

# MCnet Summer School 2025

Basics of parton showers

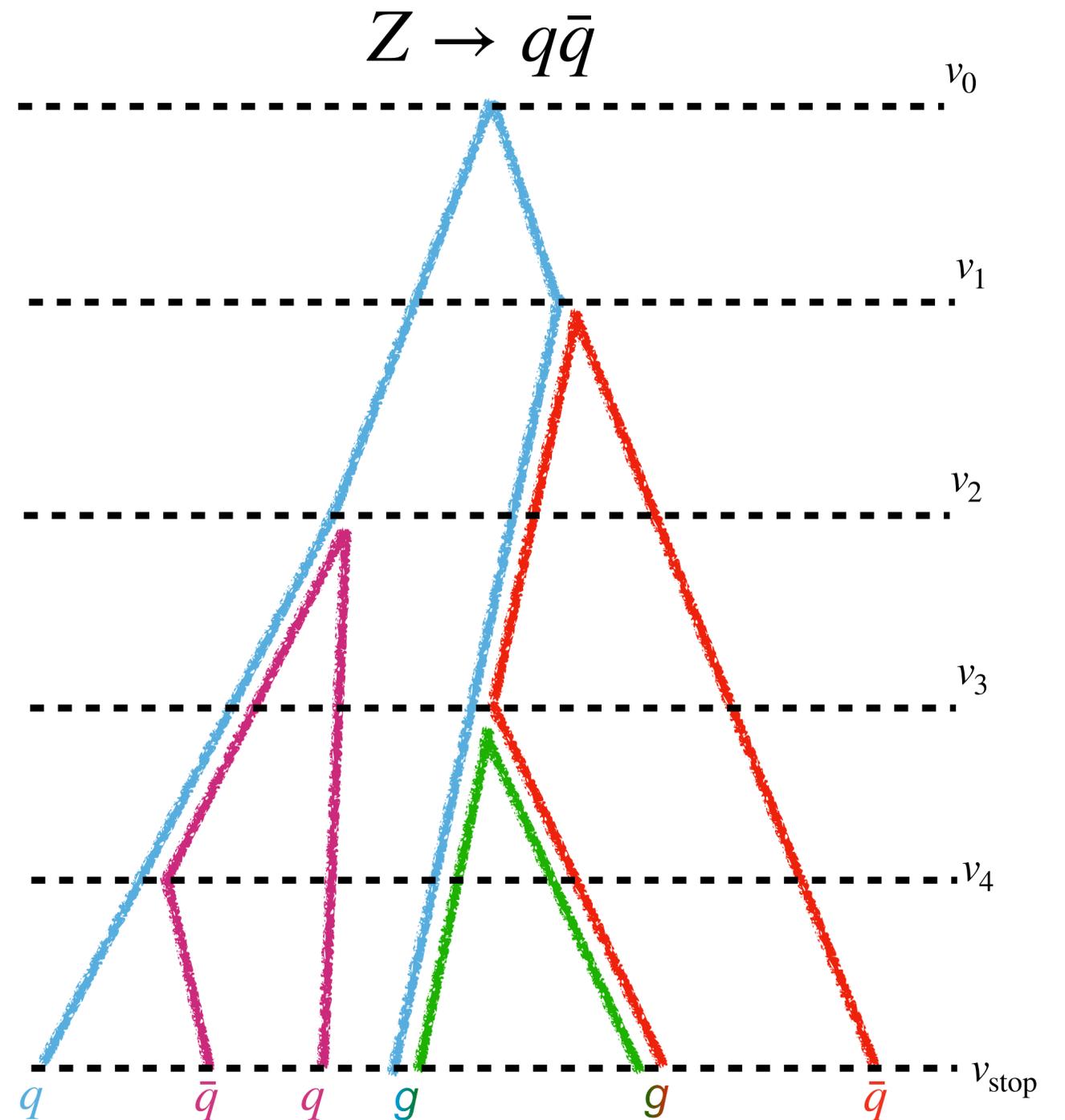
Melissa van Beekveld

Nikhef

# Recap

To build a shower, there are three key concepts to understand:

1. What is the splitting probability?
2. How are emissions ordered?
3. How do we create  $n$  momenta?
  - ▶ We've derived a splitting probability
  - ▶ We've seen that some observables develop large logarithmic corrections at the first order, necessitating a higher-order correction



# The need for all-orders

What happens to the value of the observable?

At NLO the integrated cross section for events with  $\tau < \tau_c$  is

$$\begin{aligned}\Sigma(\tau_c) &= 1 + \int [dk] |M(k)|^2 \Theta\left(\frac{k_t}{Q} e^{-|\eta|} - \tau_c\right) \\ &= 1 - \frac{\alpha_s}{\pi} 2C_l \frac{\ln^2(\tau_c)}{2}\end{aligned}$$

For small  $\tau_c$  the correction becomes large

→ spoils convergence of perturbative series

We need to calculate more terms!

# All hell breaks loose?

**We need to perform an all-order calculation**

- We clearly cannot afford to compute all orders exactly
- Rely instead on QCD factorisation properties!

The all-order structure can be obtained by iterating those factorised building blocks

$$\mathcal{O}(\alpha_s) \quad \alpha_s L^2 \quad \alpha_s L \quad \alpha_s \quad (+\mathcal{O}(\alpha_s e^{-L}) + \dots)$$

relative order  $1/L$

# All hell breaks loose?

**We need to perform an all-order calculation**

- We clearly cannot afford to compute all orders exactly
- Rely instead on QCD factorisation properties!

The all-order structure can be obtained by iterating those factorised building blocks

$\mathcal{O}(\alpha_s)$	$\alpha_s L^2$	$\alpha_s L$	$\alpha_s$		
<hr/>					
$\mathcal{O}(\alpha_s^2)$	$\alpha_s^2 L^4$	$\alpha_s^2 L^3$	$\alpha_s^2 L^2$	$\alpha_s^2 L$	$\alpha_s^2$

# All hell breaks loose?

**We need to perform an all-order calculation**

- We clearly cannot afford to compute all orders exactly
- Rely instead on QCD factorisation properties!

The all-order structure can be obtained by iterating those factorised building blocks

$\mathcal{O}(\alpha_s)$	$\alpha_s L^2$	$\alpha_s L$	$\alpha_s$				
$\mathcal{O}(\alpha_s^2)$	$\alpha_s^2 L^4$	$\alpha_s^2 L^3$	$\alpha_s^2 L^2$	$\alpha_s^2 L$	$\alpha_s^2$		
$\mathcal{O}(\alpha_s^3)$	$\alpha_s^3 L^6$	$\alpha_s^3 L^5$	$\alpha_s^3 L^4$	$\alpha_s^3 L^3$	$\alpha_s^3 L^2$	$\alpha_s^3 L$	$\alpha_s^3$

# All hell breaks loose?

**We need to perform an all-order calculation**

- We clearly cannot afford to compute all orders exactly
- Rely instead on QCD factorisation properties!

The all-order structure can be obtained by iterating those factorised building blocks

$\mathcal{O}(\alpha_s)$	$\alpha_s L^2$	$\alpha_s L$	$\alpha_s$				
$\mathcal{O}(\alpha_s^2)$	$\alpha_s^2 L^4$	$\alpha_s^2 L^3$	$\alpha_s^2 L^2$	$\alpha_s^2 L$	$\alpha_s^2$		
$\mathcal{O}(\alpha_s^3)$	$\alpha_s^3 L^6$	$\alpha_s^3 L^5$	$\alpha_s^3 L^4$	$\alpha_s^3 L^3$	$\alpha_s^3 L^2$	$\alpha_s^3 L$	$\alpha_s^3$
$\vdots$							
$\mathcal{O}(\alpha_s^n)$	$\alpha_s^n L^{2n}$	$\alpha_s^n L^{2n-1}$	...	...	...	...	...

In the singular limit, we reorganise the series in terms of  $1/L$  instead of  $\alpha_s$

# All hell breaks loose?

**We need to perform an all-order calculation**

- We clearly cannot afford to compute all orders exactly
- Rely instead on QCD factorisation properties!

The all-order structure can be obtained by iterating those factorised building blocks

$\mathcal{O}(\alpha_s)$	$\alpha_s L^2$	$\alpha_s L$	$\alpha_s$				
$\mathcal{O}(\alpha_s^2)$	$\alpha_s^2 L^4$	$\alpha_s^2 L^3$	$\alpha_s^2 L^2$	$\alpha_s^2 L$	$\alpha_s^2$		
$\mathcal{O}(\alpha_s^3)$	$\alpha_s^3 L^6$	$\alpha_s^3 L^5$	$\alpha_s^3 L^4$	$\alpha_s^3 L^3$	$\alpha_s^3 L^2$	$\alpha_s^3 L$	$\alpha_s^3$
$\vdots$							
$\mathcal{O}(\alpha_s^n)$	$\alpha_s^n L^{2n}$	$\alpha_s^n L^{2n-1}$	...	...	...	...	...

LL

$$\Sigma(\tau_c) = \sigma_0 e^{-Lg_1(\alpha_s L)}$$

# All hell breaks loose?

**We need to perform an all-order calculation**

- We clearly cannot afford to compute all orders exactly
- Rely instead on QCD factorisation properties!

The all-order structure can be obtained by iterating those factorised building blocks

$\mathcal{O}(\alpha_s)$	$\alpha_s L^2$	$\alpha_s L$	$\alpha_s$				
$\mathcal{O}(\alpha_s^2)$	$\alpha_s^2 L^4$	$\alpha_s^2 L^3$	$\alpha_s^2 L^2$	$\alpha_s^2 L$	$\alpha_s^2$		
$\mathcal{O}(\alpha_s^3)$	$\alpha_s^3 L^6$	$\alpha_s^3 L^5$	$\alpha_s^3 L^4$	$\alpha_s^3 L^3$	$\alpha_s^3 L^2$	$\alpha_s^3 L$	$\alpha_s^3$
$\vdots$							
$\mathcal{O}(\alpha_s^n)$	$\alpha_s^n L^{2n}$	$\alpha_s^n L^{2n-1}$	...	...	...	...	...

**NLL**

$$\Sigma(\tau_c) = \sigma_0 e^{-Lg_1(\alpha_s L) + g_2(\alpha_s L)}$$

# All hell breaks loose?

**We need to perform an all-order calculation**

- We clearly cannot afford to compute all orders exactly
- Rely instead on QCD factorisation properties!

The all-order structure can be obtained by iterating those factorised building blocks

$\mathcal{O}(\alpha_s)$	$\alpha_s L^2$	$\alpha_s L$	$\alpha_s$			
$\mathcal{O}(\alpha_s^2)$	$\alpha_s^2 L^4$	$\alpha_s^2 L^3$	$\alpha_s^2 L^2$	$\alpha_s^2 L$	$\alpha_s^2$	
$\mathcal{O}(\alpha_s^3)$	$\alpha_s^3 L^6$	$\alpha_s^3 L^5$	$\alpha_s^3 L^4$	$\alpha_s^3 L^3$	$\alpha_s^3 L^2$	$\alpha_s^3 L$ $\alpha_s^3$
$\vdots$						
$\mathcal{O}(\alpha_s^n)$	$\alpha_s^n L^{2n}$	$\alpha_s^n L^{2n-1}$	$\dots$	$\dots$	$\dots$	$\dots$

**NNLL**

$$\Sigma(\tau_c) = \sigma_0 e^{-Lg_1(\alpha_s L) + g_2(\alpha_s L) + \alpha_s g_3(\alpha_s L)}$$

# All hell breaks loose?

**We need to perform an all-order calculation**

- We clearly cannot afford to compute all orders exactly
- Rely instead on QCD factorisation properties!

The all-order structure can be obtained by iterating those factorised building blocks

$\mathcal{O}(\alpha_s)$	$\alpha_s L^2$	$\alpha_s L$	$\alpha_s$				
$\mathcal{O}(\alpha_s^2)$	$\alpha_s^2 L^4$	$\alpha_s^2 L^3$	$\alpha_s^2 L^2$	$\alpha_s^2 L$	$\alpha_s^2$		
$\mathcal{O}(\alpha_s^3)$	$\alpha_s^3 L^6$	$\alpha_s^3 L^5$	$\alpha_s^3 L^4$	$\alpha_s^3 L^3$	$\alpha_s^3 L^2$	$\alpha_s^3 L$	$\alpha_s^3$
$\vdots$							
$\mathcal{O}(\alpha_s^n)$	$\alpha_s^n L^{2n}$	$\alpha_s^n L^{2n-1}$	...	...	...	...	...

**It is the shower's job to resum such large logarithms!**

$\alpha_s^m L^n$  terms originate from integrating over the soft and collinear MEs

→ getting these right in the shower is crucial

# Anatomy of an all-order resummation

Let's take a simple setup

1. ME corrections of multiple emissions can be described in terms of one-emission factors

$$|M(k_1, \dots, k_n)|^2 = |M(k_1)|^2 \dots |M(k_n)|^2$$

if the emissions are strongly ordered, this is true

2. Virtual corrections are given by unitarity and exponentiate

$$V(\Phi_B) = e^{-\int [dk] M^2(k)}$$

true for eikonal corrections

# Anatomy of an all-order resummation

Let's take a simple setup

1. ME corrections of multiple emissions can be described in terms of one-emission factors

$$|M(k_1, \dots, k_n)|^2 = |M(k_1)|^2 \dots |M(k_n)|^2$$

if the emissions are strongly ordered, this is true

2. Virtual corrections are given by unitarity and exponentiate

$$V(\Phi_B) = e^{-\int [dk] M^2(k)}$$

true for eikonal corrections

$$\Sigma(\nu) = V(\Phi_B) \sum_{n=0} \frac{1}{n!} \left[ \prod_{i=1}^n \int [dk_i] |M(k_i)|^2 \right] \Theta(\nu - V(k_1, \dots, k_n))$$

# Anatomy of an all-order resummation

Let's take a simple setup

1. ME corrections of multiple emissions can be described in terms of one-emission factors

$$|M(k_1, \dots, k_n)|^2 = |M(k_1)|^2 \dots |M(k_n)|^2$$

2. Virtual corrections are given by unitarity and exponentiate

$$V(\Phi_B) = e^{-\int [dk] M^2(k)}$$

$$\Sigma(\nu) = V(\Phi_B) \sum_{n=0} \frac{1}{n!} \left[ \prod_{i=1}^n \int [dk_i] |M(k_i)|^2 \right] \Theta(\nu - V(k_1, \dots, k_n))$$

one-emission soft-collinear ME and phase space

# Anatomy of an all-order resummation

Let's take a simple setup

1. ME corrections of multiple emissions can be described in terms of one-emission factors

$$|M(k_1, \dots, k_n)|^2 = |M(k_1)|^2 \dots |M(k_n)|^2$$

2. Virtual corrections are given by unitarity and exponentiate

$$V(\Phi_B) = e^{-\int [dk] M^2(k)}$$

$$\Sigma(\nu) = V(\Phi_B) \sum_{n=0} \frac{1}{n!} \left[ \prod_{i=1}^n \int [dk_i] |M(k_i)|^2 \right] \Theta(\nu - V(k_1, \dots, k_n))$$

sum of arbitrary  
number of emissions

# Anatomy of an all-order resummation

Let's take a simple setup

1. ME corrections of multiple emissions can be described in terms of one-emission factors

$$|M(k_1, \dots, k_n)|^2 = |M(k_1)|^2 \dots |M(k_n)|^2$$

2. Virtual corrections are given by unitarity and exponentiate

$$V(\Phi_B) = e^{-\int [dk] M^2(k)}$$

$$\Sigma(\nu) = V(\Phi_B) \sum_{n=0} \frac{1}{n!} \left[ \prod_{i=1}^n \int [dk_i] |M(k_i)|^2 \right] \Theta(\nu - V(k_1, \dots, k_n))$$

value of the observable  
in the presence of  $n$   
emissions

# Anatomy of an all-order resummation

Let's take a simple setup

1. ME corrections of multiple emissions can be described in terms of one-emission factors

$$|M(k_1, \dots, k_n)|^2 = |M(k_1)|^2 \dots |M(k_n)|^2$$

2. Virtual corrections are given by unitarity and exponentiate

$$V(\Phi_B) = e^{-\int [dk] M^2(k)}$$

$$\Sigma(\nu) = V(\Phi_B) \sum_{n=0} \frac{1}{n!} \left[ \prod_{i=1}^n \int [dk_i] |M(k_i)|^2 \right] \Theta(\nu - V(k_1, \dots, k_n))$$

Both  $V$  and the integral over real emissions are divergent (as before)

→ introduce a slicing parameter  $\epsilon \nu$

# Anatomy of an all-order resummation

$$\Sigma(v) = V(\Phi_B) \sum_{n=0} \frac{1}{n!} \left[ \prod_{i=1}^n \int [dk_i] |M(k_i)|^2 \right] \Theta(v - V(k_1, \dots, k_n))$$

→ introduce a slicing parameter  $\epsilon v$

$$V(k_1, \dots, k_{i-1}, k_i, k_{i+1}, \dots, k_n) = V(k_1, \dots, k_{i-1}, k_{i+1}, \dots, k_n) + \epsilon^p v \quad \text{if } V(k_i) < \epsilon v$$

# Anatomy of an all-order resummation

$$\Sigma(v) = V(\Phi_B) \sum_{n=0} \frac{1}{n!} \left[ \prod_{i=1}^n \int [dk_i] |M(k_i)|^2 \right] \Theta(v - V(k_1, \dots, k_n))$$

→ introduce a slicing parameter  $\epsilon v$

$$V(k_1, \dots, k_{i-1}, k_i, k_{i+1}, \dots, k_n) = V(k_1, \dots, k_{i-1}, k_{i+1}, \dots, k_n) + \epsilon^p v \quad \text{if } V(k_i) < \epsilon v$$

$$\Sigma(v) = e^{-\int [dk] |M(k)|^2 \Theta(V(k) - \epsilon v)}$$

$$\times \sum_{n=0} \frac{1}{n!} \left[ \prod_{i=1}^n \int [dk_i] |M(k_i)|^2 \Theta(V(k_i) - \epsilon v) \right] \Theta(v - V(k_1, \dots, k_n))$$

# Anatomy of an all-order resummation

$$\Sigma(v) = V(\Phi_B) \sum_{n=0} \frac{1}{n!} \left[ \prod_{i=1}^n \int [dk_i] |M(k_i)|^2 \right] \Theta(v - V(k_1, \dots, k_n))$$

→ introduce a slicing parameter  $\epsilon v$

$$V(k_1, \dots, k_{i-1}, k_i, k_{i+1}, \dots, k_n) = V(k_1, \dots, k_{i-1}, k_{i+1}, \dots, k_n) + \epsilon^p v \quad \text{if } V(k_i) < \epsilon v$$

$$\Sigma(v) = e^{-\int [dk] |M(k)|^2 \Theta(V(k) - \epsilon v)} \text{ unresolved emissions and virtual corrections}$$

$$\times \sum_{n=0} \frac{1}{n!} \left[ \prod_{i=1}^n \int [dk_i] |M(k_i)|^2 \Theta(V(k_i) - \epsilon v) \right] \Theta(v - V(k_1, \dots, k_n))$$

# Anatomy of an all-order resummation

$$\Sigma(\nu) = V(\Phi_B) \sum_{n=0} \frac{1}{n!} \left[ \prod_{i=1}^n \int [dk_i] |M(k_i)|^2 \right] \Theta(\nu - V(k_1, \dots, k_n))$$

→ introduce a slicing parameter  $\epsilon\nu$

$$V(k_1, \dots, k_{i-1}, k_i, k_{i+1}, \dots, k_n) = V(k_1, \dots, k_{i-1}, k_{i+1}, \dots, k_n) + \epsilon^p \nu \quad \text{if } V(k_i) < \epsilon\nu$$

$$\Sigma(\nu) = e^{-\int [dk] |M(k)|^2 \Theta(V(k) - \epsilon\nu)}$$

$$\times \sum_{n=0} \frac{1}{n!} \left[ \prod_{i=1}^n \int [dk_i] |M(k_i)|^2 \Theta(V(k_i) - \epsilon\nu) \right] \Theta(\nu - V(k_1, \dots, k_n))$$

resolved emissions

# Anatomy of an all-order resummation

$$\Sigma(v) = V(\Phi_B) \sum_{n=0} \frac{1}{n!} \left[ \prod_{i=1}^n \int [dk_i] |M(k_i)|^2 \right] \Theta(v - V(k_1, \dots, k_n))$$

→ introduce a slicing parameter  $\epsilon v$

$$V(k_1, \dots, k_{i-1}, k_i, k_{i+1}, \dots, k_n) = V(k_1, \dots, k_{i-1}, k_{i+1}, \dots, k_n) + \epsilon^p v \quad \text{if } V(k_i) < \epsilon v$$

$$\Sigma(v) = e^{-\int [dk] |M(k)|^2 \Theta(V(k) - \epsilon v)} \times \sum_{n=0} \frac{1}{n!} \left[ \prod_{i=1}^n \int [dk_i] |M(k_i)|^2 \Theta(V(k_i) - \epsilon v) \right] \Theta(v - V(k_1, \dots, k_n))$$

dependence on  $\epsilon v$  cancels for  $\epsilon \rightarrow 0$

This equation can be solved analytically for an observable like thrust, but...

