FDC Project

Bin Gong

Collaborated with J.X. Wang and Y.D. Wang

Institute of High Energy Physics, CAS

MC4BSM 2016, UCAS, Beijing, July 21, 2016

Brief Introduction

- FDC = Feynman Diagram Calculation
- Purpose of project: automatic calculation of physical processes

Brief Introduction

- FDC = Feynman Diagram Calculation
- Purpose of project: automatic calculation of physical processes
- First developed by Prof. J.X. Wang since 1993
- The Package is written in REDUCE (RLISP) to generate Fortran Code

Brief Introduction

- FDC = Feynman Diagram Calculation
- Purpose of project: automatic calculation of physical processes
- First developed by Prof. J.X. Wang since 1993
- The Package is written in REDUCE (RLISP) to generate Fortran Code
- Including some additional parts for certain physical research e.g. FDC-PWA (Partial Wave Analysis application for experimental study)
- Homepage: http://www.ihep.ac.cn/wjx/

REDUCE

- A general-purpose Computer Algebra System geared towards applications in physics.

REDUCE

- A general-purpose Computer Algebra System geared towards applications in physics.
- Something like Mathematica, FORM and Maple etc.

REDUCE

- A general-purpose Computer Algebra System geared towards applications in physics.
- Something like Mathematica, FORM and Maple etc.
- Written in Portable Standard LISP
- Open-sourced and freed now (since December 2008)
- User-level language: RLISP
- Two modes: algebraic and symbolic

