Run of MadGraph on GPU with CUDA backend and LHAPDF and profile with flamegraphs

Runs performed on `itscrd90.cern.ch` machine:

- `gcc` 11.3.1
- Cuda compilation tools, release 12.0, V12.0.140
- `madgraph4gpu@a87e64037a8c941bf2ec3bfd78e7a38578c1d1b8`

Building LHAPDF

For the python API, need Python and Cython.

Clone the repo and build with the following options, so that debug symbols are included:

```
autoreconf -i
./configure --prefix=<prefix_absolute_path> --enable-static CXXFLAGS="-O2 -g -fno-omit-frame-pointer"
make -j4
make install
```

Set the paths:

```
prefix=<prefix_absolute_path>
export PATH="$prefix/bin":$PATH
export PYTHONPATH="$prefix/lib64/python3.9/site-packages:$PYTHONPATH" # if built with python
export LD_LIBRARY_PATH="${prefix}/lib:$LD_LIBRARY_PATH"
export LHAPDF_DATA_PATH="/cvmfs/sft.cern.ch/lcg/external/lhapdfsets/current/"
```

`LHAPDF_DATA_PATH` contains the downloaded PDF sets, so no need to do that manually.

Compile MadGraph with LHAPDF

Needs to make few changes:

- source the LHAPDF environment variables as shown above;
- add the `lhapdf-config` file to the `../../Source/make_opts` flags:

  ```
  lhapdf=<prefix_absolute_path>/bin/lhapdf-config
  ```

Compile MadGraph with the usual commands, see HERE for more details.

```
make BACKEND=cuda OMPFLAGS= -f cudacpp.mk -j4
make -C ../../Source -j4
make BACKEND=cuda -j4
```

Running MadGraph and various issues

Fail 1: Could not find PDFsets directory, quitting

Due to MadGraph automatic check of the folder where the PDF sets are stored:

- looks for `lib/PDFsets` somewhere in the project directory;
- done in `../../Source/PDF/pdfwrap_lhapdf.f`, around line 81, in the subroutine `FINDPDFPATH()`.
- not a robust check, only looking for at most 6 folders up from the current directory: if that folder is found alsewhere or it is not called `lib/PDFsets` *literally*, then it is not found and the error is thrown;
- **however** check is not needed because path is exported in `LHAPDF_DATA_PATH` environment variable and `LHAPDF` will **always** find the PDFs by itself.

Here is the snippet from `../../Source/PDF/pdfwrap_lhapdf.f`:

```
      LHAPATH='lib/PDFsets'
      INQUIRE(file=LHAPATH, EXIST=EXISTS)
      IF(EXISTS)RETURN
      UPNAME='../../../../../../../'
      DO I=1,6
        TEMPNAME2=PATH(:FINE2)//UPNAME(:3*I)//LHAPATH
```

```
C       LHAPath=up//LHAPath
        INQUIRE(file=TEMPNAME2, EXIST=EXISTS)
        IF(EXISTS)THEN
          LHAPATH = TEMPNAME2
          RETURN
        ENDIF
      ENDDO
      PRINT*,'Could not find PDFsets directory, quitting'
      STOP
```

SOLUTION: skip this function by commenting its call out. Additionally, comment also the `SETPDFPATH` subroutine call for the same reason: that subroutine will just set the path inside the LHAPDF object, which is not needed if the path is already set in the environment variable.

Fail 2: PDLABELnn23xxx not found

This error occurs because, when using `LHAPDF` library, the `PDFLABEL` should be set to `lhapdf`: To be changed in:

- `Cards/run_card.dat`;
- `Source/run_card.inc`.

Results

Process: $gg \to t\bar{t}gg$.

Perform runs with very high number of events, so that the time-consuming part stands out more with respect to the GPU initialization and setup. Use the following `input.txt`:

```
       262144         2           2      !Number of events and max and min iterations
0.1    !Accuracy
0       !Grid Adjustment 0=none, 2=adjust
0       !Suppress Amplitude 1=yes
0        !Helicity Sum/event 0=exact
1
```

This will generate a total of **802816** events.

Additionally, modify the flamegraph script to not print the `unknown`, but to print the symbol name followed by the pointer address, so that the names are unique and the various unknown blocks are not merged within each other.

Runs have been performed both with and without LHAPDF library (to compare the time saved by just using the C++ implementation instead of the native FORTRAN implementation in MadGraph), and both FORTRAN and CUDA backends (to check that the number of calls to LHAPDF is consistent).

CUDA

- **w/ LHAPDF**
- **w/o LHAPDF**

FORTRAN

- **w/ LHAPDF**
- **w/o LHAPDF**

Timings as given by `madevent`

Average of 10.

	CUDA w/ LHAPDF	CUDA w/o LHAPDF	FORTRAN w/ LHAPDF	FORTRAN w/o LHAPDF
FORTRAN Overhead	10.4023	17.4328	9.72641	16.7207
CUDA cpp MEs	2.15152	2.14973	408.712	407.811
Program total	12.55382	19.58253	418.43841	424.5317

Perf stat

- **w/ LHAPDF**

- **w/o LHAPDF**

Comparison with/without LHAPDF:

```
         12,774.34 msec task-clock                       #     |         19,868.78 msec task-clock                       #
               292      context-switches                 #   2 |               344      context-switches                 #   1
                 6      cpu-migrations                   #     |                 4      cpu-migrations                   #
             5,749      page-faults                      #  45 |             4,603      page-faults                      #  23
    38,738,465,319      cycles                           #     |    60,358,592,417      cycles                           #
    79,843,864,971      instructions                     #     |   154,662,612,426      instructions                     #
    15,611,188,417      branches                         #     |    29,373,142,218      branches                         #
       124,217,959      branch-misses                    #     |       115,744,389      branch-misses                    #

      13.347107965 seconds time elapsed                        |      19.919600172 seconds time elapsed


      12.045233000 seconds user                                |      19.086128000 seconds user
       0.650045000 seconds sys                                 |       0.660382000 seconds sys
```

Comments

- With around 800k events, already with LHAPDF we see a speedup of 3x with respect to the native MadGraph implementation.
- CUDA: `pdg2pdf` passes from the 40% with no LHAPDF to 9% with LHAPDF.
- CUDA: `pdg2pdf` is called 3 times (or, at least, we can record 3 records, which means it is called *at least* 3 times) in both scenarios with/without LHAPDF, and it takes the same time. This means it is not probably been cached like Oliver once suggested it should be.
- FORTRAN: in the case without LHAPDF, the `pdg2pdf` is called 3 times, each one with the same time length. Also not cached as expected.

Next steps

- Profile the current FORTRAN release of MadGraph to understand whether the caching mechanism of the PDFs is working (this could have been overlooked while implementing the CUDA version).
- Profile with `AdaptivePerf` to both have a new way of generating flamegraphs and to also have a chronological view of the code.