Data analysis, tools with examples

Peter Križan

University of Ljubljana and J. Stefan Institute

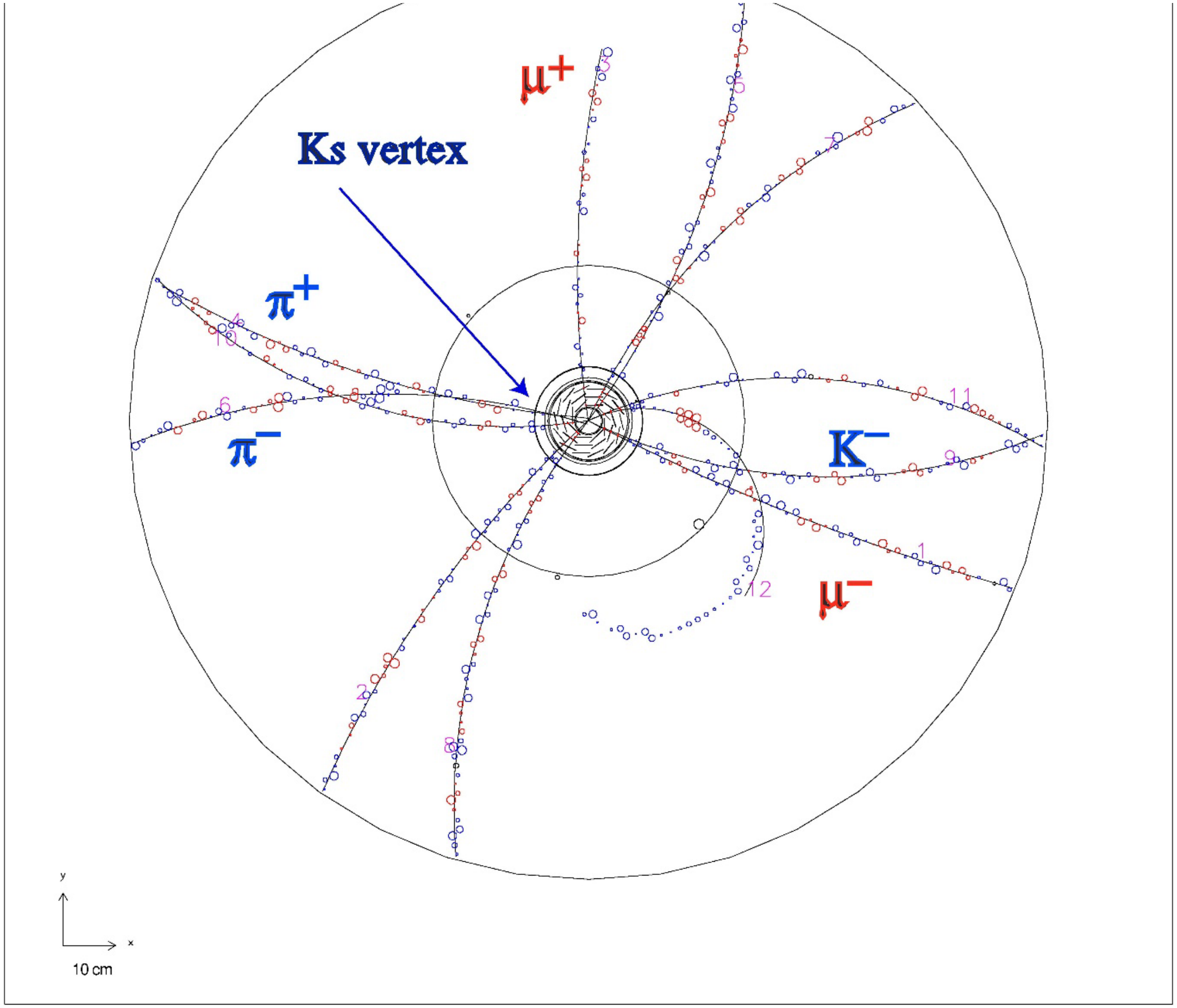
Example: selection of $B^0 \rightarrow K^0_S\, J/\psi$ events

Selection of events of the type

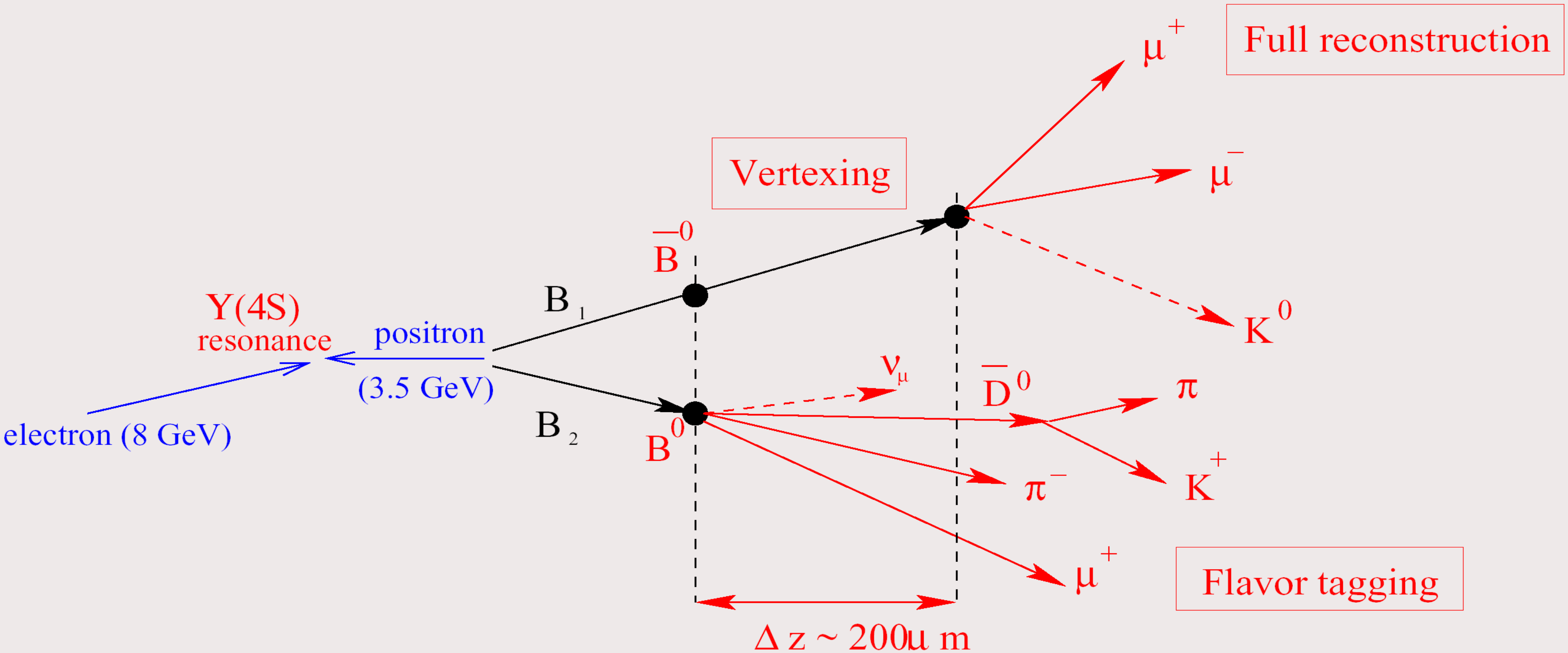
$B^0 \rightarrow K^0_S\, J/\psi$

$K^0_S \rightarrow \pi^- \pi^+$

$J/\psi \rightarrow \mu^- \mu^+$



Measurement of CP violation - continued



Determine Δt from $\Delta z = \beta\gamma c \Delta t$:

- clock start: resolution on tag side 140 μm ($\epsilon = 91\%$) - charm decays
- clock stop: resolution on CP side 75 μm ($\epsilon = 92\%$)

N.B. typically $\Delta z = \beta\gamma c \tau_B = 200$ μm

Search for unstable particles that decayed close to the production point

How do we reconstruct final states that decayed to two stable particles?

From the measured tracks calculate the invariant mass of the system (i= 1,2):

$$Mc^2 = \sqrt{(\sum E_i)^2 - (\sum \vec{p}_i)^2 c^2}$$

The candidates for the X→12 decay show up as a peak in the distribution on (mostly combinatorial) background.

The name of the game: have as little background under the peak as possible without loosing the events in the peak (=reduce background and have a small peak width).

How do we know it was precisely this reaction?

$B^0 \rightarrow K^0_S \, J/\psi$

$K^0_S \rightarrow \pi^- \pi^+$

$J/\psi \rightarrow \mu^- \mu^+$

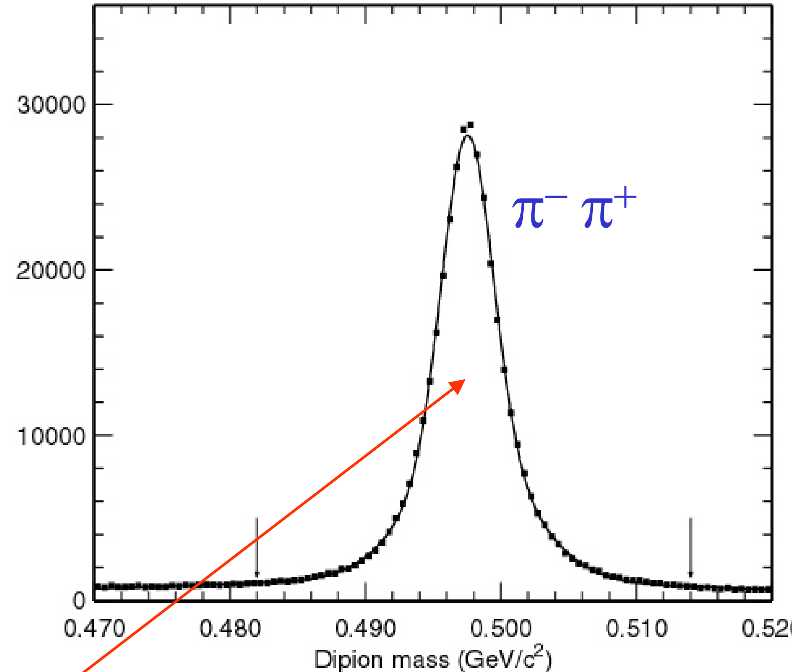
detect

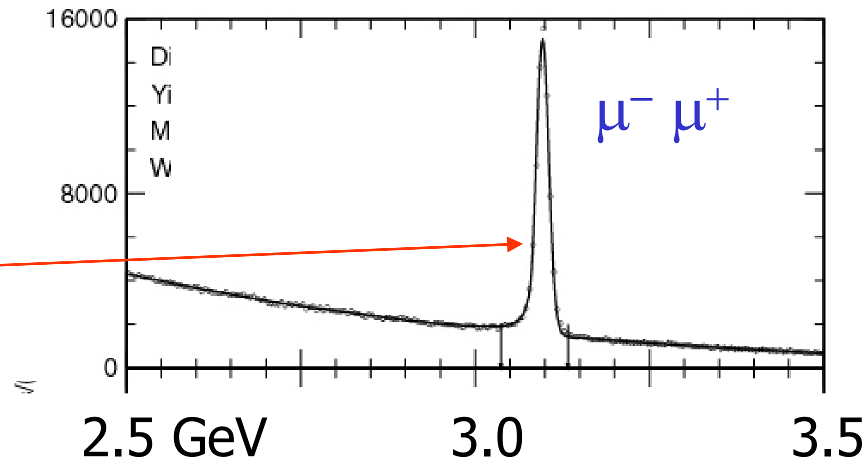
For $\pi^- \pi^+$ in $\mu^- \mu^+$ pairs we calculate the invariant mass:

