Symbolic Computation in Particle Physics using Wolfram Language

Hee Sok Chung

Korea University

November 10, 2023

Calculations in Particle Physics Theory

- Particle scattering is the main pillar of particle physics research. Hence particle physics theory research requires calculation of scattering processes.
- For well-defined models, algorithms for computing scattering processes can be found, so that much of the calculation can be automated on the computer by using symbolic computation languages.
- Naturally, MATHEMATICA has long been a standard tool for particle physics theory research.
- In this talk, I will show some typical examples for usage of MATHEMATICA in calculations of scattering processes.

Outline

Typically, computation of scattering processes involves these steps:

- Generation of Feynman diagrams depicting a scattering process
- Reduction of degrees of freedom involving spin and internal symmetries
- Reduction of Feynman integrals
- Asymptotic expansion of Feynman integrals

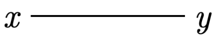
Many parts of each step can be automated.

Feynman Diagrams

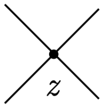
In particle physics, Feynman diagrams are graphical representations of a scattering amplitude.

- A Feynman diagram is the product of each element of the diagram divided by a symmetry factor. List of value of each element can be looked up from Feynman rules.
- Different Feynman diagrams add.

Example of Feynman rules for a simple model (ϕ^4 theory) :



$D_F(x-y)$



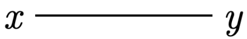
$\lambda \int dz$

Generation of Feynman Diagrams

Generating function for Feynman diagrams:

$$Z[J] = \exp\left[\frac{1}{2}\int dx \int dy J(x) D_F(x-y) J(y)\right]$$

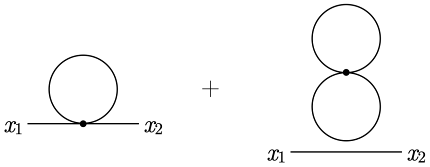
Each Feynman diagram can be generated by repeatedly differentiating by J at each end of lines and setting $J=0$ at the end. Simplest example:



$$= \frac{\partial}{\partial J(x)} \frac{\partial}{\partial J(y)} Z[J]\bigg|_{J=0} = D_F(x-y)$$

Generation of Feynman Diagrams

One-vertex example:



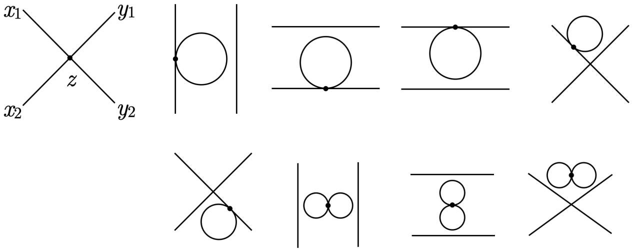
$$= \int dz \frac{\partial}{\partial J(x_1)} \frac{\partial}{\partial J(x_2)} \frac{\lambda}{4!} \left(\frac{\partial}{\partial J(z)} \right)^4 Z[J] \bigg|_{J=0}$$

$$= 2 \times 3! \times \frac{\lambda}{4!} \int dz D_F(x_1 - z) D_F(x_2 - z) D_F(z - z)$$

$$+3 \times \frac{\lambda}{4!} D_F(x_1 - x_2) \int dz D_F(z - z) \times D_F(z - z)$$

Generation of Feynman Diagrams

Another one-vertex example:



$$= \int dz \frac{\partial}{\partial J(x_1)} \frac{\partial}{\partial J(x_2)} \frac{\partial}{\partial J(y_1)} \frac{\partial}{\partial J(y_2)} \frac{\lambda}{4!} \left(\frac{\partial}{\partial J(z)} \right)^4 Z[J] \Big|_{J=0}$$

$$= 4! \times \frac{\lambda}{4!} \int dz D_F(x_1 - z) D_F(x_2 - z) D_F(y_1 - z) D_F(y_2 - z)$$

$$+2 \times 3! \times \frac{\lambda}{4!} \int dz D_F(z - z) D_F(x_1 - z) D_F(x_2 - z) D_F(y_1 - y_2) + \text{permutations}$$

$$+3 \times \frac{\lambda}{4!} \int dz D_F(z - z) \times D_F(z - z) D_F(x_1 - x_2) D_F(y_1 - y_2) + \text{permutations}$$

Generation of Feynman Diagrams

When generating Feynman diagrams by hand, we must

- account for all possible diagrams without missing any
- correctly account for all combinatorial factors (symmetry factors)

which can become labor intensive and error prone with growing number of vertices.

Such a tedious task is better done with computers.

Automated Generation of Feynman Diagrams

A sample MATHEMATICA code:

```
(* external points *)
WW0 = {sc[a1], sc[a2], sc[b1], sc[b2]};
(* vertex points *)
WW = Union[Table[sc[v[n]], {n, 1, 2}], WW0];
(* Feynman propagator *)
SetAttributes[DF, Orderless];
(* Generating functional *)
GFunc = Exp[ 1/2 Sum[J[xl] DF[xl, xr] J[xr], {xl, WW}, {xr, WW}]];
```

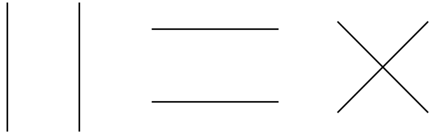
Automated Generation of Feynman Diagrams

Diagrams with no vertex:

```
CorrF = Apply[D, Union[{GFunc}, Table[J[x], {x, WW0}]]];
Corr[0] = CorrF //. {J[x_] :> 0}
```

Output:

```
DF[sc[a1], sc[b2]] DF[sc[a2], sc[b1]] +
 DF[sc[a1], sc[b1]] DF[sc[a2], sc[b2]] +
 DF[sc[a1], sc[a2]] DF[sc[b1], sc[b2]]
```



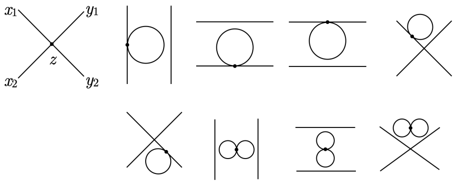
Automated Generation of Feynman Diagrams

Diagrams with one vertex:

```
CorrF = Apply[D, Union[{GFunc}, Table[J[x], {x, WW0}]]];
Corr[1] = (1/4!) (CorrF//D[#, {J[sc[v[1]]], 4}] &) //. {
            J[x_]:>0} // Expand
```

Output:

```
DF[sc[a1], sc[v[1]]] DF[sc[a2], sc[v[1]]] DF[sc[b1], sc[v[1]]] DF[sc[b2], sc[v[1]]] +
 1/2 DF[sc[a1], sc[v[1]]] DF[sc[a2], sc[v[1]]] DF[sc[b1], sc[b2]] DF[sc[v[1]], sc[v[1]]] +
 1/2 DF[sc[a1], sc[v[1]]] DF[sc[a2], sc[b2]] DF[sc[b1], sc[v[1]]] DF[sc[v[1]], sc[v[1]]] +
 1/2 DF[sc[a1], sc[b2]] DF[sc[a2], sc[v[1]]] DF[sc[b1], sc[v[1]]] DF[sc[v[1]], sc[v[1]]] +
 1/2 DF[sc[a1], sc[v[1]]] DF[sc[a2], sc[b1]] DF[sc[b2], sc[v[1]]] DF[sc[v[1]], sc[v[1]]] +
 1/2 DF[sc[a1], sc[b1]] DF[sc[a2], sc[v[1]]] DF[sc[b2], sc[v[1]]] DF[sc[v[1]], sc[v[1]]] +
 1/2 DF[sc[a1], sc[a2]] DF[sc[b1], sc[v[1]]] DF[sc[b2], sc[v[1]]] DF[sc[v[1]], sc[v[1]]] +
 1/8 DF[sc[a1], sc[b2]] DF[sc[a2], sc[b1]] DF[sc[v[1]], sc[v[1]]]^2 +
 1/8 DF[sc[a1], sc[b1]] DF[sc[a2], sc[b2]] DF[sc[v[1]], sc[v[1]]]^2 +
 1/8 DF[sc[a1], sc[a2]] DF[sc[b1], sc[b2]] DF[sc[v[1]], sc[v[1]]]^2
```



Automated Generation of Feynman Diagrams

In particle physics, we only want diagrams that are fully connected, so that they depict a genuine scattering event. Sample code:

```
Connected[a_] := Block[{topo, toporule1, toporule2, return},
  toporule1 = {DF[x__]*topo[y___] :> topo[{x}, y], DF[x__]^n_*topo[y___] :> topo[{x}, y]};
  topo[x1___, {x2___, z_, x3___}, x4___, {x5___, z_, x6___}, x7___] =
   topo[x1, {x2, x3, z, x5, x6}, x4, x7];
  return = a*topo[] //. toporule1;
  return = return //. {topo[x___] :> {x}};
  If[Length[return] > 1, 0, 1]
  ]
```

Connected diagrams with one vertex:

```
Sum[x*Connected[x], {x, Apply[List, Corr[1]]}]
```

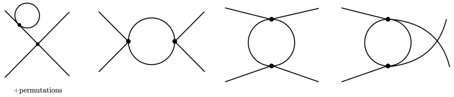
Output:

```
DF[sc[a1], sc[v[1]]] DF[sc[a2], sc[v[1]]] DF[sc[b1], sc[v[1]]] DF[sc[b2], sc[v[1]]]
```

Automated Generation of Feynman Diagrams

Connected diagrams with two vertices:

```
CorrF = Apply[D, Union[{GFunc}, Table[J[x], {x, WW0}]]];
Corr[2] = (1/2!) (1/4!)^2 (CorrF // D[#, {J[sc[v[1]]], 4}] & //
      D[#, {J[sc[v[2]]], 4}] &) //. {J[x_] :> 0} // Expand
RemoveDummy[f_, {ix__}] := Block[{return, lperm, lf, org},
  lperm = Length[{ix}]!; lf = Length[f];
  Do[return[n] = Table[f[[nf]] /.
      Table[{ix}[[nx]] -> Permutations[{ix}][[n]][[nx]], {nx, Length[{ix}]}], {nf, lf}], {n, lperm}];
  org = return[1];
  Do[Block[{comp},
    comp = return[n2];
    Do[If[org[[nx1]] == comp[[nx2]], {org[[nx2]] = return[n2][[nx2]], comp[[nx2]] = 0}], {nx1, lf}, {nx2, lf}];
    return[n2] = comp],
   {n2, 2, lperm}];
  org // Total]
Sum[x*Connected[x], {x, Apply[List, Corr[2]]}]
RemoveDummy[Apply[List, %], {v[1], v[2]}]
```



