MadForm: An Amplitude Generator

Pooja Mukherjee

In collaboration with Prof. Claude Duhr and Dr. Andres Vasquez

Bethe Center for Theoretical Physics

UNIVERSITÄT BONN

FORM and Symbolica developers meeting | May 30, 2024

Bethe Center for Theoretical Physics

UNIVERSITÄT BONN

~~MadForm~~: An Amplitude Generator

Pooja Mukherjee

In collaboration with Prof. Claude Duhr and Dr. Andres Vasquez

FORM and Symbolica developers meeting | May 30, 2024

Why another package ?

- Automation for diagram generation : qgraf or FeynArt.
- Automation for Feynman rules generation : FeynRules.
- Complete automation of one-loop computations : FeynCalc, FormCalc, QCDLoop, Madgraph, …
- But two -loops and beyond no such algorithm is known.
- Different parts of such computations are done by different tools.
- Can we try to assemble the best tools together and see how far can we automate the steps of loop computation?


ANATAR
AN Automated Tool for Higher order Amplitude geneRation
P.C. Andres & Paarth

The Assembly

CORE

- **Amplitude.m :** Definitions of main functions for Amplitude generation
- **Color.h :** Computation of Color factors - Available publicly by Vermaseren
- **DiracTraces.frm:** Computation of Dirac Traces
- **ProjectorsLib.frm :** Library function for various projectors
- **Projectors.m :** Definitions of main functions for projectors
- **IntegralDecomposition.m:** Main functions for topology reduction and completion of numerator
- **TemplateForm.m :** Writing of template files for FORM

CORE

MODEL

- Amplitude.m :
- Color.h :
- DiracTraces.frm:
- ProjectorsLib.frm :
- Projectors.m :
- IntegralDecomposition.m:
- TemplateForm.m :

CORE

MODEL

- Amplitude.m :
- Color.h :
- DiracTraces.frm:
- ProjectorsLib.frm :
- Projectors.m :
- IntegralDecomposition.m:
- TemplateForm.m :

- SM
- QCD
- QED
- Chromo

⋮

CORE

- Amplitude.m :
- Color.h :
- DiracTraces.frm:
- ProjectorsLib.frm :
- Projectors.m :
- IntegralDecomposition.m:
- TemplateForm.m :

MODEL

- SM
- QCD
- QED
- Chromo
- ⋮

Couplings.frm
Polarisations.frm
Propagations.frm
Vertices.frm
smQG.qgraf

CORE

- Amplitude.m :
- Color.h :
- DiracTraces.frm:
- ProjectorsLib.frm :
- Projectors.m :
- IntegralDecomposition.m:
- TemplateForm.m :

MODEL

- SM
- QCD
- QED
- Chromo

⋮

INTERFACE

CORE

- Amplitude.m :
- Color.h :
- DiracTraces.frm:
- ProjectorsLib.frm :
- Projectors.m :
- IntegralDecomposition.m:
- TemplateForm.m :

MODEL

- SM
- QCD
- QED
- Chromo

•
•
•

INTERFACE

- QGraf.m
- Kira.m

CORE

- Amplitude.m :
- Color.h :
- DiracTraces.frm:
- ProjectorsLib.frm :
- Projectors.m :
- IntegralDecomposition.m:
- TemplateForm.m :

MODEL

- SM
- QCD
- QED
- Chromo

•
•
•

INTERFACE

- QGraf.m
- Kira.m

OUTPUT

CORE

- Amplitude.m :
- Color.h :
- DiracTraces.frm:
- ProjectorsLib.frm :
- Projectors.m :
- IntegralDecomposition.m:
- TemplateForm.m :

MODEL

- SM
- QCD
- QED
- Chromo

⋮

INTERFACE

- QGraf.m
- Kira.m

OUTPUT

- Diagrams stored in separate folders

Example : SM at tree level

- Amplitude generation for any process :

```
In[1]:=   Amp0 = GenerateAmplitude[{G, G}→ {G, G}, "smNew",OutputName → "gg0L",
          QGLoops→ "0"];
In[2]:=   Amp0[[5, 1]]
-----------------------------------------------------------------------------
Out[2]:=  DiagramID[1] →
          (-I GC12 f[Gluon1, Gluon4, indexN1] f[Gluon2, Gluon3, indexN1]
          -I GC12 f[Gluon1, Gluon3, indexN1] f[Gluon2, Gluon4, indexN1])
          PolV[-4, Lor4, p4, 0, Gluon4] PolV[-3, Lor4, p2, 0, Gluon3]
          PolV[-2, Lor2, p3, 0, Gluon2] PolV[-1, Lor2, p1, 0, Gluon1] +···
```

- Amplitude conjugate generation for any process :

```
In[3]:=   AmpC0 = GenerateAmplitudeConjugate[{G, G}→ {G, G}, "smNew", QGLoops -> "0"];
```

Example : SM at tree level

- Amplitude square generation for any process :

```
In[4]:=    AmpSq0 = GenerateAmplitudeSquare[{G, G}→ {G, G}, "smNew", Amp0 , AmpC0 ];
In[5]:=    FullSimplify[AmpSq0, S+T+U==0]
```

Out[5]:= $\frac{1152\ gs^4(T^2+TU+U^2)^3}{S^2T^2U^2}$

- Computation can be performed in user defined symbols for parameters.

