

g14 analysis using new statistical data analysis methods

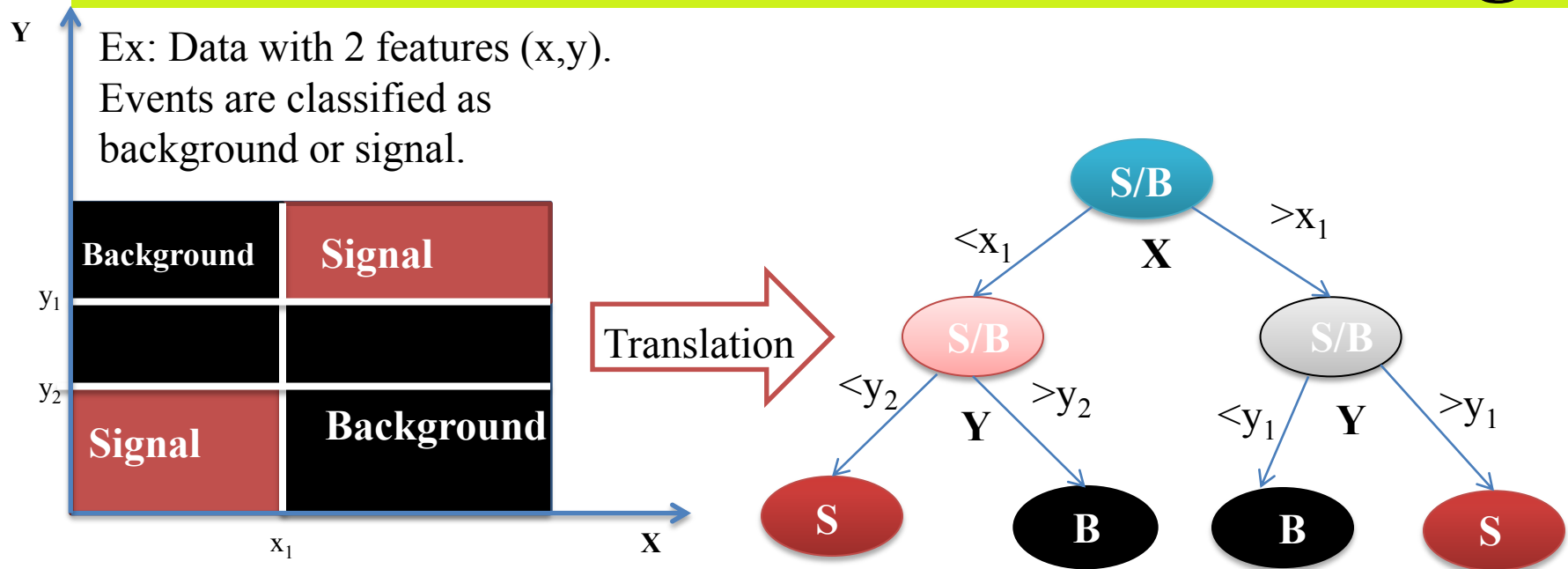
Dao Ho

03/07/2014

Outline

- **Boosted Decision Trees (BDT)**: a supervised-learning classifier, i.e, the BDT is **trained** before can be applied to **classification** tasks
- **Kernel Density Estimator (KDE)**: a smooth and continuous method to estimate density distributions
- **Bootstrap**: a data-resampling method to estimate standard deviation, or confidence interval

Decision Tree introduction: training



- Decision tree is similar to “cut” method
- More advanced because it can classify high dimensional (>3) data into hyper-cube regions simultaneously.
- No events are removed, only classified → no information is lost!!!
- **Question: what feature (variable) and what value should be used at each node?**

Decision Tree introduction: training

Goal: *To improve data “purity” after each node splitting using information entropy*

$$Entropy(n_{sig}, n_{bg}) = -\frac{n_{sig}}{n_{total}} \log \frac{n_{sig}}{n_{total}} - \frac{n_{bg}}{n_{total}} \log \frac{n_{bg}}{n_{total}}$$

Pick a variable V_i and its cut value v from a set of variables to maximize the following formula:

$$Gain(n_{sig}, n_{bg}, V_i) = Entropy(n_{sig}, n_{bg}) - \sum_{A_1, A_2} \frac{n_{A_j}}{n_{total}} Entropy(n^{A_j}_{sig}, n^{A_j}_{bg})$$

where A_1 and A_2 are two sets of events separated by v (one set with $V_i < v$, the other with $V_i > v$),

→ → Pick the cut **value** v of variable V_i such that the percentage of signal or background events after splitting is **larger** than the original percentages.

→ → Repeat for **all** V s and select one V with **maximum gain**

Decision Tree introduction: **training**

- Good: easy to understand, straight forward implementation, and no event removed.
- **Bad: the tree can perfectly classify training data if given enough splittings; overfitting always occurs → highly sensitive to statistical fluctuation.**
- **Solution: limit amount of splittings, and BOOSTing**

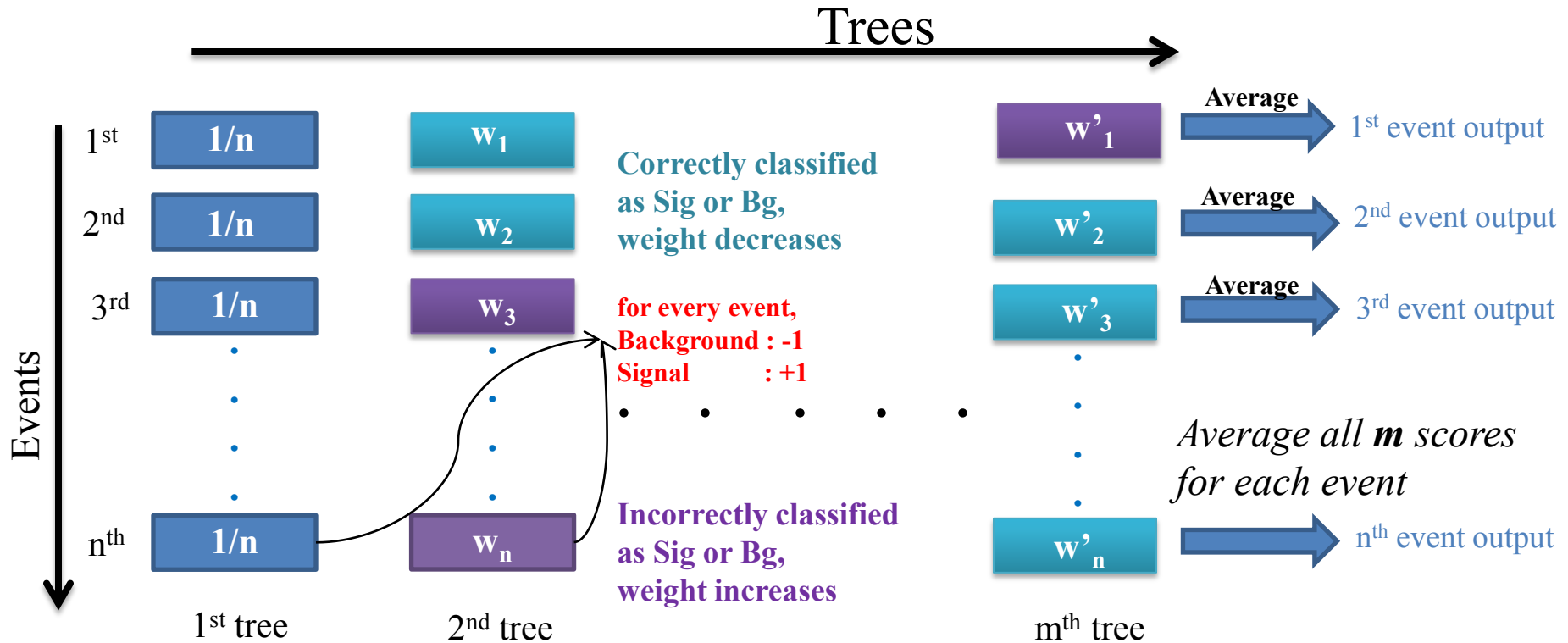
Boosting means using decision tree multiple times on reweighted training data to improve performance

