In this worksheet show how to derive the differential operator for the Ice-cream graph Witten diagram of section 5.6 of the paper
Algorithm for differential equations for Feynman integrals in general dimensions https://arxiv.org/abs/2401.09908
by
Leonardo de la Cruz and Pierre Vanhove

```
SetOptions[$FrontEnd, EvaluationCompletionAction → "ShowTiming"]
```

This worksheet needs FiniteFlow https://arxiv.org/abs/1905.08019

```
<< FiniteFlow`
```

Routines for the Griffiths-Dwork reduction

The routines follow the step of the reduction given in section 3.1.1, 3.1.2 and 3.1.3 of the paper

```
SaveFile = False;
(* Change this to True if you want to save intermediate results*)
```

```
JacIdeal[pol_, vars__] := D[pol, #] & /@ vars;
DegreeHomogeneity[f_, xxx__] :=
  Exponent[Simplify[(f /. Table[xxtmp → xxtmp λ, {xxtmp, xxx}]) / f], λ];
```

```
allMonoms[n_, deg_, x_] :=
 DeleteCases[Coefficient[(List @@ Expand[(1 + Total[Array[x, n]]) ^deg] /.
      j_Integer * monom : _ :→ monom) /. {x[i_] :→ λ x[i]}, λ^deg], 0]
```

```
homPols[n_, deg_, sym_] :=
  Module[{mons}, If[deg == 0, sym, mons = allMonoms[n, deg, x];
    Return[(Plus @@ (sym /@ ((Exponent[#, Array[x, n]] & /@ mons)) mons)) /.
      {sym[A__] :→ sym[Sequence @@ A]}];]];
```

```
getEquations[pol_, vars__] :=
  (CoefficientArrays[{pol}, vars]["NonzeroValues"]) // Through // Flatten;
```

Reduction with respect to F

```
ReductionF[M_, degreeP_, P_, coeffPtmp_, varunknown_, Fpol_, var_, varlong_] :=
 Block[{degtmp, poltoreduce, Chattmp, systmp, varsystmp,
   solfiniteflowtmp, Chattmpresult, coeffPresult, JacFtmp, Filenametmp},
  Print["Reduction with respect to Jac(", Fpol, ")"];
  JacFtmp = JacIdeal[Fpol, var];
  degtmp = degreeP - DegreeHomogeneity[JacFtmp, var][[1]];
  Print["degree coefficient ", degtmp];
  Chattmp =
   Table[homPols[Length[var], degtmp, varunknown[r]], {r, Length[var]}];
  If[ListQ[P], poltoreduce = M + coeffPtmp.P - Chattmp.JacFtmp,
   poltoreduce = M + coeffPtmp * P - Chattmp.JacFtmp];
  systmp = getEquations[poltoreduce, var];
  varsystmp = Complement[Variables[systmp], varlong];
  Print["Number of equations ", Length[systmp],
   ", number of variables ", Length[varsystmp]];
  If[SaveFile, Filenametmp =
     "Reduction-F-Case-" <> ToString[degreeP] <> "-" <> DateString[
       {"ISODate", "-", "Hour", ":", "Minute", ":", "Second"}] <> ".txt";
    Print["system saved in ", Filenametmp];
    Save[Filenametmp, {systmp, varsystmp}];] × Print["Calling Finite Flow "];
  solfiniteflowtmp = FFDenseSolve[Equal[#, 0] & /@ systmp,
    varsystmp, "ApplyFunction" → Together, MaxPrimes → 100];
  If[solfiniteflowtmp == FFImpossible, Echo["System cannot be solved"]];
  Print["finite flow done -- length result ", Length[solfiniteflowtmp]];
  Chattmpresult = Chattmp /. solfiniteflowtmp // Expand;
  coeffPresult = coeffPtmp /. solfiniteflowtmp // Expand;
  {Chattmpresult, degtmp, coeffPresult, solfiniteflowtmp}]
```

