FDC Project

Bin Gong

Collaborated with J.X. Wang

Institute of High Energy Physics, CAS

January 28, 2021

Outline

Brief Introduction

- FDC = Feynman Diagram Calculation
- Purpose: automatic calculation of physical processes
- First developed by Prof. J.X. Wang since 1993
- First version of FDC has been presented at AIHENP93.
- Written in REDUCE and RLISP to generate Fortran Code
- Including some additional parts for certain physical research e.g. FDC-PWA (Partial Wave Analysis application for experimental study)
- Web: available at ???

REDUCE

- A general-purpose Computer Algebra System geared towards applications in physics.
- Written in Portable Standard LISP
- Something like Mathematica, FORM and Maple etc.
- Open-source and free now (since December 2008)
- User-level language: RLISP
- Two modes: algebraic and symbolic
- Web: http://www.reduce-algebra.com

a simple example of reduce

```
twain@Twains-MacBook:~$ reduce
Loading image file: /Users/twain/reduce-algebra/scripts/../pslbuild/x86_64-mac_unknown_version-darwin1
Reduce (Free PSL version), 12-Dec-2015 ...

1: vector p1,p2,p3,p4;

2: a:=g(l,p1,p2,p3,p4);

a := p1.p2*p3.p4 - p1.p3*p2.p4 + p1.p4*p2.p3

3: share a;

4: symbolic;

nil

5* reval(a);

(plus (times (cons p1 p2) (cons p3 p4)) (minus (times (cons p1 p3) (cons p2 p4))
) (times (cons p1 p4) (cons p2 p3)))

6* bye;

Quitting
twain@Twains-MacBook:~$
```

$\mathrm{Tr}(\hat{p}_1\hat{p}_2\hat{p}_3\hat{p}_4)$ is calculated here, shown in both algebraic and symbolic modes

