SingularPhasespace and helicity recycling in MadGraph5_aMC@NLO

Kiran Ostrolenk[a,*]

[a] *School of Physics and Astronomy, University of Manchester, Manchester, UK*

E-mail: kiran.ostrolenk@postgrad.manchester.ac.uk, kostrolenk@gmail.com

We present here two separate projects.
The first is a phase-space generator, dubbed SingularPhasespace. It is designed to help probe the singular limits of NLO simulations. It is hoped this will prove to be a useful tool for developers. The user (or SingularPhasespace) chooses an initial momentum configuration for the hard process and then new events are generated that are successively more and more singular. This is done whilst trying to keep as close to the initial event as possible. This allows for a clear understanding of how the simulation behaves in the singular limit.
The second project is our implementation of 'helicity recycling' within MadGraph_aMC@NLO. This is a restructuring of the code that allows the program to recycle parts of the matrix-element calculation when summing it over helicities. This optimisation is applicable to all LO processes. For complex processes we can see a speed up of a factor of ~2×.

Tools for High Energy Physics and Cosmology - TOOLS2020
2-6 November, 2020
Institut de Physique des 2 Infinis (IP2I), Lyon, France

[*] Speaker

Here we present the outcome of two separate projects. In section (1) a new phase-space generator is presented that will hopefully prove useful for developers of NLO event generators. It was written by myself in collaboration with Michael Seymour of the University of Manchester and Simon Plätzer of the University of Vienna. In section (2) is presented a re-structuring of the MadGraph5_aMC@NLO code that has allowed us to speed-up event generation. This re-structuring was performed by myself in collaboration with Olivier Mattelaer of the Université catholique de Louvain.

1. SingularPhasespace

Simulations accurate to NLO are built on three key ingredients [1]:

- the matrix elements;
- the subtraction terms used to regulate the infrared divergences of the matrix elements;
- the terms used to cancel the double counting of logarithms (if the simulation is matched to a parton shower).

These terms themselves will be a sum over many terms. For example, the real matrix element alone can be built up from 1000s of diagrams (depending on how complex the simulated process is). NLO event generators, such as those in Refs. [2–4], have to handle these terms in a process independent and fully automated way and as such they become substantial, complicated algorithms.

This complexity can result in unexpected behaviour or in bugs that are difficult to diagnose. Usually such issues will be magnified in the singular limits, where the terms listed above can become arbitrarily large. As such, being able to easily view how a simulation behaves in the singular limits is a useful tool for developers. This can be an impractical task when using a standard, random phase-space generator that is attempting to mimic nature. Firstly, it can take an unreasonable amount of computing resources to wait for such a generator to happen to populate the singular regions of phase space. Secondly, because the generation is random, it is not easy to target a specific kinematic configuration within a broader singular limit.

SingularPhasespace was developed as a response to these limitations. Rather than populating phase space randomly, it is populated in a controlled and predictable way. First the user chooses which hard process they wish to simulate and an initial momentum configuration for said process. Alternatively, SingularPhasespace can choose a random configuration. The user must also choose which limit they wish to probe (soft or collinear) and which particle they wish to push into this limit (henceforth refered to as the 'target' particle).

Next, SingularPhasespace will generate a set of events where the target particle is gradually made softer and softer or more and more collinear to another particle in the event (according to the users choice). This is done whilst trying to keep the other momenta in the events as fixed as possible. This means that any variation in the matrix elements or subtraction terms will be purely down to the target particle approaching the singular limit. In practice it is not possible to keep the other momenta *entirely* fixed for two reasons. Firstly, momentum conservation between initial and final states must be respected. Secondly, all external particles must be kept on-shell. As such, SingularPhasespace

