UFO & ALOHA

Mattelaer Olivier (UCL-CP3)

UFO

Céline Degrande,
Claude Duhr,
Benjamin Fuks,
David Grellscheid,
Thomas Reiter

Aloha

Priscila Aquino,
Céline Degrande,
William Link,
Fabio Maltoni,
Tim Stelzer

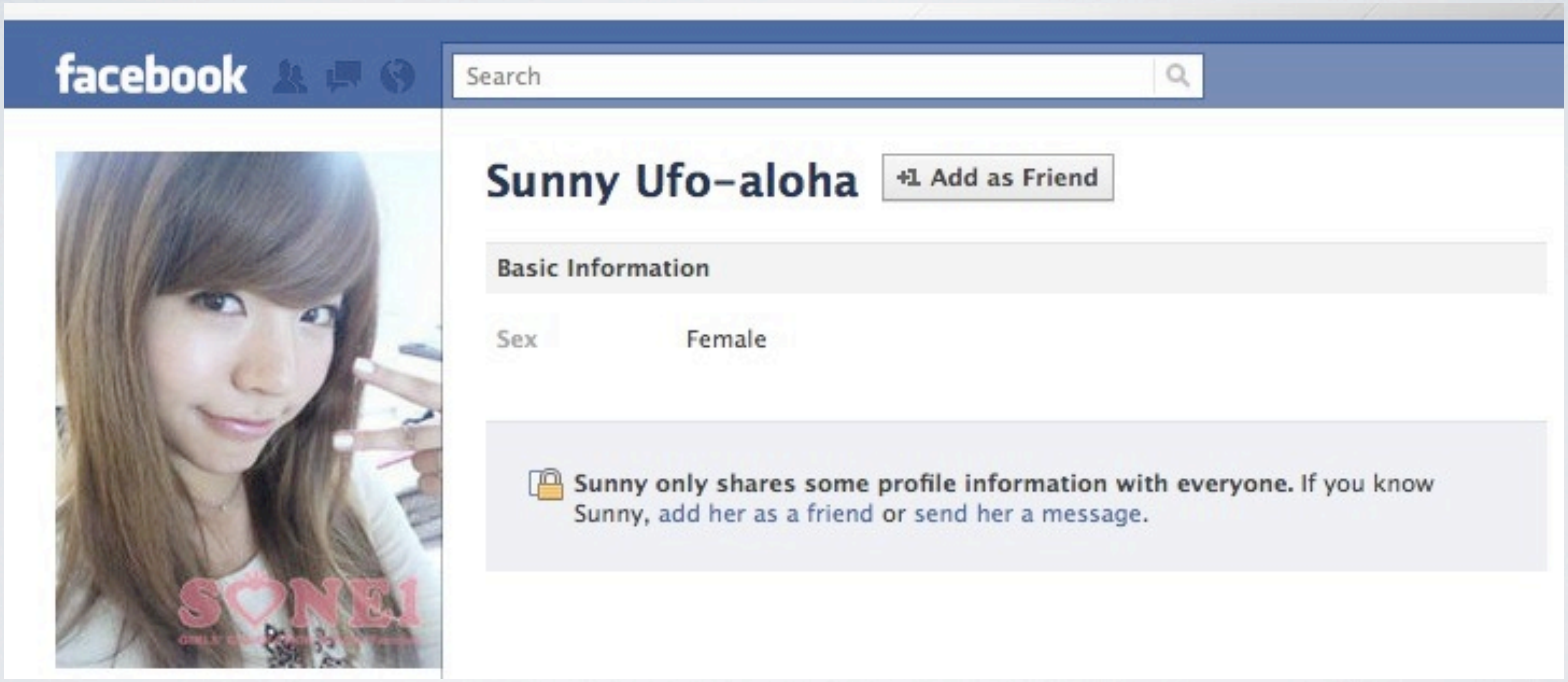
WHAT IS THIS?

WHAT IS THIS?

Google is your friend...

WHAT IS THIS?

Google is your friend... or not...



facebook

Search

Sunny Ufo-aloha

+1 Add as Friend

Basic Information

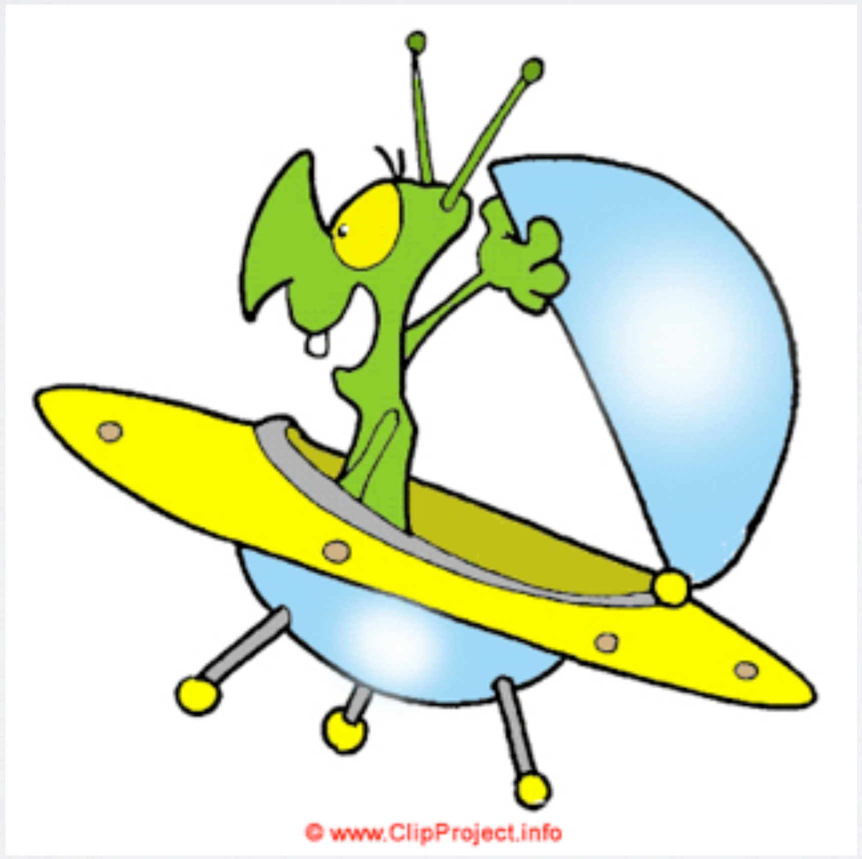
Sex Female

Sunny only shares some profile information with everyone. If you know Sunny, add her as a friend or send her a message.

WHAT IS THIS?

UFO = UNIVERSAL FEYNRULES OUTPUT

New model format



WHAT IS THIS?

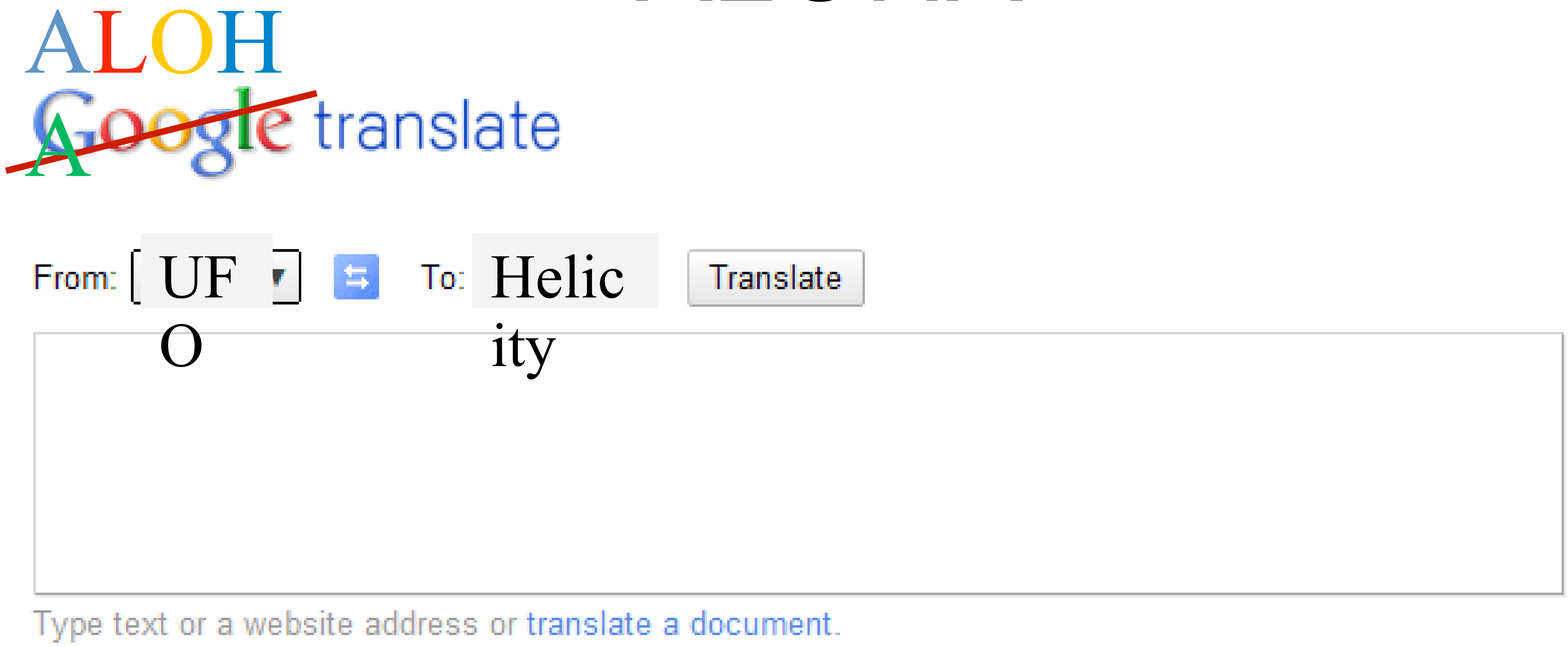
UFO = UNIVERSAL FEYNRULES OUTPUT

New model format

ALOHA= Automatic Language Independant Output Helicity Amplitude

Create Helas Routine

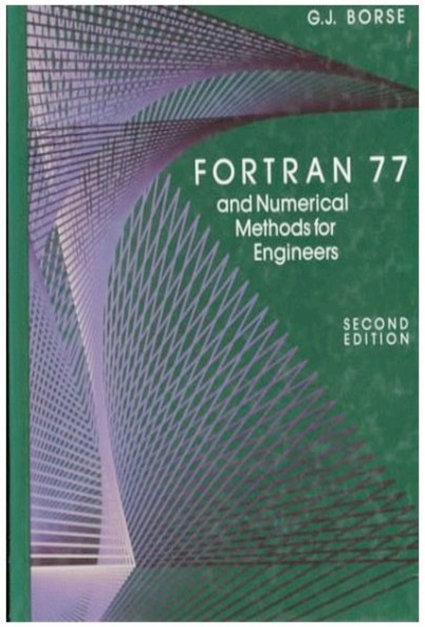
ALOHA

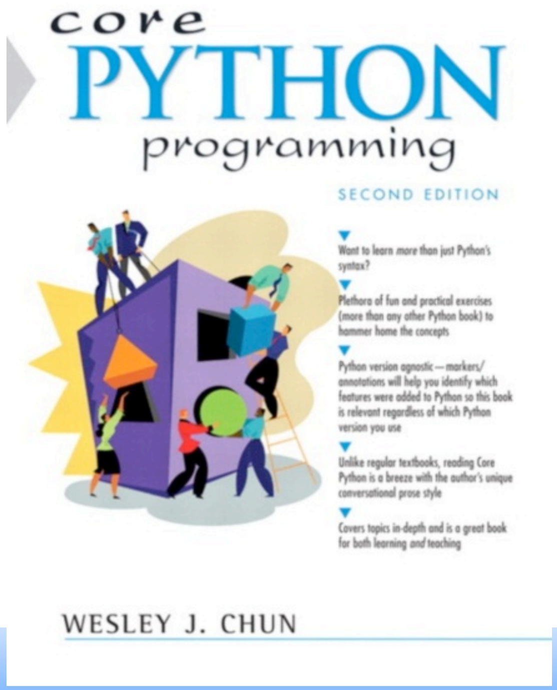


ALOH ~~Google~~ A translate

From: UFO To: Helicity Translate

Type text or a website address or translate a document.





PLAN

- UFO
  - Motivation
  - Structure of the information
- ALOHA
  - Motivation
  - HELAS
  - Automation
  - Special Routine
- Link to MG5

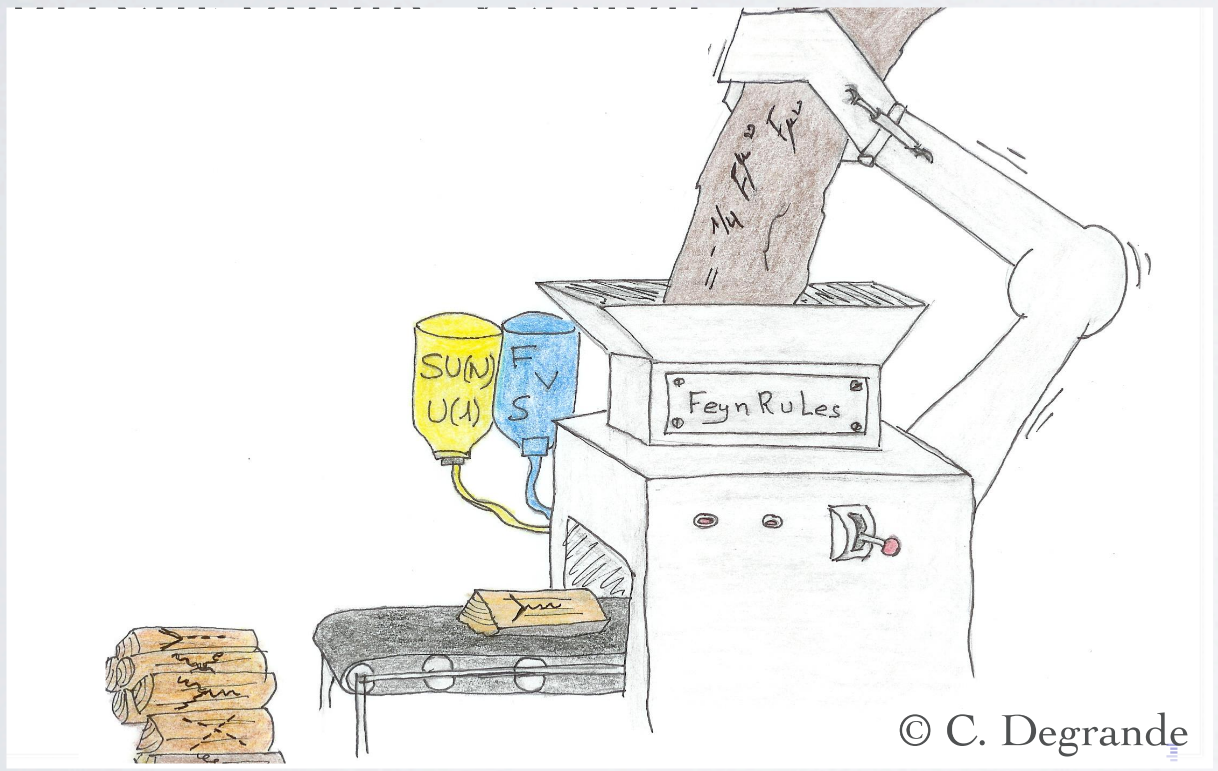
UFO: MOTIVATIONS

UFO: MOTIVATIONS

- Avoid multiple output model written by FR.

