4 Lecture 3: LHC physics with MadGraph

4.1 What is MadGraph?

MadGraph is a Monte Carlo event generator for collider studies, nowadays widely used to simulate events at the LHC. Before the LHC era, this tool was used to obtain future predictions in new physics models. Currently, one can also recast the results of the searches published by the LHC collaborations and interpret these analysis in specific models.

The MadGraph software can be extended to incorporate several programs: a random event generator, the code Pythia, used for parton showering and hadronization, and two detector simulators (PGS and Delphes). This suit allows for a complete simulation at the LHC, from events at the parton level to detector response.

Depending on the type of simulation we are interested in, some of these additional pieces might be unnecessary. For example, in case we just want to compute a cross-section at the parton level, it suffices to use the basic MadGraph software. However, if we want to go beyond and include hadronization or detector simulation we will have to use Pythia and PGS or Delphes as well. This may sound a little bit complicated, but in practice the combination of these tools is straightforward, and in fact the MadGraph suit is prepared to do it automatically.

4.2 MadGraph: Technical details, installation and load

- **Name of the tool:** MadGraph (more precisely, in this course we will use MadGraph5_aMC@NLO)
- **Author:** *The MadTeam*, composed by Johan Alwall, Rikkert Frederix, Stefano Frixione, Michel Herquet, Valentin Hirschi, Fabio Maltoni, Olivier Mattelaer, Hua-Sheng Shao, Timothy J. Stelzer, Paolo Torrielli and Marco Zaro. They can be contacted through the MadGraph website.
- **Type of code:** Python
- **Website:** http://madgraph.hep.uiuc.edu/
- **Manual:** See Refs. [22, 23].

The version of MadGraph (MadGraph5_aMC@NLO) we are going to use needs Python version 2.6 or 2.7 and gfortran/gcc 4.6 or higher (in case of NLO calculations). Besides those two requirements, which will be fulfilled in most computers, there is no need for an installation of MadGraph. The only *special* requirement is to register in the MadGraph website before being able to download the latest version of the tool. Once this registration is done and the file is downloaded, we just untar it as usual

```
$ cp Download-Directory/MG5_aMC_vX_Y_Z.tar.gz $PATH/
$ cd $PATH
$ tar -xf MG5_aMC_vX_Y_Z.tar.gz
```

Here X_Y_Z is the version that has been downloaded. And then, in order to load MadGraph we just get into the untarred folder and run the binary file

```
$ cd $PATH/MG5_aMC_vX_Y_Z
$ bin/mg5_aMC
```

This opens MadGraph. In principle, we would be ready to use it. However, before we do so let us configure some details and install additional tools that can be added to the MadGraph suit.

Configuration and installation of additional tools

Let us first comment on how to configure some options in MadGraph. We can see that in the folder input there is a file called mg5_configuration.txt. This file contains the configuration details of MadGraph, including options such as the prefered text editor (which can be user to edit some input files, the so-called *cards*, before starting the simulation), the prefered web browser (some of the results obtained in our MadGraph runs are shown in a user-friendly way with a web browser) or the time given to the user to answer questions by the code (some optoinal calculations can be switched on or off at some intermediate steps in the runs). By default, these are vi, Firefox and 60 seconds. In case you do not like these choices, you can simply modify them by removing

the # symbol in front of `text_editor`, `web_browser` and `timeout`. For example, many people will prefer `emacs` instead of `vi`.

mg5_configuration.txt

```
42 text_editor = emacs
```

Once configured, we can consider adding further pieces to the MadGraph puzzle. With the MadGraph code that we just downloaded, untarred and configured we can run simulations at the parton level. This means that we are interested in *core* processes, such as $e^+e^- \to q\bar{q}$. This is already sufficient for many collider studies, for example those aiming at the determination of an approximate number of expected events for a given process in a given new physics model. However, reality is way more complicated. The products of the process will suffer many complex processes after the collision, such as hadronization and showering, and they must be taken into account in realistic simulations. Furthermore, detectors are not perfect, and their simulation must also take into account many factors, such as inefficiencies.