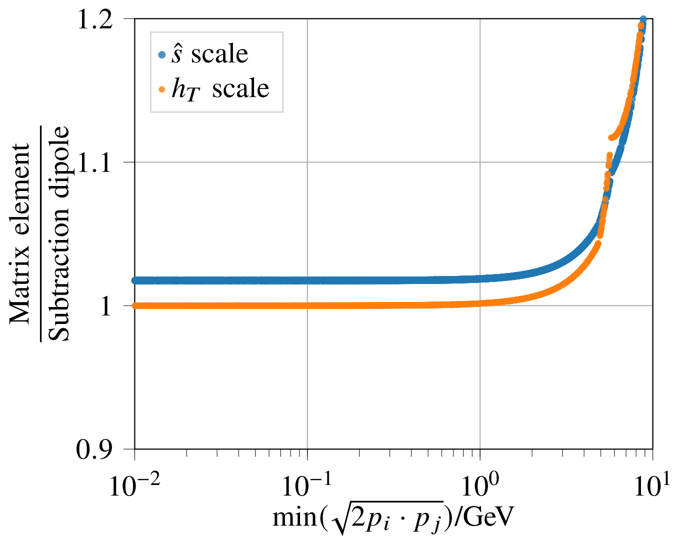

Figure 1: Each data point on this plot corresponds to a $gg \to ggg$ event generated with SingularPhasespace. The x-axis is the minimum value of $\sqrt{2p_i \cdot p_j}$ where p_i is the momentum of the forward initial state particle and p_j ranges over final state parton momenta. Going to lower values on the x-axis corresponds to p_i and p_j going collinear and as such we are approaching a singular limit. The y-axis is the ratio between the matrix element and the subtraction dipole. If the simulation is functioning correctly this ratio should tend to 1 in the singular limit. One can see this is true when using the h_T scale but not the $\hat{s}$ scale.

will implement some slight re-shuffling of the other momenta but in practice we find this does not significantly disrupt the above picture. In particular, SingularPhasespace makes sure that all events have the same total momentum as was in the initial momentum configuration.

SingularPhasespace will be integrated into an upcoming version of Herwig. However, the events it generates can be exported as a HepMC file and so used within other event generators [5].

1.1 Toy example

In this section we will use a toy example to show how SingularPhasespace can be helpful. We will do this by inserting by hand a problem into our simulation: we will use $\hat{s}$ as our scale choice, where $\hat{s}$ is defined as the total momentum of the hard process. The cancellation of infrared divergences is only guaranteed for observables that are insensitive to soft and collinear radiation. This is known as infrared-collinear (IRC) safety [6]. Notice that $\hat{s}$ *is* sensitive to initial state collinear radiation and so is not IRC safe. As such, with $\hat{s}$ as our scale choice our simulation will no longer be regularised in the collinear limit. Accordingly, the events of this simulation will have arbitrarily large weights and give the incorrect cross section.

Now imagine that we were unaware that the scale choice was responsible for these problems. As such, we will try and use SingularPhasespace to help us diagnose the issue. In figure (1) we have shown the ratio of the matrix element (ME) and the subtraction dipole (SD) for a SingularPhasespace simulation of $gg \to ggg$ using the $\hat{s}$ scale. Moving to lower values on the x-axis corresponds to moving closer to the collinear limit. As such we would expect the ratio between ME and SD to tend to 1 but instead one can see it tends to some constant number just off one. The fact that this offset is constant is an unusual and interesting feature of the plot. SingularPhasespace is encouraging us to look for some other feature of the simulation that is also held constant.

One may remember from the earlier discussion that throughout all these events the total momentum is held constant, meaning that $\hat{s}$ is kept constant. Hence, the plot in figure (1) is suggesting to us that the $\hat{s}$ scale choice is the source of the issue! Notice that if we had not used SingularPhasespace but had instead used a random phase-space generator the data points would no longer take the form of a clean line. Instead the data points would be randomly smeared out and as such we would not see the issue's constant nature. Hence, SingularPhasespace has proved useful in locating the scale choice as the source of the issue in the simulation.

In figure (1) we have also shown the same plot but now with the h_T scale[1], which *is* IRC safe. With this choice the ME to SD ratio tends to one, as expected.