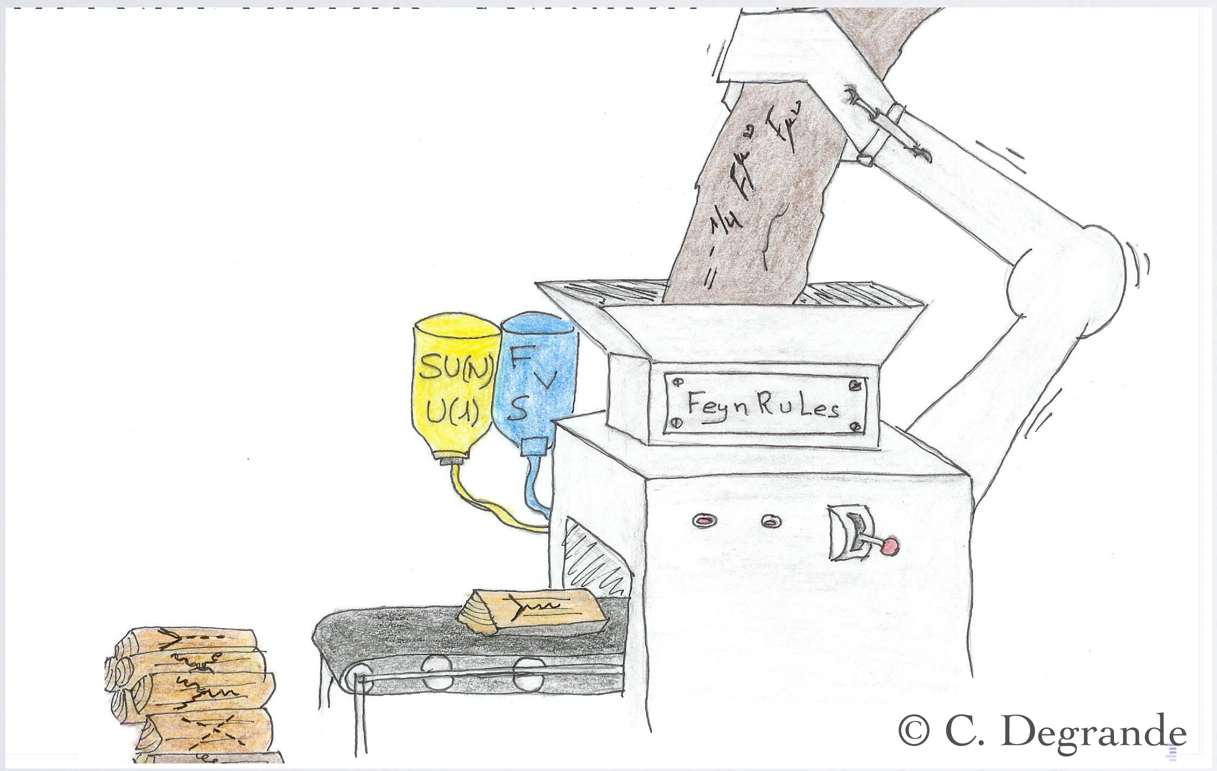
UFO: MOTIVATIONS

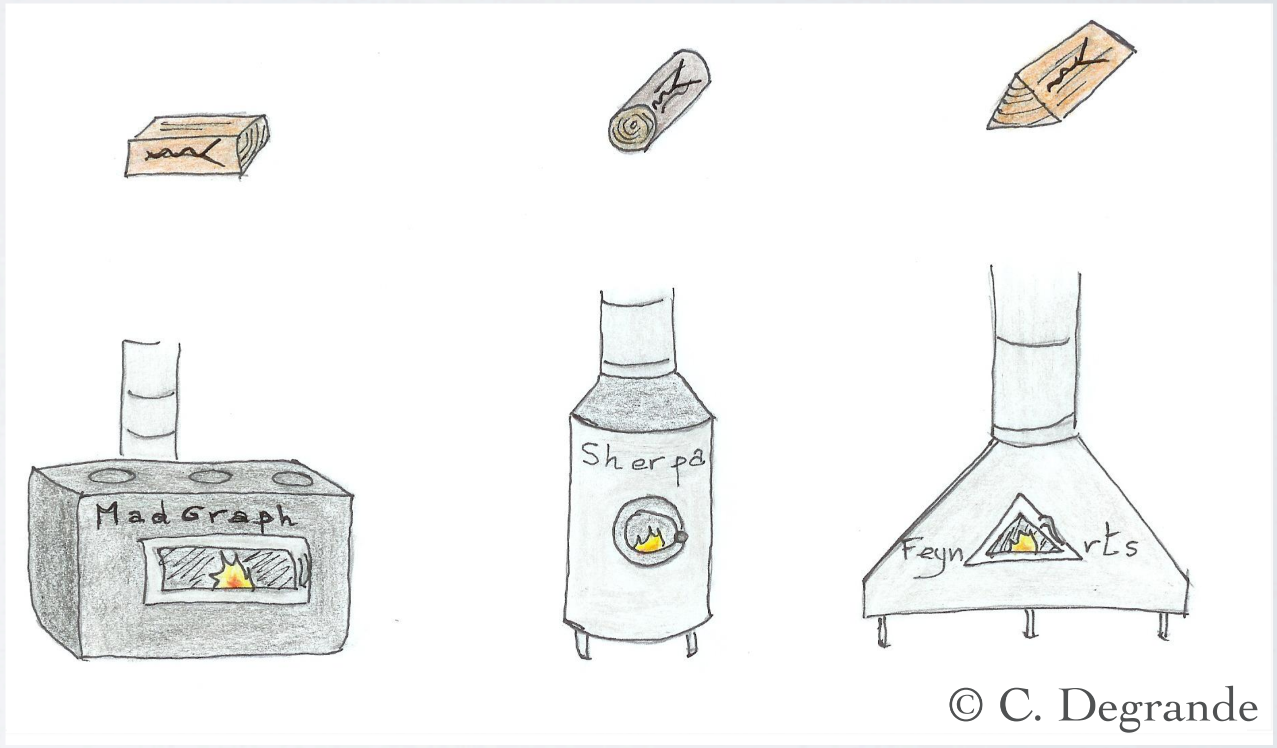
- Avoid multiple output model written by FR.



UFO: MOTIVATIONS

- Avoid multiple output model written by FR.





UFO: MOTIVATIONS

- Avoid multiple output model written by FR.
- Have the generator to adapt to the model and not the opposite.

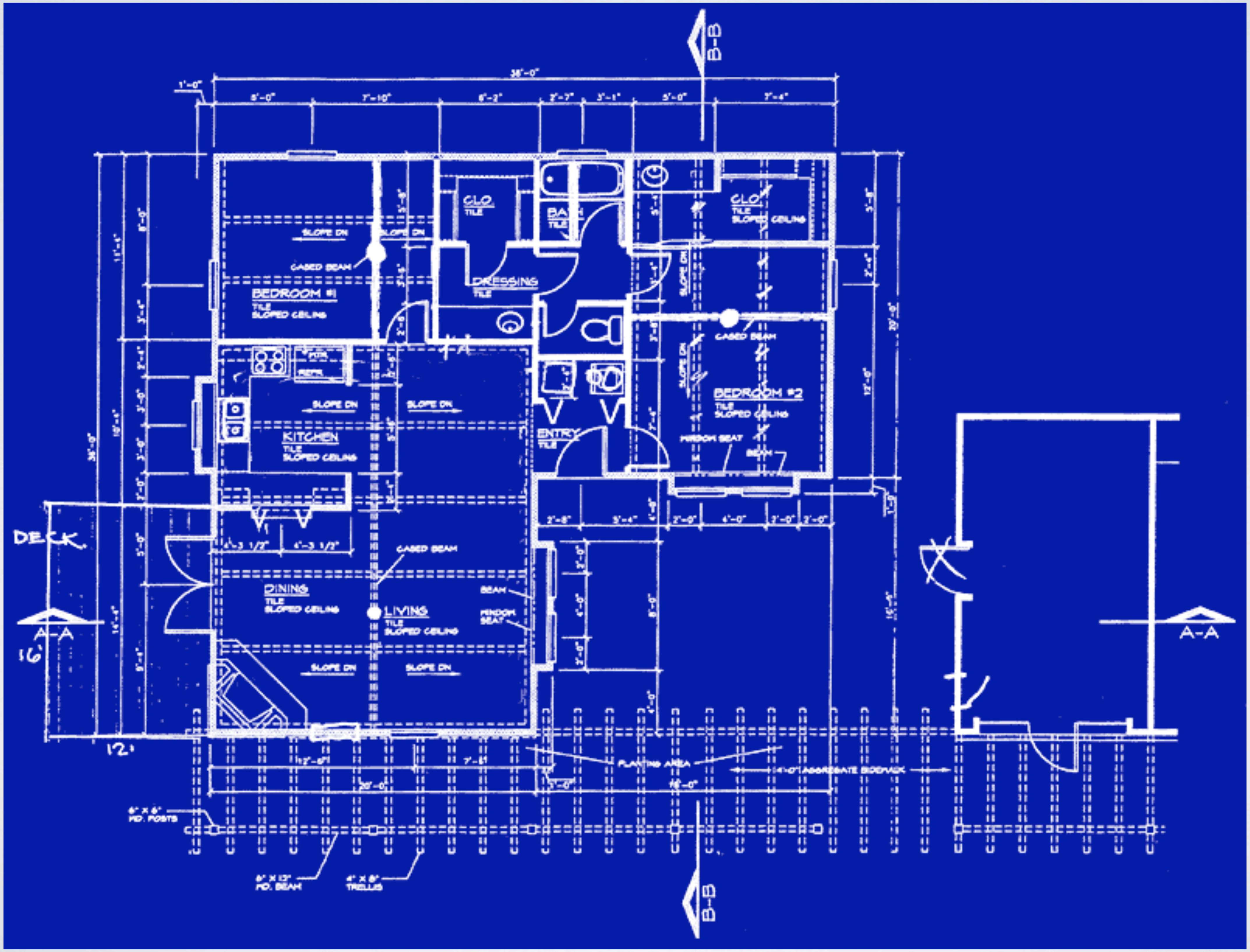
UFO: MOTIVATIONS

- Avoid multiple output model written by FR.
- Have the generator to adapt to the model and not the opposite.
- Avoid any possible limitations
  - color
  - helicity
  - number of particles in a vertex
  - gauge

UFO

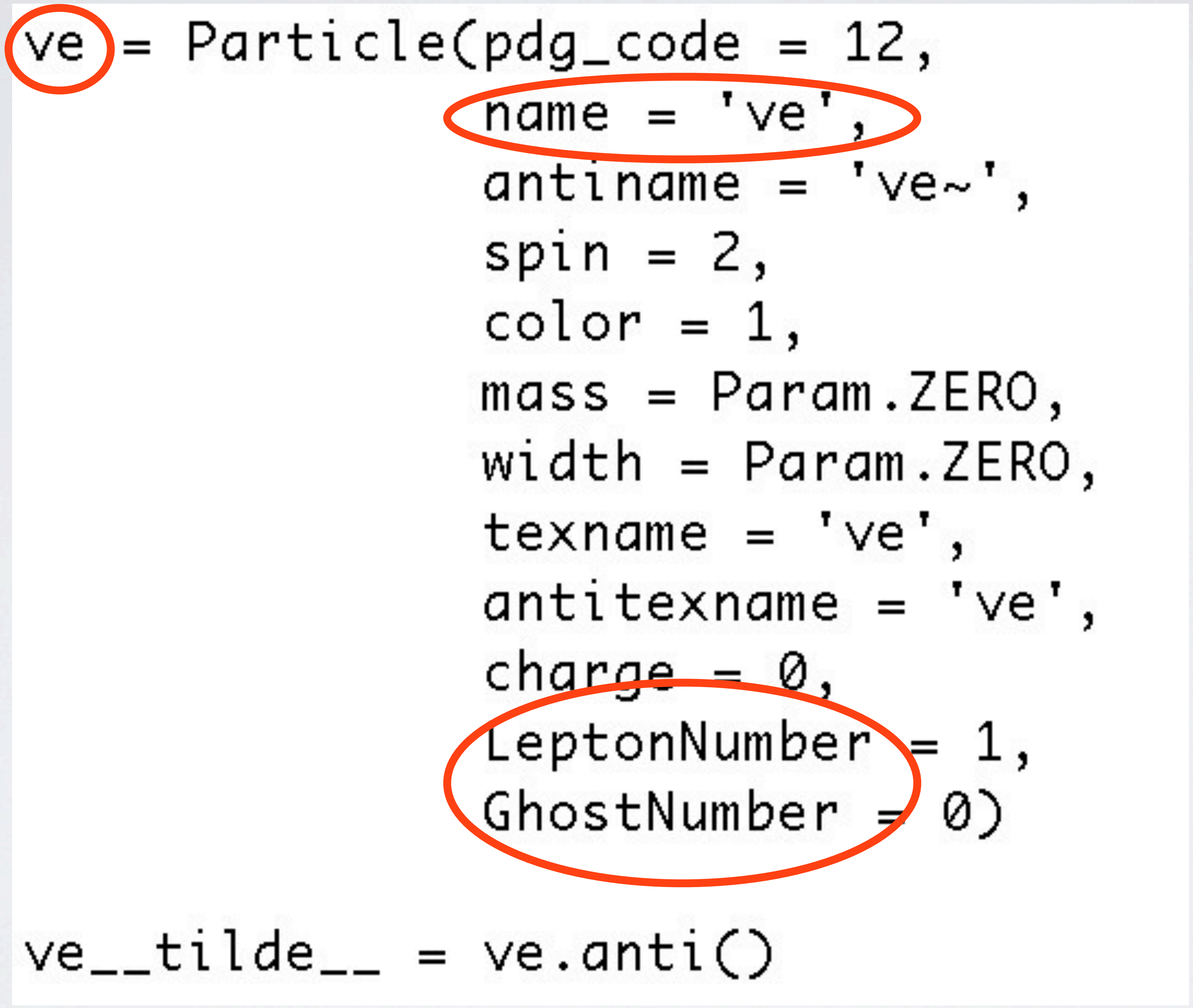
- Joint model for MG5 / GOLEM / Herwig++
- Python Object Oriented Model
  - Easy to interface with any code
- Standalone valid (python) model

FORMAT



B-B
38'-0"
BEDROOM #1
KITCHEN
DINING
LIVING
CLO
BATH
DRESSING
ENTRY
BEDROOM #2
DECK
A-A
PLAYING AREA

FORMAT



```
ve = Particle(pdg_code = 12,
              name = 've',
              antiname = 've~',
              spin = 2,
              color = 1,
              mass = Param.ZERO,
              width = Param.ZERO,
              texname = 've',
              antitexname = 've',
              charge = 0,
              LeptonNumber = 1,
              GhostNumber = 0)

ve__tilde__ = ve.anti()
```

Automatic list creation (here all_particles)

U er a e e O (U O)

particles.py:

```
G = Particle(pdg_code = 21,
             name = 'G',
             antiname = 'G',
             spin = 3,
             color = 8,
             mass = 'ZERO',
             width = 'ZERO',
             texname = 'G',
             antitexname = 'G',
             line = 'curly',
             charge = 0,
             LeptonNumber = 0,
             GhostNumber = 0)
```

lorentz.py:

```
VVV1 = Lorentz(name = 'VVV1',
               spins = [ 3, 3, 3 ],
               Structure =
                    'P(3,1)*Metric(1,2) -
                    P(3,2)*Metric(1,2) -
                    P(2,1)*Metric(1,3) +
                    P(2,3)*Metric(1,3) +
                    P(1,2)*Metric(2,3) -
                    P(1,3)*Metric(2,3)')
```

couplings.py:

```
GC_4 = Coupling(name = 'GC_4',
                value = '-G',
                order = {'QCD':1})
```

vertices.py:

```
V_2 = Vertex(name = 'V_2',
             particles = [ P.G, P.G, P.G ],
             color = [ 'f(1,2,3)' ],
             lorentz = [ L.VVV1 ],
             couplings = {(0,0):C.GC_4})
```

U er a e e O (U O)

particles.py:

```
G = Particle(pdg_code = 21,
          name = 'G',
          antiname = 'G',
          spin = 3,
          color = 8,
          mass = 'ZERO',
          width = 'ZERO',
          texname = 'G',
          antitexname = 'G',
          line = 'curly',
          charge = 0,
          LeptonNumber = 0,
          GhostNumber = 0)
```

lorentz.py:

```
VVV1 = Lorentz(name = 'VVV1',
          spins = [ 3, 3, 3 ],
          Structure =
                    'P(3,1)*Metric(1,2) -
                    P(3,2)*Metric(1,2) -
                    P(2,1)*Metric(1,3) +
                    P(2,3)*Metric(1,3) +
                    P(1,2)*Metric(2,3) -
                    P(1,3)*Metric(2,3)')
```

couplings.py:

```
GC_4 = Coupling(name = 'GC_4',
          value = '-G',
          order = {'QCD':1})
```

vertices.py:

```
V_2 = Vertex(name = 'V_2',
          particles = [ P.G, P.G, P.G ],
          color = [ 'f(1,2,3)' ],
          lorentz = [ L.VVV1 ],
          couplings = {(0,0):C.GC_4})
```

U er a e e O (U O)

particles.py:

```
G = Particle(pdg_code = 21,
          name = 'G',
          antiname = 'G',
          spin = 3,
          color = 8,
          mass = 'ZERO',
          width = 'ZERO',
          texname = 'G',
          antitexname = 'G',
          line = 'curly',
          charge = 0,
          LeptonNumber = 0,
          GhostNumber = 0)
```

lorentz.py:

```
VVV1 = Lorentz(name = 'VVV1',
          spins = [ 3, 3, 3 ],
          Structure =
                    'P(3,1)*Metric(1,2) -
                    P(3,2)*Metric(1,2) -
                    P(2,1)*Metric(1,3) +
                    P(2,3)*Metric(1,3) +
                    P(1,2)*Metric(2,3) -
                    P(1,3)*Metric(2,3)')
```

couplings.py:

```
GC_4 = Coupling(name = 'GC_4',
          value = '-G',
          order = {'QCD':1})
```

vertices.py:

```
V_2 = Vertex(name = 'V_2',
          particles = [ P.G, P.G, P.G ],
          color = [ 'f(1,2,3)' ],
          lorentz = [ L.VVV1 ],
          couplings = {(0,0):C.GC_4})
```

U er a e e O (U O)

particles.py:

```
G = Particle(pdg_code = 21,
             name = 'G',
             antiname = 'G',
             spin = 3,
             color = 8,
             mass = 'ZERO',
             width = 'ZERO',
             texname = 'G',
             antitexname = 'G',
             line = 'curly',
             charge = 0,
             LeptonNumber = 0,
             GhostNumber = 0)
```

lorentz.py:

```
VVV1 = Lorentz(name = 'VVV1',
               spins = [ 3, 3, 3 ],
               Structure =
                         'P(3,1)*Metric(1,2) -
                         P(3,2)*Metric(1,2) -
                         P(2,1)*Metric(1,3) +
                         P(2,3)*Metric(1,3) +
                         P(1,2)*Metric(2,3) -
                         P(1,3)*Metric(2,3)')
```

couplings.py:

```
GC_4 = Coupling(name = 'GC_4',
                value = '-G',
                order = {'QCD':1})
```

vertices.py:

```
V_2 = Vertex(name = 'V_2',
             particles = [ P.G, P.G, P.G ],
             color = [ 'f(1,2,3)' ],
             lorentz = [ L.VVV1 ],
             couplings = {(0,0):C.GC_4})
```