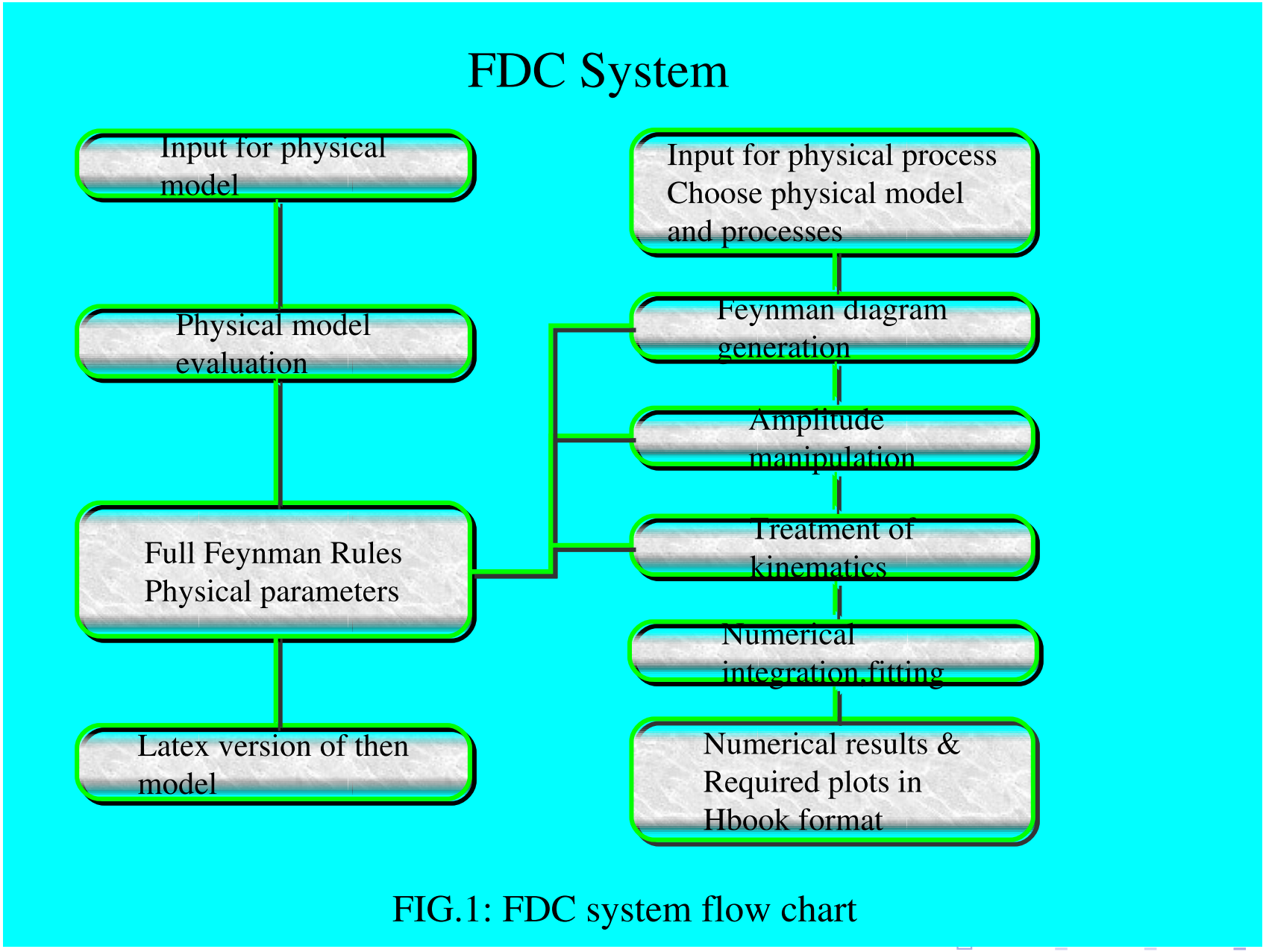


FIG.1: FDC system flow chart

Prerequisites and Environments

- A Unix-like system
- REDUCE
  - open source
  - available at http://reduce-algebra.com/downloading.htm
- A Fortran Compiler
- MPI environment if you want to use MPI
- Environment variables
  - fdc: where you store your FDC source
  - model: where you store your models
  - PATH: you have to tell your OS where to find "reduce" and other commands provided by FDC
- csh/tcsh: modify .cshrc in your home directory and add:
  - setenv fdc ~/fdc2.0
  - setenv model ~/model
  - set path=($path /usr/local/bin $fdc/bin ./)

Installation

- Install Reduce (usually psl version)
  - you will get a script called "redpsl" after installation
  - make a new script "reduce" with only two lines:
    redhome="path of your reduce/../pslbuild/..."
    exec $redhome/psl/bpsl -td 1000 -f $redhome/red/reduce.img
    and put it in the directory you choose before
    check /usr/local/bin/reduce in the virtual machine for this step
- Copy FDC source to the directory your chosen above, and run util/xbuild to build fdc source in the source directory.
- Compile Fortran Libraries of FDC and BASES (confirm Fortran compiler).
- Construct/obtain a model

Generate a new model in FDC

- No matter what tool you use in your study, a physical model is always needed, even at LO.
- A new model in FDC can be generate with following steps: (after the installation of FDC package)
  - create a new model directory from an old one with command: model_cp source_dir target_dir
  - modify the path to the directory in the file "model.tex" (% is symbol for comment in REDUCE)
  - modify the major input file "model_input" which describes your physical model
  - run "gmodel"
  - a command "glmodel" is available to generate TeX file for your model

a sample "model.tex" file

```
% Please revise the following items when a new model is created.

model_home:='"/auto/homegb/gongb4/model/ex0/";
model_name:='"SM in UG and $J/\psi$,${J/\psi}_8$,${B_c}$";
model_author:='"Jian-Xiong Wang";
model_time:='"Nov 1, 2009";

% The following three files must be prepared by USER and are placed
% in directory "model_home".

model_input:='model_input$    % description of the physical model.
symbole_list:='symbole_list$  % something needed for drawing diagram.
physical_parameters:='physical_parameters$    % physical parameters input.
%add_counter_term_input:='one_loop_qcd_counter_term;
;end;
~
~
```

