B-tagging Performance based on Boosted Decision Trees

Hai-Jun Yang
University of Michigan
(with X. Li and B. Zhou)

ATLAS B-tagging Meeting
February 9, 2009

Motivation

- To evaluate the performance of Boosted Decision Trees by combining the information from different ATLAS b-tagging algorithms into a single discriminator for jet classification
- Ref: J. Bastos, ATL-PHYS-PUB-2007-019

- To improve the b-tagging performance

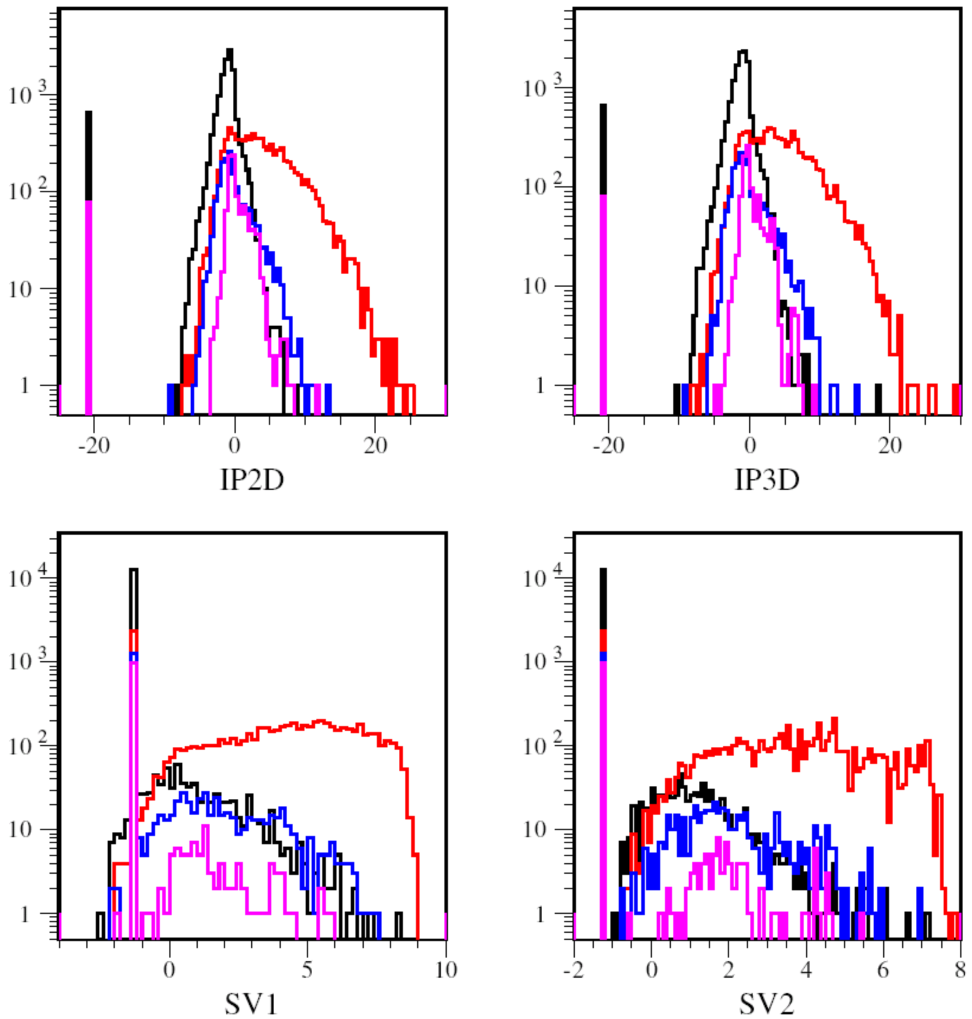
Data Samples

- Ttbar (DS5200, 387K) based on release v13.
- Preselection cuts: Et (jet) > 15 GeV, $|\eta|<2.5$

Jets	Total Jets	For training	For testing
Total No. of Jets	1906314	476580	1429734
B jets (33.8%)	643683	160921	482762
C jets (7.3%)	139246	34812	104434
τ jets (4.4%)	83457	20865	62592
Light jets(54.5%)	1039928	259982	779946

19 Variables for BDT Training

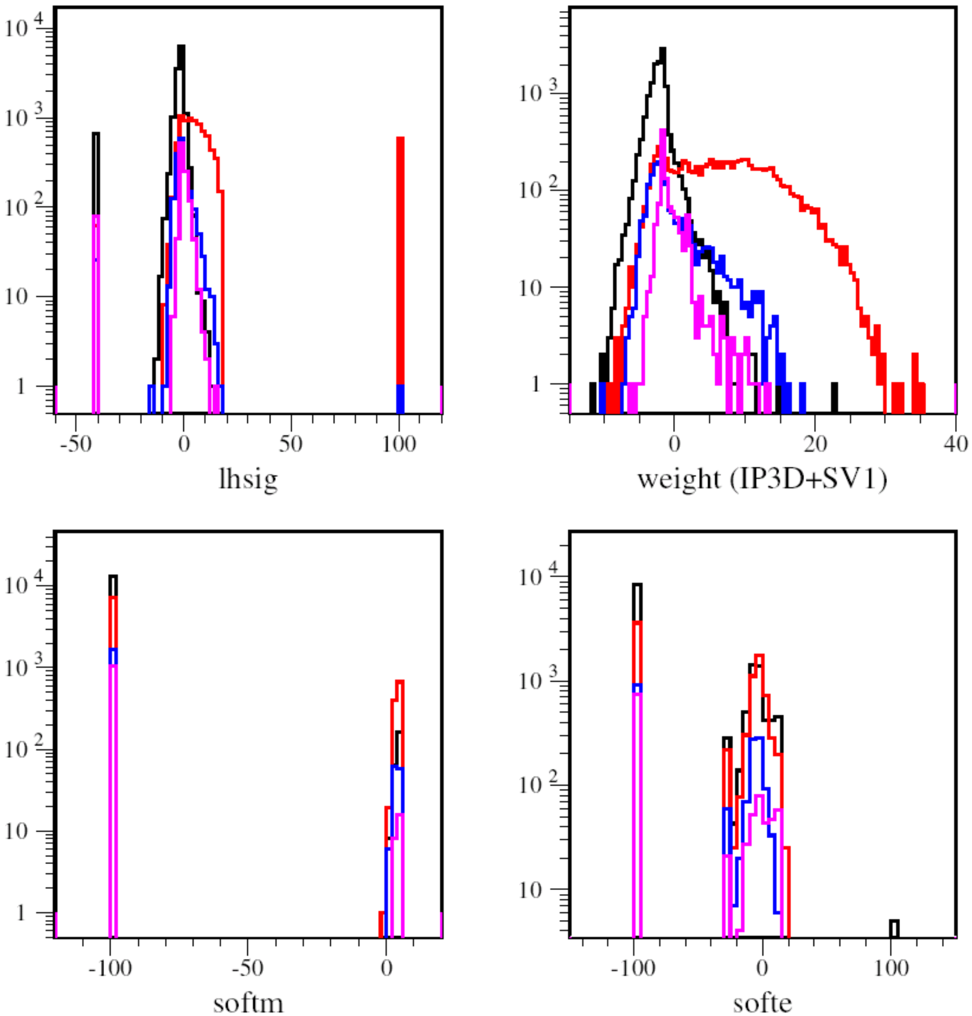
B-jet(red), C-jet(blue), τ-jet(pink), Light-jet(black)