Reduction with respect to U

```
ReductionU[M_, degree_, Qnametmp_, Upol_, var_, varlong_] :=
 Block[{degtmp, poltoreduce, Qtmp, systmp, varsystmp,
   solfiniteflowtmp, polU, Mresult, Qtmpresult, JacUtmp, Filenametmp},
  Print["Reduction with respect to Jac(", Upol, ")"];
  JacUtmp = JacIdeal[Upol, var];
  polU = M.JacUtmp;
  degtmp = degree - 1;
  Print["Degree quotient ", degtmp];
  Qtmp = homPols[Length[var], degtmp, Qnametmp];
  poltoreduce = polU - Qtmp * Upol;
  systmp = DeleteCases[Flatten[CoefficientList[poltoreduce, var]], 0];
  varsystmp = Complement[Variables[systmp], varlong];
  Print["Number of equations ", Length[systmp],
   ", number of variables ", Length[varsystmp]];
  If[SaveFile,
    Filenametmp = "Reduction-U-Case-" <> ToString[degree] <> "-" <> DateString[
       {"ISODate", "-", "Hour", ":", "Minute", ":", "Second"}] <> ".txt";
    Print["system saved in ", Filenametmp];
    Save[Filenametmp, {systmp, varsystmp}];] × Print["Calling Finite Flow "];
  solfiniteflowtmp = FFDenseSolve[Equal[#, 0] & /@ systmp,
    varsystmp, "ApplyFunction" → Together, MaxPrimes → 100];
  If[solfiniteflowtmp == FFImpossible, Echo["System cannot be solved"]];
  Print["finite flow done -- length result ", Length[solfiniteflowtmp]];
  Mresult = M /. solfiniteflowtmp;
  Qtmpresult = Qtmp /. solfiniteflowtmp;
  {Mresult, degtmp, Qtmpresult, solfiniteflowtmp}]
```

The boundary term using equation 3.39 without including the factor of Ω_Γ

```
Bvec[Chat_, derivativeorder_, Fpol_, powerF_] :=
  Chat / (Fpol ^ (derivativeorder - 1)) / (derivativeorder - 1 + powerF);
```

Routine for the parametric representation from the propagator representation of a Feynman graph

```
PropagatorToParametric3[listprop_, varloop_, rules_] :=
 Block[{listtmp, listtmp2, powertmp, Qtmp, Qtmp2, preftmp,
   mattmp, Utmp, Jtmp, ptmp, Ftmp, omegatmp}, {listtmp = listprop;
   listtmp2 = Table[List @@ listtmp⟦i⟧, {i, Length[listtmp]}];
   powertmp = Table[-listtmp2⟦i, 2⟧ // Simplify, {i, Length[listtmp2]}];
   Print[listtmp2];
   Qtmp = Sum[x[i] × listtmp2⟦i, 1⟧, {i, Length[listtmp2]}] /. xx_ . yy_ → xx yy;
   preftmp = Product[
     x[i] ^ (powertmp⟦i⟧ - 1) / Gamma[powertmp⟦i⟧], {i, Length[listtmp2]}];
   mattmp = DiagonalMatrix[Table[Coefficient[Qtmp, varloop⟦i⟧, 2], {i,
        Length[varloop]}]] + Table[Coefficient[Coefficient[Qtmp, varloop⟦i⟧],
        varloop⟦j⟧] / 2, {i, 1, Length[varloop]}, {j, 1, Length[varloop]}];
   Utmp = Det[mattmp];
   Qtmp2 = Qtmp - varloop.(mattmp.varloop) - ptmp.varloop // FullSimplify;
   ptmp =
    Table[Coefficient[Qtmp2, varloop⟦i⟧] // Simplify, {i, Length[varloop]}];
   Jtmp = Qtmp - varloop.(mattmp.varloop) - ptmp.varloop // FullSimplify;
   Ftmp = Simplify[Expand[Together[
        Det[mattmp] (Jtmp - 1 / 4 ptmp.(Inverse[mattmp].ptmp))]] //. rules];
   omegatmp = Plus @@ powertmp - Length[varloop] * dim / 2;
   {preftmp * Pi ^ (dim / 2 * Length[varloop]) * Gamma[omegatmp] *
     Utmp ^ (omegatmp - dim / 2) / Ftmp ^ (omegatmp), Utmp, Ftmp, (preftmp * Pi ^
        (dim / 2 * Length[varloop]) * Gamma[omegatmp]) /. x[i_] :→ 1, omegatmp}}]
```

Ice-cream analytic regularisation from eq. 5.64 of [arXiv:2312.13803]