→ Build a forest of many *different* decision trees

BOOSTED Decision Tree introduction:

Boosting: run decision tree multiple times on reweighted training data

→→ Build a forest of many different decision trees



- Each new tree is built differently due to having a new set of weights for all events
- Each new tree focuses more on wrongly classified events by previous trees

BDT in action:

❑ Background Subtraction: E asymmetry for $\gamma n(p_s) \rightarrow p\pi^-(p_s)$ reaction

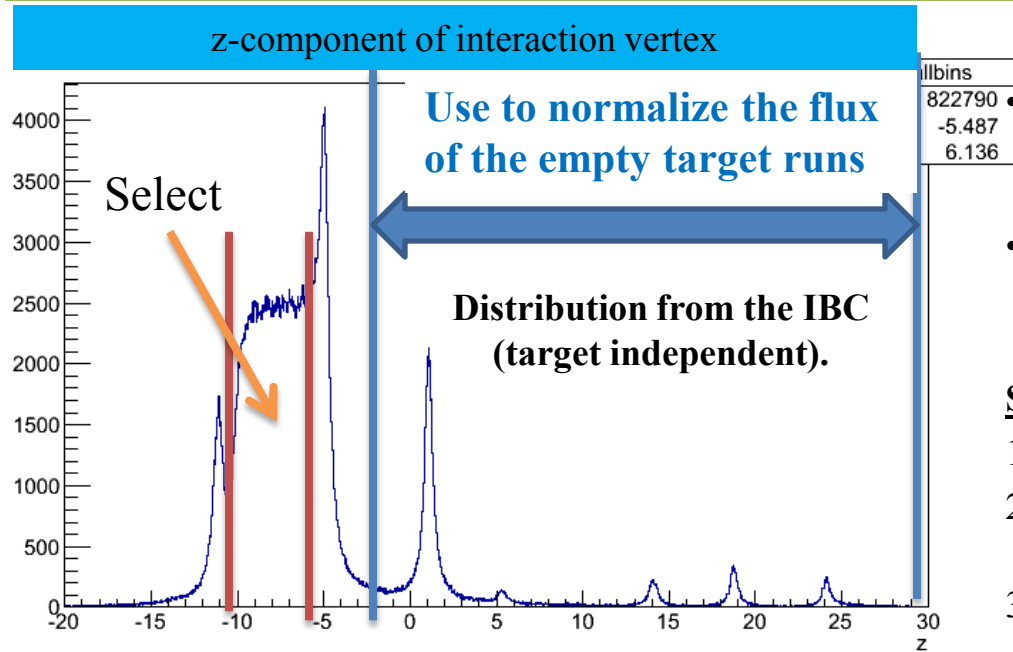
- g14 targets have Al wires, and KelF cell wall (background to remove)
- g14 experiment had empty target run, used for background subtraction
- Also, select events with low spectator proton momentum

❑ BDT: E asymmetry for $\gamma n(p_s) \rightarrow p\pi^-(p_s)$ reaction

- Using empty run as background training data
- Used simulated $\gamma n(p_s) \rightarrow p\pi^-(p_s)$ reaction as signal training data
- Train the trees to develop algorithm for classification
- Check for overfitting
- Employed the trained trees for classifying signal, and background in g14 gold2 target run period

❑ Compare the two methods

Background subtraction:



Background (BG) comes mainly from Al wires inside the target and Kelf target cell.

- Empty target runs to obtain BG distribution.

Steps:

1. Apply cuts to clean up gold2 target data.
2. Run the same analysis on empty target data.
3. Normalize the IBC flux with full target data and obtain the scaling factor.
4. Subtract scaled BG (from empty runs) to align yield ($Y^{3/2}$) and anti align yield ($Y^{1/2}$) of full target runs.