U er a e e O (U O)

particles.py:

```
G = Particle(pdg_code = 21,
             name = 'G',
             antiname = 'G',
             spin = 3,
             color = 8,
             mass = 'ZERO',
             width = 'ZERO',
             texname = 'G',
             antitexname = 'G',
             line = 'curly',
             charge = 0,
             LeptonNumber = 0,
             GhostNumber = 0)
```

lorentz.py:

```
VVV1 = Lorentz(name = 'VVV1',
               spins = [ 3, 3, 3 ],
               Structure =
                           'P(3,1)*Metric(1,2) -
                           P(3,2)*Metric(1,2) -
                           P(2,1)*Metric(1,3) +
                           P(2,3)*Metric(1,3) +
                           P(1,2)*Metric(2,3) -
                           P(1,3)*Metric(2,3)')
```

couplings.py:

```
GC_4 = Coupling(name = 'GC_4',
                value = '-G',
                order = {'QCD':1})
```

vertices.py:

```
V_2 = Vertex(name = 'V_2',
             particles = [ P.G, P.G, P.G ],
             color = [ 'f(1,2,3)' ],
             lorentz = [ L.VVV1 ],
             couplings = {(0,0):C.GC_4})
```

© kathyboast.com
Aloha

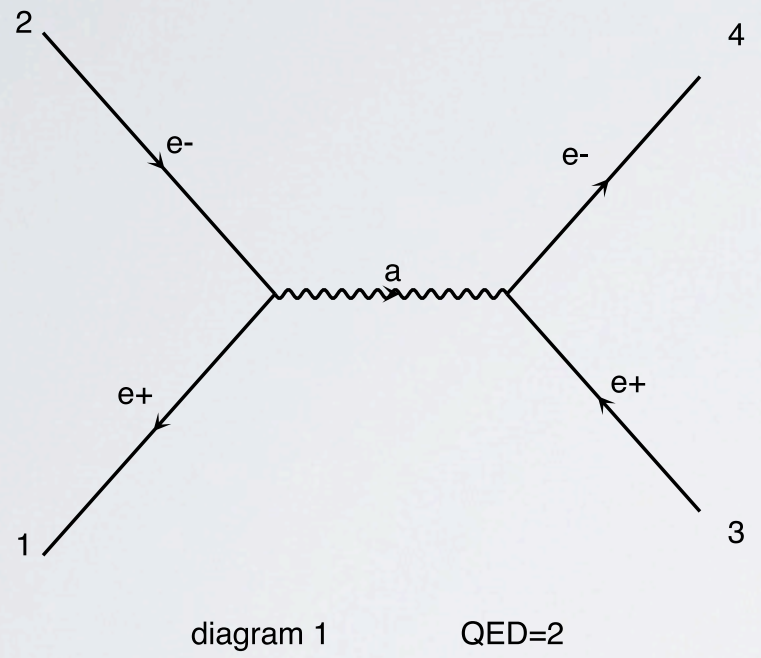
Google is sometimes your friend

PURPOSE OF ALOHA

- ALOHA = Automatic Language-independent Output of Helicity Amplitude.
- Idea: made BSM model implementation 100% automatic from FeynRules to detector simulation.
  - Color: reason for the new color module of MG5
  - HELAS: Need an automation

STANDARD HELAS

- Idea: Evaluate $\mathcal{M}$ for fixed helicity of external particles.



$$M = \bar{u}\gamma^{\mu} v \; P_{\mu\nu} \; \bar{u}\gamma^{\nu} v$$

STANDARD HELAS

- Idea: Evaluate $\mathcal{M}$ for fixed helicity of external particles.

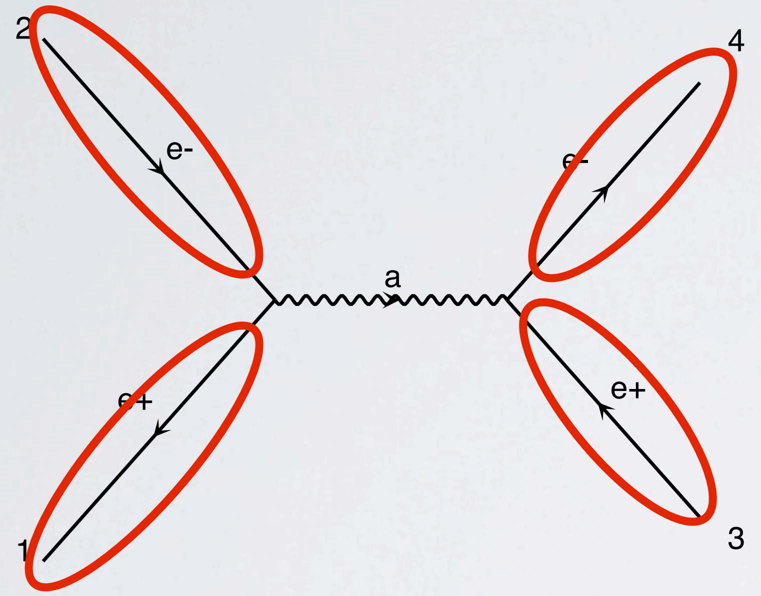


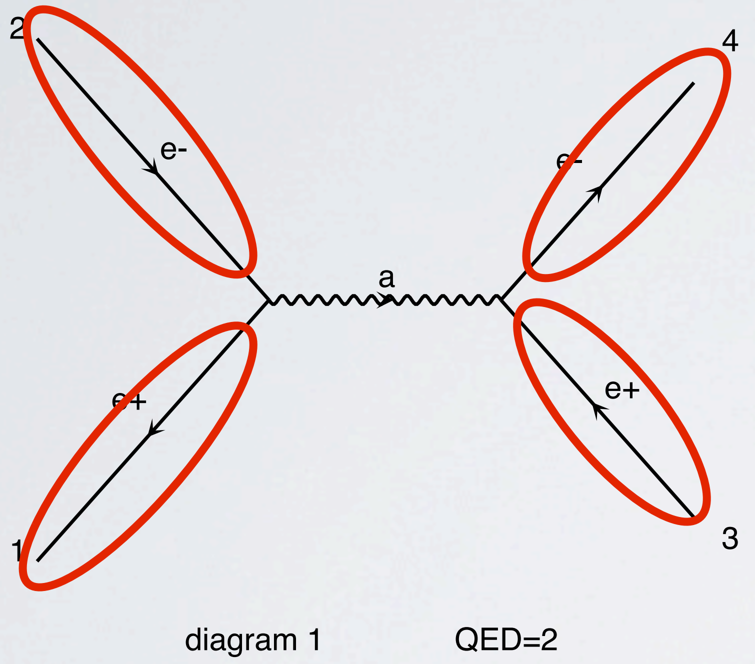
diagram 1 QED=2

$$M = \bar{u}\gamma^\mu v \, P_{\mu\nu} \, \bar{u}\gamma^\nu v$$

$\rightarrow$ Number for a given helicity

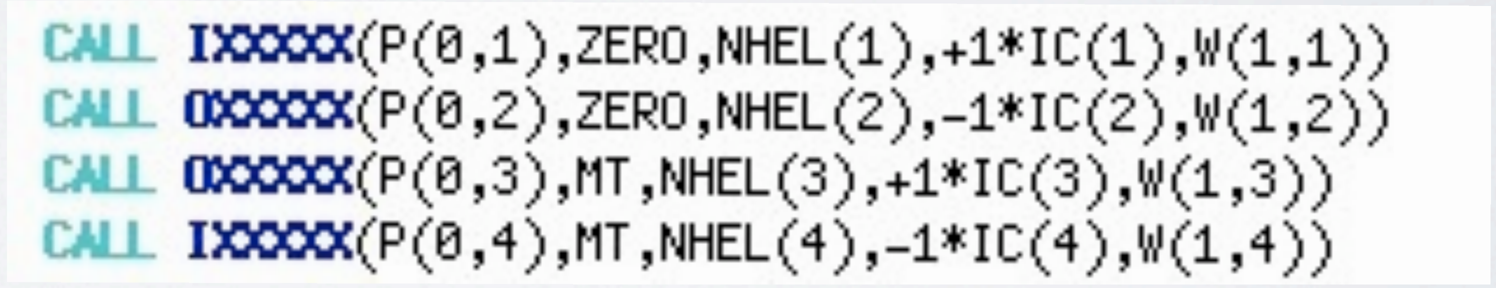
STANDARD HELAS

- Idea: Evaluate $\mathcal{M}$ for fixed helicity of external particles.



$$M = \bar{u}\gamma^\mu v \, P_{\mu\nu} \, \bar{u}\gamma^\nu v$$

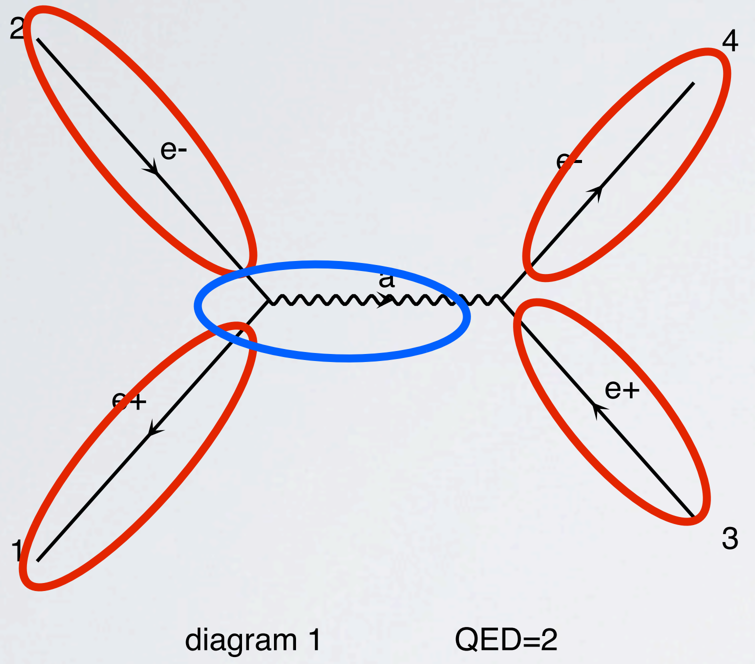
$\rightarrow$ Number for a given helicity



```
CALL IXXXXX(P(0,1),ZERO,NHEL(1),+1*IC(1),W(1,1))
CALL OXXXXX(P(0,2),ZERO,NHEL(2),-1*IC(2),W(1,2))
CALL OXXXXX(P(0,3),MT,NHEL(3),+1*IC(3),W(1,3))
CALL IXXXXX(P(0,4),MT,NHEL(4),-1*IC(4),W(1,4))
```

STANDARD HELAS

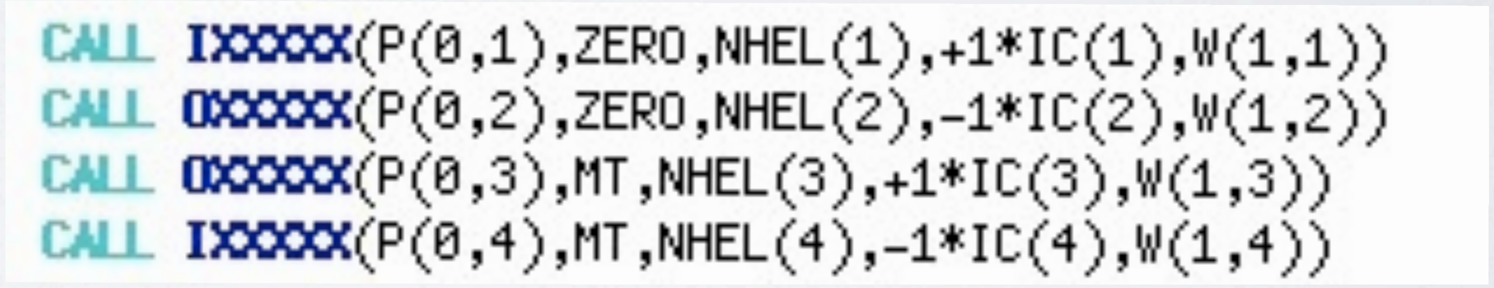
- Idea: Evaluate $\mathcal{M}$ for fixed helicity of external particles.



$$M = \boxed{\bar{u}}\gamma^\mu \boxed{v} P_{\mu\nu} \boxed{\bar{u}}\gamma^\nu \boxed{v}$$

→ Number for a given helicity

→ Evaluate Interaction by interaction



```
CALL IXXXXX(P(0,1),ZERO,NHEL(1),+1*IC(1),W(1,1))
CALL OXXXXX(P(0,2),ZERO,NHEL(2),-1*IC(2),W(1,2))
CALL OXXXXX(P(0,3),MT,NHEL(3),+1*IC(3),W(1,3))
CALL IXXXXX(P(0,4),MT,NHEL(4),-1*IC(4),W(1,4))
```

STANDARD HELAS

- Idea: Evaluate *m* for fixed helicity of external particles.

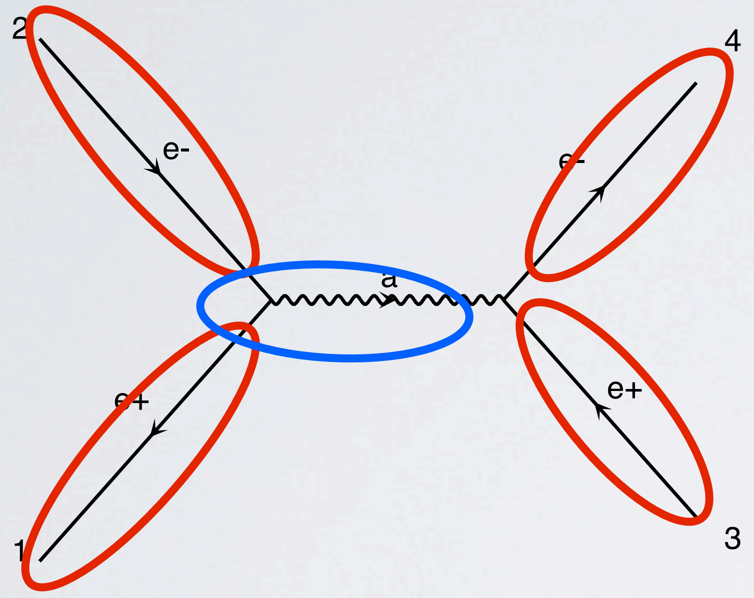
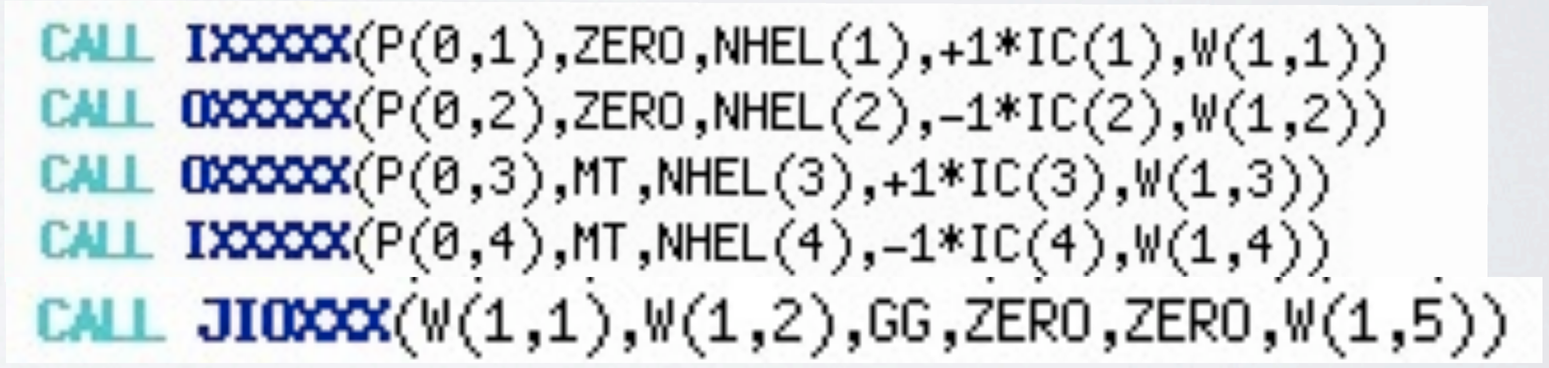


diagram 1 QED=2

$$M = \bar{u}\gamma^{\mu} v \, P_{\mu\nu} \, \bar{u}\gamma^{\nu} v$$

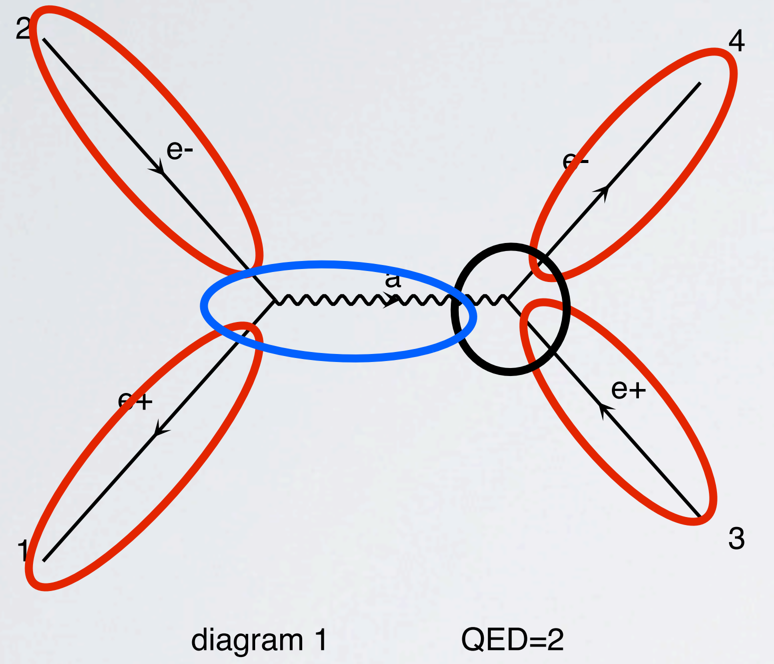
→ Number for a given helicity

→ Evaluate Interaction by interaction


```
CALL IXXXXX(P(0,1),ZERO,NHEL(1),+1*IC(1),W(1,1))
CALL OXXXXX(P(0,2),ZERO,NHEL(2),-1*IC(2),W(1,2))
CALL OXXXXX(P(0,3),MT,NHEL(3),+1*IC(3),W(1,3))
CALL IXXXXX(P(0,4),MT,NHEL(4),-1*IC(4),W(1,4))
CALL JIOXXX(W(1,1),W(1,2),GG,ZERO,ZERO,W(1,5))
```