Output:

```
1/2 DF[sc[a1],sc[v[2]]] DF[sc[a2],sc[v[2]]] DF[sc[b1],sc[v[2]]] DF[sc[b2],sc[v[1]]]
DF[sc[v[1]],sc[v[1]]] DF[sc[v[1]],sc[v[2]]] +
 1/2 DF[sc[a1],sc[v[2]]] DF[sc[a2],sc[v[2]]] DF[sc[b1],sc[v[1]]] DF[sc[b2],sc[v[2]]]
DF[sc[v[1]],sc[v[1]]] DF[sc[v[1]],sc[v[2]]] +
 1/2 DF[sc[a1],sc[v[2]]] DF[sc[a2],sc[v[1]]] DF[sc[b1],sc[v[2]]] DF[sc[b2],sc[v[2]]]
DF[sc[v[1]],sc[v[1]]] DF[sc[v[1]],sc[v[2]]] +
 1/2 DF[sc[a1],sc[v[1]]] DF[sc[a2],sc[v[2]]] DF[sc[b1],sc[v[2]]] DF[sc[b2],sc[v[2]]]
DF[sc[v[1]],sc[v[1]]] DF[sc[v[1]],sc[v[2]]] +
 1/2 DF[sc[a1],sc[v[2]]] DF[sc[a2],sc[v[2]]] DF[sc[b1],sc[v[1]]] DF[sc[b2],sc[v[1]]] DF[sc[v[1]],sc[v[2]]]^2 +
 1/2 DF[sc[a1],sc[v[2]]] DF[sc[a2],sc[v[1]]] DF[sc[b1],sc[v[2]]] DF[sc[b2],sc[v[1]]] DF[sc[v[1]],sc[v[2]]]^2 +
 1/2 DF[sc[a1],sc[v[1]]] DF[sc[a2],sc[v[2]]] DF[sc[b1],sc[v[2]]] DF[sc[b2],sc[v[1]]] DF[sc[v[1]],sc[v[2]]]^2
```

Automated Calculation of Feynman Diagrams

Almost fully-automated calculation is possible for simple cases by using publicly available MATHEMATICA packages.

- **FeynRules** : a Mathematica package to calculate Feynman rules.

  A. Alloul, N. D. Christensen, C. Degrande, C. Duhr and B. Fuks, Comput.Phys.Commun. 185 (2014) 2250-2300

- **FeynArts** : a Mathematica package for the generation and visualization of Feynman diagrams and amplitudes.

  T. Hahn, Comput. Phys. Commun., 140, 418-431, 2001

- **FeynCalc** : a Mathematica package for symbolic evaluation of Feynman diagrams and algebraic calculations in quantum field theory and elementary particle physics.

  V. Shtabovenko, R. Mertig and F. Orellana, Comput. Phys. Commun., 256 (2020), 107478, arXiv:2001.04407.
  V. Shtabovenko, R. Mertig and F. Orellana, Comput. Phys. Commun., 207, 432-444, 2016, arXiv:1601.01167.
  R. Mertig, M. Böhm, and A. Denner, Comput. Phys. Commun., 64, 345-359, 1991.

In[1]:=
```
(* :Title: PhiPhi-PhiPhi                                                *)

(*
    This software is covered by the GNU General Public License 3.
    Copyright (C) 1990-2020 Rolf Mertig
    Copyright (C) 1997-2020 Frederik Orellana
    Copyright (C) 2014-2020 Vladyslav Shtabovenko
*)

(* :Summary:  Phi Phi -> Phi Phi, Phi^4, asymptotic limit, 1-loop         *)

(* ------------------------------------------------------------------ *)
```

Phi Phi scattering at 1-loop

Load FeynCalc and the necessary add-ons or other packages

This example uses a custom Phi^4 model created with FeynRules. Please evaluate the file FeynCalc/Examples/FeynRules/Phi4/GenerateModelPhi4.m before running it for the first time.

In[2]:=
```
description="Phi Phi -> Phi Phi, Phi^4, asymptotic limit, 1-loop";
If[ $FrontEnd === Null,
    $FeynCalcStartupMessages = False;
    Print[description];
];
If[ $Notebooks === False,
    $FeynCalcStartupMessages = False
];
$LoadAddOns={"FeynArts"};
<<FeynCalc`
$FAVerbose = 0;

FCCheckVersion[9,3,1];
```

FeynCalc 9.3.1 (stable version). For help, use the documentation center, check out the wiki or visit the forum.

To save your and our time, please check our FAQ for answers to some common FeynCalc questions.

See also the supplied examples. If you use FeynCalc in your research, please cite

• V. Shtabovenko, R. Mertig and F. Orellana, Comput.Phys.Commun. 256 (2020) 107478, arXiv:2001.04407.

• V. Shtabovenko, R. Mertig and F. Orellana, Comput.Phys.Commun. 207 (2016) 432–444, arXiv:1601.01167.

• R. Mertig, M. Böhm, and A. Denner, Comput. Phys. Commun. 64 (1991) 345–359.

FeynArts 3.11 (25 Mar 2022) patched for use with FeynCalc, for documentation see the manual or visit www.feynarts.de.

If you use FeynArts in your research, please cite

• T. Hahn, Comput. Phys. Commun., 140, 418–431, 2001, arXiv:hep–ph/0012260

Configure some options

In[9]:=
```
FAPatch[PatchModelsOnly->True];
```

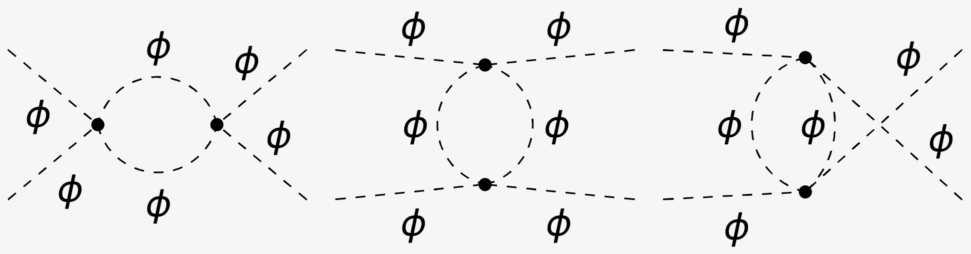
```
Patched 2 FeynArts models.
```

Generate Feynman diagrams

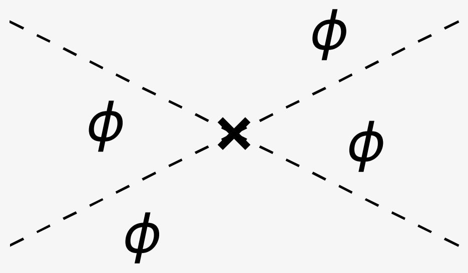
Nicer typesetting

In[10]:=
```
MakeBoxes[p1,TraditionalForm]:="\!\(\*SubscriptBox[\(p\), \(1\)]\)";
MakeBoxes[p2,TraditionalForm]:="\!\(\*SubscriptBox[\(p\), \(2\)]\)";
MakeBoxes[k1,TraditionalForm]:="\!\(\*SubscriptBox[\(k\), \(1\)]\)";
MakeBoxes[k2,TraditionalForm]:="\!\(\*SubscriptBox[\(k\), \(2\)]\)";
```

In[14]:=
```
diags = InsertFields[CreateTopologies[1, 2 -> 2,
        ExcludeTopologies->{WFCorrections}],
        {S[1],S[1]}-> {S[1],S[1]}, InsertionLevel -> {Classes},
        Model -> FileNameJoin[{"Phi4","Phi4"}]];
Paint[diags, ColumnsXRows -> {3, 1}, Numbering -> None,SheetHeader->None,
ImageSize->{512,256}];
```



In[16]:=
```
diagsCT = InsertFields[CreateCTTopologies[1, 2 -> 2,
    ExcludeTopologies->{WFCorrectionCTs}],  {S[1],S[1]}-> {S[1],S[1]},
    InsertionLevel -> {Classes},  Model -> FileNameJoin[{"Phi4","Phi4"}]];
Paint[diagsCT, ColumnsXRows -> {1, 1}, Numbering -> None,SheetHeader->None,
ImageSize->{256,256}];
```



Obtain the amplitude

The 1/(2Pi)^D prefactor is implicit.

In[18]:=
```
amp[0] = FCFAConvert[CreateFeynAmp[diags,PreFactor->1],
    IncomingMomenta->{p1,p2}, OutgoingMomenta->{k1,k2},
    LoopMomenta->{q},ChangeDimension->D,List->False,
    FinalSubstitutions->{Mphi->m}]
```

Out[18]=
$$\frac{1}{2} g^2 \text{FeynAmpDenominator}[\text{PropagatorDenominator}[\text{Momentum}[q, D], m], \text{PropagatorDenominator}[\text{Momentum}[-k1 - k2 + q, D], m]] + \frac{1}{2} g^2 \text{FeynAmpDenominator}[\text{PropagatorDenominator}[\text{Momentum}[q, D], m], \text{PropagatorDenominator}[\text{Momentum}[-k1 + p2 + q, D], m]] + \frac{1}{2} g^2 \text{FeynAmpDenominator}[\text{PropagatorDenominator}[\text{Momentum}[q, D], m], \text{PropagatorDenominator}[\text{Momentum}[-k2 + p2 + q, D], m]]$$

In[19]:=
```
ampCT[0] = FCFAConvert[CreateFeynAmp[diagsCT,PreFactor->1],
    IncomingMomenta->{p1,p2}, OutgoingMomenta->{k1,k2},
    LoopMomenta->{q},ChangeDimension->D,List->False,
    FinalSubstitutions->{Mphi->m,Zg->1+SMP["d_g^MSbar"]}]
```

Out[19]=
$$-i\, g \left(-1 + \text{Zphi}^2 (1 + \text{SMP}[\text{d\_g\^{}MSbar}])\right)$$

Fix the kinematics

For simplicity, let us consider the massless case

In[20]:=
```
FCClearScalarProducts[]
SetMandelstam[s, t, u, p1, p2, -k1, -k2, 0, 0, 0, 0];
```

Calculate the amplitude

In[22]:=
```
amp[1]=amp[0]//ReplaceAll[#,m->0]&//ToPaVe[#,q]&
```

Out[22]=
$$\frac{1}{2} i\, g^2 \pi^2 \text{B0}[s, 0, 0] + \frac{1}{2} i\, g^2 \pi^2 \text{B0}[t, 0, 0] + \frac{1}{2} i\, g^2 \pi^2 \text{B0}[u, 0, 0]$$