- IP2D – jet weight from transverse impact parameters
- IP3D – jet weight from 3D impact parameters
- SV1 – jet weigh from secondary vertices
- SV2 – jet weight from secondary vertices

Input Variables

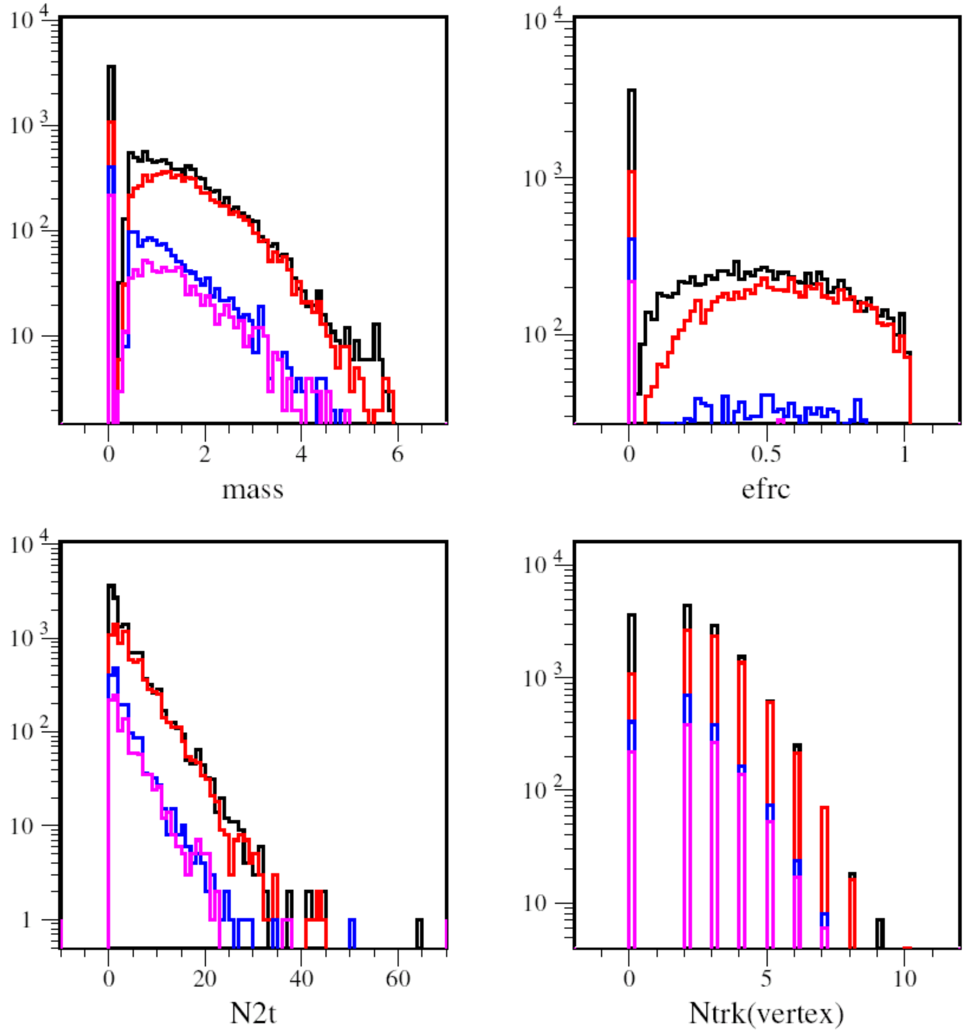
B-jet(red), C-jet(blue), τ-jet(pink), Light-jet(black)



- lhsig – combination of jet weights IP1D, IP2D & SVBU
- weight – combination of jet weights IP3D and SV1
- softm – soft muon based tagger
- softe – soft electron based tagger

Input Variables

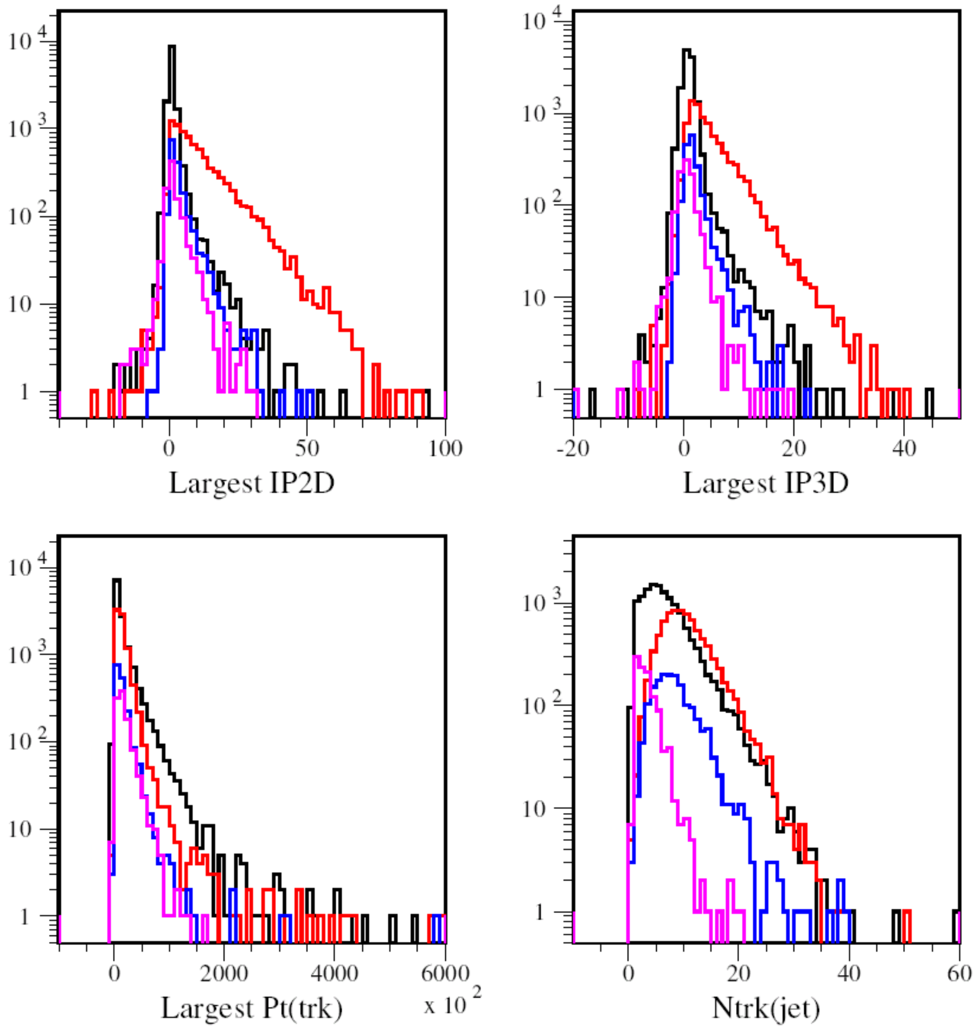
B-jet(red), C-jet(blue), τ-jet(pink), Light-jet(black)



- mass – mass of particles which participate in vertex fit
- efrc – energy ratio between particles in vertex and in jet
- N2t – No. of 2-track vertices
- Ntrk(vertex) – number of tracks in vertex

