# Jet grooming through reinforcement learning

3rd IML Machine Learning Workshop, CERN, 17 April 2019

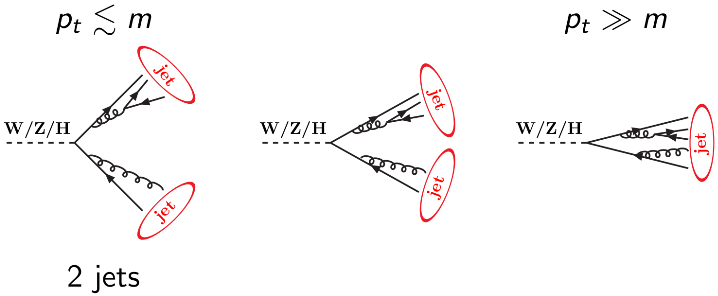
Frédéric Dreyer

UNIVERSITY OF OXFORD

based on arXiv:1903.09644 with Stefano Carrazza

# Boosted objects at the LHC

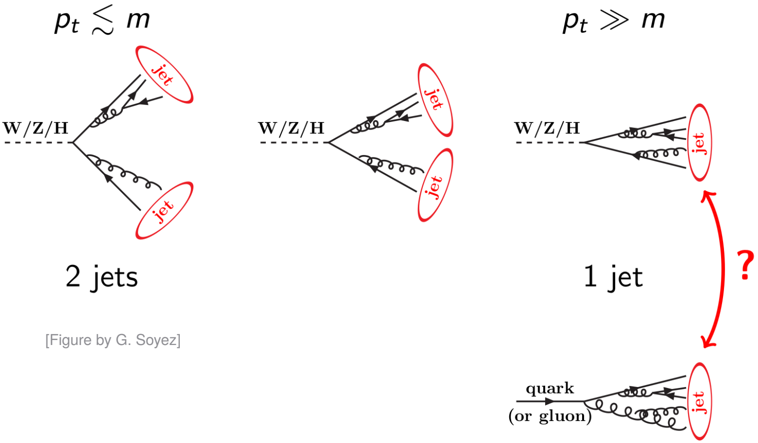
- At LHC energies, EW-scale particles (W/Z/t...) are often produced with $p_t \gg m$, leading to collimated decays.
- Hadronic decay products are thus often reconstructed into single jets.




[Figure by G. Soyez]

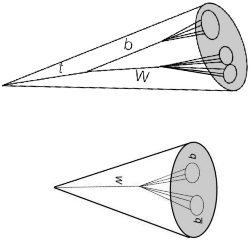
# Boosted objects at the LHC

- At LHC energies, EW-scale particles (W/Z/t…) are often produced with $p_t \gg m$, leading to collimated decays.
- Hadronic decay products are thus often reconstructed into single jets.



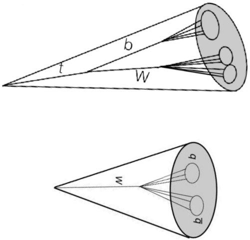

[Figure by G. Soyez]

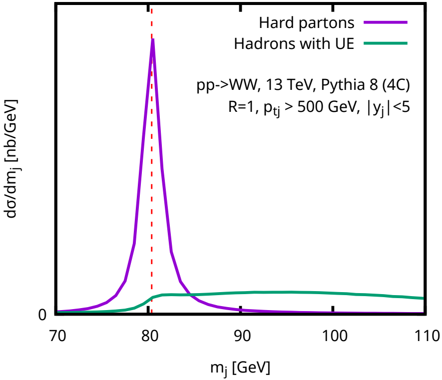
- Many techniques developed to identify hard structure of a jet based on radiation patterns.
- In principle, simplest way to identify these boosted objects is by looking at the mass of the jet.



