News on DPMJET and impy

Anatoli Fedynitch

High-Energy Theory Group, Institute of Physics, Academia Sinica

CORSIKA8 Meeting, Heidelberg, 2022/07/14

Overview

1. DPMJET-III 19.3 update

2. Event generator interface impy

DPMJET-III

years, max. E per nucleon

Hahn & Ranft (1990)

DTUJET

p-p collisions,
only gluon jets

Ranft (1997)

DPMJET II

DPMJET II.5

h-A, A-A @ cosmic ray energies

EOL!

Möhring & Ranft (1990)

DTUNUC

h-A, A-A, no minijets

Rösler & Ranft (1996)

DTUNUC 2

GVDM

Rösler, Engel & Ranft (2000)

DPMJET III (3.0.6)

h/A-A/A, ɣ-h/A, ɣ-ɣ;

Engel & Ranft (1995)

PHOJET

h-h, e/ɣ-h, e/ɣ-e/ɣ

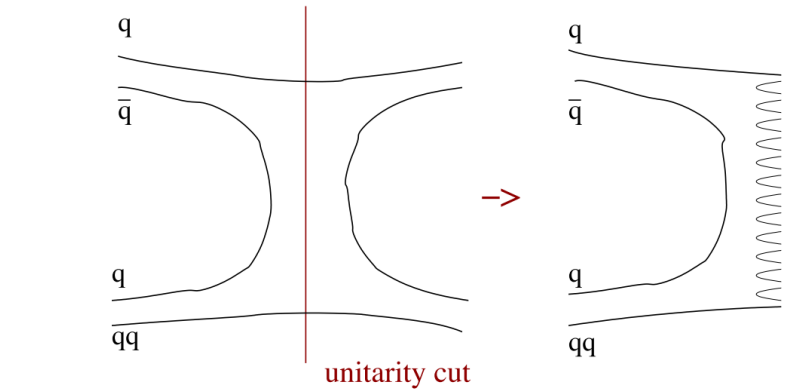
AF & Engel (2012-2016)

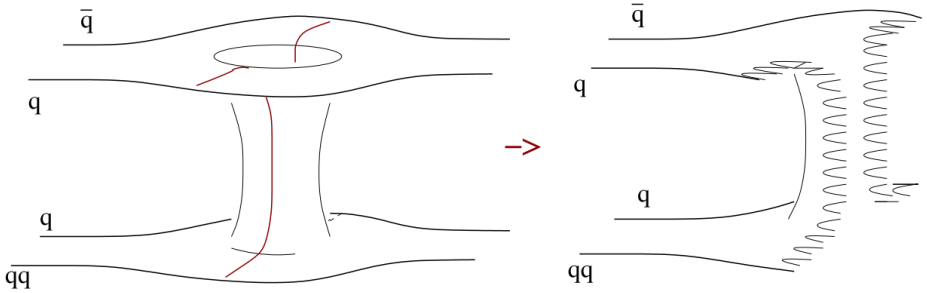
DPMJET III 17.1+
(no separate PHOJET distribution)

DPMJET-III

- Distribution via github https://github.com/DPMJET/DPMJET
- Pure FORTRAN
- High-energy hadronic model and nucleus-nucleus model in FLUKA and FLUKA
- Can do (virtual) photon-nucleus at high energy
- Output in HEPEVT-style common blocks
- Fast: factor 2-3 slower than SIBYLL in pp
- Works at low GeV energies
- Intra-nuclear cascade
- Can do evaporation in combination with FLUKA
- Separate cross sections for pi+, pi-, K+, … on proton target