a sample "model_input" file for SM

```
%%%%%%%%              The standard model in unitary gauge      %%%%%%%%%%
%------------------------ Gauge fields --------------------------------
gaugefields:=
%     No.  Name  SU(n)   notation  couping   breaking
   '((1    u     1       b         g1        yes    )
     (2    su    2       a         g         yes     )
     (3    su    3       gs        g3        no     )
    )$
%----------------------- Matter fields -------------------------------
matterinput:={{name,  2*spin,  chiral, "|g",  1,     2,    3}
              ,{ hig,   0  ,    rl  ,          1,     2,    0}
              ,{ el ,   1,      l   ,         -1,     2,    0}
              ,{ er ,   1,      r   ,         -2,     0,    0}
              ,{ q1l ,   1,      l  ,          1/3,    2,    3}
              ,{ q1dr,   1,      r  ,         -2/3,    0,    3}
              ,{ q1ur,   1,      r  ,          4/3,    0,    3}
              }$
%-------------------------------------------------------------------------
%         name  n_group   1_componet 2_componet   ...
 mdefl:={
         {hig, {2,        xx2   ,   (v0+h0-i*xx3)/2**0.5}}
         ,{el , {2,        nue   ,   ef}}
         ,{er , {2,        ef           }}
         ,{q1l , {2,       qu   ,  qd}}
         ,{q1dr, {2,       qd          }}
         ,{q1ur, {2,       qu            }}
        }$
vancumexpectation:='((hig v0));

fermion_generation:='(
                      (q1l q2l q3l) (q1dr q2dr q3dr) (q1ur q2ur q3ur)
                      (qu qc qt) (qd qs qb)
                      (el mul taul) (er mur taur)
                      (ef mu tau) (nue numu nut)
                     );

end$
```

a sample "model_input" file for IHDM

```
%%%%%%%%              The standard model in unitary gauge      %%%%%%%%%%
%------------------------ Gauge fields ---------------------------------
 gaugefields:=
 %     n-th  Name  SU(n)   notation  couping   breaking
      '((1    u     1       b         g1        yes    )
        (2    su    2       a         g         yes     )
        (3    su    3       gs        g3        no     )
       )$

%----------------------- Matter fields ---------------------------------
 matterinput:={{name,  2*spin,  chiral,   "|g",  1,     2,    3}
               ,{ hig1,    0  ,    rl  ,         1,     2,    0}
               ,{ hig2,   0  ,    rl  ,         1,     2,    0}
               ,{ el ,    1,      l  ,         -1,     2,    0}
               ,{ er ,    1,      r  ,         -2,     0,    0}
%               ,{ q1l ,   1,      l  ,          1/3,    2,    3}
%               ,{ q1dr,   1,      r  ,         -2/3,    0,    3}
%               ,{ q1ur,   1,      r  ,          4/3,    0,    3}
              }$
%-----------------------------------------------------------------------
 mdefl:={ % name   n-th group   1_componet   2_componet   ...
            {hig1,  {2,          xx2    ,     (v0+h0-i*xx3)/2**0.5     }}
           ,{hig2, {2,          xx21   ,     (h01-i*xx31)/2**0.5  }}
           ,{el ,  {2,          nue    ,     ef                       }}
           ,{er ,  {2,          ef                                    }}
           ,{q1l , {2,          qu     ,     qd                       }}
           ,{q1dr, {2,          qd                                    }}
           ,{q1ur, {2,          qu                                    }}
        }$

vancumexpectation:='((hig1 v0));

fermion_generation:='((q1l q2l q3l) (q1dr q2dr q3dr) (q1ur q2ur q3ur)
                      (qu qc qt) (qd qs qb)
                      (el mul taul) (er mur taur)
                      (ef mu tau) (nue numu nut)
                     );

give_up_compact:='ok;
```

The file "matterintinput"

- It is first generated by the command "gmodel"
- specify your model by change this file (choices of free parameters, masses, gauge parameters, etc)
- run "gmodel" again to apply the changes
- "matterintinput" will not be changed if it already exists when you run "gmodel"

