule[{Evaluate[ToExpression[StringReplace[
       "CD[-u][RDS2[-g,-h,-i,-j,-x1,-y1,-z1,v]]S1[x1]S2[y1]S3[z1]",
       PlaceholderBracketRules]]],
    Evaluate[GradTemp]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for ∂ coefficient of ∂δ(x-x₁)∂δ(x-x₂):", Red, 16]];
```

```
Print[Evaluate[ToExpression[StringReplace[
    "CD[-u][RDS3[-g,-h,-i,-j,-x1,-y1,-z1,v,z]]S1[x1]S2[y1]S3[z1]",
    PlaceholderBracketRules]]]];
GradTemp = CD[-u][Evaluate[Temp[[4]]]];
GradTemp = ToBasicForm[GradTemp, "Hard" → True, "Order" → EH0];
Print[GradTemp];
(*GradTemp=ToNesterForm[GradTemp,
   "ToShell"→True,"Hard"→True,"Order"→EH0,"GToFoliG"→False];*)
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
  MakeRule[{Evaluate[ToExpression[StringReplace[
       "CD[-u][RDS3[-g,-h,-i,-j,-x1,-y1,-z1,v,z]]S1[x1]S2[y1]S3[z1]",
       PlaceholderBracketRules]]],
    Evaluate[GradTemp]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Torsion bracket...", Blue, 20, Italic]];
Temp = PoissonBracket[Psi, PPara[-g, e] PPara[-h, f] T[-d, -e, -f],
  "ToShell" → True, "Hard" → True, "Surficial" → False,
  "Order" → EH0, "GToFoliG" → False, "NesterForm" → False];
Print[Style["Rule for coefficient of δ(x-x₁)δ(x-x₂):", Red, 16]];
Print[Evaluate[
  ToExpression[StringReplace["TD[-d,-g,-h,-x1,-y1,-z1]S1[x1]S2[y1]S3[z1]",
    PlaceholderBracketRules]]]];
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
  MakeRule[{Evaluate[
      ToExpression[StringReplace["TD[-d,-g,-h,-x1,-y1,-z1]S1[x1]S2[y1]S3[z1]",
        PlaceholderBracketRules]]],
    Evaluate[Temp[[1]]]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for coefficient of ∂δ(x-x₁)δ(x-x₂):", Red, 16]];
Print[Evaluate[
  ToExpression[StringReplace["TDS1[-d,-g,-h,-x1,-y1,-z1,z]S1[x1]S2[y1]S3[z1]",
    PlaceholderBracketRules]]]];
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
  MakeRule[{Evaluate[ToExpression[
       StringReplace["TDS1[-d,-g,-h,-x1,-y1,-z1,v]S1[x1]S2[y1]S3[z1]",
        PlaceholderBracketRules]]],
    Evaluate[Temp[[2]]]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for coefficient of δ(x-x₁)∂δ(x-x₂):", Red, 16]];
Print[Evaluate[
  ToExpression[StringReplace["TDS2[-d,-g,-h,-x1,-y1,-z1,z]S1[x1]S2[y1]S3[z1]",
    PlaceholderBracketRules]]]];
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
  MakeRule[{Evaluate[ToExpression[
       StringReplace["TDS2[-d,-g,-h,-x1,-y1,-z1,v]S1[x1]S2[y1]S3[z1]",
        PlaceholderBracketRules]]],
    Evaluate[Temp[[3]]]}, MetricOn → All, ContractMetrics → True]];
```

```
Print[Style["Rule for coefficient of ∂δ(x-x₁)∂δ(x-x₂):", Red, 16]];
Print[Evaluate[ToExpression[
   StringReplace["TDS3[-d,-g,-h,-x1,-y1,-z1,v,z]S1[x1]S2[y1]S3[z1]",
    PlaceholderBracketRules]]]];
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
  MakeRule[{Evaluate[ToExpression[
      StringReplace["TDS3[-d,-g,-h,-x1,-y1,-z1,v,z]S1[x1]S2[y1]S3[z1]",
       PlaceholderBracketRules]]],
    Evaluate[Temp[[4]]]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for ∂ coefficient of ∂δ(x-x₁)δ(x-x₂):", Red, 16]];
GradTemp = CD[-u][Evaluate[Temp[[2]]]];
GradTemp = ToBasicForm[GradTemp, "Hard" → True, "Order" → EH0];
Print[GradTemp];
(*GradTemp=ToNesterForm[GradTemp,
   "ToShell"→True,"Hard"→True,"Order"→EH0,"GToFoliG"→False];*)
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
  MakeRule[{Evaluate[ToExpression[
      StringReplace["CD[-u][TDS1[-d,-g,-h,-x1,-y1,-z1,v]]S1[x1]S2[y1]S3[z1]",
       PlaceholderBracketRules]]],
    Evaluate[GradTemp]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for ∂ coefficient of δ(x-x₁)∂δ(x-x₂):", Red, 16]];
GradTemp = CD[-u][Evaluate[Temp[[3]]]];
GradTemp = ToBasicForm[GradTemp, "Hard" → True, "Order" → EH0];
Print[GradTemp];
(*GradTemp=ToNesterForm[GradTemp,
   "ToShell"→True,"Hard"→True,"Order"→EH0,"GToFoliG"→False];*)
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
  MakeRule[{Evaluate[ToExpression[
      StringReplace["CD[-u][TDS2[-d,-g,-h,-x1,-y1,-z1,v]]S1[x1]S2[y1]S3[z1]",
       PlaceholderBracketRules]]],
    Evaluate[GradTemp]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for ∂ coefficient of ∂δ(x-x₁)∂δ(x-x₂):", Red, 16]];
GradTemp = CD[-u][Evaluate[Temp[[4]]]];
GradTemp = ToBasicForm[GradTemp, "Hard" → True, "Order" → EH0];
Print[GradTemp];
(*GradTemp=ToNesterForm[GradTemp,
   "ToShell"→True,"Hard"→True,"Order"→EH0,"GToFoliG"→False];*)
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
  MakeRule[{Evaluate[ToExpression[StringReplace[
       "CD[-u][TDS3[-d,-g,-h,-x1,-y1,-z1,v,z]]S1[x1]S2[y1]S3[z1]",
       PlaceholderBracketRules]]],
    Evaluate[GradTemp]}, MetricOn → All, ContractMetrics → True]];
(**)
Print[Style["Surface bracket...", Blue, 20, Italic]];
```

```
Temp = PoissonBracket[Psi, -V[k] G3[m, -n]
    (CD[-m][BPi[-k, n]] - A[w, -k, -m] BPi[-w, n]), "ToShell" → True, "Hard" → True,
   "Surficial" → False, "Order" → 1, "GToFoliG" → False, "NesterForm" → False];
Print[Style["Rule for coefficient of δ(x-x₁)δ(x-x₂):", Red, 16]];
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
   MakeRule[{Evaluate[ToExpression[StringReplace[
        "QD[-x1,-y1,-z1]S1[x1]S2[y1]S3[z1]", PlaceholderBracketRules]]],
     Evaluate[Temp[[1]]]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for coefficient of ∂δ(x-x₁)δ(x-x₂):", Red, 16]];
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
   MakeRule[{Evaluate[ToExpression[StringReplace[
        "QDS1[-x1,-y1,-z1,v]S1[x1]S2[y1]S3[z1]", PlaceholderBracketRules]]],
     Evaluate[Temp[[2]]]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for coefficient of δ(x-x₁)∂δ(x-x₂):", Red, 16]];
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
   MakeRule[{Evaluate[ToExpression[StringReplace[
        "QDS2[-x1,-y1,-z1,v]S1[x1]S2[y1]S3[z1]", PlaceholderBracketRules]]],
     Evaluate[Temp[[3]]]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for coefficient of ∂δ(x-x₁)∂δ(x-x₂):", Red, 16]];
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
   MakeRule[{Evaluate[ToExpression[StringReplace[
        "QDS3[-x1,-y1,-z1,v,z]S1[x1]S2[y1]S3[z1]", PlaceholderBracketRules]]],
     Evaluate[Temp[[4]]]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for ∂ coefficient of ∂δ(x-x₁)δ(x-x₂):", Red, 16]];
GradTemp = CD[-u][Evaluate[Temp[[2]]]];
GradTemp = ToBasicForm[GradTemp, "Hard" → True, "Order" → 1];
Print[GradTemp];
(*GradTemp=ToNesterForm[GradTemp,
    "ToShell"→True,"Hard"→True,"Order"→1,"GToFoliG"→False];*)
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
   MakeRule[{Evaluate[ToExpression[
       StringReplace["CD[-u][QDS1[-x1,-y1,-z1,v]]S1[x1]S2[y1]S3[z1]",
        PlaceholderBracketRules]]],
     Evaluate[GradTemp]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for ∂ coefficient of δ(x-x₁)∂δ(x-x₂):", Red, 16]];
GradTemp = CD[-u][Evaluate[Temp[[3]]]];
GradTemp = ToBasicForm[GradTemp, "Hard" → True, "Order" → 1];
Print[GradTemp];
(*GradTemp=ToNesterForm[GradTemp,
    "ToShell"→True,"Hard"→True,"Order"→1,"GToFoliG"→False];*)
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
   MakeRule[{Evaluate[ToExpression[
       StringReplace["CD[-u][QDS2[-x1,-y1,-z1,v]]S1[x1]S2[y1]S3[z1]",
```