Physics: Color Flow + Unitarization + Glauber



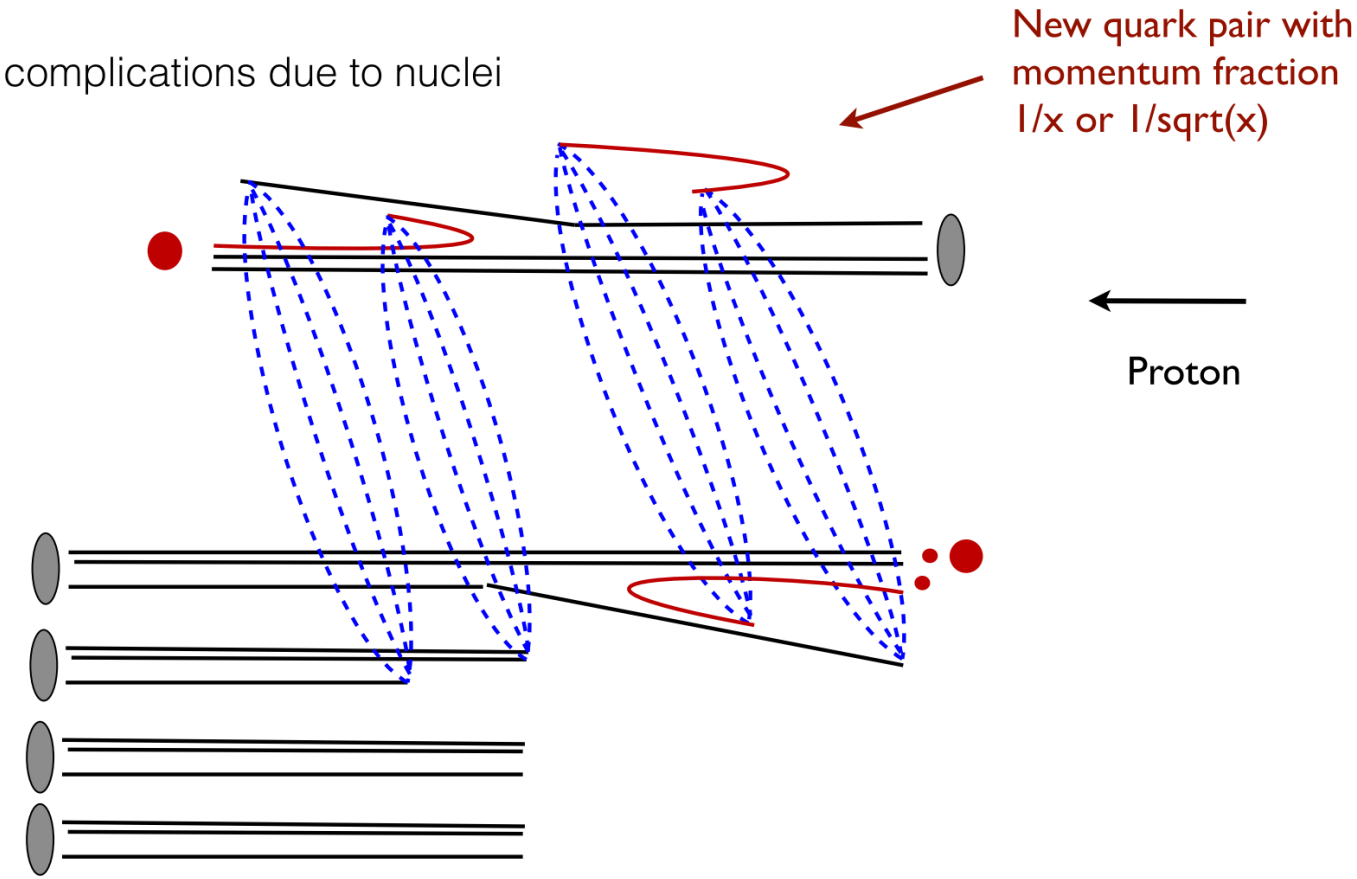


elastic scattering

inelastic scattering

Nucleus

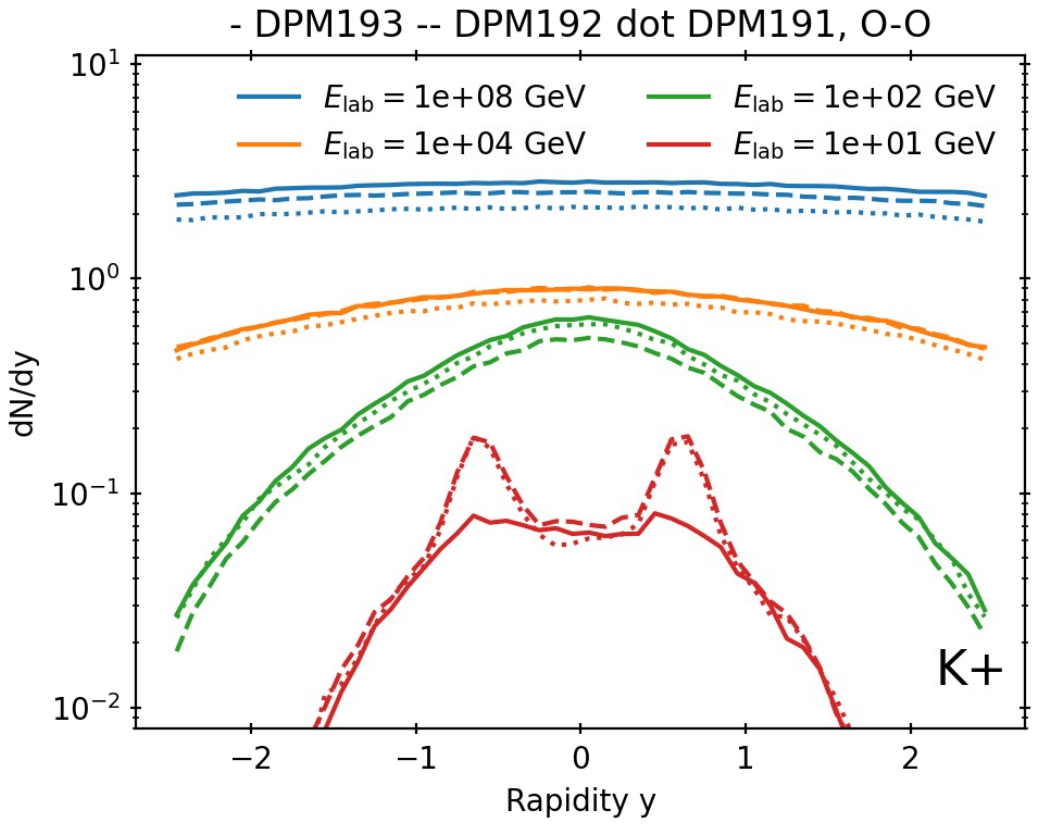
Additional complications due to nuclei



Spectator nucleons: remnant nucleus

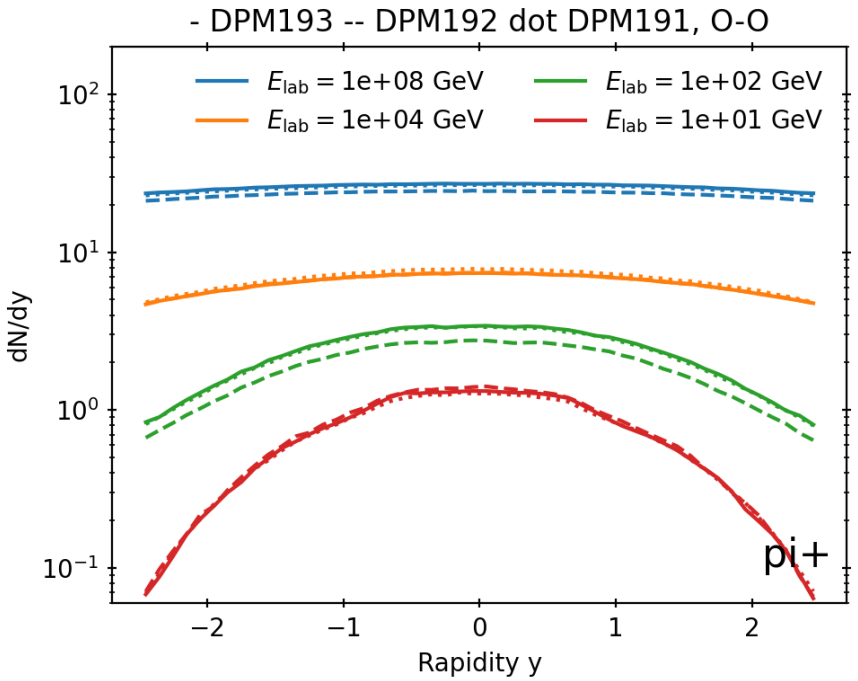
Main news in DPMJET 19.3

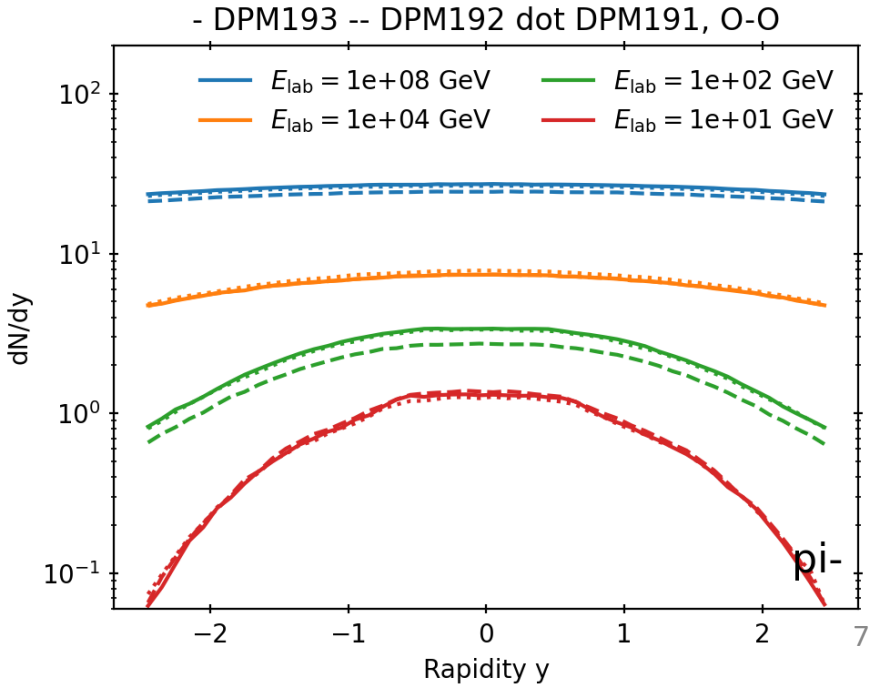
- In the case of
  1. low energy nucleus-nucleus interactions (<35 GeV lab)
  2. low mass valence-sea strings (< 0.8 GeV)
  3. and strange quarks string ends
- Fragmentation resulted in production very slow K+
- too small available phase space resulted in the kaons going into the a very narrow angular range (→ Rapidity-Rabbit)
- For these very light specific strings strange quarks replaced with light quarks → problem mitigated!
- In principle, conceptual problem in DPMJET because light flavors assigned before the strings are formed and masses known