- Computation can be performed in D dimension also.

- Has the ability to handle large number of gamma matrices for D dimension analysis.

Example : SM at tree level

- Some checks on the squaring of the amplitude :

Process	FormCalc	ANATAR
e+ e- -> e+ e-	0.229514 s	0.02664 s
g g -> g g	2.68677 s	1.52797 s
e+ e- -> t t~	0.135578 s	0.016607 s
g g -> t t~	0.4004 s	0.150944 s
u u~ -> t t~	0.148539 s	0.021193 s
b Z -> b Z	2.31182 s	1.1908 s
e+ e- -> neutrinos	0.053934 s	0.0204434 s
Z -> W+ W-	0.03028 s	0.017064 s

- Matches upto an overall factor with FormCalc
- Slightly faster than FormCalc !

Example : SM at one loop

- Amplitude generation for any process at one loop :

```
In[7]:=   Amp1 = GenerateAmplitude[{G, G} → {H}, "sm", OutputName → "ggH1",
          QGLoops → "1", QGOptions→ "onepi,notadpole,onshell",
          QGCouplingsSpecification → {{gs, 1, 2}, {ee, 0, 0}, {ymt, 1, 1}},
          FO$Trim → True];
```

- Projected amplitude   generation for any process :

```
In[8]:=   ProjAmp1 = ProjectAmplitude[Amp1, CombineDiagrams → False] /.  FO$GCouplings
```

Out[8]:= $\{\, \text{DiagramID[1]} \to \frac{-1}{(-2 + D)\ \text{S12 vev}}\ \text{gs}^2\ \text{MT}^2\ \text{Den[-k1, MT] Den[-k1 + p1, MT]}$
$\text{Den[-k1 - p2, MT]}\ (\text{S12}\ (-2\ (-1 + D)\ \text{MT}^2 + (-2 + D)\ \text{S12} + 2\ (-5 + D)\ \text{k1.k1})$
$+ 16\ \text{p1.k1 p2.k1})\ \text{FO\$Metric[Gluon1, Gluon3]},$
$\text{DiagramID[2]} \to \frac{-1}{(-2 + D)\ \text{S12 vev}}\ \text{gs}^2\ \text{MT}^2\ \text{Den[k1, MT] Den[k1 - p1, MT]}$
$\text{Den[k1 + p2, MT]}\ (\text{S12}\ (-2\ (-1 + D)\ \text{MT}^2 + (-2 + D)\ \text{S12} + 2\ (-5 + D)\ \text{k1.k1})$
$+ 16\ \text{p1.k1 p2.k1})\ \text{FO\$Metric[Gluon1, Gluon3]}\}$

Example : SM at one loop

- Finding topology at one loop :

```
In[9]:=    Topos1L = FindTopology[ProjAmp1, {k1}, {p1, p2}]

Out[9]:=   {{"TOPO1", Den[-k1, MT], Den[k1 - p1, MT], Den[-k1 - p2, MT]}}
```

- Writing the amplitude in terms of master integrals at one loop:

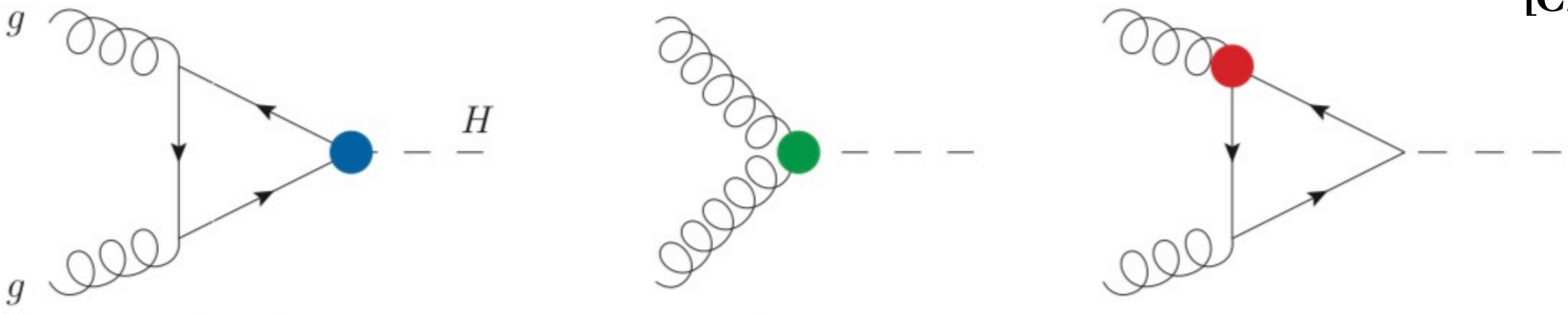
```
In[10]:=   AmpToInt1Loop = (AmplitudeToTopo[ProjAmp1, Topos1L, {k1}, {p1, p2}])
           /.  FO$ListKinematics
In[11]:=   AmpToInt1Loop[[1]]

Out[10]:=  Finding loop-momentum shifts for the ampltiudes...
           Rewriting the numerator...
           Done!
Out[11]:=  DiagramID[1] → -1/((-2 + D) S12 vev) gs^2 MT^2 (-4 FO$LoopInt[TOPO1[-1, 1, 1]]
           + 4 FO$LoopInt[TOPO1[0, 0, 1]] +4 FO$LoopInt[TOPO1[0, 1, 0]]
           + 2 (-5 + D) S12 FO$LoopInt[TOPO1[0, 1, 1]] -4 FO$LoopInt[TOPO1[1, 0, 0]]
           + S12 (-8 MT^2 + (-2 + D) S12) FO$LoopInt[TOPO1[1, 1, 1]])
           FO$Metric[Gluon1, Gluon3]
```