```
        PlaceholderBracketRules]]],
      Evaluate[GradTemp]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for ∂ coefficient of ∂δ(x-x1)∂δ(x-x2):", Red, 16]];
GradTemp = CD[-u][Evaluate[Temp[[4]]]];
GradTemp = ToBasicForm[GradTemp, "Hard" → True, "Order" → 1];
Print[GradTemp];
(*GradTemp=ToNesterForm[GradTemp,
   "ToShell"→True,"Hard"→True,"Order"→1,"GToFoliG"→False];*)
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
  MakeRule[{Evaluate[ToExpression[
      StringReplace["CD[-u][QDS3[-x1,-y1,-z1,v,z]]S1[x1]S2[y1]S3[z1]",
        PlaceholderBracketRules]]],
      Evaluate[GradTemp]}, MetricOn → All, ContractMetrics → True]];
(**)
Print[Style["Measure bracket...", Blue, 20, Italic]];
Temp = PoissonBracket[Psi, Lapse[] J[], "ToShell" → True, "Hard" → True,
  "Surficial" → False, "Order" → 1, "GToFoliG" → False, "NesterForm" → False];
Print[Style["Rule for coefficient of δ(x-x1)δ(x-x2):", Red, 16]];
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
  MakeRule[{Evaluate[ToExpression[StringReplace[
        "JD[-x1,-y1,-z1]S1[x1]S2[y1]S3[z1]", PlaceholderBracketRules]]],
      Evaluate[Temp[[1]]]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for coefficient of ∂δ(x-x1)δ(x-x2):", Red, 16]];
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
  MakeRule[{Evaluate[ToExpression[StringReplace[
        "JDS1[-x1,-y1,-z1,v]S1[x1]S2[y1]S3[z1]", PlaceholderBracketRules]]],
      Evaluate[Temp[[2]]]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for coefficient of δ(x-x1)∂δ(x-x2):", Red, 16]];
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
  MakeRule[{Evaluate[ToExpression[StringReplace[
        "JDS2[-x1,-y1,-z1,v]S1[x1]S2[y1]S3[z1]", PlaceholderBracketRules]]],
      Evaluate[Temp[[3]]]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for coefficient of ∂δ(x-x1)∂δ(x-x2):", Red, 16]];
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
  MakeRule[{Evaluate[ToExpression[StringReplace[
        "JDS3[-x1,-y1,-z1,v,z]S1[x1]S2[y1]S3[z1]", PlaceholderBracketRules]]],
      Evaluate[Temp[[4]]]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for ∂ coefficient of ∂δ(x-x1)δ(x-x2):", Red, 16]];
GradTemp = CD[-u][Evaluate[Temp[[2]]]];
GradTemp = ToBasicForm[GradTemp, "Hard" → True, "Order" → 1];
Print[GradTemp];
(*GradTemp=ToNesterForm[GradTemp,
   "ToShell"→True,"Hard"→True,"Order"→1,"GToFoliG"→False];*)
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
```

```
  MakeRule[{Evaluate[ToExpression[
      StringReplace["CD[-u][JDS1[-x1,-y1,-z1,v]]S1[x1]S2[y1]S3[z1]",
       PlaceholderBracketRules]]],
    Evaluate[GradTemp]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for ∂ coefficient of δ(x-x1)∂δ(x-x2):", Red, 16]];
GradTemp = CD[-u][Evaluate[Temp[[3]]]];
GradTemp = ToBasicForm[GradTemp, "Hard" → True, "Order" → 1];
Print[GradTemp];
(*GradTemp=ToNesterForm[GradTemp,
   "ToShell"→True,"Hard"→True,"Order"→1,"GToFoliG"→False];*)
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
  MakeRule[{Evaluate[ToExpression[
      StringReplace["CD[-u][JDS2[-x1,-y1,-z1,v]]S1[x1]S2[y1]S3[z1]",
       PlaceholderBracketRules]]],
    Evaluate[GradTemp]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for ∂ coefficient of ∂δ(x-x1)∂δ(x-x2):", Red, 16]];
GradTemp = CD[-u][Evaluate[Temp[[4]]]];
GradTemp = ToBasicForm[GradTemp, "Hard" → True, "Order" → 1];
Print[GradTemp];
(*GradTemp=ToNesterForm[GradTemp,
   "ToShell"→True,"Hard"→True,"Order"→1,"GToFoliG"→False];*)
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
  MakeRule[{Evaluate[ToExpression[
      StringReplace["CD[-u][JDS3[-x1,-y1,-z1,v,z]]S1[x1]S2[y1]S3[z1]",
       PlaceholderBracketRules]]],
    Evaluate[GradTemp]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Lapse bracket...", Blue, 20, Italic]];
Temp = PoissonBracket[Psi, Lapse[], "ToShell" → True, "Hard" → True,
  "Surficial" → False, "Order" → 1, "GToFoliG" → False, "NesterForm" → False];
Print[Style["Rule for coefficient of δ(x-x1)δ(x-x2):", Red, 16]];
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
  MakeRule[{Evaluate[ToExpression[StringReplace[
       "LapseD[-x1,-y1,-z1]S1[x1]S2[y1]S3[z1]", PlaceholderBracketRules]]],
    Evaluate[Temp[[1]]]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for coefficient of ∂δ(x-x1)δ(x-x2):", Red, 16]];
PlaceholderBracketActivate =
 Join[PlaceholderBracketActivate, MakeRule[{Evaluate[
     ToExpression[StringReplace["LapseDS1[-x1,-y1,-z1,v]S1[x1]S2[y1]S3[z1]",
       PlaceholderBracketRules]]],
    Evaluate[Temp[[2]]]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for coefficient of δ(x-x1)∂δ(x-x2):", Red, 16]];
PlaceholderBracketActivate =
 Join[PlaceholderBracketActivate, MakeRule[{Evaluate[
     ToExpression[StringReplace["LapseDS2[-x1,-y1,-z1,v]S1[x1]S2[y1]S3[z1]",
```