# Anatomy of an all-order resummation

$$\Sigma(v) = e^{-\int [dk] |M(k)|^2 \Theta(V(k) - \epsilon v)} \\ \times \sum_{n=0} \frac{1}{n!} \left[ \prod_{i=1}^n \int [dk_i] |M(k_i)|^2 \Theta(V(k_i) - \epsilon v) \right] \Theta(v - V(k_1, \dots, k_n))$$

This can also be solved with a parton shower! Let's simplify the notation to see this

# Anatomy of an all-order resummation

$$\Sigma(v) = e^{-\int [dk] |M(k)|^2 \Theta(V(k) - \epsilon v)} \\ \times \sum_{n=0} \frac{1}{n!} \left[ \prod_{i=1}^n \int [dk_i] |M(k_i)|^2 \Theta(V(k_i) - \epsilon v) \right] \Theta(v - V(k_1, \dots, k_n))$$

This can also be solved with a parton shower! Let's simplify the notation to see this

1. For thrust, we have (up to NLL):  $V(k_1, \dots, k_n) = \sum_i \tau_i$

2. Introduce

$$R(v) = \int [dk] |M(k)|^2 \Theta(V(k) - v) \quad R'(v) = dR(v)/d \ln(1/v)$$

# Anatomy of an all-order resummation

$$\Sigma(v) = e^{-\int [dk] |M(k)|^2 \Theta(V(k) - \epsilon v)} \\ \times \sum_{n=0} \frac{1}{n!} \left[ \prod_{i=1}^n \int [dk_i] |M(k_i)|^2 \Theta(V(k_i) - \epsilon v) \right] \Theta(v - V(k_1, \dots, k_n))$$

This can also be solved with a parton shower! Let's simplify the notation to see this

1. For thrust, we have (up to NLL):  $V(k_1, \dots, k_n) = \sum_i \tau_i$

2. Introduce

$$R(v) = \int [dk] |M(k)|^2 \Theta(V(k) - v) \quad R'(v) = dR(v)/d \ln(1/v)$$

$$\Sigma(\tau) = e^{-R(\epsilon\tau)} \sum_{n=0} \frac{1}{n!} \prod_{i=1}^n \int_{\epsilon v}^1 \frac{d\tau_i}{\tau_i} R'(\tau_i) \Theta\left(\tau - \sum_{i=1}^n \tau_i\right)$$

# Relation to shower veto algorithm

$$\Sigma(\tau) = e^{-R(\epsilon\tau)} \sum_{n=0} \frac{1}{n!} \prod_{i=1}^n \int_{\epsilon\nu}^1 \frac{d\tau_i}{\tau_i} R'(\tau_i) \Theta \left( \tau - \sum_{i=1}^n \tau_i \right)$$

This is a one-dimensional parton-shower equation with evolution variable  $\tau_i$

Define the Sudakov: describes the no-emission probability between  $\tau_i$  and  $\tau_{i+1}$

$$\Delta(\tau_i, \tau_{i+1}) = \frac{e^{-R(\tau_i)}}{e^{-R(\tau_{i+1})}}$$

# Relation to shower veto algorithm

$$\Sigma(\tau) = e^{-R(\epsilon\tau)} \sum_{n=0} \frac{1}{n!} \prod_{i=1}^n \int_{\epsilon\nu}^1 \frac{d\tau_i}{\tau_i} R'(\tau_i) \Theta \left( \tau - \sum_{i=1}^n \tau_i \right)$$

This is a one-dimensional parton-shower equation with evolution variable  $\tau_i$

Define the Sudakov: describes the no-emission probability between  $\tau_i$  and  $\tau_{i+1}$

$$\Delta(\tau_i, \tau_{i+1}) = \frac{e^{-R(\tau_i)}}{e^{-R(\tau_{i+1})}}$$

$$\Sigma(\tau) = \left( \Delta(1, \epsilon\tau) \right)$$

# Relation to shower veto algorithm

$$\Sigma(\tau) = e^{-R(\epsilon\tau)} \sum_{n=0} \frac{1}{n!} \prod_{i=1}^n \int_{\epsilon\tau}^1 \frac{d\tau_i}{\tau_i} R'(\tau_i) \Theta \left( \tau - \sum_{i=1}^n \tau_i \right)$$

This is a one-dimensional parton-shower equation with evolution variable  $\tau_i$

Define the Sudakov: describes the no-emission probability between  $\tau_i$  and  $\tau_{i+1}$

$$\Delta(\tau_i, \tau_{i+1}) = \frac{e^{-R(\tau_i)}}{e^{-R(\tau_{i+1})}}$$

$$\Sigma(\tau) = \left( \Delta(1, \epsilon\tau) + \int_{\epsilon\tau}^1 \frac{d\tau_1}{\tau_1} \Delta(1, \tau_1) R'(\tau_1) \Delta(\tau_1, \epsilon\tau) \right) \Theta(\tau - \tau_1)$$

no emissions until  
scale  $\tau_1$  is reached

# Relation to shower veto algorithm

$$\Sigma(\tau) = e^{-R(\epsilon\tau)} \sum_{n=0} \frac{1}{n!} \prod_{i=1}^n \int_{\epsilon\tau}^1 \frac{d\tau_i}{\tau_i} R'(\tau_i) \Theta \left( \tau - \sum_{i=1}^n \tau_i \right)$$

This is a one-dimensional parton-shower equation with evolution variable  $\tau_i$

Define the Sudakov: describes the no-emission probability between  $\tau_i$  and  $\tau_{i+1}$

$$\Delta(\tau_i, \tau_{i+1}) = \frac{e^{-R(\tau_i)}}{e^{-R(\tau_{i+1})}}$$

$$\Sigma(\tau) = \left( \Delta(1, \epsilon\tau) + \int_{\epsilon\tau}^1 \frac{d\tau_1}{\tau_1} \Delta(1, \tau_1) R'(\tau_1) \Delta(\tau_1, \epsilon\tau) \right) \Theta(\tau - \tau_1)$$

one real  
emission  
at scale  $\tau_1$

# Relation to shower veto algorithm

$$\Sigma(\tau) = e^{-R(\epsilon\tau)} \sum_{n=0} \frac{1}{n!} \prod_{i=1}^n \int_{\epsilon\tau}^1 \frac{d\tau_i}{\tau_i} R'(\tau_i) \Theta \left( \tau - \sum_{i=1}^n \tau_i \right)$$

This is a one-dimensional parton-shower equation with evolution variable  $\tau_i$

Define the Sudakov: describes the no-emission probability between  $\tau_i$  and  $\tau_{i+1}$

$$\Delta(\tau_i, \tau_{i+1}) = \frac{e^{-R(\tau_i)}}{e^{-R(\tau_{i+1})}}$$

$$\Sigma(\tau) = \left( \Delta(1, \epsilon\tau) + \int_{\epsilon\tau}^1 \frac{d\tau_1}{\tau_1} \Delta(1, \tau_1) R'(\tau_1) \Delta(\tau_1, \epsilon\tau) \right) \Theta(\tau - \tau_1)$$

no further  
emissions until  
cutoff scale

# Relation to shower veto algorithm

$$\Sigma(\tau) = e^{-R(\epsilon\tau)} \sum_{n=0} \frac{1}{n!} \prod_{i=1}^n \int_{\epsilon\tau}^1 \frac{d\tau_i}{\tau_i} R'(\tau_i) \Theta \left( \tau - \sum_{i=1}^n \tau_i \right)$$

This is a one-dimensional parton-shower equation with evolution variable  $\tau_i$

Define the Sudakov: describes the no-emission probability between  $\tau_i$  and  $\tau_{i+1}$

$$\Delta(\tau_i, \tau_{i+1}) = \frac{e^{-R(\tau_i)}}{e^{-R(\tau_{i+1})}}$$

$$\Sigma(\tau) = \left( \Delta(1, \epsilon\tau) + \int_{\epsilon\tau}^1 \frac{d\tau_1}{\tau_1} \Delta(1, \tau_1) R'(\tau_1) \Delta(\tau_1, \epsilon\tau) \right) \Theta(\tau - \tau_1)$$

at the end of evolution,  
construct the observable

# Relation to shower veto algorithm

$$\Sigma(\tau) = e^{-R(\epsilon\tau)} \sum_{n=0} \frac{1}{n!} \prod_{i=1}^n \int_{\epsilon\tau}^1 \frac{d\tau_i}{\tau_i} R'(\tau_i) \Theta \left( \tau - \sum_{i=1}^n \tau_i \right)$$

This is a one-dimensional parton-shower equation with evolution variable  $\tau_i$

Define the Sudakov: describes the no-emission probability between  $\tau_i$  and  $\tau_{i+1}$

$$\Delta(\tau_i, \tau_{i+1}) = \frac{e^{-R(\tau_i)}}{e^{-R(\tau_{i+1})}}$$

$$\Sigma(\tau) = \left( \Delta(1, \epsilon\tau) + \int_{\epsilon\tau}^1 \frac{d\tau_1}{\tau_1} \Delta(1, \tau_1) R'(\tau_1) \Delta(\tau_1, \epsilon\tau) \right. \\ \left. + \int_{\epsilon\tau}^1 \frac{d\tau_1}{\tau_1} \int_{\epsilon\tau}^{\tau_1} \frac{d\tau_2}{\tau_2} \Delta(1, \tau_1) R'(\tau_1) \Delta(\tau_1, \tau_2) R'(\tau_2) \Delta(\tau_2, \epsilon\tau) + \dots \right) \Theta \left( \tau - \sum_{i=1}^n \tau_i \right)$$

# Relation to shower veto algorithm

$$\Sigma(\tau) = e^{-R(\epsilon\tau)} \sum_{n=0} \frac{1}{n!} \prod_{i=1}^n \int_{\epsilon\tau}^1 \frac{d\tau_i}{\tau_i} R'(\tau_i) \Theta \left( \tau - \sum_{i=1}^n \tau_i \right)$$

This is a one-dimensional parton-shower equation with evolution variable  $\tau_i$

Define the Sudakov: describes the no-emission probability between  $\tau_i$  and  $\tau_{i+1}$

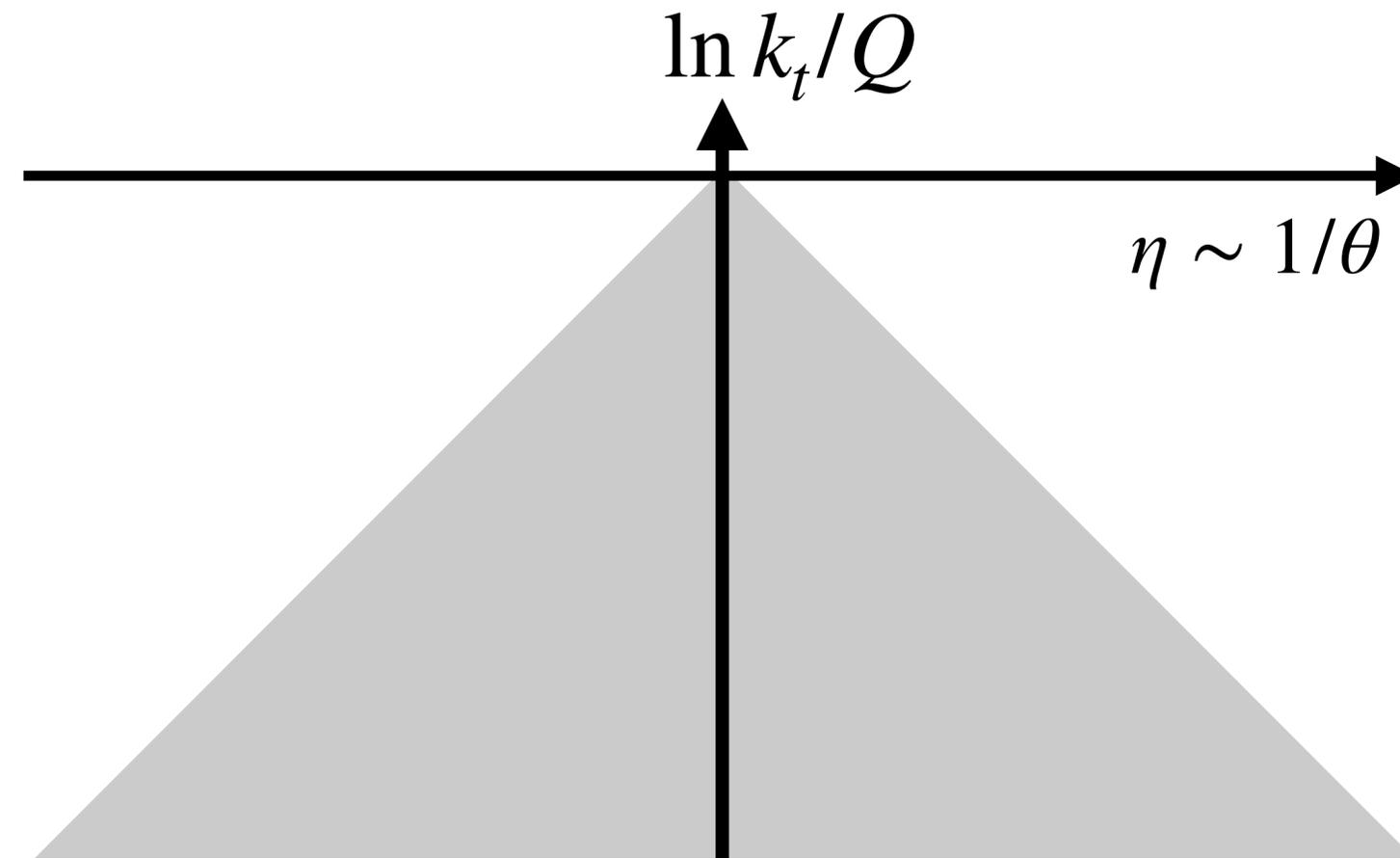
$$\Delta(\tau_i, \tau_{i+1}) = \frac{e^{-R(\tau_i)}}{e^{-R(\tau_{i+1})}}$$

$$\Sigma(\tau) = \left( \Delta(1, \epsilon\tau) + \int_{\epsilon\tau}^1 \frac{d\tau_1}{\tau_1} \Delta(1, \tau_1) R'(\tau_1) \Delta(\tau_1, \epsilon\tau) \right. \\ \left. + \int_{\epsilon\tau}^1 \frac{d\tau_1}{\tau_1} \int_{\epsilon\tau}^{\tau_1} \frac{d\tau_2}{\tau_2} \Delta(1, \tau_1) R'(\tau_1) \Delta(\tau_1, \tau_2) R'(\tau_2) \Delta(\tau_2, \epsilon\tau) + \dots \right) \Theta \left( \tau - \sum_{i=1}^n \tau_i \right)$$

we've replaced the  $1/n!$  by an ordering

# Choice of ordering variable

The ordering/evolution variable is a choice

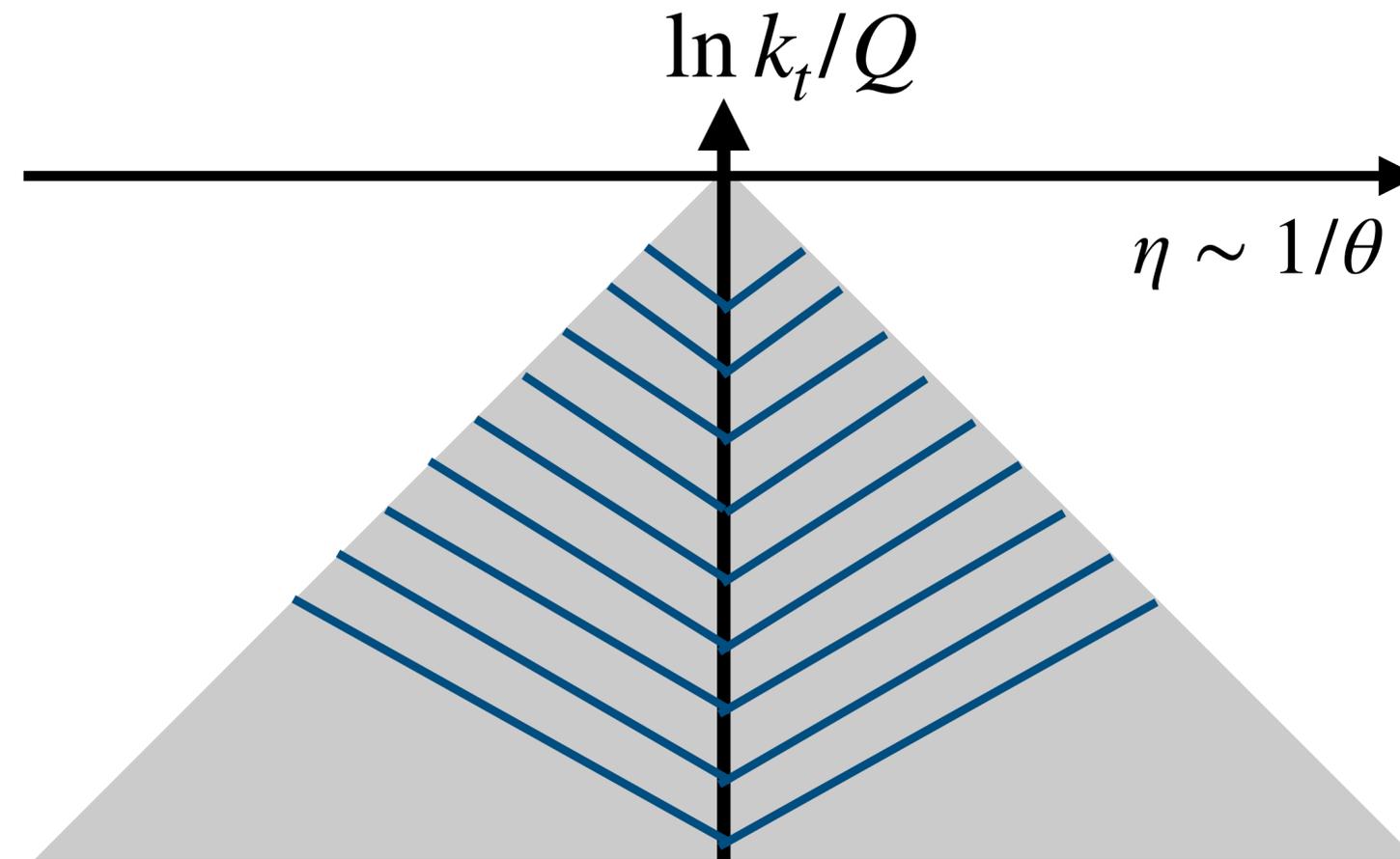


# Choice of ordering variable

The ordering/evolution variable is a choice

$$t_i = \frac{k_{t,i}}{Q} e^{-|\eta_i|}$$

example of a  
virtuality/time  
ordered shower

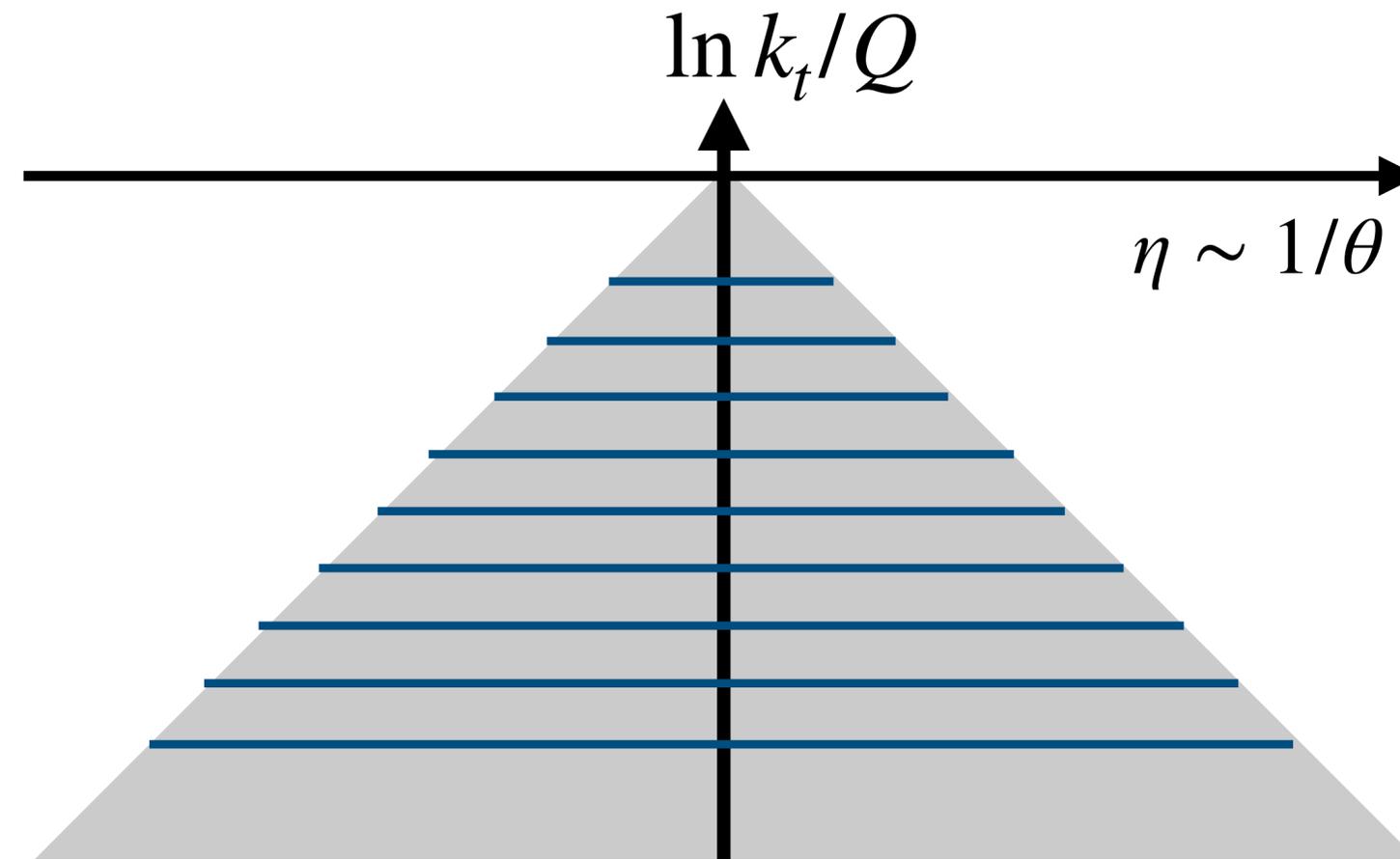


# Choice of ordering variable

The ordering/evolution variable is a choice

$$t_i = \frac{k_{t,i}}{Q}$$

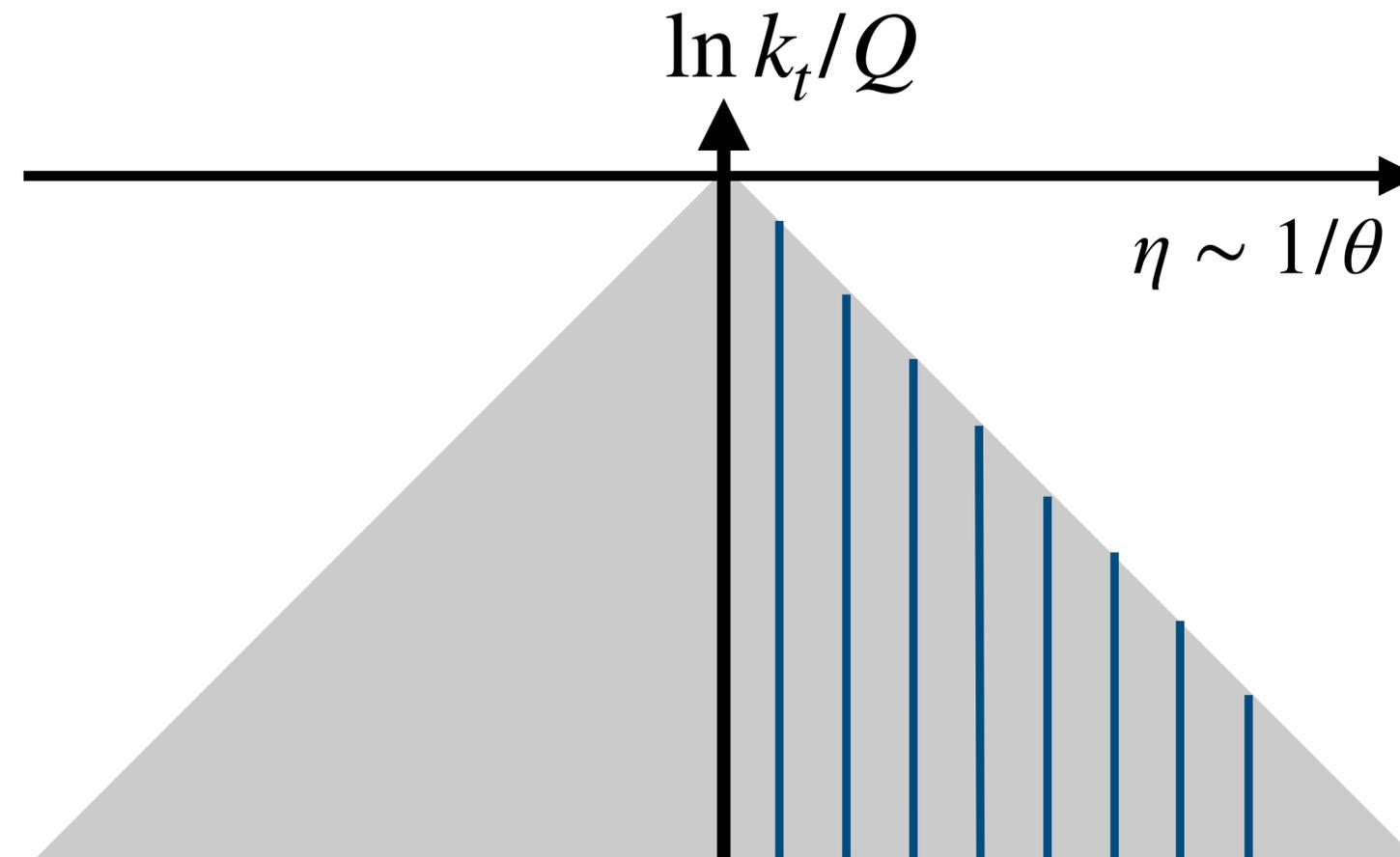
example of a  
transverse-  
momentum ordered  
shower