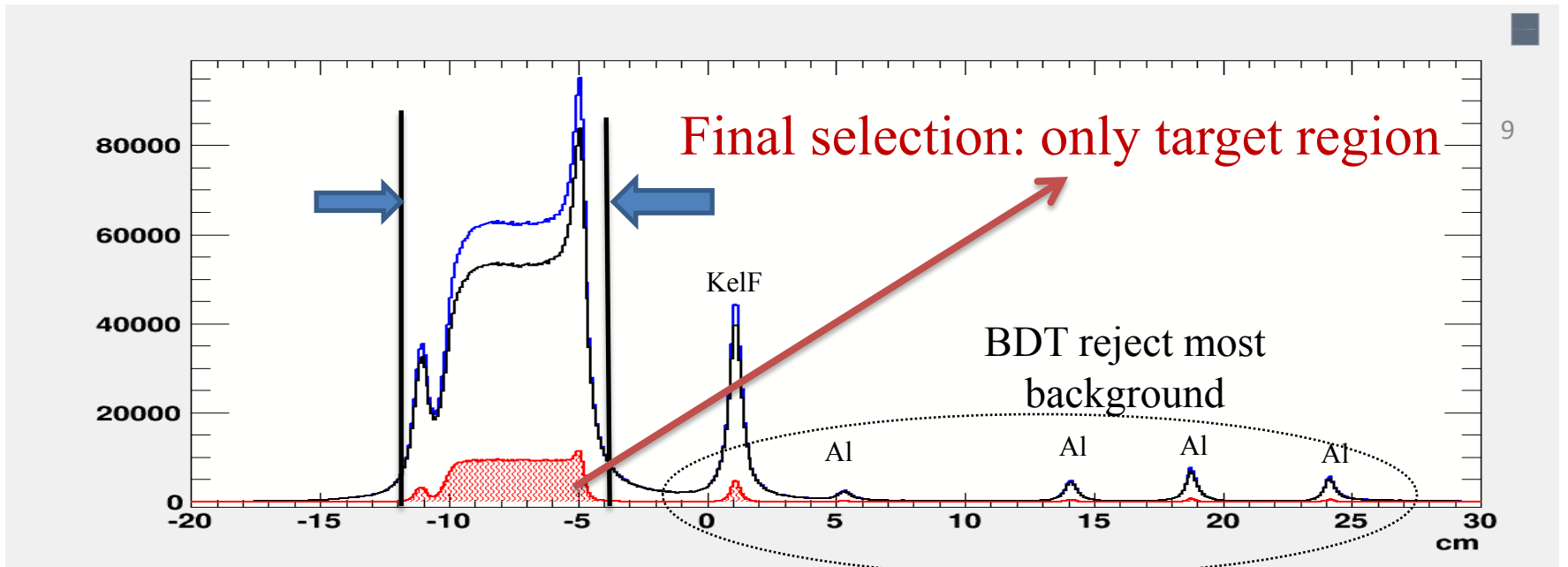
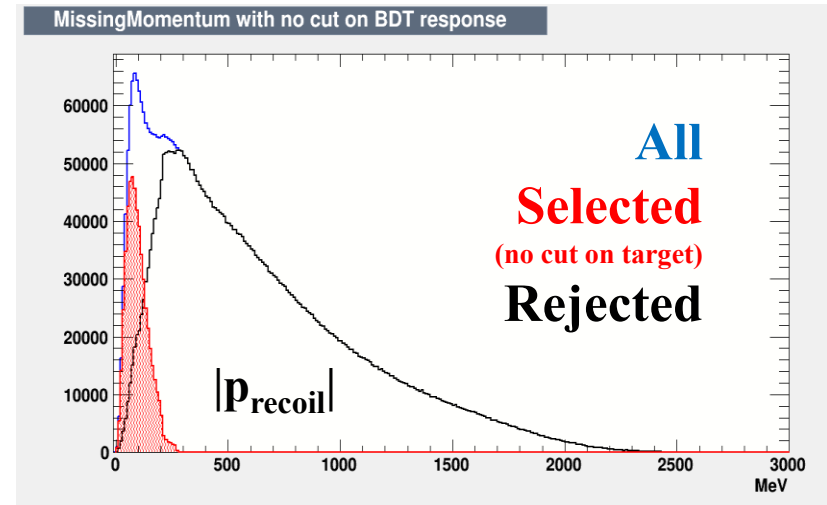
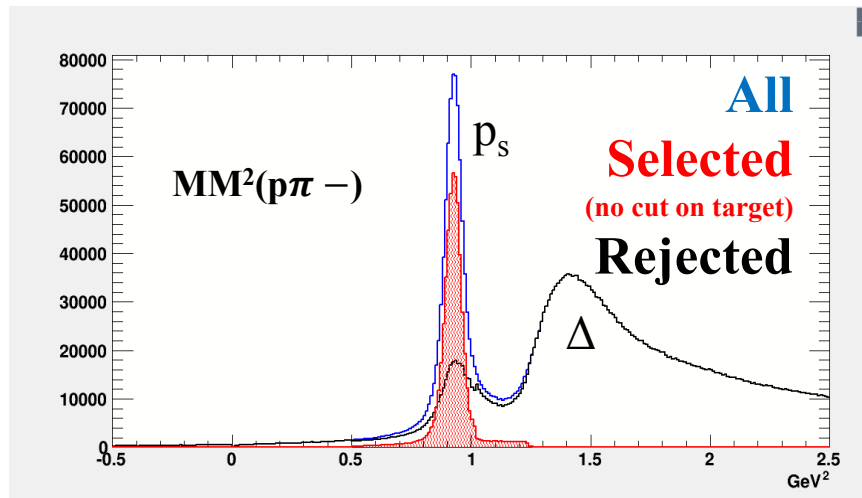
Verified for empty run $Y^{1/2} \approx Y^{3/2}$

$$Y_{BG} = 1/2 * (Y^{1/2} + Y^{3/2}) * \text{scaling factor}$$

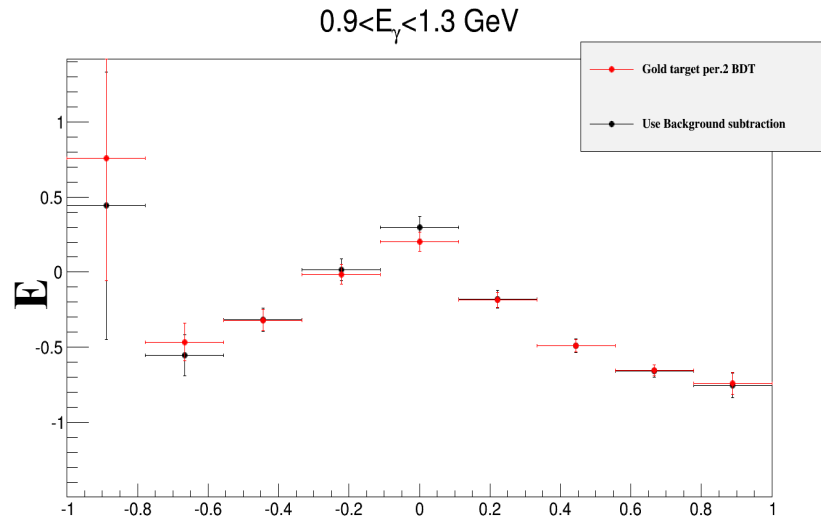
$$Y_{HD}^{1/2} = Y_{full}^{1/2} - Y_{BG} \quad Y_{HD}^{3/2} = Y_{full}^{3/2} - Y_{BG}$$

$$E = (P_\gamma \times P_{target})^{-1} \times (Y_{HD}^{1/2} - Y_{HD}^{3/2}) / (Y_{HD}^{1/2} + Y_{HD}^{3/2})$$