```
        PlaceholderBracketRules]]],
     Evaluate[Temp[[3]]]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for coefficient of ∂δ(x-x₁)∂δ(x-x₂):", Red, 16]];
PlaceholderBracketActivate =
 Join[PlaceholderBracketActivate, MakeRule[{Evaluate[
      ToExpression[StringReplace["LapseDS3[-x1,-y1,-z1,v,z]S1[x1]S2[y1]S3[z1]",
        PlaceholderBracketRules]]],
     Evaluate[Temp[[4]]]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for ∂ coefficient of ∂δ(x-x₁)δ(x-x₂):", Red, 16]];
GradTemp = CD[-u][Evaluate[Temp[[2]]]];
GradTemp = ToBasicForm[GradTemp, "Hard" → True, "Order" → 1];
Print[GradTemp];
(*GradTemp=ToNesterForm[GradTemp,
   "ToShell"→True,"Hard"→True,"Order"→1,"GToFoliG"→False];*)
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
  MakeRule[{Evaluate[ToExpression[
      StringReplace["CD[-u][LapseDS1[-x1,-y1,-z1,v]]S1[x1]S2[y1]S3[z1]",
        PlaceholderBracketRules]]],
     Evaluate[GradTemp]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for ∂ coefficient of δ(x-x₁)∂δ(x-x₂):", Red, 16]];
GradTemp = CD[-u][Evaluate[Temp[[3]]]];
GradTemp = ToBasicForm[GradTemp, "Hard" → True, "Order" → 1];
Print[GradTemp];
(*GradTemp=ToNesterForm[GradTemp,
   "ToShell"→True,"Hard"→True,"Order"→1,"GToFoliG"→False];*)
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
  MakeRule[{Evaluate[ToExpression[
      StringReplace["CD[-u][LapseDS2[-x1,-y1,-z1,v]]S1[x1]S2[y1]S3[z1]",
        PlaceholderBracketRules]]],
     Evaluate[GradTemp]}, MetricOn → All, ContractMetrics → True]];
Print[Style["Rule for ∂ coefficient of ∂δ(x-x₁)∂δ(x-x₂):", Red, 16]];
GradTemp = CD[-u][Evaluate[Temp[[4]]]];
GradTemp = ToBasicForm[GradTemp, "Hard" → True, "Order" → 1];
Print[GradTemp];
(*GradTemp=ToNesterForm[GradTemp,
   "ToShell"→True,"Hard"→True,"Order"→1,"GToFoliG"→False];*)
PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
  MakeRule[{Evaluate[ToExpression[
      StringReplace["CD[-u][LapseDS3[-x1,-y1,-z1,v,z]]S1[x1]S2[y1]S3[z1]",
        PlaceholderBracketRules]]],
     Evaluate[GradTemp]}, MetricOn → All, ContractMetrics → True]];
(**)
(**)
Phis = {PhiB0p[], PhiB1p[-i, -j], PhiB1m[-i], PhiB2p[-i, -j], PhiA0p[], PhiA0m[],
```

```
  PhiA1p[-i, -j], PhiA1m[-i], PhiA2p[-i, -j], PhiA2m[-i, -j, -k]};
For[ii = 1, ii < 11, ii++, If[Evaluate[ToExpression["ShellPrim" <>
      ToString[SectorNames[[ii]]]] /. ShellFreedomsActivate] == 1, {
   FreeConstraint = Phis[[ii]];
   PhiFreeIndexList = FindFreeIndices[Evaluate[FreeConstraint]];
   PhiFreeIndexListString = StringDelete[
     StringTrim[ToString[PhiFreeIndexList], ("IndexList[" | "]")], " "];
   If[Length[PhiFreeIndexList] ≠ 0, PhiFreeIndexListString =
     PhiFreeIndexListString <> ","];
   Print[Style["Constraint bracket...", Blue, 20, Italic]];
   Print[FreeConstraint];
   Temp = PoissonBracket[Psi, FreeConstraint,
     "ToShell" → True, "Hard" → True, "Surficial" → False,
     "Order" → EH0, "GToFoliG" → False, "NesterForm" → False];
   Print[Style["Rule for coefficient of δ(x-x1)δ(x-x2):", Red, 16]];
   PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
     MakeRule[{Evaluate[ToExpression[StringReplace["PhiD" <> ToString[
          SectorNames[[ii]]] <> "[" <> ToString[PhiFreeIndexListString] <>
         "-x1,-y1,-z1]S1[x1]S2[y1]S3[z1]", PlaceholderBracketRules]]],
      Evaluate[Temp[[1]]]}, MetricOn → All, ContractMetrics → True]];
   Print[Style["Rule for coefficient of ∂δ(x-x1)δ(x-x2):", Red, 16]];
   PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
     MakeRule[{Evaluate[ToExpression[StringReplace["PhiDS1" <> ToString[
          SectorNames[[ii]]] <> "[" <> ToString[PhiFreeIndexListString] <>
         "-x1,-y1,-z1,v]S1[x1]S2[y1]S3[z1]", PlaceholderBracketRules]]],
      Evaluate[Temp[[2]]]}, MetricOn → All, ContractMetrics → True]];
   Print[Style["Rule for coefficient of δ(x-x1)∂δ(x-x2):", Red, 16]];
   PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
     MakeRule[{Evaluate[ToExpression[StringReplace["PhiDS2" <> ToString[
          SectorNames[[ii]]] <> "[" <> ToString[PhiFreeIndexListString] <>
         "-x1,-y1,-z1,v]S1[x1]S2[y1]S3[z1]", PlaceholderBracketRules]]],
      Evaluate[Temp[[3]]]}, MetricOn → All, ContractMetrics → True]];
   Print[Style["Rule for coefficient of ∂δ(x-x1)∂δ(x-x2):", Red, 16]];
   PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
     MakeRule[{Evaluate[ToExpression[StringReplace["PhiDS3" <> ToString[
          SectorNames[[ii]]] <> "[" <> ToString[PhiFreeIndexListString] <>
         "-x1,-y1,-z1,v,z]S1[x1]S2[y1]S3[z1]", PlaceholderBracketRules]]],
      Evaluate[Temp[[4]]]}, MetricOn → All, ContractMetrics → True]];
   Print[Style["Rule for ∂ coefficient of ∂δ(x-x1)δ(x-x2):", Red, 16]];
   GradTemp = CD[-u][Evaluate[Temp[[2]]]];
   GradTemp = ToBasicForm[GradTemp, "Hard" → True, "Order" → EH0];
   Print[GradTemp];
   (*GradTemp=ToNesterForm[GradTemp,
```

```
        "ToShell"→True,"Hard"→True,"Order"→EH0,"GToFoliG"→False];*)
      PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
        MakeRule[{Evaluate[ToExpression[StringReplace["CD[-u][PhiDS1" <> ToString[
              SectorNames[[ii]]] <> "[" <> ToString[PhiFreeIndexListString] <>
             "-x1,-y1,-z1,v]]S1[x1]S2[y1]S3[z1]", PlaceholderBracketRules]]],
          Evaluate[GradTemp]}, MetricOn → All, ContractMetrics → True]];
      Print[Style["Rule for ∂ coefficient of δ(x-x1)∂δ(x-x2):", Red, 16]];
      GradTemp = CD[-u][Evaluate[Temp[[3]]]];
      GradTemp = ToBasicForm[GradTemp, "Hard" → True, "Order" → EH0];
      Print[GradTemp];
      (*GradTemp=ToNesterForm[GradTemp,
        "ToShell"→True,"Hard"→True,"Order"→EH0,"GToFoliG"→False];*)
      PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
        MakeRule[{Evaluate[ToExpression[StringReplace["CD[-u][PhiDS2" <> ToString[
              SectorNames[[ii]]] <> "[" <> ToString[PhiFreeIndexListString] <>
             "-x1,-y1,-z1,v]]S1[x1]S2[y1]S3[z1]", PlaceholderBracketRules]]],
          Evaluate[GradTemp]}, MetricOn → All, ContractMetrics → True]];
      Print[Style["Rule for ∂ coefficient of ∂δ(x-x1)∂δ(x-x2):", Red, 16]];
      GradTemp = CD[-u][Evaluate[Temp[[4]]]];
      GradTemp = ToBasicForm[GradTemp, "Hard" → True, "Order" → EH0];
      Print[GradTemp];
      (*GradTemp=ToNesterForm[GradTemp,
        "ToShell"→True,"Hard"→True,"Order"→EH0,"GToFoliG"→False];*)
      PlaceholderBracketActivate = Join[PlaceholderBracketActivate,
        MakeRule[{Evaluate[ToExpression[StringReplace["CD[-u][PhiDS3" <> ToString[
              SectorNames[[ii]]] <> "[" <> ToString[PhiFreeIndexListString] <>
             "-x1,-y1,-z1,v,z]]S1[x1]S2[y1]S3[z1]", PlaceholderBracketRules]]],
          Evaluate[GradTemp]}, MetricOn → All, ContractMetrics → True]];
    }]];
  (**)
  Print[Style["Imposing commutator replacement rules...", Blue, 20, Italic]];
  return = Evaluate[superhamiltonian] /. PlaceholderBracketActivate;
  (*Print[return];*)
  return = ToOrderCanonical[return, 1];
  Print[ToBasicForm[return, "Hard" → True, "Order" → 1]];
  Print[Style["Imposing Nester form...", Blue, 20, Italic]];
  return = ToNesterForm[return, "ToShell" → True, "Hard" → True, "Order" → 1];
  Print[Style["Re-expanding η̂ because answer is a product of Nester forms...",
    Blue, 20, Italic]];
  return = return /. FoliGToG;
  return = return // ToNewCanonical;
  return = return /. GToFoliG;
  return = return // ToNewCanonical;
```