# Choice of ordering variable

The ordering/evolution variable is a choice

$t_i = \theta$   
example of an  
angular ordered  
shower



# 'The' or 'a' shower equation?

$$\Sigma(v) = \left( \Delta(1, t_c) + \int_{t_c}^1 \frac{dt_1}{t_1} \Delta(1, t_1) R'(t_1) \Delta(t_1, t_c) \right. \\ \left. + \int_{t_c}^1 \frac{dt_1}{t_1} \int_{t_c}^{t_1} \frac{dt_2}{t_2} \Delta(1, t_1) R'(t_1) \Delta(t_1, t_2) R'(t_2) \Delta(t_1, t_c) + \dots \right) \Theta(v - V(k_1, \dots, k_n))$$

**We cannot produce strictly soft/collinear particles (this would be divergent)**

Plus remember: we want our shower to work for general observables

→ we need to be differential in the emission kinematics  $k_T$ ,  $\eta$  and  $\phi$

→ at every step of the shower, we need to generate three random numbers and choose how to interpret those in terms of actual momenta

# Momentum mapping

**Basic problem:** after every splitting, we need to decide how to go from  $n$  momenta to  $n + 1$  momenta

# Momentum mapping

**Basic problem:** after every splitting, we need to decide how to go from  $n$  momenta to  $n + 1$  momenta

- A few ‘rules’:
  - We cannot violate total momentum conservation
  - We have to make sure that all particles are on-shell at every step, as the shower can terminate at any moment or device a scheme to make up for this at the end
  - In the collinear limit we need to identify a (IR safe) momentum fraction  $z$

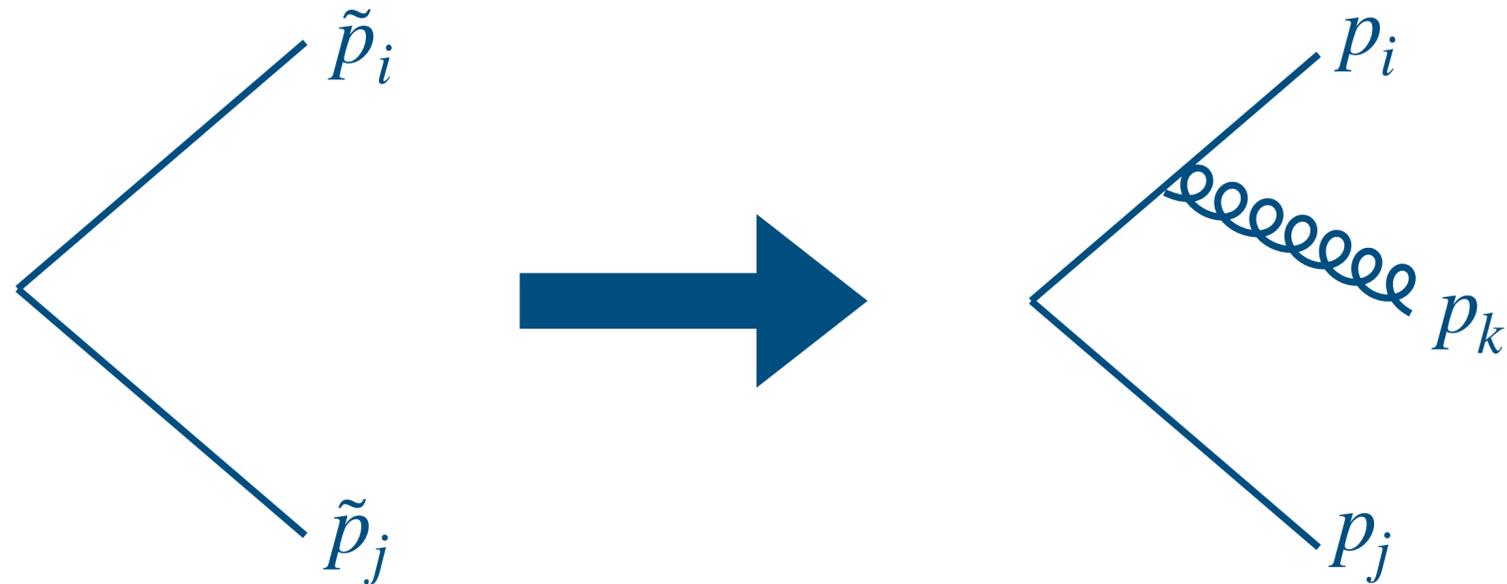
# Momentum mapping

**Basic problem:** after every splitting, we need to decide how to go from  $n$  momenta to  $n + 1$  momenta

- A few ‘rules’:
  - We cannot violate total momentum conservation
  - We have to make sure that all particles are on-shell at every step, as the shower can terminate at any moment or device a scheme to make up for this at the end
  - In the collinear limit we need to identify a (IR safe) momentum fraction  $z$
- Other than that: freedom (at LL that is)
  - Reshuffle momenta ‘dipole local’ or event-wide (global)?
  - What does the map do beyond the strict unresolved limits?

# Momentum mapping

- Example: Pythia final-state dipole-local map

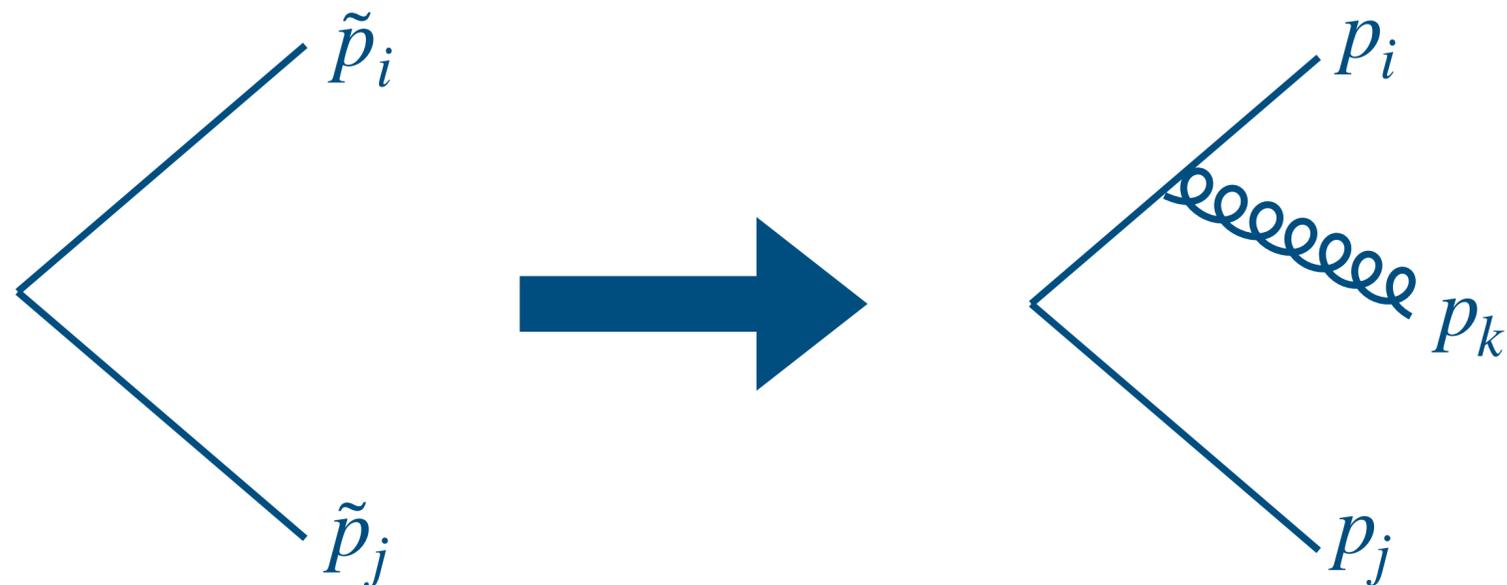


Emitter:  $p_i = a_i \tilde{p}_i + b_i \tilde{p}_j + k_{\perp}$   
Emission (new):  $p_k = a_k \tilde{p}_i + b_k \tilde{p}_j - k_{\perp}$   
Spectator:  $p_j = (1 - b_j) \tilde{p}_j$

- Transverse momentum  $k_{\perp}$  is orthogonal to pre-splitting dipole momenta  $\tilde{p}_i$ ,  $\tilde{p}_j$  (and depends on  $\phi$ )

# Momentum mapping

- Example: Pythia final-state dipole-local map



Emitter:  $p_i = a_i \tilde{p}_i + b_i \tilde{p}_j + k_{\perp}$   
Emission (new):  $p_k = a_k \tilde{p}_i + b_k \tilde{p}_j - k_{\perp}$   
Spectator:  $p_j = (1 - b_j) \tilde{p}_j$

- Transverse momentum  $k_{\perp}$  is orthogonal to pre-splitting dipole momenta  $\tilde{p}_i, \tilde{p}_j$  (and depends on  $\phi$ )
- Constraints:
  - Momenta are assumed to be massless:  $p_i^2 = p_j^2 = p_k^2 = 0$
  - Sum should be the same before and after:  $p_i + p_j + p_k = \tilde{p}_i + \tilde{p}_j$

# Momentum mapping

- Example: Pythia final-state dipole-local map

Emitter:  $p_i = a_i \tilde{p}_i + b_i \tilde{p}_j + k_\perp$

Emission (new):  $p_k = a_k \tilde{p}_i + b_k \tilde{p}_j - k_\perp$

Spectator:  $p_j = (1 - b_j) \tilde{p}_j$

- Choose to define

$$\kappa_\perp^2 \equiv \frac{v^2}{2\tilde{p}_i \cdot \tilde{p}_j} \quad y \equiv \frac{\kappa_\perp^2}{z(1-z)} \quad \tilde{z} \equiv \frac{z(1+y) - y}{1-y}$$

# Momentum mapping

- Example: Pythia final-state dipole-local map

Emitter:  $p_i = a_i \tilde{p}_i + b_i \tilde{p}_j + k_\perp$

Emission (new):  $p_k = a_k \tilde{p}_i + b_k \tilde{p}_j - k_\perp$

Spectator:  $p_j = (1 - b_j) \tilde{p}_j$

- Choose to define

$$\kappa_\perp^2 \equiv \frac{v^2}{2\tilde{p}_i \cdot \tilde{p}_j} \quad y \equiv \frac{\kappa_\perp^2}{z(1-z)} \quad \tilde{z} \equiv \frac{z(1+y) - y}{1-y}$$

- And then set  $a_k = 1 - \tilde{z}$  and  $b_k = y\tilde{z}$ 
  - for collinear emissions:  $y \rightarrow 0$  and  $\tilde{z} \simeq z$
  - for soft emissions:  $\tilde{z} \rightarrow 1$  with  $y \rightarrow 0$

# Momentum mapping

- Example: Pythia final-state dipole-local map

Emitter:  $p_i = a_i \tilde{p}_i + b_i \tilde{p}_j + k_\perp$

Emission (new):  $p_k = a_k \tilde{p}_i + b_k \tilde{p}_j - k_\perp$

Spectator:  $p_j = (1 - b_j) \tilde{p}_j$

- Choose to define

$$\kappa_\perp^2 \equiv \frac{v^2}{2\tilde{p}_i \cdot \tilde{p}_j} \quad y \equiv \frac{\kappa_\perp^2}{z(1-z)} \quad \tilde{z} \equiv \frac{z(1+y) - y}{1-y}$$

- And then set  $a_k = 1 - \tilde{z}$  and  $b_k = y\tilde{z}$ 
  - for collinear emissions:  $y \rightarrow 0$  and  $\tilde{z} \simeq z$
  - for soft emissions:  $\tilde{z} \rightarrow 1$  with  $y \rightarrow 0$

From the constraints one can derive the other coefficients and  $k_\perp^2$

# Momentum mapping

- Example: Pythia final-state dipole-local map

Emitter:  $p_i = a_i \tilde{p}_i + b_i \tilde{p}_j + k_\perp$

Emission (new):  $p_k = a_k \tilde{p}_i + b_k \tilde{p}_j - k_\perp$

Spectator:  $p_j = (1 - b_j) \tilde{p}_j$

- Choose to define

$$\kappa_\perp^2 \equiv \frac{v^2}{2\tilde{p}_i \cdot \tilde{p}_j} \quad y \equiv \frac{\kappa_\perp^2}{z(1-z)} \quad \tilde{z} \equiv \frac{z(1+y) - y}{1-y}$$

- And then set  $a_k = 1 - \tilde{z}$  and  $b_k = y\tilde{z}$ 
  - for collinear emissions:  $y \rightarrow 0$  and  $\tilde{z} \simeq z$
  - for soft emissions:  $\tilde{z} \rightarrow 1$  with  $y \rightarrow 0$

But this is just one set of choices!

# Connection to resummation

What all showers do:

- Decide on a physical meaning of  $\nu$  (the evolution variable)
- Implement some form of momentum mapping
- Distribute probabilities proportional to QCD splitting functions

Where showers differ:

- What happens after one single emission (i.e. a correlated pair of emissions)?
- What happens outside the strict collinear and soft limits?

# Connection to resummation

What all showers do:

- Decide on a physical meaning of  $\nu$  (the evolution variable)
- Implement some form of momentum mapping
- Distribute probabilities proportional to QCD splitting functions

Where showers differ:

- What happens after one single emission (i.e. a correlated pair of emissions)? **important for resummation**
- What happens outside the strict collinear and soft limits? **important for higher-order matching**

# Connection to resummation

What all showers do:

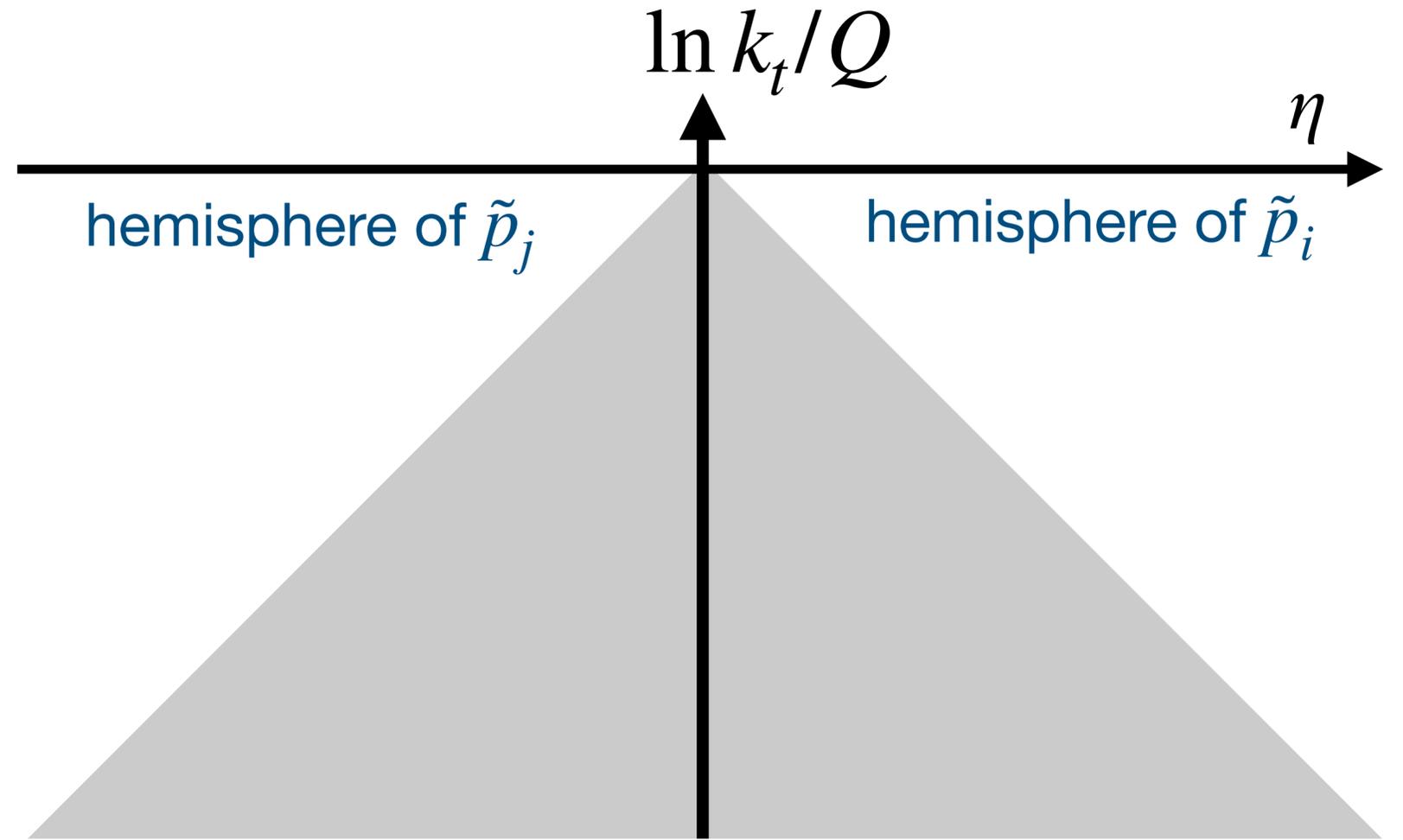
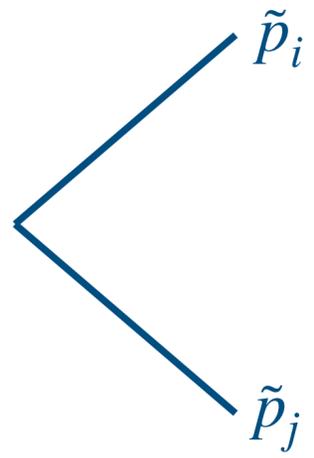
- Decide on a physical meaning of  $\nu$  (the evolution variable)
- Implement some form of momentum mapping
- Distribute probabilities proportional to QCD splitting functions

Where showers differ:

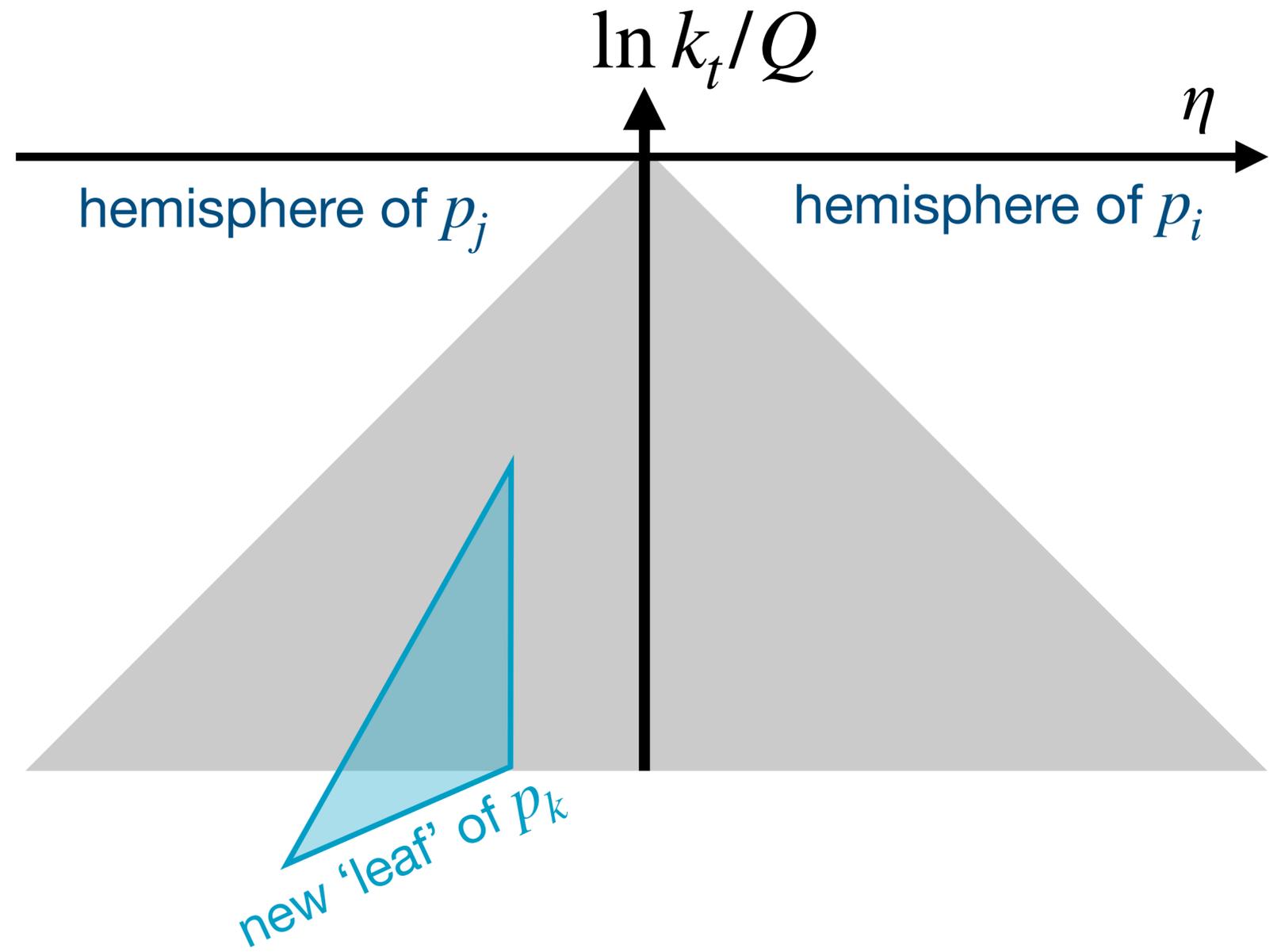
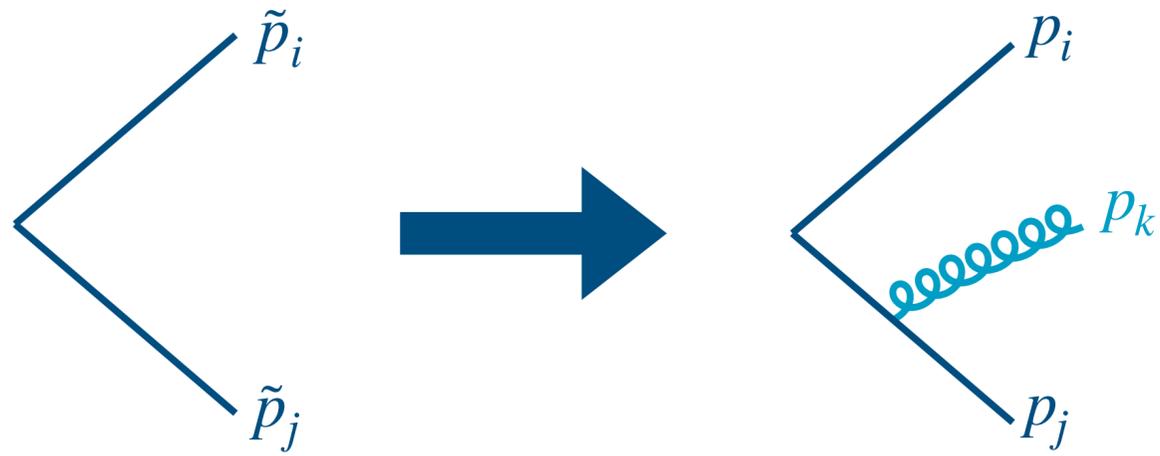
- What happens after one single emission (i.e. a correlated pair of emissions)? **important for resummation**
- What happens outside the strict collinear and soft limits? **important for higher-order matching**

Choices are not equal when requiring that the shower has to reproduce a resummed calculation of given accuracy!