The explicit value of the integral can be obtained from Package-X via the FeynHelpers add-on.

```
In[23]:= loopInt={
B0[s_,0,0]:>-(-2 + Log[4*Pi] -
    Log[(-4*Pi*ScaleMu^2)/s])/(16*Pi^4) + SMP["Delta"]/(16*Pi^4)
};
```

```
In[24]:= amp[2]=(amp[1]/.loopInt)//Simplify
```

Out[24]=
$$-\frac{i\, g^2 \left(-6+3\,\text{Log}[4\,\pi]-\text{Log}\left[-\frac{4\,\pi\,\text{ScaleMu}^2}{s}\right]-\text{Log}\left[-\frac{4\,\pi\,\text{ScaleMu}^2}{t}\right]-\text{Log}\left[-\frac{4\,\pi\,\text{ScaleMu}^2}{u}\right]-3\,\text{SMP}[\text{Delta}]\right)}{32\,\pi^2}$$

```
In[25]:= ampFull[0]=Expand[(amp[2]+ampCT[0])/.
    {SMP["d_g^MSbar"] ->(3*g*SMP["Delta"])/(32*Pi^2),Zphi->1}]
```

Out[25]=
$$\frac{3\,i\,g^2}{16\,\pi^2}-\frac{3\,i\,g^2\,\text{Log}[4\,\pi]}{32\,\pi^2}+\frac{i\,g^2\,\text{Log}\left[-\frac{4\,\pi\,\text{ScaleMu}^2}{s}\right]}{32\,\pi^2}+\frac{i\,g^2\,\text{Log}\left[-\frac{4\,\pi\,\text{ScaleMu}^2}{t}\right]}{32\,\pi^2}+\frac{i\,g^2\,\text{Log}\left[-\frac{4\,\pi\,\text{ScaleMu}^2}{u}\right]}{32\,\pi^2}$$

```
In[26]:= FCCompareResults[FreeQ[ampFull[0],SMP["Delta"]],True,
Text->{"\tThe UV divergence is cancelled by the counter-term:",
"CORRECT.","WRONG!"}, Interrupt->{Hold[Quit[1]],Automatic}];
```

The UV divergence is cancelled by the counter–term: CORRECT.

Now let us look at the asymptotic limit where s goes to infinity and t is fixed

```
In[27]:= ampFullAsy[0]=Series[ampFull[0]/.u->-s-t,{s,Infinity,0}]//Normal
```

Out[27]=
$$-\frac{i\left(-6\,g^2+3\,g^2\,\text{Log}[4\,\pi]-g^2\,\text{Log}\left[-\frac{4\,\pi\,\text{ScaleMu}^2}{s}\right]-g^2\,\text{Log}\left[\frac{4\,\pi\,\text{ScaleMu}^2}{s}\right]-g^2\,\text{Log}\left[-\frac{4\,\pi\,\text{ScaleMu}^2}{t}\right]\right)}{32\,\pi^2}$$

The leading order behavior is governed by the log of s

```
In[28]:= ampFullAsy[1]=ampFullAsy[0]//PowerExpand//SelectNotFree2[#,s]&
```

Out[28]= $-\frac{i\, g^2 \operatorname{Log}[s]}{16\, \pi^2}$

Check the final results

```
In[29]:= knownResult = ((-I/16)*g^2*Log[s])/Pi^2;
FCCompareResults[ampFullAsy[1],knownResult,
Text->{"\tCompare to Peskin and Schroeder, An Introduction to QFT, \
Ex 10.4:",
"CORRECT.","WRONG!"}, Interrupt->{Hold[Quit[1]],Automatic}];
Print["\tCPU Time used: ", Round[N[TimeUsed[],4],0.001], " s."];
```

Compare to Peskin and Schroeder, An Introduction to QFT, Ex 10.4: CORRECT.

```
CPU Time used: 2.577 s.
```

Reduction of spin

Particles carry spin (intrinsic angular momentum). When particles carry spin $1/2$, Feynman diagrams can involve linear combinations of products of Pauli matrices. There are three Pauli matrices in 3-dimensional space:

$$\sigma_1 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \sigma_2 = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad \sigma_3 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

They satisfy

$$\{\sigma_i, \sigma_j\} = \sigma_i\sigma_j + \sigma_j\sigma_i = 2\delta_{ij},$$

$$[\sigma_1, \sigma_2] = 2i\sigma_3, \quad [\sigma_2, \sigma_3] = 2i\sigma_1, \quad [\sigma_3, \sigma_1] = 2i\sigma_2.$$

Reduction of spin

Usually we end up with a long chain of product of Pauli matrices, with most pairs of indices summed over spatial dimensions. For example, these appear in hyperfine interactions:

$$\sum_{i,j=1}^{3} [\sigma_i, \sigma_j]\sigma_k[\sigma_i, \sigma_j], \quad \sum_{i,j=1}^{3} [\sigma_i, \sigma_j][\sigma_i, \sigma_j].$$

In principle, we could compute them with explicit σ_1, σ_2, and σ_3.
A better way to compute them is to use only the Clifford algebra $\{\sigma_i, \sigma_j\} = 2\delta_{ij}$, which is useful for generalizing to higher spacetime and matrix dimensions.
For example, the 4-dimensional Dirac gamma matrices in 4 spacetime dimensions appear in quantum electrodynamics and the Standard Model of particle physics, which satisfy $\{\gamma^\mu, \gamma_\nu\} = 2\delta^\mu_\nu$. Hypothetical models often have higher dimensional gamma matrices satisfying the same algebra.

Reduction of spin

By using

$$\sigma_i\sigma_j = 2\delta_{ij} - \sigma_j\sigma_i,$$

we can move σ_i to the right of σ_j until it meets another σ_i. Then we can use

$$\sum_{i=1}^{d-1} \sigma_i\sigma_i = d - 1.$$

$d = 4 =$number of spacetime dimensions.

Reduction of spin

Sample MATHEMATICA code:

```
(* define chain of products of Pauli matrices *)
Unprotect[Dot];
Sigma[a___].Sigma[b___] = Sigma[a, b];
(-Sigma[a___]).Sigma[b___] = -Sigma[a, b];
Sigma[a___].(-Sigma[b___]) = -Sigma[a, b];
(-Sigma[a___]).(-Sigma[b___]) = Sigma[a, b];

(* replacement rules for moving repeated indices closer *)
Clifford = {Sigma[a___, i_, b_, c___, i_,
     d___] :> -Sigma[a, b, i, c, i, d] + 2*Sigma[a, c, b, d],
   Sigma[a___, i_, i_, b___] :> (dim - 1)*Sigma[a, b],
   Sigma[] :> 1};
```

Reduction of spin

Calculation of $\sum_{i,j=1}^{d-1}[\sigma_i,\sigma_j]\sigma_v[\sigma_i,\sigma_j]$:

```
(Sigma[i].Sigma[j] - Sigma[j].Sigma[i]).Sigma[
   v].(Sigma[i].Sigma[k] - Sigma[k].Sigma[i]) // Distribute
% //. Clifford
% //. {k :> j} //. Clifford // Simplify
```

Output:

```
-4 (10 - 7 dim + dim^2) Sigma[v]
```

Reduction of spin

Calculation of $\sum_{i,j=1}^{d-1}[\sigma_i,\sigma_j][\sigma_i,\sigma_j]$:

```
(Sigma[i].Sigma[j] - Sigma[j].Sigma[i]).(Sigma[i].Sigma[k] -
    Sigma[k].Sigma[i]) // Distribute
% //. Clifford
% //. {k :> j} //. Clifford // Simplify
```

Output:

```
-4 (2 - 3 dim + dim^2)
```

Reduction of spin

Calculation of $\sum_{i,j=1}^{d-1}[\sigma_i,\sigma_j]\sigma_5[\sigma_i,\sigma_j]$, where
$\sigma_5 = \sigma_1\sigma_2\sigma_3 - \sigma_1\sigma_3\sigma_2 - \sigma_2\sigma_1\sigma_3 + \sigma_2\sigma_3\sigma_1 + \sigma_3\sigma_1\sigma_2 - \sigma_3\sigma_2\sigma_1$.
In 3 spatial dimensions $(d = 4)$, $\sigma_5 = i3! \times I$.

```
sigma5 = Sum[
   Signature[Permutations[{a, b, c}][[n]]]*
    Apply[Sigma, Permutations[{a, b, c}][[n]]], {n, 3!}];
(Sigma[i].Sigma[j] - Sigma[j].Sigma[i]).sigma5.(Sigma[i].Sigma[k] -
    Sigma[k].Sigma[i]) // Distribute
% //. Clifford
% //. {k :> j} //. Clifford
%/sigma5 // Simplify
```

Output:

```
-4 (50 - 15 dim + dim^2)
```

Reduction of spin

In many cases we also need to compute traces of Pauli matrices. The following identities can be derived by only using the Clifford algebra and $\mathrm{tr}(AB) = \mathrm{tr}(BA)$:

$$\begin{aligned} \mathrm{tr}[\sigma_i\sigma_j] &= \delta_{ij}\mathrm{tr}I, \\ \mathrm{tr}[\sigma_i\sigma_j\sigma_k\sigma_l] &= (\delta_{ij}\delta_{kl} - \delta_{ik}\delta_{jl} + \delta_{il}\delta_{jk})\mathrm{tr}I. \end{aligned}$$

Traces of 6, 8, . . . Pauli matrices can also be computed. These also straightforwardly generalize to higher dimensions.

