Evaluation of Pentagons and Hexagons

Francisco Campanario

University of Kalsruhe(ITP) & University of Valencia

- Introduction
- Automatic calculation up to Hexagons.
- Stabilities issues.
- Results
- Summary and Outlook

Pentagons and Hexagons: Where?

In 1 loop Feynman Diagrams with 5/6 external legs(particles).

External legs	Diagram	Integral	Scalar	Tensor
$2(e^- \to e^-)$	Self Energies	2 pt Functions	B_0	$\mathrm{B}^{\mu}/\mathrm{I}_2^{\mu}$
$3(e^- e^+ \to \gamma)$	Vertices	3 pt Functions	C_0	$\mathrm{C}^{\mu_1\mu_2}/\mathrm{I}_3^{\mu_1\mu_2}$
$4(u\bar{u} \to WW)$	Boxes	4 pt Functions	D_0	$\mathrm{D}^{\mu_1\mu_2\mu_3}/\mathrm{I}_4^{\mu_1\mu_2\mu_3}$
$5(u\bar{u} \to W\gamma W)$	Pentagons	5 pt Functions	E_0	$\mathrm{E}^{\mu_1\mu_2\mu_3\mu_4}/\mathrm{I}_5^{\mu_1\mu_2\mu_3\mu_4}$
$6(u\bar{u} \to WWgg)$	Hexagons	6 pt Functions	F_0	$\mathrm{F}^{\mu_1\mu_2\mu_3\mu_4\mu_5}/\mathrm{I}_6^{\mu_1\mu_2\mu_3\mu_4\mu_5}$

$$I_6 = \sum_{i_1 \ldots i_5 = 1}^{5} p_{i_1}^{\mu_1} p_{i_2}^{\mu_2} p_{i_3}^{\mu_3} p_{i_4}^{\mu_4} p_{i_5}^{\mu_5} \underbrace{F_{i_1 i_2 i_3 i_4 i_5}}_{\text{Tensor coefficients}} + \ldots \to 5000 \text{ terms}$$

Pentagons and Hexagons are complicated, but many $2 \to 3$ and $2 \to 4$ processes are required at NLO accuracy at the LCH

FeynCalc

Generally, not possible to automatically compute what you want!

- Pentagons and Hexagons not implemented in FeynCalc.
- Require mathematica computing knowlegde: Definition of Functions, Manipulation of expressions(sustitution rules,...)

Nevertheless:

- FeynCalc can use FeynArt output
- Vary many useful functions

With practice, you can define your own functions within FeynCalc to evaluate automatically your processes.

OneLoop Calculation for $u\bar{u} \to WW\gamma$

- Load FeynCalc and FeynArts
- Define kinematics: <<../Kinematics/Kinematicsppwγw.m
- Create diagrams and amplitudes: <<ppwγw_fa_fc.m
- Simplify:

$\mathcal{M}$=MyOneLoopSimplify[amp,q,{p1,p2,p3,p4,p5(,p6)},{Options}]

The result:

1. For finite pieces $\to B_0, C_0, D_0, E_0, F_0,\ E_{ij}, \ldots$ (PaVe notation)
2. For (IR,UV) divergences $\to$ 1/Eps^n terms (Decomposition=true)

Write the result semi-automatically into Fortran.

Modular structure of VBFNLO(heading of subroutines,...).

MyLoopSimplify

- Cancellation of scalar products against denominators
  $\frac{l^2}{(l^2)(l+p)^2} \longrightarrow \frac{1}{(l+p)^2}$
- Write Expression in terms of

$$\sum_i \underbrace{(\text{Tensor integral})}_{A[i]} \times \underbrace{(\text{Matrix Elements})}_{B[i]}$$

- A[i]: Classification of Integral Basis
  - If Decomposition=true $\longrightarrow$ 1/Eps^n terms
    - Divergent Tensor coefficients from library
- Plug Tensor coeffients in Tensor integrals ($I_3^{\mu,\nu} = p^\mu p^\nu C_{ij}$)
- Classification of Matrix Elements and simplification to SMB

Finally, Sum, Collect and Simplify $\longrightarrow$

$$\boldsymbol{\mathcal{M}} = \sum_{\boldsymbol{i},\boldsymbol{j}} \mathrm{SMB}(\boldsymbol{i})\ \mathrm{F1}(\boldsymbol{j}),$$