$$M^2c^4=(E_1+ E_2)^2- (\vec{p}_1+ \vec{p}_2)^2$$

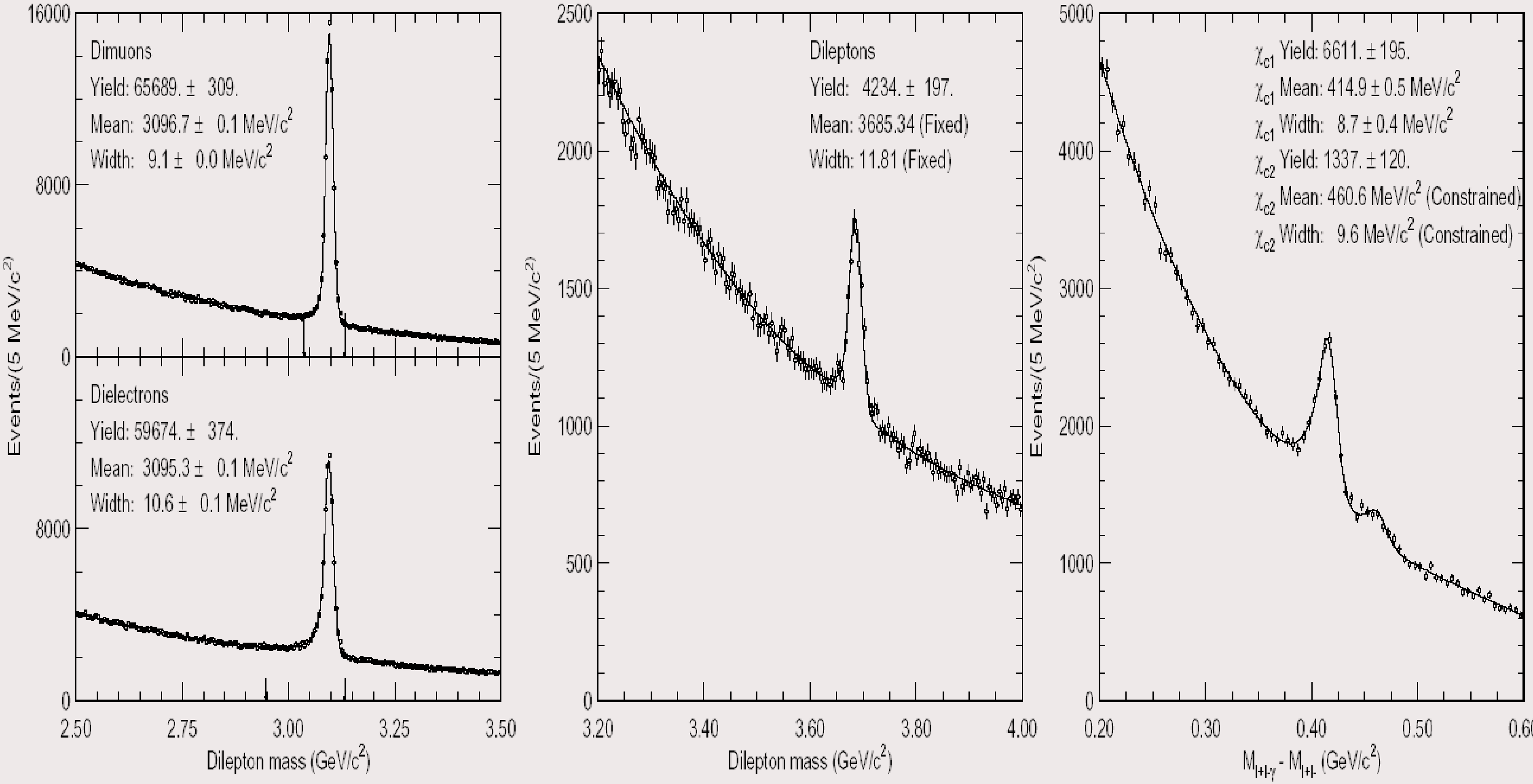
Mc^2 must be for K^0_S close to 0.5 GeV, for J/ψ close to 3.1 GeV.

Rest in the histrogram: random coincidences ('combinatorial background')





Also important: other charmonium states (in addition to J/ψ), e.g. $B^0 \rightarrow K^0_S\,\psi(2S)$



$J/\psi \rightarrow \mu^+\mu^-, e^+e^-$

$\sigma_M = 9.6(10.7)\mathrm{GeV}/c^2$

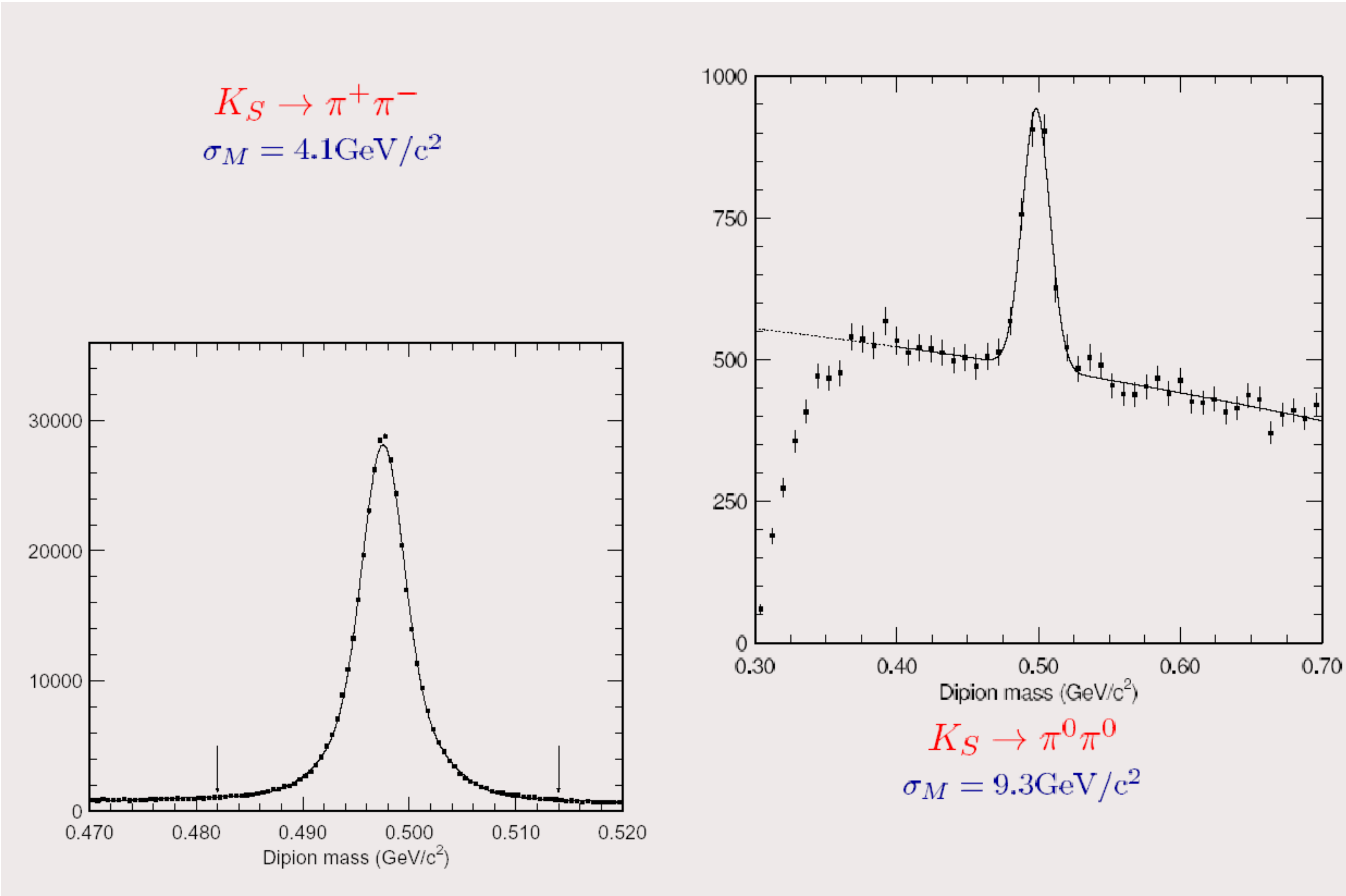
$\psi(2s) \rightarrow \mu^+\mu^-, e^+e^-$

$\sigma_M = 12.1\mathrm{GeV}/c^2$

$\chi_{c1}, \chi_{c2} \rightarrow J/\psi\gamma$

$\sigma_{\Delta M} = 7.0\mathrm{GeV}/c^2$

Also important: K^0_S decays to neutral pions



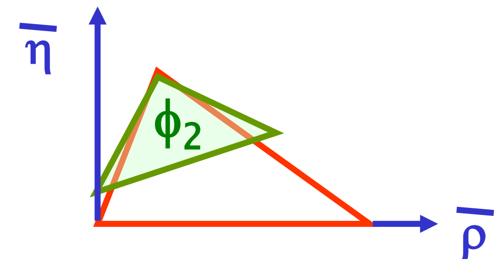
Reconstruction B meson decays

Reconstructing B meson decay at Y(4s):
Improve the resolution by taking into account that only two B mesons are produced in an Y(4s) decay.
In the expression for the invariant mass use the energy of the beam in cms (1/2 total energy in cms) instead of the reconstructed energy (which involves information on particle identification)

→ **beam constrained mass M_{bc}**

$$\boxed{M_{bc} = \sqrt{(E_{CM}/2)^2 - (\sum \vec{p}_i)^2}}$$

Example 2: CP asymmetry measurement B -> $\pi^+\pi^-$
Extraction of $\alpha(\phi_2)$



Br(B→$\pi^+\pi^-$)= 0.48 10^{-5}
-> Rare decay, have to fight against many background sources.