> **TIP:** Although we will not use some of these additional codes for the moment, it is convenient to install them as soon as possible. This way we will be able to detect incompatibility issues which later, when we have already configured and run MadGraph many times, might be problematic.

For this reason, we may want to install Pythia and PGS. These two additional codes are necessary if you are interested in hadronization and detector response. otherwise, you do not need to install them. However, before we install these two codes we must install a prerequisite tool: ROOT. This popular code is an object oriented framework for large scale data analysis, and has been developed by CERN. In order to take advantage of the rest of the tools, we should install ROOT before. The way this is done is explained in Appendix E.

Once ROOT is already installed, the installation of Pythia and PGS is trivial. We just have to open MadGraph and execute a command:

```
$ bin/mg5_aMC
$ install pythia-pgs
```

This way, MadGraph makes sure that these two codes are installed in the correct locations and makes the required links to combine them in future simulations. However, note that you must be connected to the internet for this command to work.

Last but not least, there is another useful code: MadAnalysis. This piece of the MadGraph suit is devoted to the analysis of our simulations and is usually employed for plotting purposes. The output of our simulations is given in several formats and we must use a tool to *extract* the relevant information (this can be done, for example, with ROOT). MadAnalysis simplifies our life. One can use it to read the MadGraph output and present the results in a graphical way. Since we will interested in such graphical representations, we are going to incorporate it to our MadGraph suit.

Like MadGraph, MadAnalysis does not require any special installation. It only requires to have ROOT properly installed.

```
$ cp Download-Directory/MadAnalysis5_vX.tar.gz $PATH/
$ cd $PATH
$ tar -xf MadAnalysis5_vX.tar.gz
```

Here X is the MadAnalysis version that we have downloaded. Once this is done, MadGraph will be absolutely ready to run our own collider studies.

4.3 General usage and description of the input files

As already explained, the MadGraph suit allows for different levels of sofistication in the simulation:

Events at the parton level (MadEvent) $\Rightarrow$ Showering and hadronization (Pythia) $\Rightarrow$ Detector response (PGS or Delphes)

A complete review of all the possibilities is clearly beyond this course. Fortunately, there are many resources to learn about `MadGraph` in the internet: courses, guides and presentations. The first thing we can do is to run the tutorial provided with `MadGraph`:

```
$ cd $PATH/MG5_aMC_vX_Y_Z
$ bin/mg5_aMC
MG5_aMC> tutorial
```

This tutorial shows the basic steps to simulate $t\bar{t}$ production in the SM. In fact, we should note that when `MadGraph` loads the model that is loaded by default is the SM: we can read on the screen `Loading default model: sm`. The first command of the tutorial is the generation of a new process

```
MG5_aMC> generate p p > t t~
```

This tells `MadGraph` that we want to simulate

$$pp \to t\bar{t}$$

Next, we must export the process to several formats. This is obtained with the `MadGraph` command

```
MG5_aMC> output MY_FIRST_MG5_RUN
```

This command will do all sorts of things. Among them, it will create a new folder called `MY_FIRST_MG5_RUN`. This is where the information about this process the output of all future runs are saved. There is a sub-folder we should visit. It is the one named `Cards`. It is full of `dat` files with information about the process and about the simulation we are about to begin. There are two files of special relevance to us: `param_card.dat` and `run_card.dat`. The first one contains the input values for all parameters of the model, whereas the second one sets the parameters of the run itself. For example, the user can determine in the `run_card.dat` file the beam type, the energy, the renormalization scale or the number of random events to launch in the Monte Carlo.