part of "matterintinputt" file for SM

```
+hig(-1)*er(-1)*el(1)*c(0,2)
+hig(-1)*hig(1)*c(0,1)
+hig(-1)*hig(1)*hig(-1)*hig(1)*c(0,5)
+hig(-1)*q1dr(-1)*q1l(1)*c(0,3)
+higtid(-1)*q1ur(-1)*q1l(1)*c(0,4)
+q1l(-1)*q1dr(1)*hig(1)*c(0,3)
+q1l(-1)*q1ur(1)*higtid(1)*c(0,4)$

gauge_boson_redefine:={a(1,~v) => (w(1,v) + w(-1,v))/sqrt(2),
a(2,~v) => ((w(-1,v) - w(1,v))/sqrt(2))/i,
a(3,~v) => cos(theta)*z(0,v) + p(0,v)*sin(theta),
b(~v) => cos(theta)*p(0,v) - sin(theta)*z(0,v)}$

% where g1 can be any one of g1,g,sin(theta)$
construles:={g1 => g*sin(theta)/cos(theta),
zm => wm/cos(theta),
c(0,1) => ( - 8*c(0,5)*wm**3 + g**3*td1)/(2*g**2*wm),
v0 => (2*wm)/g,
c(0,5) => ( - g**2*h0m**2)/(8*wm**2)}$

% where mass of each particle can be changed, such as just put it to 0$

phymass:='((pg 0) (gsg 0) (gs 0) (w wm) (z zm) (p 0) (qb fmb) (qs fms) (qd fmd) (qt fmt) (qc fmc) (qu fmu) (tau fmtau) (mu fmmu) (ef fmef) (
nut 0) (numu 0) (nue 0) (h0 h0m) (xx3 xx3m) (xx2 xx2m))$

% where please take away  zero mass$

phyinput:='(wm fmb fms fmd fmt fmc fmu fmtau fmmu fmef h0m theta g g3)$

pchalist:='((taul (tau -1) (nut 0)) (mul (mu -1) (numu 0)) (taur (tau -1)) (mur (mu -1)) (q3l (qb (quotient (minus 1) 3)) (qt (quotient 2 3)
)) (q2l (qs (quotient (minus 1) 3)) (qc (quotient 2 3))) (q3dr (qb (quotient (minus 1) 3))) (q2dr (qs (quotient (minus 1) 3))) (q3ur (qt (
quotient 2 3))) (q2ur (qc (quotient 2 3))) (gs (gs 0)) (a (w 1) (z 0) (p 0)) (b (p 0) (z 0)) (q1ur (qu (quotient 2 3))) (q1dr (qd (quotient
(minus 1) 3))) (q1l (qd (quotient (minus 1) 3)) (qu (quotient 2 3))) (er (ef -1)) (el (ef -1) (nue 0)) (hig (xx3 0) (h0 0) (xx2 1)))
 $


gauge_fix_term_list:={f(p,ksi1,ksi1p,pd(p(0,v),v) + xx3(0)*ksi1p*zm,pg(-1)),
f(z,ksi2,ksi2p,pd(z(0,v),v) + xx3(0)*ksi2p*zm,zg(-1)),
f(w,ksi3,ksi3p,pd(w(-1,v),v) + xx2(-1)*i*ksi3p*wm,wgm(-1)),
f(w,ksi3,ksi3p,pd(w(1,v),v) - xx2(1)*i*ksi3p*wm,wgp(-1)),
f(gs(ic10),ksi4,ksi4p,pd(gs(ic10,v),v),gsg(-1,ic10))}$

r_ksi_value:='((ksi1 . 1) (ksi2 . 1) (ksi3 . 1) (ksi4 . 1));

r_ksip_value:='((ksi1p . 0) (ksi2p . ksi2) (ksi4p . 0) (ksi3p . ksi3));
```