STANDARD HELAS

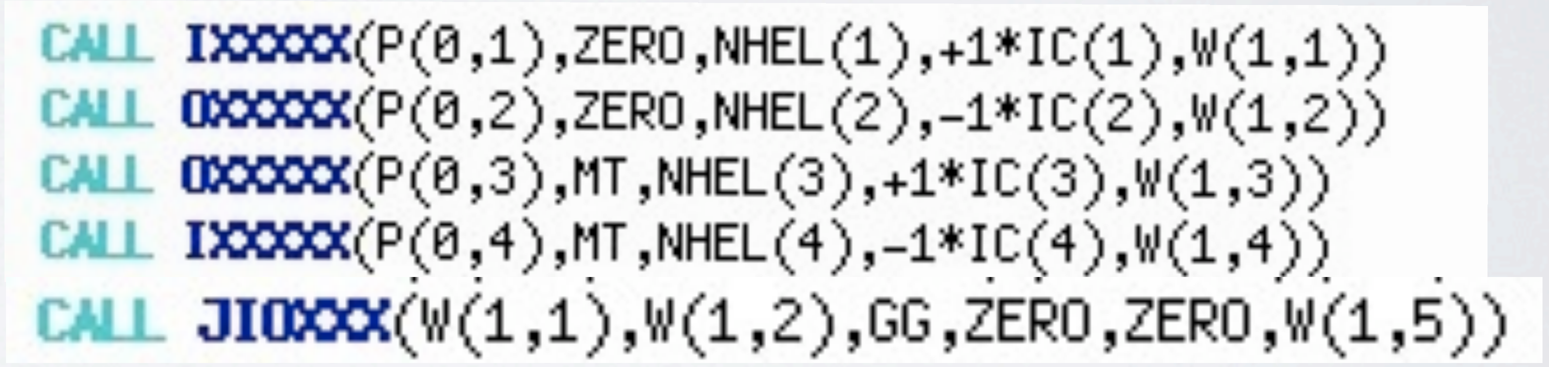
- Idea: Evaluate $\mathcal{M}$ for fixed helicity of external particles.



$$M = \bar{u}\gamma^{\mu} v \, P_{\mu\nu} \, \bar{u}\gamma^{\nu} v$$

→ Number for a given helicity

→ Evaluate Interaction by interaction



```
CALL IXXXXX(P(0,1),ZERO,NHEL(1),+1*IC(1),W(1,1))
CALL OXXXXX(P(0,2),ZERO,NHEL(2),-1*IC(2),W(1,2))
CALL OXXXXX(P(0,3),MT,NHEL(3),+1*IC(3),W(1,3))
CALL IXXXXX(P(0,4),MT,NHEL(4),-1*IC(4),W(1,4))
CALL JIOXXX(W(1,1),W(1,2),GG,ZERO,ZERO,W(1,5))
```

STANDARD HELAS

- Idea: Evaluate $\mathcal{M}$ for fixed helicity of external particles.

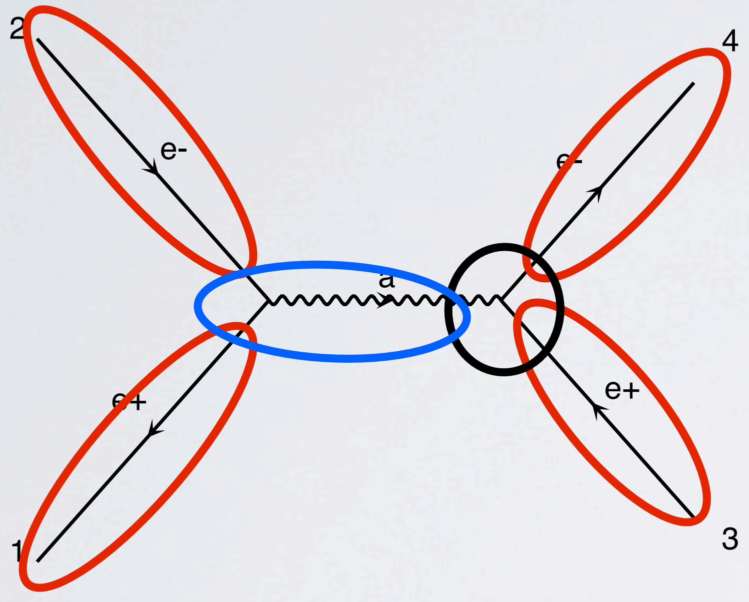
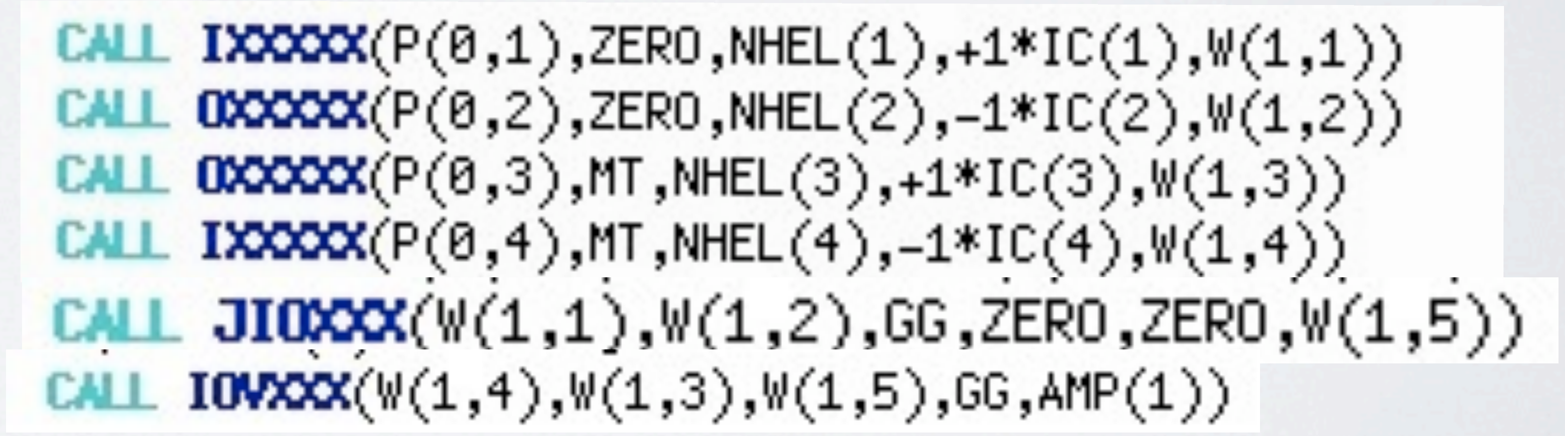


diagram 1 QED=2

$$M = \bar{u}\gamma^{\mu}v\,P_{\mu\nu}\,\bar{u}\gamma^{\nu}v$$

→ Number for a given helicity

→ Evaluate Interaction by interaction



```
CALL IXXXXX(P(0,1),ZERO,NHEL(1),+1*IC(1),W(1,1))
CALL OXXXXX(P(0,2),ZERO,NHEL(2),-1*IC(2),W(1,2))
CALL OXXXXX(P(0,3),MT,NHEL(3),+1*IC(3),W(1,3))
CALL IXXXXX(P(0,4),MT,NHEL(4),-1*IC(4),W(1,4))
CALL JIOXXX(W(1,1),W(1,2),GG,ZERO,ZERO,W(1,5))
CALL IOVXXX(W(1,4),W(1,3),W(1,5),GG,AMP(1))
```

STANDARD HELAS

- Idea: Evaluate $\mathcal{M}$ from impulsions and helicity of external particles.
- 1: Evaluate Wavefunctions of external particles

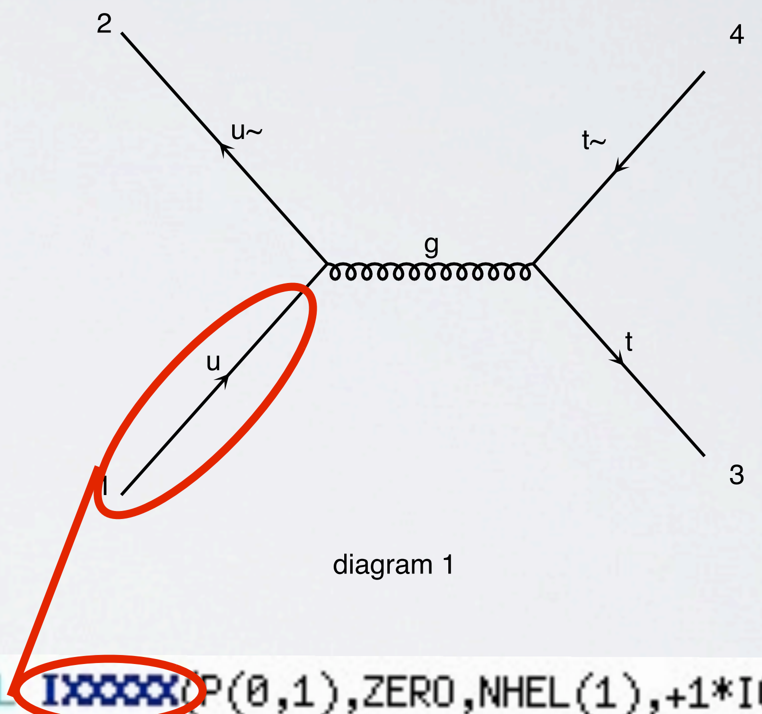
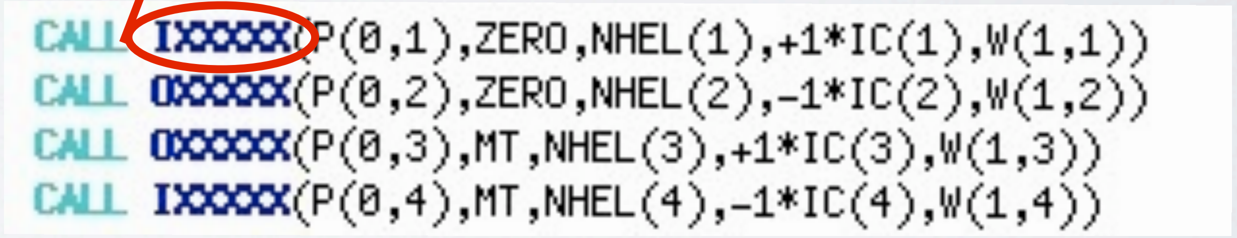


diagram 1



```
CALL IXXXXX(P(0,1),ZERO,NHEL(1),+1*IC(1),W(1,1))
CALL OXXXXX(P(0,2),ZERO,NHEL(2),-1*IC(2),W(1,2))
CALL OXXXXX(P(0,3),MT,NHEL(3),+1*IC(3),W(1,3))
CALL IXXXXX(P(0,4),MT,NHEL(4),-1*IC(4),W(1,4))
```

STANDARD HELAS

- Idea: Evaluate $\mathcal{M}$ from impulsions and helicity of external particles.
- 1: Evaluate Wavefunctions of external particles
- 2 : Evaluate Wavefunctions of internal particles

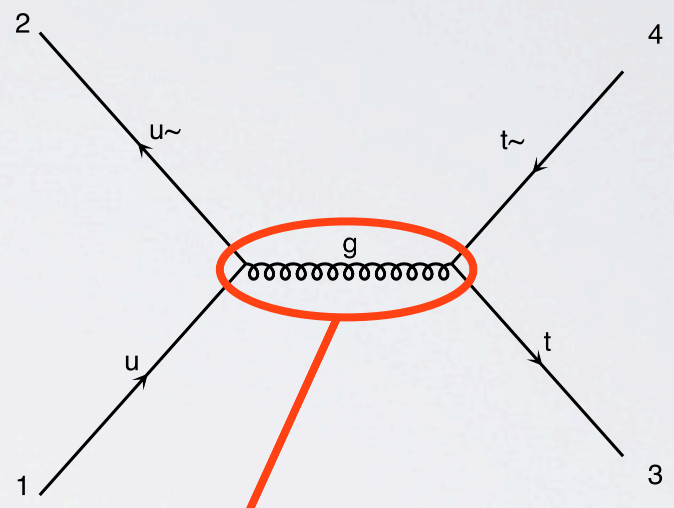
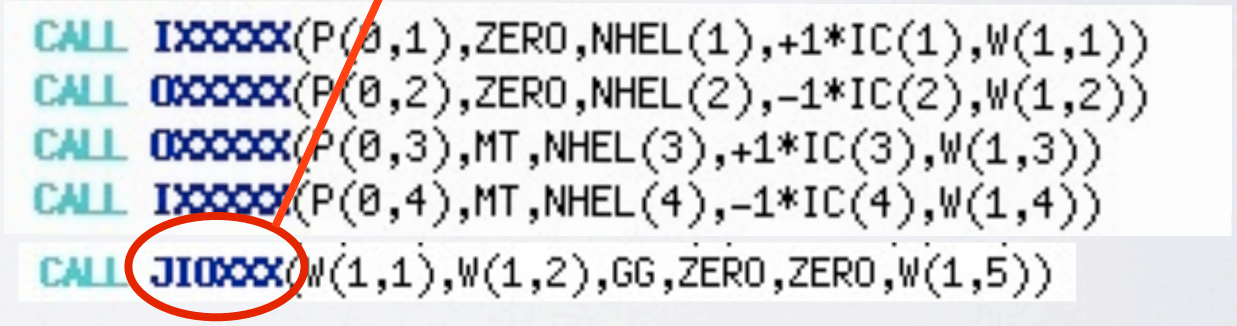


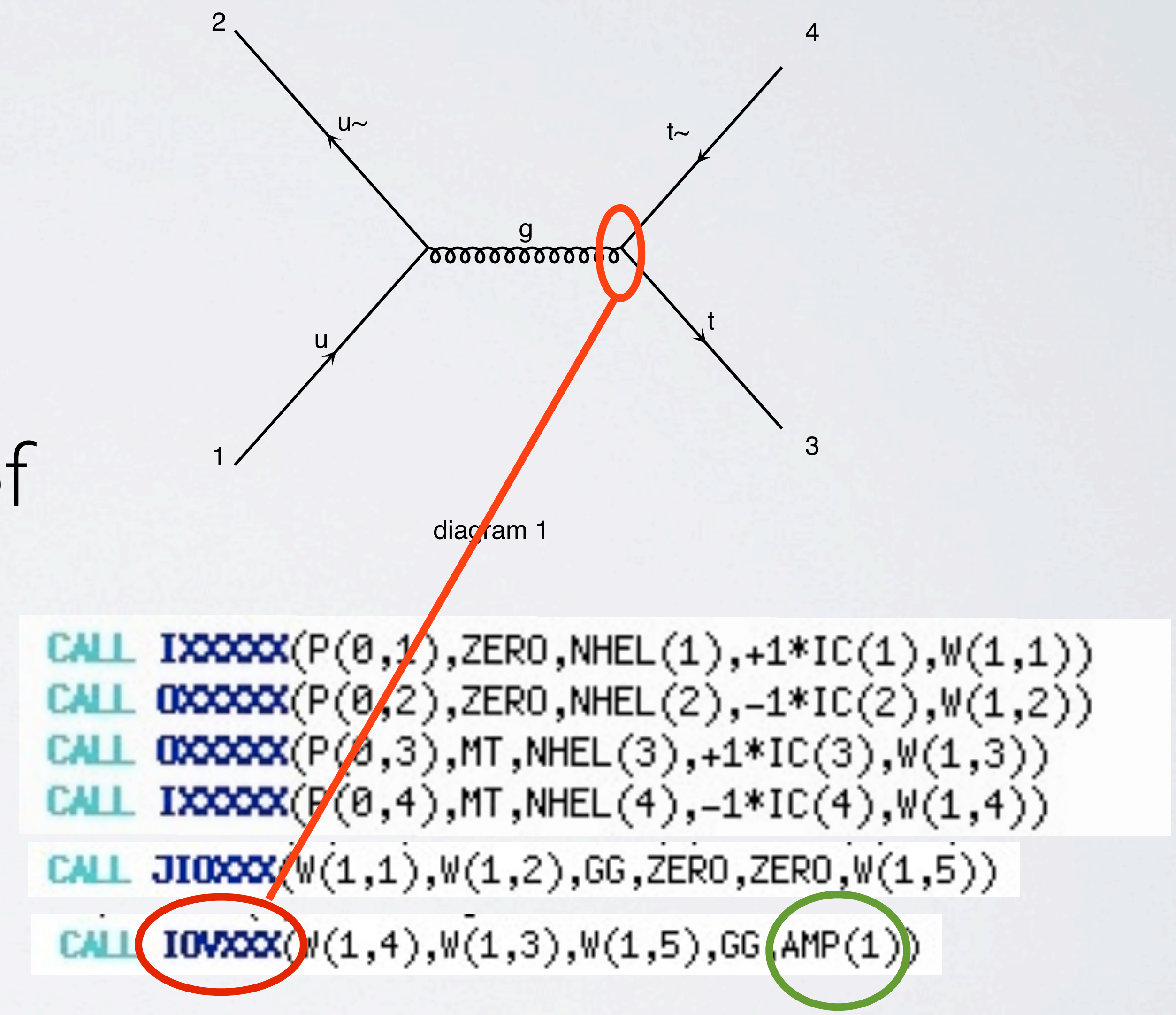
diagram 1



```
CALL IXXXXX(P(0,1),ZERO,NHEL(1),+1*IC(1),W(1,1))
CALL OXXXXX(P(0,2),ZERO,NHEL(2),-1*IC(2),W(1,2))
CALL OXXXXX(P(0,3),MT,NHEL(3),+1*IC(3),W(1,3))
CALL IXXXXX(P(0,4),MT,NHEL(4),-1*IC(4),W(1,4))
CALL JIOXXX(W(1,1),W(1,2),GG,ZERO,ZERO,W(1,5))
```