```
Print[Style["Final form of linear velocity:", Blue, 20, Italic]];
Print[return];
return];
```

Velocities

Simple spin – 1^+ case

```
(*
ChiB1mSimple1p=CanonicalVelocity[PhiB1m[-q1],SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ChiB1mSimple1p"<>".mx",
 {ChiB1mSimple1p}];
Print["finished printing"];
Quit[];
*)
MyImport["ChiB1mSimple1p.mx"];
(*
ChiA0mSimple1p=CanonicalVelocity[PhiA0m[],SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ChiA0mSimple1p"<>".mx",
 {ChiA0mSimple1p}];
Print["finished printing"];
(*Quit[];*)
*)
MyImport["ChiA0mSimple1p.mx"];
(*
ChiA2mSimple1p=CanonicalVelocity[PhiA2m[-q1,-p1,-v1],SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ChiA2mSimple1p"<>".mx",
 {ChiA2mSimple1p}];
Print["finished printing"];
Quit[];
MyImport["ChiA2mSimple1p.mx"];
*)(*this was just too complicated*)

(*
ChiA0mSimple1p=(ChiA0mSimple1p/Lapse[])/.XToV//ToNewCanonical;
Print[ChiA0mSimple1p];
PoissonBracket[ChiA0mSimple1p,PhiA0m[],
 "ToShell"→True,"Hard"→False,"Surficial"→True,"Order"→1];
Quit[];
*)
(*
ChiB1mSimple1p=(ChiB1mSimple1p/Lapse[])/.XToV//ToNewCanonical;
Print[ChiB1mSimple1p];
ChiB1mSimple1p=StripPlaceholderVectors[PhiB1m[i],ChiB1mSimple1p];
Print[ChiB1mSimple1p];

PoissonBracket[ChiB1mSimple1p,PhiB1m[-l],
 "ToShell"→True,"Hard"→False,"Surficial"→True,"Order"→1];
Quit[];
*)
```

Case 32

```
(*
ChiB0pCase32=CanonicalVelocity[PhiB0p[],SuperHamiltonian,Infinity];
DumpSave[
 NotebookDirectory[]<>"mx_cache/"<>"ChiB0pCase32"<>".mx",{ChiB0pCase32}];
ChiA0pCase32=CanonicalVelocity[PhiA0p[],SuperHamiltonian,Infinity];
DumpSave[
 NotebookDirectory[]<>"mx_cache/"<>"ChiA0pCase32"<>".mx",{ChiA0pCase32}];
ChiA1pCase32=CanonicalVelocity[PhiA1p[-q1,-p1],SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ChiA1pCase32"<>".mx",{ChiA1pCase32}];
ChiA1mCase32=CanonicalVelocity[PhiA1m[-q1],SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ChiA1mCase32"<>".mx",{ChiA1mCase32}];
ChiA2pCase32=CanonicalVelocity[PhiA2p[-q1,-p1],SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ChiA2pCase32"<>".mx",{ChiA2pCase32}];
ChiA2mCase32=CanonicalVelocity[PhiA2m[-q1,-p1,-v1],SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ChiA2mCase32"<>".mx",{ChiA2mCase32}];
Print["finished printing"];
Quit[];
*)

MyImport["ChiB0pCase32.mx"];
MyImport["ChiA0pCase32.mx"];
MyImport["ChiA1pCase32.mx"];
MyImport["ChiA1mCase32.mx"];
MyImport["ChiA2pCase32.mx"];
MyImport["ChiA2mCase32.mx"];
Print["begin"]
Print[ChiB0pCase32];
Print[ChiA0pCase32];
Print[ChiA1pCase32];
Print[ChiA1mCase32];
Print[ChiA2pCase32];
Print[ChiA2mCase32];
Print["end"]
ChiA1pCase32 = StripPlaceholderVectors[PhiA1p[i, j], ChiA1pCase32];
ChiA1mCase32 = StripPlaceholderVectors[PhiA1m[i], ChiA1mCase32];
ChiA2pCase32 = StripPlaceholderVectors[PhiA2p[i, j], ChiA2pCase32];
ChiA2mCase32 = StripPlaceholderVectors[PhiA2m[i, j, k], ChiA2mCase32];
```

```
ChiB0pActivate = MakeRule[{ChiB0p[], Evaluate[ChiB0pCase32]},
   MetricOn → All, ContractMetrics → True];
```

```
ChiA0pActivate = MakeRule[{ChiA0p[], Evaluate[ChiA0pCase32]},
   MetricOn → All, ContractMetrics → True];
ChiA1pActivate = MakeRule[{ChiA1p[-i, -j], Evaluate[ChiA1pCase32]},
   MetricOn → All, ContractMetrics → True];
ChiA1mActivate = MakeRule[{ChiA1m[-i], Evaluate[ChiA1mCase32]},
   MetricOn → All, ContractMetrics → True];
ChiA2pActivate = MakeRule[{ChiA2p[-i, -j], Evaluate[ChiA2pCase32]},
   MetricOn → All, ContractMetrics → True];
ChiA2mActivate = MakeRule[{ChiA2m[-i, -j, -k], Evaluate[ChiA2mCase32]},
   MetricOn → All, ContractMetrics → True];


ChiActivate = Join[ChiB0pActivate, ChiA0pActivate,
   ChiA1pActivate, ChiA1mActivate, ChiA2pActivate, ChiA2mActivate];

(**)
(*
ZetaB0pCase32=CanonicalVelocity[ChiB0p[],SuperHamiltonian,Infinity];
DumpSave[
 NotebookDirectory[]<>"mx_cache/"<>"ZetaB0pCase32"<>".mx",{ZetaB0pCase32}];
ZetaA0pCase32=CanonicalVelocity[ChiA0p[],SuperHamiltonian,Infinity];
DumpSave[
 NotebookDirectory[]<>"mx_cache/"<>"ZetaA0pCase32"<>".mx",{ZetaA0pCase32}];
*)
(*
ZetaA1pCase32=CanonicalVelocity[ChiA1p[-q1,-p1],SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ZetaA1pCase32"<>".mx",
 {ZetaA1pCase32}];
*)(*This turned out to be hard!*)
(*
ZetaA1mCase32=CanonicalVelocity[ChiA1m[-q1],SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ZetaA1mCase32"<>".mx",
 {ZetaA1mCase32}];
ZetaA2pCase32=CanonicalVelocity[ChiA2p[-q1,-p1],SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ZetaA2pCase32"<>".mx",
 {ZetaA2pCase32}];
*)
(*
ZetaA2mCase32=CanonicalVelocity[ChiA2m[-q1,-p1,-v1],SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ZetaA2mCase32"<>".mx",
 {ZetaA2mCase32}];
Print["finished printing"];
*)(*This also turned out to be hard!*)
(**)
```

```
MyImport["ZetaB0pCase32.mx"];
MyImport["ZetaA0pCase32.mx"];
MyImport["ZetaA1mCase32.mx"];
MyImport["ZetaA2pCase32.mx"];


Print[ZetaB0pCase32];
Print[ZetaA0pCase32];
Print[ZetaA1mCase32];
Print[ZetaA2pCase32];



Phis = {PhiB0p[], PhiA0p[], PhiA1p[-l, -m],
   PhiA1m[-l], PhiA2p[-l, -m], PhiA2m[-l, -m, -n]};