# Boosted objects at the LHC

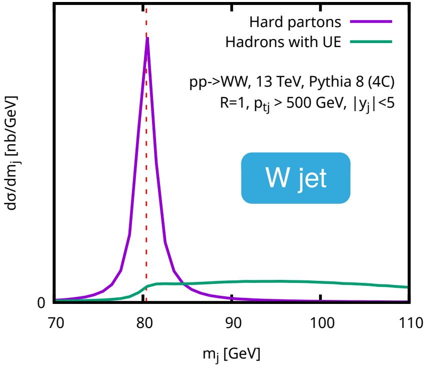
- Many techniques developed to identify hard structure of a jet based on radiation patterns.
- In principle, simplest way to identify these boosted objects is by looking at the mass of the jet.
- But jet mass distribution is highly distorted by QCD radiation and pileup.



- Mass peak can be partly reconstructed by removing unassociated soft wide-angle radiation (grooming).
- Recurse through clustering tree and remove soft branch if

$$\frac{\min(p_{t,1}, p_{t,2})}{p_{t,1} + p_{t,2}} > z_{\text{cut}} \left( \frac{\Delta R_{12}}{R_0} \right)^{\beta}$$

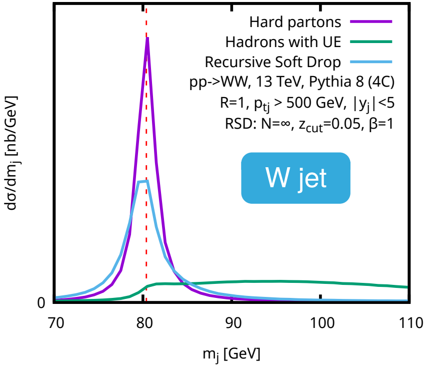



[Dasgupta, Fregoso, Marzani, Salam JHEP 1309 (2013) 029]
[Larkoski, Marzani, Soyez, Thaler JHEP 1405 (2014) 146]
[FD, Necib, Soyez, Thaler JHEP 1806 (2018) 093]

# Jet grooming: (Recursive) Soft Drop / mMDT

- Mass peak can be partly reconstructed by removing unassociated soft wide-angle radiation (grooming).
- Recurse through clustering tree and remove soft branch if

$$\frac{\min(p_{t,1}, p_{t,2})}{p_{t,1} + p_{t,2}} > z_{\mathrm{cut}} \left( \frac{\Delta R_{12}}{R_0} \right)^{\beta}$$

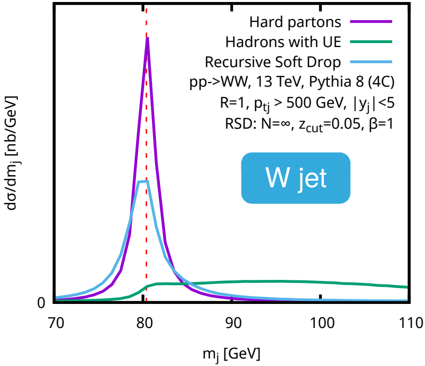



[Dasgupta, Fregoso, Marzani, Salam JHEP 1309 (2013) 029]
[Larkoski, Marzani, Soyez, Thaler JHEP 1405 (2014) 146]
[FD, Necib, Soyez, Thaler JHEP 1806 (2018) 093]

# Jet grooming: (Recursive) Soft Drop / mMDT

- Mass peak can be partly reconstructed by removing unassociated soft wide-angle radiation (grooming).
- Recurse through clustering tree and remove soft branch if

$$\frac{\min(p_{t,1}, p_{t,2})}{p_{t,1} + p_{t,2}} > z_{\text{cut}} \left(\frac{\Delta R_{12}}{R_0}\right)^{\beta}$$




[Dasgupta, Fregoso, Marzani, Salam JHEP 1309 (2013) 029]
[Larkoski, Marzani, Soyez, Thaler JHEP 1405 (2014) 146]
[FD, Necib, Soyez, Thaler JHEP 1806 (2018) 093]