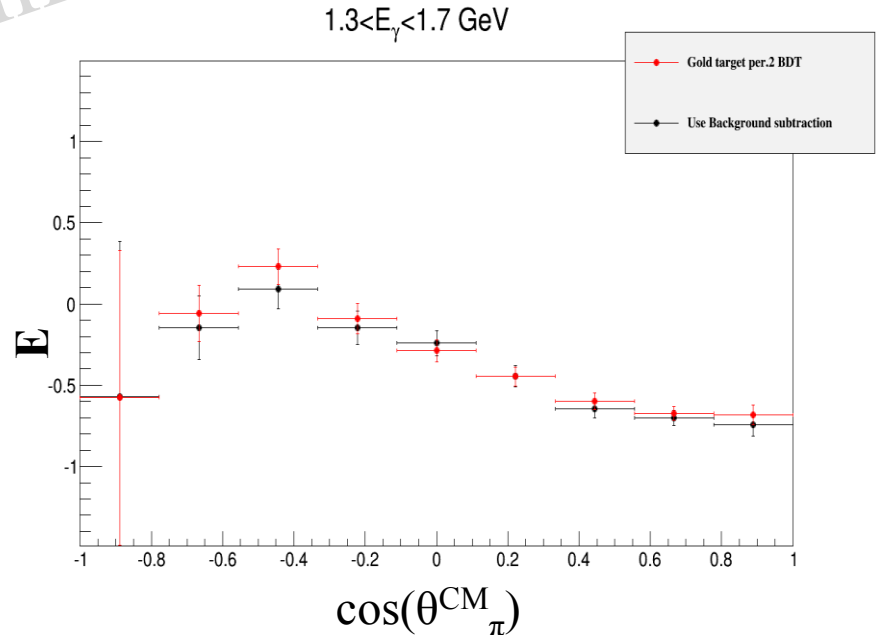
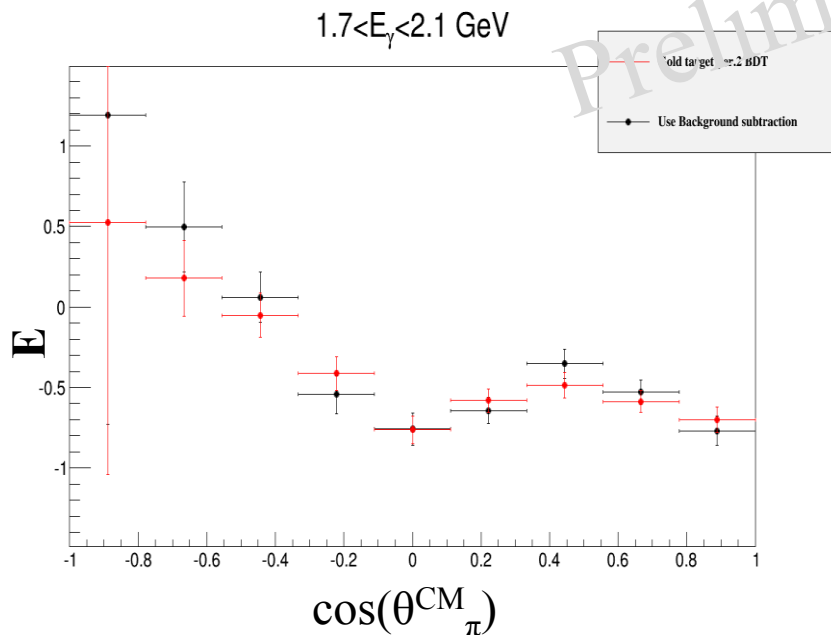
Event selection using BDT: use trained trees



Background subtraction vs. BDT:

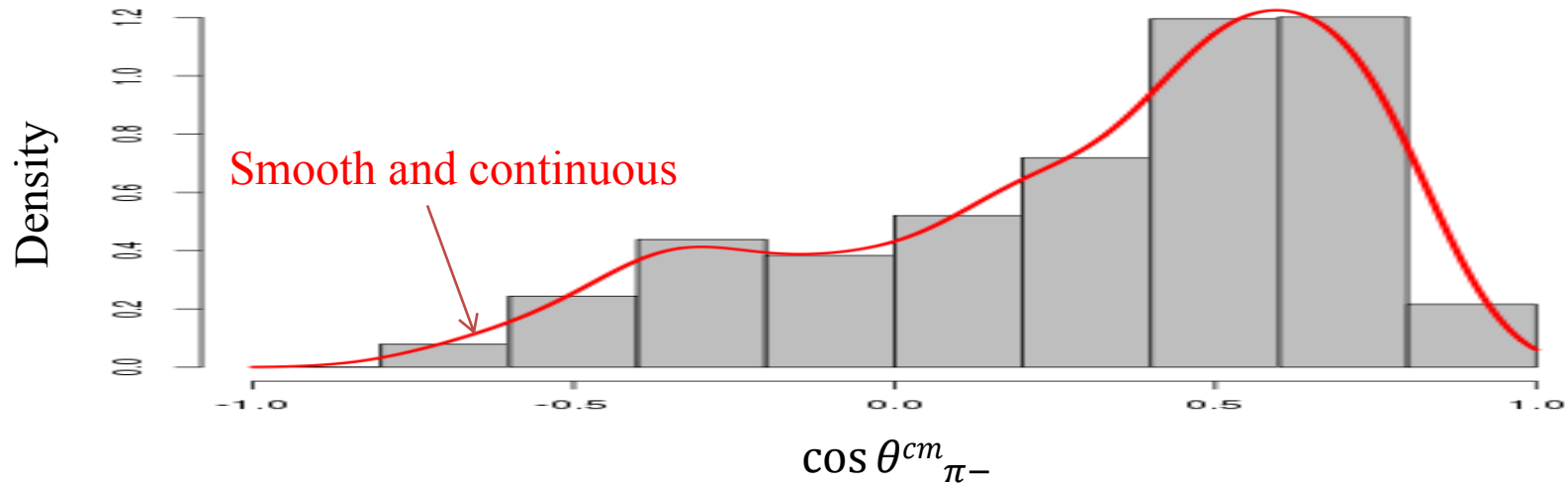


- Using gold2 period data for this check
- Red points: BDT (error bars slightly smaller)**
- Black points: Background subtraction**
- Good agreement between the two methods
- BDT can be used for other classification tasks as well ($K^0\Lambda$ selection in $p\pi^+\pi^-$)**



Kernel Density Estimator (KDE) introduction:

Histogram and Kernel Density Estimator (red)

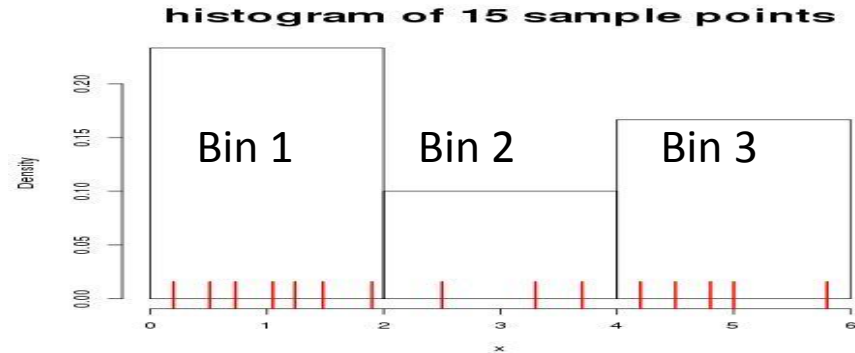


- KDE is a **non-parametric method** to estimate a probability density distribution (others is the histogram estimator)
- Every data point X_i is “**smeared**” by a density function with mean at X_i (for example, Gaussian density function)
- KDE then “**sums up**” these distributions (one at each data point X_i) to estimate the underlying distribution that the X_i s were sampled from