Initialisation

The propagator representation is given by

```
IceCream = PropagatorToParametric3[{1 / (L2.L2) ^ (1 + κ),
    1 / (L4.L4) ^ (1 + κ), 1 / ((L2 + L4 + Q).(L2 + L4 + Q)) ^ (1 + κ),
    1 / ((L2 + L4 + Qt).(L2 + L4 + Qt)) ^ (1 + κ)}, {L2, L4},
   {Q^2 → u, Qt^2 → v, Q Qt → (u + v - 1) / 2}][[1]] /. dim → 4 // Simplify;
```

```
{{L2.L2, -1 - κ}, {L4.L4, -1 - κ},
 {(L2 + L4 + Q).(L2 + L4 + Q), -1 - κ}, {(L2 + L4 + Qt).(L2 + L4 + Qt), -1 - κ}}
```

```
Uicecream = IceCream[[2]]
```

```
x[2] (x[3] + x[4]) + x[1] (x[2] + x[3] + x[4])
```

```
Ficecream = IceCream[[3]]
```

```
u x[1] × x[2] × x[3] + (v x[1] × x[2] + (x[1] + x[2]) x[3]) x[4]
```

```
IntIceCream = IceCream[[1]]
```

$$\frac{1}{\text{Gamma}[1+\kappa]^4}\pi^4\,\text{Gamma}[4\kappa]\,x[1]^\kappa\,x[2]^\kappa\,x[3]^\kappa\,x[4]^\kappa\,(u\,x[1]\times x[2]\times x[3]+(v\,x[1]\times x[2]+(x[1]+x[2])\,x[3])\,x[4])^{-4\kappa}\,(x[2]\,(x[3]+x[4])+x[1]\,(x[2]+x[3]+x[4]))^{-2+4\kappa}$$

```
var = Sort[Variables[Uicecream]]
```

```
{x[1], x[2], x[3], x[4]}
```

```
Kinematics = Complement[Variables[Ficecream], Variables[Uicecream]]
```

```
{u, v}
```

```
varlong = Join[{κ}, Kinematics, var]
```

```
{κ, u, v, x[1], x[2], x[3], x[4]}
```

```
listderivativeu = Reverse[
  Table[Numerator[Simplify[D[IntIceCream, {u, i}] / IntIceCream]], {i, 1, 3}]]
listdegreeu = DegreeHomogeneity[listderivativeu, var]
```

$$\{-8\,\kappa\,(1+2\,\kappa)\,(1+4\,\kappa)\,x[1]^3\,x[2]^3\,x[3]^3,\ 4\,\kappa\,(1+4\,\kappa)\,x[1]^2\,x[2]^2\,x[3]^2,\ -4\,\kappa\,x[1]\times x[2]\times x[3]\}$$

{9, 6, 3}

```
listderivativev = Reverse[
  Table[Numerator[Simplify[D[IntIceCream, {v, i}] / IntIceCream]], {i, 1, 3}]]
listdegreev = DegreeHomogeneity[listderivativev, var]
```

$$\{-8\,\kappa\,(1+2\,\kappa)\,(1+4\,\kappa)\,x[1]^3\,x[2]^3\,x[4]^3,\ 4\,\kappa\,(1+4\,\kappa)\,x[1]^2\,x[2]^2\,x[4]^2,\ -4\,\kappa\,x[1]\times x[2]\times x[4]\}$$

{9, 6, 3}

```
listderivativeuv =
 {Numerator[Simplify[D[D[IntIceCream, {v, 1}], {u, 1}] / IntIceCream]]}
listdegreeuv = DegreeHomogeneity[listderivativeuv, var]
```

$$\{4\,\kappa\,(1+4\,\kappa)\,x[1]^2\,x[2]^2\,x[3]\times x[4]\}$$

{6}

```
powerU = Coefficient[Simplify[PowerExpand[Log[IntIceCream]]], Log[Uicecream]]
```

$-2+4\,\kappa$

```
powerF =
 -Coefficient[Simplify[PowerExpand[Log[IntIceCream]]], Log[Ficecream]]
```

$4\,\kappa$

```
powerQ = Coefficient[Simplify[PowerExpand[Log[IntIceCream]]], Log[x[1]]]
```