The next step in the tutorial is to launch the event generator itself. For this we must execute the command

```
MG5_aMC> launch MY_FIRST_MG5_RUN
```

After we push enter, a question will appear on the screen. `MadGraph` needs to know the type of run. We will have 60 seconds to answer (unless we changed this option in `mg5_configuration.txt`. You can see that the options `pythia` and `pgs` are `OFF`. In order tu run a complete simulation including hadronization and showering (with `Pythia`) and detector response (with `PGS`) we must press `2`. Since `PGS` requires `Pythia` to run before, `MadGraph` will ask for confirmation. We can simply push enter. Then, a new question will appear. `MadGraph` wants to know if we want to modify any of the cards or just prefer to use the ones by default. For the moment we will simply stick to the ones by default in the `Cards` sub-folder. Therefore, we just go ahead by pushing enter. And then the web browser will open. This is a very user-friendly feature of `MadGraph`. As soon as the simulation begins, all information is nicely displayed using a web browser. After some time (not too long) `MadGraph` will be done. We will be able to read the results on the terminal or on the web page shown by the web browser. Either way, we find that the cross-section for $t\bar{t}$ production is about 504.9 ± 0.8 pb. Of course, the exact number might be different (it has been obtained by Monte Carlo methods), but it should agree within the error.

We can also explore these results using the information on the generated web page. The first thing we can see is that `MadGraph` shows the Feynman diagrams used for the computation of the events. These can be found in `Process information`. For $t\bar{t}$ production in the SM these can be classified into two types: gluon-induced and quark-induced diagrams. Moreover, in the first case there are three sub-diagrams. If we click on `Results and event database` we can see the numerical results obtained with the simulation. By clicking on the cross-section result, we can even read the individual contributions given by the different Feynman diagrams. In our case, the dominant production channel is the gluon-induced one.

This concludes the tutorial, where we already run an interesting calculation.

4.4 Computing a cross-section

As an example of what we can do with `MadGraph`, we are going to compute a production cross-section in the scotogenic model. For more fancy simulations we refer to the next lecture, where we will make use of all the tools installed in Sec. 4.2.

Since the scotogenic model is not among the models provided with `MadGraph`, we must add it. In the `MadGraph` folder there is a sub-folder named `models`, where all the model definitions are saved in UFO format. In order to implement the scotogenic model, we just have to copy the files that we produced with `SARAH` in the first lecture. These are located in `$PATH/SARAH-X.Y.Z/Output/Scotogenic/EWSB/UFO`. Therefore, we must execute the following terminal commands:

```
$ cd $PATH/MG5_aMC_vX_Y_Z/models
$ mkdir Scotogenic
$ cp $PATH/SARAH-X.Y.Z/Output/Scotogenic/EWSB/UFO/* Scotogenic
```

Next, we go back to the main `MadGraph` folder, open `MadGraph` and load the model:

```
$ cd ..
$ $ bin/mg5_aMC
MG5_aMC > import model Scotogenic -modelname
MG5_aMC > define p d1 d1bar d2 d2bar u1 u1bar u2 u2bar g
```

Note that we added the option `-modelname`. This is used to keep the names of the particles as given in the `SARAH` model files. Although sometimes this will not be necessary, it is convenient to use this option when loading models created with `SARAH`. Next, we defined the multiparticle `p`, including the gluon (`g`) and all the SM quarks. This is required because we are not using `MadGraph`'s naming conventions, and thus the multiparticle states must be redefined.

Now we can run a simple simultation where we compute the cross-section for the production of a pair of η scalars, a CP-even neutral one and a charged one,

$$p\,p \to \eta_R\,\eta^+$$

We do this with the command

```
MG5_aMC > generate p p > etr etp
```

Next, we can create the output folder, which we will call `SimScotogenic`, and launch the simulation

```
MG5_aMC > output SimScotogenic
MG5_aMC > launch SimScotogenic
```

To the first question we will simply press enter (equivalent to option `0`), since we just want to compute the cross-section at partonic level. Then we will get the second question, asking about whether we want to modify any cards. And in this case we cannot simply press enter. The `param_card.dat` by default does not correspond to our **BS1** benchmark point. In fact, all the scotogenic model parameters are zero in the file by default. Therefore, we must replace the file by one with the correct input values for the model parameters. Since the `param_card.dat` file uses the standard LesHouches format, we can simply use the `SPheno.spc.Scotogenic` file that we generated with `SPheno` as input for `MadGraph`. This is as simple as typing in the terminal

```
$ cp $PATH/SPheno-X.Y.Z/SPheno.spc.Scotogenic ↪
    $PATH/MG5_aMC_vX/SimScotogenic/Cards/param_card.dat
```