Mathematical Formula for Histogram Density Estimator:

Example: Given 15 sample points below, divided into 3 bins (see figure); $X_i \in (0.0, 6.0), i \in \{1, 2, 3, \dots, 14, 15\}$

Sample Data Points (X_i)	
0.73	0.20
0.51	1.24
1.90	1.05
1.48	2.50
3.70	3.30
4.80	5.00
4.50	4.20
5.80	



Draw back of histogram method (more severe with low-statistics data):

- Sensitive to bin width (bias toward data points closed to bin edges).
- Use only partial data for estimation. i.e., to estimate $f(x)$ with $x \in B_j$ use only data points $X_i \in B_j$.
- $\hat{f}_d(x)$ is a discontinuous function (step function).

Define: $B_j = (x_0 + (j - 1)d, x_0 + jd)$

where x_0 is the origin, and d is bin width.

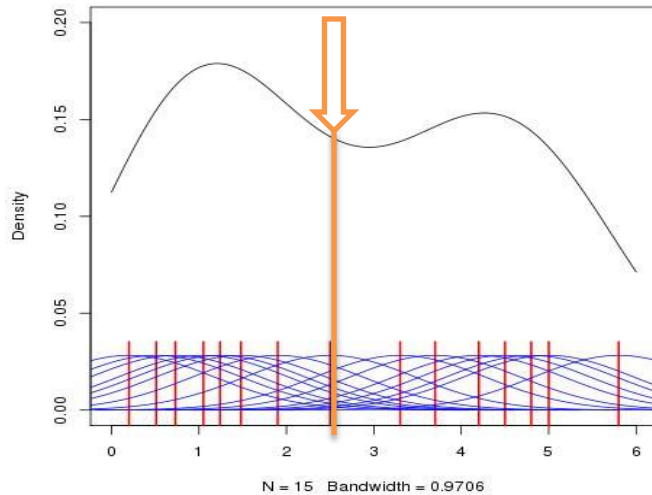
$$\hat{f}_d(x) = \frac{1}{nd} \sum_{i=1}^n \sum_{j=1}^{\#bins} 1_{(X_i \in B_j)} 1_{(x \in B_j)}$$

where X_i is the sample data points. $\hat{f}_d(x)$ is constant for all x in a same bin.

X_i : recorded sample data points; x : estimating points

Mathematical Formula for Kernel Density Estimator (KDE):

kernel density estimator



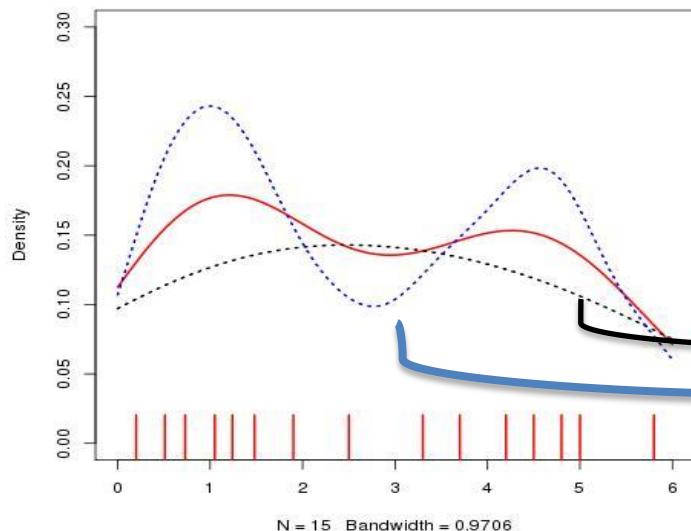
$$\hat{f}_h(x) = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{x-X_i}{h}\right) \text{ where}$$

- X_i is the sample data points,
- h is the bandwidth (smoothing parameter),
- $K(x)$ can be any symmetric density function.
- Often, $K(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(x)^2}$, the normal distribution

Ex:

$$\hat{f}_h(2.5) = \frac{1}{15h} \sum_{i=1}^{15} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(2.5 - X_i)^2}{2h^2}\right)$$

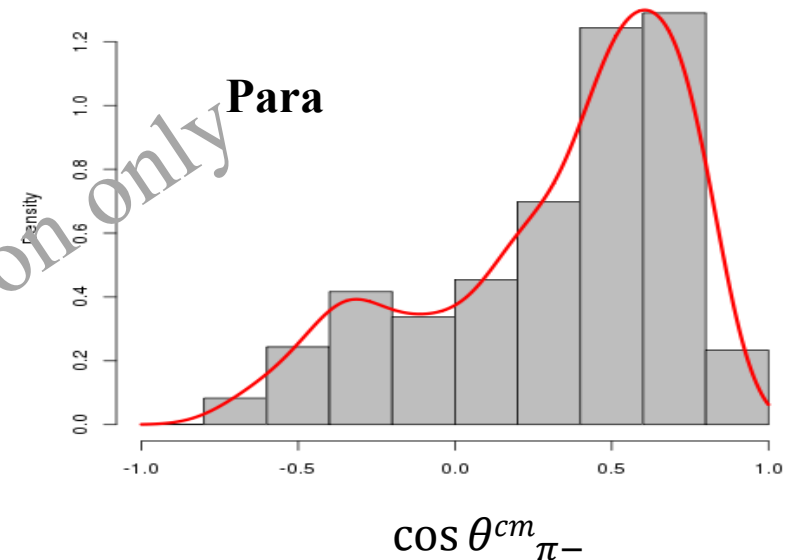
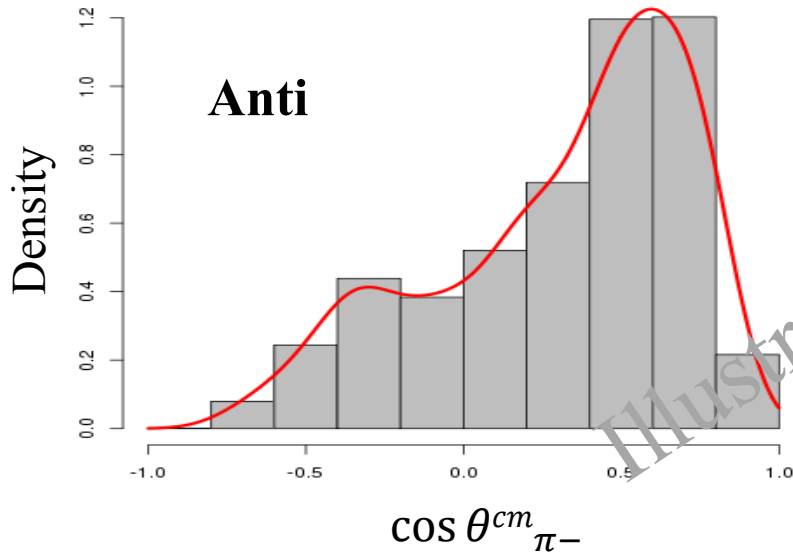
kernel density estimator