part of "matterintinputt" file for IHDM

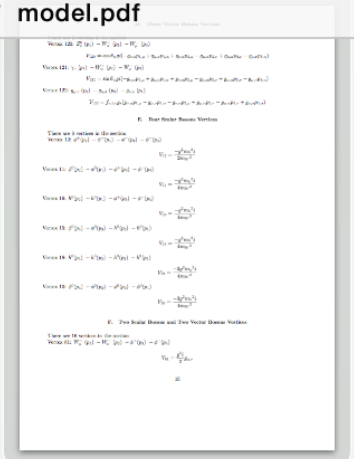
```
coupling_comment:='((g3 " strong interaction ") (g "electro-weak interaction ")     )$
model_input_comment:='"This is Standard Model in unitary gauge, include electro-weak interaction and QCD, Quark mixing terms are droped."$

matterinteraction:=el(-1)*er(1)*hig1(1)*c(0,5)
+hig1(-1)*er(-1)*el(1)*c(0,5)
+hig1(-1)*hig1(1)*c(0,1)
+hig1(-1)*hig1(1)*hig1(-1)*hig1(1)*c(0,10)
+hig2(-1)*hig2(1)*c(0,3)
+hig1(-1)*hig1(1)*hig2(-1)*hig2(1)*c(0,9)*g**2
+hig1(-1)*hig2(1)*hig1(-1)*hig2(1)*c(0,7)*g**2
+hig1(-1)*hig2(1)*hig2(-1)*hig1(1)*c(0,8)*g**2
+hig2(-1)*hig1(1)*hig2(-1)*hig1(1)*c(0,7)*g**2
+hig2(-1)*hig2(1)*hig2(-1)*hig2(1)*c(0,6)*g**2$
%+hig1(-1)*hig1(1)*hig2(-1)*hig2(1)*c(0,9)
%+hig1(-1)*hig2(1)*hig1(-1)*hig2(1)*c(0,7)
%+hig1(-1)*hig2(1)*hig2(-1)*hig1(1)*c(0,8)
%+hig2(-1)*hig1(1)*hig2(-1)*hig1(1)*c(0,7)
%+hig2(-1)*hig2(1)*hig2(-1)*hig2(1)*c(0,6)$


gauge_boson_redefine:={a(1,~v) => (w(1,v) + w(-1,v))/sqrt(2),
a(2,~v) => ((w(-1,v) - w(1,v))/sqrt(2))/i,
a(3,~v) => cos(theta)*z(0,v) + p(0,v)*sin(theta),
b(~v) => cos(theta)*p(0,v) - sin(theta)*z(0,v)}$

% where g1 can be any one of g1,g,sin(theta)$
construles:={g1 => g*sin(theta)/cos(theta),
zm => wm/cos(theta),
c(0,1) => ( - 8*c(0,10)*wm**3 + g**3*td1)/(2*g**2*wm),
v0 => (2*wm)/g,
%c(0,3) =>  - 2*c(0,9)*wm**2 - xx21m**2,
c(0,9) =>  -(c(0,3)+xx21m**2)/2/wm**2,
c(0,8) => ( - h01m**2 + 2*xx21m**2 - xx31m**2)/(4*wm**2),
c(0,7) => ( - h01m**2 + xx31m**2)/(8*wm**2),
c(0,3) =>mu2p,
c(0,6) =>lambda2,
c(0,10) => ( - g**2*h0m**2)/(8*wm**2)
}$

% where mass of each particle can be changed, such as just put it to 0$

phymass:='((pg 0) (gsg 0) (gs 0) (w wm) (z zm) (p 0) (tau fmtau) (mu fmmu) (ef fme) (nut 0) (numu 0) (nue 0) (h01 h01m) (xx31 xx31m) (xx21
xx21m) (h0 h0m) (xx3 xx3m) (xx2 xx2m))$

% where please take away  zero mass$
```

part of "model.pdf" for SM

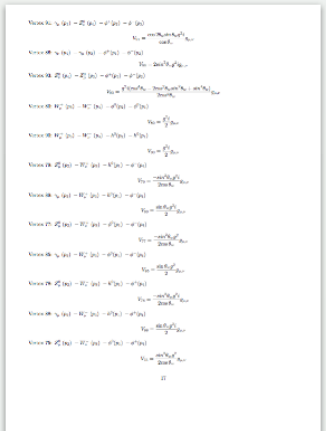
model.pdf（第 17 页，共 23 页）

搜索

model.pdf



17



D. Three Vector Bosons Vertices

There are 3 vertices in the section
Vertex 120: $Z_\nu^0\ (p_1)\ -W_\alpha^+\ (p_2)\ -W_\mu^-\ (p_3)$

$$V_{120} = \cos\theta_w g i(-g_{\nu,\alpha}p_{1,\mu} + g_{\mu,\nu}p_{1,\alpha} + g_{\nu,\alpha}p_{2,\mu} - g_{\mu,\alpha}p_{2,\nu} + g_{\mu,\alpha}p_{3,\nu} - g_{\mu,\nu}p_{3,\alpha})$$

Vertex 121: $\gamma_\nu\ (p_1)\ -W_\alpha^+\ (p_2)\ -W_\mu^-\ (p_3)$

$$V_{121} = \sin\theta_w g i(-g_{\nu,\alpha}p_{1,\mu} + g_{\mu,\nu}p_{1,\alpha} + g_{\nu,\alpha}p_{2,\mu} - g_{\mu,\alpha}p_{2,\nu} + g_{\mu,\alpha}p_{3,\nu} - g_{\mu,\nu}p_{3,\alpha})$$

Vertex 122: $g_{\mu,c}\ (p_3)\ -g_{\alpha,b}\ (p_2)\ -g_{\nu,a}\ (p_1)$

$$V_{122} = f_{a,b,c}g_s(g_{\nu,\alpha}p_{1,\mu} - g_{\mu,\nu}p_{1,\alpha} - g_{\nu,\alpha}p_{2,\mu} + g_{\mu,\alpha}p_{2,\nu} - g_{\mu,\alpha}p_{3,\nu} + g_{\mu,\nu}p_{3,\alpha})$$

