

## V. Scattering amplitudes

### 1) Ampiezze di scattering in QFT

1.1) Input: impulsi e phase space

1.2) Esempi:  $e^+e^- \rightarrow qq$  e  $e^+e^- \rightarrow qqg$

1.3) Algoritmi Numeriche per le Ampiezze

### 2) Python Interface to N<sub>J</sub>ET

I)

## Scattering Amplitudes

$$\mathcal{A} = \sum_i (\text{Feynman diagram})_i \quad \langle |\mathcal{A}|^2 \rangle = \sum_{\text{spins}} \mathcal{A}^\dagger \mathcal{A}$$

$$\mathcal{A} : (p_i^\mu; p_i^2 = m_i^2, \sum p_i^\mu = 0) \rightarrow \mathbb{C}$$

amplitudes are a function of momenta (also other quantum numbers of the particles: helicity, charges etc)

I.I)

## Momenta and phase space

(see notes)

$$d\Phi_n = \delta^{(4)}(Q - \sum_{i=1}^{n_{\text{final}}} p_i) \prod_{i=1}^{n_{\text{final}}} d^4 p_i \delta^{(+)}(p_i^2)$$

(modulo factors of  $2\pi$ , massless only)

need to solve:

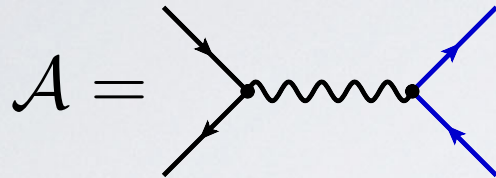
- on-shell ( $p_i^2=0$ )
- momentum conservation

uniform phase space sampling  
RAMBO algorithm

1.2)

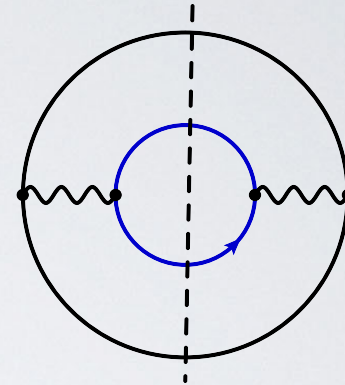
Example:  $e^+e^- \rightarrow qq$

(see notes)



$$d\hat{\sigma} = \frac{1}{2s_{ab}} d\Phi_2$$

flux



$$\langle |\mathcal{A}|^2 \rangle = \left( \frac{\alpha}{4\pi} \right)^2 N_c Q_q^2 \frac{s_{a1}^2 + s_{a2}^2}{s_{ab}^2}$$

$$s_{ab} = (p_a + p_b)^2$$

$$s_{ai} = (p_a - p_i)^2$$



1.2)

Example:  $e^+e^- \rightarrow qqg$

computation of 2 diagrams leads to:

$$\langle |\mathcal{A}|^2 \rangle = 4 \left( \frac{\alpha}{4\pi} \right)^2 \left( \frac{\alpha_s}{4\pi} \right) N_c C_F Q_q^2 \frac{s_{a1}^2 + s_{a2}^2 + s_{b1}^2 + s_{b2}^2}{s_{ab} s_{13} s_{23}}$$

$$C_F = \frac{N_c^2 - 1}{2N_c}$$

very compact - easy to  
attempt integration of the  
phase space

1.2)

## Monte Carlo integration

$$\hat{\sigma} \sim \int d\Phi_n \langle |\mathcal{A}|^2 \rangle \approx \frac{V}{N} \sum_{i=1}^N \langle |\mathcal{A}(p_i)|^2 \rangle$$

$V$  - phase space volume i.e.

$$V = \int d\Phi_n$$

For RAMBO,  $V=1$  since we map everything to  $\int_0^1 dx_i$

If we apply cuts we must apply it also the the phase space volume:  $V = N_{\text{trials}}/N_{\text{samples}}$

simple implementation is very inefficient - adaptive methods  
and importance sampling used in most applications

1.3)

## Numerical algorithms

There are many efficient methods for computing scattering amplitudes - many at tree-level. New generation of tools able to automate one-loop processes too...

OPENLOOPS, MADLOOP (aMC@NLO), BLACKHAT, **NJET**, ...

[njet-3.1.1-1.L.tar.gz](https://njet.hepforge.org/)

any of these tools would do, I happen to know NJet better than most  
[SB, Biedermann, Uwer, Yundin <https://arxiv.org/abs/1209.0100>] [SB, Moodie (2019-)]

let's try one of these as a source of data for training an efficient MC integration tool

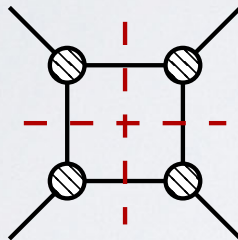


1.3)

# What is NJet doing?

recursive tree-level amplitudes  
(Berends-Giele)

$$\begin{aligned} \mathcal{J}^K(1 \dots n) &= \text{diagram with } K \text{ external lines and } n \text{ internal lines} \\ &= \sum_{K=1}^{n-1} \text{diagram with } K \text{ external lines and } n \text{ internal lines} \\ &\quad + \sum_{K=1}^{n-2} \sum_{\ell=K+1}^{n-1} \text{diagram with } K \text{ external lines and } n \text{ internal lines} \end{aligned}$$



(generalised) unitarity cuts  
(Bern, Dixon, Dunbar, Kosower),  
(Britto-Cachazo-Feng-Witten)

integrand reduction  
(Ossola, Papadopoulos, Pittau)

$$\text{diagram with } K \text{ external lines and } n \text{ internal lines} = \sum_{t \in \text{boxes}} c_{4;t} \text{diagram with } K \text{ external lines and } n \text{ internal lines} + \sum_{t \in \text{triangles}} c_{3;t} \text{diagram with } K \text{ external lines and } n \text{ internal lines} + \sum_{t \in \text{bubbles}} c_{2;t} \text{diagram with } K \text{ external lines and } n \text{ internal lines}$$



1.3)

## Installing NJET

If we manage this we have the option of exploring the NN approximation over a large class of processes...

If not, the NN examples will provide hard coded data as well

NJET is written in C++, I dependency in Fortran :(

you'll need: automake, autoconf, qd

- 1) download <https://www.davidhbailey.com/dhbsoftware/qd-2.3.23.tar.gz>  
install (click the link)
- 2) download <https://bitbucket.org/njet/njet/downloads/njet-3.1.1-1L.tar.gz>  
and follow install instructions...

1.3)

## Installing NJET (tips)

e.g. inside qd directory

(specify compilers if necessary)

```
CXX=g++-mp-10 CC=gcc-mp-10 FC=gfortran-mp-10 ./configure \  
--enable-fortran --prefix=$HOME/local/qd/ --with-pic  
make -j && make install  
export PKG_CONFIG_PATH=$PKG_CONFIG_PATH:$HOME/local/qd/lib/pkgconfig
```

e.g. specify compilers if necessary, use 'oneloop' package for integrals

```
CXX=g++-mp-10 CC=gcc-mp-10 FC=gfortran-mp-10 ./configure --with-  
oneloop --prefix=$HOME/local/njet  
make -j && make check -j && make install
```