- $\hat{f}_d(x)$ is estimated using all data points (all X_i s).
 - $K(x)$ is smooth, so $\hat{f}_d(x)$ is smooth (continuous).
 - h is strongly influenced on the resulting estimate,
- too large h obscures underlying structure
- too small h results in many fluctuations

Using KDE to “plot” E asymmetry:

Histogram and KDE for gold2, left is anti ($Y^{1/2}$), right is para ($Y^{3/2}$)



Using histogram

$$E(x_{B_j}) = \frac{1}{pol} \frac{n_{B_j}^{\downarrow\uparrow} - n_{B_j}^{\uparrow\uparrow}}{n_{B_j}^{\downarrow\uparrow} + n_{B_j}^{\uparrow\uparrow}} \text{ where } n_{B_j}^{\uparrow\uparrow} = \# \text{ events in bin } B_j \text{ for para data.}$$

Using KDE

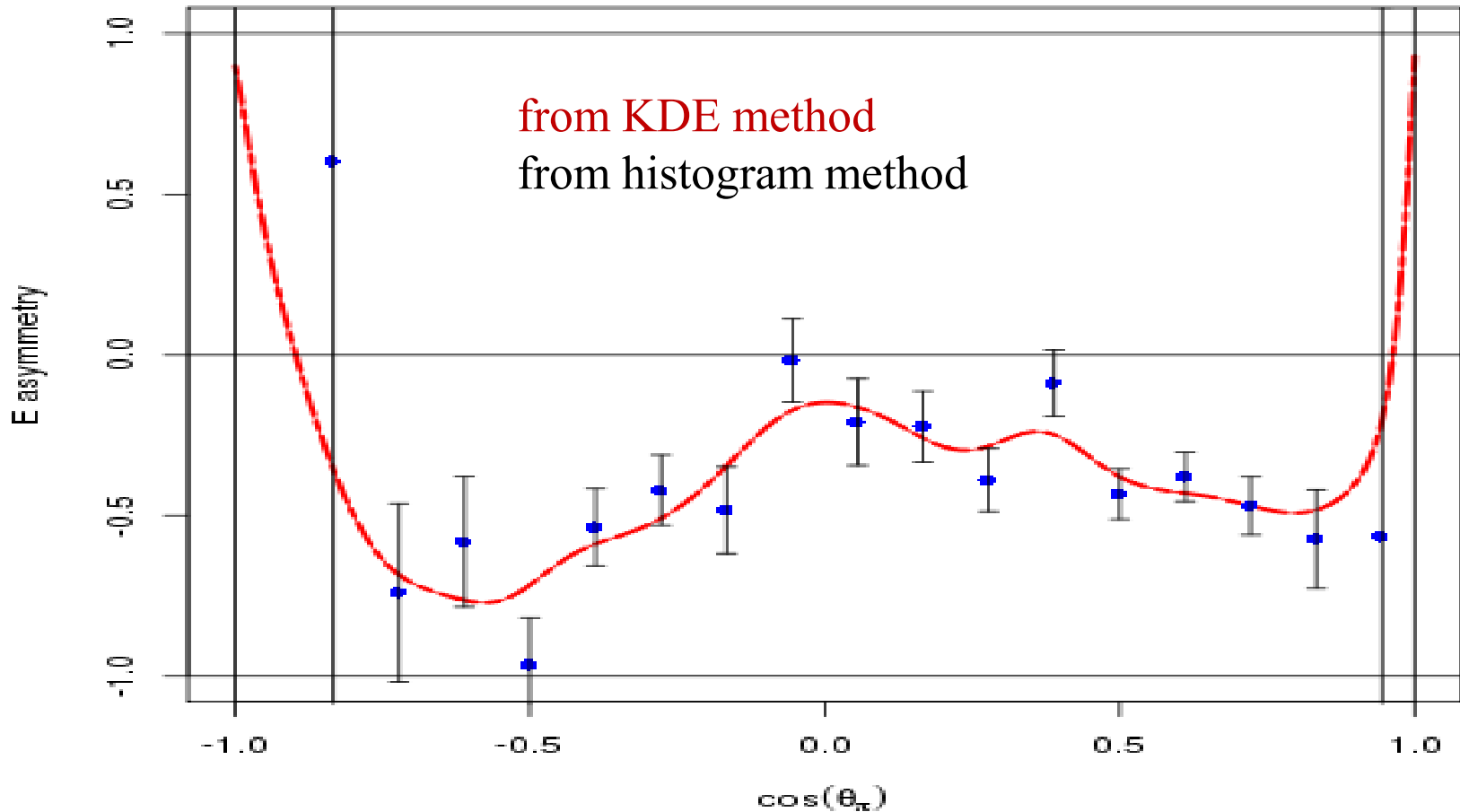
$$E(x) = \frac{1}{pol} \frac{n^{\downarrow\uparrow} \hat{f}(x)_{\downarrow\uparrow} - n^{\uparrow\uparrow} \hat{f}(x)_{\uparrow\uparrow}}{n^{\downarrow\uparrow} \hat{f}(x)_{\downarrow\uparrow} + n^{\uparrow\uparrow} \hat{f}(x)_{\uparrow\uparrow}} \text{ where } n^{\uparrow\uparrow} = \text{total \# events for para data, and}$$

$$\int_{-1}^1 \hat{f}(x)_{\downarrow\uparrow} dx = \int_{-1}^1 \hat{f}(x)_{\uparrow\uparrow} dx = 1$$

Using KDE to “plot” E asymmetry:

$0.9 \text{ GeV} < E_\gamma < 1.0 \text{ GeV}$; gold2 run period

gold2 period using GAUSSIAN KERNEL



How to pick the bandwidth h ?