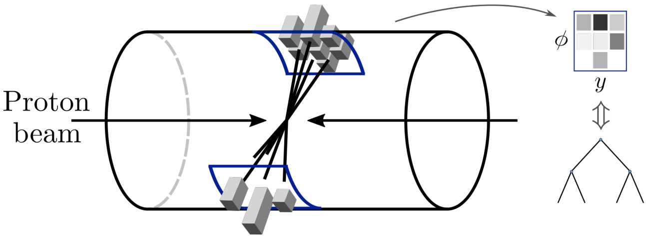
**Aim of this talk:** Introduce a framework based on reinforcement learning to tackle the problem of jet grooming more systematically.

# What's a jet, anyway?

A jet can be described in a variety of ways

- Sum of its particle constituents
- Image of energy deposits in rapidity-azimuth
- Complete basis of observables (e.g. energy flow polynomials)
- Image of primary Lund declustering sequence
- Binary tree associated with the jet clustering sequence
- …

Most of them involve some trade-off between interpretability, representability, robustness and performance.



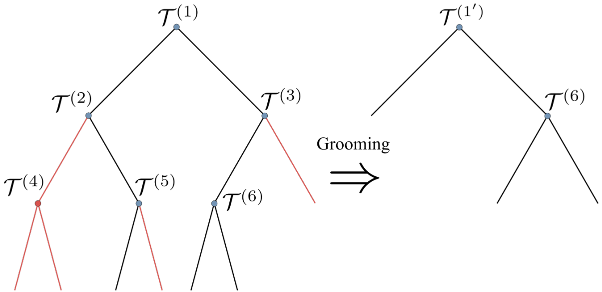
# Grooming a jet tree

- Cast jet as clustering tree where state of each node $\mathcal{T}^{(i)}$ is a tuple with kinematic information on splitting

$$s_t = \{z, \Delta_{ab}, \psi, m, k_t\}$$

- Grooming algorithm defined as a function $\pi_g$ observing a state and returning an action $\{0, 1\}$ on the removal of the softer branch, e.g.

$$\pi_{\text{RSD}}(s_t) = \begin{cases} 0 & \text{if} \quad z > z_{\text{cut}}\left(\frac{\Delta_{ab}}{R_0}\right)^{\beta} \\ 1 & \text{else} \end{cases}$$

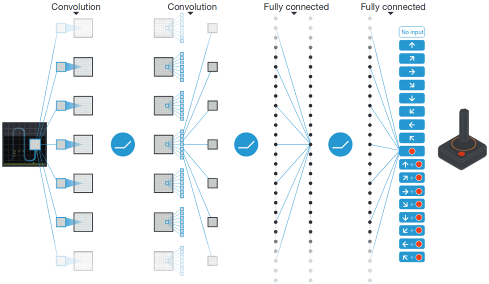


# Reinforcement learning with Deep-Q-Networks

Reinforcement learning are usually built from two elements:

- an agent deciding which actions to take in order to maximize reward
- an environment, observed by the agent and affected by the action

Deep Q-Network is a RL algorithm which uses a table of $Q$-values $Q(s,a)$, determining the next action as the one that maximizes $Q$.




A neural network is used to approximate the optimal action-value function

$$Q^*(s,a) = \max_{\pi} \mathbb{E}[r_t + \gamma r_{t+1} + \ldots | s_t = s, a_t = a, \pi]$$

[Mnih et al, Nature 2015]

# Defining a grooming environment

To find optimal grooming policy $\pi_g$, define an environment and a reward function so that problem can be solved with RL.

- Initialize list of all trees used for training.
- Each episode starts by randomly selecting a tree and adding its root to a priority queue (ordered in $\Delta_{ab}$).
- Each step removes first node from priority queue, then takes action on removal of soft branch based on state $s_t$ of node.
- After action, update kinematics of parent nodes, add current children to priority queue, and evaluate reward function.
- Episode terminates once priority queue is empty.