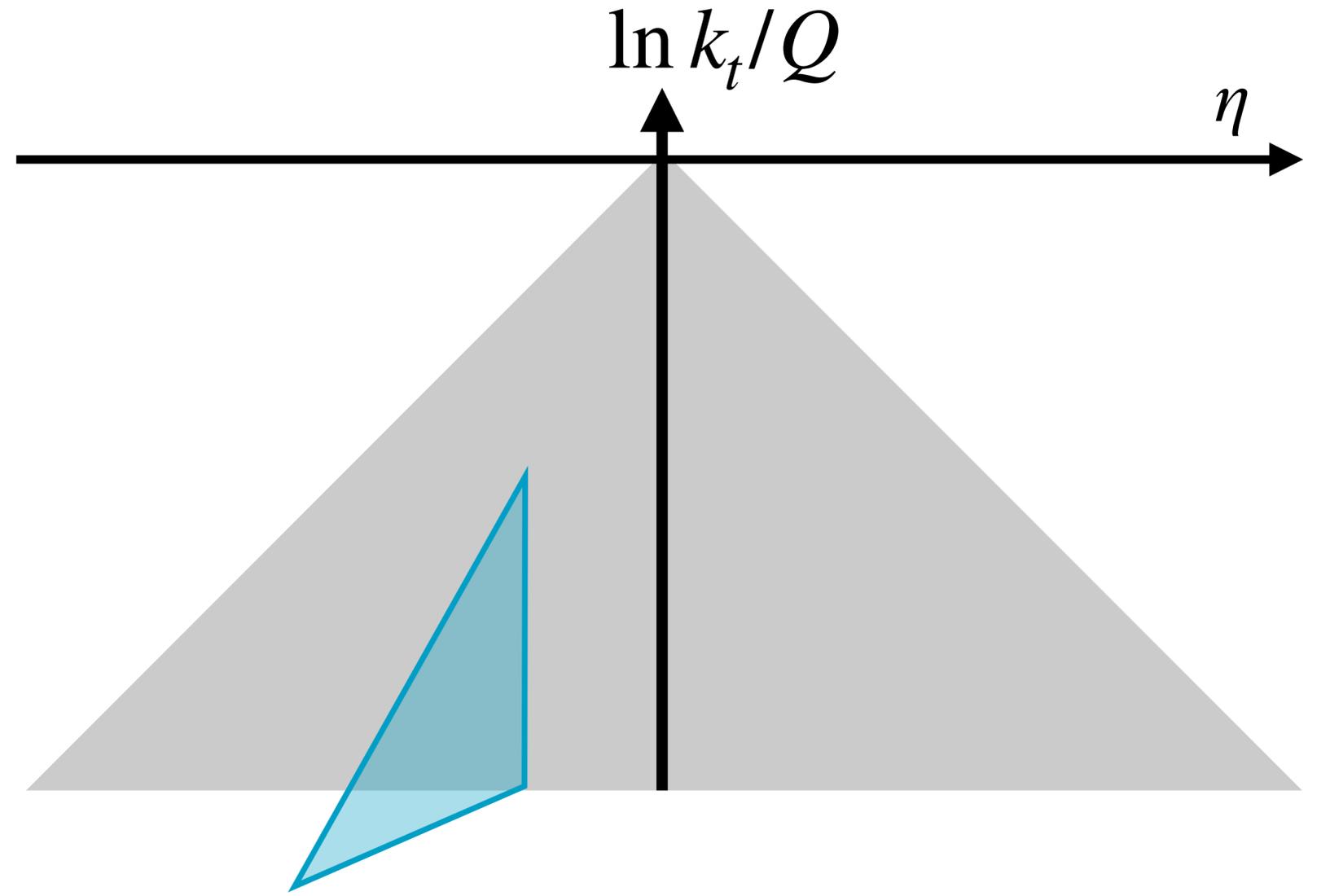
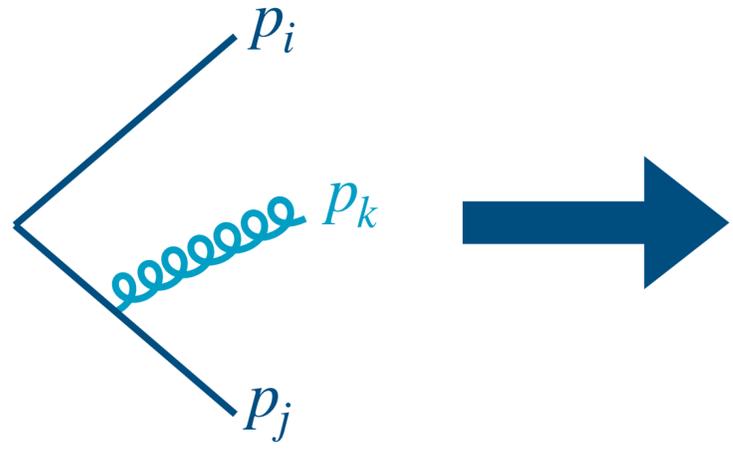
# Momentum mapping



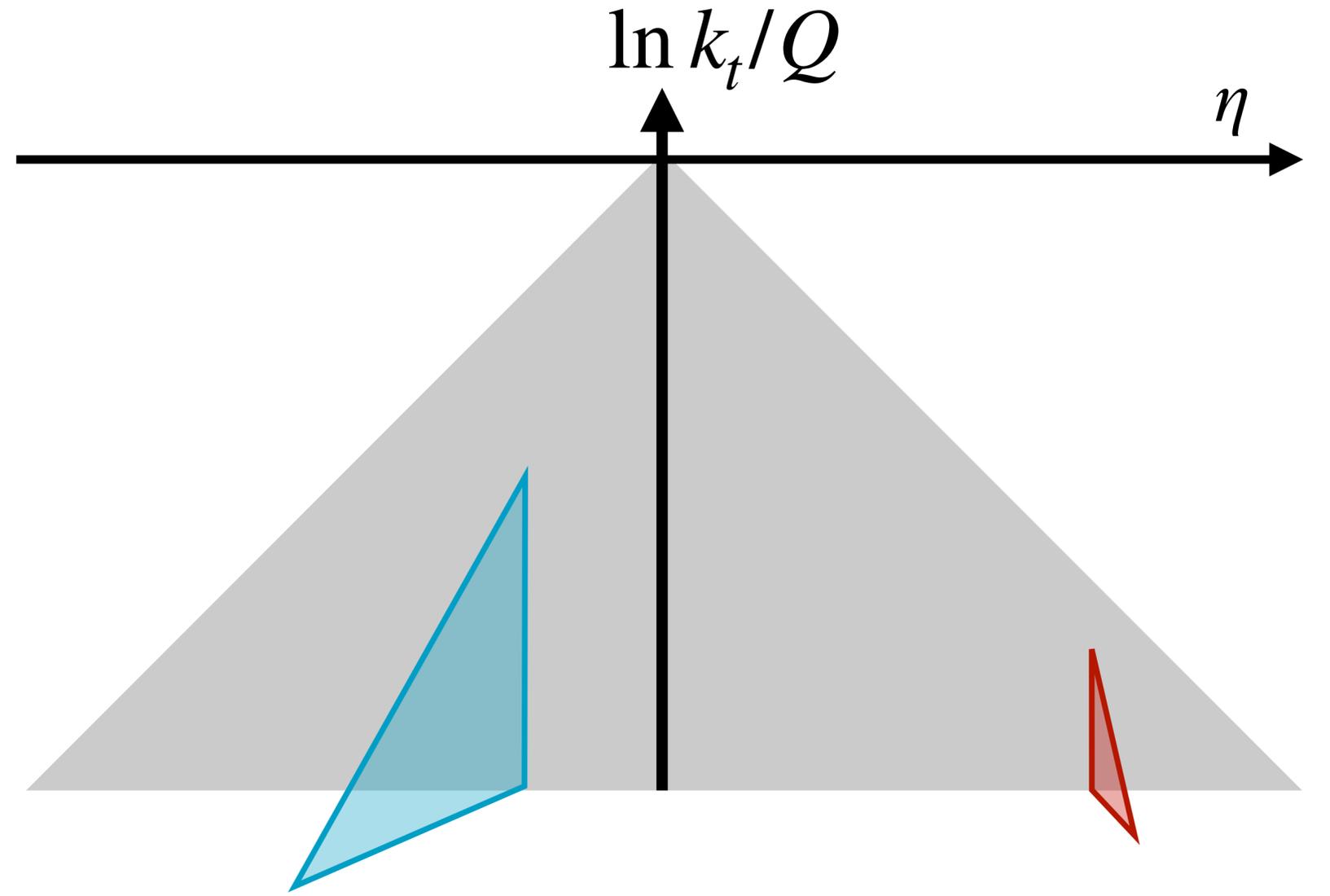
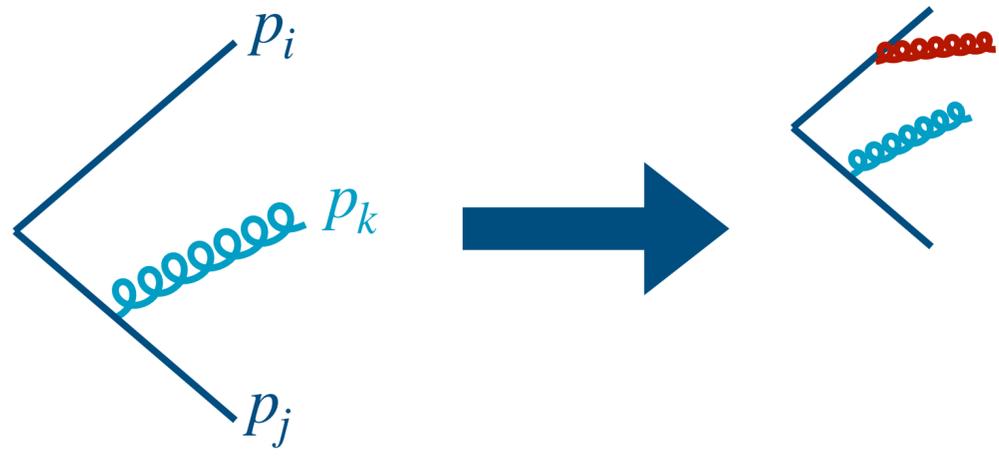
# Momentum mapping



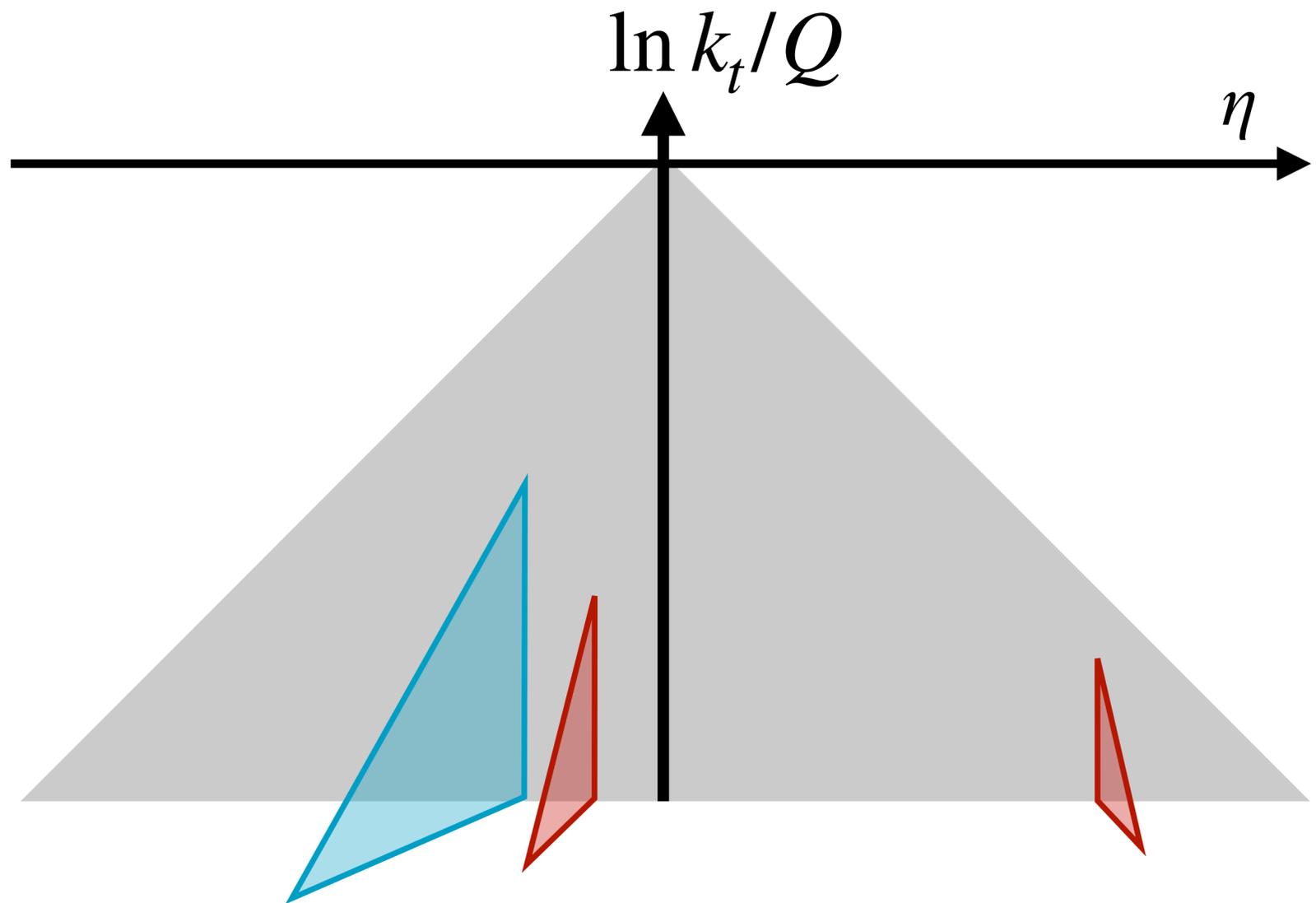
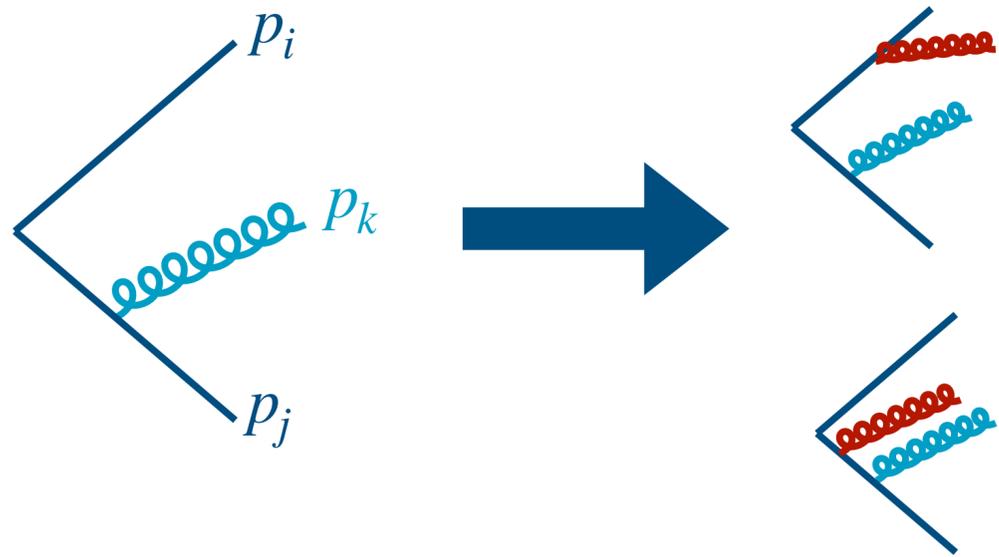
# Momentum mapping



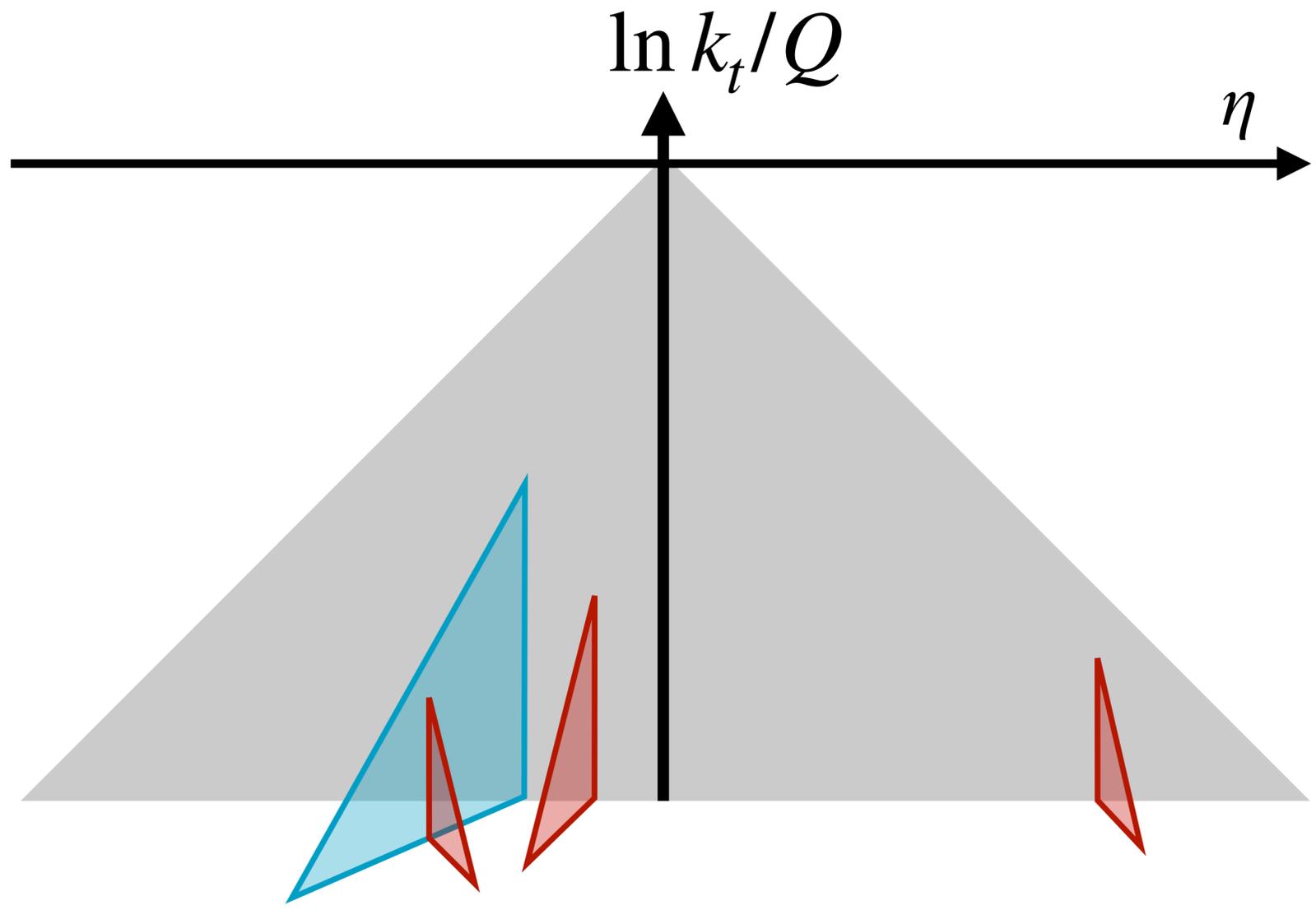
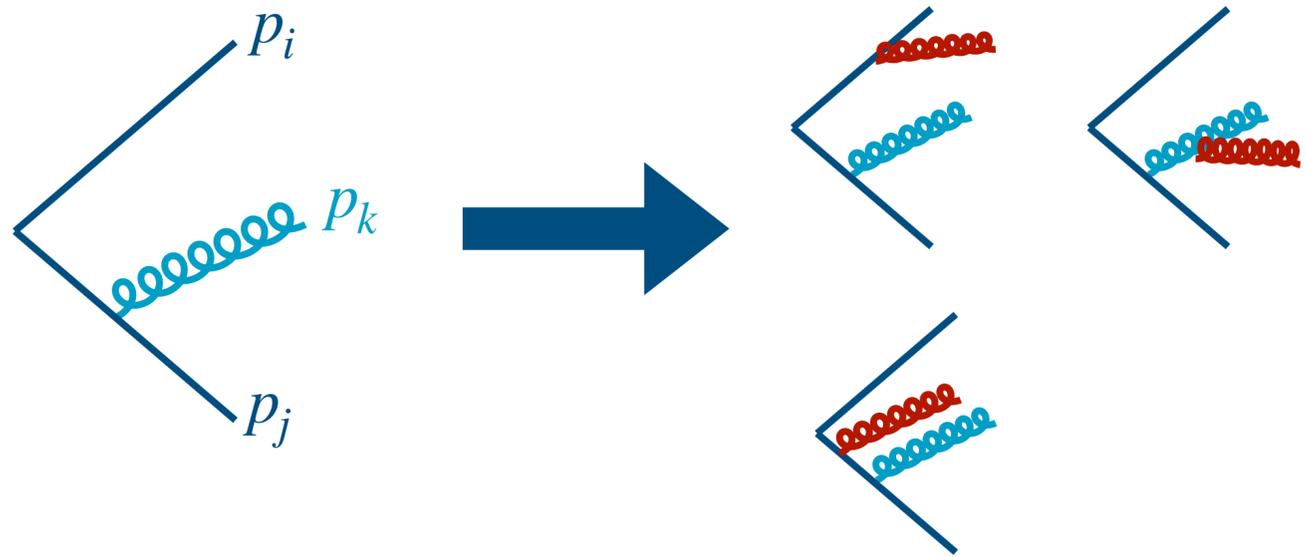
# Momentum mapping