Reduction of spin

Calculating trace of two Pauli matrices using MATHEMATICA:

```
SetAttributes[KD, Orderless];
Clifford = {TrSigma[b___, a[1], a[nx_],
     c___] :> -TrSigma[b, a[nx], a[1], c] +
     2*TrSigma[b, c]*KD[a[nx], a[1]]};
TrCycle = {TrSigma[b___, a[1]] :> TrSigma[a[1], b]};
```

```
TrSigma[a[1], a[2]] //. Clifford
Solve[TrSigma[a[1], a[2]] == % //. TrCycle, TrSigma[a[1], a[2]
```

Output:

```
{{TrSigma[a[1], a[2]] -> KD[a[1], a[2]] TrSigma[]}}
```

Reduction of spin

Trace of four Pauli matrices from MATHEMATICA:

```
Rule2 = TrSigma[a1_, a2_] -> KD[a1, a2] TrSigma[];
TrSigma[a[1], a[2], a[3], a[4]] //. Clifford
Solve[TrSigma[a[1], a[2], a[3], a[4]] == % //. TrCycle,
   TrSigma[a[1], a[2], a[3], a[4]]] //. Rule2 // Simplify
```

Output:

```
{{TrSigma[a[1], a[2], a[3],
    a[4]] -> (KD[a[1], a[4]] KD[a[2], a[3]] -
      KD[a[1], a[3]] KD[a[2], a[4]] +
      KD[a[1], a[2]] KD[a[3], a[4]]) TrSigma[]}}
```

Reduction of spin

Trace of six Pauli matrices from MATHEMATICA:

```
Rule4 = TrSigma[a1_, a2_, a3_, a4_] -> (KD[a1, a4] KD[a2, a3] - KD[a1, a3] KD[a2, a4] +
      KD[a1, a2] KD[a3, a4]) TrSigma[];
TrSigma[a[1], a[2], a[3], a[4], a[5], a[6]] //. Clifford
Solve[TrSigma[a[1], a[2], a[3], a[4], a[5], a[6]] == % //. TrCycle,
    TrSigma[a[1], a[2], a[3], a[4], a[5], a[6]]] //. Rule4 //.  Rule2 // Simplify
```

Output:

```
{{TrSigma[a[1], a[2], a[3], a[4], a[5],
    a[6]] -> (KD[a[1], a[6]] (KD[a[2], a[5]] KD[a[3], a[4]] -
         KD[a[2], a[4]] KD[a[3], a[5]] + KD[a[2], a[3]] KD[a[4], a[5]]) -
      KD[a[1], a[5]] (KD[a[2], a[6]] KD[a[3], a[4]] -
         KD[a[2], a[4]] KD[a[3], a[6]] + KD[a[2], a[3]] KD[a[4], a[6]]) +
      KD[a[1], a[4]] (KD[a[2], a[6]] KD[a[3], a[5]] -
         KD[a[2], a[5]] KD[a[3], a[6]] + KD[a[2], a[3]] KD[a[5], a[6]]) -
      KD[a[1], a[3]] (KD[a[2], a[6]] KD[a[4], a[5]] -
         KD[a[2], a[5]] KD[a[4], a[6]] + KD[a[2], a[4]] KD[a[5], a[6]]) +
      KD[a[1], a[2]] (KD[a[3], a[6]] KD[a[4], a[5]] -
         KD[a[3], a[5]] KD[a[4], a[6]] + KD[a[3], a[4]] KD[a[5], a[6]])) TrSigma[]}}
```

Trace of eight Pauli matrices from MATHEMATICA:

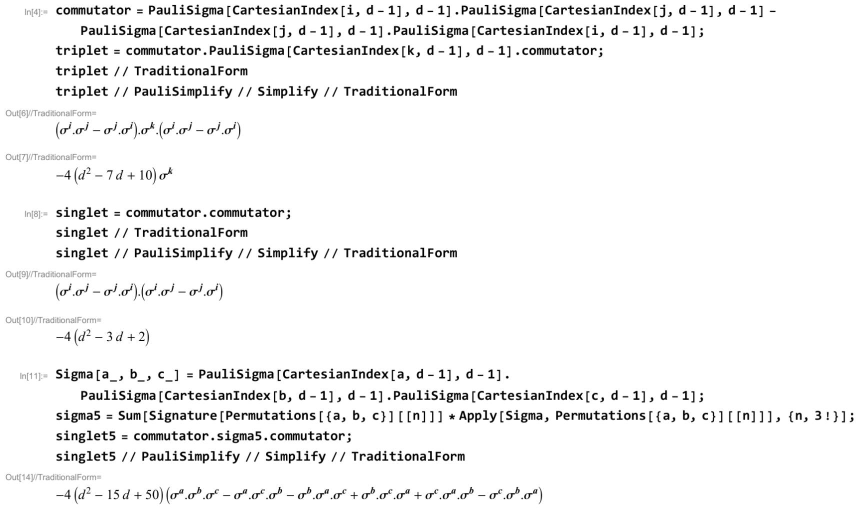
```
Rule6 = TrSigma[a1_, a2_, a3_, a4_, a5_, a6_] -> (KD[a1, a6] (KD[a2, a5] KD[a3, a4] - KD[a2, a4] KD[a3, a5] + KD[a2, a3] KD[a4, a5]) -
      KD[a1, a5] (KD[a2, a6] KD[a3, a4] - KD[a2, a4] KD[a3, a6] + KD[a2, a3] KD[a4, a6]) +
      KD[a1, a4] (KD[a2, a6] KD[a3, a5] - KD[a2, a5] KD[a3, a6] + KD[a2, a3] KD[a5, a6]) -
      KD[a1, a3] (KD[a2, a6] KD[a4, a5] - KD[a2, a5] KD[a4, a6] + KD[a2, a4] KD[a5, a6]) +
      KD[a1, a2] (KD[a3, a6] KD[a4, a5] - KD[a3, a5] KD[a4, a6] + KD[a3, a4] KD[a5, a6])) TrSigma[];
TrSigma[a[1], a[2], a[3], a[4], a[5], a[6], a[7], a[8]] //. Clifford
Solve[TrSigma[a[1], a[2], a[3], a[4], a[5], a[6], a[7], a[8]] == % //. TrCycle,
     TrSigma[a[1], a[2], a[3], a[4], a[5], a[6], a[7], a[8]]] //.  Rule6 //. Rule4 //. Rule2 // Simplify

{{TrSigma[a[1], a[2], a[3], a[4], a[5], a[6], a[7],
    a[8]] -> (KD[a[1], a[8]] (KD[a[2], a[7]] (KD[a[3], a[6]] KD[a[4], a[5]] - KD[a[3], a[5]] KD[a[4], a[6]] + KD[a[3], a[4]] KD[a[5],
         KD[a[2], a[6]] (KD[a[3], a[7]] KD[a[4], a[5]] - KD[a[3], a[5]] KD[a[4], a[7]] + KD[a[3], a[4]] KD[a[5], a[7]]) +
         KD[a[2], a[5]] (KD[a[3], a[7]] KD[a[4], a[6]] - KD[a[3], a[6]] KD[a[4], a[7]] + KD[a[3], a[4]] KD[a[6], a[7]]) -
         KD[a[2], a[4]] (KD[a[3], a[7]] KD[a[5], a[6]] - KD[a[3], a[6]] KD[a[5], a[7]] + KD[a[3], a[5]] KD[a[6], a[7]]) +
         KD[a[2], a[3]] (KD[a[4], a[7]] KD[a[5], a[6]] - KD[a[4], a[6]] KD[a[5], a[7]] + KD[a[4], a[5]] KD[a[6], a[7]])) -
      KD[a[1], a[7]] (KD[a[2], a[8]] (KD[a[3], a[6]] KD[a[4], a[5]] - KD[a[3], a[5]] KD[a[4], a[6]] + KD[a[3], a[4]] KD[a[5], a[6]]) -
         KD[a[2], a[6]] (KD[a[3], a[8]] KD[a[4], a[5]] - KD[a[3], a[5]] KD[a[4], a[8]] + KD[a[3], a[4]] KD[a[5], a[8]]) +
         KD[a[2], a[5]] (KD[a[3], a[8]] KD[a[4], a[6]] - KD[a[3], a[6]] KD[a[4], a[8]] + KD[a[3], a[4]] KD[a[6], a[8]]) -
         KD[a[2], a[4]] (KD[a[3], a[8]] KD[a[5], a[6]] - KD[a[3], a[6]] KD[a[5], a[8]] + KD[a[3], a[5]] KD[a[6], a[8]]) +
         KD[a[2], a[3]] (KD[a[4], a[8]] KD[a[5], a[6]] - KD[a[4], a[6]] KD[a[5], a[8]] + KD[a[4], a[5]] KD[a[6], a[8]])) +
      KD[a[1], a[6]] (KD[a[2], a[8]] (KD[a[3], a[7]] KD[a[4], a[5]] - KD[a[3], a[5]] KD[a[4], a[7]] + KD[a[3], a[4]] KD[a[5], a[7]]) -
         KD[a[2], a[7]] (KD[a[3], a[8]] KD[a[4], a[5]] - KD[a[3], a[5]] KD[a[4], a[8]] + KD[a[3], a[4]] KD[a[5], a[8]]) +
         KD[a[2], a[5]] (KD[a[3], a[8]] KD[a[4], a[7]] - KD[a[3], a[7]] KD[a[4], a[8]] + KD[a[3], a[4]] KD[a[7], a[8]]) -
         KD[a[2], a[4]] (KD[a[3], a[8]] KD[a[5], a[7]] - KD[a[3], a[7]] KD[a[5], a[8]] + KD[a[3], a[5]] KD[a[7], a[8]]) +
         KD[a[2], a[3]] (KD[a[4], a[8]] KD[a[5], a[7]] - KD[a[4], a[7]] KD[a[5], a[8]] + KD[a[4], a[5]] KD[a[7], a[8]])) -
      KD[a[1], a[5]] (KD[a[2], a[8]] (KD[a[3], a[7]] KD[a[4], a[6]] - KD[a[3], a[6]] KD[a[4], a[7]] + KD[a[3], a[4]] KD[a[6], a[7]]) -
         KD[a[2], a[7]] (KD[a[3], a[8]] KD[a[4], a[6]] - KD[a[3], a[6]] KD[a[4], a[8]] + KD[a[3], a[4]] KD[a[6], a[8]]) +
         KD[a[2], a[6]] (KD[a[3], a[8]] KD[a[4], a[7]] - KD[a[3], a[7]] KD[a[4], a[8]] + KD[a[3], a[4]] KD[a[7], a[8]]) -
         KD[a[2], a[4]] (KD[a[3], a[8]] KD[a[6], a[7]] - KD[a[3], a[7]] KD[a[6], a[8]] + KD[a[3], a[6]] KD[a[7], a[8]]) +
         KD[a[2], a[3]] (KD[a[4], a[8]] KD[a[6], a[7]] - KD[a[4], a[7]] KD[a[6], a[8]] + KD[a[4], a[6]] KD[a[7], a[8]])) +
      KD[a[1], a[4]] (KD[a[2], a[8]] (KD[a[3], a[7]] KD[a[5], a[6]] - KD[a[3], a[6]] KD[a[5], a[7]] + KD[a[3], a[5]] KD[a[6], a[7]]) -
         KD[a[2], a[7]] (KD[a[3], a[8]] KD[a[5], a[6]] - KD[a[3], a[6]] KD[a[5], a[8]] + KD[a[3], a[5]] KD[a[6], a[8]]) +
         KD[a[2], a[6]] (KD[a[3], a[8]] KD[a[5], a[7]] - KD[a[3], a[7]] KD[a[5], a[8]] + KD[a[3], a[5]] KD[a[7], a[8]]) -
         KD[a[2], a[5]] (KD[a[3], a[8]] KD[a[6], a[7]] - KD[a[3], a[7]] KD[a[6], a[8]] + KD[a[3], a[6]] KD[a[7], a[8]]) +
         KD[a[2], a[3]] (KD[a[5], a[8]] KD[a[6], a[7]] - KD[a[5], a[7]] KD[a[6], a[8]] + KD[a[5], a[6]] KD[a[7], a[8]])) -
      KD[a[1], a[3]] (KD[a[2], a[8]] (KD[a[4], a[7]] KD[a[5], a[6]] - KD[a[4], a[6]] KD[a[5], a[7]] + KD[a[4], a[5]] KD[a[6], a[7]]) -
         KD[a[2], a[7]] (KD[a[4], a[8]] KD[a[5], a[6]] - KD[a[4], a[6]] KD[a[5], a[8]] + KD[a[4], a[5]] KD[a[6], a[8]]) +
         KD[a[2], a[6]] (KD[a[4], a[8]] KD[a[5], a[7]] - KD[a[4], a[7]] KD[a[5], a[8]] + KD[a[4], a[5]] KD[a[7], a[8]]) -
         KD[a[2], a[5]] (KD[a[4], a[8]] KD[a[6], a[7]] - KD[a[4], a[7]] KD[a[6], a[8]] + KD[a[4], a[6]] KD[a[7], a[8]]) +
         KD[a[2], a[4]] (KD[a[5], a[8]] KD[a[6], a[7]] - KD[a[5], a[7]] KD[a[6], a[8]] + KD[a[5], a[6]] KD[a[7], a[8]])) +
      KD[a[1], a[2]] (KD[a[3], a[8]] (KD[a[4], a[7]] KD[a[5], a[6]] - KD[a[4], a[6]] KD[a[5], a[7]] + KD[a[4], a[5]] KD[a[6], a[7]]) -
         KD[a[3], a[7]] (KD[a[4], a[8]] KD[a[5], a[6]] - KD[a[4], a[6]] KD[a[5], a[8]] + KD[a[4], a[5]] KD[a[6], a[8]]) +
         KD[a[3], a[6]] (KD[a[4], a[8]] KD[a[5], a[7]] - KD[a[4], a[7]] KD[a[5], a[8]] + KD[a[4], a[5]] KD[a[7], a[8]]) -
         KD[a[3], a[5]] (KD[a[4], a[8]] KD[a[6], a[7]] - KD[a[4], a[7]] KD[a[6], a[8]] + KD[a[4], a[6]] KD[a[7], a[8]]) +
         KD[a[3], a[4]] (KD[a[5], a[8]] KD[a[6], a[7]] - KD[a[5], a[7]] KD[a[6], a[8]] + KD[a[5], a[6]] KD[a[7], a[8]]))) TrSigma[]}}
```