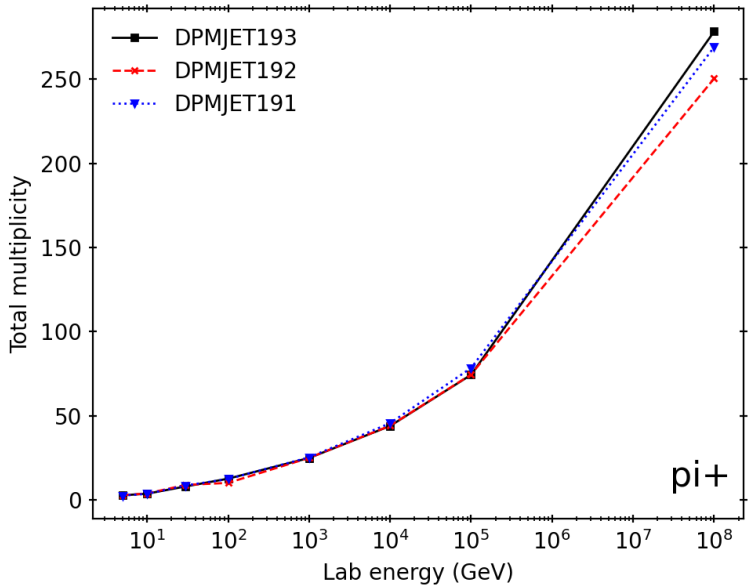
DPMJET 19.3

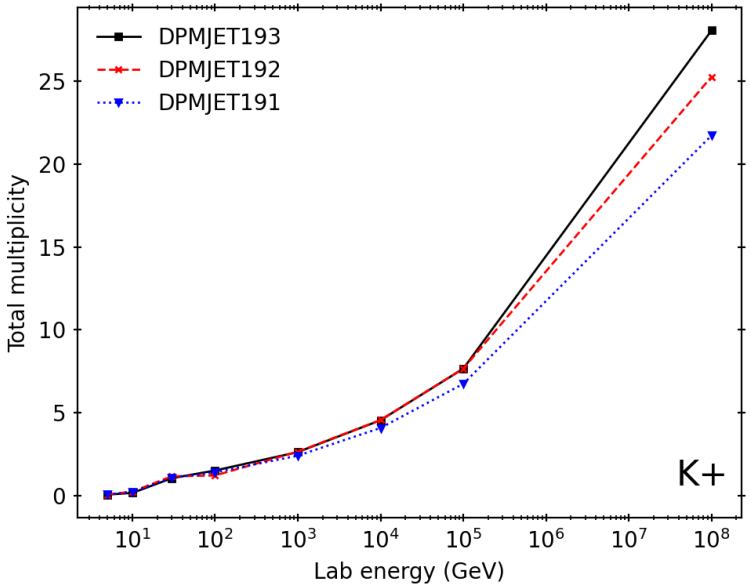
- For DPMJET 19.2, we went back to an older set of fragmentation parameters, which were inconsistent between those used in DPMJET, DPMJET in FLUKA, and PHOJET
- No significant impact observed on comparisons with data
- Data validation reports are now published along with updates (see 100p PDF file here from MCVD code, in prep for open sourcing)
- There is some impact on multiplicities from both changes