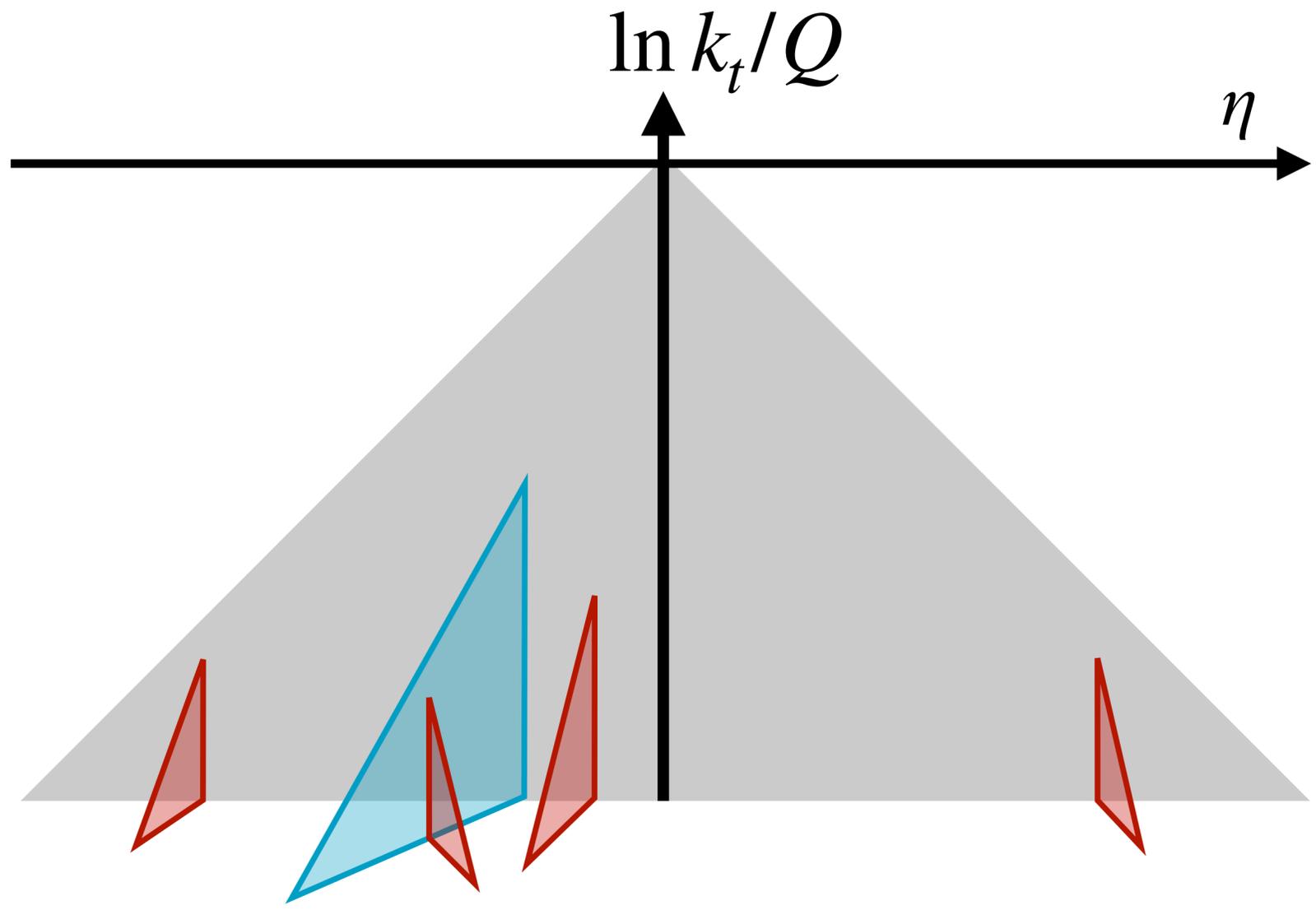
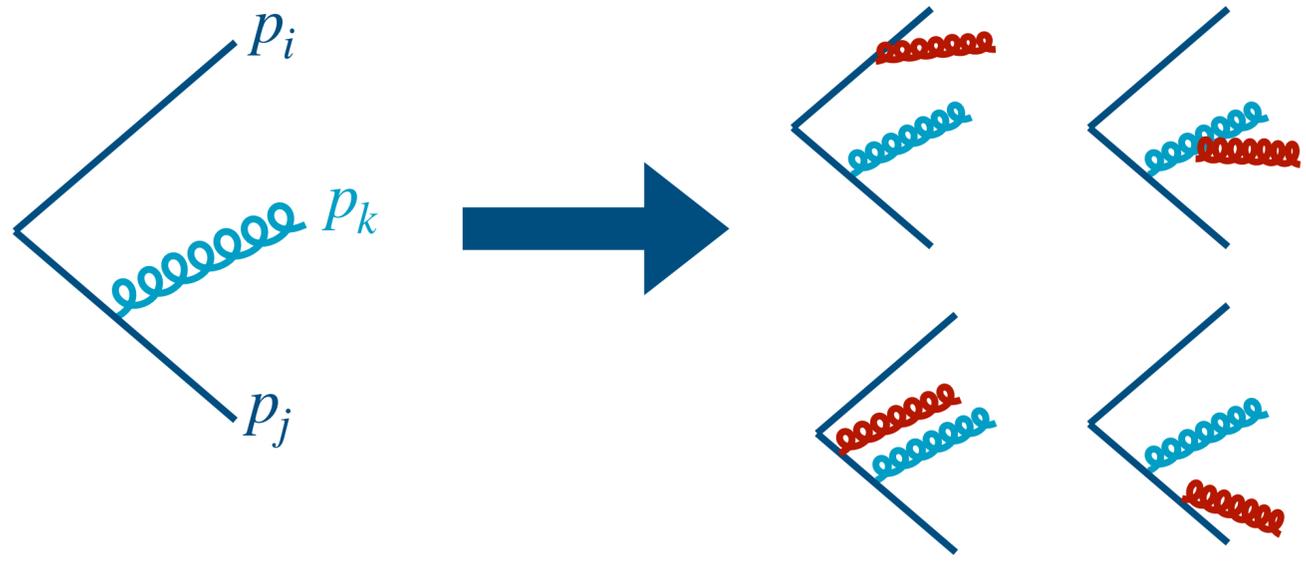
# Momentum mapping



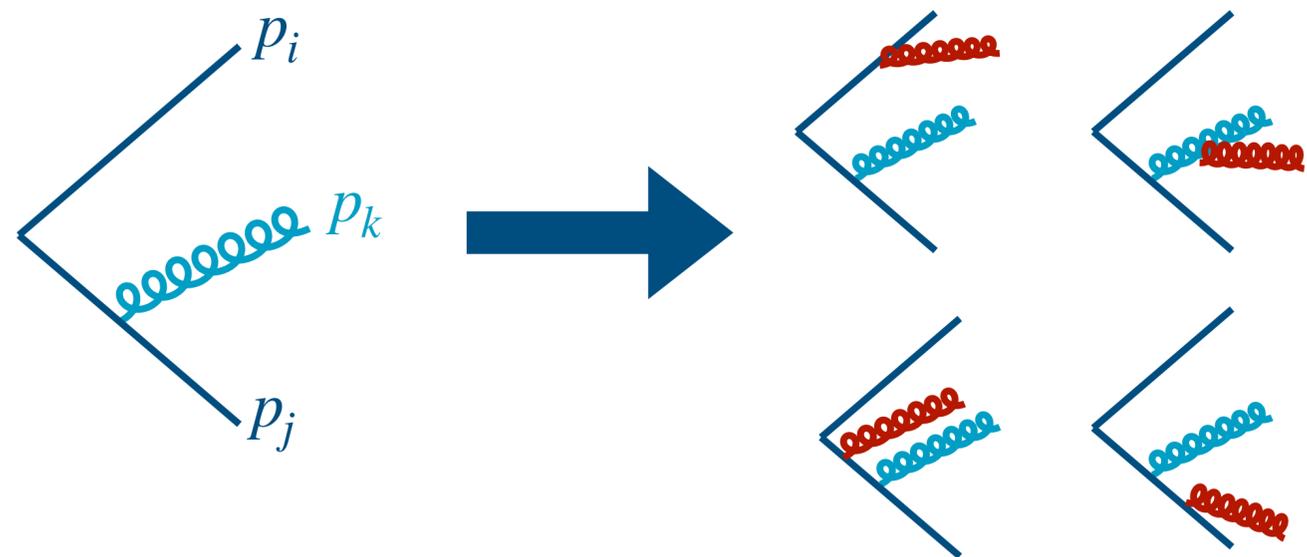
# Momentum mapping



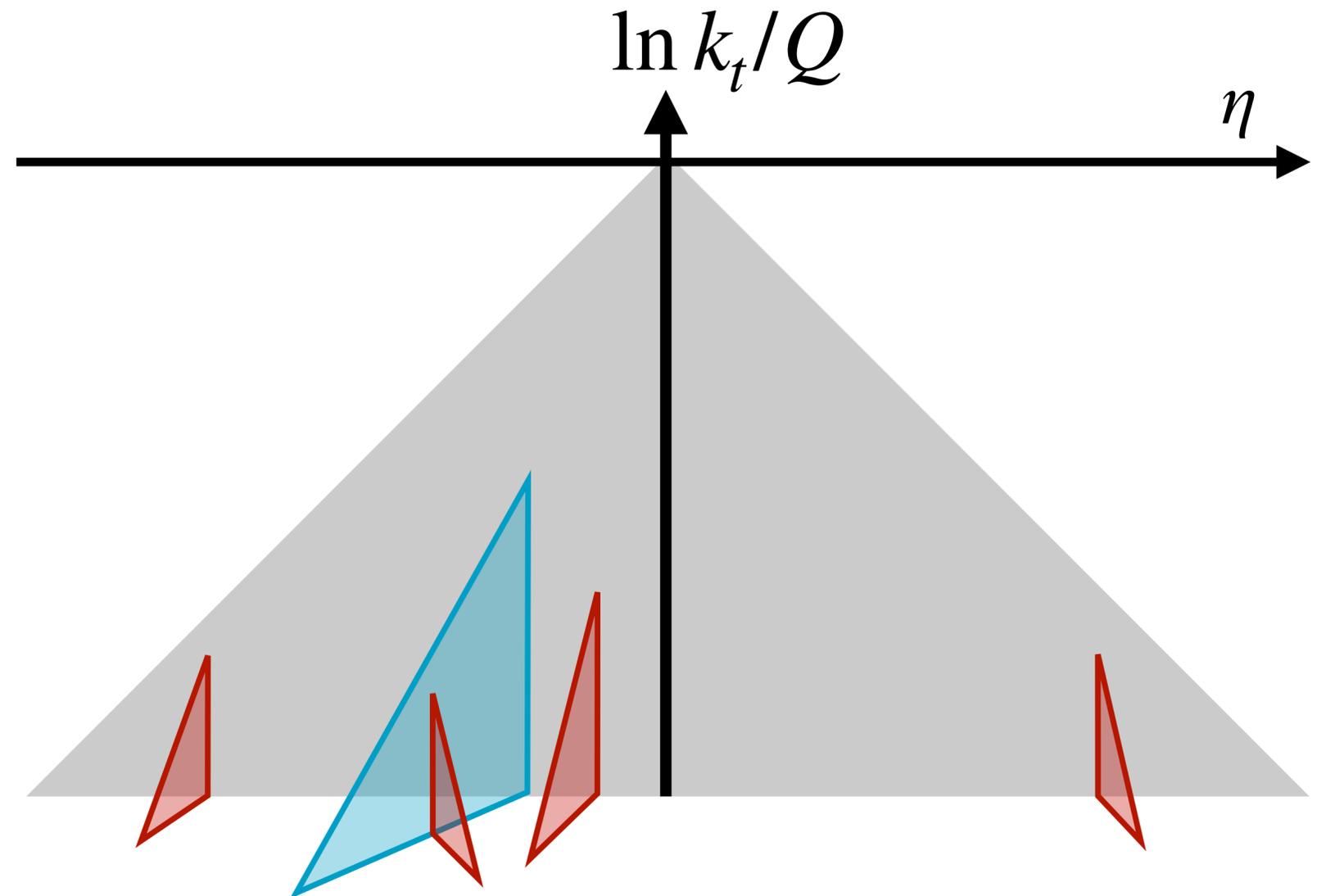
# Momentum mapping



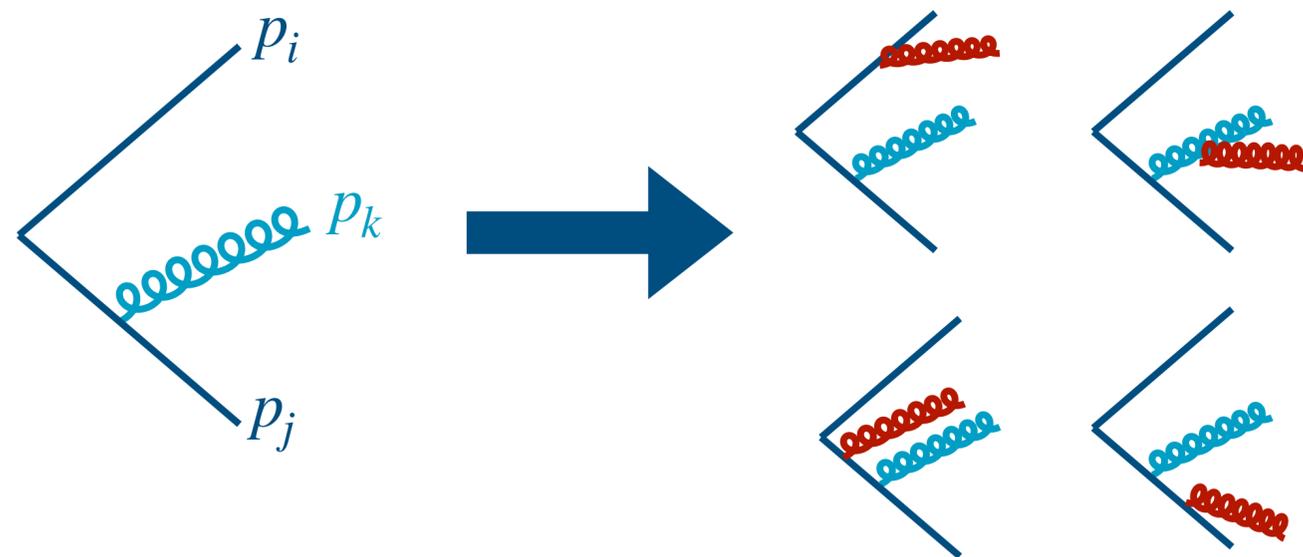
# Momentum mapping



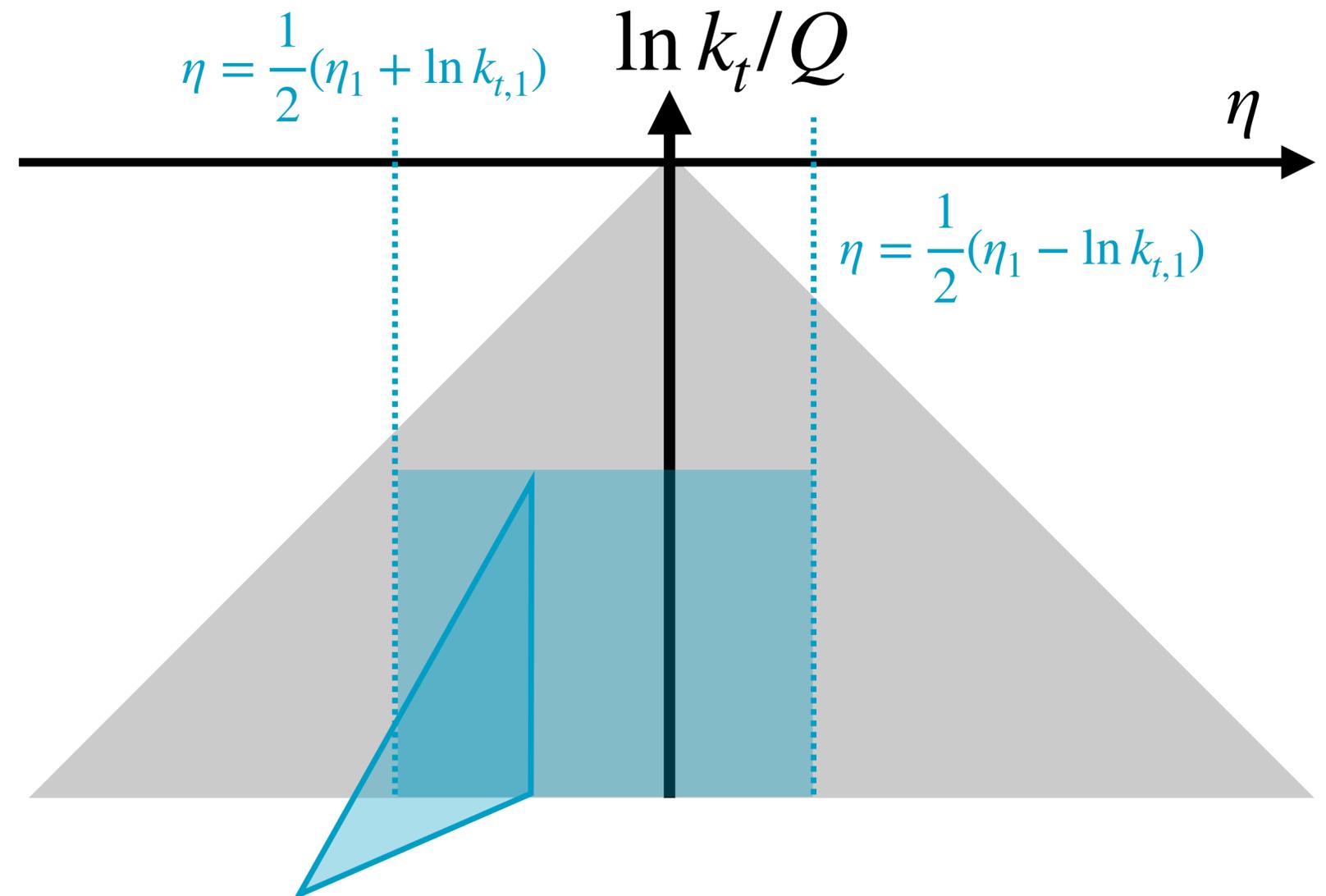
the colour factor in the ME has to be  $C_F$  if emitted from the primary plane,  $C_A/2$  if emitted from the secondary plane



# Momentum mapping



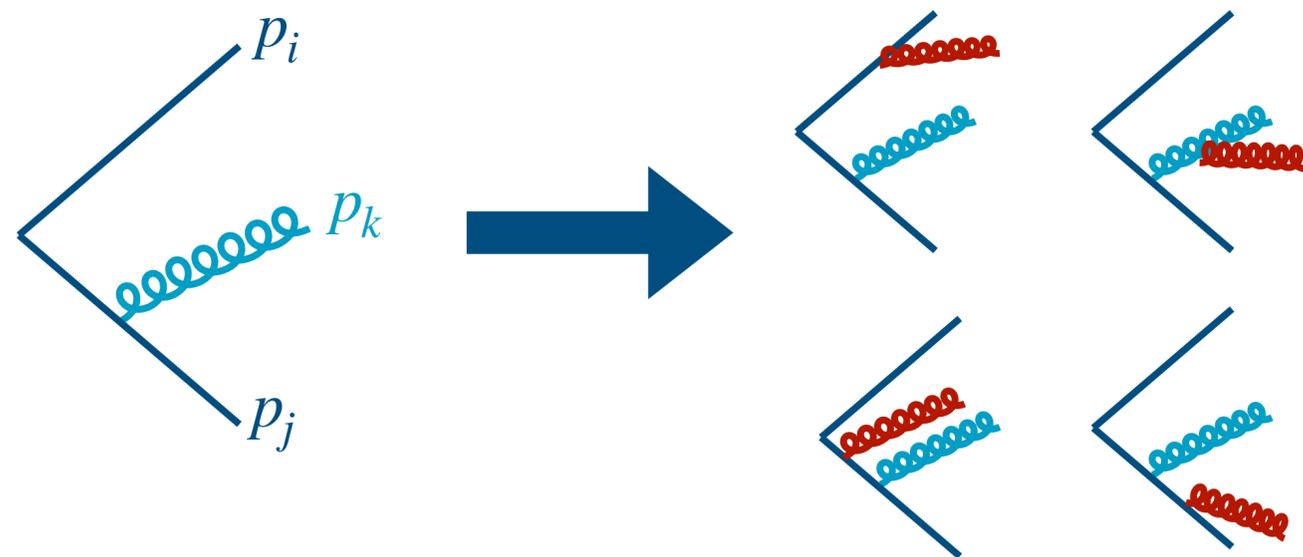
the colour factor in the ME has to be  $C_F$  if emitted from the primary plane,  
 $C_A/2$  if emitted from the secondary plane



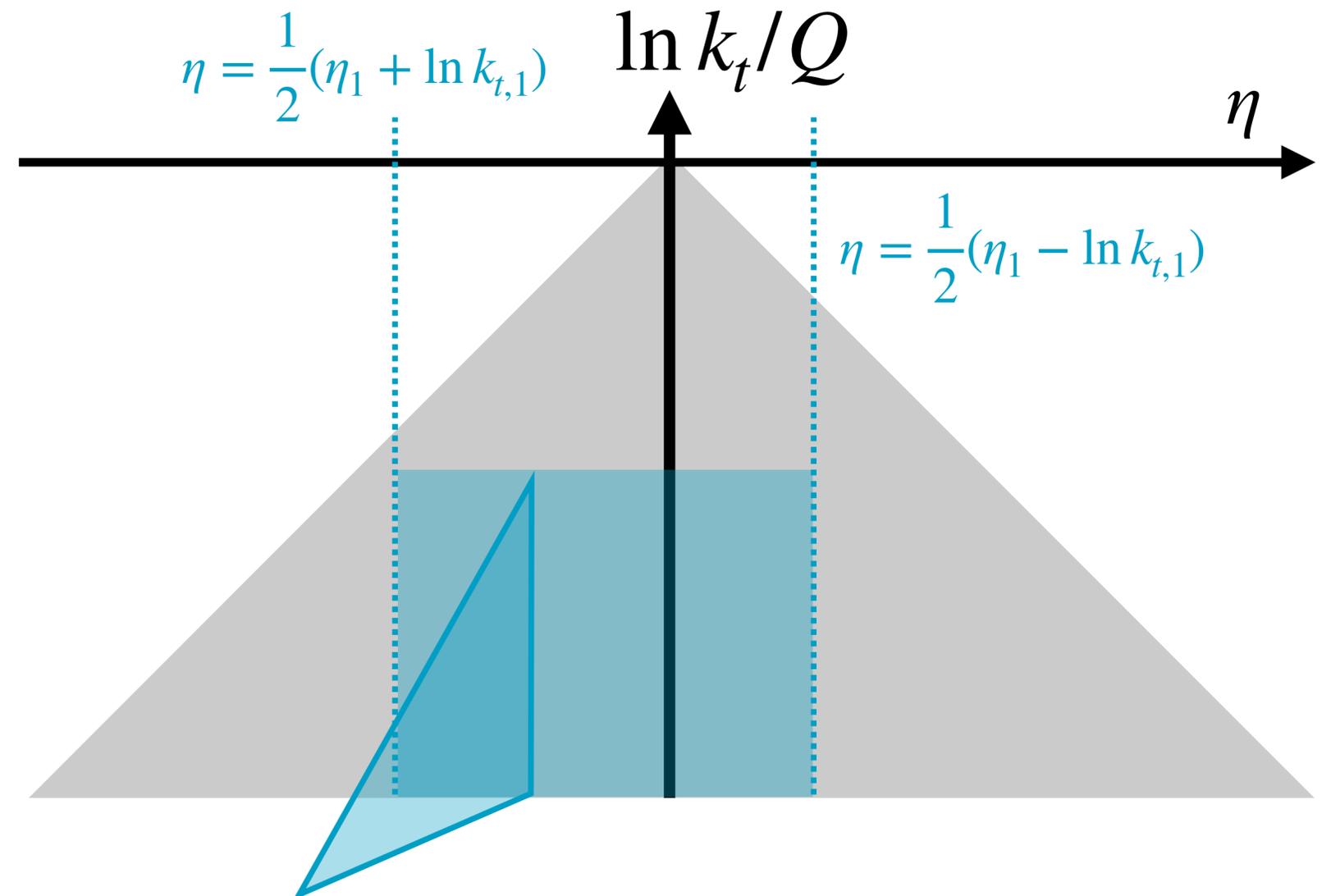
**Issue 1:** 'standard'  $k_T$ -ordered dipole showers get this wrong, and assign a  $C_A/2$  in an extended (logarithmic) region of phase-space

this is a consequence of choosing the 'wrong' emitter, a problem that angular-ordered parton showers do not have