Automated Spin Algebra

Almost fully-automated calculation is possible by using **FeynCalc** and **FeynOnium** MATHEMATICA packages.

N. Brambilla, H. S. Chung, V. Shtabovenko and A. Vairo, JHEP 11 (2020) 130



```
In[4]:= commutator = PauliSigma[CartesianIndex[i, d - 1], d - 1].PauliSigma[CartesianIndex[j, d - 1], d - 1] -
           PauliSigma[CartesianIndex[j, d - 1], d - 1].PauliSigma[CartesianIndex[i, d - 1], d - 1];
        triplet = commutator.PauliSigma[CartesianIndex[k, d - 1], d - 1].commutator;
        triplet // TraditionalForm
        triplet // PauliSimplify // Simplify // TraditionalForm
```

Out[6]//TraditionalForm=

$$(\sigma^i.\sigma^j - \sigma^j.\sigma^i).\sigma^k.(\sigma^i.\sigma^j - \sigma^j.\sigma^i)$$

Out[7]//TraditionalForm=

$$-4\left(d^2 - 7\,d + 10\right)\sigma^k$$

```
In[8]:= singlet = commutator.commutator;
        singlet // TraditionalForm
        singlet // PauliSimplify // Simplify // TraditionalForm
```

Out[9]//TraditionalForm=

$$(\sigma^i.\sigma^j - \sigma^j.\sigma^i).(\sigma^i.\sigma^j - \sigma^j.\sigma^i)$$

Out[10]//TraditionalForm=

$$-4\left(d^2 - 3\,d + 2\right)$$

```
In[11]:= Sigma[a_, b_, c_] = PauliSigma[CartesianIndex[a, d - 1], d - 1].
            PauliSigma[CartesianIndex[b, d - 1], d - 1].PauliSigma[CartesianIndex[c, d - 1], d - 1];
         sigma5 = Sum[Signature[Permutations[{a, b, c}][[n]]] * Apply[Sigma, Permutations[{a, b, c}][[n]]], {n, 3!}];
         singlet5 = commutator.sigma5.commutator;
         singlet5 // PauliSimplify // Simplify // TraditionalForm
```

Out[14]//TraditionalForm=

$$-4\left(d^2 - 15\,d + 50\right)\left(\sigma^a.\sigma^b.\sigma^c - \sigma^a.\sigma^c.\sigma^b - \sigma^b.\sigma^a.\sigma^c + \sigma^b.\sigma^c.\sigma^a + \sigma^c.\sigma^a.\sigma^b - \sigma^c.\sigma^b.\sigma^a\right)$$

Reduction of Feynman Integrals

Feynman diagrams involve integrals over momentum for every closed loop. Generic form of Feynman integrals:

$$\int \frac{d^dk}{(2\pi)^d} \frac{\text{Polynomials of } k}{[(k+p_1)^2-m^2][(k+p_2)^2-m^2]\cdots[(k+p_N)^2-m^2]}$$

In general, these can be reduced to linear combinations of

$$I_{n_1 n_2 \cdots n_N} = \int \frac{d^dk}{(2\pi)^d} \frac{1}{[(k+p_1)^2-m^2]^{n_1}[(k+p_2)^2-m^2]^{n_2}\cdots[(k+p_N)^2-m^2]^{n_N}}$$

Usually, we do not have to explicitly compute every $I_{n_1 n_2 \cdots n_N}$ that we encounter, because we can derive recursion relations.

Reduction of Feynman Integrals

Real-world example:

$$\frac{(d-2)I_{1,-1,1}}{8(d-1)m^2} + \frac{(2-d)I_{1,0,0}}{4(d-1)m^2} + \frac{(d-2)I_{1,1,-1}}{8(d-1)m^2} + (d-2)I_{0,1,1}$$
$$+\frac{(2-d)I_{1,0,1}}{d-1} + \frac{(2-d)I_{1,1,0}}{d-1} - 4m^2 I_{1,1,1},$$
$$I_{abc} = \int \frac{d^d k}{(2\pi)^d} \frac{1}{(k^2)^a[(k+p)^2-m^2]^b[(k-p)^2-m^2]^c}, \quad p^2 = m^2.$$

Reduction of Feynman Integrals

Recurrence relations

$$
\begin{aligned}
I_{abc} &= \frac{1}{2}(I_{a+1,b-1,c} + I_{a+1,b,c-1}) \\
I_{ab-c} &= 2I_{a-1,b,-c+1} - I_{a,b-1,-c+1}, \quad (b, c > 0) \\
I_{a,b,0} &= \frac{d-2a-b-1}{b-1} I_{a+1,b-1,0}, \quad (a > 0, b > 1) \\
I_{a,1,0} &= \frac{d-a-1}{2m^2(d-2a-1)} I_{a-1,1,0}, \quad (a > 0)
\end{aligned}
$$

and $I_{abc} = I_{acb}$.

Reduction of Feynman Integrals

MATHEMATICA code:

```
reduction = {
   IA[a_, b_, c_] :> IA[a, c, b] /; c > b,
   IA[a_, b_, c_] :> 1/2 (IA[a + 1, b - 1, c] + IA[a + 1, b, c - 1]) /; b > 0 && c > 0,
   IA[a_, b_, c_] :> 2 IA[a - 1, b, c + 1] - IA[a, b - 1, c + 1] /; b > 0 && c < 0,
   IA[a_, b_, 0] :> (d - 2 a - b - 1)/(b - 1) IA[a + 1, b - 1, 0] /; a > 0 && b > 1,
   IA[a_, 1, 0] :> (d - a - 1)/((2 m^2) (d - 2 a - 1)) IA[a - 1, 1, 0] /; a > 0
};

(-2 + d) IA[0, 1, 1] + ((-2 + d) IA[1, -1, 1])/(
   8 (-1 + d) m^2) + ((2 - d) IA[1, 0, 0])/(
   4 (-1 + d) m^2) + ((2 - d) IA[1, 0, 1])/(-1 +
    d) + ((-2 + d) IA[1, 1, -1])/(
   8 (-1 + d) m^2) + ((2 - d) IA[1, 1, 0])/(-1 + d) -
   4 m^2 IA[1, 1, 1] //. reduction // FullSimplify
```

Output:

```
((-2 + d) ((-7 + d) (-1 + d) IA[0, 1, 0] - (-5 + d) IA[1, 0, 0]))/(2 (-5 + d) (-1 + d) m^2)
```

We only need to compute two integrals I_{010} and I_{100}.

Asymptotic Expansion of Feynman Integrals

Feynman integrals are often ill defined. To work with well-defined integrals, we modify them as functions of ϵ, in a way that we recover the ill-defined integral at $\epsilon = 0$. Integrals modified in this way will diverge as $\epsilon \to 0$. Well-defined calculations are organized so that individual divergences cancel in final results, although individual divergences are usually unavoidable.

A simple example :

$$\int_0^\infty dx \int_0^\infty dy \frac{1}{(x+y)^{1-\epsilon}} e^{-xy/(x+y)} = -\frac{2^{1+2\epsilon}\sqrt{\pi}\Gamma(1-\epsilon)\Gamma(\epsilon)}{\Gamma(\frac{1}{2}-\epsilon)}$$

which is well defined for $-1/2 < \epsilon < 0$. This integral is ill defined at $\epsilon = 0$, as can be seen from the Laurent series

$$\Gamma(\epsilon) = \frac{1}{\epsilon} - \gamma + O(\epsilon)$$

Asymptotic Expansion of Feynman Integrals

Since divergences like $1/\epsilon$ will cancel in the final results, after which we can set $\epsilon \to 0$, we often only need the expansion to a finite order:

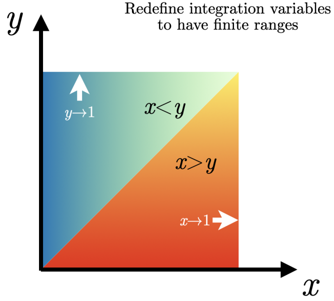
$$\int_0^\infty dx \int_0^\infty dy \frac{1}{(x+y)^{1-\epsilon}} e^{-xy/(x+y)} = -\frac{2}{\epsilon} + 2\gamma + O(\epsilon)$$

These kinds of integrals can become very difficult to compute when they have more integration variables and are less symmetric.
There is a systematic way to organize the integrand so that we have a divergent part that contains the $1/\epsilon$ term and a finite integral that can be computed at $\epsilon = 0$.
We show one example called sector decomposition, which can be easily implemented in MATHEMATICA.
T. Binoth and G. Heinrich, Nucl. Phys. B 585 (2000) 741
G. Heinrich, Sector Decomposition, 2008, Int.J.Mod.Phys.A23,
S. Borowka, G. Heinrich, S. P. Jones, M. Kerner, J. Schlenk, T. Zirke, SecDec-3.0: numerical evaluation of multi-scale integrals beyond one loop, 2015, Comput.Phys.Comm.196

Asymptotic Expansion of Feynman Integrals

A schematic view of sector decomposition in two dimensions:

- Divide integration region into sectors $x > y$ and $y > x$ to avoid double counting.
- Identify singularities in the integrand at boundaries of integration region.
- Compute $1/\epsilon$ terms by series expanding integrands at the boundaries.
- Compute remaining finite terms by subtracting the divergent pieces from the integrand and setting $\epsilon = 0$.