E. Four Scalar Bosons Vertices

There are 6 vertices in the section
Vertex 12: $\phi^+(p_2)\ -\phi^+(p_1)\ -\phi^-(p_4)\ -\phi^-(p_3)$

$$V_{12} = \frac{-g^2{m_h}^2 i}{2{m_W}^2}$$

Vertex 11: $\phi^0(p_2)\ -\phi^0(p_1)\ -\phi^+(p_3)\ -\phi^-(p_4)$

$$V_{11} = \frac{-g^2{m_h}^2 i}{4{m_W}^2}$$

Vertex 16: $h^0(p_2)\ -h^0(p_1)\ -\phi^+(p_3)\ -\phi^-(p_4)$

$$V_{16} = \frac{-g^2{m_h}^2 i}{4{m_W}^2}$$

Vertex 15: $\phi^0(p_4)\ -\phi^0(p_3)\ -h^0(p_2)\ -h^0(p_1)$

$$V_{15} = \frac{-g^2{m_h}^2 i}{4{m_W}^2}$$

Vertex 18: $h^0(p_4)\ -h^0(p_3)\ -h^0(p_2)\ -h^0(p_1)$

$$V_{18} = \frac{-3g^2{m_h}^2 i}{}$$

Current Models

- The Standard Model (SM) has already been constructed.
- The Minimal Supersymmetric Standard Model (MSSM) also has been constructed.
- Some other new NP models (2HDM, IHDM)
- Also compatible with phenomenological models (add_vertices).
- Most intermediate states and effective vertices from NRQCD has been implemented in the SM.
- QCD counter terms have been manually inserted in the SM.

Automatic Renormalization (still in process)

- In high order calculation, renormalization of the model is needed to generate counter terms which cancel the UV divergences from virtual corrections.
- The renormalization of QCD is simple ($\overline{\rm MS}$ for coupling, $\overline{\rm MS}$/OS for fields).
- But for electroweak, you have so many choices. Things are totally different:
  - mixing between particles (more renormalization constants)
  - renormalization constants before/after SSB
  - dependence of counter terms on gauge-fixing and gauge parameters
  - different renormalization conditions ($\overline{\rm MS}$/OS, zero/nonzero masses)
  - Some NP model needs special treatment (FJ tadpole scheme)

Preset renormalization schemes in FDC

```
%-------------------------------------------------------------------------------------------------------
%For scheme_switch,you can choose number 1,2,3... and what are these number represent list as follows.

% n   model         gauge-fixing      gauge-symmetrization      gauge             renormalization
%-------------------------------------------------------------------------------------------------------

% 1   Electroweak   kyoto-scheme      gauge-asymmetric-scheme   Feynman-gauge     on-shell

% 2   Electroweak   European-scheme   gauge-asymmetric-scheme   Feynman-gauge     on-shell

% 3   Electroweak   kyoto-scheme      gauge-symmetric-scheme    Feynman-gauge     on-shell

% 4   Electroweak   kyoto-scheme      gauge-asymmetric-scheme   unitary-gauge     on-shell

% 5   Electroweak   kyoto-scheme      gauge-asymmetric-scheme   R_ksi-gauge       on-shell

% 6   Electroweak   kyoto-scheme      gauge-asymmetric-scheme   nonlinear-gauge   on-shell

% 7   Electroweak   kyoto-scheme      gauge-asymmetric-scheme   Landau-gauge      on-shell

% 8   Electroweak   kyoto-scheme      gauge-asymmetric-scheme   unitary-gauge     MS(coupling constant) on-shell(others)

% 9   Electroweak   kyoto-scheme      gauge-asymmetric-scheme   Feynman-gauge     MS(coupling constant) on-shell(others)

% 10   QCD          kyoto-scheme      gauge-asymmetric-scheme   Feynman-gauge     MS

% 11   QCD          kyoto-scheme      gauge-asymmetric-scheme   Feynman-gauge     on-shell(all fields) MS(coupling constants)

%-------------------------------------------------------------------------------------------------------
```

Calculation (for both LO and NLO)