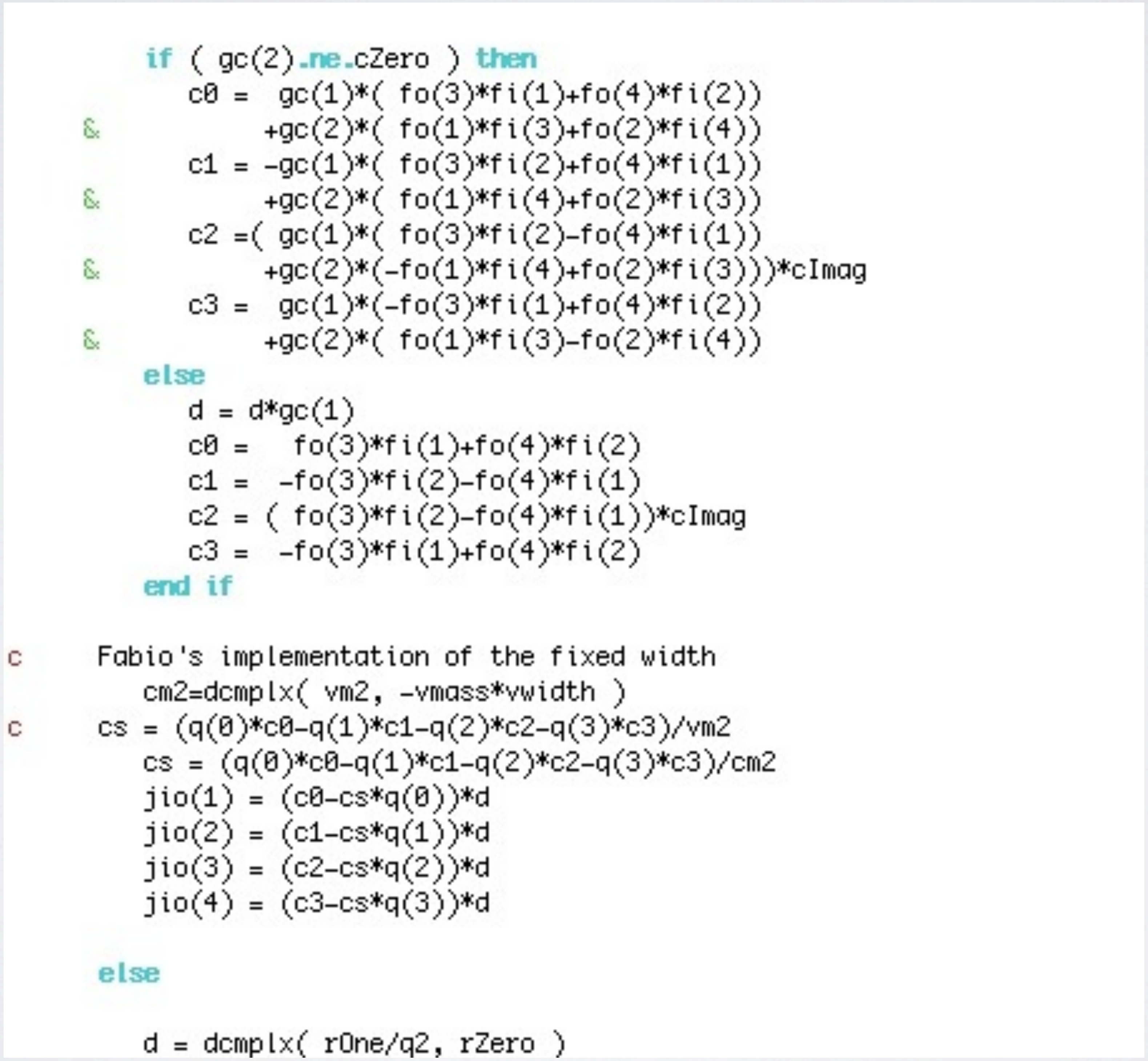
STANDARD HELAS

- Idea: Evaluate $\mathcal{M}$ from impulsions and helicity of external particles.
- 1: Evaluate Wavefunctions of external particles
- 2 : Evaluate Wavefunctions of internal particles
- 3 : Evaluate the Amplitude



```
CALL IXXXXX(P(0,1),ZERO,NHEL(1),+1*IC(1),W(1,1))
CALL OXXXXX(P(0,2),ZERO,NHEL(2),-1*IC(2),W(1,2))
CALL OXXXXX(P(0,3),MT,NHEL(3),+1*IC(3),W(1,3))
CALL IXXXXX(P(0,4),MT,NHEL(4),-1*IC(4),W(1,4))
CALL JIOXXX(W(1,1),W(1,2),GG,ZERO,ZERO,W(1,5))
CALL IOVXXX(W(1,4),W(1,3),W(1,5),GG,AMP(1))
```

ONE HELAS ROUTINE



```
         if ( gc(2).ne.cZero ) then
            c0 =  gc(1)*( fo(3)*fi(1)+fo(4)*fi(2))
     &           +gc(2)*( fo(1)*fi(3)+fo(2)*fi(4))
            c1 = -gc(1)*( fo(3)*fi(2)+fo(4)*fi(1))
     &           +gc(2)*( fo(1)*fi(4)+fo(2)*fi(3))
            c2 =( gc(1)*( fo(3)*fi(2)-fo(4)*fi(1))
     &           +gc(2)*(-fo(1)*fi(4)+fo(2)*fi(3)))*cImag
            c3 =  gc(1)*(-fo(3)*fi(1)+fo(4)*fi(2))
     &           +gc(2)*( fo(1)*fi(3)-fo(2)*fi(4))
         else
            d = d*gc(1)
            c0 =   fo(3)*fi(1)+fo(4)*fi(2)
            c1 =  -fo(3)*fi(2)-fo(4)*fi(1)
            c2 = ( fo(3)*fi(2)-fo(4)*fi(1))*cImag
            c3 =  -fo(3)*fi(1)+fo(4)*fi(2)
         end if

c     Fabio's implementation of the fixed width
         cm2=dcmplx( vm2, -vmass*vwidth )
c     cs = (q(0)*c0-q(1)*c1-q(2)*c2-q(3)*c3)/vm2
         cs = (q(0)*c0-q(1)*c1-q(2)*c2-q(3)*c3)/cm2
         jio(1) = (c0-cs*q(0))*d
         jio(2) = (c1-cs*q(1))*d
         jio(3) = (c2-cs*q(2))*d
         jio(4) = (c3-cs*q(3))*d

      else

         d = dcmplx( rOne/q2, rZero )
```

PHYSICAL CONTENT

- Lorentz structure associated to "e+ e- A" vertex is γ^μ

- So the Associate Amplitude (IOV) will be:

$$-i\, W_f(e^-)\, \gamma^\mu\, W_f(e^+)\, A_\mu$$

- And the computation of the vector wavefunctions (JIO) is

$$W_f(e^-)\, \gamma^\mu\, W_f(e^+)\, \frac{-i\, \eta_{\mu\nu}}{p_A^2}$$

- From the Lorentz structure it's easy to compute automaticaly the HELAS routine

PHYSICAL CONTENT

- Lorentz structure associated to "e+ e- A" vertex is γ^μ
- So the Associate Amplitude (IOV) will be:

$$-i\, W_f(e^-)\, \gamma^\mu\, W_f(e^+)\, A_\mu$$

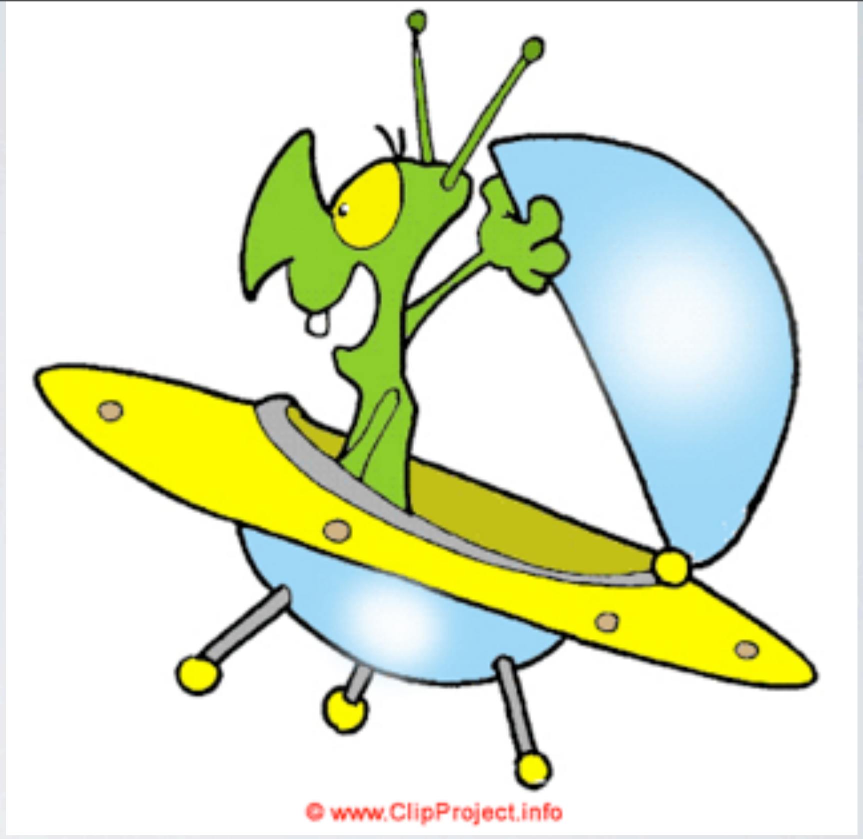
- And the computation of the vector wavefunctions (JIO) is

$$W_f(e^-)\, \gamma^\mu\, W_f(e^+)\, \frac{-i\,\eta_{\mu\nu}}{p_A^2}$$

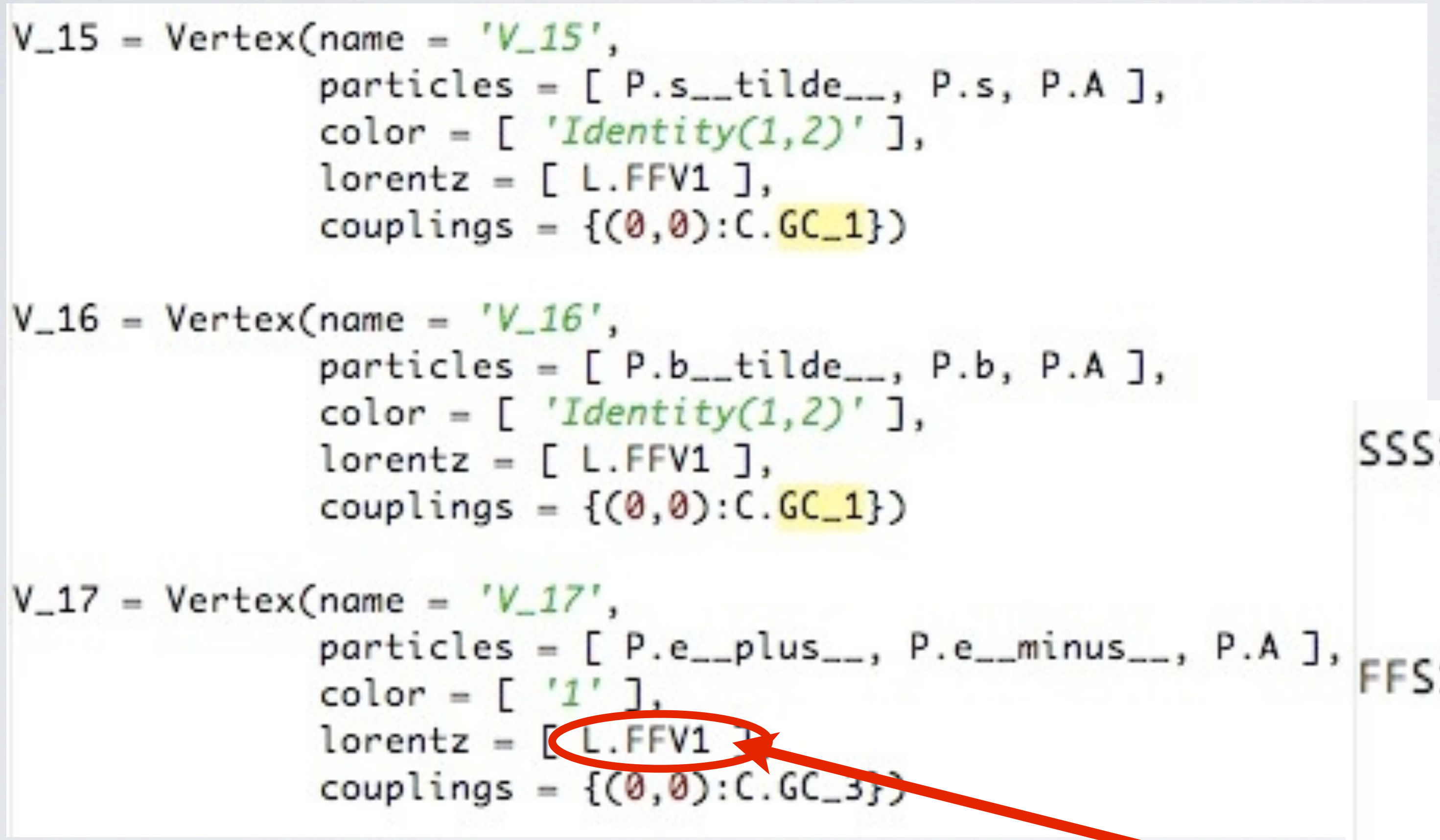
- From the Lorentz structure it's easy to compute automaticaly the HELAS routine

Hard to do in Python

UFO

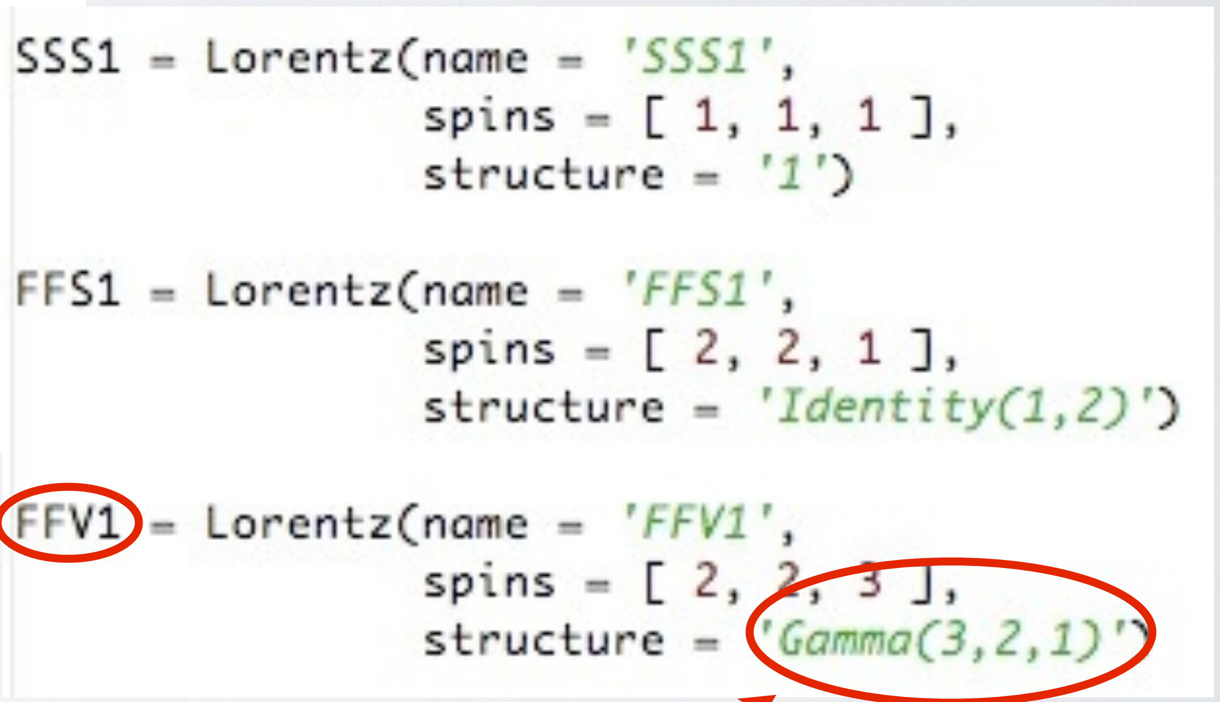


Vertices.py



```
V_15 = Vertex(name = 'V_15',
              particles = [ P.s__tilde__, P.s, P.A ],
              color = [ 'Identity(1,2)' ],
              lorentz = [ L.FFV1 ],
              couplings = {(0,0):C.GC_1})

V_16 = Vertex(name = 'V_16',
              particles = [ P.b__tilde__, P.b, P.A ],
              color = [ 'Identity(1,2)' ],
              lorentz = [ L.FFV1 ],
              couplings = {(0,0):C.GC_1})

V_17 = Vertex(name = 'V_17',
              particles = [ P.e__plus__, P.e__minus__, P.A ],
              color = [ '1' ],
              lorentz = [ L.FFV1 ],
              couplings = {(0,0):C.GC_3})
```