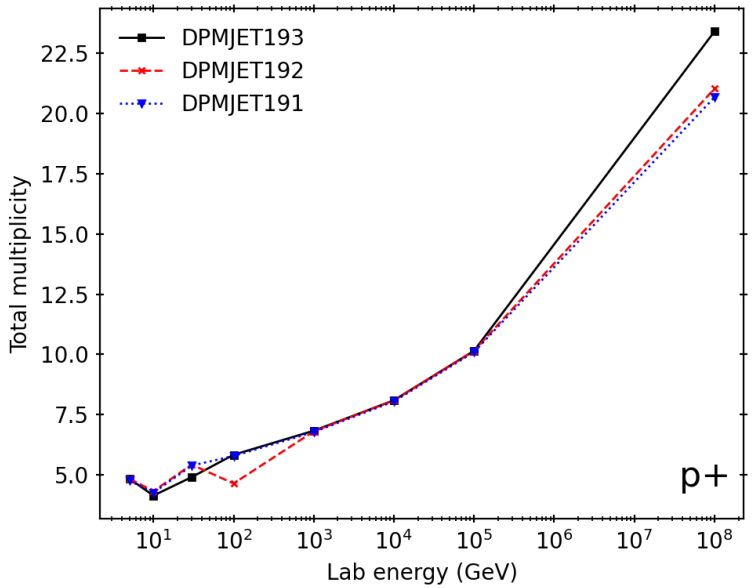


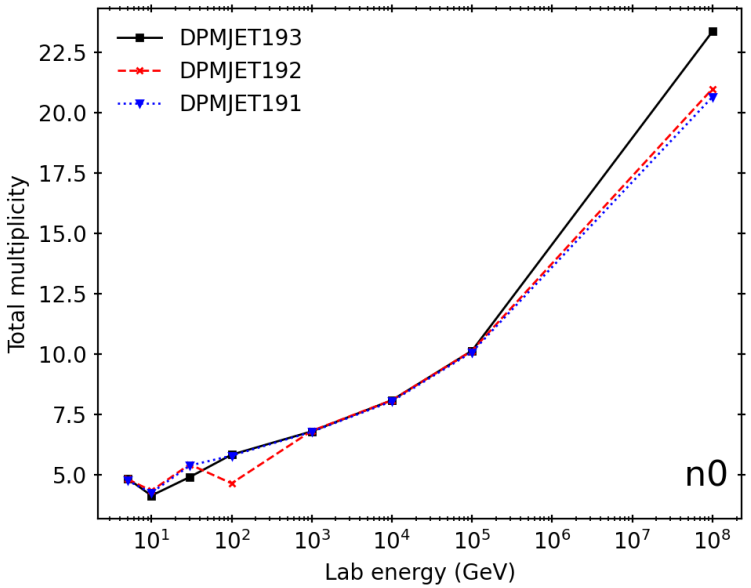


Impact on multiplicities

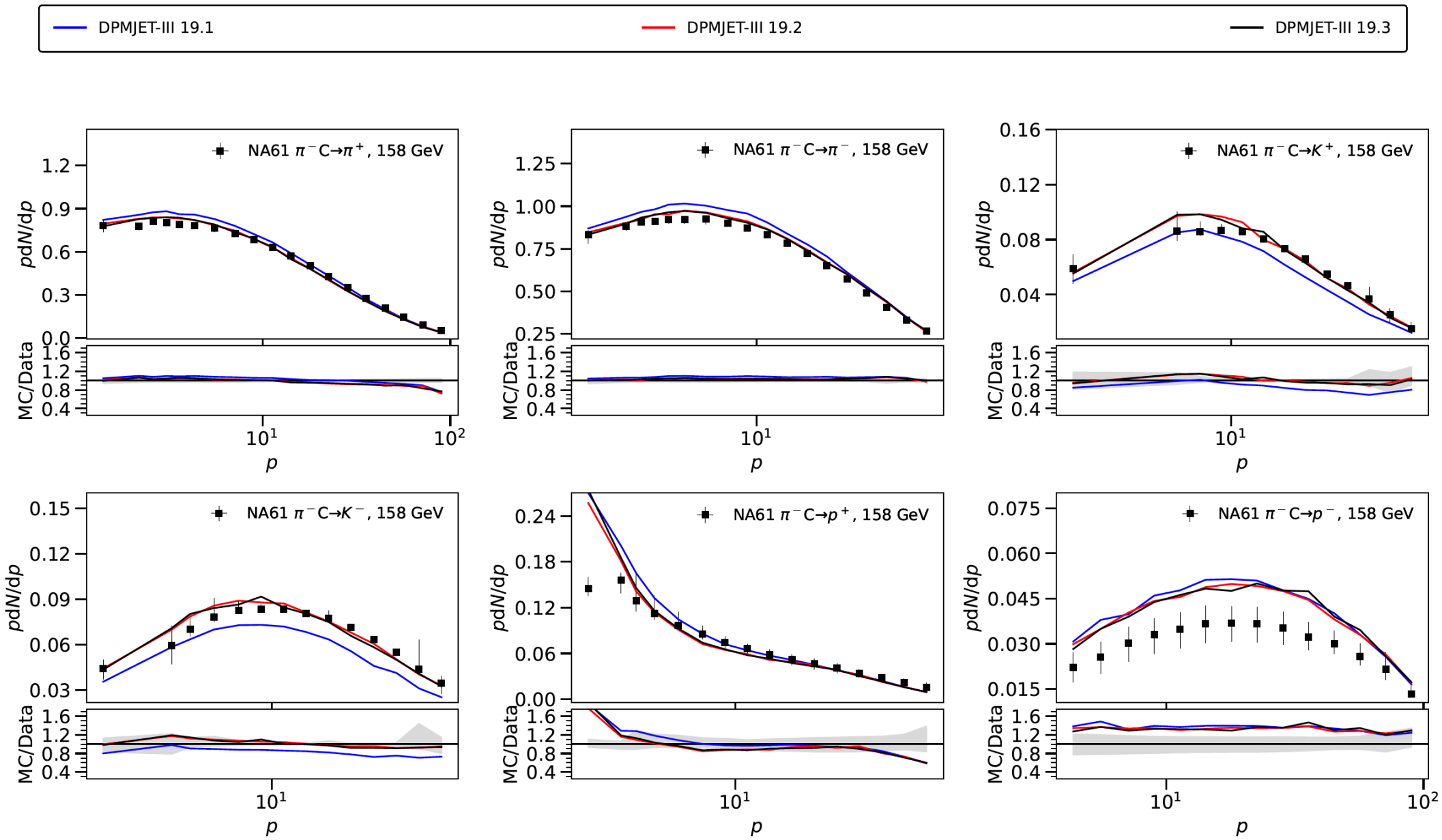








Some improvement for pion-carbon

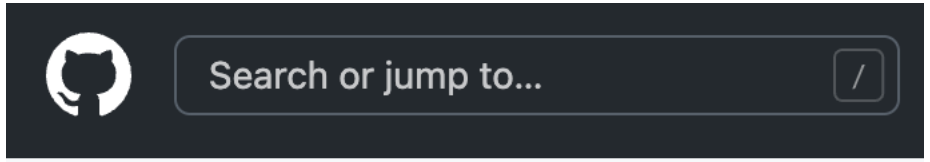


(eternal) TODO's

- Look more carefully at extremely high energies if cross section integrals properly converge
- Charm production
  - comes from CT14 PDFs, somewhat accidentally
  - Incomplete matrix elements (and nonadequate k-factor)
  - Too hard longitudinal spectra
  - CT14LO has maybe too much charm at small-x
  - Charm quark is massless → incorrect threshold
- Baryons insufficiently flat in xf → too small elasticity
- Anti-baryons too hard (same origin as in SIBYLL 2.1)
- No s-channel photo-hadronic/-nuclear processes (such as Delta-Resonance production)
  → can not be used out of the box for photo-nuclear calculations in air-showers
- No enhanced rho-meson production in hadron-nucleus → muon number not expected to be very high
- String-fusion produces slightly asymmetric rapidity distributions in nucleus-nucleus

Interaction Models in PYthon (impy)

```python
from impy.definitions import *
from impy.constants import *
from impy.kinematics import EventKinematics
from impy import impy_config

# Define the parameters of the collisions
event_kinematics = EventKinematics(
    ecm=13 * TeV, p1pdg=2212, p2pdg=2212)

# Create an instance of an event generator by passing
# the model name as a string
generator = make_generator_instance(
    interaction_model_by_tag['SIBYLL2.3D'])

# Initialize it
generator.init_generator(event_kinematics)