- create a directory for the process use process_cp
- modify "process.def" and "option" files in the directory to specific
  - physical model of your process
  - incoming and outgoing particles
  - order of result
  - way to obtain squared amplitude
  - some others...
- use doall to perform the following:
  - gen_diag: generate diagrams of the process (psdraw)
  - amp: manipulate square of amplitude and output in Fortran
  - kine: generate code for phase space integration
  - make: compile the Fortran code
- run the Fortran code with int (int2, int3, int4)
- LO only:
  - parton level event generation (SPRING inside BASES)
  - parton shower and hadronization (PYTHIA)

a sample "process.def"

```
% Please revise the following items when a new process is created.
 algebraic;
 model_home:='"/auto/homegb/gongb/model/smu2nloop_up_to_c/";
 process_name:='"$g g-->J/psi$ Production in $e^+ e^-$ collider";
 process_author:='"Jian-Xiong Wang";
 process_time:='"March 1, 2006";

 namel:='(ef efb jpsi etac)$
 inpl:='(1 1 -1 -1)$
 ncorrection:='( (g 0)  (g3 2) )$
 ncolor:='1$
 %check_gauge_invariance:='ok;
 approximation_rules:={hjpm=>2*fmc,hjpm2=>4*fmc2,hetacm=>2*fmc,hetacm2=>4*fmc2};
 %special_case_option:='((1 p ec ef));
 mass_drop_list:='((fme 0));
 no_abs_list:='ok;
n_charge_c:='0$
%do_not_sum_ferry_partner:='ok;
input_list:='(
      (ec  10.6)
      (alpha  (quotient 1 137.0))
      (acc1  0.01)
      (acc2  0.01)
      (itmx1  10)
      (itmx2  10)
      (ncall  2000)
      (mxdim  50)
     )$
end$
histograms:={

      {1,pt(p3),50,1,5,"Pt of Jpsi"},

      {2,wcos(p3,z),50,"cos(Jpsi and beam) "}
```

a sample "option"

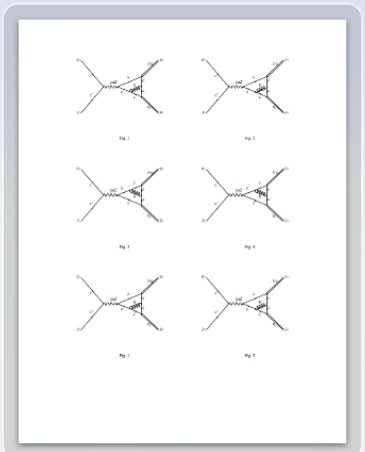
```
%*************************************************************************
% This file contains all the options which could be used in FDC system
% and all the option are set to default values. User can change them
% by set their value to other possible value explained in the comment
% line.
%*************************************************************************
%%%%%%%%%%% The following is options for kinematics %%%%%%%%%%%%%%%%
%   check_kin:='ok$
%   physical_cut='ok$
%The list of diagram to be considered in kinematics generation
%  keep_list:='(1 2 3 4 5 6);
%%%%%%%%% The following is options for diagram generation %%%%%%%%
%The list of diagram to be considered in amplitudes calculation
    ec_sqrts:='ok;
       pan_list_drop:='((pa1 qb qt));
%  diagram_keep_list:='(1);
%  diagram_keep_list1:='(1 80 81 82 83 84);
%  diagram_keep_list1:='(1 37);
%  diagram_keep_list1:='(48);
%  ct_choose:='1;
%   ncolor:='1$
%   selfenergy:='ok$
%  To add width in calculation
%    addwidth:='ok$
   amp_method:='tree$
   amp_method:='2$
%   diag_drop:='ok$
```

part of Feynman diagrams for $e^+e^- \to J/\psi + \eta_c$

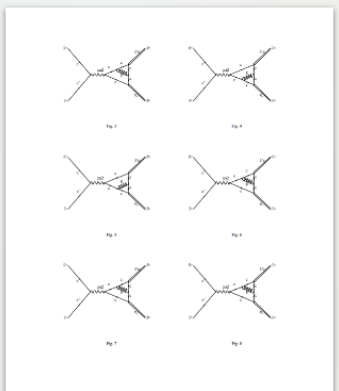
diag.pdf（第 1 页，共 15 页）

搜索

diag.pdf



1



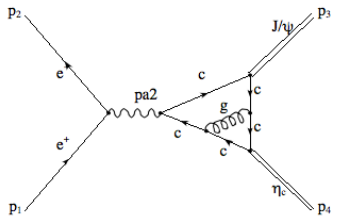


Fig. 1

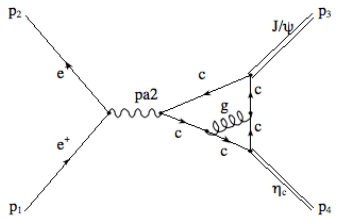


Fig. 2

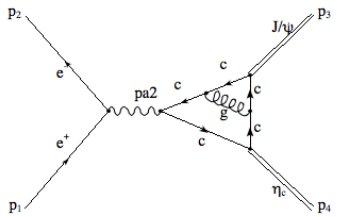


Fig. 3

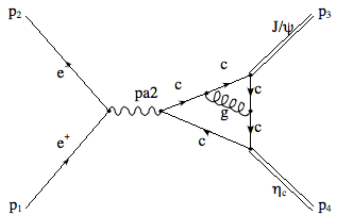


Fig. 4

One-loop Part of FDC (EW still in process)