Using Least Square Cross Validation to pick h:

Consider:

$$\hat{f}_h(x) = \sum_i^n \frac{1}{nh} K\left(\frac{x-X_i}{h}\right), f(x) \equiv \text{true density function}$$

Define:

$$L_h = \int dx (\hat{f}_h(x) - f(x))^2 = \int dx (\hat{f}_h(x))^2 - 2 \int dx (\hat{f}_h(x) f(x)) + \int dx (f(x))^2 \geq 0$$

$$\rightarrow \tilde{L}_h = \int dx (\hat{f}_h(x))^2 - 2 \int dx (\hat{f}_h(x) f(x)) \geq - \int dx (f(x))^2 = \text{constant}$$

$$\rightarrow \tilde{L}_h \approx \sum_i^n \sum_j^n \frac{1}{(nh)^2} \int K\left(\frac{x-X_i}{h}\right) K\left(\frac{x-X_j}{h}\right) dx - \frac{2}{n-1} \sum_{k \neq l}^n \sum_l^n \frac{1}{nh} K\left(\frac{X_k - X_l}{h}\right) \geq -\text{const} (*)$$

exact (arrow from $\int dx (\hat{f}_h(x))^2$ to the first sum)
approximate (arrow from $-2 \int dx (\hat{f}_h(x) f(x))$ to the second sum)

→ Find h which minimizes (*)

Advantage : non-parametric method to estimate the smoothing parameter h

Disadvantage: computing intensive n^2 operations

if there are many pairs of X_i, X_j with $X_i \approx X_j$, then $\tilde{L}_h \rightarrow -\infty$, as $h \rightarrow 0$;
tends to pick small h (undersmooth)

Bootstrap short introduction

Measure value of X n times

Estimate $f(X) = \text{mean } f(X_i)$

Need to estimate uncertainty

- *Draw new samples from the true distribution, i.e, repeat the experiment m times.*

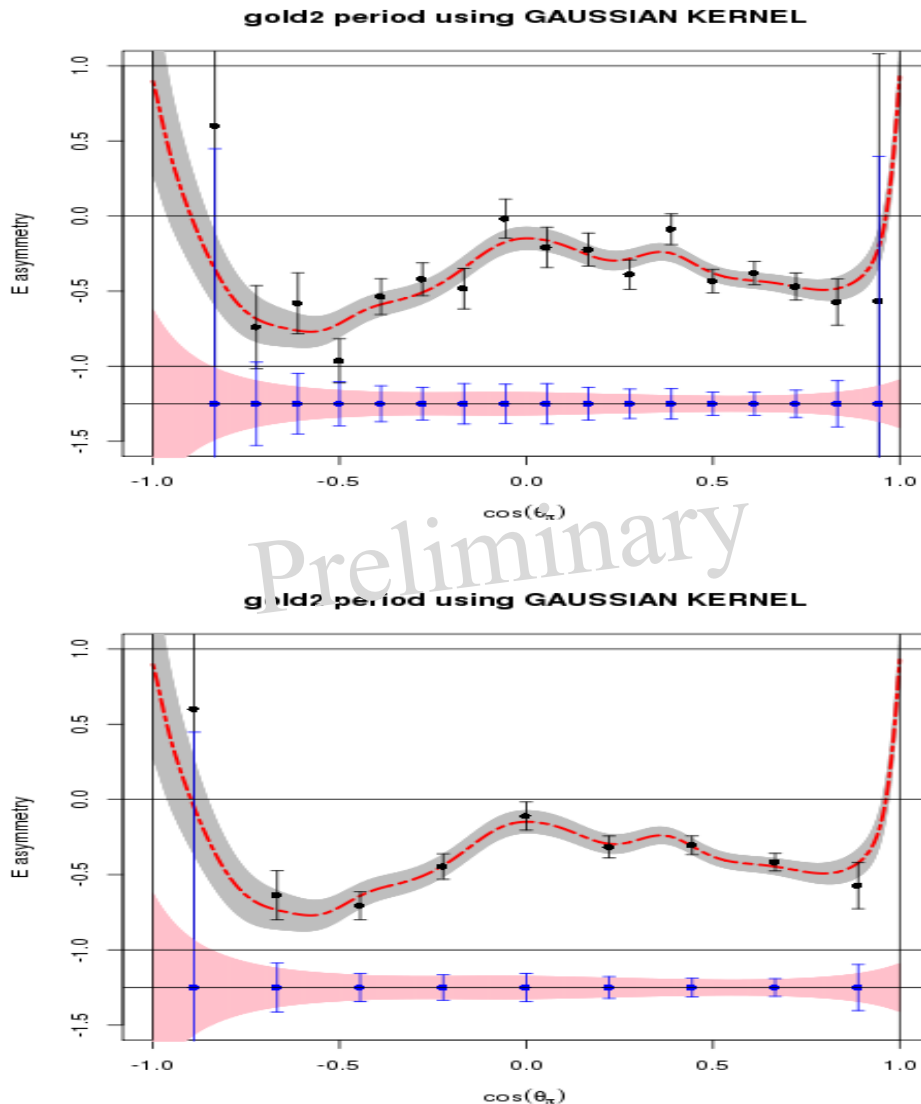
- Estimate uncertainty from the m samples

- **CAN'T** measure again, then sampling the n X_i with replacement m' times.

- *Draw new samples from an approximate distribution (the data obtained)*
- Estimate uncertainty by from the m' samples