Example : SMEFT at two loop

- Interested in operators that modify heavy quarks contributions: :

[Claude Duhr et. al. ´18]



$(\Phi^\dagger\Phi)\,\bar{Q}_L\Phi\,q_R$ $(\Phi^\dagger\Phi)\,GG$ $\bar{Q}_L\Phi\,\sigma q_R G$

- Amplitude generation for the above model :

```
In[12]:=   Amp2Loop = GenerateAmplitude[{G, G} → {H}, "chromoNew",
           OutputName → "gg2H2L", QGLoops→ "2",
           QGOptions → "onepi,notadpole,nosnail,onshell",
           NoPropagatorSpecification → {{H, 0, 0}},
           QGCouplingsSpecification → {{gs, 0, 4}, {ymt, 0, 1}, {cEFT, 1, 5}},
           FO$Trim → True, AmpCouplingsSpecification → {{cEFT, 1}}];
```

Example : SMEFT at two loop

- Projected amplitude   generation for the process :

```
In[13]:=   ProjAmp2 = ProjectAmplitude[Amp2Loop, CombineDiagrams -> False]
           /.FO$GCouplings
```

- Finding momentum shifts for the topology given in the paper:

```
In[14]:=   Topos2L = { topo1 → {Den[k1, 0], Den[k1 + p1, 0], Den[k1 + p1 + p2, 0],
           Den[k2 + p1 + p2, MT], Den[k2 + p1, MT], Den[k2, MT], Den[k1 - k2, MT]},
           topo2→ {Den[k1, MT], Den[k1 + p2, MT], Den[k1 + p1 + p2, MT],
           Den[k2 + p1 + p2, MT], Den[k2 + p2, MT], Den[k2, MT], Den[k1 - k2, 0]},
           topo3 → {Den[k1, MT], Den[k1 - k2 - p1, 0], Den[k1 + p1 + p2, MT],
           Den[k2 + p1 + p2, MT], Den[k2 + p1, MT], Den[k1 + p1, MT], Den[k1 - k2, 0]}};

In[15]:=   FindShiftTransformations[ Topos2L, #, k1, k2, p1, p2] & /@ ProjAmp2;
In[16]:=   %[[1;;2]];

Out[16]:=  {{topo1, {k1 → k1 + p1, k2→ k2 + p1}},
           {topo1, {k1 → k2 + p1, k2→ -k1 + k2}}}
```

Example : SMEFT at two loop

- Applying the momentum shifts and writing the denominators in terms of the topologies .

- Doing the reduction and mapping in terms of the master intergrals :

```
In[17]:=   reducedAmp = IntegralReduceKira[AmpToInt2Loop, Topos2L, {k1, k2}, {p1, p2},
           {p3}, BasisMI → basisPaper]

Out[17]:=  Running Kira
           IBP's computed.
           Reduction of integrals in amplitude.
           There are 18 Master Integrals on the right-hand-side of substitutions
           from Kira.
           Summing up all the amplitudes and simplifying...
           18 Master Integrals found in the final expression of the amplitude.
           They can be found in the FO$MasterIntegrals list.
```

Key Features:

`GenerateAmplitude:`	Computation of amplitude for a given model at a particular order.
`ProjectAmplitude:`	Computation of projected amplitude with user defined projectors.
`GenerateSquareAmplitude:`	Computation of square of the amplitude at a particular order.
`FindTopology:`	Find the list of independent sets of denominators.
`FindShiftTransformations:`	Find momentum shifts for user defined topology.
`AmplitudeToTopo:`	Writes the amplitude in terms of topologies.
`IntegralReduceKira:`	Extracts the integrals from the amplitude and does the reduction in kira.

Thank you !

Please share lots of suggestion's / comments