Input Variables

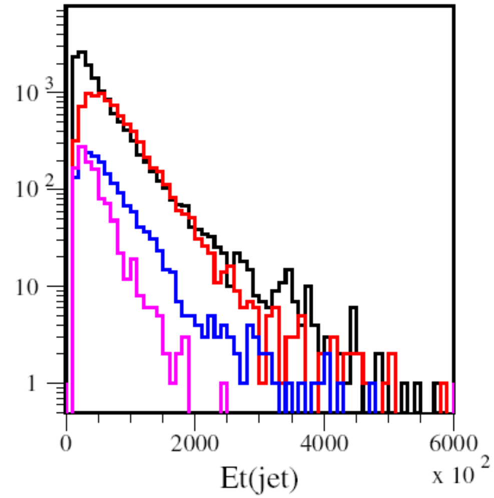
B-jet(red), C-jet(blue), τ-jet(pink), Light-jet(black)

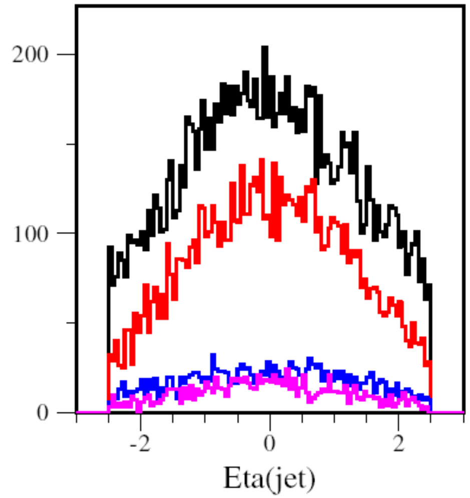


- Largest IP2D – largest transverse IP significance of tracks in the jet
- Largest IP3D – largest longitudinal IP significance of tracks in the jet
- Largest Pt – largest transverse momentum of tracks in the jet
- Ntrk(jet) – track multiplicity in jet

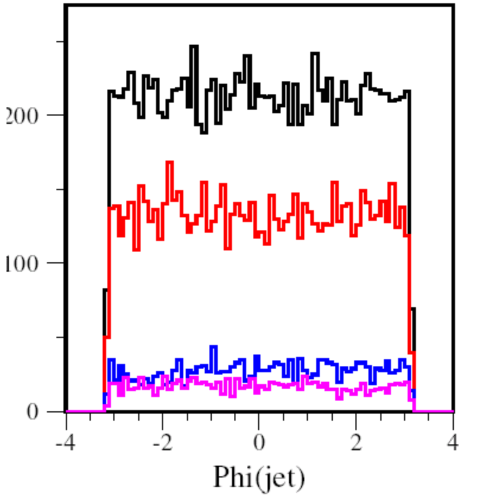
Input Variables

B-jet(red), C-jet(blue), τ-jet(pink), Light-jet(black)





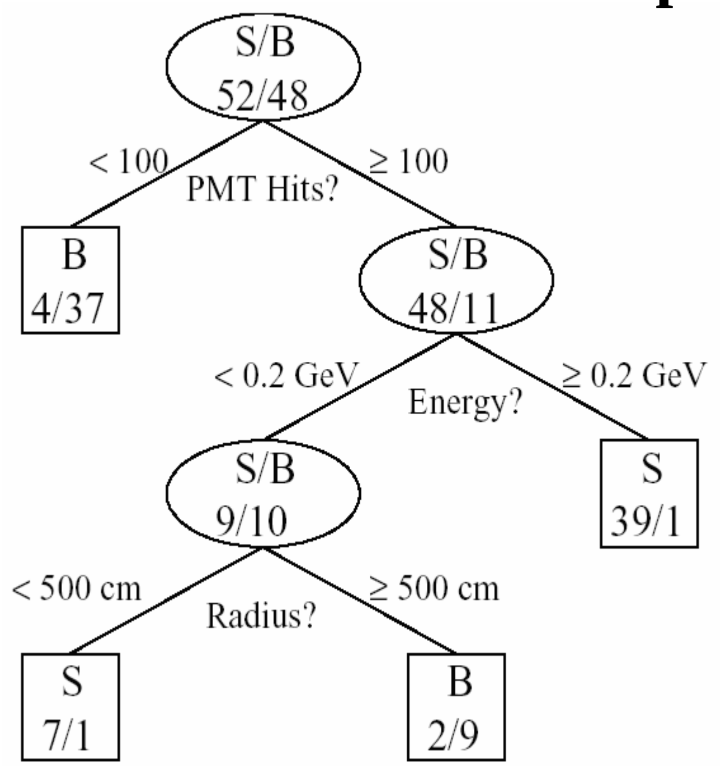
- Et(jet) – Et of jet
- Eta(jet) – η of jet
- Phi(jet) – ϕ of jet



Boosted Decision Trees

➔Relatively new in HEP – MiniBooNE, BaBar, D0(single top discovery), ATLAS
➔Advantages: robust, understand 'powerful' variables, relatively transparent, …

"A procedure that combines many weak classifiers to form a powerful committee"



BDT Training Process

•Split data recursively based on input variables until a stopping criterion is reached (e.g. purity, too few events)

• Every event ends up in a "signal" or a "background" leaf

• Misclassified events will be given larger weight in the next decision tree (boosting)

H. Yang et.al. NIM A555 (2005)370, NIM A543 (2005)577, NIM A574(2007) 342

A set of decision trees can be developed,

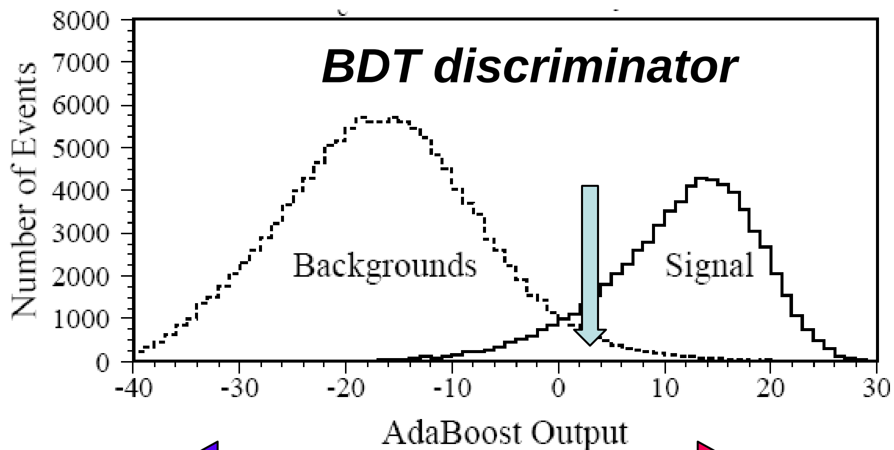
each re-weighting the events to enhance identification of backgrounds misidentified by earlier trees (“boosting”)

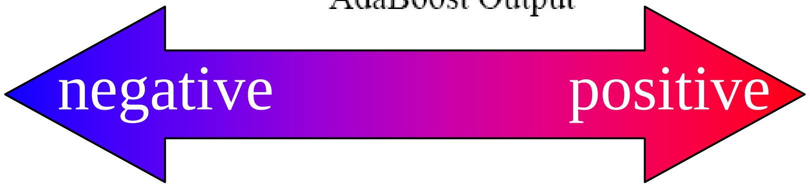
For each tree, the data event is assigned