# Momentum mapping



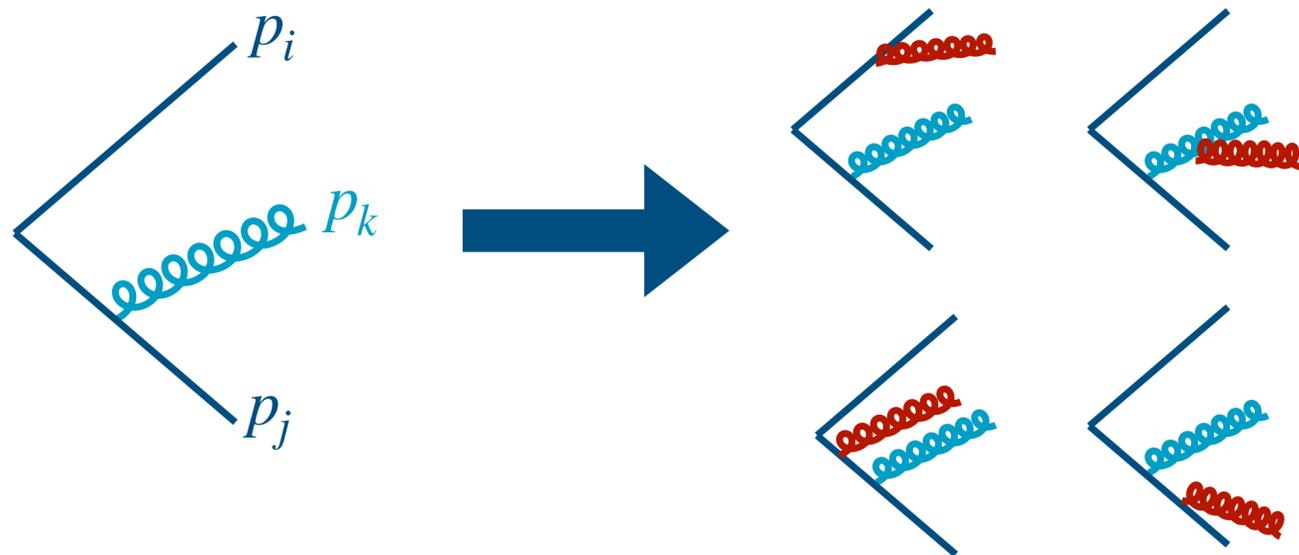
the colour factor in the ME has to be  $C_F$  if emitted from the primary plane,  $C_A/2$  if emitted from the secondary plane



**Issue 1:** 'standard'  $k_T$ -ordered dipole showers get this wrong, and assign a  $C_A/2$  in an extended (logarithmic) region of phase-space

This breaks LL accuracy for global event shapes (like thrust) beyond leading  $N_c$

# Momentum mapping

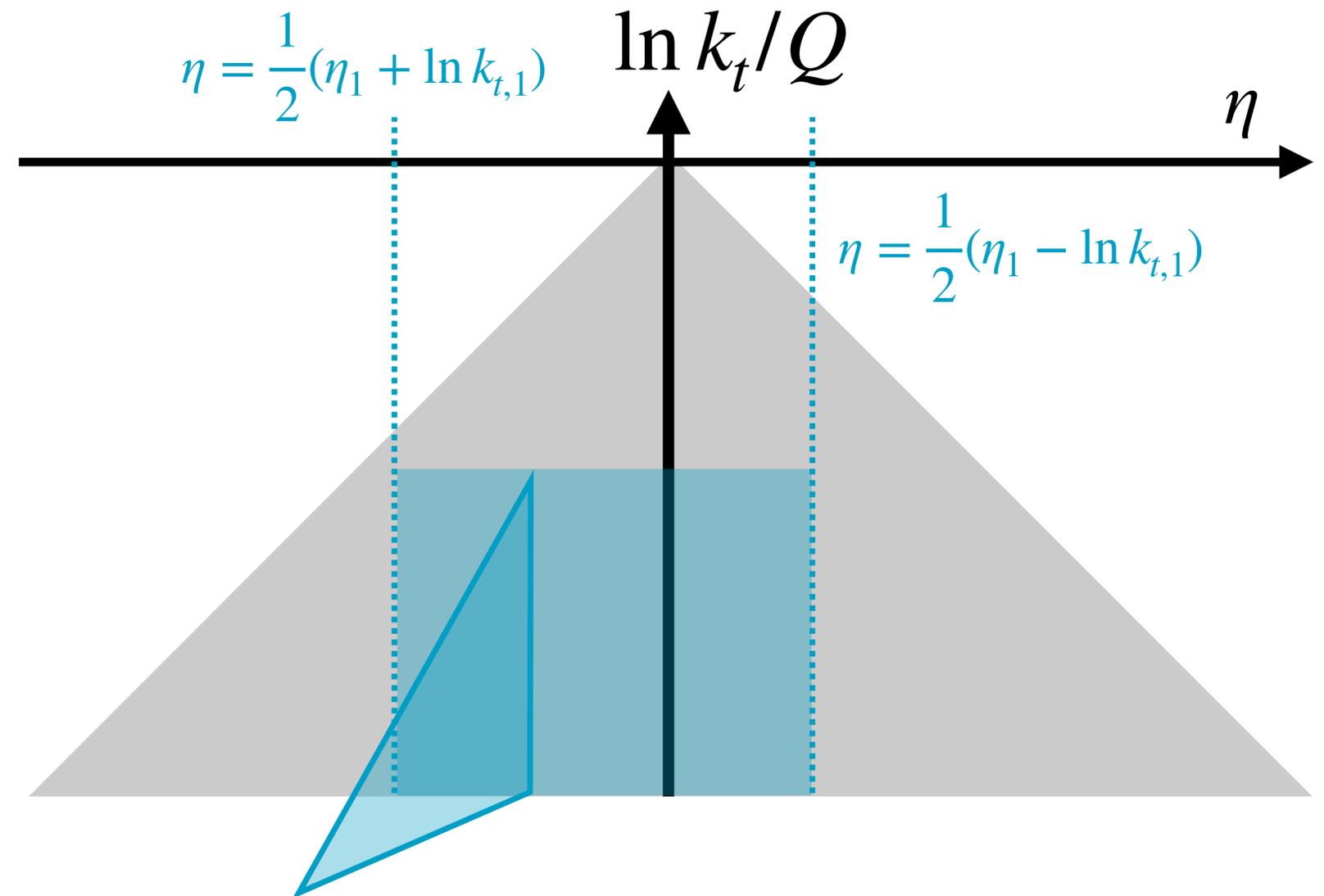


**Issue 2:** in this same region, the new leaf is picked as the emitter  
 → this means it will get a momentum modification

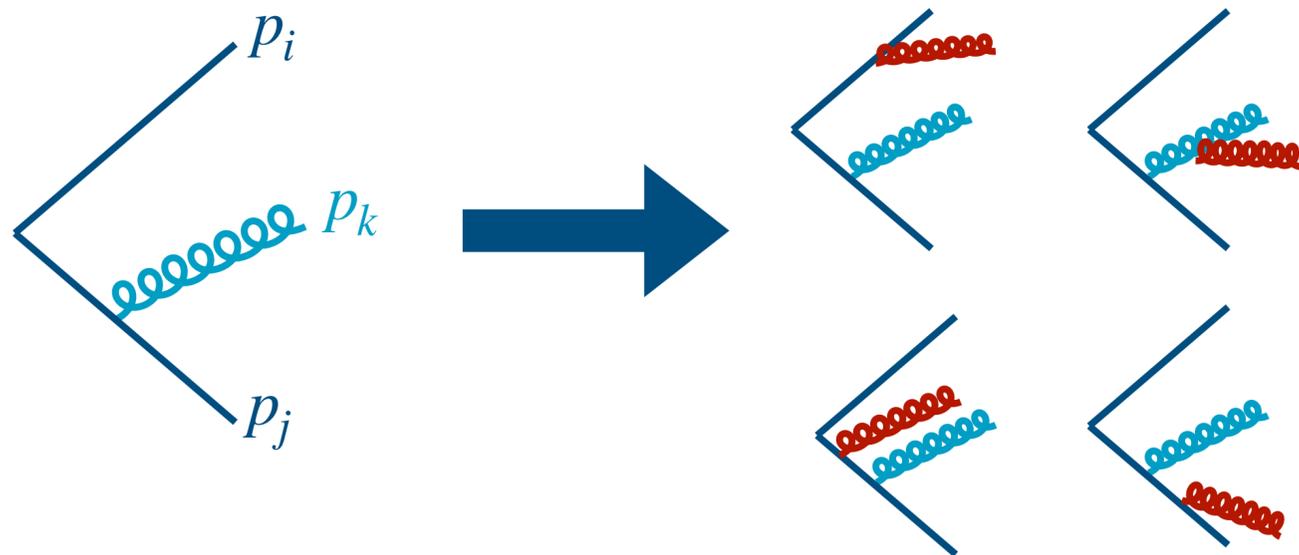
Old emission:  $p_k = a_i \tilde{p}_k + b_i \tilde{p}_j + k_\perp$

New emission:  $p_{k'} = a_k \tilde{p}_k + b_k \tilde{p}_j - k_\perp$

Spectator:  $p_j = (1 - b_j) \tilde{p}_j$



# Momentum mapping

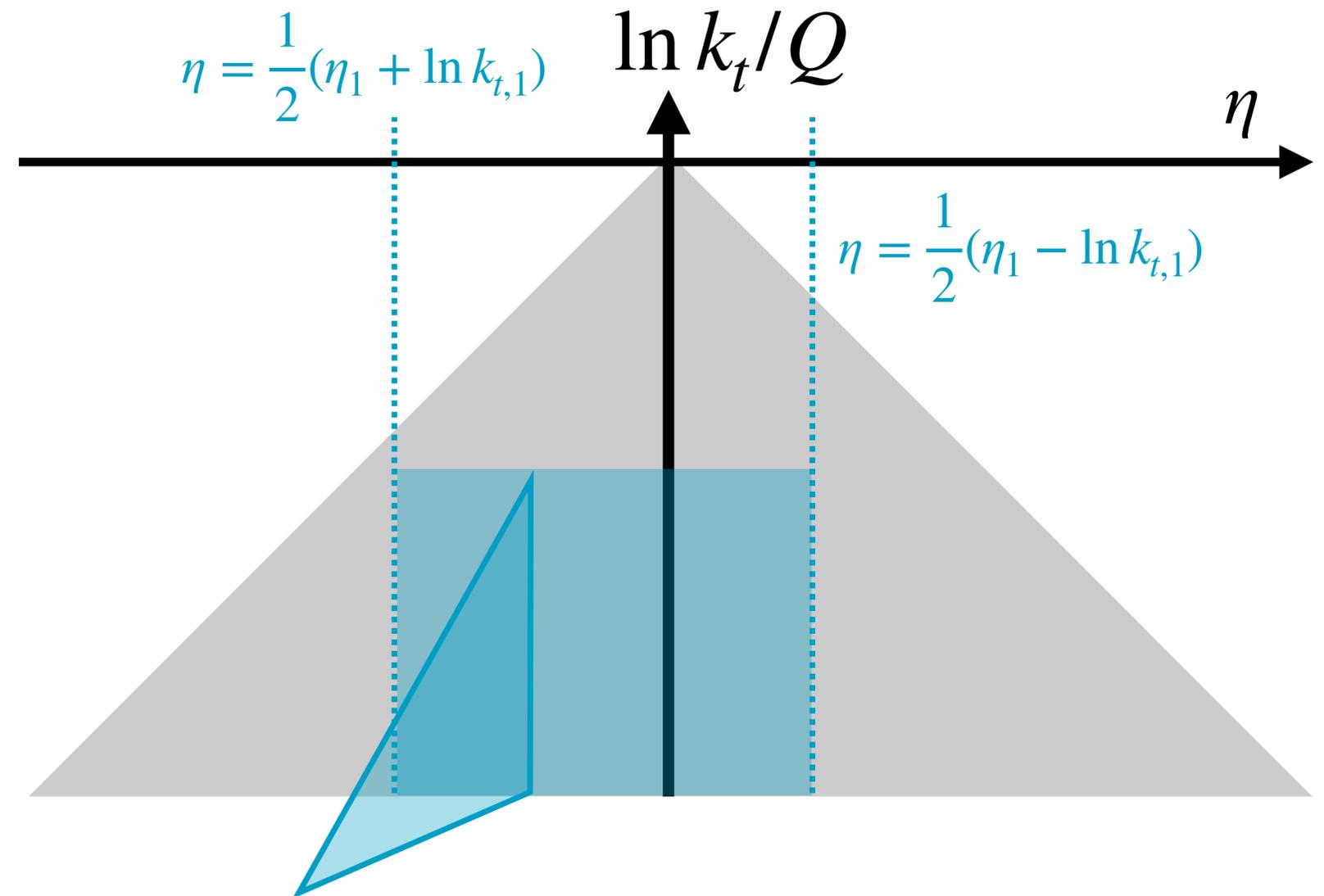


**Issue 2:** in this same region, the new leaf is picked as the emitter  
 → this means it will get a momentum modification

Old emission:  $p_k = a_i \tilde{p}_k + b_i \tilde{p}_j + k_\perp$

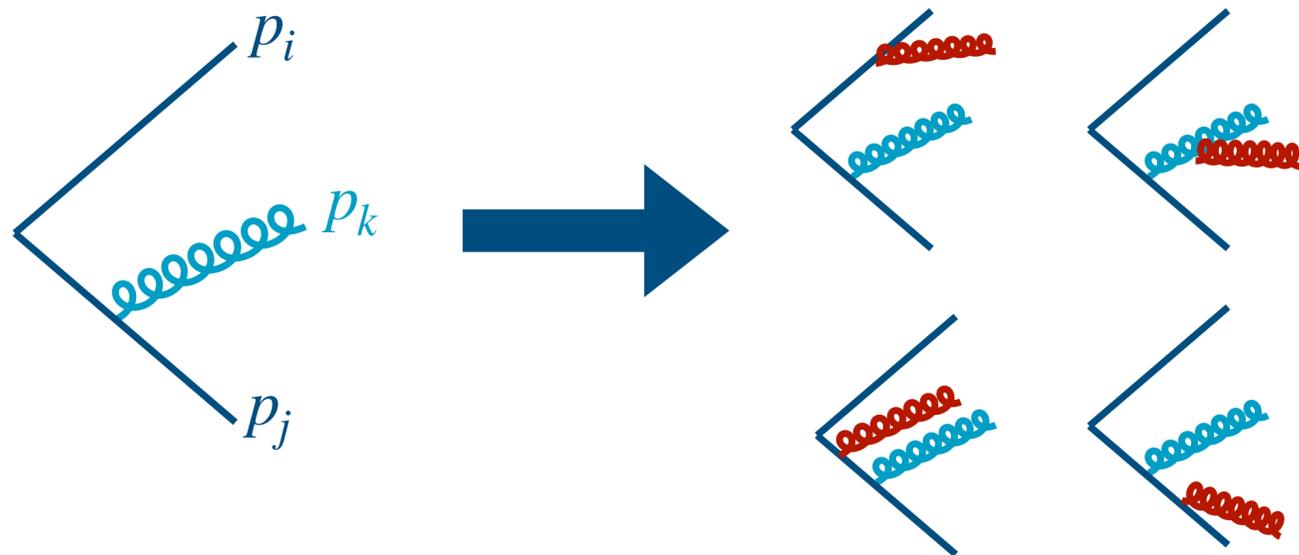
New emission:  $p_{k'} = a_k \tilde{p}_k + b_k \tilde{p}_j - k_\perp$

Spectator:  $p_j = (1 - b_j) \tilde{p}_j$



This is OK in principle, but not if  $p_k$  and  $p_{k'}$  are **not close** in either angle or  $k_t$   
 → QCD tells us that emissions should be independent there

# Momentum mapping



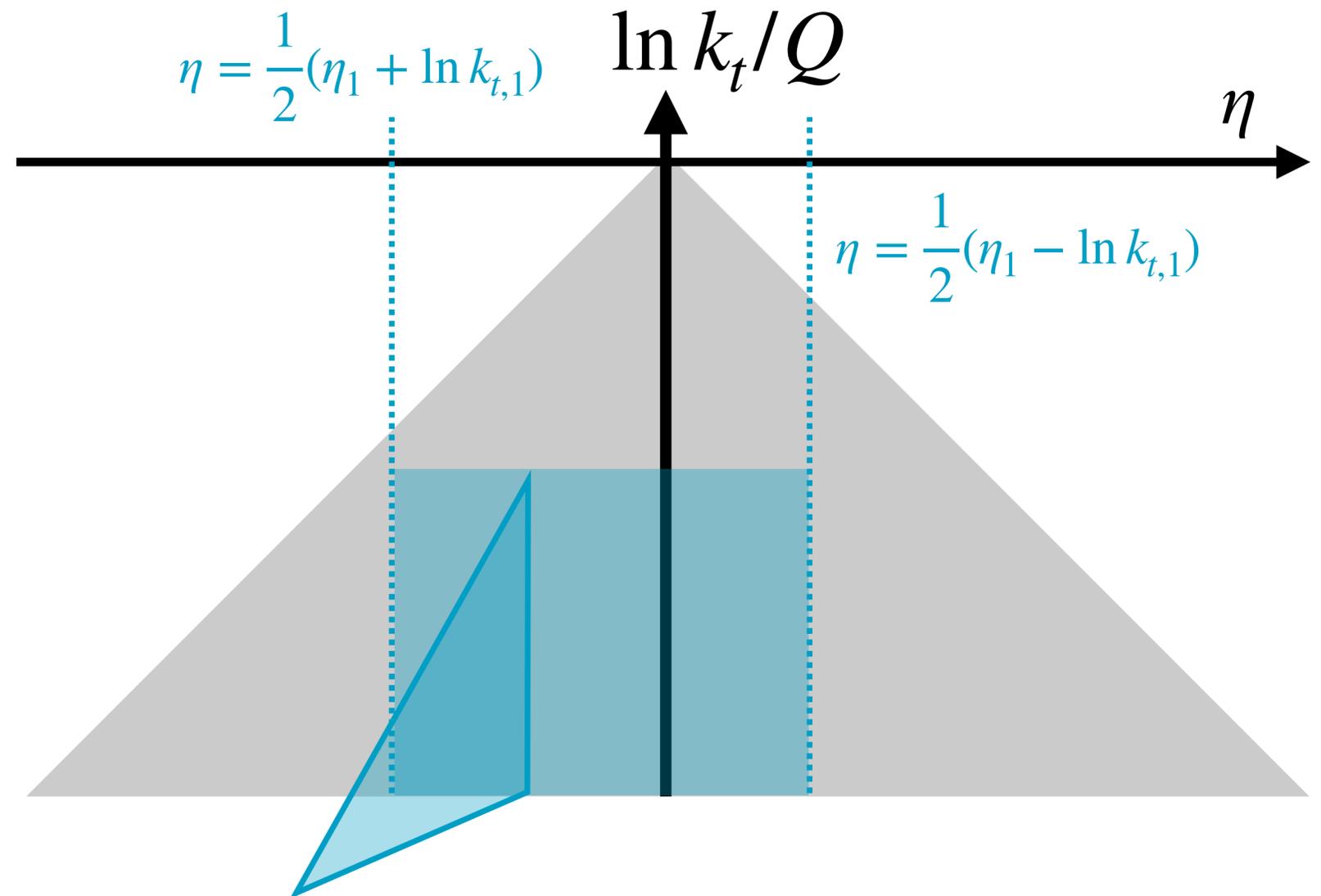
**Issue 2:** in this same region, the new leaf is picked as the emitter  
 → this means it will get a momentum modification

Old emission:  $p_k = a_i \tilde{p}_k + b_i \tilde{p}_j + k_\perp$

New emission:  $p_{k'} = a_k \tilde{p}_k + b_k \tilde{p}_j - k_\perp$

Spectator:  $p_j = (1 - b_j) \tilde{p}_j$

This breaks NLL accuracy for global event shapes at leading  $N_c$



This is OK in principle, but not if  $p_k$  and  $p_{k'}$  are not close in either angle or  $k_t$   
 → QCD tells us that emissions should be independent there

# NLL accuracy is becoming the new standard

## Logarithmic accuracy of parton showers: a fixed-order study

Dasgupta, Dreyer, Hamilton, Monni, Salam [1805.09327]

## Parton showers beyond leading logarithmic accuracy

Dasgupta, Dreyer, Hamilton, Monni, Salam, Soyez [2002.11114]

## A new approach to color-coherent parton evolution

Herren, Höche, Krauss, Reichelt, Schönherr [2208.06057]

## New approach to QCD final-state evolution in Alaric processes with massive partons

Assi, Höche [2307.00728]

## The Alaric parton shower for hadron colliders

Höche, Krauss, Reichelt [2404.14360]

## A partitioned dipole-antenna shower with improved transverse recoil Apollo

Preuss [2403.19452]

## Summation of large logarithms by parton showers Deductor

Nagy, Soper [2011.04773]

## Summation by parton showers of large logarithms in electron-positron annihilation

Nagy, Soper [2011.04777]

## Initial state radiation in the Herwig 7 angular-ordered parton shower Herwig

Bewick, Ferrario Ravasio, Richardson, Seymour [2107.04051]

## Spin correlations in final-state parton showers and jet observables

Karlberg, Salam, Scyboz, Verheyen [2103.16526]