Lorentz.py

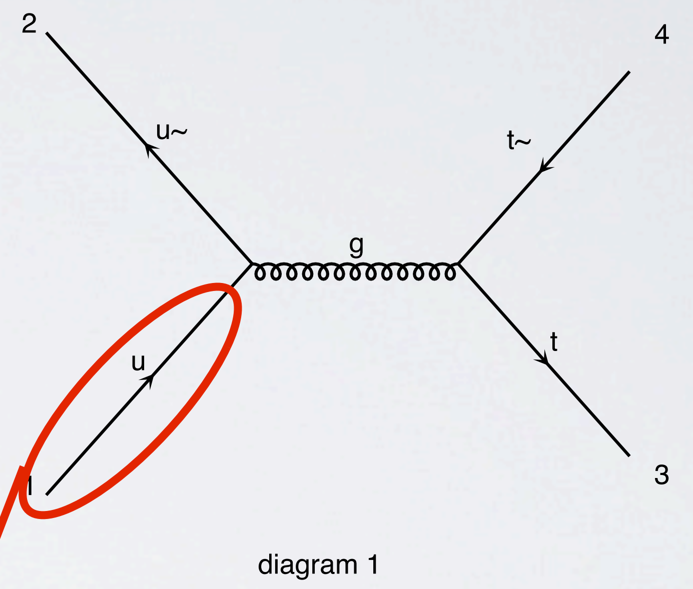


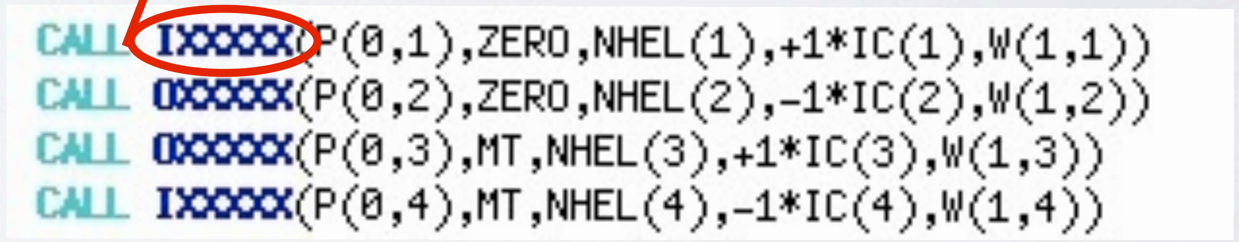
```
SSS1 = Lorentz(name = 'SSS1',
               spins = [ 1, 1, 1 ],
               structure = '1')

FFS1 = Lorentz(name = 'FFS1',
               spins = [ 2, 2, 1 ],
               structure = 'Identity(1,2)')

FFV1 = Lorentz(name = 'FFV1',
               spins = [ 2, 2, 3 ],
               structure = 'Gamma(3,2,1)')
```

γ^μ

ALOHA

- Idea: Evaluate $\mathcal{M}$ from impulsions and helicity of external particles.
- 1: Evaluate Wavefunctions of external particles

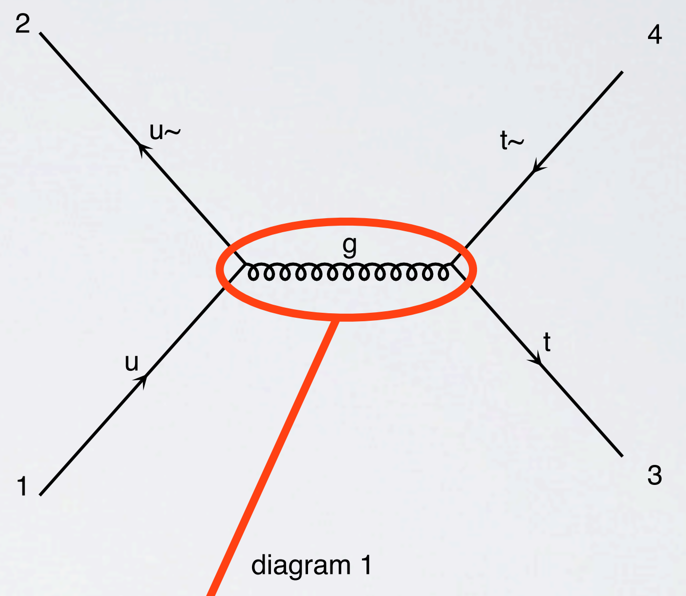


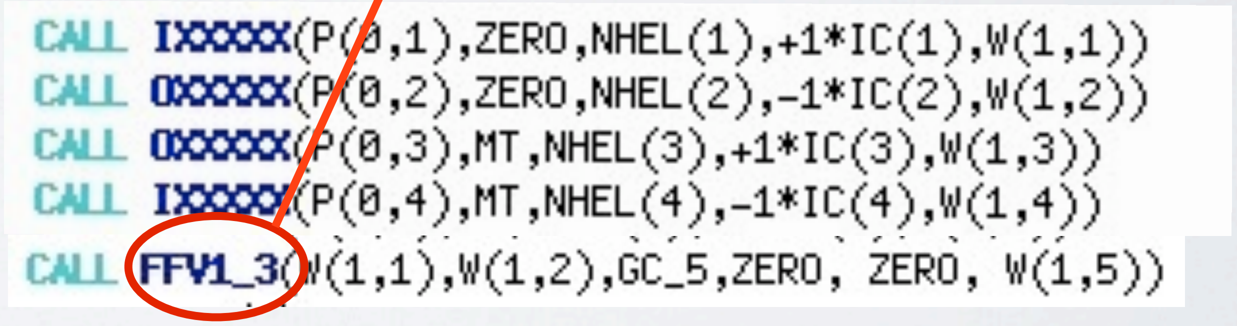


```
CALL IXXXXX(P(0,1),ZERO,NHEL(1),+1*IC(1),W(1,1))
CALL OXXXXX(P(0,2),ZERO,NHEL(2),-1*IC(2),W(1,2))
CALL OXXXXX(P(0,3),MT,NHEL(3),+1*IC(3),W(1,3))
CALL IXXXXX(P(0,4),MT,NHEL(4),-1*IC(4),W(1,4))
```

STANDARD HELAS

- Idea: Evaluate $\mathcal{M}$ from impulsions and helicity of external particles.
- 1: Evaluate Wavefunctions of external particles
- 2 : Evaluate Wavefunctions of internal particles





```
CALL IXXXXX(P(0,1),ZERO,NHEL(1),+1*IC(1),W(1,1))
CALL OXXXXX(P(0,2),ZERO,NHEL(2),-1*IC(2),W(1,2))
CALL OXXXXX(P(0,3),MT,NHEL(3),+1*IC(3),W(1,3))
CALL IXXXXX(P(0,4),MT,NHEL(4),-1*IC(4),W(1,4))
CALL FFV1_3(W(1,1),W(1,2),GC_5,ZERO, ZERO, W(1,5))
```

STANDARD HELAS

- Idea: Evaluate $\mathcal{M}$ from impulsions and helicity of external particles.
- 1: Evaluate Wavefunctions of external particles
- 2 : Evaluate Wavefunctions of internal particles

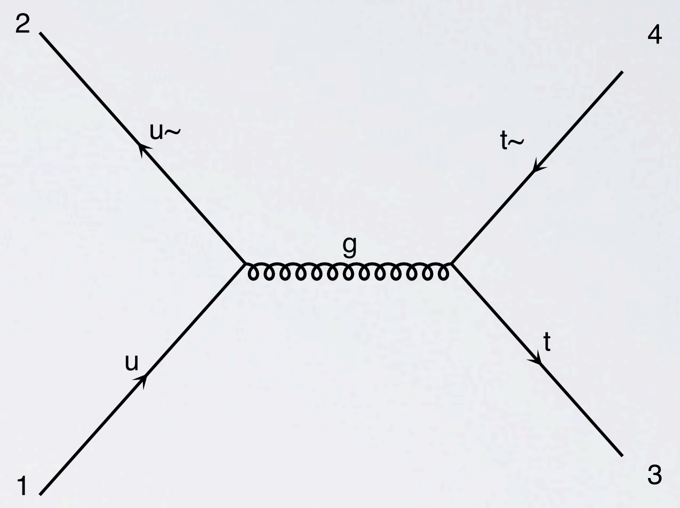
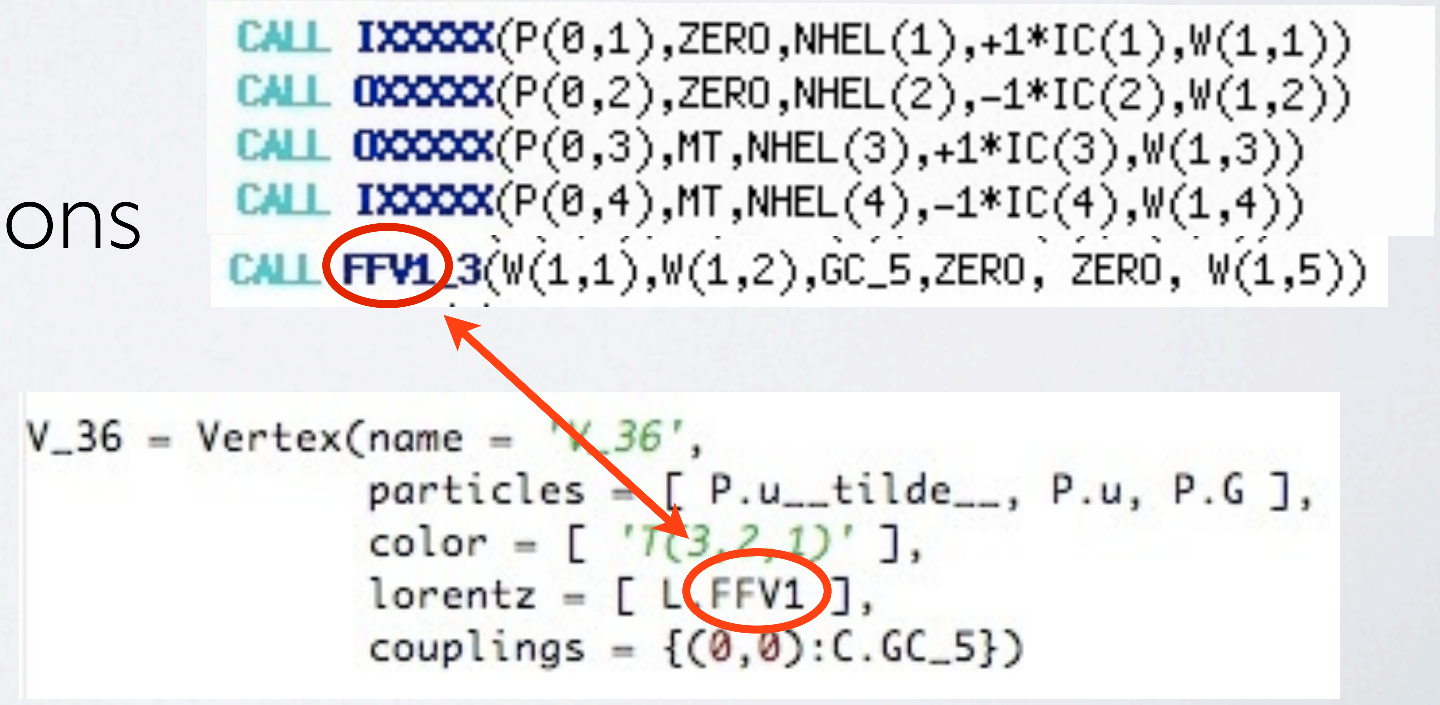


diagram 1



```
CALL IXXXXX(P(0,1),ZERO,NHEL(1),+1*IC(1),W(1,1))
CALL OXXXXX(P(0,2),ZERO,NHEL(2),-1*IC(2),W(1,2))
CALL OXXXXX(P(0,3),MT,NHEL(3),+1*IC(3),W(1,3))
CALL IXXXXX(P(0,4),MT,NHEL(4),-1*IC(4),W(1,4))
CALL FFV1_3(W(1,1),W(1,2),GC_5,ZERO, ZERO, W(1,5))
```

```
V_36 = Vertex(name = 'V_36',
              particles = [ P.u__tilde__, P.u, P.G ],
              color = [ 'T(3,2,1)' ],
              lorentz = [ L.FFV1 ],
              couplings = {(0,0):C.GC_5})
```

STANDARD HELAS

- Idea: Evaluate $\mathcal{M}$ from impulsions and helicity of external particles.
- 1: Evaluate Wavefunctions of external particles
- 2 : Evaluate Wavefunctions of internal particles

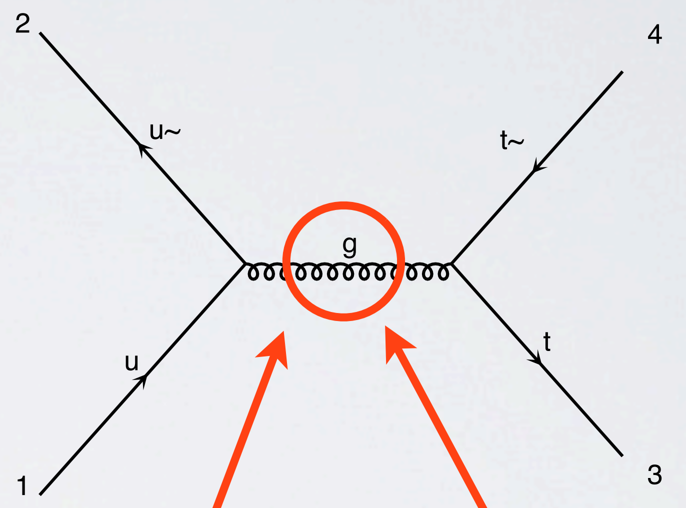
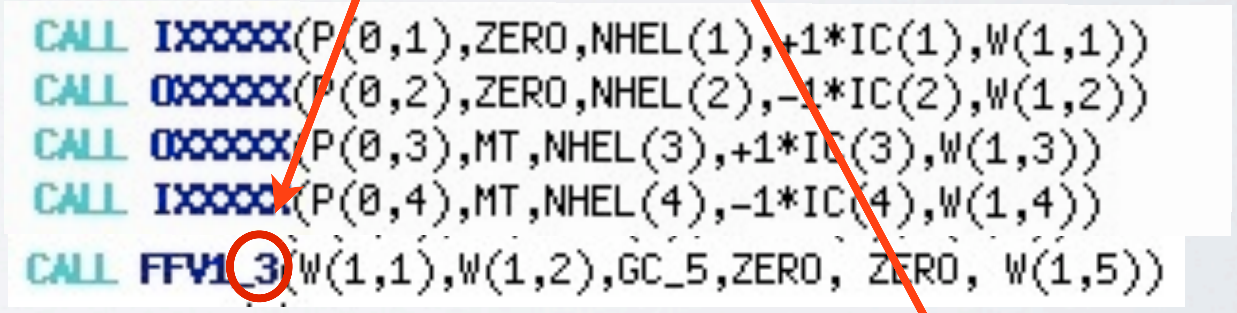
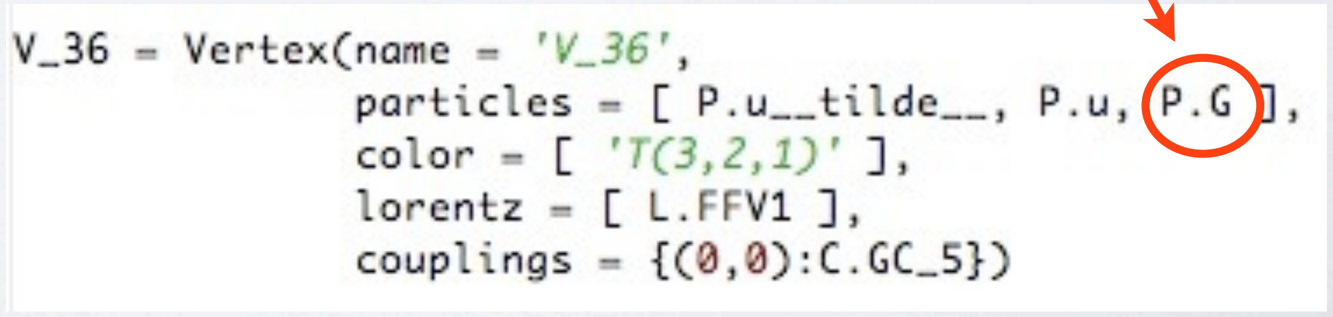