(**)
SecondaryCommutatorsOfChiB0pCase32 =
  (PoissonBracket[ChiB0p[], #, "ToShell" → True, "Hard" → False, "Surficial" → False,
      "Order" → 0, "PreTruncate" → True]) & /@ {ChiB0p[], ChiA0p[],
    ChiA1p[-l, -m], ChiA1m[-l], ChiA2p[-l, -m], ChiA2m[-l, -m, -n]};
DumpSave[NotebookDirectory[] <> "mx_cache/" <>
   "SecondaryCommutatorsOfChiB0pCase32" <> ".mx",
  {SecondaryCommutatorsOfChiB0pCase32}];
Print["here are secondary commutators"];
Print[SecondaryCommutatorsOfChiB0pCase32];
(**)
(**)
SecondaryCommutatorsOfChiA0pCase32 =
  (PoissonBracket[ChiA0p[], #, "ToShell" → True, "Hard" → False,
      "Surficial" → False, "Order" → 0, "PreTruncate" → True]) & /@
   {ChiA0p[], ChiA1p[-l, -m], ChiA1m[-l], ChiA2p[-l, -m], ChiA2m[-l, -m, -n]};
DumpSave[NotebookDirectory[] <> "mx_cache/" <>
   "SecondaryCommutatorsOfChiA0pCase32" <> ".mx",
  {SecondaryCommutatorsOfChiA0pCase32}];
Print["here are secondary commutators"];
Print[SecondaryCommutatorsOfChiA0pCase32];
(**)
(**)
SecondaryCommutatorsOfChiA1pCase32 =
  (PoissonBracket[ChiA1p[-i, -j], #, "ToShell" → True, "Hard" → False,
      "Surficial" → False, "Order" → 0, "PreTruncate" → True]) & /@
```

```
    {ChiA1p[-l, -m], ChiA1m[-l], ChiA2p[-l, -m], ChiA2m[-l, -m, -n]};
DumpSave[NotebookDirectory[] <> "mx_cache/" <>
    "SecondaryCommutatorsOfChiA1pCase32" <> ".mx",
   {SecondaryCommutatorsOfChiA1pCase32}];
Print["here are secondary commutators"];
Print[SecondaryCommutatorsOfChiA1pCase32];
(**)
(**)
SecondaryCommutatorsOfChiA1mCase32 =
   (PoissonBracket[ChiA1m[-i], #, "ToShell" → True, "Hard" → False,
        "Surficial" → False, "Order" → 0, "PreTruncate" → True]) & /@
    {ChiA1m[-l], ChiA2p[-l, -m], ChiA2m[-l, -m, -n]};
DumpSave[NotebookDirectory[] <> "mx_cache/" <>
    "SecondaryCommutatorsOfChiA1mCase32" <> ".mx",
   {SecondaryCommutatorsOfChiA1mCase32}];
Print["here are secondary commutators"];
Print[SecondaryCommutatorsOfChiA1mCase32];
(**)
(**)
SecondaryCommutatorsOfChiA2pCase32 =
   (PoissonBracket[ChiA2p[-i, -j], #, "ToShell" → True, "Hard" → False,
        "Surficial" → False, "Order" → 0, "PreTruncate" → True]) & /@
    {ChiA2p[-l, -m], ChiA2m[-l, -m, -n]};
DumpSave[NotebookDirectory[] <> "mx_cache/" <>
    "SecondaryCommutatorsOfChiA2pCase32" <> ".mx",
   {SecondaryCommutatorsOfChiA2pCase32}];
Print["here are secondary commutators"];
Print[SecondaryCommutatorsOfChiA2pCase32];
(**)
(**)
SecondaryCommutatorsOfChiA2mCase32 =
   (PoissonBracket[ChiA2m[-i, -j, -k], #, "ToShell" → True,
        "Hard" → False, "Surficial" → False, "Order" → 0,
        "PreTruncate" → True]) & /@ {ChiA2m[-l, -m, -n]};
DumpSave[NotebookDirectory[] <> "mx_cache/" <>
    "SecondaryCommutatorsOfChiA2mCase32" <> ".mx",
   {SecondaryCommutatorsOfChiA2mCase32}];
Print["here are secondary commutators"];
Print[SecondaryCommutatorsOfChiA2mCase32];
(**)

(**)
PrimaryCommutatorsOfChiB0pCase32 =
```

```
  (PoissonBracket[ChiB0p[], #, "ToShell" → True, "Hard" → False,
      "Surficial" → False, "Order" → 0, "PreTruncate" → True]) & /@ Phis;
DumpSave[NotebookDirectory[] <> "mx_cache/" <> "PrimaryCommutatorsOfChiB0pCase32" <>
   ".mx", {PrimaryCommutatorsOfChiB0pCase32}];
Print["here are Primary commutators"];
Print[PrimaryCommutatorsOfChiB0pCase32];
(**)
(**)
PrimaryCommutatorsOfChiA0pCase32 =
  (PoissonBracket[ChiA0p[], #, "ToShell" → True, "Hard" → False,
      "Surficial" → False, "Order" → 0, "PreTruncate" → True]) & /@ Phis;
DumpSave[NotebookDirectory[] <> "mx_cache/" <> "PrimaryCommutatorsOfChiA0pCase32" <>
   ".mx", {PrimaryCommutatorsOfChiA0pCase32}];
Print["here are Primary commutators"];
Print[PrimaryCommutatorsOfChiA0pCase32];
(**)
(**)
PrimaryCommutatorsOfChiA1pCase32 =
  (PoissonBracket[ChiA1p[-i, -j], #, "ToShell" → True, "Hard" → False,
      "Surficial" → False, "Order" → 0, "PreTruncate" → True]) & /@ Phis;
DumpSave[NotebookDirectory[] <> "mx_cache/" <> "PrimaryCommutatorsOfChiA1pCase32" <>
   ".mx", {PrimaryCommutatorsOfChiA1pCase32}];
Print["here are Primary commutators"];
Print[PrimaryCommutatorsOfChiA1pCase32];
(**)
(**)
PrimaryCommutatorsOfChiA1mCase32 =
  (PoissonBracket[ChiA1m[-i], #, "ToShell" → True, "Hard" → False,
      "Surficial" → False, "Order" → 0, "PreTruncate" → True]) & /@ Phis;
DumpSave[NotebookDirectory[] <> "mx_cache/" <> "PrimaryCommutatorsOfChiA1mCase32" <>
   ".mx", {PrimaryCommutatorsOfChiA1mCase32}];
Print["here are Primary commutators"];
Print[PrimaryCommutatorsOfChiA1mCase32];
(**)
(**)
PrimaryCommutatorsOfChiA2pCase32 =
  (PoissonBracket[ChiA2p[-i, -j], #, "ToShell" → True, "Hard" → False,
      "Surficial" → False, "Order" → 0, "PreTruncate" → True]) & /@ Phis;
DumpSave[NotebookDirectory[] <> "mx_cache/" <> "PrimaryCommutatorsOfChiA2pCase32" <>
   ".mx", {PrimaryCommutatorsOfChiA2pCase32}];
Print["here are Primary commutators"];
Print[PrimaryCommutatorsOfChiA2pCase32];
(**)
```

```
(**)
PrimaryCommutatorsOfChiA2mCase32 =
  (PoissonBracket[ChiA2m[-i, -j, -k], #, "ToShell" → True, "Hard" → False,
      "Surficial" → False, "Order" → 0, "PreTruncate" → True]) & /@ Phis;
DumpSave[NotebookDirectory[] <> "mx_cache/" <> "PrimaryCommutatorsOfChiA2mCase32" <>
   ".mx", {PrimaryCommutatorsOfChiA2mCase32}];
Print["here are Primary commutators"];
Print[PrimaryCommutatorsOfChiA2mCase32];
(**)

Quit[];

Print["here are the other commutators"];

MyImport["PrimaryCommutatorsOfChiB0pCase32.mx"];
MyImport["PrimaryCommutatorsOfChiB1mCase32.mx"];
MyImport["PrimaryCommutatorsOfChiB2pCase32.mx"];
MyImport["PrimaryCommutatorsOfChiA0pCase32.mx"];
MyImport["PrimaryCommutatorsOfChiA2pCase32.mx"];
MyImport["SecondaryCommutatorsOfChiB0pCase32.mx"];
MyImport["SecondaryCommutatorsOfChiB1mCase32.mx"];
MyImport["SecondaryCommutatorsOfChiB2pCase32.mx"];
MyImport["SecondaryCommutatorsOfChiA0pCase32.mx"];
MyImport["SecondaryCommutatorsOfChiA2pCase32.mx"];

Print[Style["Primaries", Red, 30]];

Print[PrimaryCommutatorsOfChiB0pCase32];
Print[PrimaryCommutatorsOfChiB1mCase32];
Print[PrimaryCommutatorsOfChiB2pCase32];
Print[PrimaryCommutatorsOfChiA0pCase32];
Print[PrimaryCommutatorsOfChiA2pCase32];

