Generation of counterterms in MadNkLO

Simone Lionetti

April 1, 2019

This document describes the contents of the file `subtraction.py` (located in `madgraph/core`).

1 Classes

1.1 `SubtractionLeg`

For the purpose of the subtraction, it is impractical to carry around the whole information that is contained in an object of the type `base_objects.Leg`. A simpler object `SubtractionLeg` is therefore defined with the following three attributes:

- `n`: an integer that indicates the leg number in the process,
- `pdg`: the PDG identifier which specifies the type of particle,
- `state`: a flag to specify if the leg is in the initial or in the final state.

For convenience, a class `SubtractionLegSet` which represents a set of `SubtractionLeg`s is also implemented. Internally, this is just a sorted `tuple` since it is assumed that order is irrelevant. However, this object also provides some additional useful methods.

1.2 `SingularStructure`

The `SingularStructure` class is designed to identify unresolved regions of phase space, and by extension counterterms. It is a recursive structure which represents a tree of `SubtractionLeg`s in a process. At each level, the leaves are gathered into a `SubtractionLegSet` attribute called '`legs`' and the `SingularStructure` that specify sub-trees are grouped into a `list` called '`substructures`'. There are currently three classes that inherit from `SingularStructure` and represent different types of phase space regions:

- `SoftStructure` indicates that all of its sub-legs are soft,
- `CollStructure` indicates that all of its sub-legs are collinear,
- `BeamStructure` specifies that a leg is taken from a hadron beam after some splitting.

The generic parent class `SingularStructure` can be instantiated to group together several structures and specify an additional set of legs; this feature is exploited in the implementation of momentum mappings to specify recoilers.

For the sake of concreteness, a non-trivial example of `SingularStructure` is illustrated in fig. 1. In the conversion to a string, by default the PDGs and the state labels of `SubtractionLeg`s are suppressed, and `SingularStructure`s are converted to a single character such that the object of fig. 1 prints as

$$\texttt{C(C(C(5,13),C(7,10,11,16)),S(4,6,8),1,9).} \tag{1}$$

The conversion of objects to characters is carried out through the method `name`, according to the rules

$$\begin{aligned} &\texttt{SingularStructure} \to \text{`\,'}, && \texttt{CollStructure} \to \text{`}\texttt{C}\text{'}, \\ &\texttt{SoftStructure} \to \text{`}\texttt{S}\text{'}, && \texttt{BeamStructure} \to \text{`}\texttt{F}\text{'}. \end{aligned} \tag{2}$$

2 Generation of elementary operators

Given a process, the first

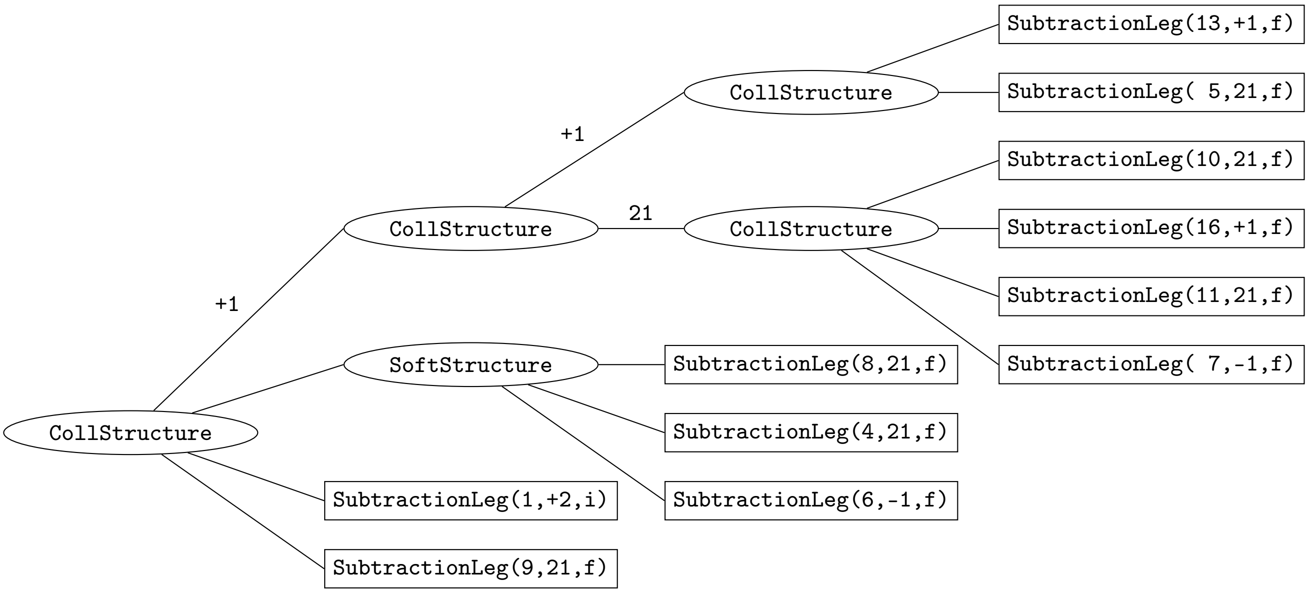

Figure 1: Example scheme of a `SingularStructure` object. The nodes in bubbles belong to the list of `substructure` of the structure they are linked to on the left, while the ones in boxes belong to its `legs`. Within `SubtractionLeg` objects, initial and final states are here abbreviated with the letters `i` and `f` respectively.

References