# Defining the reward function

- Key ingredient for optimization of grooming policy is reward function used at each training step.
- We construct a reward with two components
  - First piece $R_M$ evaluated on the full jet tree, comparing the jet mass to a target value.
  - Second component $R_{\mathrm{SD}}$ looks at kinematics of current node.
- Total reward is then given by

$$R(m, a_t, \Delta, z) = R_M(m) + \frac{1}{N_{\mathrm{SD}}} R_{\mathrm{SD}}(a_t, \Delta, z)$$

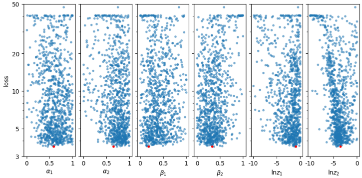
- where mass reward is defined using a Cauchy distribution

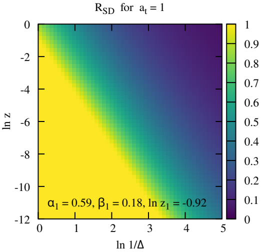
$$R_M(m) = \frac{\Gamma^2}{\pi(|m - m_{\mathrm{target}}|^2 + \Gamma^2)}$$

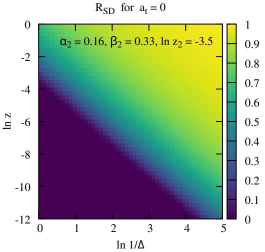
# Defining the reward function

- To provide baseline behaviour for the groomer, we include a "Soft-Drop" reward $R_{\text{SD}}$ evaluated on the current node
- Calculated on the current node state, gives positive reward for removal of wide-angle soft radiation and for keeping hard-collinear emissions.

$$R_{\text{SD}}(a_t, \Delta, z) = a_t \min\left(1, e^{-\alpha_1 \ln(1/\Delta) + \beta_1 \ln(z_1/z)}\right) \\ + (1 - a_t) \max\left(0, 1 - e^{-\alpha_2 \ln(1/\Delta) + \beta_2 \ln(z_2/z)}\right)$$



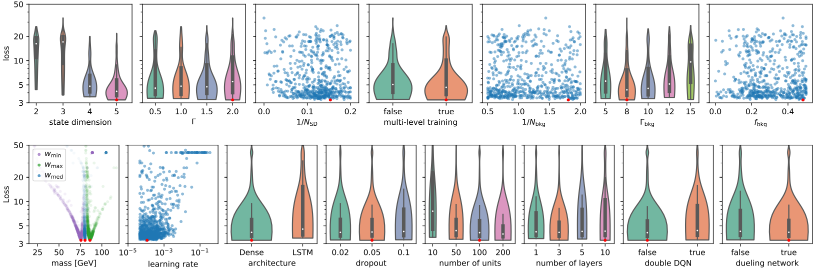



# Implementation and multi-level training

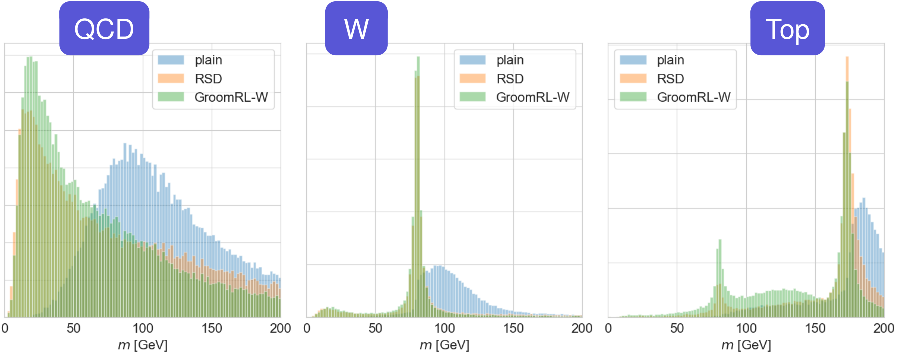
- Train RL agent with multi-level approach using both signal and bkg into account. Sample consists of 500k $W$/QCD or Top/QCD `Pythia` 8 jets.
- At the beginning of each episode, randomly select a signal or background jet with probability $1 - p_{\text{bkg}}$.
- In the background case, mass reward function is changed to

$$R_M^{\text{bkg}}(m) = \frac{m}{\Gamma_{\text{bkg}}} \exp\left(-\frac{m}{\Gamma_{\text{bkg}}}\right).$$