## PanScales parton showers for hadron collisions: formulation and fixed-order studies

van Beekveld, Ferrario Ravasio, Salam, Soto Ontoso, Soyez, Verheyen [2205.02237]

## Next-to-leading-logarithmic PanScales showers for deep inelastic scattering and vector boson fusion

van Beekveld, Ferrario Ravasio [2305.08645]

PanScales

## Soft spin correlations in final-state parton showers

Hamilton, Karlberg, Salam, Scyboz, Verheyen [2111.01161]

## PanScales parton showers for hadron collisions: all-order validation

van Beekveld, Ferrario Ravasio, Hamilton, Salam, Soto Ontoso, Soyez, Verheyen [2207.09467]

## Introduction to the PanScales framework, version 0.1

van Beekveld, Dasgupta, El-Menoufi, Ferrario Ravasio, Hamilton, Helliwell, Karlberg, Medves, Monnim Salam, Scyboz, Soto Ontoso, Soyez, Verheyen [2312.13275]

## Building a consistent parton shower

Forshaw, Holquin, Plätzer [2003.06400]

## Improvements on dipole shower colour

Forshaw, Holquin, Plätzer [2011.15087]

## Logarithmic accuracy of angular-ordered parton showers

Bewick, Ferrario Ravasio, Richardson, Seymour [1904.11866]

# NLL accuracy for event shapes

To get to NLL accuracy for event shapes one needs three ingredients:

- Soft-collinear, wide-soft and hard-collinear matrix elements (splitting probability)
- A correct scaling of the kinematics of emissions (map property)
- A correct treatment of  $\alpha_s$

# Treatment of $\alpha_s$

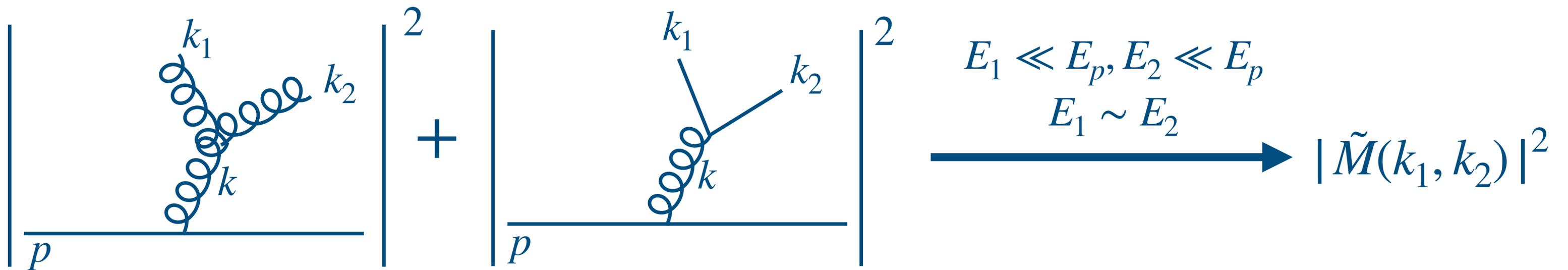
Remember the assumption we made:

1. ME corrections of multiple emissions can be described in terms of one-emission MEs:

$$|M(k_1, \dots, k_n)|^2 = |M(k_1)|^2 \dots |M(k_n)|^2$$

This is not true when (e.g.) emissions are commensurate in energy

→ in this case one needs to consider a ‘correlated block’ of emissions



# Treatment of $\alpha_s$

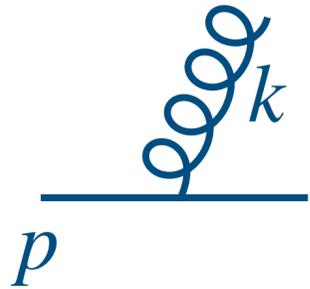
Summing this together with the virtuals:

$$\int [dk] |M^{(0)}(k) + M^{(1)}(k)|^2 + \int [dk_1][dk_2] |\tilde{M}^{(0)}(k_1, k_2)|^2 \delta^{(2)}(k_T - k_{t,1} - k_{t,2}) \delta(y - y_{12})$$

# Treatment of $\alpha_s$

Summing this together with the virtuals:

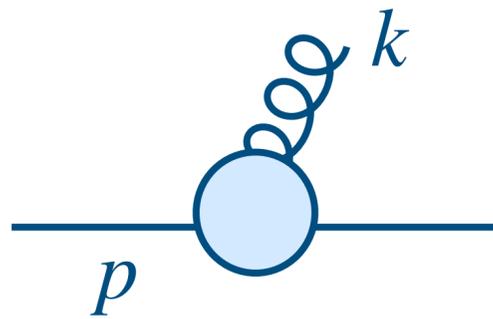
$$\int [dk] |M^{(0)}(k) + M^{(1)}(k)|^2 + \int [dk_1][dk_2] |\tilde{M}^{(0)}(k_1, k_2)|^2 \delta^{(2)}(k_T - k_{t,1} - k_{t,2}) \delta(y - y_{12})$$



# Treatment of $\alpha_s$

Summing this together with the virtuals:

$$\int [dk] |M^{(0)}(k) + M^{(1)}(k)|^2 + \int [dk_1][dk_2] |\tilde{M}^{(0)}(k_1, k_2)|^2 \delta^{(2)}(k_T - k_{t,1} - k_{t,2}) \delta(y - y_{12})$$

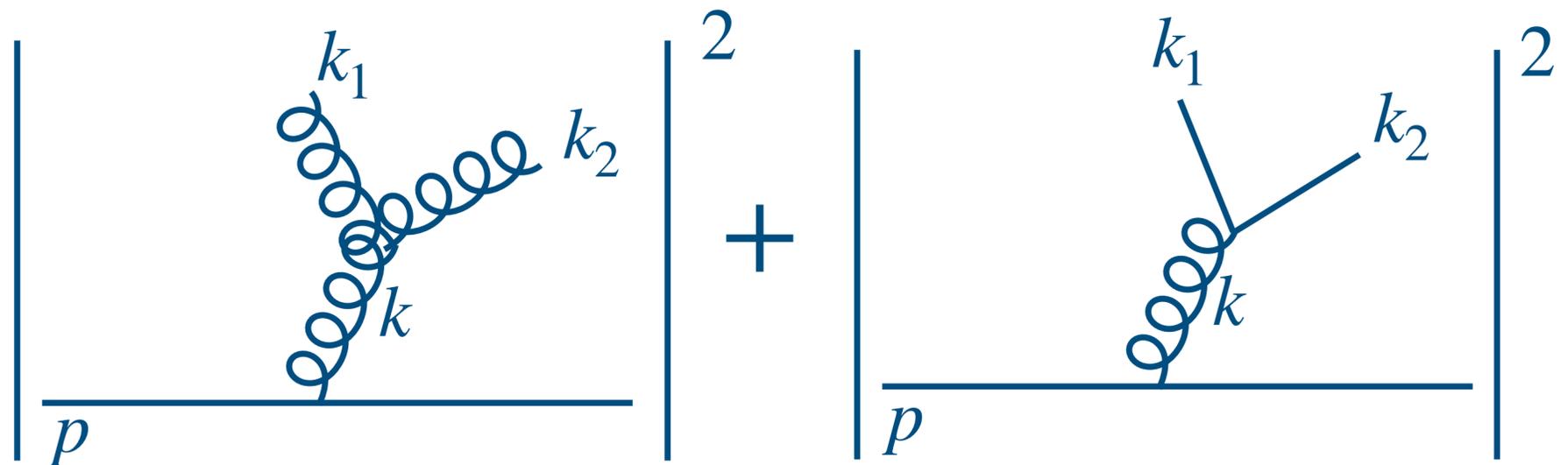


# Treatment of $\alpha_s$

Summing this together with the virtuals:

$$\int [dk] |M^{(0)}(k) + M^{(1)}(k)|^2 + \int [dk_1][dk_2] |\tilde{M}^{(0)}(k_1, k_2)|^2 \delta^{(2)}(k_T - k_{t,1} - k_{t,2}) \delta(y - y_{12})$$

integrate over  $k_1$  and  $k_2$  such that the rapidity and transverse momentum of  $k$  is held fixed



# Treatment of $\alpha_s$

Summing this together with the virtuals:

$$\int [dk] |M^{(0)}(k) + M^{(1)}(k)|^2 + \int [dk_1][dk_2] |\tilde{M}^{(0)}(k_1, k_2)|^2 \delta^{(2)}(k_T - k_{t,1} - k_{t,2}) \delta(y - y_{12})$$

$$= \int [dk] \frac{\alpha_s(\mu)}{2\pi} |M^{(0)}(k)|^2 \left( 1 + \alpha_s(\mu) \left( \beta_0 \ln \frac{k_t^2}{\mu^2} + \frac{K}{2\pi} \right) \right)$$

# Treatment of $\alpha_s$

Summing this together with the virtuals:

$$\int [dk] |M^{(0)}(k) + M^{(1)}(k)|^2 + \int [dk_1][dk_2] |\tilde{M}^{(0)}(k_1, k_2)|^2 \delta^{(2)}(k_T - k_{t,1} - k_{t,2}) \delta(y - y_{12})$$

gives an LL contribution

→ absorb in  $\alpha_s(\mu = k_t)$

$$= \int [dk] \frac{\alpha_s(\mu)}{2\pi} |M^{(0)}(k)|^2 \left( 1 + \alpha_s(\mu) \left( \beta_0 \ln \frac{k_t^2}{\mu^2} + \frac{K}{2\pi} \right) \right)$$

# Treatment of $\alpha_s$

Summing this together with the virtuals:

$$\int [dk] |M^{(0)}(k) + M^{(1)}(k)|^2 + \int [dk_1][dk_2] |\tilde{M}^{(0)}(k_1, k_2)|^2 \delta^{(2)}(k_T - k_{t,1} - k_{t,2}) \delta(y - y_{12})$$

gives an LL contribution

→ absorb in  $\alpha_s(\mu = k_t)$

$$= \int [dk] \frac{\alpha_s(\mu)}{2\pi} |M^{(0)}(k)|^2 \left( 1 + \alpha_s(\mu) \left( \beta_0 \ln \frac{k_t^2}{\mu^2} + \frac{K}{2\pi} \right) \right)$$

gives an NLL contribution

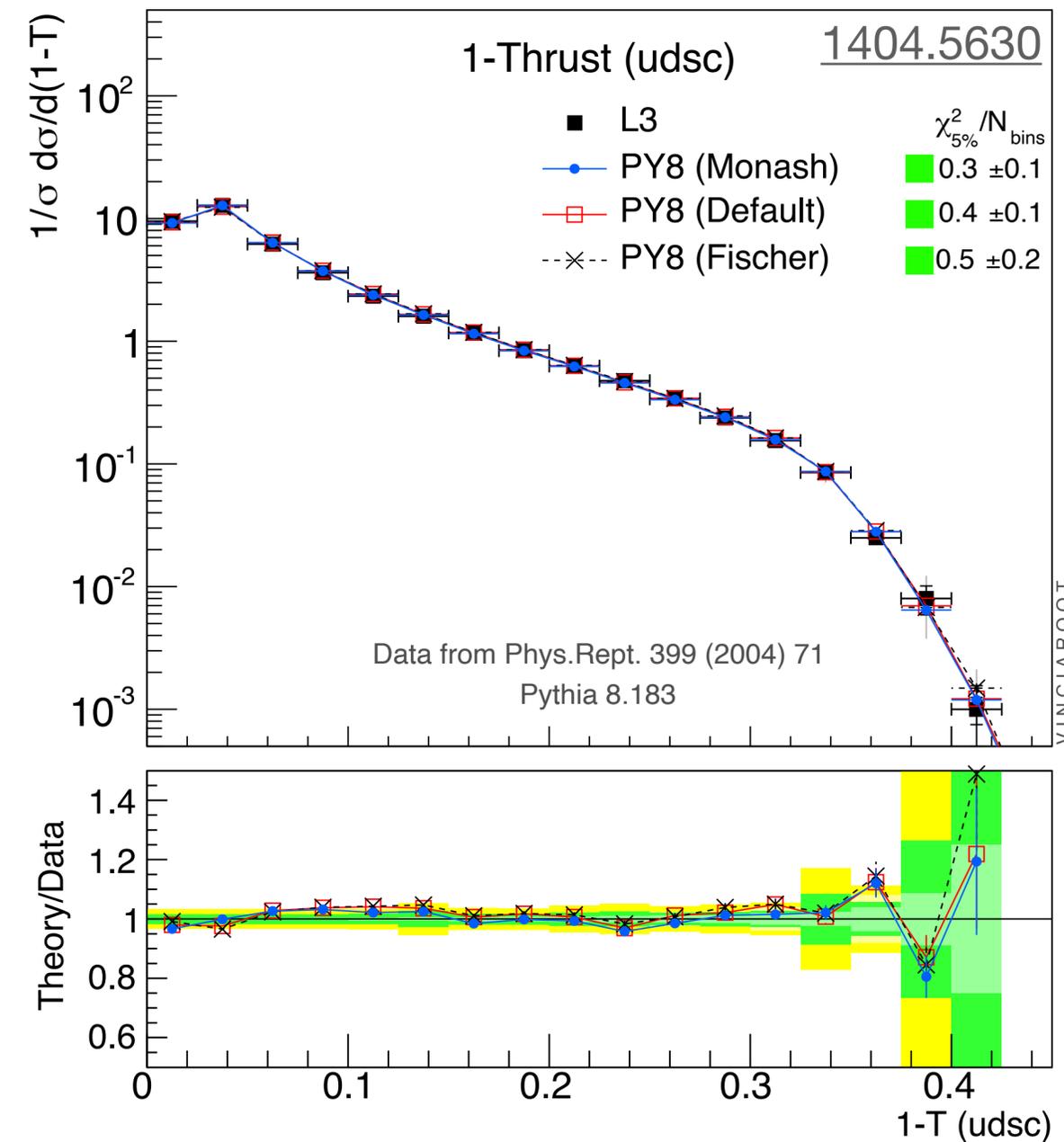
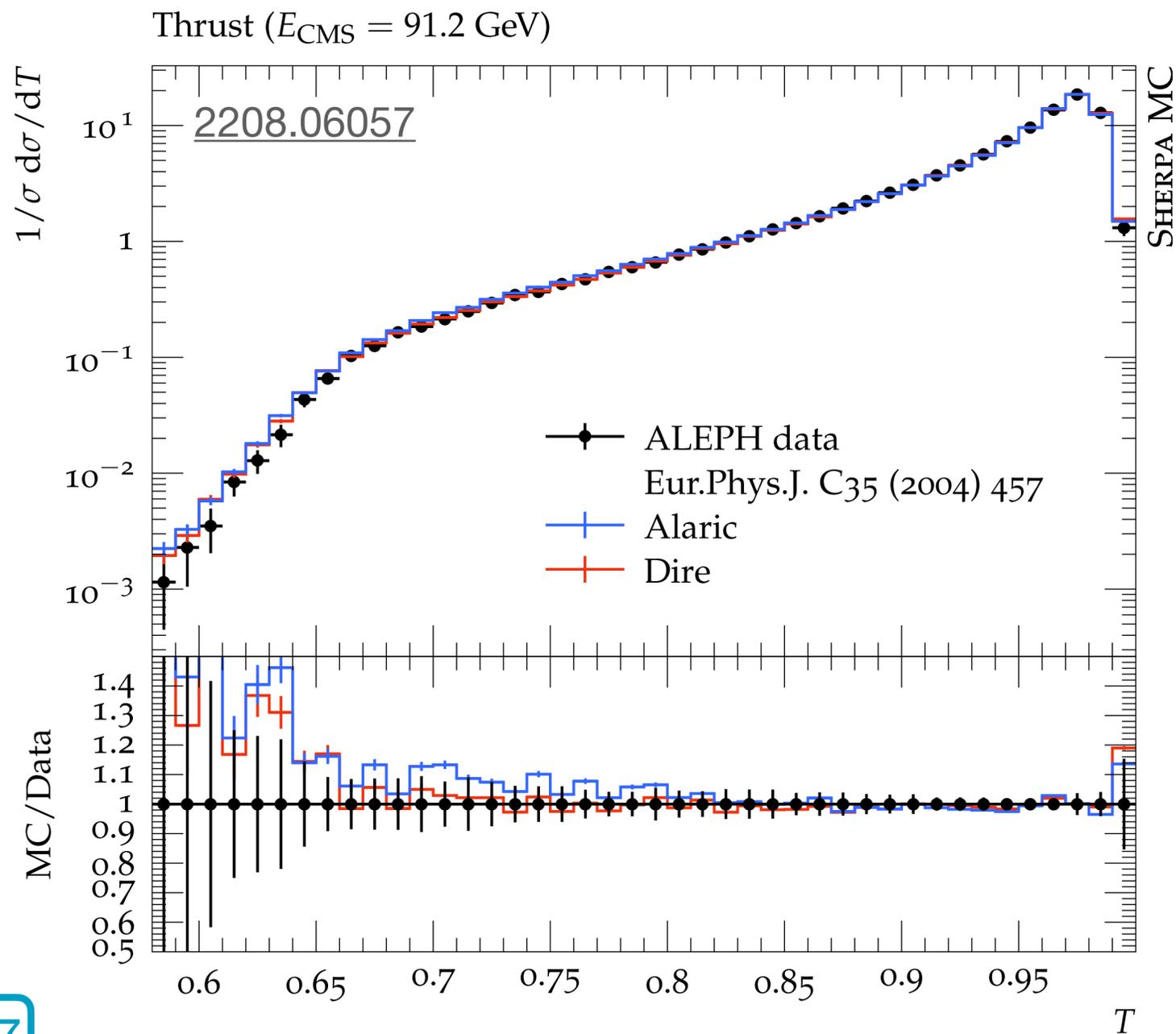
→ absorb in  $\alpha_s^{\text{CMW}} \equiv \alpha_s(k_t) \left( 1 + K/(2\pi) \right)$

The ‘CMW’ scheme for  $\alpha_s$  is commonly used in showers, crucial (but not sufficient) for the shower to be NLL accurate

# After all this work, what do we get?

and some more...

Ability to compare with data, e.g. for  $e^+e^-$  thrust

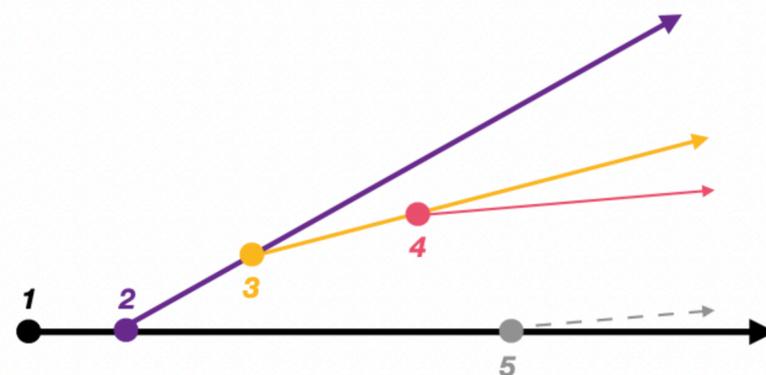
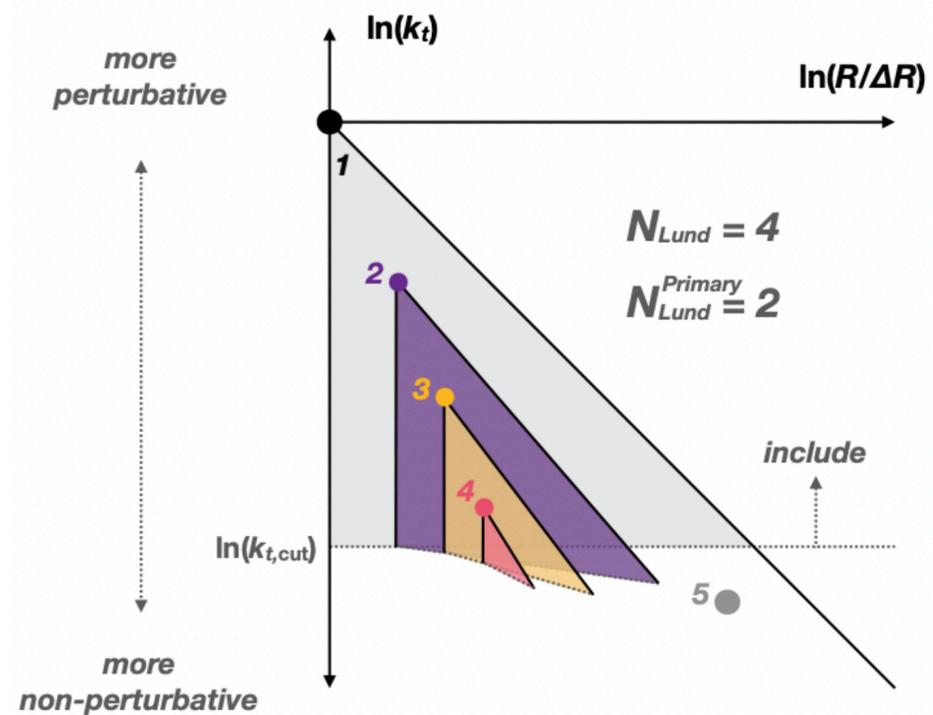