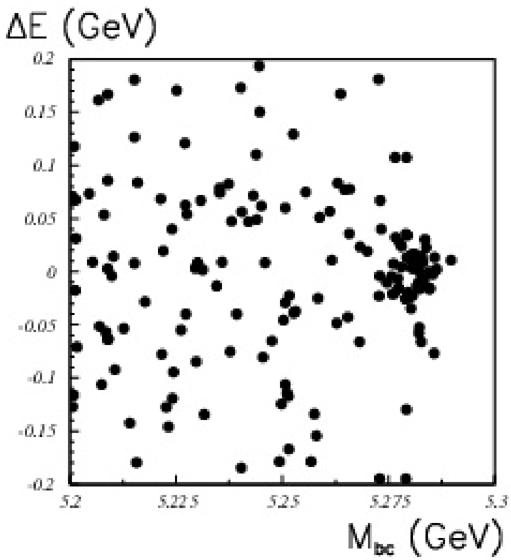
Reconstructing rare B meson decays at Υ(4s): use two variables, beam constrained mass M_{bc} and energy diference ΔE

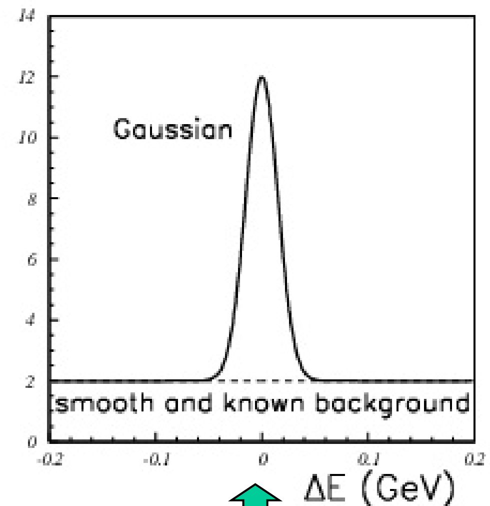
Use event topology parameters to suppress the continuum backgrounds.

Use particle identification to reduce the background from 4x more copious **B → $K^+\pi^-$** decays.

Exploit the very good momentum resolution to kinematically separate the remaining **B → $K^+\pi^-$** contribution.

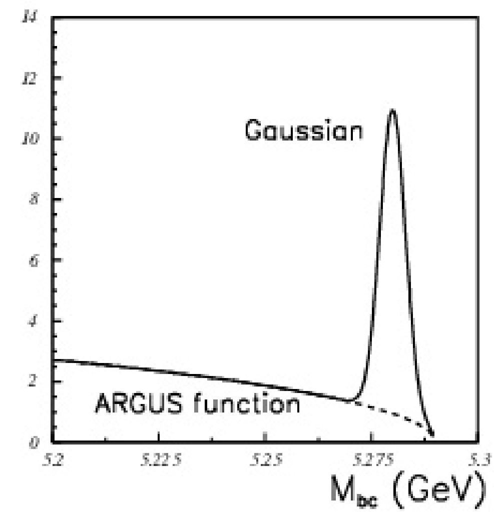
Reconstruction of rare B meson decays





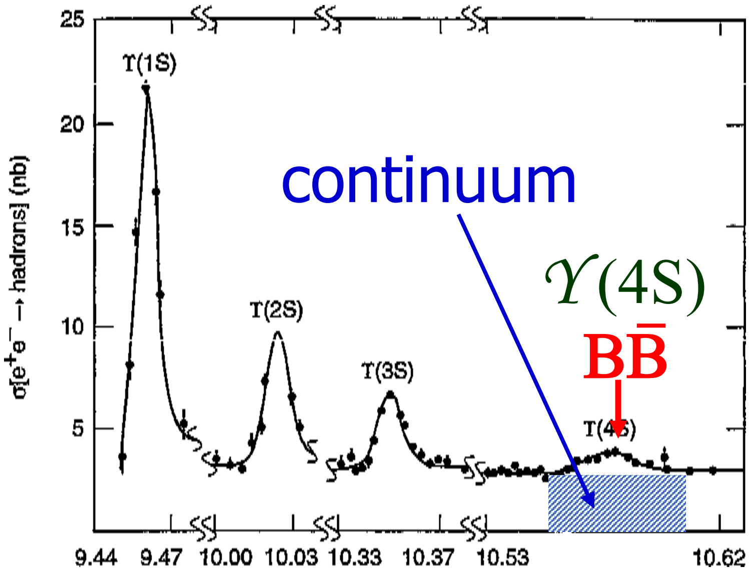
Reconstructing rare B meson decays at Υ(4s): use two variables,
beam constrained mass M_{bc}
and
energy diference ΔE

$$\boxed{\Delta E \equiv \sum E_i - E_{CM}/2}$$



$$\boxed{M_{bc} = \sqrt{(E_{CM}/2)^2 - (\sum \vec{p}_i)^2}}$$

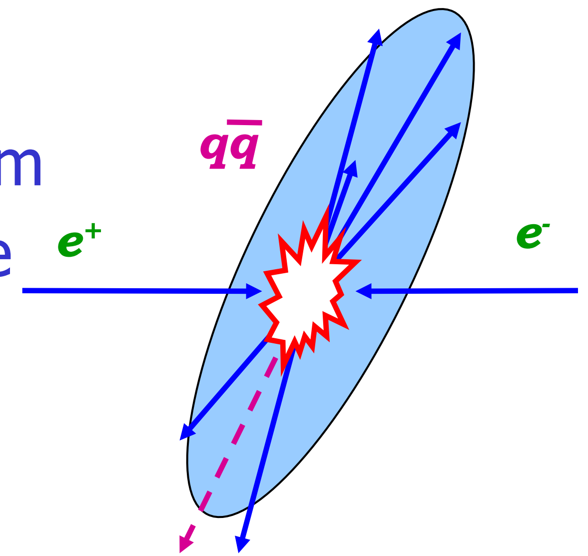
Continuum suppression



$e^+e^- \rightarrow qq$ "continuum" (~3x BB)

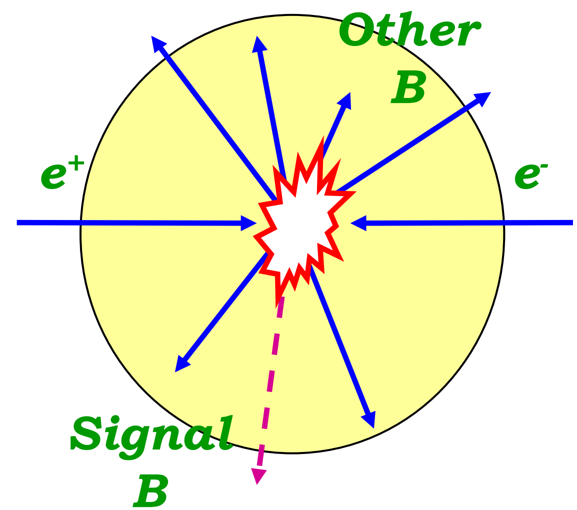
Continuum

Jet-like



BB

spherical



To suppress: use event shape variables

Continuum suppression

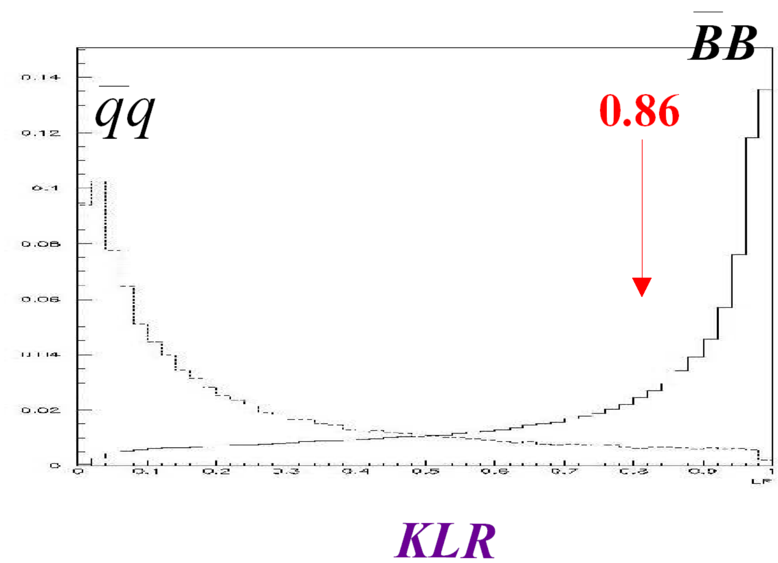
$e^+e^- \rightarrow qq$ "continuum" (~3x BB)

To suppress it use:

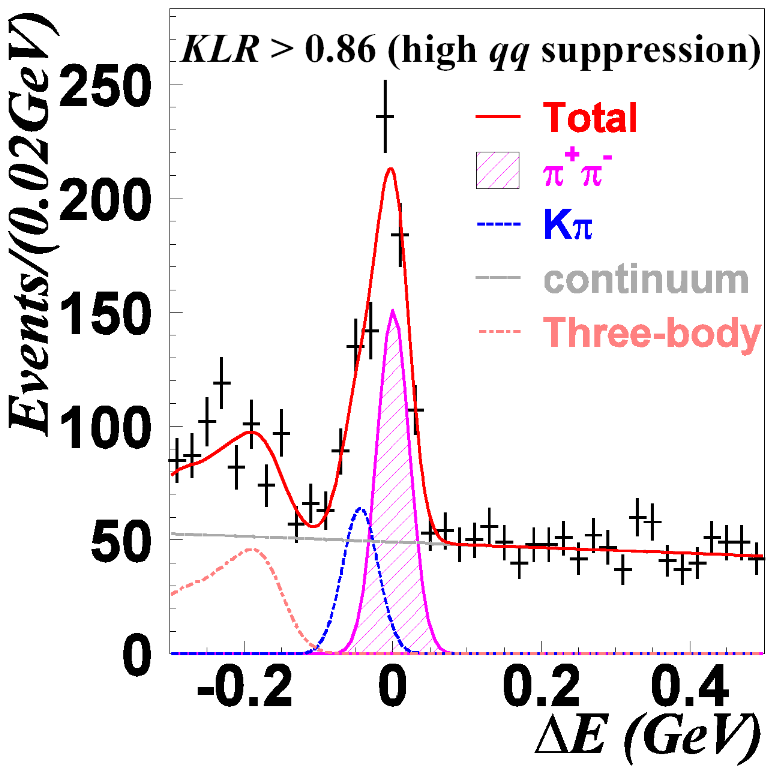
- event shape variables
- event axis direction