+1 if it is identified as signal,

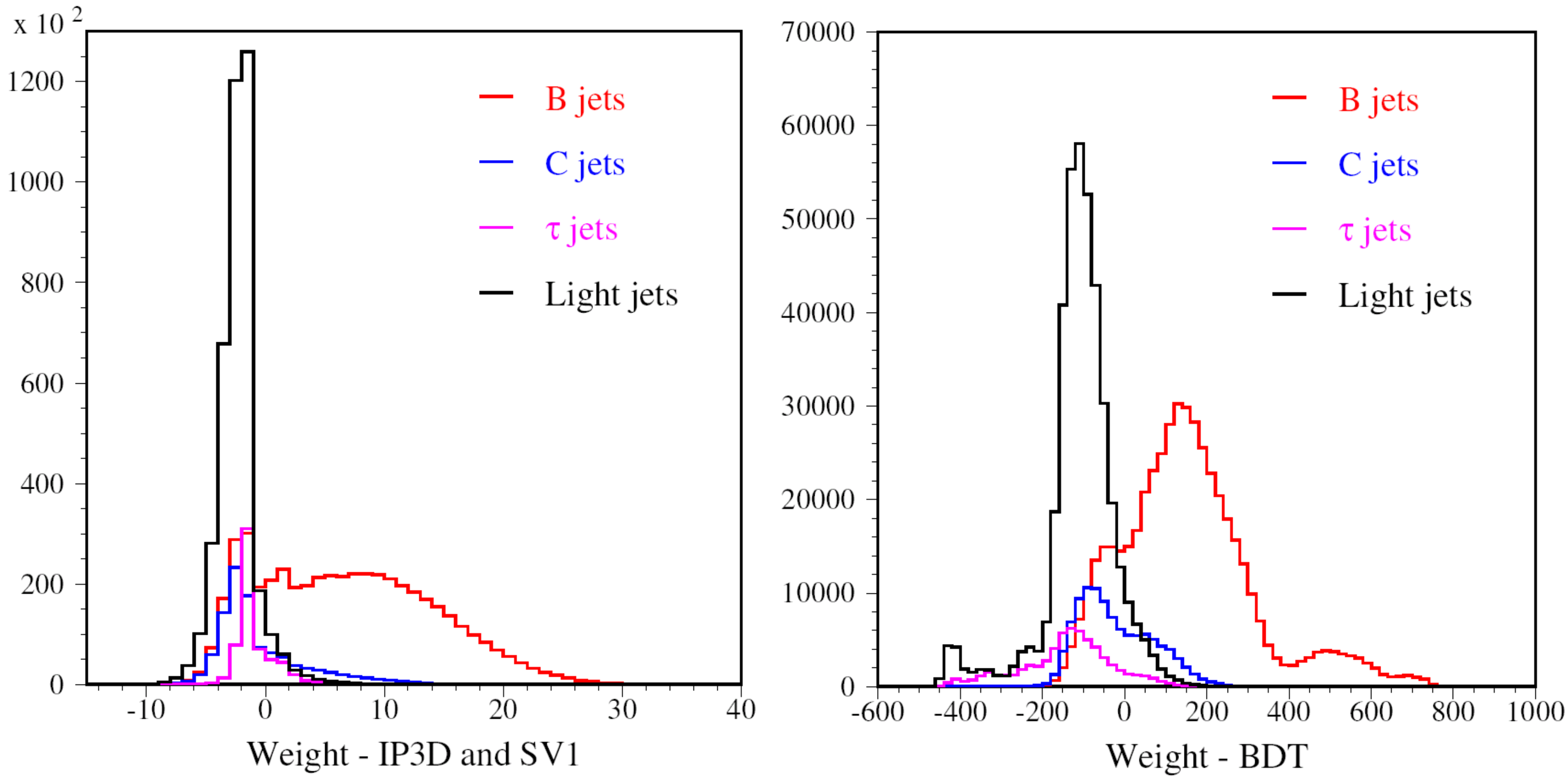
- 1 if it is identified as background.

The total for all trees is combined into a “**score**”



Background-like  signal-like

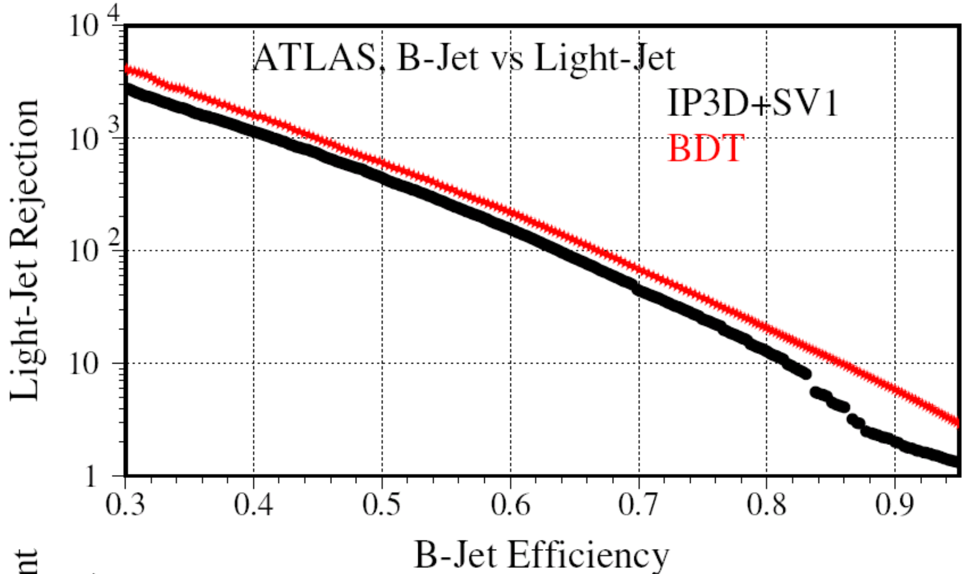
B-tagging Weights vs BDT Discriminator

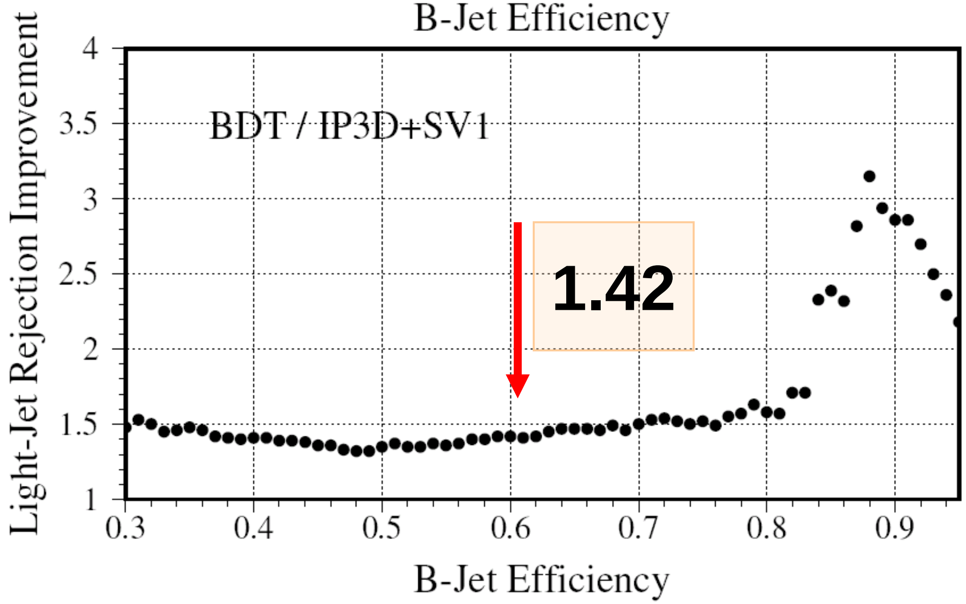


Discriminating Power of Input Variables

Rank	Description of Input Variable	Gini Index(%)
1	IP3D + SV1	83.04%
2	Ntrk(jet) – number of tracks in jet	6.96%
3	Largest trans. IP significance of tracks in jet	2.41%
4	Largest Pt(trk) – largest Pt of tracks in jet	2.28%
5	Softm – soft muon based tagger	1.24%
6	Efrc – e ratio of particles in vertex & in jet	0.79%
7	Mass – mass of particles used in vertex fit	0.63%
8	Et(jet) – transverse energy of jet	0.54%
9	IP2D – jet weight from transverse IP	0.44%
10	IP3D – jet weight from 3D IP	0.28%