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

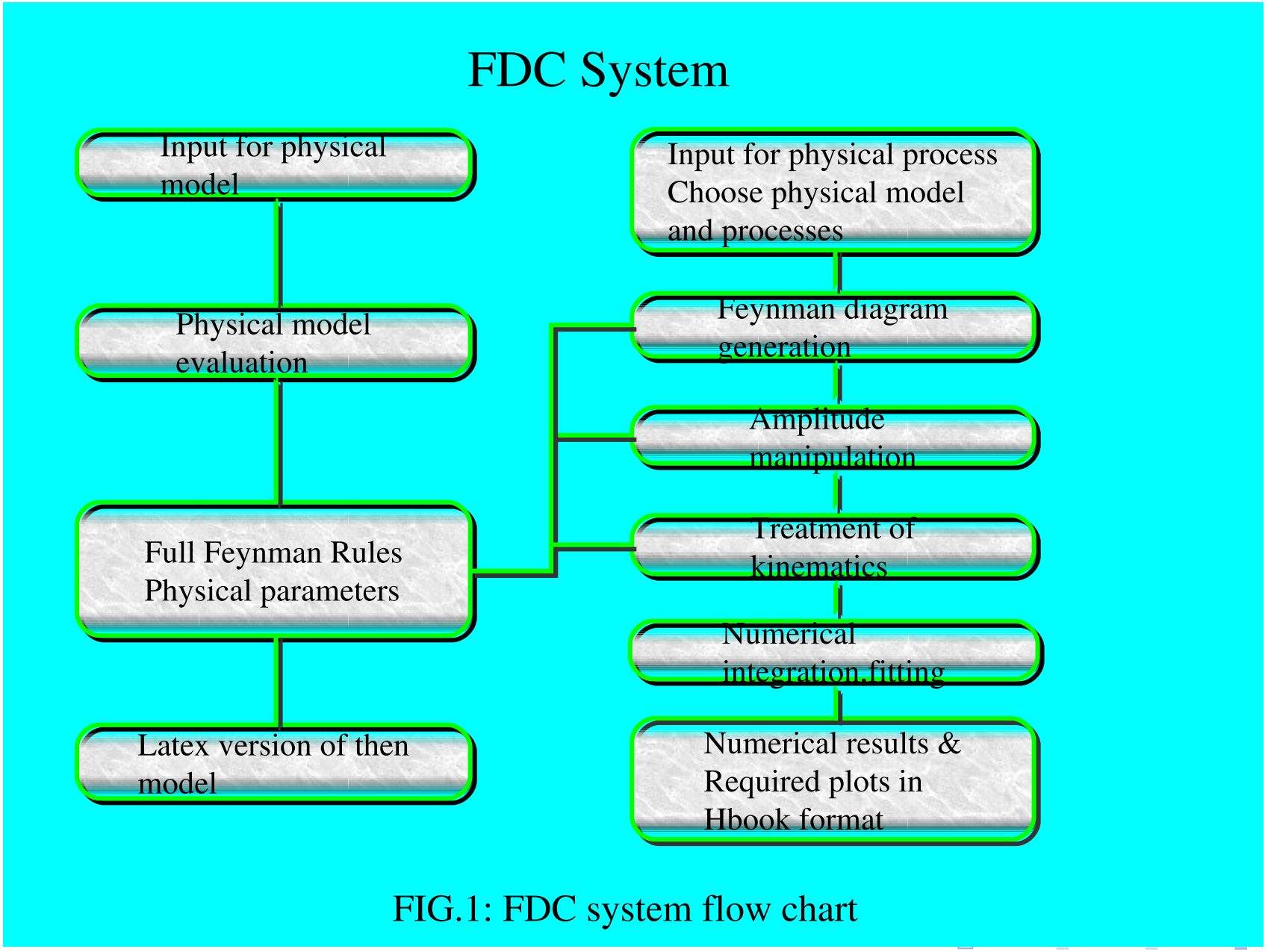

FIG.1: FDC system flow chart

Model Construction

- FDC do following things automatically in model construction:
  - generate matter interaction and soft-breaking term according to gauge invariance and global symmetry
  - construct Lagrangian and deduce Feynman rules
  - specific the model for FDC to accelerate further calculation
  - generate LaTeX description file of the model

Model Construction

- FDC do following things automatically in model construction:
  - generate matter interaction and soft-breaking term according to gauge invariance and global symmetry
  - construct Lagrangian and deduce Feynman rules
  - specific the model for FDC to accelerate further calculation
  - generate LaTeX description file of the model
- The SM and MSSM have already been constructed in FDC.
- Also compatible with phenomenological models.
- Most intermediate states and effective vertices from NRQCD has been implemented in the SM.

Model Construction

- FDC do following things automatically in model construction:
  - generate matter interaction and soft-breaking term according to gauge invariance and global symmetry
  - construct Lagrangian and deduce Feynman rules
  - specific the model for FDC to accelerate further calculation
  - generate LaTeX description file of the model
- The SM and MSSM have already been constructed in FDC.
- Also compatible with phenomenological models.
- Most intermediate states and effective vertices from NRQCD has been implemented in the SM.
- Automatic renormalization of models is under development.
- QCD counter terms have been manually inserted in the SM temperorily at present.

Part of symbols in FDC

name	*name in FDC*	*mass*	*mass in FDC*	*width in FDC*	*charge*	*spin*
ν_e	*nue*	0	0	0	0	1/2
ν_μ	*numu*	0	0	0	0	1/2
ν_τ	*nut*	0	0	0	0	1/2
e^-	*ef*	m_e	*fme*	*whe*	-1	1/2
μ^-	*mu*	m_μ	*fmmu*	*whmu*	-1	1/2
τ^-	*tau*	m_τ	*fmtau*	*whtau*	-1	1/2
u	*qu*	m_u	*fmu*	*whu*	2/3	1/2
c	*qc*	m_c	*fmc*	*whc*	2/3	1/2
t	*qt*	m_t	*fmt*	*wht*	2/3	1/2
d	*qd*	m_d	*fmd*	*whd*	$-1/3$	1/2
s	*qs*	m_s	*fms*	*whs*	$-1/3$	1/2
b	*qb*	m_b	*fmb*	*whb*	$-1/3$	1/2
γ	*p*	0	0	0	0	1
Z^0	*z*	m_{Z^0}	*zm*	*wh*	0	1
W^+	*w*	m_w	*wm*	*wh*	1	1
g	*gs*	0	0	0	0	1
h^0	*h0*	m_h	*hm*	*wh*	0	0
g_g	*gsg*	0	0	0	0	0