$$
\begin{aligned}
d\sigma^{(1)} &= d\sigma_V(\mathrm{Loop}+\mathrm{CT}) + d\sigma_R \\
d\sigma_R &= d\sigma_S(\delta_s) + d\sigma_{HC}(\delta_s,\delta_c) + d\sigma_{H\overline{C}}(\delta_s,\delta_c)
\end{aligned}
$$

- The one-loop part of FDC is completed in 2007, and upgraded in 2011 to calculate processes involving P-wave particles
- The results are obtained analytically.
  - at the level of amplitude square, before the integration of phase space.
  - usually they are still in numerical form (Fortran codes), as in most cases, they are too complicated to read.
- A two-cutoff phase space slicing method (PSS) is realized in FDC to deal with IR divergences in real correction processes
- The divergences are factorized in soft/collinear limit, and added to corresponding virtual correction processes.

- Counter term diagrams are generated automatically (after the input of renormalization constant)
- Loop integrals are calculated analytically under dimensional regularization.
- All the divergence (both UV and IR) are separated during the calculation of amplitude analytically, and summed up to check if they are really cancelled with others.
- In 2007 version, Passarino-Veltman reduction method is used for tensors reduction
- In 2011, new reduction method (a kind of IBP) for loop integrals is realized.
- The cutoff independence has to be checked after summing up both real and virtual corrections.

Fortran Codes

- all stored in the directory fort
- makefile
- int.f: main program, need parameters in input.dat
- parameter(1).f: physical parameters
- func.f: phase space
- amps2.f: squared amplitude
  - method 1:
    - amp???.f /ampl???.f: LO/NLO amplitude of corresponding diagram
    - ams??.f: square of LO amplitude
    - amsl??.f square of NLO amplitude with LO amplitude
  - method 2
    - amps20.f: square of LO amplitude
    - amp???.f: square of corresponding NLO diagram with LO amplitude
- output: convergence.dat fresult.dat

a sample "makefile"

```
# FILE makefile

# FDC1.0 Version   1.00

SHELL      = /bin/csh
f77        = ifort
fopt       =
psintlib   = basesd
fdclib   = fdcd

intldir    = /auto/homegb/gongb4/fdc2.0/basesv5.1/lib

cernldir    = /cern/2001/lib

fdcldir    = /auto/homegb/gongb4/fdc2.0/f77

psintlib1   = ooptools

LFLAGS=-lpdflib804 -lpacklib -lmathlib -lkernlib
.suffixes: .f .o
.f.o:
      $(f77) $(fopt) -c $*.f
#
.suffixes: .F .o
.F.o:
      $(f77) $(fopt) -c -g -I$(LT)/include $*.F
#
objs       = int.o amps2.o func.o \
             ampl83.o ampl81.o ampl80.o ampl79.o \
```

part of “amps2.f”

```
      call ampl58()
      call ampl59()
      call ampl60()
      call ampl75()
      call ampl76()
      call ampl77qd()
      call ampl77qs()
      call ampl77qu()
      call ampl77qc()
      call ampl78()
      call ampl79()
      call ampl80()
      call ampl81()
      call ampl83()
      call genppp()
C     The following is tree-diagram contribution
      a0=amps20()

      a1=cl(1)
C      amps2=wc0**2*a0+wc0*wc1*(2*a1+b1)
      amps2=wc0**2*a0
      return
      end
```

```
      FUNCTION amps20()
      IMPLICIT real*8(A-H,O-Z)
      IMPLICIT  integer (I-N)
      include 'inclppp.f'
      include 'inclcon.f'
      include 'inclcon1.f'
      real*8 u,t,s
      COMMON / stu / u,t,s
      ans=(8*s**2+16*s*t-8*s+16*t**2-8*t+1)/(8*fmc**6*s**5)
      amps20=ans
      return
      end
~
```

Some others

- combination of Fortran libraries using pre-compilation
- geometric strategy for Sector Decomposition
- numerical method for Fermion Lines
- automatic generation of R2 terms for OPP method
- reduction of coefficient (rational fraction)

Thanks for your attention!