diagram 1



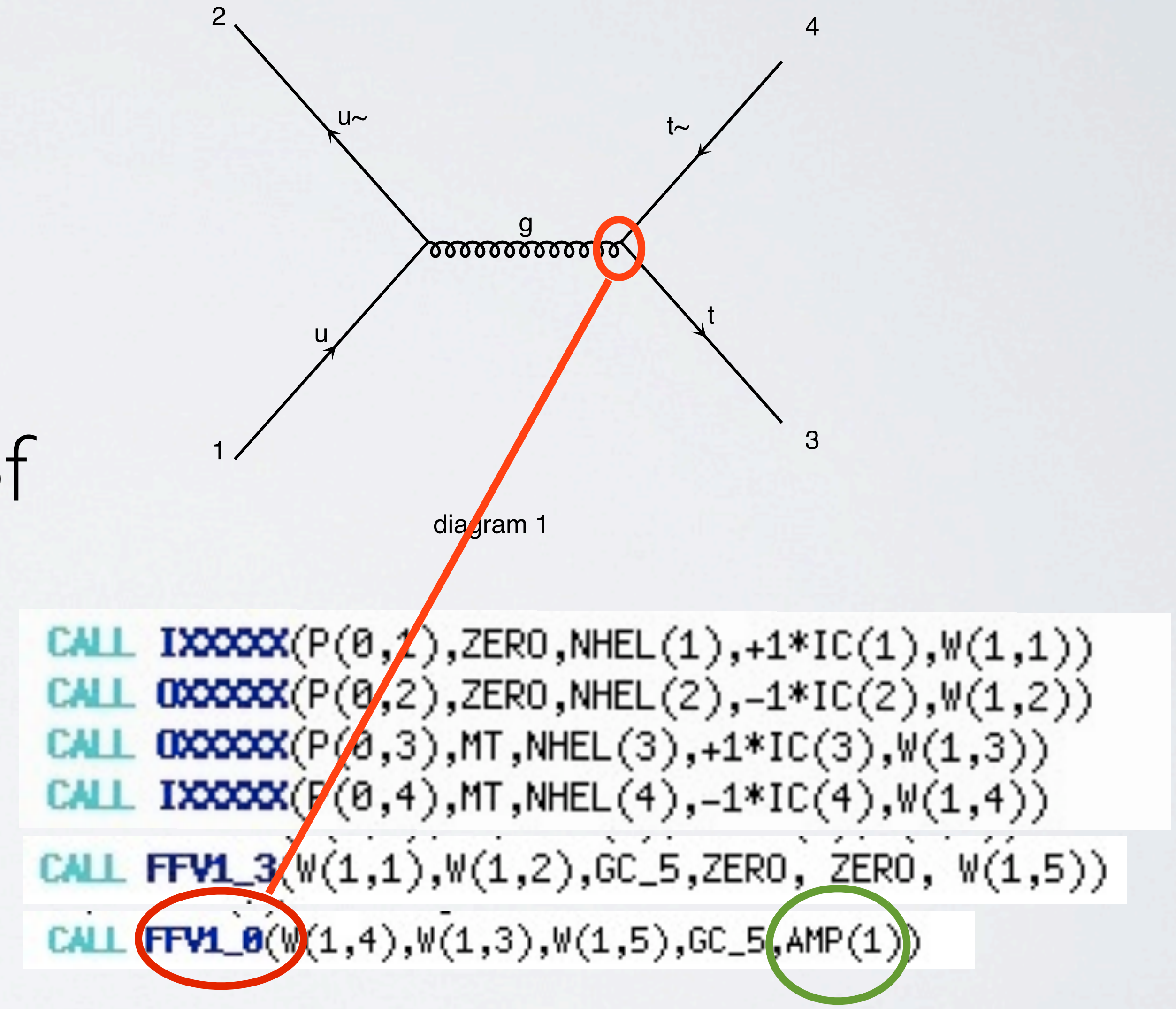
```
CALL IXXXXX(P(0,1),ZERO,NHEL(1),+1*IC(1),W(1,1))
CALL OXXXXX(P(0,2),ZERO,NHEL(2),-1*IC(2),W(1,2))
CALL OXXXXX(P(0,3),MT,NHEL(3),+1*IC(3),W(1,3))
CALL IXXXXX(P(0,4),MT,NHEL(4),-1*IC(4),W(1,4))
CALL FFV1_3(W(1,1),W(1,2),GC_5,ZERO, ZERO, W(1,5))
```



```
V_36 = Vertex(name = 'V_36',
              particles = [ P.u__tilde__, P.u, P.G ],
              color = [ 'T(3,2,1)' ],
              lorentz = [ L.FFV1 ],
              couplings = {(0,0):C.GC_5})
```

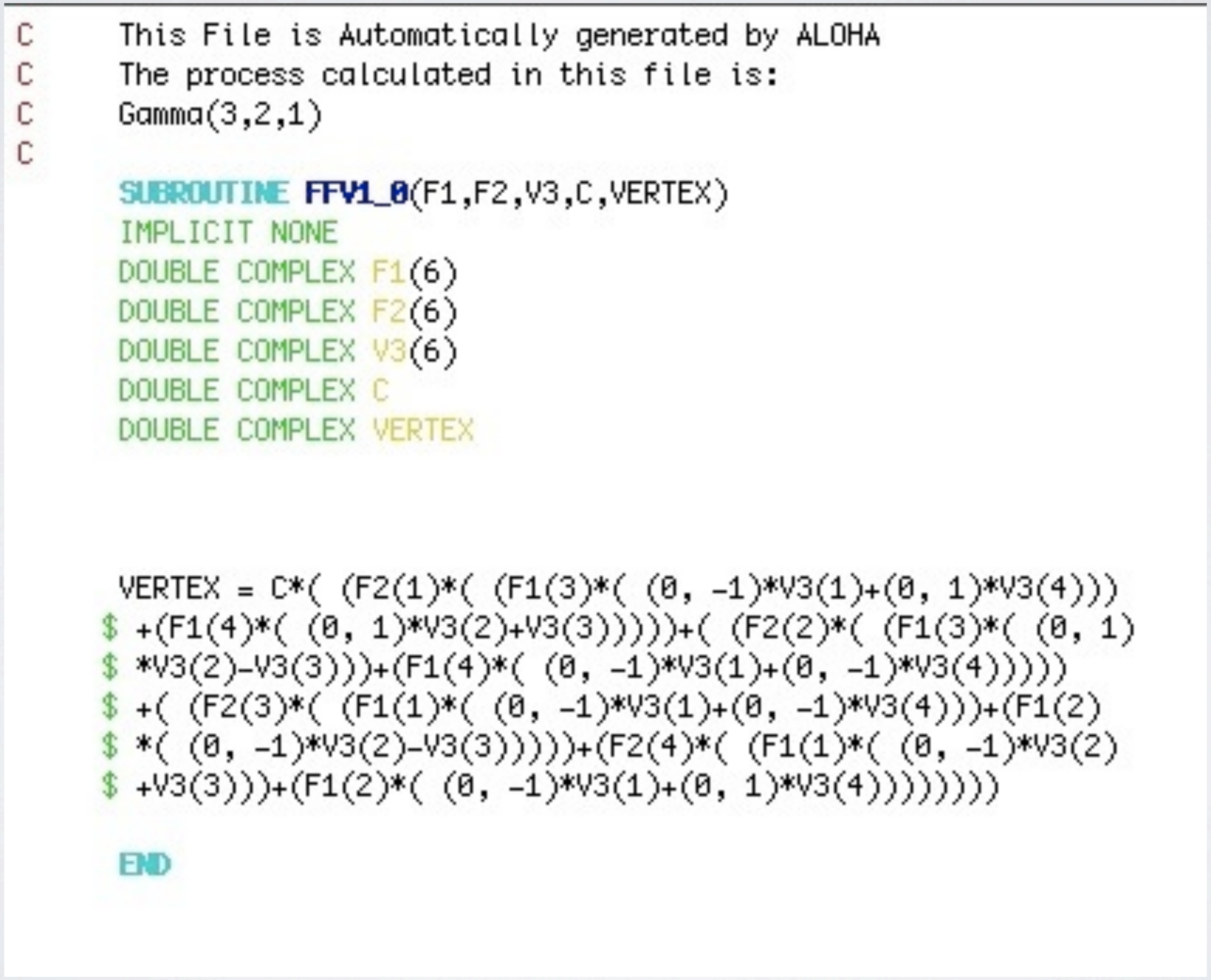
STANDARD HELAS

- Idea: Evaluate $\mathcal{M}$ from impulsions and helicity of external particles.
- 1: Evaluate Wavefunctions of external particles
- 2 : Evaluate Wavefunctions of internal particles
- 3 : Evaluate the Amplitude



```
CALL IXXXXX(P(0,1),ZERO,NHEL(1),+1*IC(1),W(1,1))
CALL OXXXXX(P(0,2),ZERO,NHEL(2),-1*IC(2),W(1,2))
CALL OXXXXX(P(0,3),MT,NHEL(3),+1*IC(3),W(1,3))
CALL IXXXXX(P(0,4),MT,NHEL(4),-1*IC(4),W(1,4))
CALL FFV1_3(W(1,1),W(1,2),GC_5,ZERO, ZERO, W(1,5))
CALL FFV1_0(W(1,4),W(1,3),W(1,5),GC_5,AMP(1))
```

ONE ALOHA ROUTINE



```
C       This File is Automatically generated by ALOHA
C       The process calculated in this file is:
C       Gamma(3,2,1)
C
        SUBROUTINE FFV1_0(F1,F2,V3,C,VERTEX)
        IMPLICIT NONE
        DOUBLE COMPLEX F1(6)
        DOUBLE COMPLEX F2(6)
        DOUBLE COMPLEX V3(6)
        DOUBLE COMPLEX C
        DOUBLE COMPLEX VERTEX



        VERTEX = C*( (F2(1)*( (F1(3)*( (0, -1)*V3(1)+(0, 1)*V3(4)))
       $ +(F1(4)*( (0, 1)*V3(2)+V3(3)))))+( (F2(2)*( (F1(3)*( (0, 1)
       $ *V3(2)-V3(3)))+(F1(4)*( (0, -1)*V3(1)+(0, -1)*V3(4)))))
       $ +( (F2(3)*( (F1(1)*( (0, -1)*V3(1)+(0, -1)*V3(4)))+(F1(2)
       $ *( (0, -1)*V3(2)-V3(3)))))+(F2(4)*( (F1(1)*( (0, -1)*V3(2)
       $ +V3(3)))+(F1(2)*( (0, -1)*V3(1)+(0, 1)*V3(4)))))))

        END
```

ALOHA FEATURE

ALOHA FEATURE

- ALOHA IS PURE PYTHON and standalone

ALOHA FEATURE

- ALOHA IS PURE PYTHON and standalone
- ALOHA IS FAST

ALOHA FEATURE

- ALOHA IS PURE PYTHON and standalone
- ALOHA IS FAST
  - SM in 3s and MSSM in 5s

ALOHA FEATURE

- ALOHA IS PURE PYTHON and standalone
- ALOHA IS FAST
  - SM in 3s and MSSM in 5s
- Possible to ask a subset of routine (Done in MG5)

ALOHA FEATURE

- ALOHA IS PURE PYTHON and standalone
- ALOHA IS FAST
  - SM in 3s and MSSM in 5s
- Possible to ask a subset of routine (Done in MG5)
- Output in Python / Fortran / C

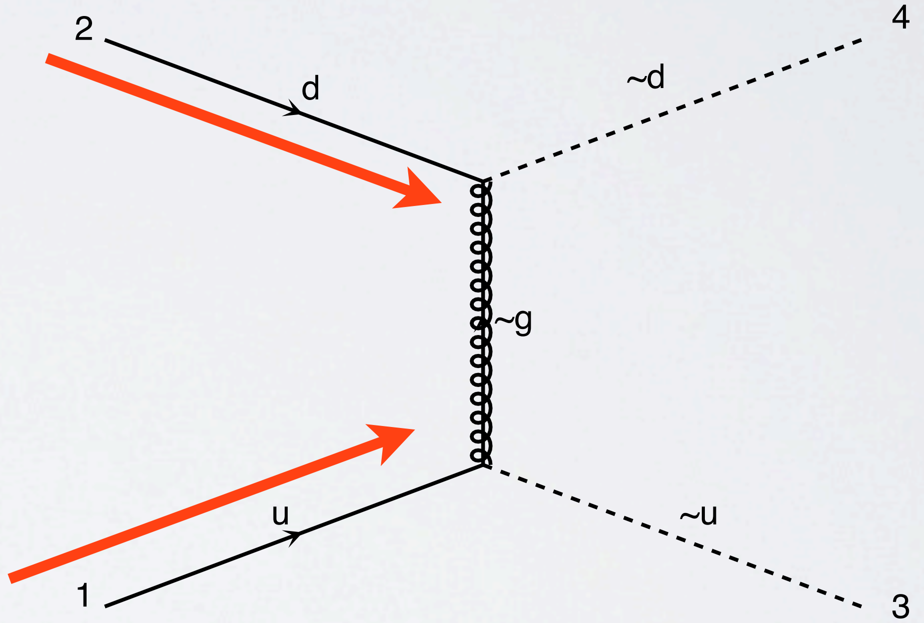
ALOHA FEATURE

- ALOHA IS PURE PYTHON and standalone
- ALOHA IS FAST
  - SM in 3s and MSSM in 5s
- Possible to ask a subset of routine (Done in MG5)
- Output in Python / Fortran / C
- Particles spin implemented Scalar Fermion Vector Spin2