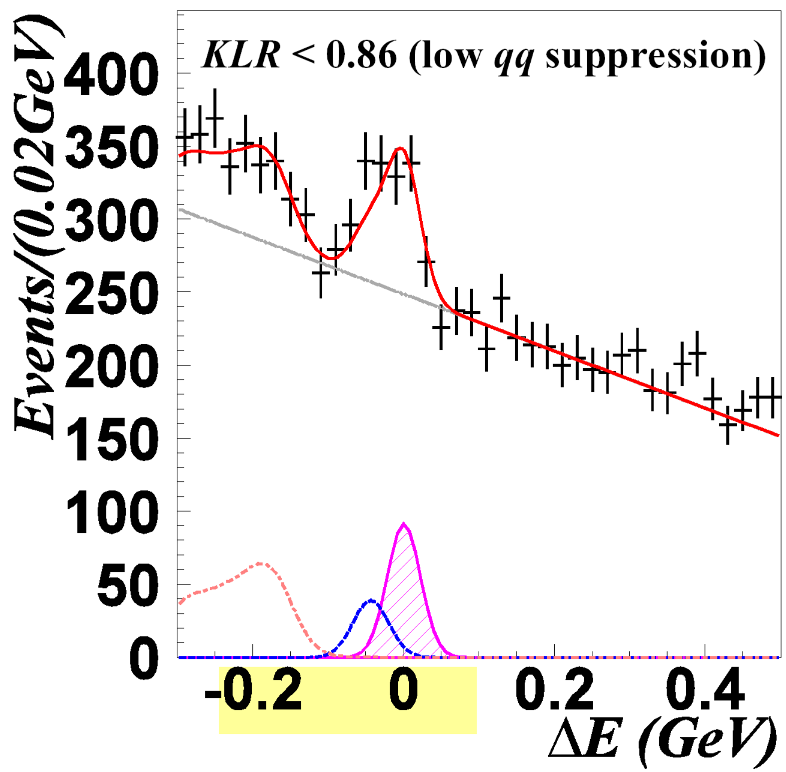
Combine to a likelihood ratio:

$$KLR \equiv \frac{\mathcal{L}_{B\overline{B}}}{\left(\mathcal{L}_{B\overline{B}} + \mathcal{L}_{q\bar{q}}\right)}$$



B → $\pi^+ \pi^-$ decays





$N_{\pi\pi} = 415 \pm 13$

$N_{\pi\pi} = 251 \pm 8$

Advanced event selection methods

Problem

Neural nets

Decision trees / Boosted decision trees (BDT)

Problem

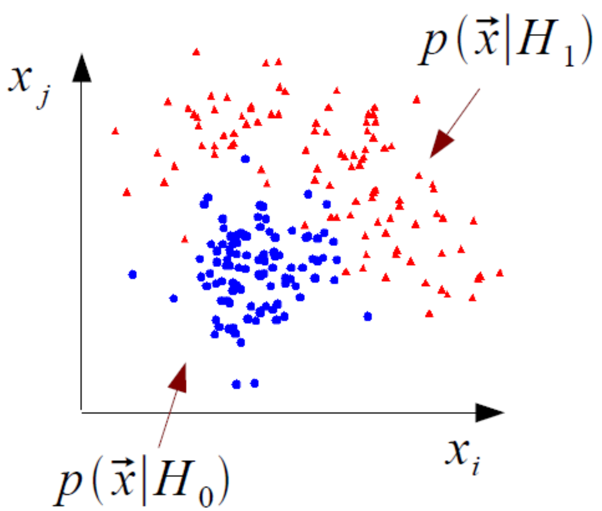
Suppose for each event we measure a set of numbers $\vec{x} = (x_1, \dots, x_n)$

$x_1 =$ jet p_{T}

$x_2 =$ missing energy

$x_3 =$ particle i.d. measure, ...

$\vec{x}$ follows some n-dimensional joint probability density, which depends on the type of event produced, i.e., was it $\mathrm{pp} \to \mathrm{t}\,\bar{\mathrm{t}}$, $\mathrm{pp} \to \tilde{\mathrm{g}}\,\tilde{\mathrm{g}}, \dots$



E.g. hypotheses (class labels) H_0, H_1, ...

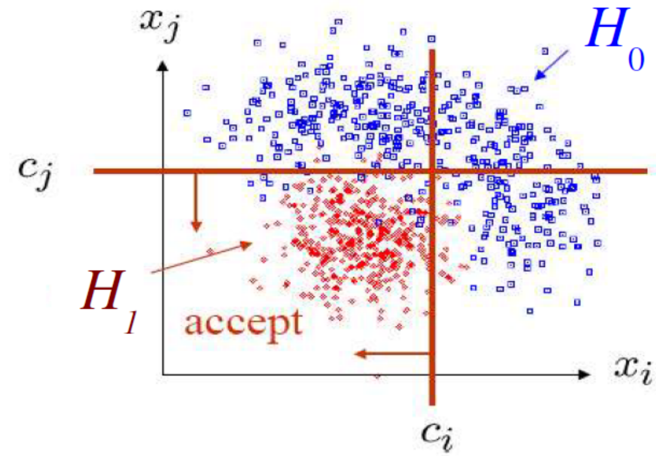
Often simply "signal", "background"

We want to separate (classify) the event types in a way that exploits the information carried in many variables.

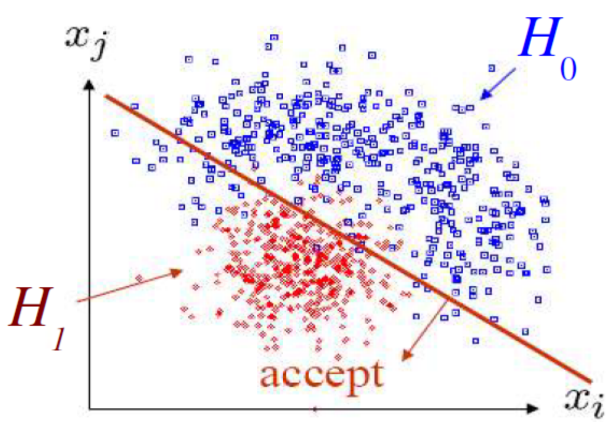
Finding an optimal decision boundary

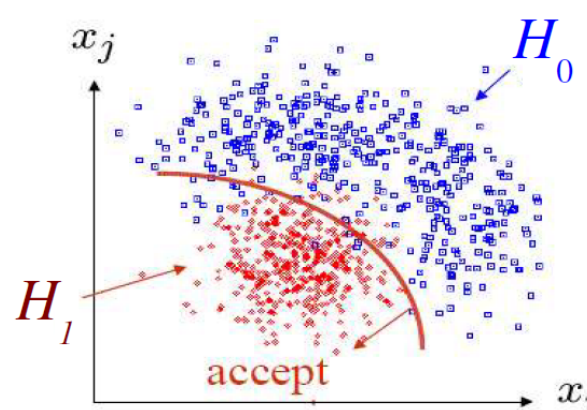
Maybe select events with "cuts":

$$x_i < c_i$$
$$x_j < c_j$$



Or maybe use some other type of decision boundary:





Goal of multivariate analysis is to do this in an "optimal" way.

Test statistics

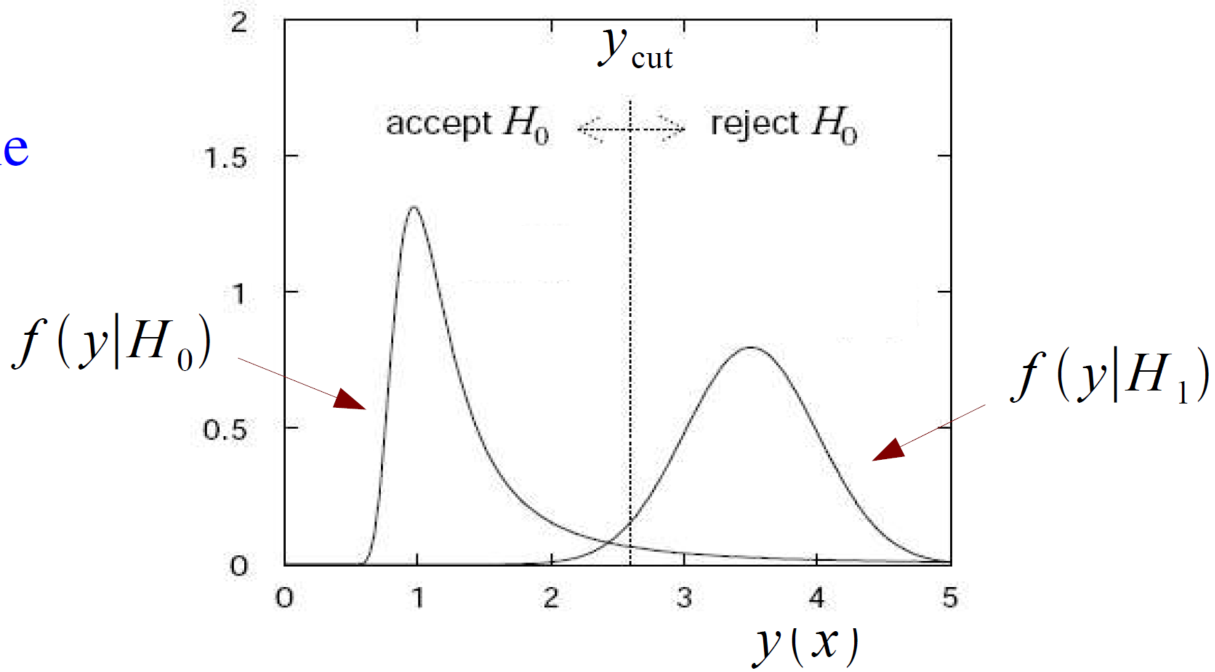
The decision boundary is a surface in the n-dimensional space of input variables, e.g., $y(\vec{x}) = \text{const}$.