Here, remember, `X.Y.Z` and `X` are the versions of the two codes. This way, we have replaced the `param_card.dat` file by default by one of our own, with the correct parameter values in the **BS1** benchmark point. Once

this is done, we can press enter and continue with the simulation. MadGraph will print some warning messages. These are due to some format issues in the SPheno.spc.Scotogenic file. However, they are not relevant at all. After a few minutes, we will get a result for the cross-section. This is found to be $\sigma = 0.7943 \pm 0.0008$ fb (or something very similar, since this is a random simulation).

We actually expected to get a small cross-section. The $\eta_R\,\eta^+$ states are produced at the LHC by electroweak interactions. This is shown in the Feynman diagrams produced by MadGraph, see Process Information in the web page, where we see that $p\,p \to \eta_R\,\eta^+$ is induced by W^+ s-channel exchange. Therefore, it is not surprising that the resulting cross-section turns out to be much lower than the usual QCD induced cross-sections at the LHC. With an integrated luminosity of 300 fb^{-1}, as expected by the end of the LHC Phase I, we would obtain about 240 events of this type.

MadGraph can be also used to generate all possible chains (Feynman diagrams) leading to a specific final state. For example, in the case we just studied, the η_R and η^+ scalars are not stable, but they decay to final states including the lightest right-handed neutrino N_1. For example, one can have the decays

$$\begin{aligned}\eta_R &\to N_1\,\nu\\ \eta^+ &\to N_1\,\mu^+\end{aligned}$$

where ν is any of the light neutrinos (undistinguishable at the LHC). Therefore, we eventually get the process

$$p\,p \to N_1\,N_1\,\mu^+\,\nu \tag{25}$$

wich would be seen as an antimuon plus missing energy at the LHC. However, the final state $N_1\,N_1\,\mu^+\,\nu$ can be reached by other production mechanisms (not only through intermediate $\eta_R\,\eta^+$). How can we obtain all possible topologies in the scotogenic model leading to $N_1\,N_1\,\mu^+\,\nu$? MadGraph can do it for us.

In fact, in order to see all possible diagrams leading to a specific final state we do not even need to launch the simultation. We just have to generate it. In this case we just need open MadGraph and run a few commands

```
$ bin/mg5_aMC
MG5_aMC> import model Scotogenic -modelname
MG5_aMC> define p d1 d1bar d2 d2bar u1 u1bar u2 u2bar g
MG5_aMC> generate p p > n1 n1 e2bar vl
MG5_aMC> output SimScotogenic2
```

Note that in the generation of the process we have used the multiparticle state vl, containing all light neutrinos. This simulation would take quite a long time before is finished. However, we can already see the Feynman diagrams in the web page generated by MadGraph. We can load it with

```
MG5_aMC> open index.html
```

Then, we just have to browse via Process Information until we find all the Feynman diagrams that participate in this process. The Feynman diagram with intermediate $\eta_R\,\eta^+$ scalars is diagram 5, and it is just one among many. We also find diagrams where only one of them, η_R or η^+, participates.

> **TIP:** By default, MadGraph uses 10^4 events for a simulation. This number will be sufficient in many cases, but not for rare LHC events. We must then use a larger number of events (10^5 or 10^6) when we simulate events with small cross-sections. Otherwise, the resulting simulation will contain large statistical errors.

4.5 Summary of the lecture

MadGraph is one of the most popular computer tools in particle physics, due to its high level of sofistication. In this lecture we learned how to use it to run simple LHC simulations. Since this code offers many more possibilities, clearly beyond the scope of this introductory course, we strongly encourage to explore its potential capabilities. The internet is full of resources to learn how to use MadGraph, from slides to tutorials, including many detailed guides.