Print[Style["Secondaries", Red, 30]];

Print[SecondaryCommutatorsOfChiB0pCase32];
Print[SecondaryCommutatorsOfChiB1mCase32];
Print[SecondaryCommutatorsOfChiB2pCase32];
Print[SecondaryCommutatorsOfChiA0pCase32];
Print[SecondaryCommutatorsOfChiA2pCase32];

Quit[];
```

```
Quit[];
```

Case28

```
(*
(*
ChiB0pCase28=CanonicalVelocity[PhiB0p[],SuperHamiltonian,Infinity];
DumpSave[
 NotebookDirectory[]<>"mx_cache/"<>"ChiB0pCase28"<>".mx",{ChiB0pCase28}];
ChiB1mCase28=CanonicalVelocity[PhiB1m[-q1],SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ChiB1mCase28"<>".mx",{ChiB1mCase28}];
ChiB2pCase28=CanonicalVelocity[PhiB2p[-q1,-p1],SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ChiB2pCase28"<>".mx",{ChiB2pCase28}];
ChiA0pCase28=CanonicalVelocity[PhiA0p[],SuperHamiltonian,Infinity];
DumpSave[
 NotebookDirectory[]<>"mx_cache/"<>"ChiA0pCase28"<>".mx",{ChiA0pCase28}];
ChiA2pCase28=CanonicalVelocity[PhiA2p[-q1,-p1],SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ChiA2pCase28"<>".mx",{ChiA2pCase28}];
ChiA2mCase28=CanonicalVelocity[PhiA2m[-q1,-p1,-v1],SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ChiA2mCase28"<>".mx",{ChiA2mCase28}];
Print["finished printing"];
Quit[];
*)

MyImport["ChiB0pCase28.mx"];
MyImport["ChiB1mCase28.mx"];
MyImport["ChiB2pCase28.mx"];
MyImport["ChiA0pCase28.mx"];
MyImport["ChiA2pCase28.mx"];

Print[ChiB0pCase28];
Print[ChiB1mCase28];
Print[ChiB2pCase28];
Print[ChiA0pCase28];
Print[ChiA2pCase28];

ChiB1mCase28=StripPlaceholderVectors[PhiB1m[i],ChiB1mCase28];
ChiB2pCase28=StripPlaceholderVectors[PhiB2p[i,j],ChiB2pCase28];
ChiA2pCase28=StripPlaceholderVectors[PhiA2p[i,j],ChiA2pCase28];
```

```
ChiB0pActivate=
 MakeRule[{ChiB0p[],Evaluate[ChiB0pCase28]},MetricOn→All,ContractMetrics→True];
ChiB1mActivate=MakeRule[{ChiB1m[-i],Evaluate[ChiB1mCase28]},
  MetricOn→All,ContractMetrics→True];
ChiB2pActivate=MakeRule[{ChiB2p[-i,-j],Evaluate[ChiB2pCase28]},
  MetricOn→All,ContractMetrics→True];
ChiA0pActivate=MakeRule[{ChiA0p[],Evaluate[ChiA0pCase28]},
  MetricOn→All,ContractMetrics→True];
ChiA2pActivate=MakeRule[{ChiA2p[-i,-j],Evaluate[ChiA2pCase28]},
  MetricOn→All,ContractMetrics→True];

ChiActivate=Join[ChiB0pActivate,ChiB1mActivate,
  ChiB2pActivate,ChiA0pActivate,ChiA2pActivate];

tmp={ChiB0p[],ChiB1m[-i],ChiB2p[-i,-j],ChiA0p[],ChiA2p[-i,-j]}/.ChiActivate;
Print/@tmp;

(*
ZetaB0pCase28=CanonicalVelocity[ChiB0p[],SuperHamiltonian,Infinity];
DumpSave[
 NotebookDirectory[]<>"mx_cache/"<>"ZetaB0pCase28"<>".mx",{ZetaB0pCase28}];
ZetaB1mCase28=CanonicalVelocity[ChiB1m[-q1],SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ZetaB1mCase28"<>".mx",
 {ZetaB1mCase28}];
ZetaB2pCase28=CanonicalVelocity[ChiB2p[-q1,-p1],SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ZetaB2pCase28"<>".mx",
 {ZetaB2pCase28}];
ZetaA0pCase28=CanonicalVelocity[ChiA0p[],SuperHamiltonian,Infinity];
DumpSave[
 NotebookDirectory[]<>"mx_cache/"<>"ZetaA0pCase28"<>".mx",{ZetaA0pCase28}];
ZetaA2pCase28=CanonicalVelocity[ChiA2p[-q1,-p1],SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ZetaA2pCase28"<>".mx",
 {ZetaA2pCase28}];
Print["finished printing"];
*)

MyImport["ZetaB0pCase28.mx"];
MyImport["ZetaB1mCase28.mx"];
MyImport["ZetaB2pCase28.mx"];
MyImport["ZetaA0pCase28.mx"];

Print[ZetaB0pCase28];
Print[ZetaB1mCase28];
```

```
Print[ZetaB2pCase28];
Print[ZetaA0pCase28];


Phis={PhiB0p[],PhiB1m[-l],PhiB2p[-l,-m],PhiA0p[],PhiA2p[-l,-m],PhiA2m[-l,-m,-n]};

(**)
SecondaryCommutatorsOfChiB0pCase28=
 (PoissonBracket[ChiB0p[],#,"ToShell"→True,"Hard"→False,
     "Surficial"→False,"Order"→0,"PreTruncate"→True])&/@
  {ChiB0p[],ChiB1m[-l],ChiB2p[-l,-m],ChiA0p[],ChiA2p[-l,-m]};
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"SecondaryCommutatorsOfChiB0pCase28"<>
  ".mx",{SecondaryCommutatorsOfChiB0pCase28}];
Print["here are secondary commutators"];
Print[SecondaryCommutatorsOfChiB0pCase28];
(**)
(**)
SecondaryCommutatorsOfChiB1mCase28=
 (PoissonBracket[ChiB1m[-i],#,"ToShell"→True,"Hard"→False,
     "Surficial"→False,"Order"→0,"PreTruncate"→True])&/@
  {ChiB1m[-l],ChiB2p[-l,-m],ChiA0p[],ChiA2p[-l,-m]};
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"SecondaryCommutatorsOfChiB1mCase28"<>
  ".mx",{SecondaryCommutatorsOfChiB1mCase28}];
Print["here are secondary commutators"];
Print[SecondaryCommutatorsOfChiB1mCase28];
(**)
(**)
SecondaryCommutatorsOfChiB2pCase28=
 (PoissonBracket[ChiB2p[-i,-j],#,"ToShell"→True,"Hard"→False,"Surficial"→False,
     "Order"→0,"PreTruncate"→True])&/@{ChiB2p[-l,-m],ChiA0p[],ChiA2p[-l,-m]};
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"SecondaryCommutatorsOfChiB2pCase28"<>
  ".mx",{SecondaryCommutatorsOfChiB2pCase28}];
Print["here are secondary commutators"];
Print[SecondaryCommutatorsOfChiB2pCase28];
(**)
(**)
SecondaryCommutatorsOfChiA0pCase28=
 (PoissonBracket[ChiA0p[],#,"ToShell"→True,"Hard"→False,"Surficial"→False,
     "Order"→0,"PreTruncate"→True])&/@{ChiA0p[],ChiA2p[-l,-m]};
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"SecondaryCommutatorsOfChiA0pCase28"<>
  ".mx",{SecondaryCommutatorsOfChiA0pCase28}];
Print["here are secondary commutators"];
Print[SecondaryCommutatorsOfChiA0pCase28];
```

```
(**)
(**)
SecondaryCommutatorsOfChiA2pCase28=
 (PoissonBracket[ChiA2p[-i,-j],#,"ToShell"→True,"Hard"→False,
     "Surficial"→False,"Order"→0,"PreTruncate"→True])&/@{ChiA2p[-l,-m]};
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"SecondaryCommutatorsOfChiA2pCase28"<>
  ".mx",{SecondaryCommutatorsOfChiA2pCase28}];
Print["here are secondary commutators"];
Print[SecondaryCommutatorsOfChiA2pCase28];
(**)