Process Calculation at Tree Level

- Choose your model and process you want to study

```
model_home:='"$model/smu2nloop_up_to_c/";
namel:='(ef efb jpsi etac)$
inpl:='(1 1 -1 -1)$
ncorrection:='( (g 0)  (g3 0) )$
```

Process Calculation at Tree Level

- Choose your model and process you want to study

```
model_home:='"$model/smu2nloop_up_to_c/";
namel:='(ef efb jpsi etac)$
inpl:='(1 1 -1 -1)$
ncorrection:='( (g 0)  (g3 0) )$
```

- gen_diag: generate Feynman diagrams
- amp:
  - manipulate the amplitude diagram by diagram
  - square the amplitude analytically or numerically according to your choice
  - generate Fortran codes for the (squared) amplitude
- kine: generate Fortran codes for phase space integration

Process Calculation at Tree Level

- Choose your model and process you want to study

```
model_home:='"$model/smu2nloop_up_to_c/";
namel:='(ef efb jpsi etac)$
inpl:='(1 1 -1 -1)$
ncorrection:='( (g 0)  (g3 0) )$
```

- gen_diag: generate Feynman diagrams
- amp:
  - manipulate the amplitude diagram by diagram
  - square the amplitude analytically or numerically according to your choice
  - generate Fortran codes for the (squared) amplitude
- kine: generate Fortran codes for phase space integration
- Run the Fortran codes for (differential) cross section (BASES)
- parton level event generation (SPRING inside BASES)
- parton shower and hadronization (PYTHIA)

One-loop Part of FDC

- It is more and more important to include higher order corrections in theoretical predictions as the development of high energy physics.
- The development of one-loop calculation part of FDC started from 2002 and finished in 2007
- It is upgraded in 2011 to deal with processes involving P-wave state particles (e.g. χ_{cJ}).
- The results are obtained analytically.
  - At the level of amplitude square, before the integration of phase space ($\mathrm{Re}(M_0 * M_1)$,$|M_1|^2$).
  - Usually they are still in numerical form (Fortran Code), as in most cases, they are too complicated to read.

Real Corrections

A two-cutoff phase space slicing method (PSS) [Harris and Owens (2002)] is realized in FDC to deal with IR divergence.

- Two cutoffs, δ_s and δ_c are introduced to separate the phase space into three parts: soft, hard-collinear, hard-noncollinear $\mathrm{d}\sigma_R = \mathrm{d}\sigma_S + \mathrm{d}\sigma_{HC} + \mathrm{d}\sigma_{H\overline{C}}$
- The hard-noncollinear part is finite, and can be calculated numerically with traditional Monte-Carlo method.
- Both the soft and hard-collinear part are factorized in the soft/collinear limit, and added to corresponding virtual correction processes.
- All the divergence (including those in virtual corrections) are separated analytically, and then sum up to check if they are really cancelled with others.

Virtual Corrections

- Counter term diagrams are generated automatically (after the input of renormalization constant)
- In 2007 version, Passarino-Veltman reduction method is used for tensors reduction. After 2011 upgrade, new reduction method (a kind of IBP) for loop integrals is realized.
- Two ways to generate square of amplitude (same as tree level):
  - square the amplitude analytically
  - generate numerical result (Fortran Code) of amplitude, then square it.
  - these two ways will lead to different tensor reduction and then bring a cross-check inside FDC
- All the divergence (both UV and IR) are separated during the calculation of amplitude analytically
- The cutoff (δ_s and δ_c) independence has to be checked after summing up both real and virtual corrections.