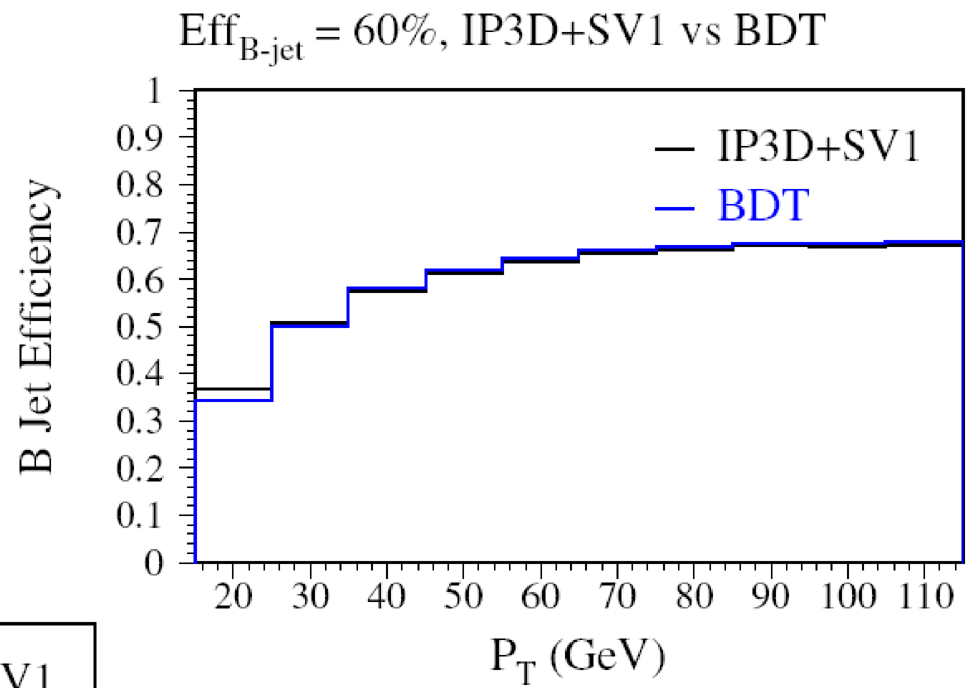
Results: B-jets vs Light-jets

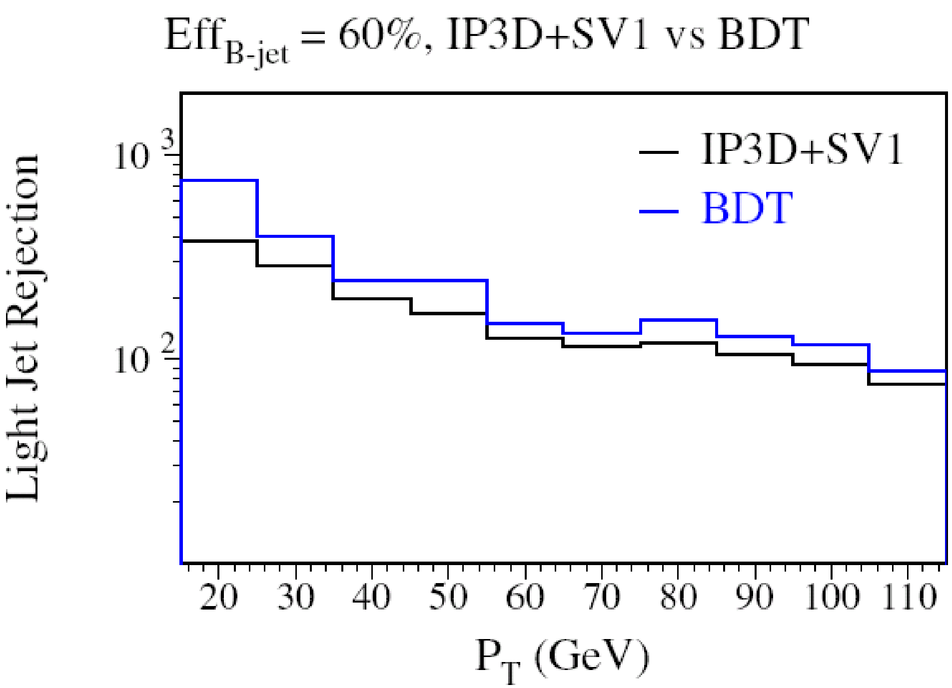




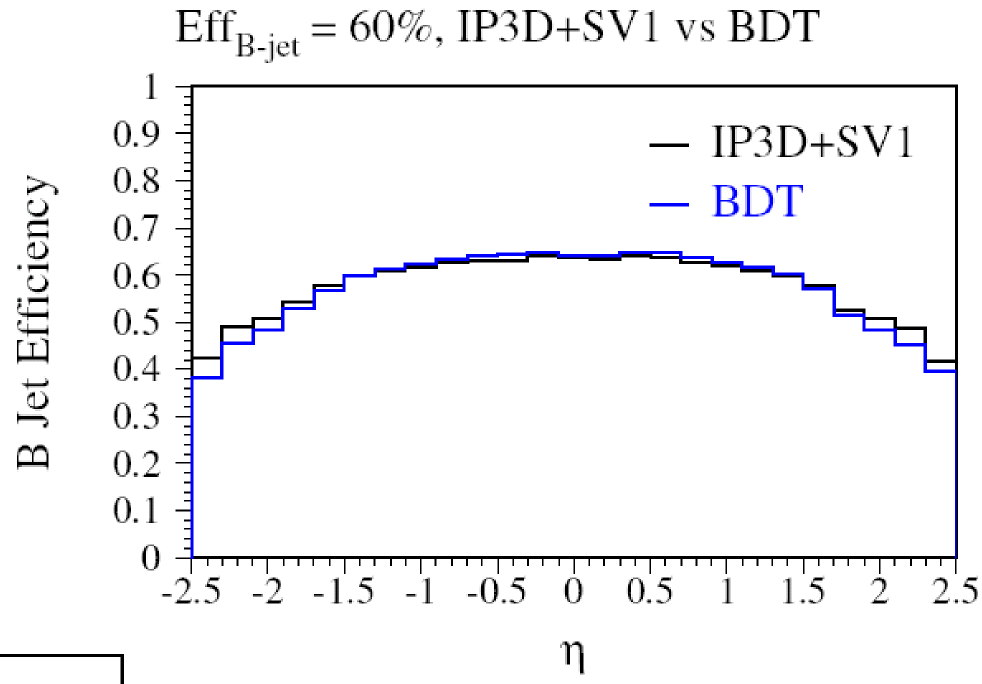
$$\frac{\text{Rejection (BDT)}}{\text{Rejection (IP3D+SV1)}}$$