(**)
PrimaryCommutatorsOfChiB0pCase28=
 (PoissonBracket[ChiB0p[],#,"ToShell"→True,"Hard"→False,
     "Surficial"→False,"Order"→0,"PreTruncate"→True])&/@Phis;
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"PrimaryCommutatorsOfChiB0pCase28"<>
  ".mx",{PrimaryCommutatorsOfChiB0pCase28}];
Print["here are Primary commutators"];
Print[PrimaryCommutatorsOfChiB0pCase28];
(**)
(**)
PrimaryCommutatorsOfChiB1mCase28=
 (PoissonBracket[ChiB1m[-i],#,"ToShell"→True,"Hard"→False,
     "Surficial"→False,"Order"→0,"PreTruncate"→True])&/@Phis;
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"PrimaryCommutatorsOfChiB1mCase28"<>
  ".mx",{PrimaryCommutatorsOfChiB1mCase28}];
Print["here are Primary commutators"];
Print[PrimaryCommutatorsOfChiB1mCase28];
(**)
(**)
PrimaryCommutatorsOfChiB2pCase28=
 (PoissonBracket[ChiB2p[-i,-j],#,"ToShell"→True,"Hard"→False,
     "Surficial"→False,"Order"→0,"PreTruncate"→True])&/@Phis;
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"PrimaryCommutatorsOfChiB2pCase28"<>
  ".mx",{PrimaryCommutatorsOfChiB2pCase28}];
Print["here are Primary commutators"];
Print[PrimaryCommutatorsOfChiB2pCase28];
(**)
(**)
PrimaryCommutatorsOfChiA0pCase28=
 (PoissonBracket[ChiA0p[],#,"ToShell"→True,"Hard"→False,
     "Surficial"→False,"Order"→0,"PreTruncate"→True])&/@Phis;
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"PrimaryCommutatorsOfChiA0pCase28"<>
```

```
  ".mx",{PrimaryCommutatorsOfChiA0pCase28}];
Print["here are Primary commutators"];
Print[PrimaryCommutatorsOfChiA0pCase28];
(**)
(**)
PrimaryCommutatorsOfChiA2pCase28=
 (PoissonBracket[ChiA2p[-i,-j],#,"ToShell"→True,"Hard"→False,
     "Surficial"→False,"Order"→0,"PreTruncate"→True])&/@Phis;
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"PrimaryCommutatorsOfChiA2pCase28"<>
  ".mx",{PrimaryCommutatorsOfChiA2pCase28}];
Print["here are Primary commutators"];
Print[PrimaryCommutatorsOfChiA2pCase28];
(**)

Print["here are the other commutators"];

MyImport["PrimaryCommutatorsOfChiB0pCase28.mx"];
MyImport["PrimaryCommutatorsOfChiB1mCase28.mx"];
MyImport["PrimaryCommutatorsOfChiB2pCase28.mx"];
MyImport["PrimaryCommutatorsOfChiA0pCase28.mx"];
MyImport["PrimaryCommutatorsOfChiA2pCase28.mx"];
MyImport["SecondaryCommutatorsOfChiB0pCase28.mx"];
MyImport["SecondaryCommutatorsOfChiB1mCase28.mx"];
MyImport["SecondaryCommutatorsOfChiB2pCase28.mx"];
MyImport["SecondaryCommutatorsOfChiA0pCase28.mx"];
MyImport["SecondaryCommutatorsOfChiA2pCase28.mx"];

Print[Style["Primaries",Red,30]];

Print[PrimaryCommutatorsOfChiB0pCase28];
Print[PrimaryCommutatorsOfChiB1mCase28];
Print[PrimaryCommutatorsOfChiB2pCase28];
Print[PrimaryCommutatorsOfChiA0pCase28];
Print[PrimaryCommutatorsOfChiA2pCase28];

Print[Style["Secondaries",Red,30]];

Print[SecondaryCommutatorsOfChiB0pCase28];
Print[SecondaryCommutatorsOfChiB1mCase28];
Print[SecondaryCommutatorsOfChiB2pCase28];
Print[SecondaryCommutatorsOfChiA0pCase28];
Print[SecondaryCommutatorsOfChiA2pCase28];
```

```
Quit[];
*)
```

Case 17

```
(*
(*
ChiB0pCase17=CanonicalVelocity[PhiB0p[],SuperHamiltonian,Infinity];
DumpSave[
 NotebookDirectory[]<>"mx_cache/"<>"ChiB0pCase17"<>".mx",{ChiB0pCase17}];
(*
ChiB1pCase17=CanonicalVelocity[PhiB1p[-q1,-p1],SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ChiB1pCase17"<>".mx",{ChiB1pCase17}];
*)
ChiA0pCase17=CanonicalVelocity[PhiA0p[],SuperHamiltonian,Infinity];
DumpSave[
 NotebookDirectory[]<>"mx_cache/"<>"ChiA0pCase17"<>".mx",{ChiA0pCase17}];
(*
ChiA0mCase17=CanonicalVelocity[PhiA0m[],SuperHamiltonian,Infinity];
DumpSave[
 NotebookDirectory[]<>"mx_cache/"<>"ChiA0mCase17"<>".mx",{ChiA0mCase17}];
*)

ChiA2pCase17=CanonicalVelocity[PhiA2p[-q1,-p1],SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ChiA2pCase17"<>".mx",{ChiA2pCase17}];
ChiA2mCase17=CanonicalVelocity[PhiA2m[-q1,-p1,-v1],SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ChiA2mCase17"<>".mx",{ChiA2mCase17}];
Print["finished printing"];
Quit[];
*)

MyImport["ChiB0pCase17.mx"];
MyImport["ChiB1pCase17.mx"];
MyImport["ChiA0pCase17.mx"];
MyImport["ChiA0mCase17.mx"];
MyImport["ChiA2pCase17.mx"];
MyImport["ChiA2mCase17.mx"];

Print[ChiB0pCase17];
Print[ChiB1pCase17];
Print[ChiA0pCase17];
Print[ChiA0mCase17];
Print[ChiA2pCase17];
Print[ChiA2mCase17];
```

```
(*ChiB0pCase17=StripPlaceholderVectors[PhiB0p[],ChiB0pCase17];*)
ChiB1pCase17=StripPlaceholderVectors[PhiB1p[i,j],ChiB1pCase17];
(*ChiA0pCase17=StripPlaceholderVectors[PhiA0p[],ChiA0pCase17];*)
(*ChiA0mCase17=StripPlaceholderVectors[PhiA0m[],ChiA0mCase17];*)
ChiA2pCase17=StripPlaceholderVectors[PhiA2p[i,j],ChiA2pCase17];
ChiA2mCase17=StripPlaceholderVectors[PhiA2m[i,j,k],ChiA2mCase17];

ChiB0pActivate=
 MakeRule[{ChiB0p[],Evaluate[ChiB0pCase17]},MetricOn→All,ContractMetrics→True];
ChiB1pActivate=MakeRule[{ChiB1p[-i,-j],Evaluate[ChiB1pCase17]},
  MetricOn→All,ContractMetrics→True];
ChiA0pActivate=MakeRule[{ChiA0p[],Evaluate[ChiA0pCase17]},
  MetricOn→All,ContractMetrics→True];
ChiA0mActivate=MakeRule[{ChiA0m[],Evaluate[ChiA0mCase17]},
  MetricOn→All,ContractMetrics→True];
ChiA2pActivate=MakeRule[{ChiA2p[-i,-j],Evaluate[ChiA2pCase17]},
  MetricOn→All,ContractMetrics→True];
ChiA2mActivate=MakeRule[{ChiA2m[-i,-j,-k],Evaluate[ChiA2mCase17]},
  MetricOn→All,ContractMetrics→True];

ChiActivate=Join[ChiB0pActivate,ChiB1pActivate,
  ChiA0pActivate,ChiA0mActivate,ChiA2pActivate,ChiA2mActivate];

(*
ZetaB0pCase17=CanonicalVelocity[ChiB0pCase17,SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ZetaB0pCase17"<>".mx",
 {ZetaB0pCase17}];
ZetaA0pCase17=CanonicalVelocity[ChiA0pCase17,SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ZetaA0pCase17"<>".mx",
 {ZetaA0pCase17}];
ZetaA2pCase17=CanonicalVelocity[ChiA2pCase17,SuperHamiltonian,Infinity];
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"ZetaA2pCase17"<>".mx",
 {ZetaA2pCase17}];
Print["finished printing"];
*)
MyImport["ZetaB0pCase17.mx"];
MyImport["ZetaA0pCase17.mx"];
MyImport["ZetaA2pCase17.mx"];