2. Helicity recycling

MadGraph5_aMC@NLO (MG5aMC for short) is a program designed for, amongst other things, evaluating a given matrix element across a set of generated phase-space points. This matrix element will be a sum over Feynman diagrams. A diagram is calculated by iteratively working through each vertex and evaluating the so-called 'wave functions' of the lines associated with that vertex [7, 8]. For external particles this wave function will simply be a spinor or polarisation vector (e.g. u, v, ϵ^{μ}, . . .). For internal particles this wave function will be a Lorentz or spinor contraction of the other wave functions joining the vertex plus factors introduced by the particle's propagator and the vertex itself (according to the relevant Feynman rules).

Once the program reaches the final vertex, the wave functions connected to it (which contain information from all the other wave functions) can be contracted to compute the final diagram (also referred to as an amplitude). MG5aMC employs the helicity amplitude formalism and so each wave function, and by extension each amplitude, is dependent on the helicities of the external particles [9–11]. As such the matrix element must be summed over all combinations of external particle helicities in order to get the final result. Here we present our work to greatly increase the computational speed with which this sum is evaluated.

2.1 Implementation

Prior to the work of this project, MG5aMC would evaluate the matrix element independently for each helicity combination. It is common for a given particle to appear with the same helicity in multiple helicity combinations. This means MG5aMC would recalculate the same wave function multiple times when performing the sum over helicities. This is a waste of computation. We have restructured the MG5aMC code such that it can now identify duplicated wave functions and avoid recalculating them. Instead, the wave function is calculated once, the result is stored as a float and then the float is called upon when need be.

We found the most elegant way to implement this was to make use of a directed acyclic graph (DAG). Each node in the graph is a wave function or amplitude evaluated at a given helicity combination and a connection between nodes implies dependency. As such, it becomes easy to determine

[1]This scale is defined as the sum of jet p_T

Process	Speed up
$gg \to t\bar{t}$	1.36×
$gg \to t\bar{t}g$	1.43×
$gg \to t\bar{t}gg$	2.27×
$qq \to t\bar{t}qq$	1.27×

(a) $t\bar{t}$ processes.

Process	Speed up
$qq \to W^+W^-qq$	1.67×
$qq \to W^+W^-gg$	1.89×
$gg \to W^+W^-qq$	1.89×
$gq \to W^+W^-gq$	2.13×

(b) $pp \to W^+W^-jj$ processes.

Process	Speed up
$qq \to e^+e^-$	1.02×
$qq \to e^+e^-g$	1.03×
$gg \to e^+e^-qq$	1.09×

(c) e^+e^- processes.

Table 1: The speed-up of the `madevent` binary thanks to helicity recycling. This speed-up is represented as the ratio between the time taken without and with helicity recycling. Here q represents a sum over the light quarks.

which wave functions contribute to which amplitudes and so avoid duplication. Furthermore, some amplitudes will be zero for certain helicity combinations. This DAG allows us to also easily determine which wave functions would contribute to these null-amplitudes and so avoid calculating them. This adds further optimisation.

Finally, we were able to not only optimise wave functions calculations but also the calculation of amplitudes. In general, part of the amplitude calculation will involve taking the Lorentz or spinor product between two wave functions. If these two wave functions are repeated across multiple helicity combinations then this mean we will evaluate the same product multiple times. As such we were also able to recycle this part of the amplitude calculation. This optimisation is especially important for processes with many diagrams such as those involving multiple gluons.

2.2 Results

The sum of all the optimisations detailed in section (2.1) is what we refer to as helicity recycling. In this section we will show the resultant speed up of the code. We shall focus on the speed of the `madevent` binary, responsible for generating phase-space points, evaluating the matrix element and some other things such as evaluating the PDFs. Remember, helicity recycling only optimises the matrix-element evaluation.

In table (1) we show the factor speed up of `madevent` thanks to helicity recycling. This speed up is represented as the ratio between the time taken without and with helicity recycling. One can see that the impact is highly processes dependent. This is because the more wave functions and amplitudes there are to evaluate, the more the program can recycle. As such processes with many and more complex diagrams see a higher speed up. Take for example the $t\bar{t}$ processes in table (1a). One can see how adding one and two extra gluons into the final state greatly increases the speed up, from 1.36× to 2.27×.