κ

```
Q = x[1] × x[2] × x[3] × x[4];
```

operator L1

We derive the partial differential operator

```
derivativeorder = 2;
RedFStep1 = ReductionF[listderivativev[[2]], listdegreev[[2]],
   {listderivativeuv[[1]]}, {c1}, λ1, Ficecream, var, varlong];
RedUStep1 =
  ReductionU[RedFStep1[[1]], RedFStep1[[2]], μ2, Uicecream, var, varlong];
solnRstep1 = ReductionU[RedUStep1[[1]], RedFStep1[[2]], μ21, Q, var, varlong];
Mstep1 =
  (Sum[D[RedUStep1[[1]][[i]], var[[i]]], {i, Length[var]}] + powerU * RedUStep1[[3]] +
      powerQ * solnRstep1[[3]] ) / (derivativeorder - 1 + powerF) /. solnRstep1[[4]];
```

```
Reduction with respect to Jac(u x[1] × x[2] × x[3] + (v x[1] × x[2] + (x[1] + x[2]) x[3]) x[4])
degree coefficient 4
Number of equations 80, number of variables 141
Calling Finite Flow
finite flow done -- length result 80
Reduction with respect to Jac(x[2] (x[3] + x[4]) + x[1] (x[2] + x[3] + x[4]))
Degree quotient 3
Number of equations 48, number of variables 81
Calling Finite Flow
finite flow done -- length result 46
Reduction with respect to Jac(x[1] × x[2] × x[3] × x[4])
Degree quotient 3
Number of equations 68, number of variables 55
Calling Finite Flow
finite flow done -- length result 41
```

```
derivativeorder = 1;
RedFStep2 = ReductionF[Mstep1, listdegreev[[3]], {listderivativeu[[3]],
    listderivativev[[3]]}, {c2u, c2v}, λ2, Ficecream, var, varlong];
RedUStep2 =
  ReductionU[RedFStep2[[1]], RedFStep2[[2]], μ2, Uicecream, var, varlong];
solnRstep2 = ReductionU[RedUStep2[[1]], RedFStep2[[2]], μ22, Q, var, varlong];
Mstep2 =
  (Sum[D[solnRstep2[[1]][[i]], var[[i]]], {i, Length[var]}] + powerU * RedUStep2[[3]] +
      powerQ * solnRstep2[[3]]) / (derivativeorder - 1 + powerF) /. solnRstep2[[4]];
```

```
Reduction with respect to Jac(u x[1] × x[2] × x[3] + (v x[1] × x[2] + (x[1] + x[2]) x[3]) x[4])
degree coefficient 1
Number of equations 20, number of variables 32
Calling Finite Flow
finite flow done -- length result 19
Reduction with respect to Jac(x[2] (x[3] + x[4]) + x[1] (x[2] + x[3] + x[4]))
Degree quotient 0
Number of equations 7, number of variables 14
Calling Finite Flow
finite flow done -- length result 7
Reduction with respect to Jac(x[1] × x[2] × x[3] × x[4])
Degree quotient 0
Number of equations 13, number of variables 8
Calling Finite Flow
finite flow done -- length result 8
```

```
listPF1 =
 {-Mstep2, c2 u, c2 v, c1, 1} //. RedFStep1[[4]] //. RedUStep1[[4]] //. solnRstep1[[
    4]] //. RedFStep2[[4]] //. RedUStep2[[4]] //. solnRstep2[[4]] // Simplify
```

$$\left\{\frac{8\,\kappa^2}{v\,(-1+u+v)},\ \frac{u+3\,u\,\kappa}{v\,(-1+u+v)},\ \frac{v+3\,(-1+u)\,\kappa+6\,v\,\kappa}{v\,(-1+u+v)},\ \frac{2\,u}{-1+u+v},\ 1\right\}$$