→ this is the bootstrap method

Plotting E asymmetry



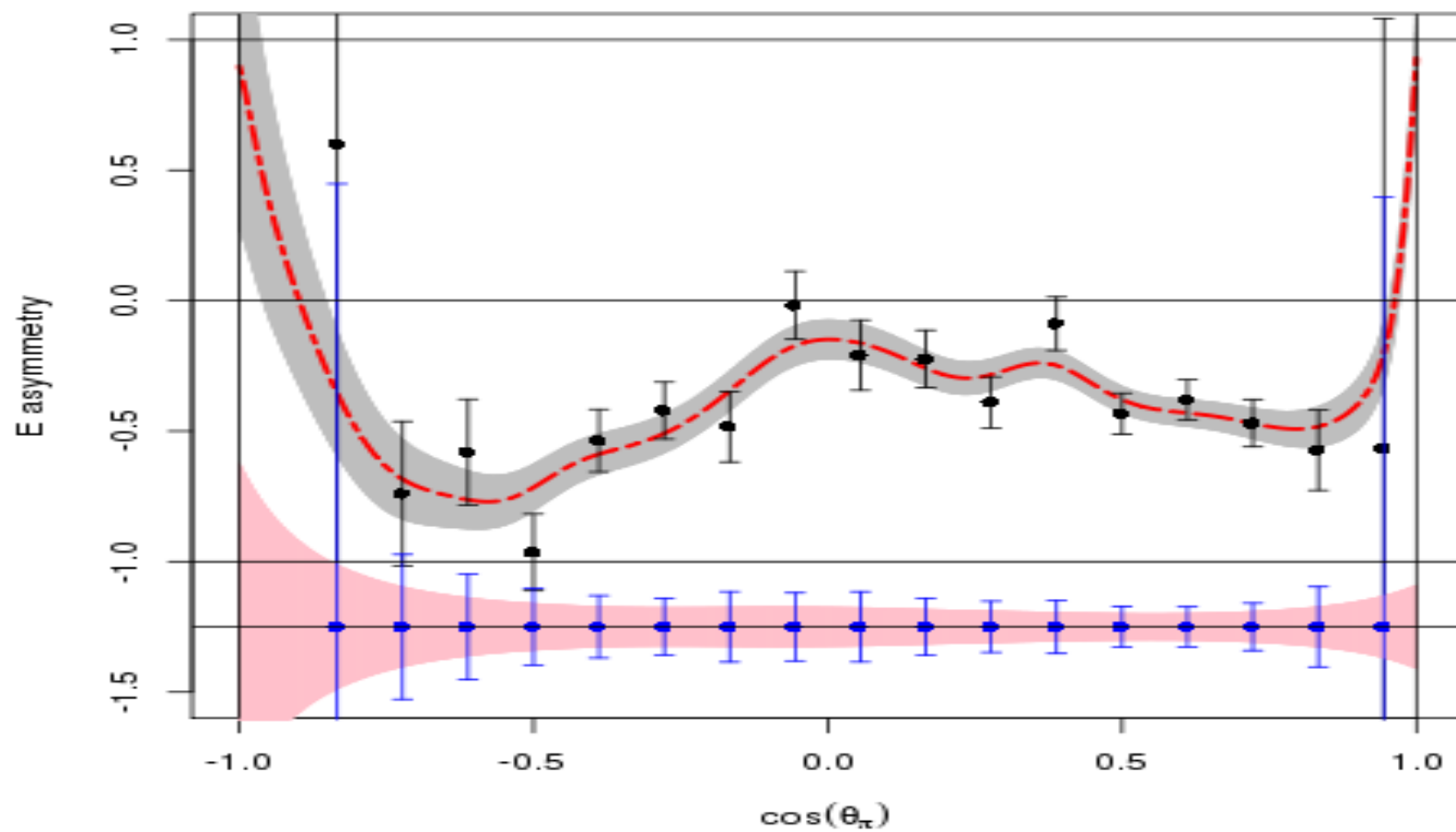
- E curve from KDE method **agrees** well with histogram method for two different sets of binning.
- Histogram: compromise between a more precise measurement (small uncertainty) or a better representation of data (more bins).
- KDE does NOT have that issue, **better choice for low statistics data**, still has **bias near boundaries**.
- KDE makes plotting a **continuous error**, or **confidence level band** possible.

Future Works:

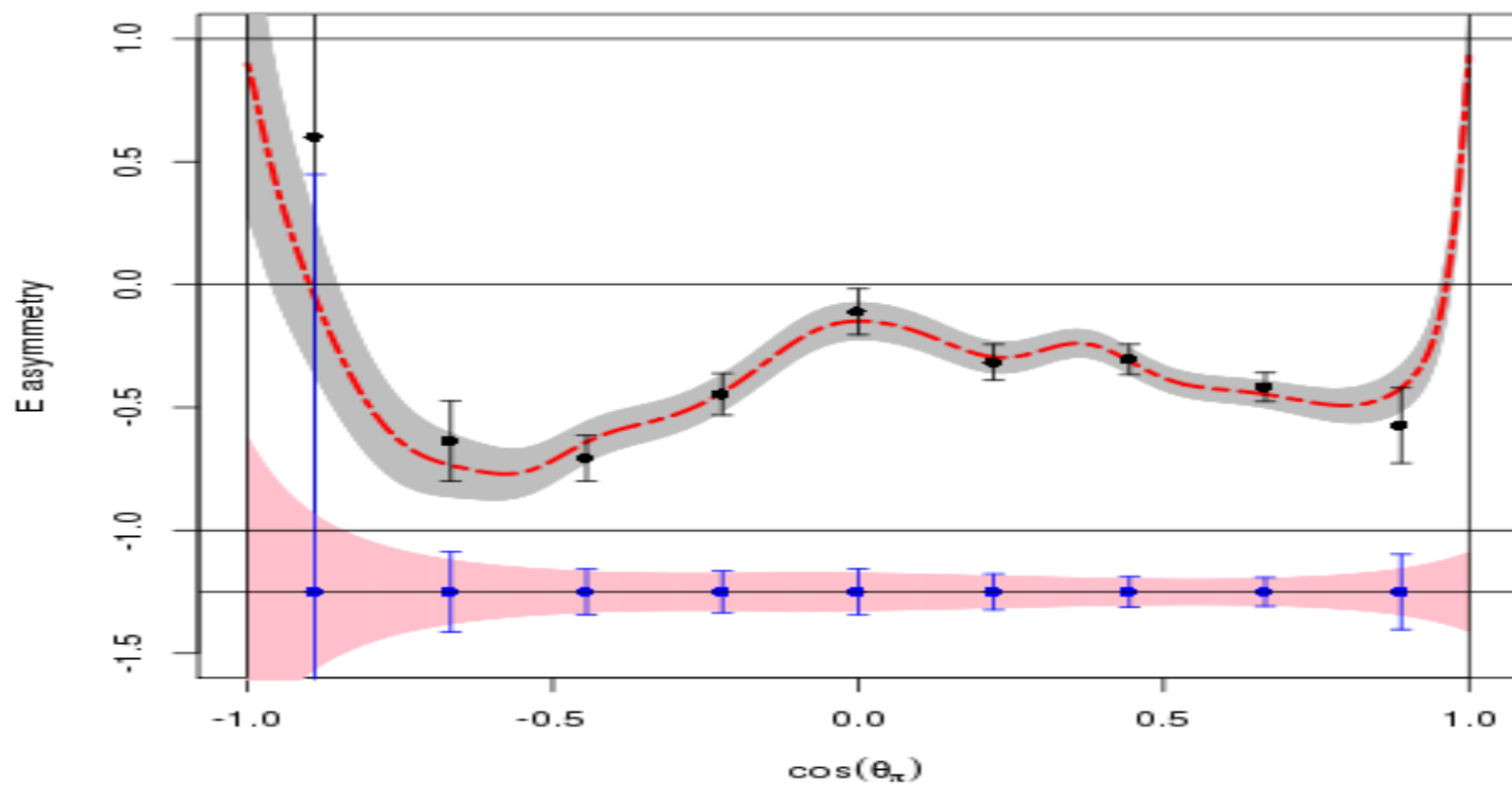
- Study and improve MC simulation for g14 run
- Use BDT for $K^0\Lambda$ and $K^0\Sigma^0$ selections
- Learn and implement bias reduction techniques for bounded data (cosine near ± 1)
- Learn more advanced KDE methods to estimate low statistic data better
- Apply these new tools to low statistics $K^0\Lambda$ and $K^0\Sigma^0$ data

Back Up

gold2 period using GAUSSIAN KERNEL



gold2 period using GAUSSIAN KERNEL



TMVA overtraining check for classifier: BDT