The $x \to 1$ and $y \to 1$ behaviors are especially simple and directly give the $1/\epsilon$ term

$$2\int_0^1 \frac{dy}{(1-y)^{1+\epsilon}} \int_0^1 dx \frac{e^{-x/(1-x)}}{(1-x)^2} = -\frac{2}{\epsilon}$$

Asymptotic Expansion of Feynman Integrals

MATHEMATICA implementation

```
intg[x_, y_] = 1/(x + y)^(1 - ep) Exp[-x y/(x + y)];
intgnorm[x_, y_] = 1/(1 - x)^2*1/(1 - y)^2 intg[x/(1 - x), y/(1 - y)] // Simplify;

sectors = Permutations[{x1, x2}];
Do[
  Block[{},
   SECfuncUnnorm[x1_, x2_] = Apply[intgnorm, sectors[[sec]]];
   SECfunc[x1_, x2_] = SECfuncUnnorm[x1 x2, x2]*x2;
   divform = Series[SECfunc[x1, x2], {x2, 1, -1}] // Normal;
   divfac = Simplify[divform/(divform //. {x2 :> 0}), Assumptions -> x1 > 0 && x2 > 0];
   divcoef = Series[SECfunc[x1, x2]/divfac // Simplify, {x2, 1, 0}, Assumptions -> x1 > 0 && x2 > 0] // Normal /
   poles[sec] = Normal[Integrate[divfac, {x2, 0, 1}]]* Normal[Integrate[divcoef, {x1, 0, 1}]];
   SECsubtract[x1_, x2_] = divfac*divcoef;
   SECfinite[sec][x1_, x2_] = SECfunc[x1, x2] - SECsubtract[x1, x2] //. {ep :> 0};
   ], {sec, Length[sectors]}];


Sum[poles[sec], {sec, Length[sectors]}] +
   Integrate[ Sum[SECfinite[sec][x1, x2], {sec, Length[sectors]}], {x1, 0,
     1}, {x2, 0, 1}] // Series[#, {ep, 0, 0}] & // Normal
```

Output:

```
-(2/ep) + 2 EulerGamma
```

Asymptotic Expansion of Feynman Integrals

More complicated integral, (almost) the same implementation

```
intg[x_, y_, s_] = 1/(x + y + 2 s)^(3 - ep) Exp[(s^2 - x y)/(2 s + x + y)] // Simplify;
intgnorm[x_, y_, s_] = t/(1 - x)^2*t/(1 - y)^2*t intg[t x/(1 - x), t y/(1 - y), t s] // Simplify;

sectors = Permutations[{x1, x2, x3}];
Do[
  Block[{},
   SECfuncUnnorm[x1_, x2_, x3_] = Apply[intgnorm, sectors[[sec]]];
   SECfunc[x1_, x2_, x3_] = SECfuncUnnorm[x1 x2 x3, x2 x3, x3]*x2*x3^2;
   divform = Series[SECfunc[x1, x2, x3], {x3, 0, -1}] // Normal;
   divfac = Simplify[divform/(divform //. {x3 :> 1}), Assumptions -> x1 > 0 && x2 > 0 && x3 > 0];
   divcoef = Series[SECfunc[x1, x2, x3]/divfac // Simplify, {x3, 0, 0}, Assumptions -> x1 > 0 && x2 > 0 && x3 >
   poles[sec] = Normal[Integrate[divfac, {x3, 0, 1}]]* Normal[Integrate[divcoef, {x1, 0, 1}, {x2, 0, 1}]];
   SECsubtract[x1_, x2_, x3_] = divfac*divcoef;
   SECfinite[sec][x1_, x2_, x3_] = SECfunc[x1, x2, x3] - SECsubtract[x1, x2, x3] //. {ep :> 0};
   ], {sec, Length[sectors]}];

Sum[poles[sec], {sec, Length[sectors]}] // Series[#, {ep, 0, 0}] & // Normal
```

Output:

```
1/(4 ep) + 1/8 (3 - 4 Log[3] + 2 Log[4] + 2 Log[t])
```

The finite part is a complicated integral that is difficult to be computed exactly, but can be computed numerically. Its integrand can be printed with

```
Sum[SECfinite[sec][x1, x2, x3], {sec, Length[sectors]}] // Simplify
```

Sector Decomposition

- **SecDec** and **pySecDec** : fully numerical calculations are available for standard Feynman integrals in MATHEMATICA and python languages.
  S. Borowka, G. Heinrich, S. P. Jones, M. Kerner, J. Schlenk, T. Zirke, Comput. Phys. Comm. 196 (2015) 470
  S. Borowka, G. Heinrich, S. Jahn, S. P. Jones, M. Kerner, J. Schlenk, T. Zirke, Comput.Phys.Comm. 222 (2018)
- **FIESTA** : Feynman Integral Evaluation by a Sector decomposiTion Approach is another MATHEMATICA package for numerical evaluation of Feynman integrals.

For analytical calculations or nonstandard Feynman integrals, custom codes can be written in MATHEMATICA without much complication.

Conclusions

- MATHEMATICA is a powerful tool for particle physics theory research. There are many publicly available packages that can be made to work together, and custom codes can be written without much effort.
- Capability of MATHEMATICA ranges from assisting or replacing pen&paper calculations to almost fully automatically calculating a whole scattering process.

In[1]:=
```
(* :Title: ElAel-MuAmu                                                  *)

(*
   This software is covered by the GNU General Public License 3.
   Copyright (C) 1990-2020 Rolf Mertig
   Copyright (C) 1997-2020 Frederik Orellana
   Copyright (C) 2014-2020 Vladyslav Shtabovenko
*)

(* :Summary:  El Ael -> Mu Amu, QED, Born-virtual, 1-loop               *)

(* ------------------------------------------------------------------ *)
```

Muon production

Load FeynCalc and the necessary add-ons or other packages

In[2]:=

```
description="El Ael -> Mu Amu, QED, Born-virtual, 1-loop";
If[ $FrontEnd === Null,
    $FeynCalcStartupMessages = False;
    Print[description];
];
If[ $Notebooks === False,
    $FeynCalcStartupMessages = False
];
$LoadAddOns={"FeynArts"};
<<FeynCalc`
$FAVerbose = 0;

FCCheckVersion[9,3,1];
```

FeynCalc 9.3.1 (stable version). For help, use the documentation center, check out the wiki or visit the forum.

To save your and our time, please check our FAQ for answers to some common FeynCalc questions.

See also the supplied examples. If you use FeynCalc in your research, please cite

- V. Shtabovenko, R. Mertig and F. Orellana, Comput.Phys.Commun. 256 (2020) 107478, arXiv:2001.04407.
- V. Shtabovenko, R. Mertig and F. Orellana, Comput.Phys.Commun. 207 (2016) 432–444, arXiv:1601.01167.
- R. Mertig, M. Böhm, and A. Denner, Comput. Phys. Commun. 64 (1991) 345–359.

FeynArts 3.11 (25 Mar 2022) patched for use with FeynCalc, for documentation see the manual or visit www.feynarts.de.

If you use FeynArts in your research, please cite

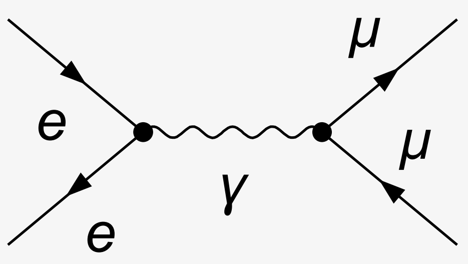
- T. Hahn, Comput. Phys. Commun., 140, 418–431, 2001, arXiv:hep–ph/0012260

Generate Feynman diagrams

Nicer typesetting

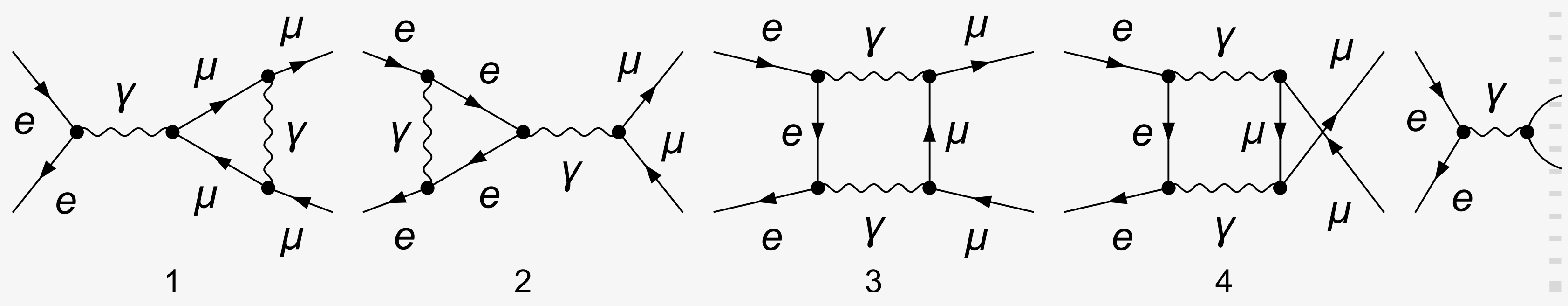
```
In[9]:= MakeBoxes[p1,TraditionalForm]:="\!\(\*SubscriptBox[\(p\), \(1\)]\)";
        MakeBoxes[p2,TraditionalForm]:="\!\(\*SubscriptBox[\(p\), \(2\)]\)";
        MakeBoxes[k1,TraditionalForm]:="\!\(\*SubscriptBox[\(k\), \(1\)]\)";
        MakeBoxes[k2,TraditionalForm]:="\!\(\*SubscriptBox[\(k\), \(2\)]\)";
```