SPECIAL ROUTINE

- Fermion clashes routine

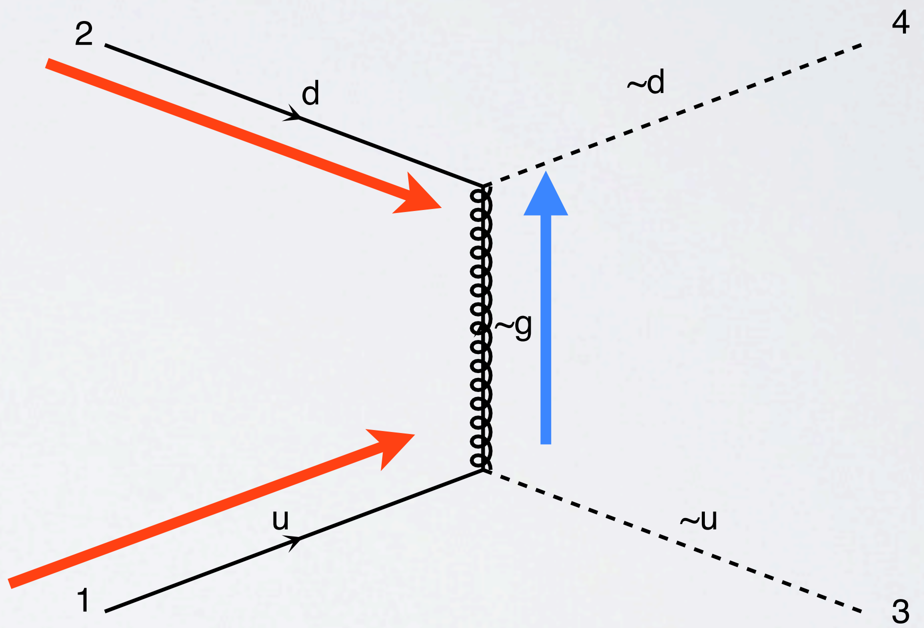
Denner's method



SPECIAL ROUTINE

- Fermion clashes routine

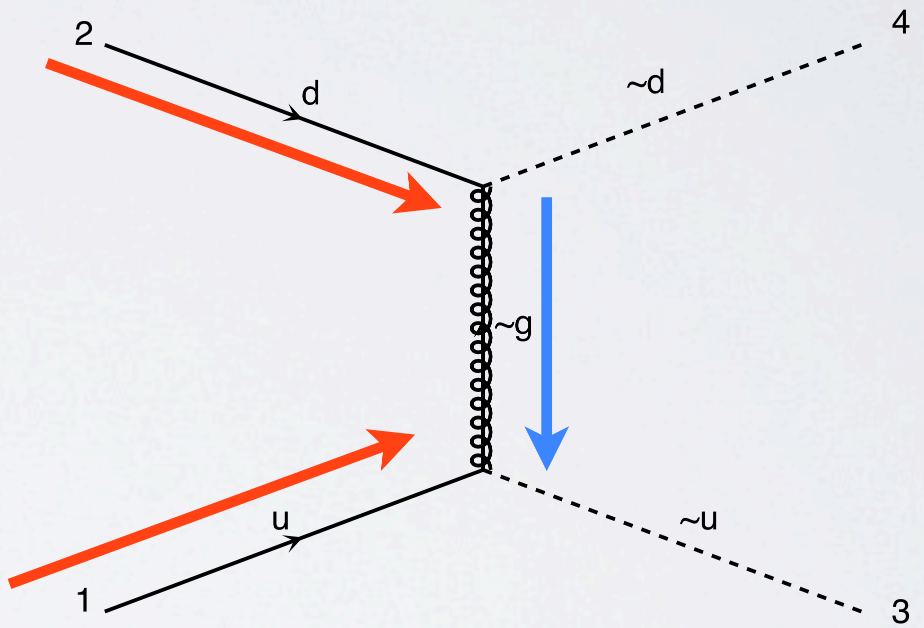
Denner's method



SPECIAL ROUTINE

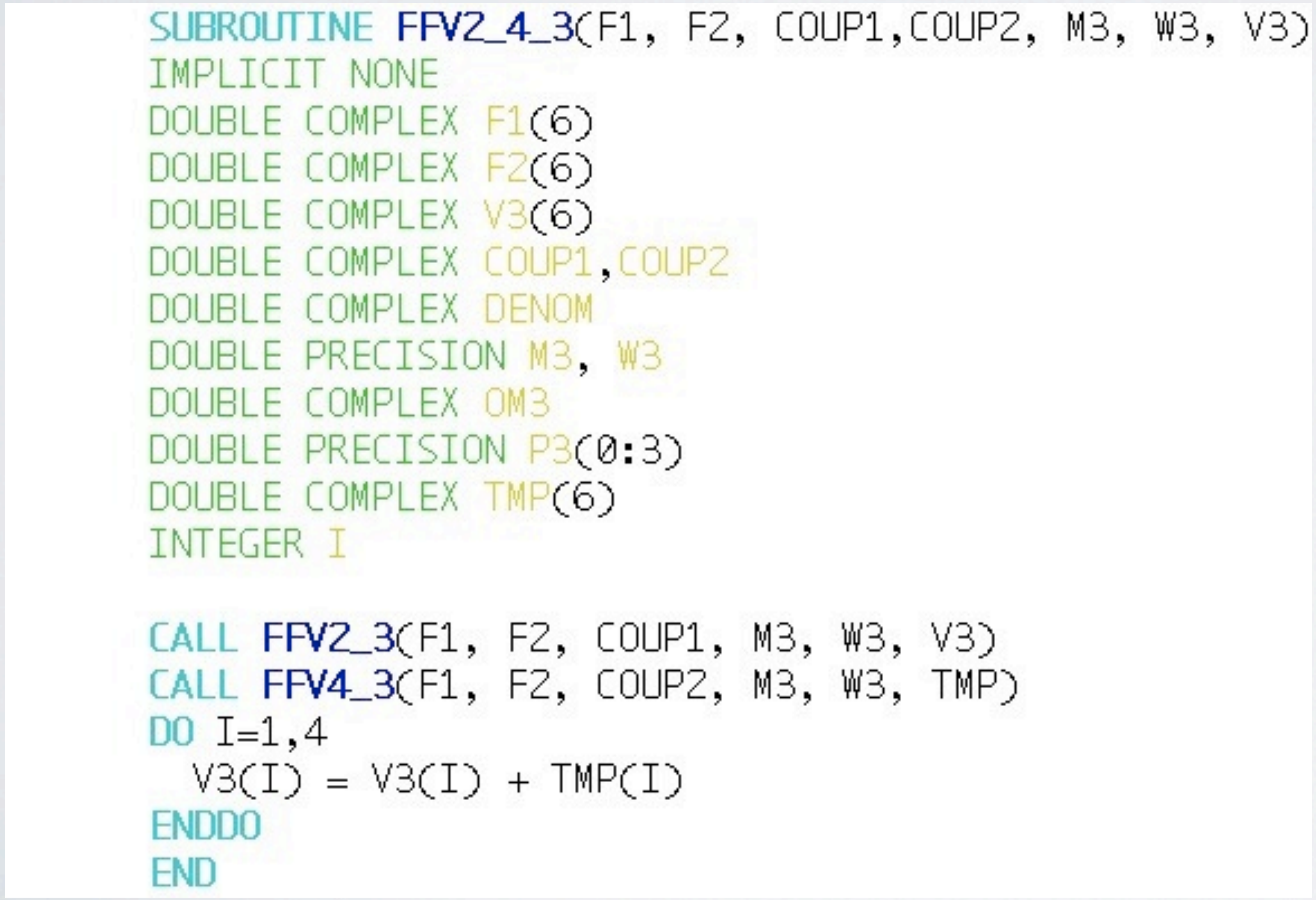
- Fermion clashes routine

Denner's method



SPECIAL ROUTINE

- Fermion clashes routine: Denner's method
- Multi ALOHA routine:



```
SUBROUTINE FFV2_4_3(F1, F2, COUP1,COUP2, M3, W3, V3)
IMPLICIT NONE
DOUBLE COMPLEX F1(6)
DOUBLE COMPLEX F2(6)
DOUBLE COMPLEX V3(6)
DOUBLE COMPLEX COUP1,COUP2
DOUBLE COMPLEX DENOM
DOUBLE PRECISION M3, W3
DOUBLE COMPLEX OM3
DOUBLE PRECISION P3(0:3)
DOUBLE COMPLEX TMP(6)
INTEGER I

CALL FFV2_3(F1, F2, COUP1, M3, W3, V3)
CALL FFV4_3(F1, F2, COUP2, M3, W3, TMP)
DO I=1,4
  V3(I) = V3(I) + TMP(I)
ENDDO
END
```

VERY IMPORTANT
for Z vertex

UFO/ALOHA IN MG5

UFO/ALOHA IN MG5

- UFO is the default model

UFO/ALOHA IN MG5

- UFO is the default model
- Only Model with Full Option

UFO/ALOHA IN MG5

- UFO is the default model
- Only Model with Full Option
- MG5 provides a way to display the UFO model

UFO/ALOHA IN MG5

- UFO is the default model
- Only Model with Full Option
- MG5 provides a way to display the UFO model
- MG5 provides a way to restrict the UFO model

UFO/ALOHA IN MG5

- UFO is the default model
- Only Model with Full Option
- MG5 provides a way to display the UFO model
- MG5 provides a way to restrict the UFO model
- MG5 is able to convert the UFO model to fortran/C++

UFO/ALOHA IN MG5

- UFO is the default model
- Only Model with Full Option
- MG5 provides a way to display the UFO model
- MG5 provides a way to restrict the UFO model
- MG5 is able to convert the UFO model to fortran/C++
- ALOHA is call on the fly, producing only the require routine

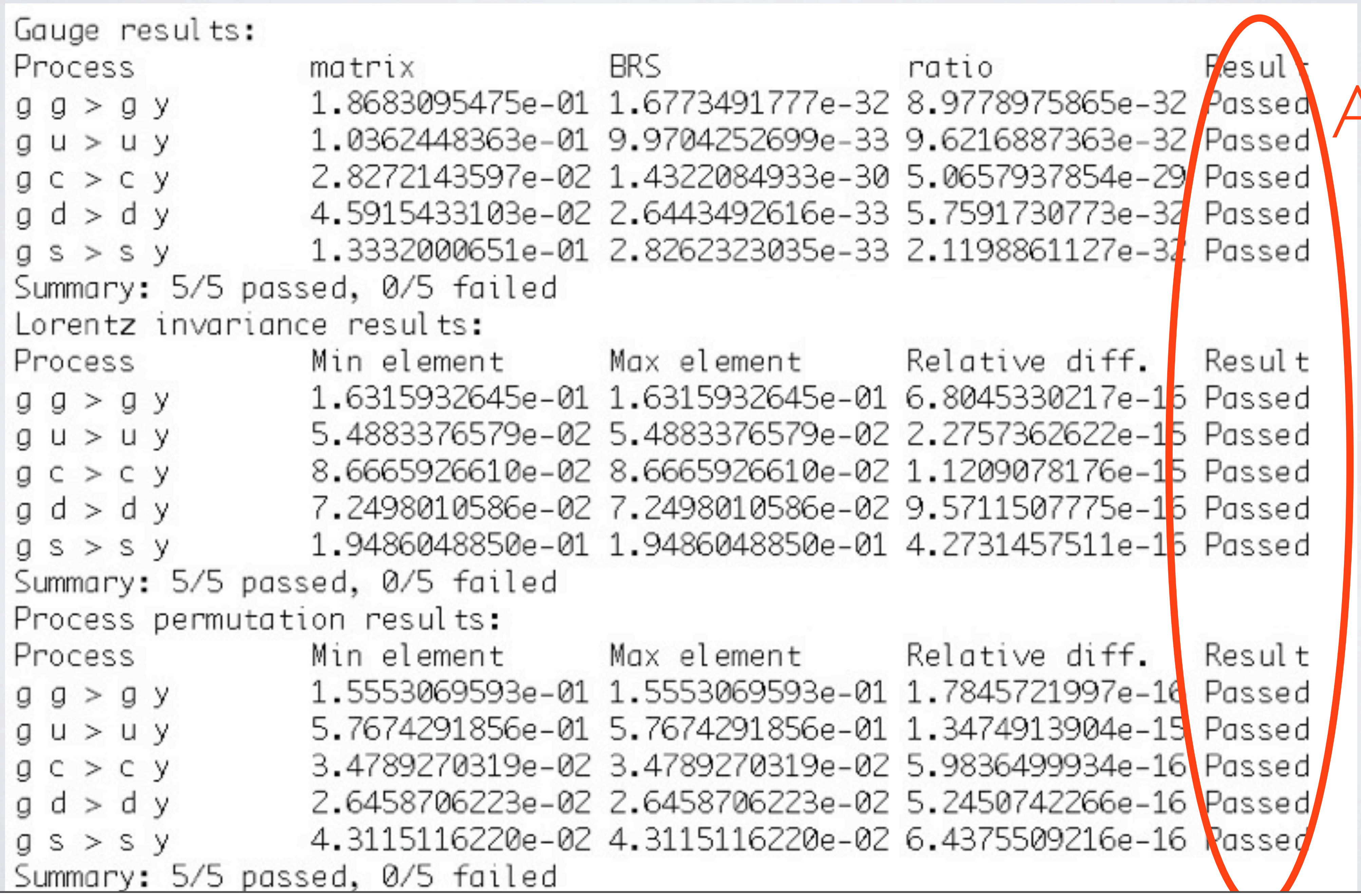
VALIDATION

```
mg5>import model RS
mg5>check full p p > j y
```

VALIDATION

mg5>import model RS

mg5>check full p p > j y

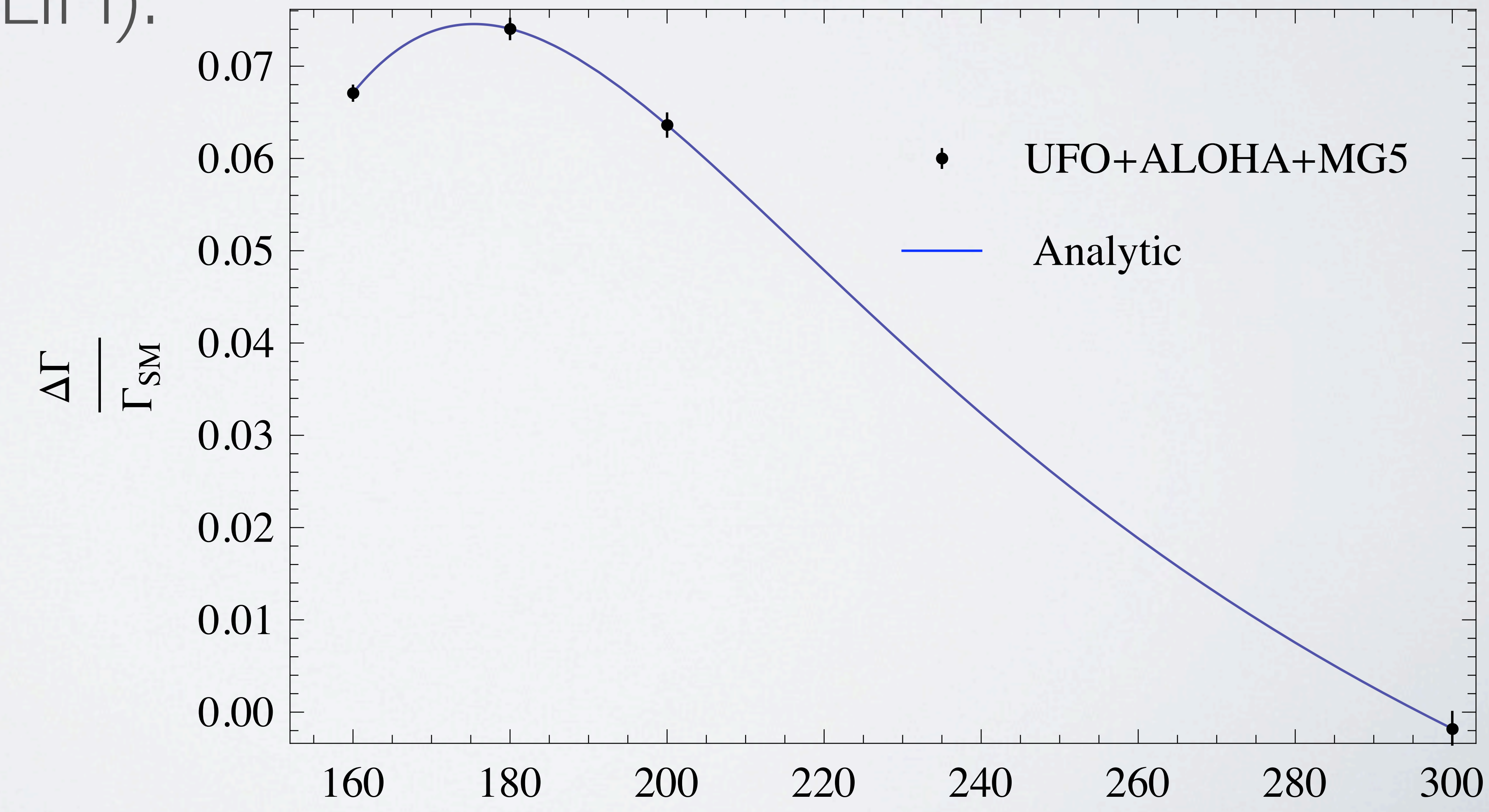


```
Gauge results:
Process          matrix           BRS              ratio            Result
g g > g y        1.8683095475e-01 1.6773491777e-32 8.9778975865e-32 Passed
g u > u y        1.0362448363e-01 9.9704252699e-33 9.6216887363e-32 Passed
g c > c y        2.8272143597e-02 1.4322084933e-30 5.0657937854e-29 Passed
g d > d y        4.5915433103e-02 2.6443492616e-33 5.7591730773e-32 Passed
g s > s y        1.3332000651e-01 2.8262323035e-33 2.1198861127e-32 Passed
Summary: 5/5 passed, 0/5 failed
Lorentz invariance results:
Process          Min element      Max element      Relative diff.   Result
g g > g y        1.6315932645e-01 1.6315932645e-01 6.8045330217e-16 Passed
g u > u y        5.4883376579e-02 5.4883376579e-02 2.2757362622e-15 Passed
g c > c y        8.6665926610e-02 8.6665926610e-02 1.1209078176e-15 Passed
g d > d y        7.2498010586e-02 7.2498010586e-02 9.5711507775e-16 Passed
g s > s y        1.9486048850e-01 1.9486048850e-01 4.2731457511e-16 Passed
Summary: 5/5 passed, 0/5 failed
Process permutation results:
Process          Min element      Max element      Relative diff.   Result
g g > g y        1.5553069593e-01 1.5553069593e-01 1.7845721997e-16 Passed
g u > u y        5.7674291856e-01 5.7674291856e-01 1.3474913904e-15 Passed
g c > c y        3.4789270319e-02 3.4789270319e-02 5.9836499934e-16 Passed
g d > d y        2.6458706223e-02 2.6458706223e-02 5.2450742266e-16 Passed
g s > s y        4.3115116220e-02 4.3115116220e-02 6.4375509216e-16 Passed
Summary: 5/5 passed, 0/5 failed
```

All good

VALIDATION

- MG4/MG5 validation:
  - SM / MSSM / HEFT / RS: more than 4000 processes
- MG5/analytical (SLIH):

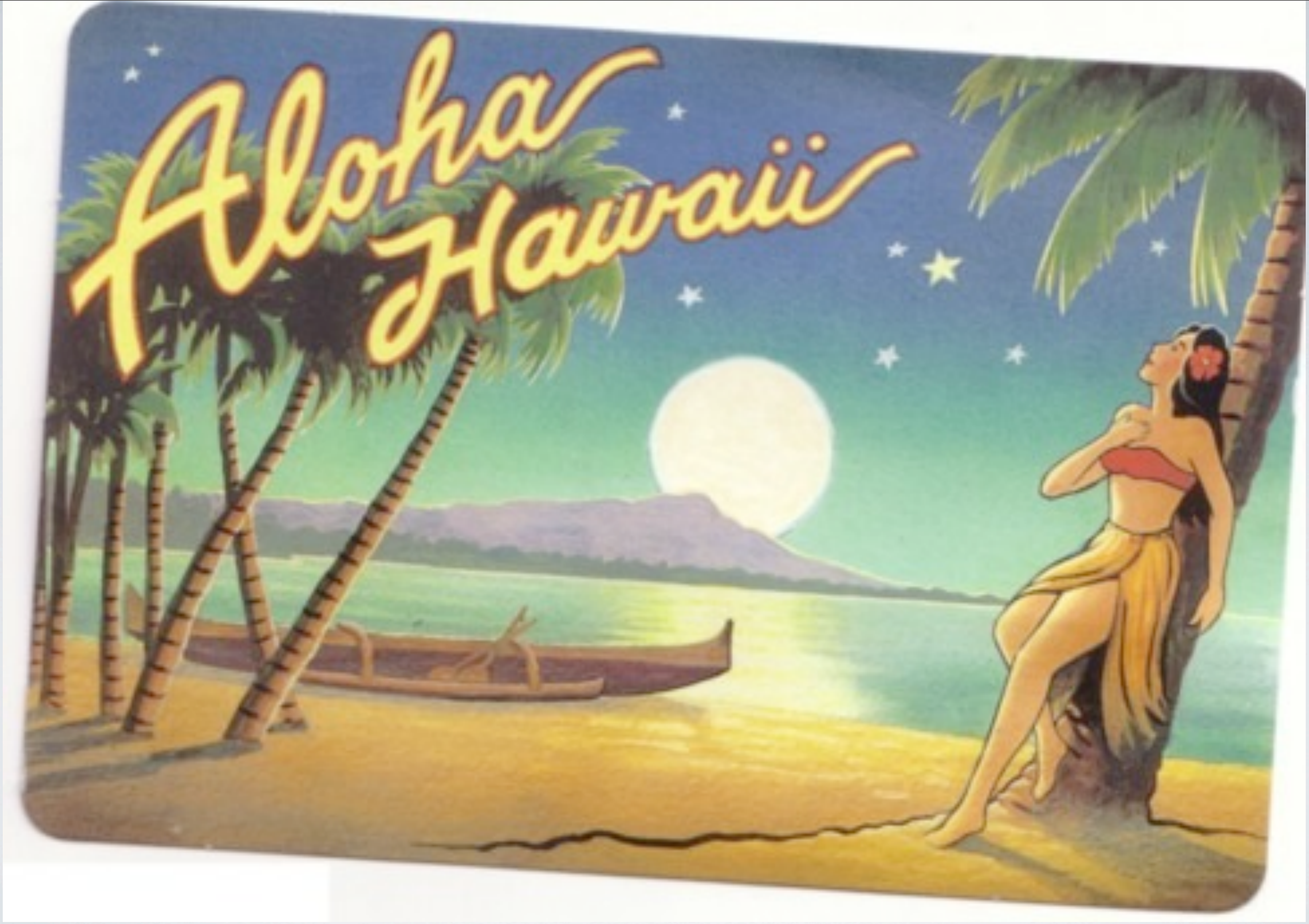


PERSPECTIVE

- Spin 3/2
- GPU
- More special routine

CONCLUSION

- UFO: Fully complete easy model
- UFO: Easy format to dealt with
- ALOHA: The Helas routine for BSM without the pain to write it.
- Fully interfaced with MG5.


Aloha
Hawaii