- SMB(i): Basis of Matrix Elements
- F1(j):
$$\mathrm{F1}(\boldsymbol{j}) = \sum_{\boldsymbol{k},\boldsymbol{l}} \mathrm{F}(\boldsymbol{k})\ \mathrm{Ch}(\boldsymbol{l})$$
- F(k): Loop integrals and kinematics
- Ch(j):Polarization Vectors,..(eg,Ch(j)= $\prod_{\boldsymbol{m},\boldsymbol{n}} \boldsymbol{p_m} \cdot \boldsymbol{\epsilon(p_n)}$)

Change	SMB's	F1's	F's
Spinors	Yes	No	No
$\boldsymbol{\epsilon(p_i) \to p_i}$	Yes	Yes	No
% CPU	$\approx$ 1%	$\approx$ 5-20%	$\approx$ 95-80%

Calls for Gauge test and different Helicities a fraction of the time

Modular structure computed so far

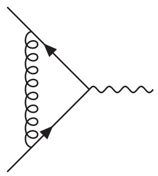
- For pp$\longrightarrow$ VVV and towards pp $\longrightarrow$ WWjj
  - boxlineAbe, boxline1gNoAbe
  - penlineAbe, penline1gNoAbe, penlineA2gNoAbe
  - HexlineAbe, Hexline1gNoAbe, HexlineA2gNoAbe
- For Gluon Fusion(H$\longrightarrow$jjj):
  - boxHiggs"x"
  - penHiggs"x"
  - HexHiggs"x"

"x" means:(Even,Odd,sfermion)

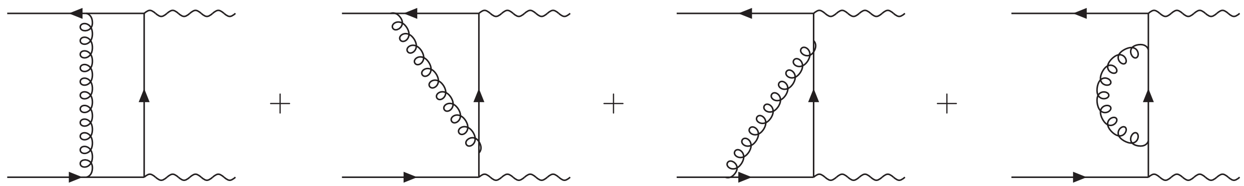
- Tensor reductions subroutines upto Hexagons of Rank 5(D-D but with PaVe Notation)

Abelian Type

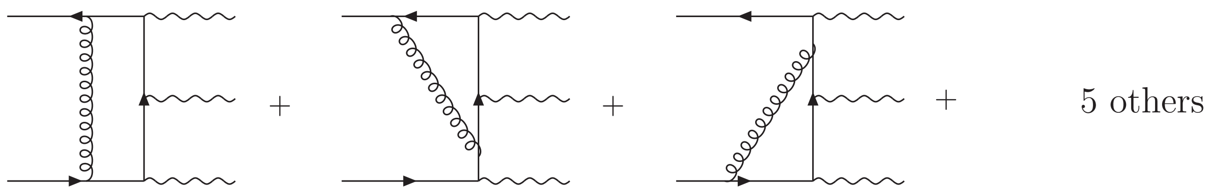
a)



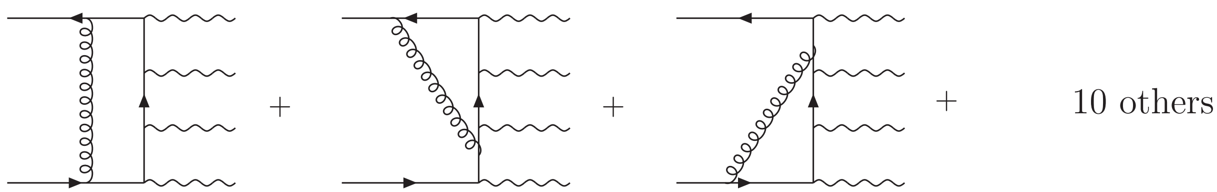
b)



c)

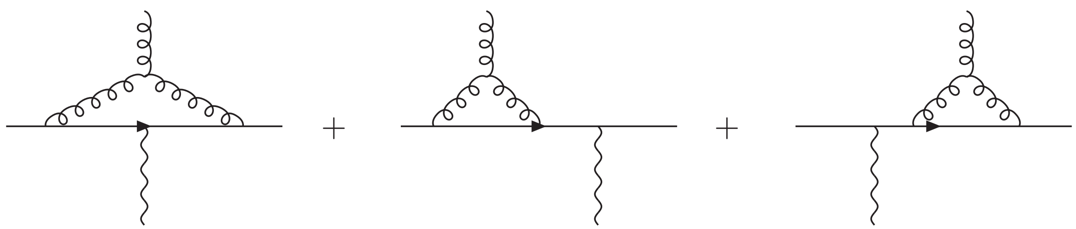


d)