In[13]:=
```
diagsTree=InsertFields[CreateTopologies[0, 2 -> 2,
    ExcludeTopologies->{Tadpoles,WFCorrections}], {F[2, {1}], -F[2, {1}]} ->
    {F[2,{2}], -F[2, {2}]}, InsertionLevel -> {Particles},
    Restrictions->QEDOnly,ExcludeParticles->{F[1|3|4,_],F[2,{3}]}];
Paint[diagsTree, ColumnsXRows -> {2, 1}, Numbering -> Simple,
    SheetHeader->None,ImageSize->{1024,256}];
```



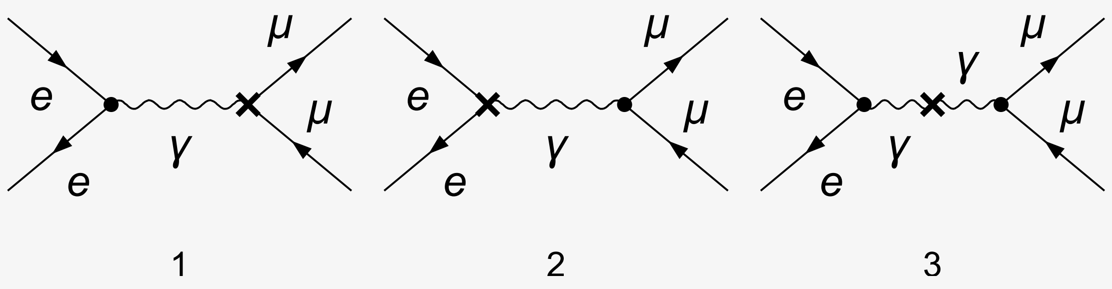
1

In[15]:=
```
diagsLoop=InsertFields[CreateTopologies[1, 2 -> 2,
    ExcludeTopologies->{Tadpoles,WFCorrections}], {F[2, {1}], -F[2, {1}]} ->
    {F[2,{2}], -F[2, {2}]}, InsertionLevel -> {Particles},
    Restrictions->QEDOnly,ExcludeParticles->{F[1|3|4,_],F[2,{3}]}];
Paint[DiagramExtract[diagsLoop,1..5], ColumnsXRows -> {5, 1}, Numbering -> Simple,
    SheetHeader->None,ImageSize->{1024,196}];
```



In[17]:=
```
diagsLoopCT=InsertFields[CreateCTTopologies[1, 2 -> 2,ExcludeTopologies->{Tadpoles,WFCorrectionCTs}],
        {F[2, {1}], -F[2, {1}]} ->{F[2,{2}], -F[2, {2}]}, InsertionLevel -> {Particles},
        Restrictions->QEDOnly,ExcludeParticles->{F[1|3|4,_],F[2,{3}]}];

Paint[diagsLoopCT, ColumnsXRows -> {3, 1}, Numbering -> Simple,
    SheetHeader->None,ImageSize->{1024,196}];
```



Obtain the amplitudes

In[19]:=
```
ampLoopCT[0]=FCFAConvert[CreateFeynAmp[diagsLoopCT,Truncated -> False,PreFactor->1]//.
{(h:dZfL1|dZfR1)[z__]:>dZf1[z],Conjugate[(h:dZfL1|dZfR1)[z__]]:>dZf1[z],dZZA1->0},
IncomingMomenta->{p1,p2},OutgoingMomenta->{k1,k2},LoopMomenta->{l},ChangeDimension->D,
DropSumOver->True,UndoChiralSplittings->True,SMP->True,
FinalSubstitutions->{SMP["m_e"]->0,SMP["m_mu"]->0}];
```

In[20]:=
```
ampLoop[0]=FCFAConvert[CreateFeynAmp[DiagramExtract[diagsLoop,1..5],
    Truncated -> False,PreFactor->1],IncomingMomenta->{p1,p2},OutgoingMomenta->{k1,k2},
    LoopMomenta->{q},ChangeDimension->D,DropSumOver->True,UndoChiralSplittings->True,
    SMP->True,FinalSubstitutions->{SMP["m_e"]->0,SMP["m_mu"]->0}];
```

In[21]:=
```
ampTree[0]=FCFAConvert[CreateFeynAmp[diagsTree,Truncated -> False,PreFactor->1],
    IncomingMomenta->{p1,p2},OutgoingMomenta->{k1,k2},
    ChangeDimension->D,DropSumOver->True,UndoChiralSplittings->True,
    SMP->True, FinalSubstitutions->{SMP["m_e"]->0,SMP["m_mu"]->0}];
```

Fix the kinematics

In[22]:=
```
FCClearScalarProducts[];
SetMandelstam[s, t, u, p1, p2, -k1, -k2, 0, 0, 0, 0];
```

Evaluate the amplitudes

In[24]:=
```
$KeepLogDivergentScalelessIntegrals=True;
```

In[25]:=
```
ampLoop[1]=
    (FCTraceFactor/@DotSimplify[#,Expanding->False]&/@Join[ampLoop[0][[1;;4]],Nf ampLoop[0][[5;;5]]]);
```

In[26]:=
```
ampTree[1]=
    (FCTraceFactor/@DotSimplify[#,Expanding->False]&/@ampTree[0]);
```

In[27]:=
```
ampLoopCT[1]=
    (FCTraceFactor/@DotSimplify[#,Expanding->False]&/@ampLoopCT[0]);
```

In[28]:=
```
evalFuSimple[ex_]:=ex//Contract//DiracSimplify//TID[#,q,ToPaVe->True]&//
    DiracSimplify//Contract//ReplaceAll[#,(h:A0|B0|C0|D0)[x__]:>
    TrickMandelstam[h[x],{s,t,u,0}]]&//
    FeynAmpDenominatorExplicit//Collect2[#,{A0,B0,C0,D0},
    Factoring->Function[x,Factor2[TrickMandelstam[x,{s,t,u,0}]]]]&;
```

```
In[29]:= (*about 50 seconds*)
         AbsoluteTiming[ampLoop[2]=evalFuSimple/@ampLoop[1];]

Out[29]= {26.0554, Null}

In[30]:= ampTree[2]=(Total[ampTree[1]]//Contract//DiracSimplify)//FeynAmpDenominatorExplicit//
             FCCanonicalizeDummyIndices[#,LorentzIndexNames->{mu}]&

Out[30]= -(1/s) i Spinor[Momentum[k1, D], 0, 1].DiracGamma[LorentzIndex[mu, D], D].Spinor[-Momentum[k2, D], 0, 1] ×
          Spinor[-Momentum[p2, D], 0, 1].DiracGamma[LorentzIndex[mu, D], D].Spinor[Momentum[p1, D], 0, 1] SMP[e]^2
```

Obtain the Born-virtual interference term

```
In[31]:= (*about 3 seconds*)
         AbsoluteTiming[bornVirtualUnrenormalized[0]=
             Collect2[Total[ampLoop[2]],Spinor,LorentzIndex,IsolateNames->KK] *
             ComplexConjugate[ampTree[2]]//
             FermionSpinSum[#,ExtraFactor->1/2^2]&//DiracSimplify//
             FRH//TrickMandelstam[#,{s,t,u,0}]&//Collect2[#,B0,C0,D0]&;]

Out[31]= {2.83058, Null}
```

The explicit expressions for the PaVe functions can be obtained e.g. using Package-X / PaXEvaluate

In[32]:=
```
PaVeEvalRules={
B0[0, 0, 0] -> -1/(16*EpsilonIR*Pi^4) + 1/(16*EpsilonUV*Pi^4),
B0[s_, 0, 0]:> 1/(16*EpsilonUV*Pi^4) - (-2 + EulerGamma - Log[4*Pi] - Log[-(ScaleMu^2/s)])/
   (16*Pi^4),
C0[0, s_, 0, 0, 0, 0] :> C0[0, 0, s, 0, 0, 0],
C0[0, 0, s_, 0, 0, 0] :> 1/(16*EpsilonIR^2*Pi^4*s) -
  (EulerGamma - Log[4*Pi] - Log[-(ScaleMu^2/s)])/(16*EpsilonIR*Pi^4*s) -
  (-6*EulerGamma^2 + Pi^2 + 12*EulerGamma*Log[4*Pi] - 6*Log[4*Pi]^2 +
    12*EulerGamma*Log[-(ScaleMu^2/s)] - 12*Log[4*Pi]*Log[-(ScaleMu^2/s)] -
    6*Log[-(ScaleMu^2/s)]^2)/(192*Pi^4*s),
D0[0, 0, 0, 0, s_, t_, 0, 0, 0, 0] :> 1/(4*EpsilonIR^2*Pi^4*s*t) -
  (2*EulerGamma - 2*Log[4*Pi] - Log[-(ScaleMu^2/s)] - Log[-(ScaleMu^2/t)])/
   (8*EpsilonIR*Pi^4*s*t) - (-3*EulerGamma^2 + 2*Pi^2 + 6*EulerGamma*Log[4*Pi] -
    3*Log[4*Pi]^2 + 3*EulerGamma*Log[-(ScaleMu^2/s)] - 3*Log[4*Pi]*Log[-(ScaleMu^2/s)] +
    3*EulerGamma*Log[-(ScaleMu^2/t)] - 3*Log[4*Pi]*Log[-(ScaleMu^2/t)] -
    3*Log[-(ScaleMu^2/s)]*Log[-(ScaleMu^2/t)])/(24*Pi^4*s*t)
};
```

In[33]:=
```
bornVirtualUnrenormalized[1]=bornVirtualUnrenormalized[0]//.PaVeEvalRules;
```

Put together the counter-term contribution and the residue pole contribution

In[34]:=
```
MSbarRC={
    SMP["dZ_psi"]->- SMP["e"]^2/(16Pi^2) 1/EpsilonUV,
    SMP["dZ_A"]-> - Nf SMP["e"]^2/(12Pi^2) 1/EpsilonUV
};
```

In[35]:=
```
RuleRS={
    dZe1-> - 1/2 SMP["dZ_A"],
    dZAA1->SMP["dZ_A"],
    (dZf1|dZf2)[__]-> SMP["dZ_psi"]
};
```

```
In[36]:= legResidueContrib= 1 + SMP["e"]^2/(4 Pi)*1/(4 Pi)×1/EpsilonIR;
```

```
In[37]:= aux0=(Total[ampLoopCT[1]]/.RuleRS/.MSbarRC)//FeynAmpDenominatorExplicit//Contract//
            DiracSimplify//FCCanonicalizeDummyIndices[#,LorentzIndexNames->{mu}]&;
```

```
In[38]:= ctContrib=(aux0/ampTree[2])//Simplify;
```

```
In[39]:= fullCTAndResidue[0]=(ctContrib+(4*1/2)(legResidueContrib-1))ampTree[2]
```