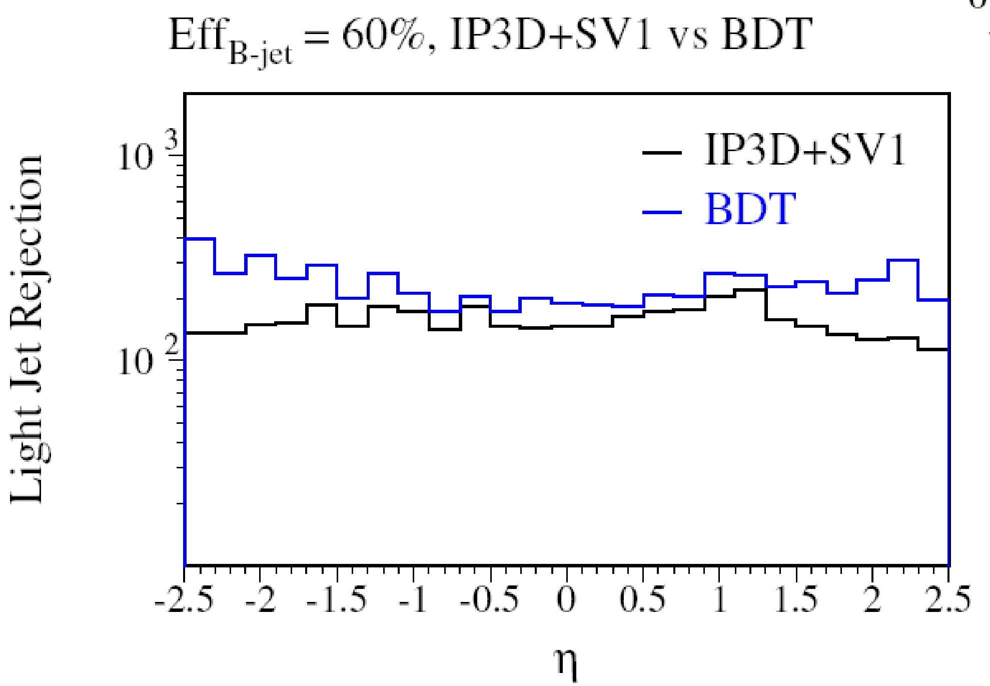
B-jet Eff. vs P_T



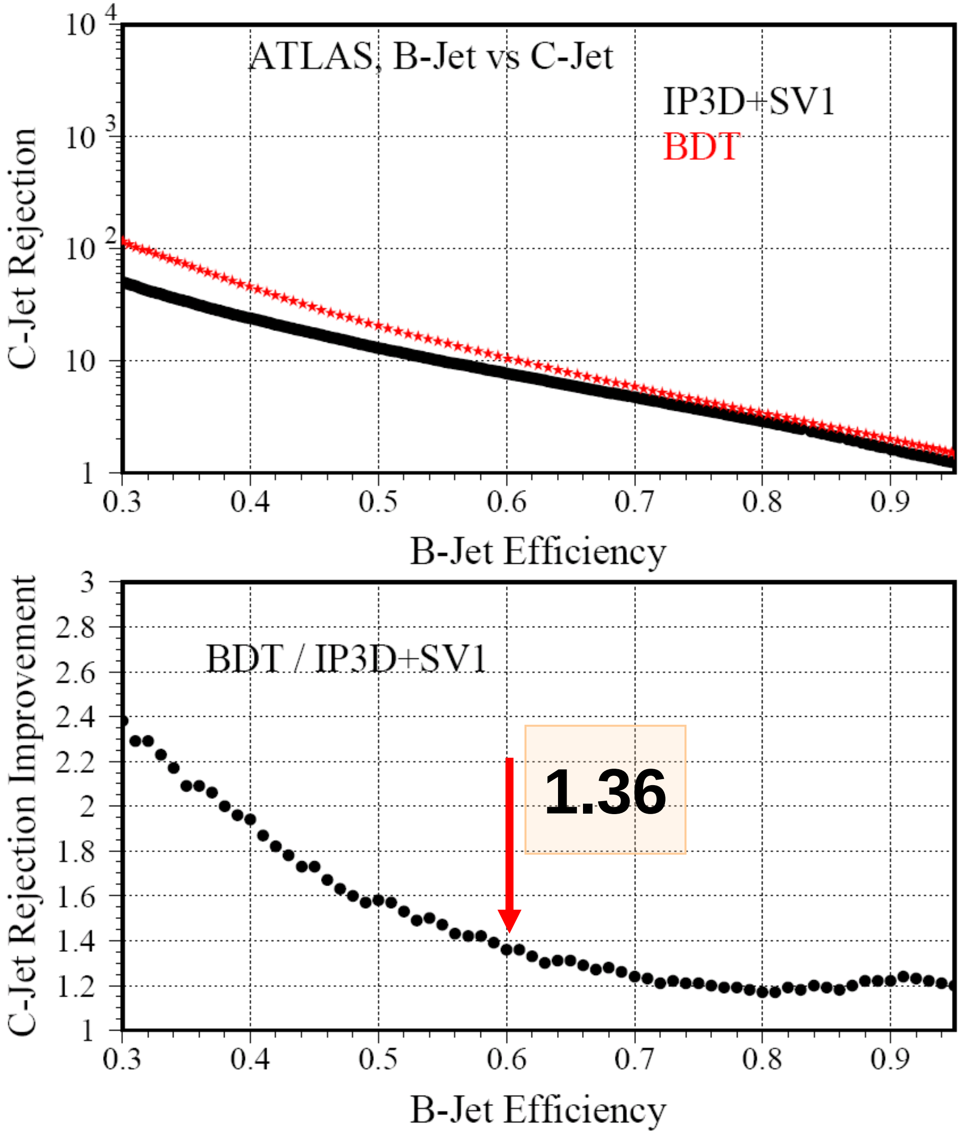


Light-jet Rejection vs η



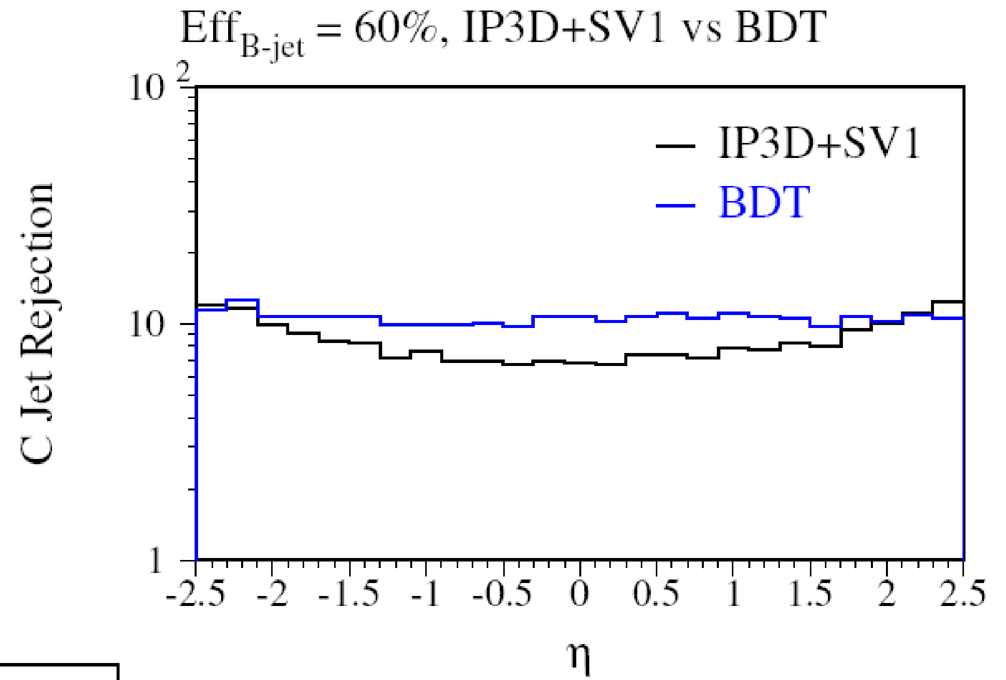