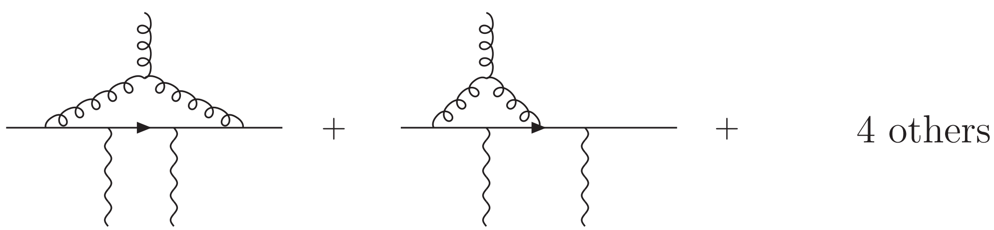


Non-Abelien Type(1 gluon)

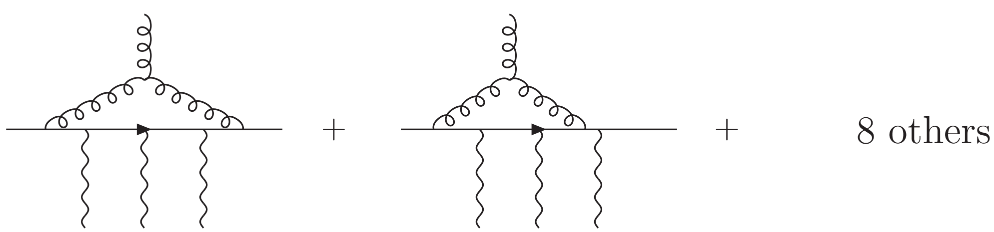
a)



b)



c)



Checks

- Gauge Test for Abe and 1gNoAbe: $(\boldsymbol{p_i \cdot \epsilon(p_i) \longrightarrow p_i \cdot p_i})$

penline=boxline(1)-boxline(2)

hexline=penline(1)-penline(2)

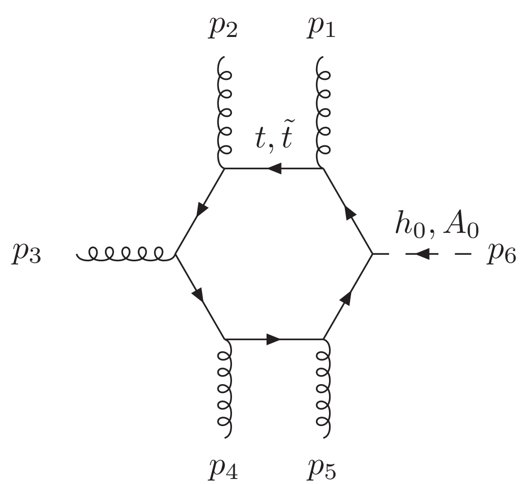
- In the Gauge test, C$_{\boldsymbol{A}}$ term for 1 g production cancel between Abe and NoAbe:

$\longrightarrow$ Check NoAbe agains Abe blocks

- Divergent pieces factorizes against born amplitude

For on-shell massless particles(g,γ), transversability must be used($\boldsymbol{\epsilon(p_i) \cdot p_i = 0}$)

Gluon Fusion

HexHiggs"x"=  ,x $\in$(Even,Odd,sfermion)

- 120 Hexagons from different (p1,p2,p3,p4,p5) permutations

Check:

- Gauge tests($\boldsymbol{\epsilon(p_i)} \longrightarrow \boldsymbol{p_i}$)

Instabilities

Point instable: Gauge test $\geq \mathbf{10^{-6}}$

- Pentagons: ~2/1000
- Hexagons: 2%-10%

Instabilites due to small Gram determinants in C and D functions

$\det/\lvert\det\rvert \approx \mathbf{10^{(-5)}}$	$\longrightarrow$	cancellation of 3-5 digits expected
D's, up to Rank 3	$\longrightarrow$	9-15 digits loss over 16 of Dble precision

- For Hexagons:

1/3 pts fails the gauge test for an accuracy of $\mathbf{10^{-1}}$

We need more digits for those$\longrightarrow$ Quadruple precision

QUAD Precision:34-36 digits

However, QUAD $\sim$ 40 times slower than Dble precision!!!!