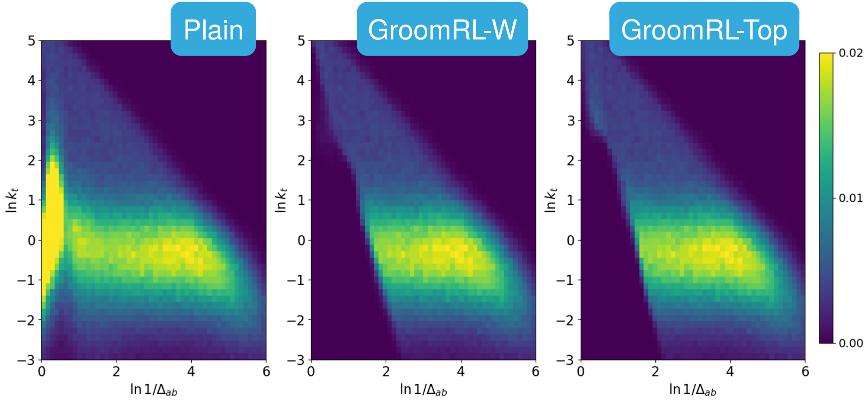
# Groomed jet mass spectrum

- To test the grooming algorithm derived from the DQN agent, we apply our groomer to three test samples: QCD, $W$ and Top jets.
- Improvement in jet mass resolution compared to RSD.
- Algorithm performs well on data beyond its training range.



# Lund jet plane density

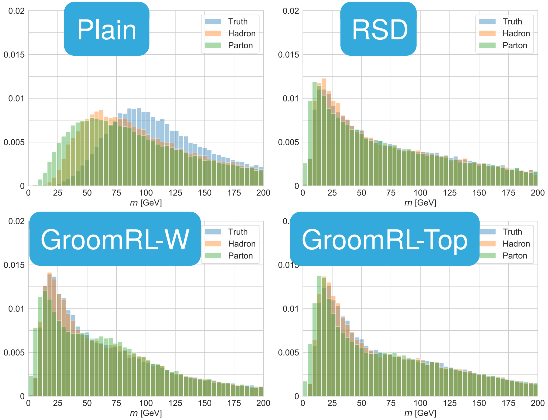
- Visualize radiation patterns after grooming with Lund jet plane, defined as $(\ln 1/\Delta_{ab}, \ln k_t)$ of "primary" declustering sequence.
- Soft and wide-angle radiation is removed by the groomer.
- Training grooming strategy on top data results in differences at wide angles, despite similar performance.




[FD, Salam, Soyez, JHEP 1812 (2018) 064]

# Robustness to non-perturbative effects

- Resilience to hadronisation and underlying event corrections is a key feature of modern grooming algorithms
- Strategy derived from reinforcement learning shows similar behaviour to heuristic method
- No parton or hadron-level data was used in the training!



# Conclusions

- Promising application of RL to the issue of jet grooming.
- Carefully designed reward function can be used to construct a groomer from NN trained with DQN agent.
- Groomer can be applied with good performance to wide range of data.
- Due to its simplicity, the algorithm should retain some of the calculability of existings methods such as Soft Drop.
- Framework is generic and can easily be extended to higher-dimensional inputs or to other problems.

The **GroomRL** code is available online:
*github.com/JetsGame/GroomRL*