Phis={PhiB0p[],PhiB1p[-l,-m],PhiA0p[],PhiA0m[],PhiA2p[-l,-m],PhiA2m[-l,-m,-n]};
```

```
(**)
SecondaryCommutatorsOfChiB0pCase17=
 (PoissonBracket[ChiB0p[],#,"ToShell"→True,"Hard"→False,
     "Surficial"→False,"Order"→0,"PreTruncate"→True])&/@
  {ChiB0p[],ChiA0p[],ChiA0m[],ChiA2p[-l,-m],ChiA2m[-l,-m,-n]};
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"SecondaryCommutatorsOfChiB0pCase17"<>
  ".mx",{SecondaryCommutatorsOfChiB0pCase17}];
Print["here are secondary commutators"];
Print[SecondaryCommutatorsOfChiB0pCase17];
(**)
(**)
SecondaryCommutatorsOfChiA0pCase17=
 (PoissonBracket[ChiA0p[],#,"ToShell"→True,"Hard"→False,
     "Surficial"→False,"Order"→0,"PreTruncate"→True])&/@
  {ChiA0p[],ChiA0m[],ChiA2p[-l,-m],ChiA2m[-l,-m,-n]};
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"SecondaryCommutatorsOfChiA0pCase17"<>
  ".mx",{SecondaryCommutatorsOfChiA0pCase17}];
Print["here are secondary commutators"];
Print[SecondaryCommutatorsOfChiA0pCase17];
(**)
(*
SecondaryCommutatorsOfChiA0mCase17=
 (PoissonBracket[ChiA0m[],#,"ToShell"→True,"Hard"→False,"Surficial"→False,
     "Order"→0,"PreTruncate"→True])&/@{ChiA0m[],ChiA2p[-l,-m],ChiA2m[-l,-m,-n]};
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"SecondaryCommutatorsOfChiA0mCase17"<>
  ".mx",{SecondaryCommutatorsOfChiA0mCase17}];
Print["here are secondary commutators"];
Print[SecondaryCommutatorsOfChiA0mCase17];
*)
(*
SecondaryCommutatorsOfChiA2pCase17=
 (PoissonBracket[ChiA2p[-i,-j],#,"ToShell"→True,"Hard"→False,"Surficial"→False,
     "Order"→0,"PreTruncate"→True])&/@{ChiA2p[-l,-m],ChiA2m[-l,-m,-n]};
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"SecondaryCommutatorsOfChiA2pCase17"<>
  ".mx",{SecondaryCommutatorsOfChiA2pCase17}];
Print["here are secondary commutators"];
Print[SecondaryCommutatorsOfChiA2pCase17];
*)
(*
SecondaryCommutatorsOfChiA2mCase17=
 (PoissonBracket[ChiA2m[-i,-j,-k],#,"ToShell"→True,"Hard"→False,
     "Surficial"→False,"Order"→0,"PreTruncate"→True])&/@{ChiA2m[-l,-m,-n]};
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"SecondaryCommutatorsOfChiA2mCase17"<>
```

```
  ".mx",{SecondaryCommutatorsOfChiA2mCase17}];
Print["here are secondary commutators"];
Print[SecondaryCommutatorsOfChiA2mCase17];
*)

(*
PrimaryCommutatorsOfChiB0pCase17=
 (PoissonBracket[ChiB0p[],#,"ToShell"→True,"Hard"→False,
     "Surficial"→False,"Order"→0,"PreTruncate"→True])&/@Phis;
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"PrimaryCommutatorsOfChiB0pCase17"<>
  ".mx",{PrimaryCommutatorsOfChiB0pCase17}];
Print["here are Primary commutators"];
Print[PrimaryCommutatorsOfChiB0pCase17];
*)
(*
PrimaryCommutatorsOfChiA0pCase17=
 (PoissonBracket[ChiA0p[],#,"ToShell"→True,"Hard"→False,
     "Surficial"→False,"Order"→0,"PreTruncate"→True])&/@Phis;
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"PrimaryCommutatorsOfChiA0pCase17"<>
  ".mx",{PrimaryCommutatorsOfChiA0pCase17}];
Print["here are Primary commutators"];
Print[PrimaryCommutatorsOfChiA0pCase17];
*)
(*
PrimaryCommutatorsOfChiA0mCase17=
 (PoissonBracket[ChiA0m[],#,"ToShell"→True,"Hard"→False,
     "Surficial"→False,"Order"→0,"PreTruncate"→True])&/@Phis;
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"PrimaryCommutatorsOfChiA0mCase17"<>
  ".mx",{PrimaryCommutatorsOfChiA0mCase17}];
Print["here are Primary commutators"];
Print[SPrimaryCommutatorsOfChiA0mCase17];
*)
(*
PrimaryCommutatorsOfChiA2pCase17=
 (PoissonBracket[ChiA2p[-i,-j],#,"ToShell"→True,"Hard"→False,
     "Surficial"→False,"Order"→0,"PreTruncate"→True])&/@Phis;
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"PrimaryCommutatorsOfChiA2pCase17"<>
  ".mx",{PrimaryCommutatorsOfChiA2pCase17}];
Print["here are Primary commutators"];
Print[PrimaryCommutatorsOfChiA2pCase17];
*)
(*
PrimaryCommutatorsOfChiA2mCase17=
```

```
 (PoissonBracket[ChiA2m[-i,-j,-k],#,"ToShell"→True,"Hard"→False,
     "Surficial"→False,"Order"→0,"PreTruncate"→True])&/@Phis;
DumpSave[NotebookDirectory[]<>"mx_cache/"<>"PrimaryCommutatorsOfChiA2mCase17"<>
  ".mx",{PrimaryCommutatorsOfChiA2mCase17}];
Print["here are Primary commutators"];
Print[PrimaryCommutatorsOfChiA2mCase17];
*)

Print["here is the B1p primary"];


Print["here are the other commutators"];

MyImport["PrimaryCommutatorsOfChiB0pCase17.mx"];
MyImport["PrimaryCommutatorsOfChiA0pCase17.mx"];
MyImport["PrimaryCommutatorsOfChiA0mCase17.mx"];
MyImport["PrimaryCommutatorsOfChiA2pCase17.mx"];
MyImport["PrimaryCommutatorsOfChiA2mCase17.mx"];
MyImport["SecondaryCommutatorsOfChiB0pCase17.mx"];
MyImport["SecondaryCommutatorsOfChiA0pCase17.mx"];
MyImport["SecondaryCommutatorsOfChiA0mCase17.mx"];
MyImport["SecondaryCommutatorsOfChiA2pCase17.mx"];
MyImport["SecondaryCommutatorsOfChiA2mCase17.mx"];

Print[Style["Primaries",Red,30]];

Print[PrimaryCommutatorsOfChiB0pCase17];
Print[PrimaryCommutatorsOfChiA0pCase17];
Print[PrimaryCommutatorsOfChiA0mCase17];
Print[PrimaryCommutatorsOfChiA2pCase17];
Print[PrimaryCommutatorsOfChiA2mCase17];

Print[Style["Secondaries",Red,30]];

Print[SecondaryCommutatorsOfChiB0pCase17];
Print[SecondaryCommutatorsOfChiA0pCase17];
Print[SecondaryCommutatorsOfChiA0mCase17];
Print[SecondaryCommutatorsOfChiA2pCase17];
Print[SecondaryCommutatorsOfChiA2mCase17];

Quit[];
*)
```

cache

Cache binaries

cache

Cache the kernel state as the main HiGGS binary:

cache

In[244]:=
```
Print["The context on quitting HiGGS_sources.nb is ", $Context, "."];
DumpSave[NotebookDirectory[] <> "bin/HiGGS.mx"];
```

documentation

Build documentation

documentation

Export this notebook as the documentation:

documentation

In[246]:=
```
FrontEndExecute@{FrontEndToken[InputNotebook[], "SelectAll"],
    FrontEndToken[InputNotebook[], "SelectionOpenAllGroups"]};
Export[NotebookDirectory[] <> "Documentation/HiGGS_sources.pdf",
   EvaluationNotebook[]];
```