First: Evaluate Dble Precision and apply Gauge Test

If Gauge test Failed(10% of the times)

Option 1

1) Revaluation with QUAD precision

Option 2

1) QUAD only to determine C_0,D_0,C_{ij},...(10% of the CPU)

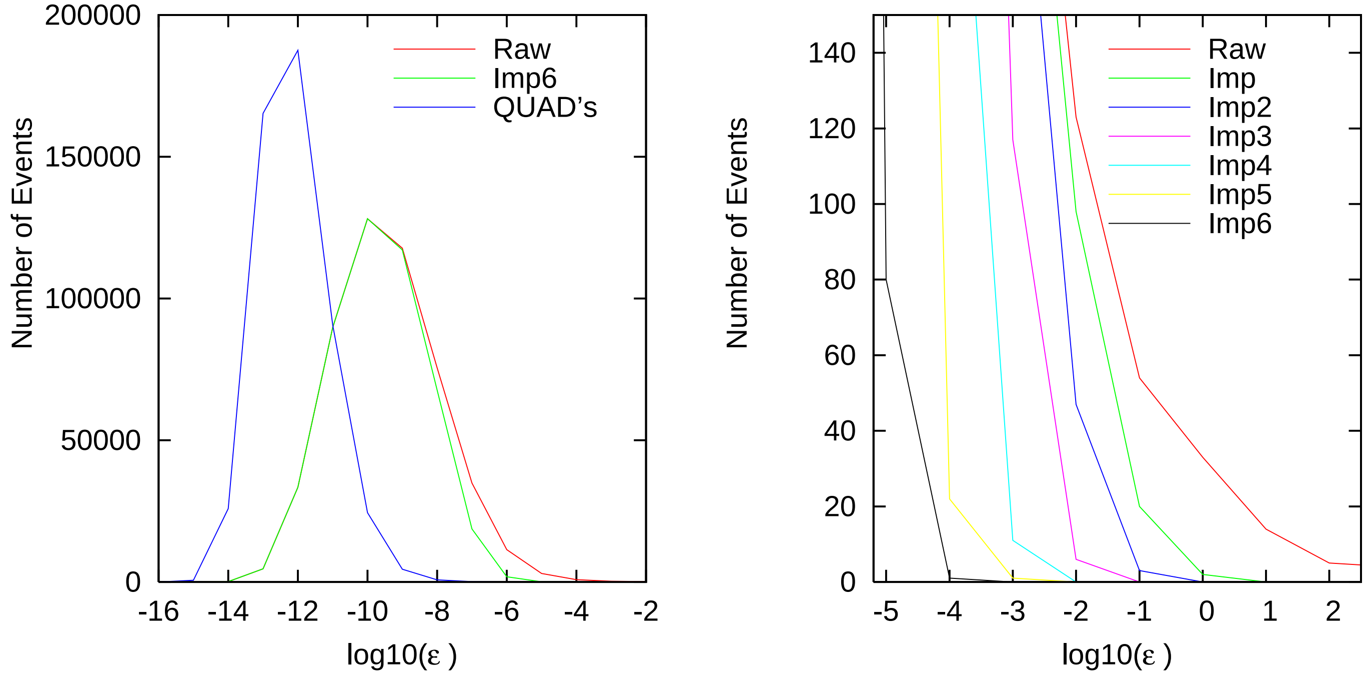
Gauge test failed: 10% $\longrightarrow$ 0.2% (0.02%,0.0008%,0)

2) Full QUAD for 0.2%

Gauge test failed: 0.2% $\longrightarrow$ 0.1% (0.007%,0)

From 400% $\longrightarrow$ 50% CPU time in addition

For 5 10^5 cut-accepted points with VBFNLO for EW pp→WWjj



ϵ=abs((QUAD-Dbl))/abs(QUAD)

Imp(,2,3,n): Gauge test $\leq 10^{(-1,-2,-3,n)} \rightarrow$ Dble=QUAD

Sumary and Outlook

- With some efforts and using FeynCalc, you can create your own code to automatically compute up to Hexagons.
- Calcutation towards pp$\rightarrow$ WW jj and H $\rightarrow$ jjj is on the way.
- Instabilities are under control with QUAD precision.
- Gauge tests and tricks reduce the slowering factor of QUAD.
- Speed of the code is competitive against Helicity method($\sim$2ms).

Outlook:

- Implementation for H$\rightarrow$ jjj of the Hexagons(Michael Kubocz)
- Real emision,ampitudes and implementation for pp$\rightarrow WWjj$
- Heptagons can be handled, but not clear motivation.