operator L2

We derive the partial differential operator

```
derivativeorder = 2;
RedFStep1 = ReductionF[listderivativeu[[2]], listdegreeu[[2]],
  listderivativev[[2]], c1, Subscript[λ, 1], Ficecream, var, varlong];
RedUStep1 =
  ReductionU[RedFStep1[[1]], RedFStep1[[2]], μ1, Uicecream, var, varlong];
solnRstep1 = ReductionU[RedUStep1[[1]], RedFStep1[[2]], μ11, Q, var, varlong];
Mstep1 =
  (Sum[D[RedUStep1[[1]][[i]], var[[i]]], {i, Length[var]}] + powerU * RedUStep1[[3]] +
     powerQ * solnRstep1[[3]] ) / (derivativeorder - 1 + powerF) /. solnRstep1[[4]];
```

```
Reduction with respect to Jac(u x[1] × x[2] × x[3] + (v x[1] × x[2] + (x[1] + x[2]) x[3]) x[4])
degree coefficient 4
Number of equations 80, number of variables 141
Calling Finite Flow
finite flow done -- length result 80
Reduction with respect to Jac(x[2] (x[3] + x[4]) + x[1] (x[2] + x[3] + x[4]))
Degree quotient 3
Number of equations 48, number of variables 81
Calling Finite Flow
finite flow done -- length result 46
Reduction with respect to Jac(x[1] × x[2] × x[3] × x[4])
Degree quotient 3
Number of equations 68, number of variables 55
Calling Finite Flow
finite flow done -- length result 41
```

```
derivativeorder = 1;
RedFStep2 = ReductionF[Mstep1, listdegreev[[3]], {listderivativeu[[3]],
    listderivativev[[3]]}, {c2u, c2v}, λ2, Ficecream, var, varlong];
RedUStep2 =
  ReductionU[RedFStep2[[1]], RedFStep2[[2]], μ2, Uicecream, var, varlong];
solnRstep2 = ReductionU[RedUStep2[[1]], RedFStep2[[2]], μ22, Q, var, varlong];
Mstep2 =
 (Sum[D[solnRstep2[[1]][[i]], var[[i]]], {i, Length[var]}] + powerU * RedUStep2[[3]] +
    powerQ * solnRstep2[[3]]) / (derivativeorder - 1 + powerF) /. solnRstep2[[4]]
```

```
Reduction with respect to Jac(u x[1] × x[2] × x[3] + (v x[1] × x[2] + (x[1] + x[2]) x[3]) x[4])
degree coefficient 1
Number of equations 20, number of variables 32
Calling Finite Flow
finite flow done -- length result 19
Reduction with respect to Jac(x[2] (x[3] + x[4]) + x[1] (x[2] + x[3] + x[4]))
Degree quotient 0
Number of equations 7, number of variables 14
Calling Finite Flow
finite flow done -- length result 7
Reduction with respect to Jac(x[1] × x[2] × x[3] × x[4])
Degree quotient 0
Number of equations 13, number of variables 8
Calling Finite Flow
finite flow done -- length result 8
```

```
0
```

```
listPF2 =
 {-Mstep2, c2u, c2v, c1, 1} //. RedFStep1[[4]] //. RedUStep1[[4]] //. solnRstep1[[
     4]] //. RedFStep2[[4]] //. RedUStep2[[4]] //. solnRstep2[[4]] // Simplify
```