# Number of events to generate
nevents = 100

for event in generator.event_generator(event_kinematics, nevents):
    event.filter_final_state_charged()
    # do something with event.p_ids, event.eta, event.en, event.pt, etc.
    # these variables are numpy arrays, that can be histogrammed or counted like
    average_pt += 1/float(nevents)*np.mean(event.pt[np.abs(event.p_ids) == 211])

print('Average pT for charged pions {0:4.3f}'.format(average_pt))
```



impy-project / impy Public

Supported models

- DPMJET-III 3.0.6
- DPMJET-III 19.1
- EPOS-LHC
- PHOJET 1.12-35
- PHOJET 19.1
- PYTHIA 6
- PYTHIA 8 (not yet bundled)
- QGSJet-01
- QGSJet-II-03
- QGSJet-II-04
- SIBYLL-2.1
- SIBYLL-2.3
- SIBYLL-2.3c
- SIBYLL-2.3d
- SOPHIA (needs update)
- DPMJET-II (also needs update but model deprecated)
- UrQMD 3.4

Authors:

- Anatoli Fedynitch
- Hans Dembinski
- Sonia El Hadri
- Keito Watanabe

New maintainer!
Anton Prosekin (ASIoP)

What it does to the fortran codes

pyf-file: definition generated by f2py (part of numpy)

```
!    -*- f90 -*-
! Note: the context of this file is case sensitive.

python module sib23d ! in
    interface  ! in :sib23d
        subroutine sibyll(k_beam,iatarg,ecm) ! in
        :sib23d:sibyll2.3d.f
            integer :: k_beam
            integer :: iatarg
            double precision :: ecm
            integer :: ncall
            integer :: ndebug
            integer :: lun
            integer :: ldiff
            double precision dimension(8000,5) :: p
            integer dimension(8000) :: llist
            integer :: np
```