We can treat the $y(x)$ as a scalar test statistic or discriminating function, and try to define this function so that its distribution has the maximum possible separation between the event types:

The decision boundary is now effectively a single cut on $y(\boldsymbol{x})$, dividing $\boldsymbol{x}$-space into two regions:

R_0 (accept H_0)

R_1 (reject H_0)

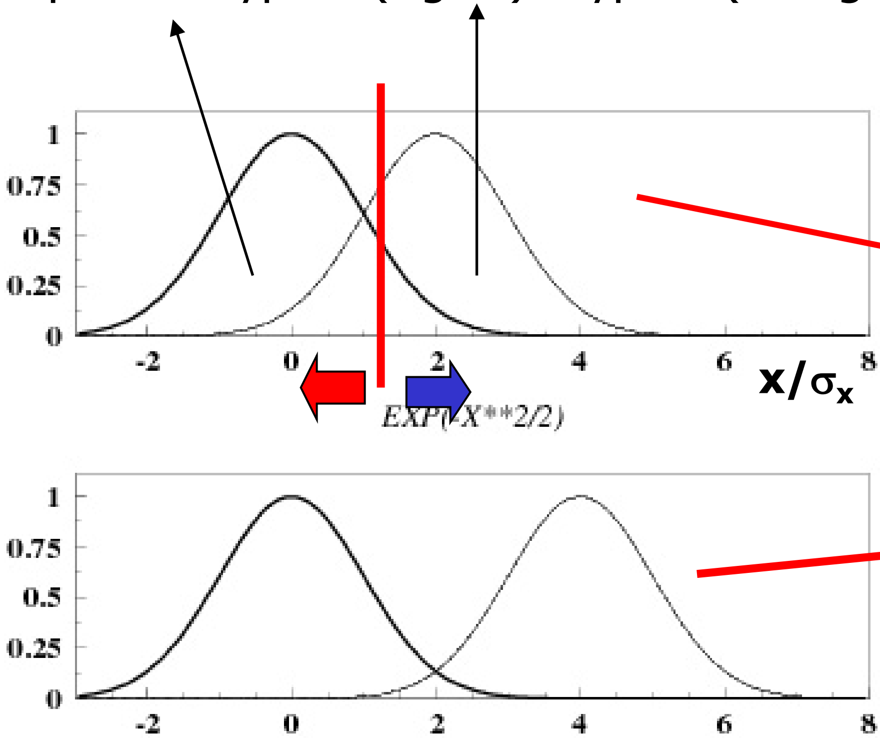


Efficiency and purity in event selection

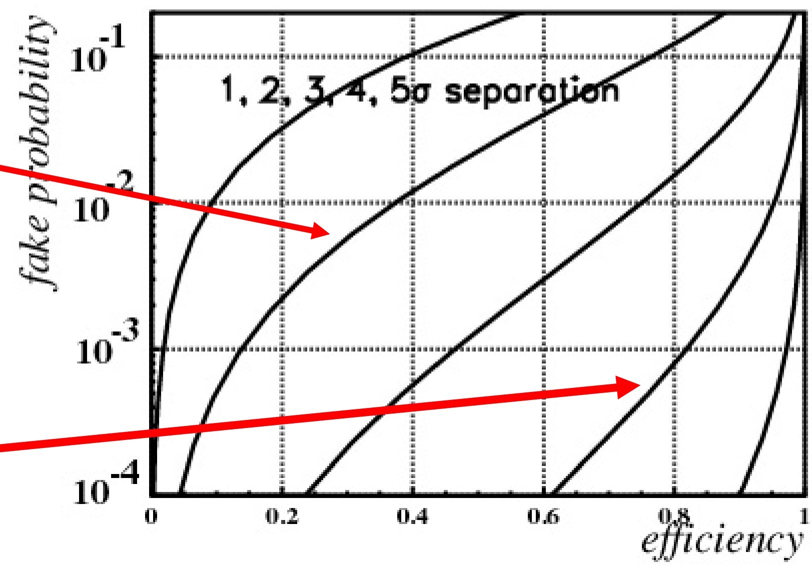
Efficiency and purity are tightly coupled!

Two examples:

process type 0 (signal) type 1 (background)

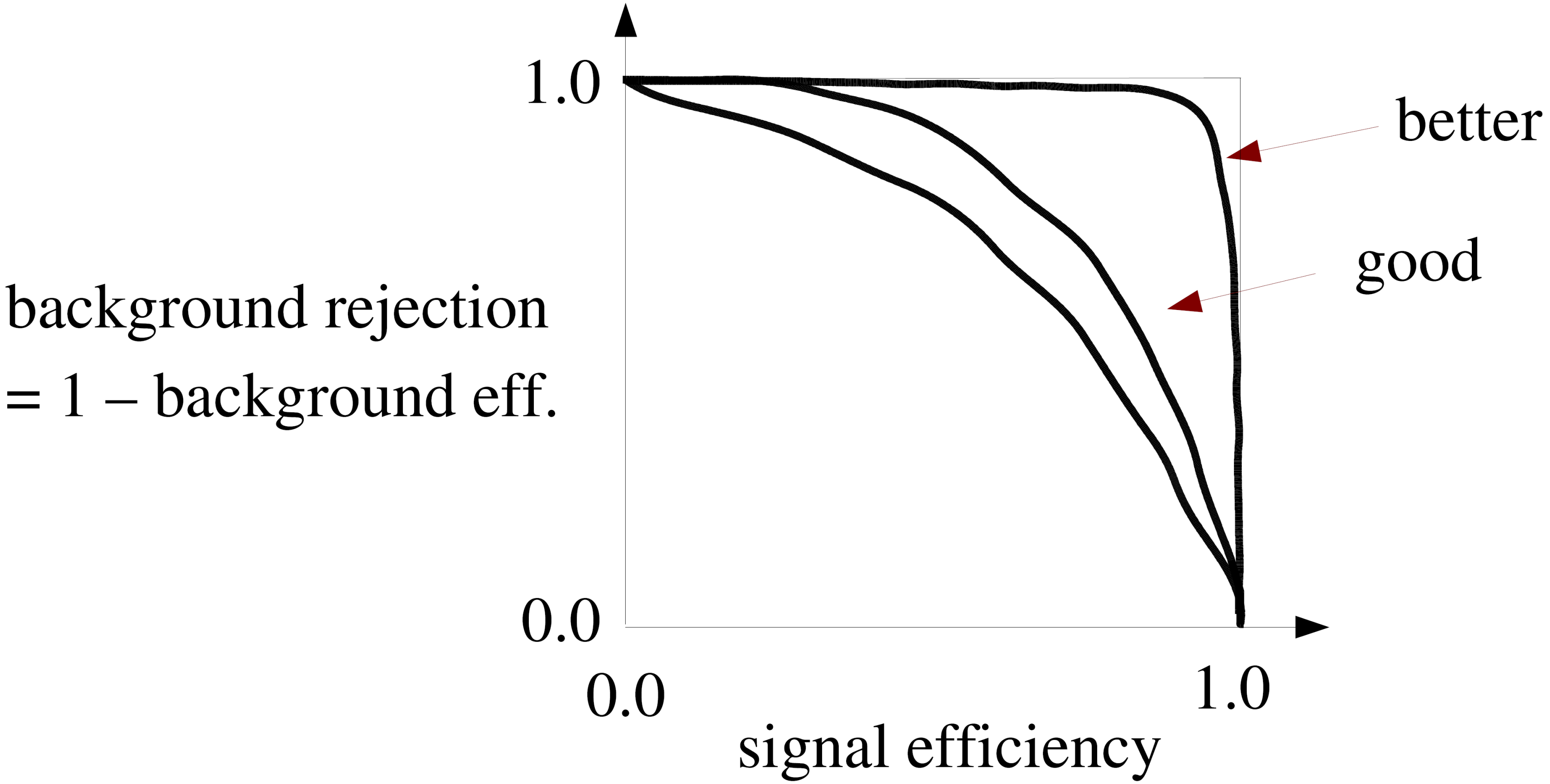


eff. vs fake probability
(for Gaussian distributions)



some discriminating variable **x**

We can characterize the quality of a classification procedure with the receiver operating characteristic (ROC curve)



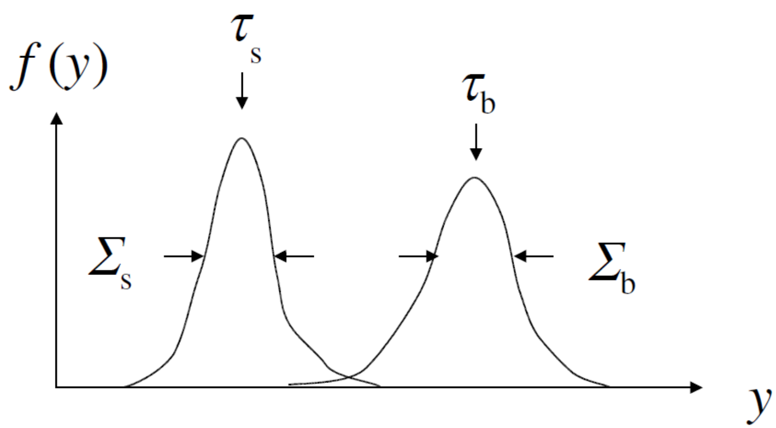
Area under curve: measure of the selection quality

Linear test statistic

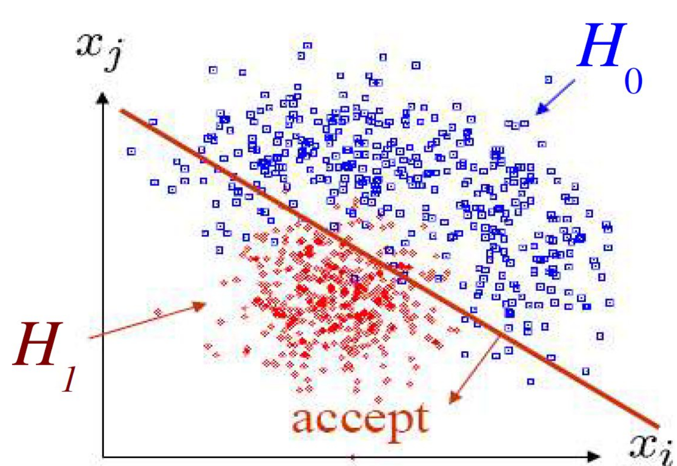
Ansatz: $y(\vec{x}) = \sum_{i=1}^{n} w_i x_i = \vec{w}^T \vec{x}$