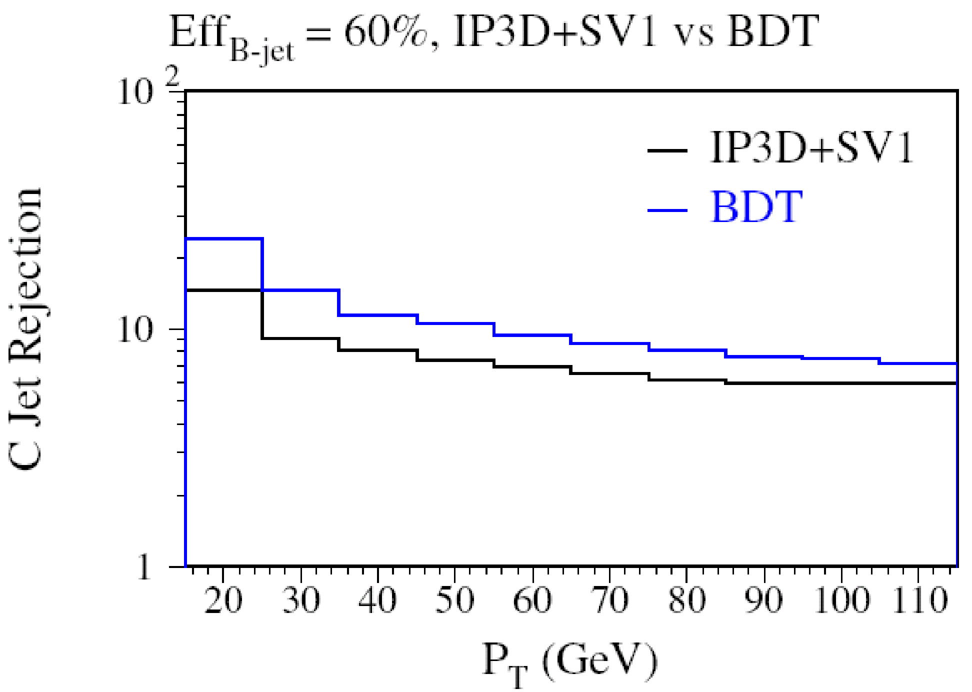
Results: B-jets vs C-jets



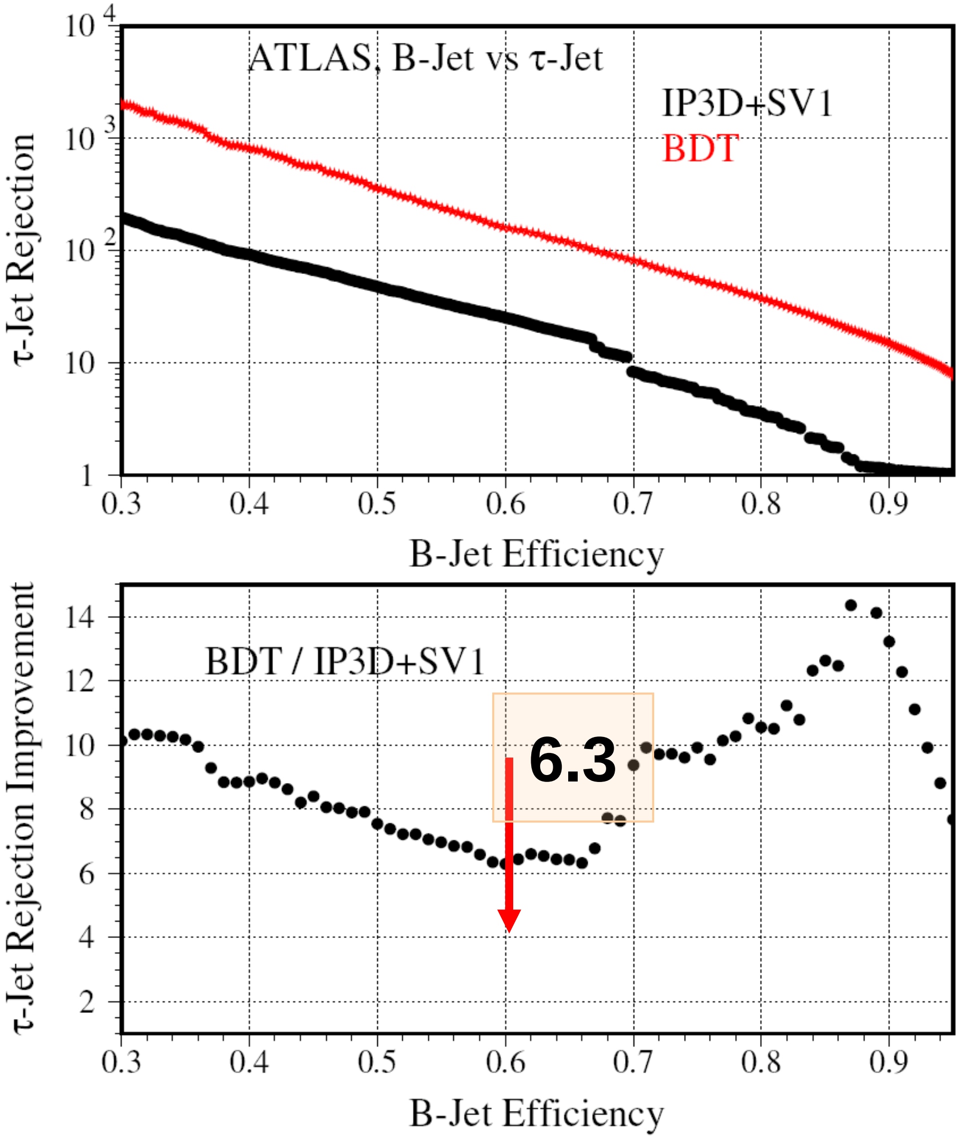
$$\frac{\text{Rejection (BDT)}}{\text{Rejection (IP3D+SV1)}}$$