+

Original fortran code

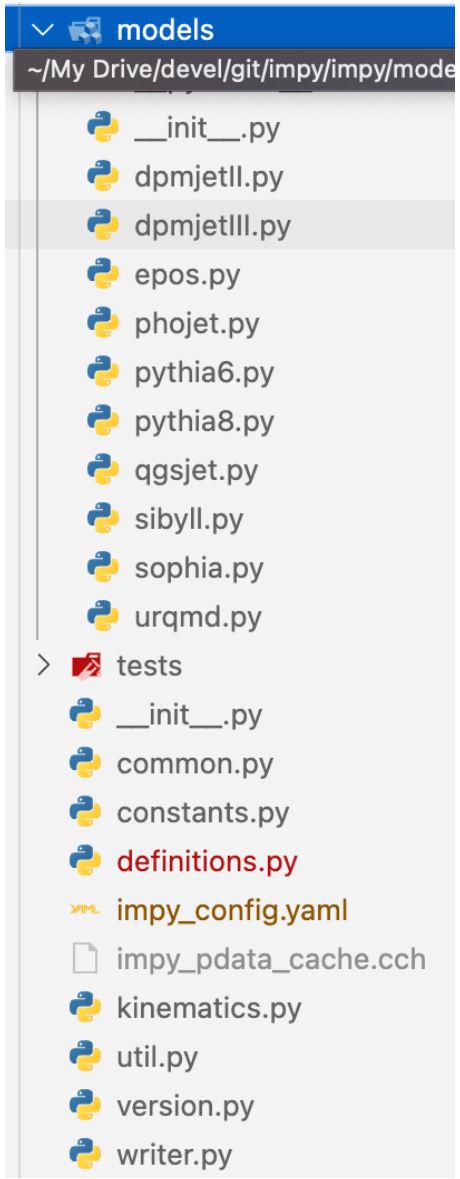
=

```
(base) anatoli@tapc604:~/devel/git/impy/impy/lib$ ipython
Python 3.9.12 (main, Apr  5 2022, 06:56:58)
Type 'copyright', 'credits' or 'license' for more information
IPython 8.2.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: import sib23d

In [2]: sib23d.sibini(1234)
 ==================================================
 |                                                |
 |                  S I B Y L L  2.3d             |
 |                                                |
 |         HADRONIC INTERACTION MONTE CARLO       |
 |                       BY                       |
 |            Eun-Joo AHN, Felix RIEHN            |
 |      R. ENGEL, A. FEDYNITCH, R.S. FLETCHER,    |
 |       T.K. GAISSER, P. LIPARI, T. STANEV       |
 |                                                |
 | Publication to be cited when using this program:|
 | Eun-Joo AHN et al., Phys.Rev. D80 (2009) 094003 |
 | F. RIEHN et al., hep-ph: 1912.03300            |
 | last modifications: F. Riehn (05/20/2020)      |
 ==================================================

  Calculating cross section tables...
 SIG_AIR_INI: initializing target: (i,A)          1          0 air .
 SIG_AIR_INI: initializing target: (i,A)          2         14 nit .
 SIG_AIR_INI: initializing target: (i,A)          3         16 oxy .
```

Shared libraries that can be imported from from Python

```
(base) anatoli@tapc604:~/devel/git/impy$ ls impy/lib/
dpmjet306.cpython-39-darwin.so              qgsII03.cpython-39-x86_64-linux-gnu.so  sib23c02.cpython-39-darwin.so
dpmjet306.cpython-39-x86_64-linux-gnu.so    qgsII04.cpython-39-darwin.so            sib23c02.cpython-39-x86_64-linux-gnu.so
dpmjetIII191.cpython-39-darwin.so           qgsII04.cpython-39-x86_64-linux-gnu.so  sib23c03.cpython-39-darwin.so
dpmjetIII191.cpython-39-x86_64-linux-gnu.so sib21.cpython-39-darwin.so              sib23c03.cpython-39-x86_64-linux-gnu.so
eposlhc.cpython-39-darwin.so                sib21.cpython-39-x86_64-linux-gnu.so    sib23d.cpython-39-darwin.so
eposlhc.cpython-39-x86_64-linux-gnu.so      sib23.cpython-39-darwin.so              sib23d.cpython-39-x86_64-linux-gnu.so
pythia6.cpython-39-darwin.so                sib23.cpython-39-x86_64-linux-gnu.so    sophia.cpython-39-darwin.so
pythia6.cpython-39-x86_64-linux-gnu.so      sib23c00.cpython-39-darwin.so           sophia.cpython-39-x86_64-linux-gnu.so
qgs01.cpython-39-darwin.so                  sib23c00.cpython-39-x86_64-linux-gnu.so urqmd34.cpython-39-darwin.so
qgs01.cpython-39-x86_64-linux-gnu.so        sib23c01.cpython-39-darwin.so           urqmd34.cpython-39-x86_64-linux-gnu.so
qgsII03.cpython-39-darwin.so                sib23c01.cpython-39-x86_64-linux-gnu.so
```

What happens in the python side: event generation



Each generator is wrapped by a derived class, that handles specific of intialization and setting of inputs