Choose the parameters $w_1, \ldots, w_n$ so that the pdfs $f(y|\mathrm{s}), f(y|\mathrm{b})$ have maximum 'separation'. We want:

large distance between mean values, small widths

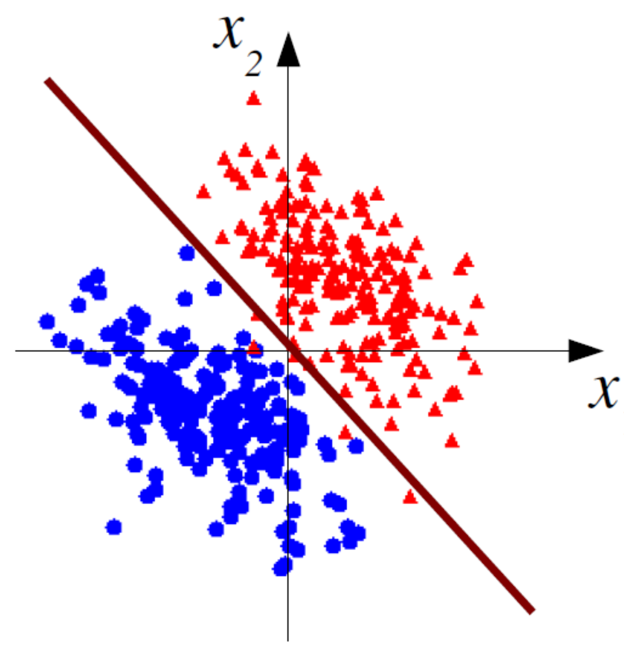


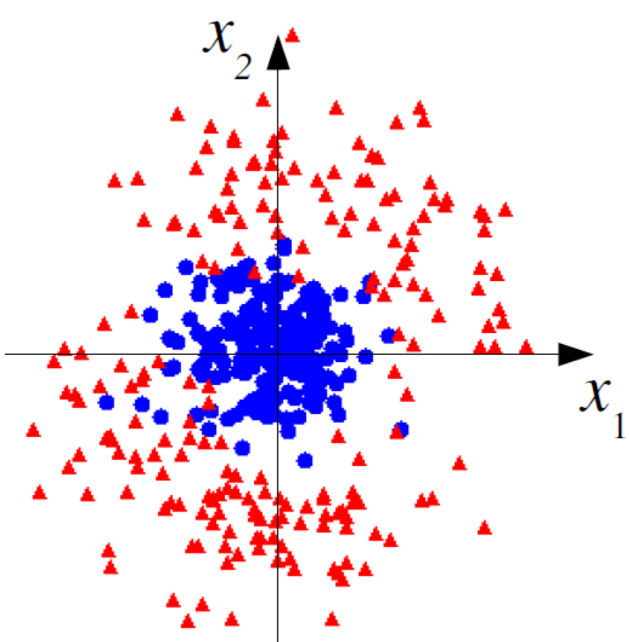
→ Fisher: maximize $J(\vec{w}) = \frac{(\tau_s - \tau_b)^2}{\Sigma_s^2 + \Sigma_b^2}$



Linear decision boundaries

A linear decision boundary is only optimal when both classes follow multivariate Gaussians with equal covariances and different means.





For some other cases a linear boundary is almost useless.

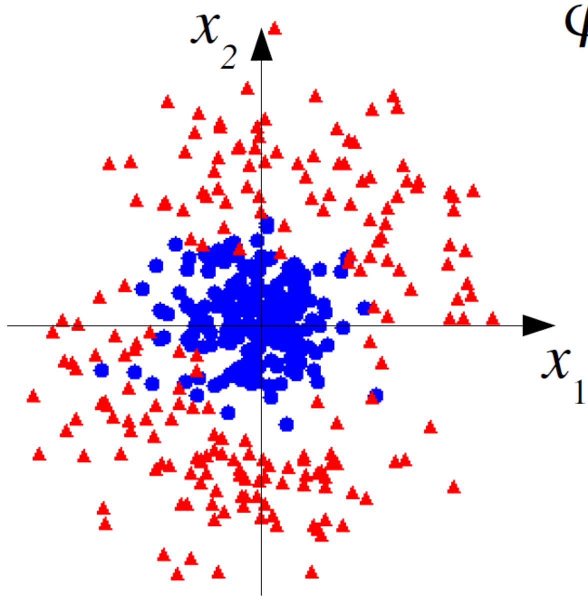
Nonlinear transformation of inputs

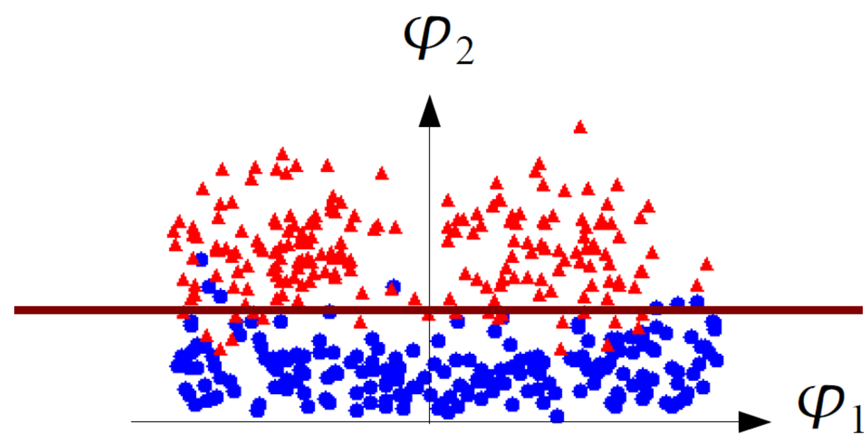
We can try to find a transformation, $x_1, \ldots, x_n \rightarrow \varphi_1(\vec{x}), \ldots, \varphi_m(\vec{x})$ so that the transformed "feature space" variables can be separated better by a linear boundary:

$$\varphi_1 = \tan^{-1}(x_2/x_1)$$

$$\varphi_2 = \sqrt{x_1^2 + x_2^2}$$

Here, guess fixed basis functions (no free parameters)



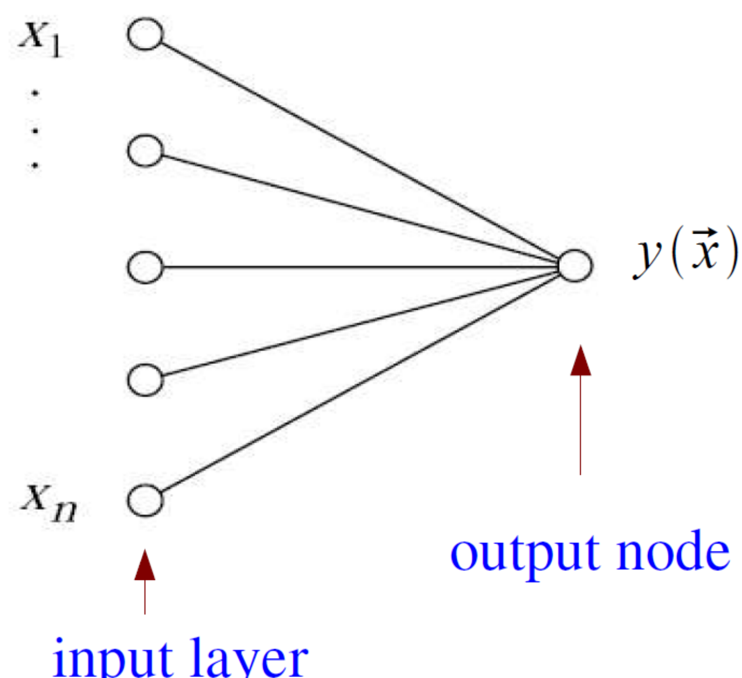


Neural nets

Define the discriminant using $y(\vec{x}) = h\left(w_0 + \sum_{i=1}^{n} w_i x_i\right)$

where h is a nonlinear, monotonic activation function; we can use e.g. the logistic sigmoid $h(x) = (1 + e^{-x})^{-1}$.

If the activation function is monotonic, the resulting $y(\boldsymbol{x})$ is equivalent to the original linear discriminant. This is an example of a "generalized linear model" called the single layer perceptron.



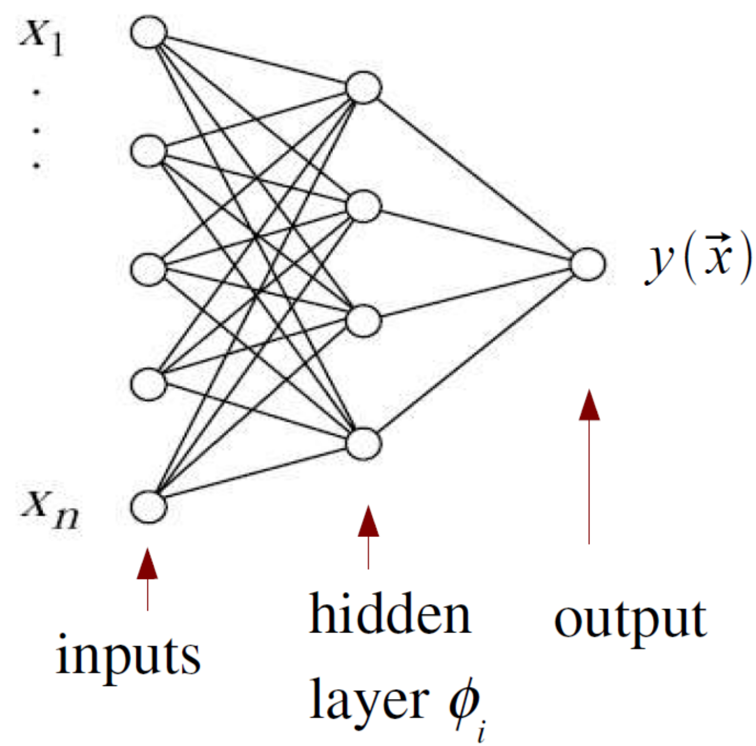
The multilayer perceptron

Now use this idea to define not only the output $y(\boldsymbol{x})$, but also the set of transformed inputs $\varphi_1(\vec{x}), \ldots, \varphi_m(\vec{x})$ that form a "hidden layer":

Superscript for weights indicates layer number

$$\varphi_i(\vec{x}) = h\left(w_{i0}^{(1)} + \sum_{j=1}^{n} w_{ij}^{(1)} x_j \right)$$

$$y(\vec{x}) = h\left(w_{10}^{(2)} + \sum_{j=1}^{n} w_{1j}^{(2)} \varphi_j(\vec{x}) \right)$$



This is the multilayer perceptron, our basic neural network model; straightforward to generalize to multiple hidden layers.