$$\left\{0, \frac{3\kappa}{u}, -\frac{3\kappa}{u}, -\frac{v}{u}, 1\right\}$$

operator L3

We derive the partial differential operator

```
derivativeorder = 3;
RedFStep1 = ReductionF[listderivativeu[[1]],
   listdegreeu[[1]], 0, 0, λ1, Ficecream, var, varlong];
RedUStep1 =
  ReductionU[RedFStep1[[1]], RedFStep1[[2]], μ1, Uicecream, var, varlong];
solnRstep1 = ReductionU[RedUStep1[[1]], RedFStep1[[2]], μ11, Q, var, varlong];
(*Building M using equation 3.31*)
Mstep1 =
  (Sum[D[RedUStep1[[1]][[i]], var[[i]]], {i, Length[var]}] + powerU * RedUStep1[[3]] +
      powerQ * solnRstep1[[3]] ) / (derivativeorder - 1 + powerF) /. solnRstep1[[4]];
```

```
Reduction with respect to Jac(u x[1] × x[2] × x[3] + (v x[1] × x[2] + (x[1] + x[2]) x[3]) x[4])

degree coefficient 7

Number of equations 216, number of variables 480

Calling Finite Flow

finite flow done -- length result 216

Reduction with respect to Jac(x[2] (x[3] + x[4]) + x[1] (x[2] + x[3] + x[4]))

Degree quotient 6

Number of equations 154, number of variables 348

Calling Finite Flow

finite flow done -- length result 152

Reduction with respect to Jac(x[1] × x[2] × x[3] × x[4])

Degree quotient 6

Number of equations 216, number of variables 280

Calling Finite Flow

finite flow done -- length result 168
```

```
derivativeorder = 2;
RedFStep2 = ReductionF[Mstep1, listdegreev[[2]],
   listderivativev[[2]], c2 v, λ2, Ficecream, var, varlong];
RedUStep2 =
  ReductionU[RedFStep2[[1]], RedFStep2[[2]], μ2, Uicecream, var, varlong];
solnRstep2 = ReductionU[RedUStep2[[1]], RedFStep2[[2]], μ23, Q, var, varlong];
(*Building M using equation 3.31*)
Mstep2 =
  (Sum[D[solnRstep2[[1]][[i]], var[[i]]], {i, Length[var]}] + powerU * RedUStep2[[3]] +
      powerQ * solnRstep2[[3]]) / (derivativeorder - 1 + powerF) /. solnRstep2[[4]];
```

```
Reduction with respect to Jac(u x[1] × x[2] × x[3] + (v x[1] × x[2] + (x[1] + x[2]) x[3]) x[4])
degree coefficient 4
Number of equations 84, number of variables 253
Calling Finite Flow
finite flow done -- length result 84
Reduction with respect to Jac(x[2] (x[3] + x[4]) + x[1] (x[2] + x[3] + x[4]))
Degree quotient 3
Number of equations 56, number of variables 154
Calling Finite Flow
finite flow done -- length result 52
Reduction with respect to Jac(x[1] × x[2] × x[3] × x[4])
Degree quotient 3
Number of equations 80, number of variables 122
Calling Finite Flow
finite flow done -- length result 72
```

```
derivativeorder = 1;
RedFStep3 = ReductionF[Mstep2, listdegreev⟦3⟧, {listderivativev⟦3⟧,
    listderivativeu⟦3⟧}, {c1v, c1u}, λ3, Ficecream, var, varlong];
RedUStep3 =
  ReductionU[RedFStep3⟦1⟧, RedFStep3⟦2⟧, μ3, Uicecream, var, varlong];
solnRstep3 = ReductionU[RedUStep3⟦1⟧, RedFStep3⟦2⟧, μ32, Q, var, varlong];
(*Building M using equation 3.31*)
Mstep3 =
  (Sum[D[solnRstep3⟦1⟧⟦i⟧, var⟦i⟧], {i, Length[var]}] + powerU * RedUStep3⟦3⟧ +
      powerQ * solnRstep3⟦3⟧) / (derivativeorder - 1 + powerF) /. solnRstep3⟦4⟧;
```

```
Reduction with respect to Jac(u x[1] × x[2] × x[3] + (v x[1] × x[2] + (x[1] + x[2]) x[3]) x[4])
degree coefficient 1
Number of equations 20, number of variables 68
Calling Finite Flow
finite flow done -- length result 20
Reduction with respect to Jac(x[2] (x[3] + x[4]) + x[1] (x[2] + x[3] + x[4]))
Degree quotient 0
Number of equations 10, number of variables 16
Calling Finite Flow
finite flow done -- length result 9
Reduction with respect to Jac(x[1] × x[2] × x[3] × x[4])
Degree quotient 0
Number of equations 13, number of variables 8
Calling Finite Flow
finite flow done -- length result 8
```

```
listPF3 =
  {-Mstep3, c1 u, c1 v, c2 v, 1} //. RedFStep1[[4]] //. RedUStep1[[4]] //. solnRstep1[[
        4]] //. RedFStep2[[4]] //. RedUStep2[[4]] //. solnRstep2[[4]] //.
     RedFStep3[[4]] //. RedUStep3[[4]] //. solnRstep3[[4]] // Simplify
```

$$\left\{-\frac{12\,(1+u-v)\,(-1+\kappa)\,\kappa^2}{u^2\left(u^2+(-1+v)^2-2\,u\,(1+v)\right)},\right.$$

$$\frac{-6\,(-1+v)^2\,\kappa\,(1+3\,\kappa)+u^2\left(5-29\,\kappa^2\right)+u\left(1+18\,\kappa+29\,\kappa^2+v\left(-1+6\,\kappa+43\,\kappa^2\right)\right)}{2\,u^2\left(u^2+(-1+v)^2-2\,u\,(1+v)\right)},$$

$$\left(v^2\left(-3-9\,\kappa+36\,\kappa^2\right)-v\left(-3+u-12\,\kappa+18\,u\,\kappa+63\,\kappa^2+59\,u\,\kappa^2\right)+3\,(-1+u)\,\kappa\,(1-9\,\kappa+9\,u\,(1+\kappa))\right)\Big/\left(2\,u^2\left(u^2+(-1+v)^2-2\,u\,(1+v)\right)\right),$$

$$\left.\frac{v\left(9\,u^2\,(1+\kappa)+(-1+v)^2\,(-1+9\,\kappa)-2\,u\,(1+v)\,(4+9\,\kappa)\right)}{2\,u^2\left(u^2+(-1+v)^2-2\,u\,(1+v)\right)},\,1\right\}$$