Notice though that the similar process $qq \to t\bar{t}qq$ sees a much more modest speed up of 1.27×. This is because this process is dominated by fermions. For such a process, there are much fewer helicity combinations that contribute a non-zero matrix element. As such, the sum of these combinations is much shorter and so it is responsible for a smaller percentage of the overall computation.

We see a similar principle at play when looking at table (1b) and (1c). W bosons are not fermions and produce complex diagrams, hence the processes involving them see a large speed up. Electrons on the other hand are fermions and so processes involving them see an almost insignificant speed up.

Notice that helicity recycling's tendency to have a bigger impact on the more complex processes is highly desirable. These processes are the ones that take the longest and so it is most efficient to target them.

3. Conclusion

SingularPhasespace is a phase-space generator that allows users to see exactly how an NLO algorithm behaves in the singular limits. These limits are where features of the code become most exaggerated and so SingularPhasespace can help with diagnosing bugs and understanding unexpected behaviour. This generator will be available in an upcoming release of Herwig.

We have also implemented a restructuring of MG5aMC that allows it to recycle parts of the matrix-element calculation across the sum over helicity combinations. We have dubbed this restructuring helicity recycling. The resultant speed up is highly processes dependent. For complex processes one can see a speed up of $\sim 2\times$ whereas for simpler processes involving fermions the factor is more modest. This tendency is desirable as the most complex processes are also the most computationally intensive. A possible extension of this project would be to extend it from LO to NLO processes. There the real matrix element dominates the computational cost and so there is scope for a large improvement.

References

[1] A. Buckley *et al.*, "General-purpose event generators for LHC physics," *Phys. Rept.*, vol. 504, pp. 145–233, 2011.

[2] J. Alwall, R. Frederix, S. Frixione, V. Hirschi, F. Maltoni, O. Mattelaer, H. S. Shao, T. Stelzer, P. Torrielli, and M. Zaro, "The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations," *JHEP*, vol. 07, p. 079, 2014.

[3] P. Nason, "A New method for combining NLO QCD with shower Monte Carlo algorithms," *JHEP*, vol. 11, p. 040, 2004.

[4] J. Bellm *et al.*, "Herwig 7.2 release note," *Eur. Phys. J. C*, vol. 80, no. 5, p. 452, 2020.

[5] M. Dobbs and J. B. Hansen, "The HepMC C++ Monte Carlo event record for High Energy Physics," *Comput. Phys. Commun.*, vol. 134, pp. 41–46, 2001.

[6] G. P. Salam, "Towards Jetography," *Eur. Phys. J. C*, vol. 67, pp. 637–686, 2010.

[7] H. Murayama, I. Watanabe, and K. Hagiwara, "HELAS: HELicity amplitude subroutines for Feynman diagram evaluations," 1 1992.

[8] P. de Aquino, W. Link, F. Maltoni, O. Mattelaer, and T. Stelzer, "ALOHA: Automatic Libraries Of Helicity Amplitudes for Feynman Diagram Computations," *Comput. Phys. Commun.*, vol. 183, pp. 2254–2263, 2012.

[9] P. De Causmaecker, R. Gastmans, W. Troost, and T. T. Wu, "Helicity Amplitudes for Massless QED," *Phys. Lett. B*, vol. 105, p. 215, 1981.

[10] P. De Causmaecker, R. Gastmans, W. Troost, and T. T. Wu, "Multiple Bremsstrahlung in Gauge Theories at High-Energies. 1. General Formalism for Quantum Electrodynamics," *Nucl. Phys. B*, vol. 206, pp. 53–60, 1982.

[11] R. Gastmans, "THE HELICITY METHOD: A REVIEW," *AIP Conf. Proc.*, vol. 201, pp. 58–72, 1990.