Network training

The type of each training event is known, i.e., for event a we have:

$\vec{x}_a = (x_1, \ldots, x_n)$ the input variables, and

$t_a = 0, 1$ a numerical label for event type ("target value")

Let $\boldsymbol{w}$ denote the set of all of the weights of the network. We can determine their optimal values by minimizing a sum-of-squares "error function"

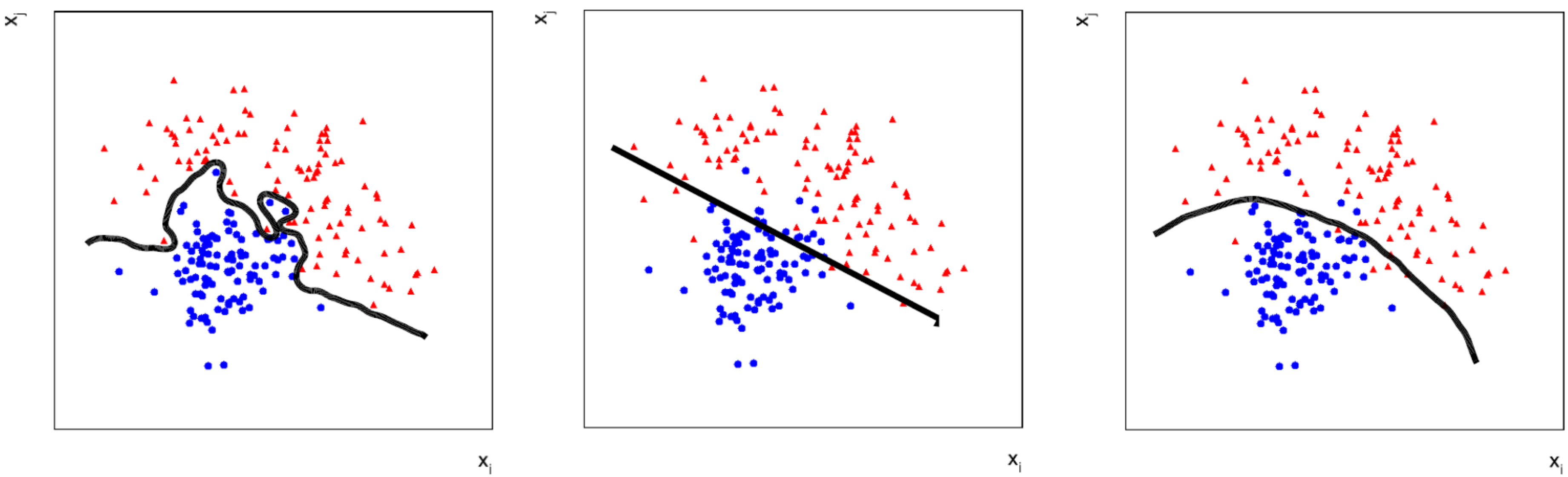
$$E(\boldsymbol{w}) = \frac{1}{2} \sum_{a=1}^{N} |y(\vec{x}_a, \boldsymbol{w}) - t_a|^2 = \sum_{a=1}^{N} E_a(\boldsymbol{w})$$

Contribution to error function from each event

Bias – variance trade-off

For a finite amount of training data, an increasing number of network parameters (layers, nodes) means that the estimates of these parameters have increasingly large statistical errors (variance, overtraining).

Having too few parameters doesn't allow the network to exploit the existing nonlinearities, i.e., it has a bias.



high variance

high bias

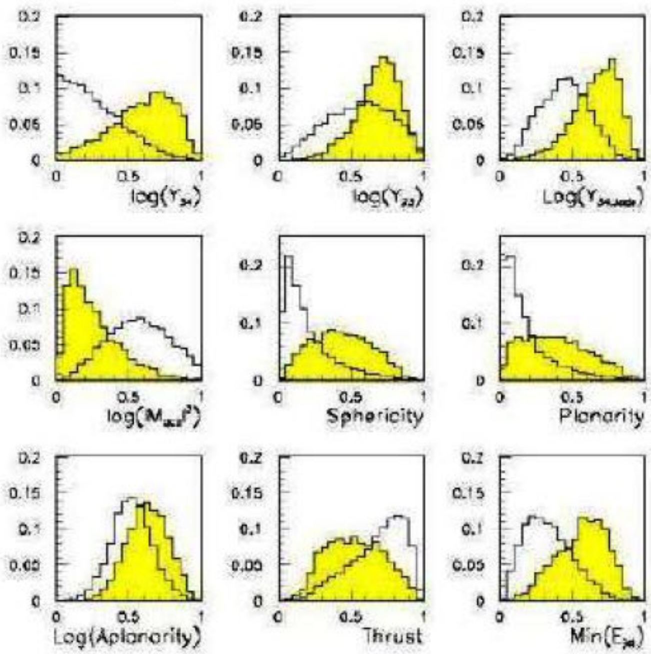
good trade-off

One of the early examples in particle physics

Neural network example from LEP II

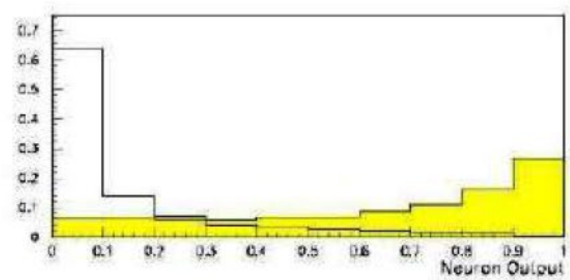
Signal: $e^+e^- \rightarrow W^+W^-$ (often 4 well separated hadron jets)

Background: $e^+e^- \rightarrow qqgg$ (4 less well separated hadron jets)



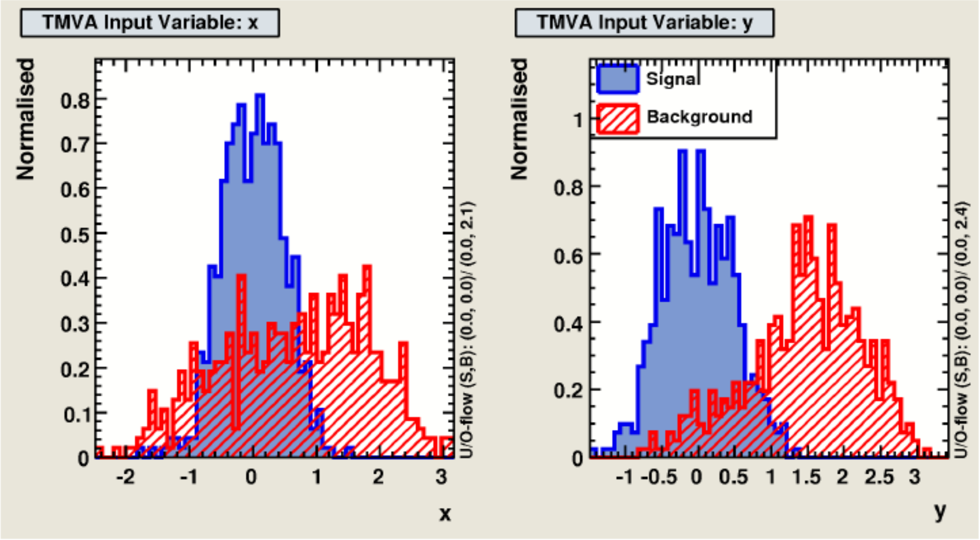
← input variables based on jet structure, event shape, ...
none by itself gives much separation.

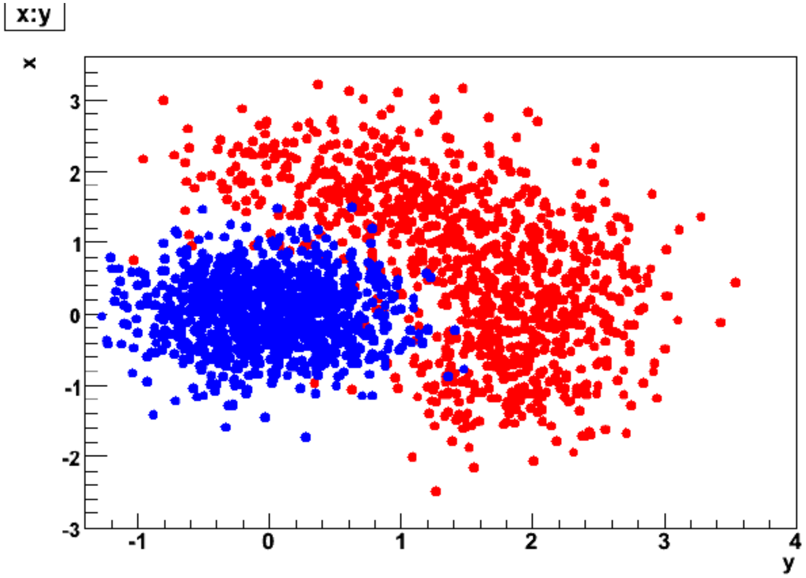
Neural network output does better...



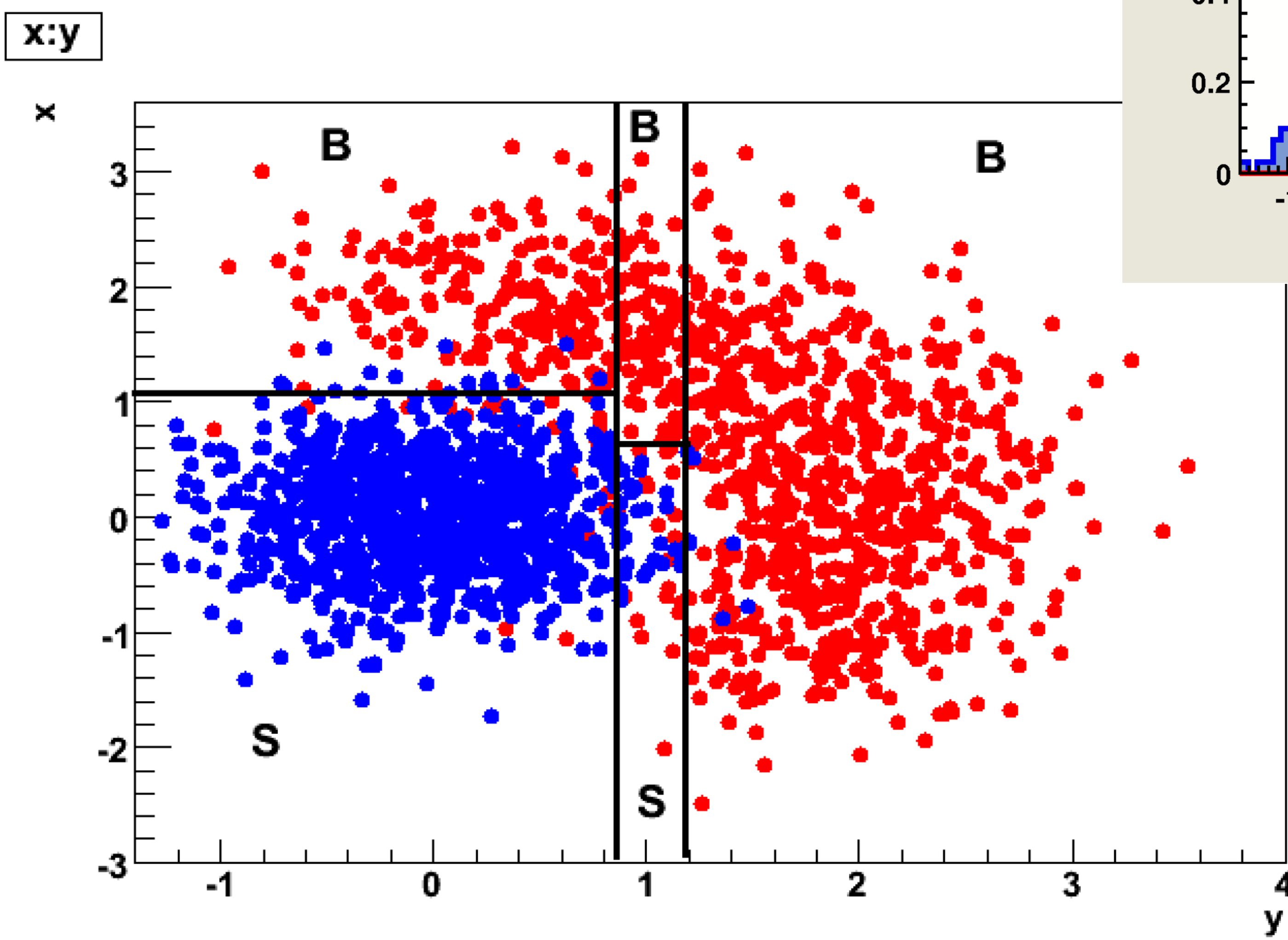
(Garrido, Juste and Martinez, ALEPH 96-144)