C-jets Rejection vs P_T, η



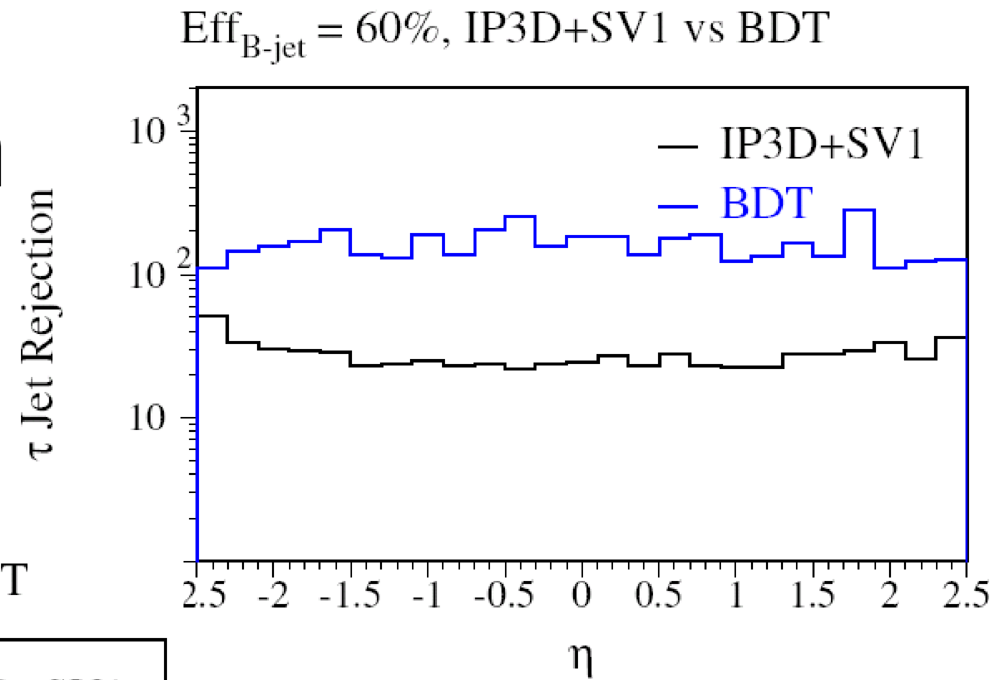


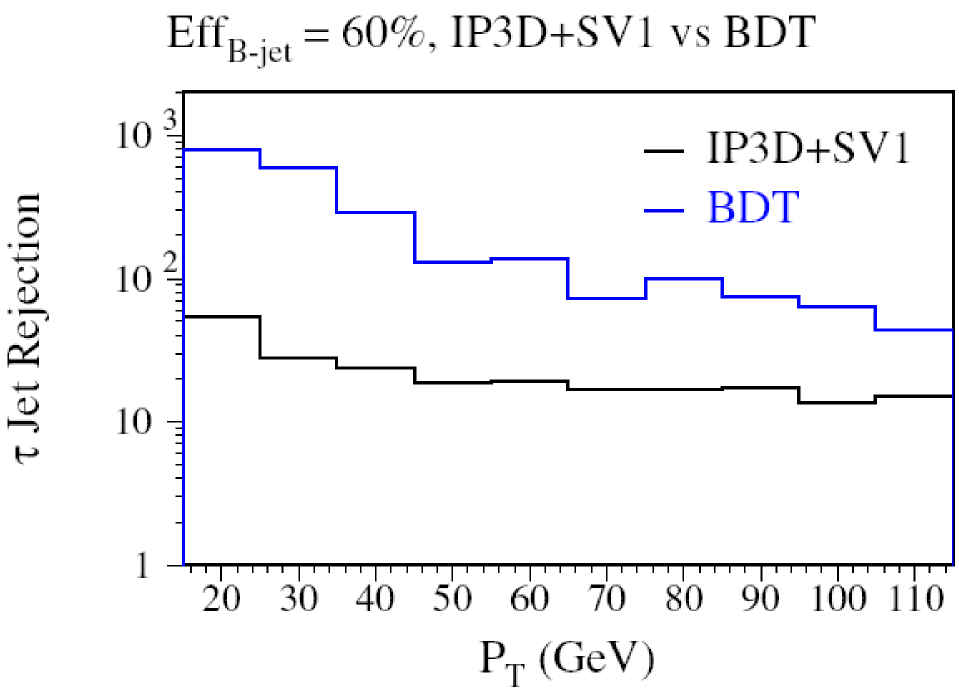
Results: B-jets vs τ-jets



$$\frac{\text{Rejection (BDT)}}{\text{Rejection (IP3D+SV1)}}$$

τ-jets Rejection vs P_T, η





Summary and Future Plan

- BDT works better than IP3D+SV1 by combining several existing b-tagging weights. The light jet rejection is improved by 40%-60% for wide b-tagging efficiency range of 30%-80%. For 60% b-tagging efficiency, c jet and τ jet rejection are improved by 1.36 and 6.3, respectively.

- Considering more discriminating variables which may help for b-tagging, eg. signed transverse impact parameter, individual track probability, jet probability / mass / width, number of tracks with certain decay length, 2D/3D decay length.
- Using v14 MC samples ($\sqrt{s}$ = 10 TeV) to evaluate b-tagging performance.

Backup Slides for BDT

Criterion for “Best” Tree Split

- Purity, *P,* is the fraction of the weight of a node (leaf) due to signal events.
- Gini Index: Note that Gini index is 0 for all signal or all background.

$$Gini = \left(\sum_{i=1}^{n} W_i\right) P(1-P)$$

- The criterion is to minimize

  Gini_left_node+ Gini_right_node.

Criterion for Next Node to Split

- Pick the node to maximize the change in Gini index. Criterion =

  $Gini_{parent\_node} – Gini_{right\_child\_node} – Gini_{left\_child\_node}$

- We can use Gini index contribution of tree split variables to sort the importance of input variables.
- We can also sort the importance of input variables based on how often they are used as tree splitters.

Signal and Background Leaves

- Assume an equal weight of signal and background training events.
- If event weight of signal is larger than ½ of the total weight of a leaf, it is a signal leaf; otherwise it is a background leaf.
- Signal events on a background leaf or background events on a signal leaf are misclassified events.

How to Boost Decision Trees ?

- For each tree iteration, same set of training events are used but the weights of misclassified events in previous iteration are increased (boosted). Events with higher weights have larger impact on Gini index values and Criterion values. The use of boosted weights for misclassified events makes them possible to be correctly classified in succeeding trees.

- Typically, one generates several hundred to thousand trees until the performance is optimal.

- The score of a testing event is assigned as follows: If it lands on a signal leaf, it is given a score of 1; otherwise -1. The sum of scores (weighted) from all trees is the final score of the event.

Two Boosting Algorithms

- AdaBoost Algorithm:

1. Initialize the observation weights $w_i = 1/\text{n}$, i = 1, 2,..., n
2. For m = 1 to M:
   2.a Fit a classifier $T_m(x)$ to the training data using weights w_i
   2.b Compute

$$err_m = \frac{\sum_{i=1}^{n} w_i I(y_i \neq T_m(x_i))}{\sum_{i=1}^{n} w_i}$$

I = 1, if a training event is misclassified; Otherwise, I = 0

   2.c Compute $\alpha_m = \beta \times log((1 - err_m)/err_m)$
   2.d Set $w_i \leftarrow w_i \times exp(\alpha_m I(y_i \neq T_m(x_i)))$, i=1, 2,...,n
   2.e Re-normalize $w_i = w_i / \sum_{i=1}^{n} w_i$
3. Output T(x) = $\sum_{m=1}^{M} \alpha_m T_m(x)$

- $\epsilon-$boosting Algorithm:

1. Initialize the observation weights $w_i = 1/\text{n}$, i = 1, 2,..., n
2. For m = 1 to M:
   2.a Fit a classifier $T_m(x)$ to the training data using weights w_i
   2.b Set $w_i \leftarrow w_i \times exp(2\epsilon I(y_i \neq T_m(x_i)))$, i=1, 2,...,n
   2.c Re-normalize $w_i = w_i / \sum_{i=1}^{n} w_i$
3. Output T(x) = $\sum_{m=1}^{M} \epsilon T_m(x)$

Example

- **AdaBoost: the weight of misclassified events is increased by**
  - error rate=0.1 and β = 0.5, α_m = 1.1, exp(1.1) = 3
  - error rate=0.4 and β = 0.5, α_m = 0.203, exp(0.203) = 1.225
  - Weight of a misclassified event is multiplied by a large factor which depends on the error rate.

- **ε–boost: the weight of misclassified events is increased by**
  - If $\varepsilon = 0.01$, exp(2*0.01) = 1.02
  - If $\varepsilon = 0.04$, exp(2*0.04) = 1.083
  - It changes event weight a little at a time.

➔ AdaBoost converges faster than ε-boost. However, the performance of AdaBoost and ε–boost are very comparable with sufficient tree iterations.