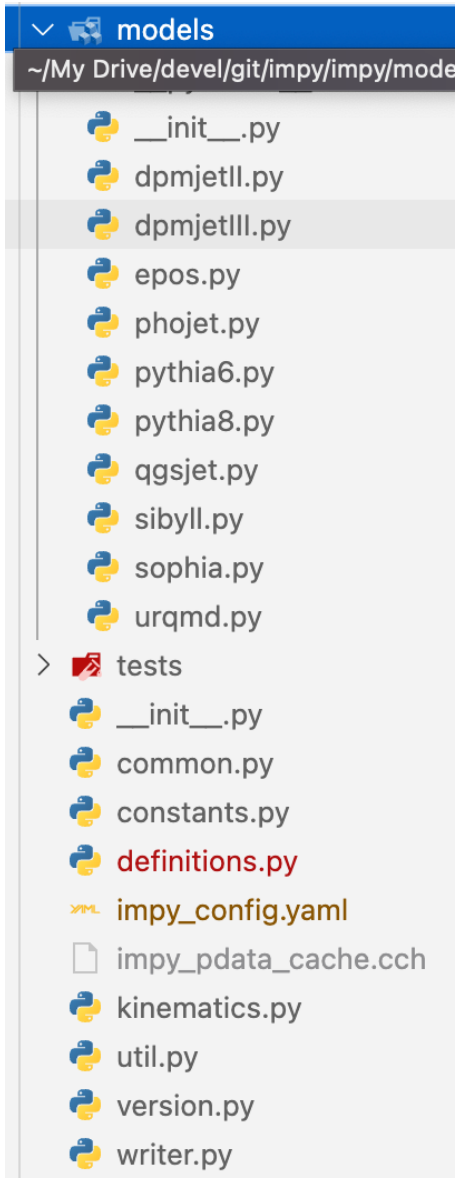
```
class SIBYLLRun(MCRun):
    """Implements all abstract attributes of MCRun for the
    SIBYLL 2.1, 2.3 and 2.3c event generators."""
```

Events are generated by calls to the original routines.

```
def generate_event(self):
    self.lib.sibyll(self._sibproj, self._iatarg, self._ecm)
    self.lib.decsib()
    self.conv_hepevt()
    return 0  # SIBYLL never rejects
```

For generators with non-standard output data structures, conversion to HEPEVT happens in a dedicated fortran subroutine. It can also contain more custom fortran code…

What happens in the python side: analyzing or exporting events



After event generation, instances of MCRun wrap the original fortran common blocks into numpy arrays (at almost no cost!)
Cost comes only when data is really used or read.

```
class SibyllEvent(MCEvent):
    """Wrapper class around SIBYLL 2.1 & 2.3 particle stack."""
    # Workaround for no data on vertext positions in SIBYLL
    _no_vertex_data = None

    def __init__(self, lib, event_kinematics, event_frame):
        # HEPEVT (style) common block
        evt = lib.hepevt
        You, 4 years ago • New event base class …
        # Save selector for implementation of on-demand properties
        px, py, pz, en, m = evt.phep
        vx, vy, vz, vt = evt.vhep
```

- Lot's of @properties that return pt2, rapidity, boost into lab frame, etc. Computed on demand…
- Can be analyzed directly in python, or written to custom files, C-structs, etc…

Using Hans' pyhempc, fotran arrays can be pumped into HepMC3 (C++) output library.

Performance comparisons between impy and CRMC

Thx to Anton Prosekin

	Sibyll-2.3d	DPMJet-III 19.1	QGSJETII-04	EPOS LHC
IMPY: pyhepmc → ASCII	1116.91 ± 31.41	474.94 ± 20.55	264.78 ± 21.61	35.99 ± 2.70
IMPY: no output	6228.12 ± 281.10	1554.85 ± 272.18	382.32 ± 18.69	43.23 ± 3.13

	Sibyll-2.3d	DPMJet-III 19.1	QGSJETII-04	EPOS LHC
CRMC: HepMC3 → ASCII	30.24 ± 6.76	16.57 ± 3.37	21.09 ± 2.99	11.14 ± 1.14
CRMC: no output	2602.06 ± 166.96	914.84 ± 43.42	305.99 ± 29.15	38.00 ± 6.40

in events/sec

- For pp collisions at 7 TeV ecm
- IMPY factor 2-> 35 faster.
- CRMC (no output) shown with subroutine “bjinta” commented out. With that it’s a factor 5 slower.
- IMPY + pyhepmc produce correct output only for DPMJET so far, problem is under investigation by Anton

Summary and outlook

- **DPMJET-III** is maintained. Sometimes there are bugs, and they get fixed, but on long timescales. Some improvements in version 19.3.
- Please always use the official version from github! Post issues.
- The TODO list is long and the model has sufficient advantages and good features to survive. If you, someone you know, or will know in future might be interested in working on it, please contact me.

- **impy** works well despite some remaining cosmetical issues
- Anton Prosekin (my team) has been tasked to maintain it and to work making it public, so it will improve soon. Contributions welcome.
- People who use it never ask questions claiming to work with it. So it's a good sign that the code mostly works.
- It is faster then CRMC and easier to maintain.
- It can be suitable as a generic interface to CORSIKA8 and for testing.
- For the main models in C8, I support the current approach of manual implementation.