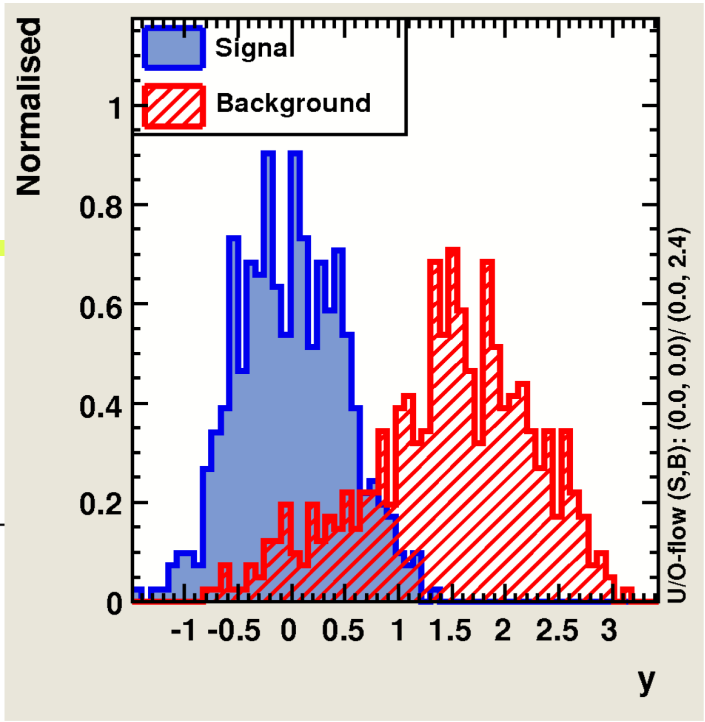
Decision trees



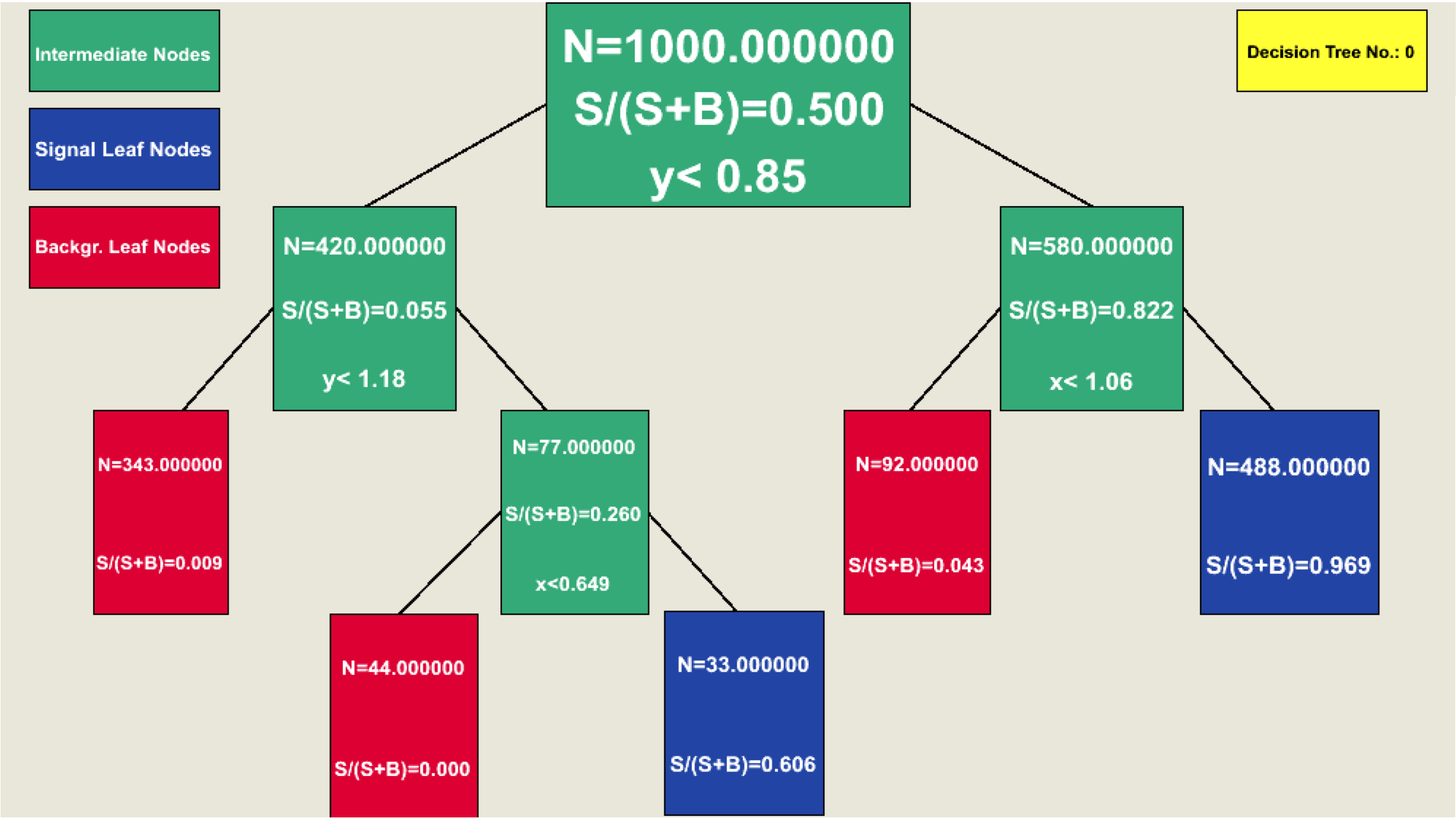


Decision Tree (DT) example

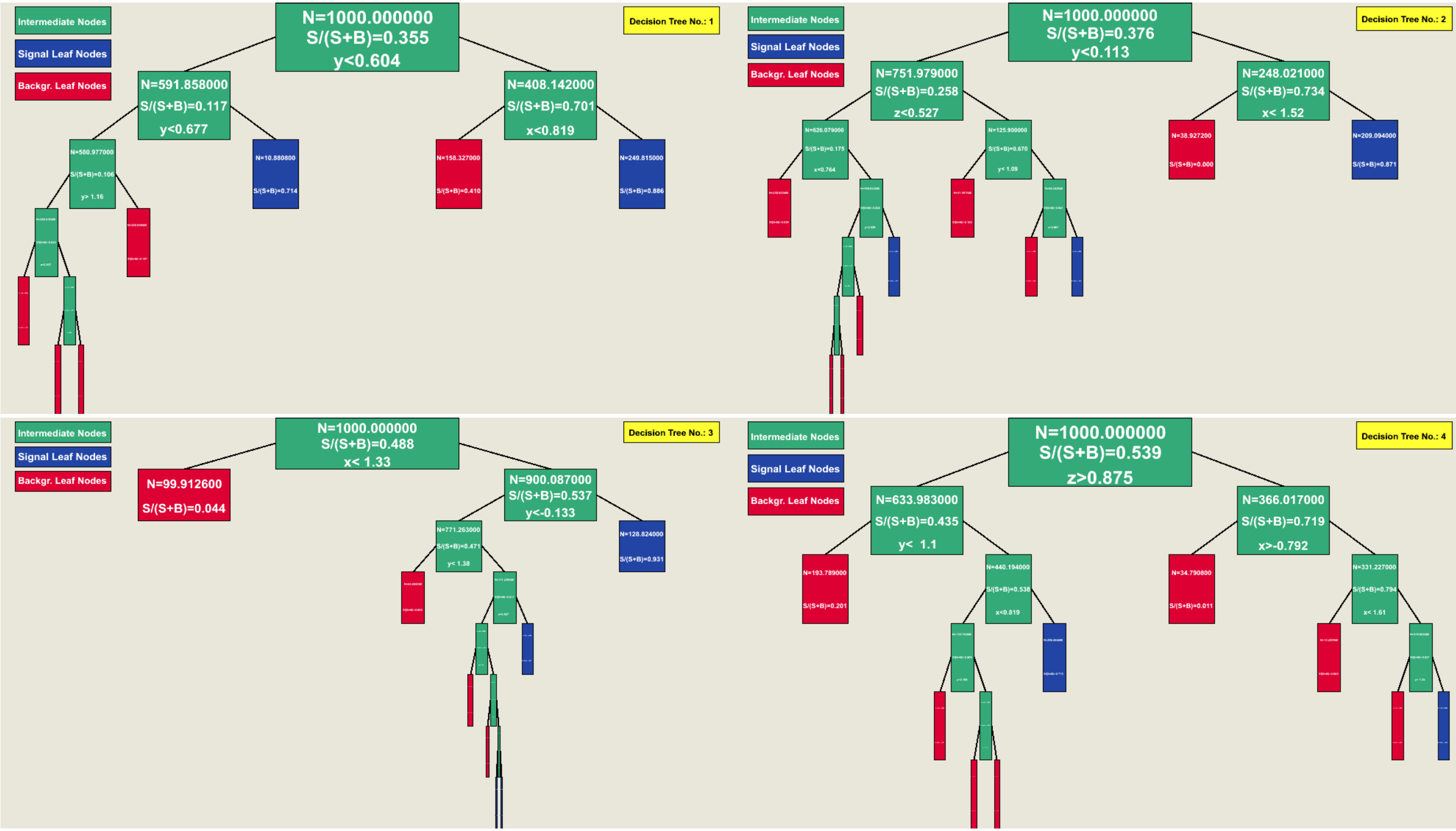




Decision Tree (DT) example



Decision Tree (DT) example: specialized trees



Boosted Decision Tree (BDT)

- Train classifier T_1 on N events
- Train T_2 on new N-sample, half of which misclassified by T_1
- Build T_3 on events where T_1 and T_2 disagree
- Boosted classifier: MajorityVote(T_1, T_2, T_3)