Out[39]= $-\frac{1}{s}\,\mathrm{i}$ Spinor[Momentum[k1, D], 0, 1].DiracGamma[LorentzIndex[mu, D], D].Spinor[-Momentum[k2, D], 0, 1] × Spinor[-Momentum[p2, D], 0, 1].DiracGamma[LorentzIndex[mu, D], D].Spinor[Momentum[p1, D], 0, 1] $\mathrm{SMP}[e]^2\left(\frac{\mathrm{SMP}[e]^2}{8\,\mathrm{EpsilonIR}\,\pi^2}+\frac{(-3+2\,\mathrm{Nf})\,\mathrm{SMP}[e]^2}{24\,\mathrm{EpsilonUV}\,\pi^2}\right)$

Now get the interference of the counter term and residue contribution with the Born amplitude

```
In[40]:= bornCTAndResidue[0]= fullCTAndResidue[0]×ComplexConjugate[ampTree[2]]//
         FermionSpinSum[#,ExtraFactor->1/2^2]&//DiracSimplify//Simplify//
            TrickMandelstam[#,{s,t,u,0}]&
```

Out[40]= $-\frac{(3\,\mathrm{EpsilonIR}-3\,\mathrm{EpsilonUV}-2\,\mathrm{EpsilonIR}\,\mathrm{Nf})\left(D\,s^2-2\,t^2-8\,t\,u-2\,u^2\right)\mathrm{SMP}[e]^6}{24\,\mathrm{EpsilonIR}\,\mathrm{EpsilonUV}\,\pi^2\,s^2}$

For convenience, let us pull out an overall prefactor to get rid of ScaleMu, EulerGamma and some Pi's

In[41]:=
```
aux1=FCSplit[bornCTAndResidue[0],{EpsilonUV}]//
    ReplaceAll[#,{EpsilonIR->1/SMP["Delta_IR"],EpsilonUV->1/SMP["Delta_UV"]}]&;
bornCTAndResidue[1]=(FCReplaceD[1/Exp[EpsilonIR(Log[4Pi]-EulerGamma)] aux1[[1]],D->4-2EpsilonIR]+
    FCReplaceD[1/Exp[EpsilonUV(Log[4Pi]-EulerGamma)] aux1[[2]],D->4-2EpsilonUV])//
    FCShowEpsilon//Series[#,{EpsilonUV,0,0}]&//
    Normal//Series[#,{EpsilonIR,0,0}]&//Normal//Collect2[#,EpsilonUV,EpsilonIR]&
```

Out[42]=
$$-\frac{\text{Nf}\,\text{SMP}[e]^6}{6\,\pi^2}+\frac{\left(2\,s^2-t^2-4\,t\,u-u^2\right)\text{SMP}[e]^6}{4\,\text{EpsilonIR}\,\pi^2\,s^2}+\frac{(-3+2\,\text{Nf})\left(2\,s^2-t^2-4\,t\,u-u^2\right)\text{SMP}[e]^6}{12\,\text{EpsilonUV}\,\pi^2\,s^2}$$

In[43]:=
```
aux2=FCSplit[bornVirtualUnrenormalized[1],{EpsilonUV}];
bornVirtualUnrenormalized[2]=FCReplaceD[1/ScaleMu^(2EpsilonIR)*
    1/Exp[EpsilonIR(Log[4Pi]-EulerGamma)] aux2[[1]],
    D->4-2EpsilonIR]+FCReplaceD[1/ScaleMu^(2EpsilonUV)*
    1/Exp[EpsilonUV(Log[4Pi]-EulerGamma)] aux2[[2]],D->4-2EpsilonUV]//
    Collect2[#,EpsilonUV,EpsilonIR]&//Normal//Series[#,{EpsilonUV,0,0}]&//
    Normal//Series[#,{EpsilonIR,0,0}]&//Normal//
    ReplaceAll[#,Log[-ScaleMu^2/(h:s|t|u)]:>2 Log[ScaleMu]-Log[-h]]&//
    TrickMandelstam[#,{s,t,u,0}]&//Collect2[#,EpsilonUV,EpsilonIR]&;
```

Finally, we obtain the UV-finite but IR-divergent Born-virtual interference term

In[45]:=
```
bornVirtualRenormalized[0]=(bornVirtualUnrenormalized[2]+bornCTAndResidue[1])//
    TrickMandelstam[#,{s,t,u,0}]&//Collect2[#,EpsilonUV,EpsilonIR]&
```

Out[45]=
$$-\frac{\left(t^2+u^2\right)\text{SMP}[e]^6}{2\,\text{EpsilonIR}^2\,\pi^2\,s^2}+$$
$$\frac{\left(-t^2+4\,t\,u-u^2+2\,t^2\,\text{Log}[-s]+2\,u^2\,\text{Log}[-s]+2\,t^2\,\text{Log}[-t]+2\,u^2\,\text{Log}[-t]-2\,t^2\,\text{Log}[-u]-2\,u^2\,\text{Log}[-u]\right)\text{SMP}[e]^6}{4\,\text{EpsilonIR}\,\pi^2\,s^2}+\frac{1}{72\,\pi^2\,s^2}$$
$$\Big(-90\,t^2-20\,\text{Nf}\,t^2+21\,\pi^2\,t^2+108\,t\,u-90\,u^2-20\,\text{Nf}\,u^2-15\,\pi^2\,u^2+36\,t^2\,\text{Log}[-s]+12\,\text{Nf}\,t^2\,\text{Log}[-s]-72\,t\,u\,\text{Log}[-s]+12\,\text{Nf}\,u^2\,\text{Log}[-s]-$$
$$36\,u^2\,\text{Log}[-s]^2+36\,s\,t\,\text{Log}[-t]+18\,s\,u\,\text{Log}[-t]-54\,t^2\,\text{Log}[-s]\,\text{Log}[-t]-18\,u^2\,\text{Log}[-s]\,\text{Log}[-t]+9\,t^2\,\text{Log}[-t]^2-9\,u^2\,\text{Log}[-t]^2-$$
$$18\,s\,t\,\text{Log}[-u]-36\,s\,u\,\text{Log}[-u]+18\,t^2\,\text{Log}[-s]\,\text{Log}[-u]+54\,u^2\,\text{Log}[-s]\,\text{Log}[-u]+9\,t^2\,\text{Log}[-u]^2-9\,u^2\,\text{Log}[-u]^2\Big)\,\text{SMP}[e]^6$$

We can compare our O(eps^0) result to Eq. 2.22 in arXiv:hep-ph/0010075

In[46]:=
```
ClearAll[LitA,LitATilde,auxBox6,Box6Eval,TriEval];
Li4=PolyLog[4,#1]&;
ruleLit={LitV->Log[-s/u],LitW->Log[-t/u],v->s/u,w->t/u};
```

In[49]:=
```
LitA= (
4*GaugeXi*(1-2 Epsilon)*u/s^2((2-3*Epsilon)u^2-6*Epsilon*t*u+3(2-Epsilon)t^2)*Box6[s,t]

-4 GaugeXi/(1-2 Epsilon)*t/s^2*((4-12*Epsilon+7*Epsilon^2)t^2-
6*Epsilon*(1-2*Epsilon)*t*u+(4-10*Epsilon+5*Epsilon^2)*u^2)*Tri[t]

-8/((1-2*Epsilon)(3-2*Epsilon))*1/s*(2Epsilon(1-Epsilon)*t*((1-Epsilon)*t-Epsilon*u)*Nf-
Epsilon(3-2*Epsilon)*(2-Epsilon+2*Epsilon^2)*t*u+
(1-Epsilon)(3-2*Epsilon)(2-(1-GaugeXi)*Epsilon+2 Epsilon^2)t^2)*Tri[s]);
```

In[50]:=
```
auxBox6=(1/2((LitV-LitW)^2+Pi^2)+2*Epsilon*(Li3[-v]-LitV Li2[-v]-1/3 LitV^3-Pi^2/2 LitV)
-2 Epsilon^2 (Li4[-v]+LitW Li3[-v]-1/2 LitV^2 Li2[-v]-1/8 LitV^4-
1/6 LitV^3 LitW + 1/4*LitV^2*LitW^2- Pi^2/4 LitV^2-Pi^2/3 LitV LitW - 2 Zeta4));

Box6Eval[s,t]=u^(-1-Epsilon)/(2(1-2*Epsilon))(1- Pi^2/12 Epsilon^2)(
auxBox6 + (auxBox6/.{LitW->LitV,LitV->LitW,v->w,w->v}));

Box6Eval[s,u]=Box6Eval[s,t]/.ruleLit/.{t->u,u->t};

TriEval[s_]:=-(-s)^(-1-Epsilon)/Epsilon^2 (1-Pi^2/12 Epsilon^2-
    7/3 Zeta[3] Epsilon^3-47/16 Zeta4 Epsilon^4)
```

In[54]:=
```
knownResult=(( 2/3 Nf/Epsilon*8((t^2+u^2)/s^2-Epsilon)

+((LitA/.{Tri->TriEval,Box6->Box6Eval}/.ruleLit)+
 (LitA/.{Tri->TriEval,Box6->Box6Eval}/.
{GaugeXi->-GaugeXi}/. ruleLit/.{t->u,u->t}))/.GaugeXi->1));
```

knownResult is the 1-loop result. Notice that is also an implicit overall prefactor prefLit from Eq. 2.8

```
In[55]:= prefLit=32 Pi^2/SMP["e"]^6;
```

```
In[56]:= diff=Series[knownResult-
prefLit(bornVirtualRenormalized[0]/.EpsilonIR->Epsilon),{Epsilon,0,0}]//Normal//
TrickMandelstam[#,{s,t,u,0}]&//PowerExpand//SimplifyPolyLog//TrickMandelstam[#,{s,t,u,0}]&
```

```
Out[56]= 0
```

Check the final results

```
In[57]:= FCCompareResults[0,diff,
Text->{"\tCompare to arXiv:hep-ph/0010075:",
"CORRECT.","WRONG!"}, Interrupt->{Hold[Quit[1]],Automatic}];
Print["\tCPU Time used: ", Round[N[TimeUsed[],4],0.001], " s."];
```

```
	Compare to arXiv:hep–ph/0010075:  CORRECT.
	CPU Time used: 32.018 s.
```