# After all this work, what do we get?

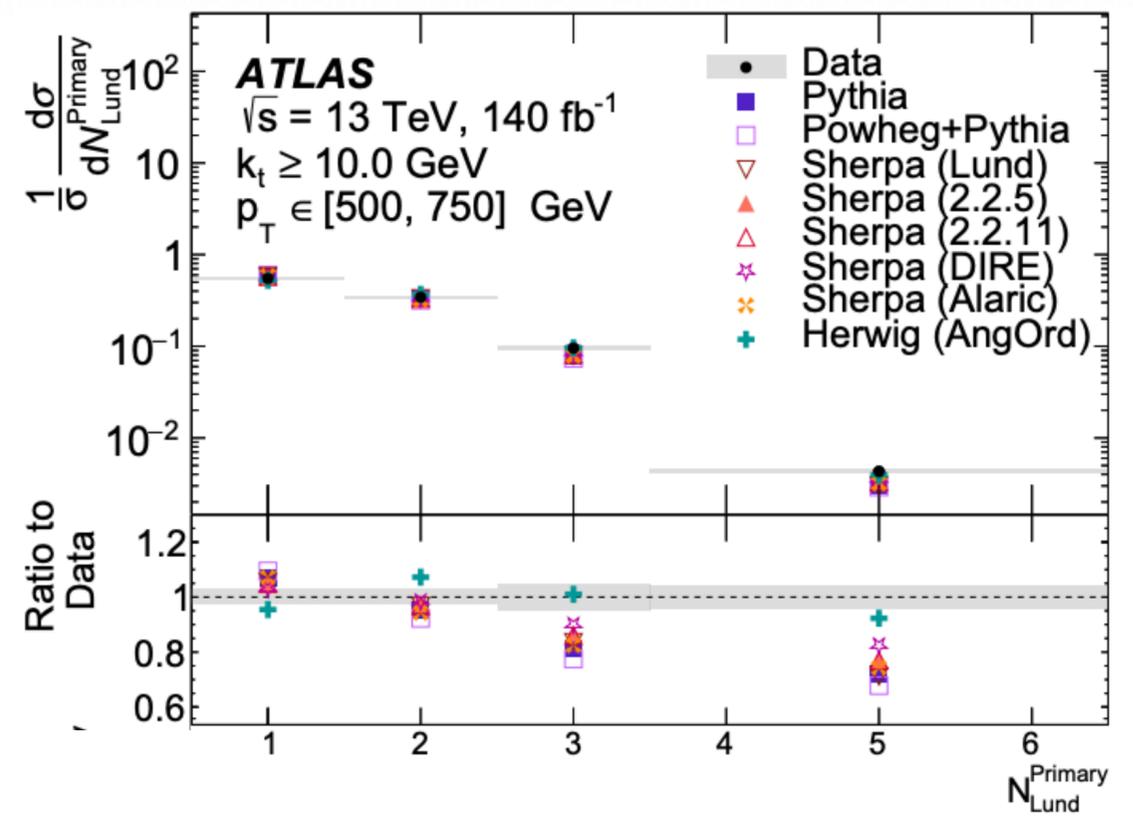
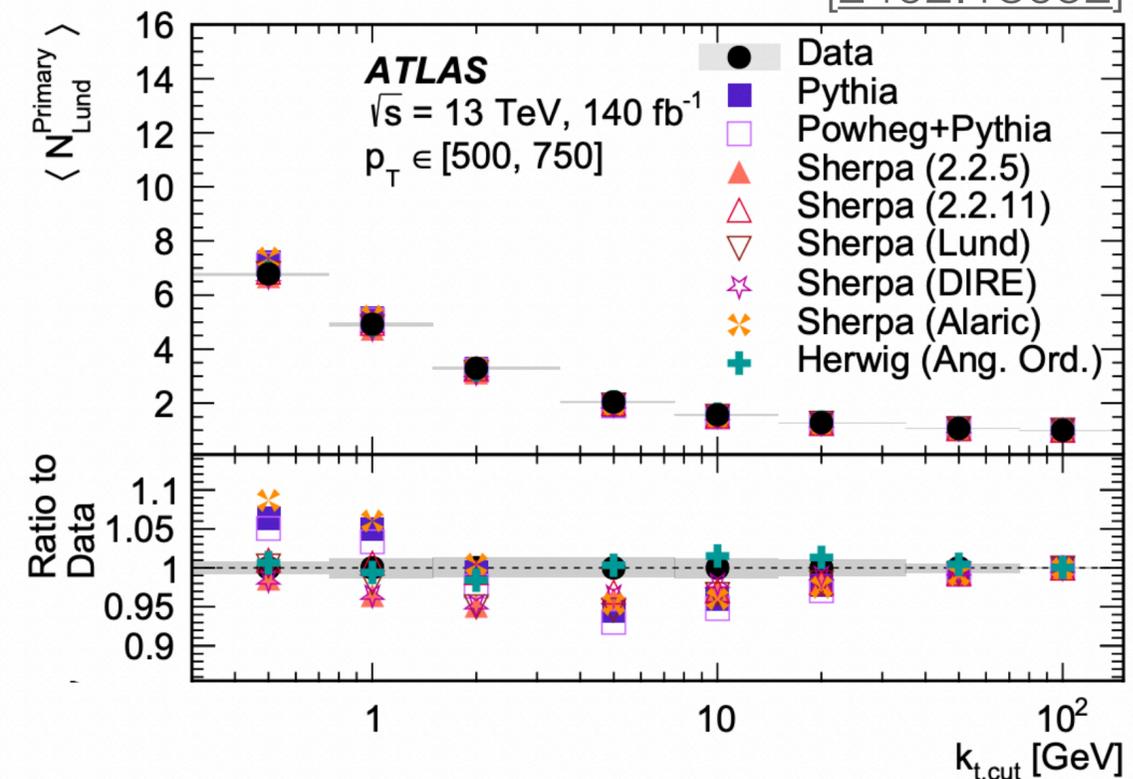
and some more...

But it does not always work so well...

Consider measurement of Lund (sub)jet multiplicity



[2402.13052]



# Other important ingredients and developments

- Initial-state radiation
  - Parton masses
  - Going beyond leading-color approximation
    - Both fixes to dipole shower picture...
    - ... but also full amplitude evolution
  - Including spin correlations
  - Going beyond NLL
  - Beyond QCD: QED and EW showers
  - Matching to higher-order predictions
  - Merging multiple born states after showering
- see Stefan's lectures

# Conclusions

In these lectures I've tried to give you an overview of the QCD ingredients that enter a shower algorithm

Even though we have been 'showering' for more than four decades, there is still a lot to be understood, not in the least driven by important theoretical developments on higher-order calculations

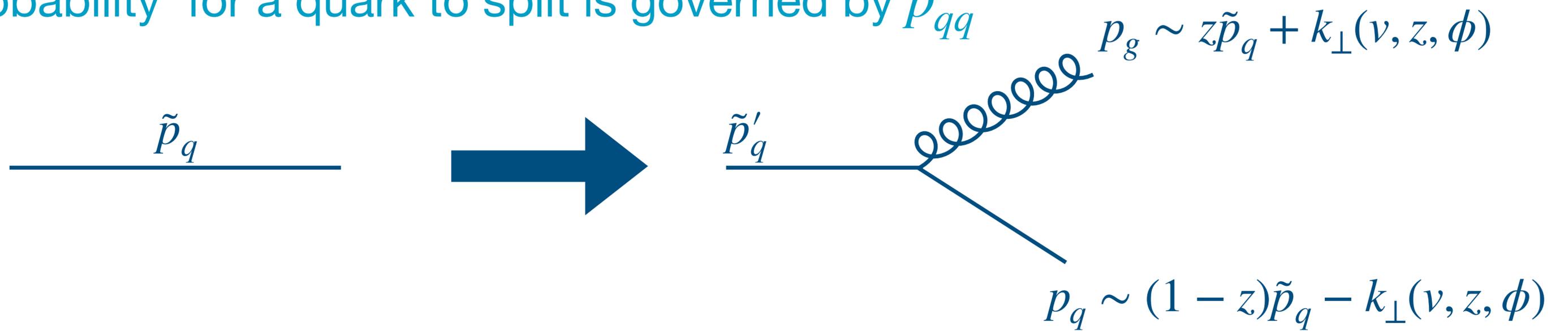
But also non-perturbative aspects play a role in the theory-data comparison for some (IR unsafe) observables, another game entirely

Importantly: showers (as part as MC event generators) enable everyone to create their own theoretical predictions for their favorite observables!

Backup on veto algorithm

# Probability to generate an emission

- The 'probability' for a quark to split is governed by  $P_{qq}$



# Probability to generate an emission

- The 'probability' for a quark to split is governed by  $p_{qq}$

$$\text{Total probability: } p_{1 \rightarrow 2} = \int_{z_-}^{z_+} dz \frac{\alpha_s}{2\pi} P_{qq}(z)$$

- The probability to split in a small evolution interval is  $p_{1 \rightarrow 2} \delta v$
- The probability not to split in a small evolution interval is  $p_{\text{nosplit}} = 1 - p_{1 \rightarrow 2} \delta v$

# Probability to generate an emission

- The 'probability' for a quark to split is governed by  $P_{qq}$

$$\text{Total probability: } p_{1 \rightarrow 2} = \int_{z_-}^{z_+} dz \frac{\alpha_s}{2\pi} P_{qq}(z)$$

- The probability to split in a small evolution interval is  $p_{1 \rightarrow 2} \delta v$
- The probability not to split in a small evolution interval is  $p_{\text{nosplit}} = 1 - p_{1 \rightarrow 2} \delta v$

→ integrating over a larger interval  $\Delta v$  gives

$$p_{\text{nosplit}}(v_0, v) = \lim_{n \rightarrow \infty} p_{\text{nosplit}}(v_0, v_0 + \delta v) \cdot p_{\text{nosplit}}(v_0 + \delta v, v_0 + 2\delta v) \cdot \dots \cdot p_{\text{nosplit}}(v_0 + n\delta v, v)$$

# Probability to generate an emission

- The 'probability' for a quark to split is governed by  $P_{qq}$

$$\text{Total probability: } p_{1 \rightarrow 2} = \int_{z_-}^{z_+} dz \frac{\alpha_s}{2\pi} P_{qq}(z)$$

- The probability to split in a small evolution interval is  $p_{1 \rightarrow 2} \delta v$
- The probability not to split in a small evolution interval is  $p_{\text{nosplit}} = 1 - p_{1 \rightarrow 2} \delta v$

→ integrating over a larger interval  $\Delta v$  gives

$$\begin{aligned} p_{\text{nosplit}}(v_0, v) &= \lim_{n \rightarrow \infty} p_{\text{nosplit}}(v_0, v_0 + \delta v) \cdot p_{\text{nosplit}}(v_0 + \delta v, v_0 + 2\delta v) \cdot \dots \cdot p_{\text{nosplit}}(v_0 + n\delta v, v) \\ &= \lim_{n \rightarrow \infty} \left( 1 + \frac{1}{n} \int_{v_0}^v dv' p_{1 \rightarrow 2} \right)^n \end{aligned}$$

# Probability to generate an emission

- The 'probability' for a quark to split is governed by  $P_{qq}$

$$\text{Total probability: } p_{1 \rightarrow 2} = \int_{z_-}^{z_+} dz \frac{\alpha_s}{2\pi} P_{qq}(z)$$

- The probability to split in a small evolution interval is  $p_{1 \rightarrow 2} \delta v$
- The probability not to split in a small evolution interval is  $p_{\text{nosplit}} = 1 - p_{1 \rightarrow 2} \delta v$

→ integrating over a larger interval  $\Delta v$  gives

$$\begin{aligned} p_{\text{nosplit}}(v_0, v) &= \lim_{n \rightarrow \infty} p_{\text{nosplit}}(v_0, v_0 + \delta v) \cdot p_{\text{nosplit}}(v_0 + \delta v, v_0 + 2\delta v) \cdot \dots \cdot p_{\text{nosplit}}(v_0 + n\delta v, v) \\ &= \lim_{n \rightarrow \infty} \left( 1 + \frac{1}{n} \int_{v_0}^v dv' p_{1 \rightarrow 2} \right)^n = \exp \left[ - \int_{v_0}^v dv' p_{1 \rightarrow 2} \right] = \Delta(v_0, v) \end{aligned}$$

this is called the Sudakov, or no-branching probability

# Probability to generate an emission

- The probability of branching at a scale  $\nu$  is given by the probability of branching times the probability that the quark did not branch before

$$p_{\text{branch}}(\nu) = p_{1 \rightarrow 2}(\nu) \Delta(\nu_0, \nu)$$

# Probability to generate an emission

- The probability of branching at a scale  $\nu$  is given by the probability of branching times the probability that the quark did not branch before

$$P_{\text{branch}}(\nu) = p_{1 \rightarrow 2}(\nu) \Delta(\nu_0, \nu)$$

- $\nu$  is an evolution variable that orders the sequence of emissions

# Probability to generate an emission

- The probability of branching at a scale  $\nu$  is given by the probability of branching times the probability that the quark did not branch before

$$P_{\text{branch}}(\nu) = p_{1 \rightarrow 2}(\nu) \Delta(\nu_0, \nu)$$

- $\nu$  is an evolution variable that orders the sequence of emissions

Often taken to be related to the transverse momentum of the emission in the collinear limit (Pythia, Sherpa)...

# Probability to generate an emission

- The probability of branching at a scale  $\nu$  is given by the probability of branching times the probability that the quark did not branch before

$$P_{\text{branch}}(\nu) = p_{1 \rightarrow 2}(\nu) \Delta(\nu_0, \nu)$$

- $\nu$  is an evolution variable that orders the sequence of emissions

Often taken to be related to the transverse momentum of the emission in the collinear limit (Pythia, Sherpa)...

...but also angular ordering is a popular choice (Herwig)

# Probability to generate an emission

- The probability of branching at a scale  $\nu$  is given by the probability of branching times the probability that the quark did not branch before

$$P_{\text{branch}}(\nu) = p_{1 \rightarrow 2}(\nu) \Delta(\nu_0, \nu)$$

- $\nu$  is an evolution variable that orders the sequence of emissions

Often taken to be related to the transverse momentum of the emission in the collinear limit (Pythia, Sherpa)...

...but also angular ordering is a popular choice (Herwig)

**This** is the ‘master equation’ of the shower algorithm...

... but how to solve it?

# The Sudakov veto algorithm

- Define  $p_{1 \rightarrow 2}(v) \equiv f(v)$  and  $\Delta(0, v) \equiv \exp \left[ - \int_0^v dv' f(v') \right]$
- Assume we know  $F(v) = \int_0^v dv' f(v')$  and its inverse

# The Sudakov veto algorithm

- Define  $p_{1 \rightarrow 2}(v) \equiv f(v)$  and  $\Delta(0, v) \equiv \exp \left[ - \int_0^v dv' f(v') \right]$
- Assume we know  $F(v) = \int_0^v dv' f(v')$  and its inverse

To then select  $v$  values according to

$$P_{\text{branch}}(v) = p_{1 \rightarrow 2}(v) \Delta(0, v)$$

we can use 'standard' Monte-Carlo techniques:

$$\int_0^v dv' f(v') \Delta(v') = \Delta(0) - \Delta(0, v) = 1 - \exp \left[ -F(v) \right] \equiv 1 - r$$

$$\text{Solve for } v \rightarrow v = F^{-1} \left( F(0) - \ln r \right)$$

```

# Constants
N = 50000
v0 = 1
c = 10.

# define the integrand
def f(v):
    return c/v

# define the integrated function
def integrated_f(v):
    return c*np.log(v)

# define the inverse of the integrated function
def inverse_integrated_f(v):
    return np.exp(v/c)

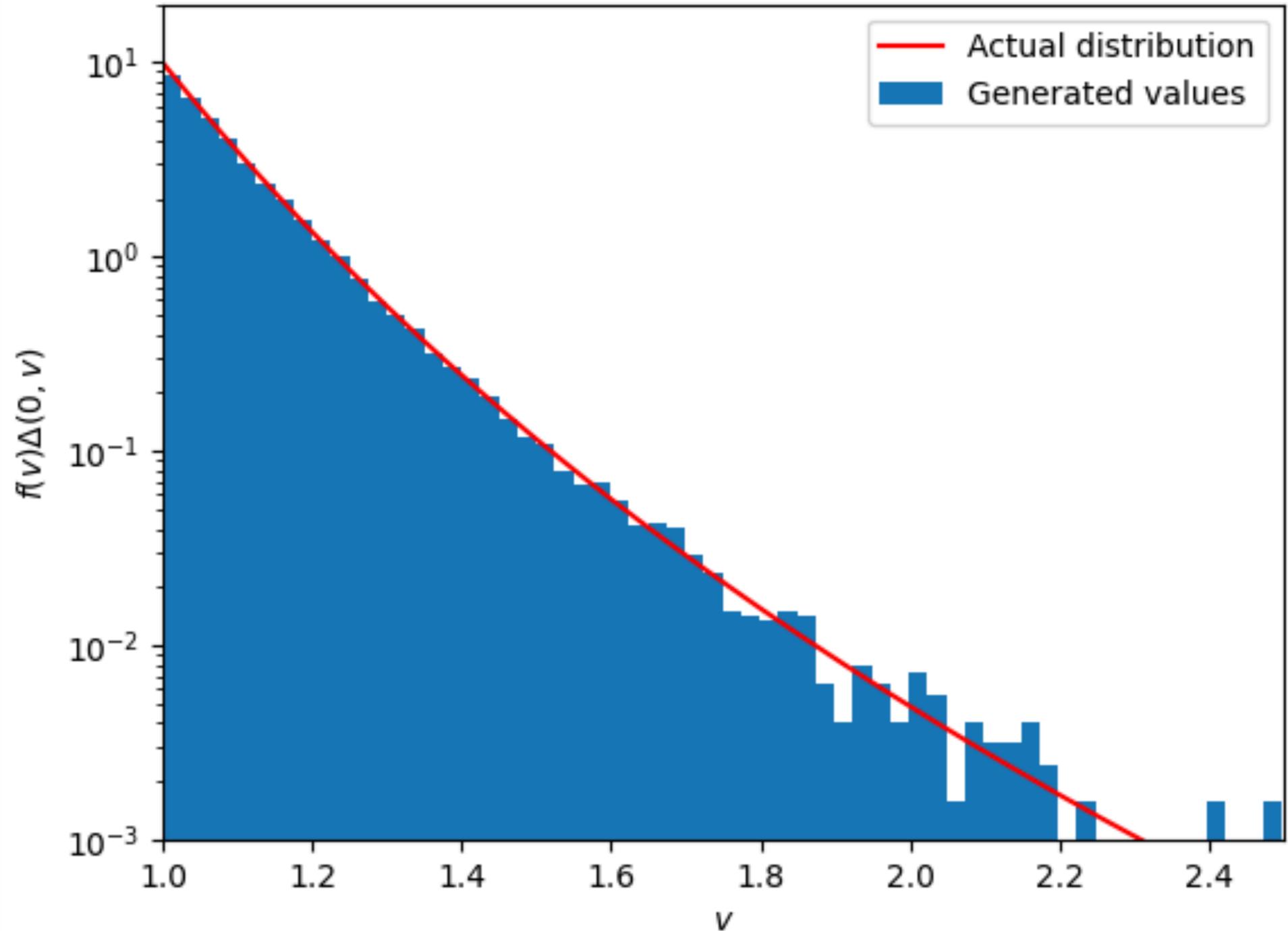
# generate values for v
def generate_v(v0, n_samples):
    v = np.zeros(n_samples)
    for i in range(n_samples):
        r = random.uniform(0,1)
        v[i] = inverse_integrated_f(integrated_f(v0) - np.log(r))
    return v

# generate the values
v = generate_v(v0, N)
# histogram of the generated values
plt.hist(v, bins = 100, density=True)

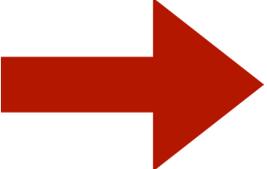
# compare to the target distribution
v = np.linspace(1,20,1000)
plt.plot(v, f(v)*np.exp(-integrated_f(v0)-integrated_f(v)), 'r')

# label the plot
plt.xlabel(r'$v$')
plt.ylabel(r'$f(v)\Delta(0,v)$')
plt.legend(['Generated values', 'Actual distribution'])
# set xrange
plt.xlim([1,2.5])
plt.ylim([1E-3,20])
# set y scale logarithmic
plt.yscale('log')
plt.show()

```



# The Sudakov veto algorithm

- Define  $p_{1 \rightarrow 2}(v) \equiv f(v)$  and  $\Delta(0, v) \equiv \exp \left[ - \int_0^v dv' f(v') \right]$
- Assume we know  $F(v) = \int_0^v dv' f(v')$  and its inverse  This is very often not possible!

To then select  $v$  values according to

$$P_{\text{branch}}(v) = p_{1 \rightarrow 2}(v) \Delta(0, v)$$

we can use 'standard' Monte-Carlo techniques:

$$\int_0^v dv' f(v') \Delta(v') = \Delta(0) - \Delta(0, v) = 1 - \exp \left[ -F(v) \right] \equiv 1 - r$$

$$\text{Solve for } v \rightarrow v = F^{-1} \left( F(0) - \ln r \right)$$

# The Sudakov veto algorithm

The Sudakov veto algorithm:

1. Start with  $i = 0$  and  $v_0 = 0$

# The Sudakov veto algorithm

The Sudakov veto algorithm:

1. Start with  $i = 0$  and  $v_0 = 0$
2.  $i + = 1$ ; select  $v_i = G^{-1}(G(v_i) - \ln(r_1))$  with  $g(v) \geq f(v)$

# The Sudakov veto algorithm

The Sudakov veto algorithm:

1. Start with  $i = 0$  and  $v_0 = 0$
2.  $i + = 1$ ; select  $v_i = G^{-1}(G(v_i) - \ln(r_1))$  with  $g(v) \geq f(v)$
3. Compare  $r_2$  to  $f(v_i)/g(v_i)$ 
  - if  $f(v_i)/g(v_i) \leq r_2$ : return to 2
  - else: accept  $v_i$

# The Sudakov veto algorithm

The Sudakov veto algorithm:

1. Start with  $i = 0$  and  $v_0 = 0$
2.  $i + = 1$ ; select  $v_i = G^{-1}(G(v_i) - \ln(r_1))$  with  $g(v) \geq f(v)$
3. Compare  $r_2$  to  $f(v_i)/g(v_i)$ 
  - if  $f(v_i)/g(v_i) \leq r_2$ : return to 2
  - else: accept  $v_i$
4. Continue until  $v_i$  reaches  $v_{\text{cutoff}}$

# The Sudakov veto algorithm

The Sudakov veto algorithm:

1. Start with  $i = 0$  and  $v_0 = 0$
2.  $i + = 1$ ; select  $v_i = G^{-1}(G(v_i) - \ln(r_1))$  with  $g(v) \geq f(v)$
3. Compare  $r_2$  to  $f(v_i)/g(v_i)$ 
  - if  $f(v_i)/g(v_i) \leq r_2$ : return to 2
  - else: accept  $v_i$
4. Continue until  $v_i$  reaches  $v_{\text{cutoff}}$

One can analytically prove that this gives the right answer by considering all the different ways of selecting a new  $v_i$

# The Sudakov veto algorithm

The Sudakov veto algorithm:

1. Start with  $i = 0$  and  $v_0 = 0$
2.  $i + = 1$ ; select  $v_i = G^{-1}(G(v_i) - \ln(r_1))$  with  $g(v) \geq f(v)$
3. Compare  $r_2$  to  $f(v_i)/g(v_i)$ 
  - if  $f(v_i)/g(v_i) \leq r_2$ : return to 2
  - else: accept  $v_i$
4. Continue until  $v_i$  reaches  $v_{\text{cutoff}}$

The other splitting variables ( $z$  and  $\phi$ ) can then be generated using standard Monte-Carlo techniques

# Selecting $z$

Aim: select  $z$  such that probability to find  $z$  in an interval  $dz$  is  $f(z)dz$

# Selecting $z$

Aim: select  $z$  such that probability to find  $z$  in an interval  $dz$  is  $f(z)dz$

- A fraction  $r$  of total area under  $f(z)$  should be to the left of  $z$

$$\int_{z_{\min}}^z dz' f(z') = r \int_{z_{\min}}^{z_{\max}} dz' f(z')$$

$$\text{Solve for } z \rightarrow z = F^{-1}(F(z_{\min}) + r(F(z_{\max}) - F(z_{\min})))$$

# Selecting $z$

Aim: select  $z$  such that probability to find  $z$  in an interval  $dz$  is  $f(z)dz$

- A fraction  $r$  of total area under  $f(z)$  should be to the left of  $z$

$$\int_{z_{\min}}^z dz' f(z') = r \int_{z_{\min}}^{z_{\max}} dz' f(z')$$

$$\text{Solve for } z \rightarrow z = F^{-1}(F(z_{\min}) + r(F(z_{\max}) - F(z_{\min})))$$

- This assumes everything is known analytically  $\rightarrow$  if not we can use the same ‘overestimate’ trick as before

Also e.g. multiple splitting channels ( $g \rightarrow gg, g \rightarrow q\bar{q}$ )  
and running  $\alpha_s$  effects can be accounted for this way