Process Calculation at NLO

One-loop calculation is done with following procedure:

- Generate directory and input files for all possible subprocesses (process_cp, not automatic)

Process Calculation at NLO

One-loop calculation is done with following procedure:

- Generate directory and input files for all possible subprocesses (process_cp, not automatic)
- For all real correction subprocesses, do the same as LO calculation but with cutoffs in the phase space.
- For all virtual correction subprocesses
  - almost same as LO calculation, different ncorrection
  - check cancellation of all the divergences in each virtual subprocess (parts from real subprocesses have been added to corresponding virtual subprocesses)

Process Calculation at NLO

One-loop calculation is done with following procedure:

- Generate directory and input files for all possible subprocesses (process_cp, not automatic)
- For all real correction subprocesses, do the same as LO calculation but with cutoffs in the phase space.
- For all virtual correction subprocesses
  - almost same as LO calculation, different ncorrection
  - check cancellation of all the divergences in each virtual subprocess (parts from real subprocesses have been added to corresponding virtual subprocesses)
- Do phase space integration in all the subprocesses, and sum up the results; check cutoff independence (some scripts available, not fully automatic)

Work Done with FDC at one-loop level (mostly in quarkonium physics)

- Quarkonium production in e^+e^- annihilation
  - $J/\psi + \eta_c$, $J/\psi + J/\psi$, $J/\psi + X$
- Quarkonium production and polarization puzzle at hadron colliders.
  - $J/\psi(\Upsilon)$: ${}^3S_1^{[1]}$, ${}^3S_1^{[8]}$, ${}^1S_0^{[8]}$, ${}^3P_J^{[8]}$
  - $\chi_{c(b)}$: ${}^3P_J^{[1]}$, ${}^3S_1^{[8]}$
- Decays
  - $\eta_b \to J/\psi + J/\psi$
- Others

n(intermediate state)	${}^3S_1^{[1]}$	${}^3S_1^{[8]}$	${}^1S_0^{[8]}$	${}^3P_J^{[8]}$
$gg \to \langle c\bar{c}\rangle_n + g$	6/129	16/413	12/267	12/267
$gq \to \langle c\bar{c}\rangle_n + q$	-	5/111	2/49	2/49
$q\bar{q} \to \langle c\bar{c}\rangle_n + g$	-	5/111	2/49	2/49
$gg \to \langle c\bar{c}\rangle_n + gg$	60	123	98	98
$gg \to \langle c\bar{c}\rangle_n + q\bar{q}$	6	36	20	20
$gq \to \langle c\bar{c}\rangle_n + gq$	6	36	20	20
$q\bar{q} \to \langle c\bar{c}\rangle_n + gg$	6*	36	20	20
$q\bar{q} \to \langle c\bar{c}\rangle_n + q\bar{q}$	-	14	4	4
$q\bar{q} \to \langle c\bar{c}\rangle_n + q'\bar{q}'$	-	7	2	2
$qq \to \langle c\bar{c}\rangle_n + qq$	-	14	4	4
$qq' \to \langle c\bar{c}\rangle_n + qq'$	-	7	2	2

Nos. of diagrams for subprocesses in direct $J/\psi(\Upsilon)$ hadroproduction at NLO

Some notes for FDC installation

Prerequisites

- REDUCE (make a symbolic link with the name "reduce")
- A Unix-like system
- A Fortran Compiler
- MPI environment if you want to use MPI

Some notes for FDC installation

Prerequisites

- REDUCE (make a symbolic link with the name "reduce")
- A Unix-like system
- A Fortran Compiler
- MPI environment if you want to use MPI

Required Environment Variables

- fdc: path for your FDC source
- model: path for your models
- PATH: add $fdc/bin and the path for "reduce" into it

Conclusion and outlook

- In general, FDC is a package to generate Fortran codes (amplitude, phase space), and use BASES for phase space integration
- It is a ME generator at NLO

Conclusion and outlook

- In general, FDC is a package to generate Fortran codes (amplitude, phase space), and use BASES for phase space integration
- It is a ME generator at NLO
- The following is to be added/improved in future:
  - Model Construction
    - More models (even at tree level)
    - Automatic renormalization (one-loop QCD is not enough)
    - Conversion between UFO models
  - Parton level event generation at NLO (experimentalists need events, not distributions)
  - Even higher order calculation for more precise study
  - Not so easy to use (too many options and commands, user-friendly interface)

Thanks for your attention!

B. W. Harris and J. F. Owens, Phys. Rev. **D65**, 094032 (2002).

G. Duplancic and B. Nizic, Eur.Phys.J. **C35**, 105 (2004), `hep-ph/0303184`.