*T*MVA

– toolkit for parallel multivariate data analysis –

Andreas Höcker (ATLAS), Helge Voss (LHCb), Kai Voss (ATLAS)

Andreas.Hoecker@cern.ch

CATPhys Meeting, CERN, March 13, 2006

TMVA http://tmva.sourceforge.net/

MVA Experience

- Any HEP data analysis uses multivariate techniques (also cuts are MV)
- Often analysts use custom tools, without much comparison
  - MVAs tedious to implement, therefore few true comparisons between methods !
  - most accepted: cuts
  - also widely understood and accepted: likelihood (probability density estimators – PDE)
  - often disliked, but more and more used (LEP, BABAR): Artificial Neural Networks
  - much used in BABAR and Belle: Fisher discriminants
  - introduced by D0 (for electron id): H-Matrix
  - used by MiniBooNE and recently by BABAR: Boosted Decision Trees
- All interesting methods … but how to dissipate the widespread skepticism ?
  - black boxes !
  - what if the training samples incorrectly describe the data ?
  - how can one evaluate systematics ?
  - you want to use MVAs, but how to convince your Professor ?

MVA Experience

- All interesting methods … but how to dissipate the widespread skepticism ?

➡ black boxes !

Certainly, cuts are transparent, so
- if cuts are competitive (rarely the case) → use them
- in presence of correlations, cuts loose transparency

➡ what if the training samples incorrectly de-scribe the data ?

Not good, but not necessarily a huge problem:
- performance on real data will be worse than training results
- however: bad training does not create a bias !
- only if the training efficiencies are used in data analysis → bias
- optimized cuts are <u>not</u> in general less vulnerable to systematics (on the contrary !)

➡ how can one evaluate systematics ?

There is no principle difference in systematics evaluation between single variables and MVAs
- need control sample for MVA output (not necessarily for each input variable)

➡ you want to use MVAs, but how to convince your Professor ?

Tell her/him you'll miss the Higgs

Better: show him the *T*MVA results !

ATLAS Analysis in a Nutshell

1. Full event reconstruction information → ESD
   - assume that it will be impossible to analyse data with these
2. High level reconstruction information → AOD
   - used for analysis
3. Apply high efficient first path selection on AODs
   - create specific analysis objects: EventView, CBNT, …
4. Select personalized analysis objects
   - ntuples, …
5. Apply analysis tools
   - multivariate analysis to purify signal (*T*MVA)
   - count, or perform unbinned maximum likelihood fit to extract event yield (RooFit)

What is *T*MVA

- **Toolkit for Multivariate Analysis (*T*MVA):** provides a ROOT-integrated environment for the **parallel** processing and evaluation of MVA techniques to discriminate *signal* from *background* samples.

- *T*MVA presently includes (ranked by complexity):
  - Rectangular cut optimisation
  - Correlated likelihood estimator (PDE approach)
  - Multi-dimensional likelihood estimator (PDE - range-search approach)
  - Fisher (and Mahalanobis) discriminant
  - H-Matrix approach (χ^2 estimator)
  - Artificial Neural Network (two different implementations)
  - Boosted Decision Trees

- The *T*MVA analysis provides **training**, **testing** and **evaluation** of the MVAs

- The training results are written to specific **weight files**

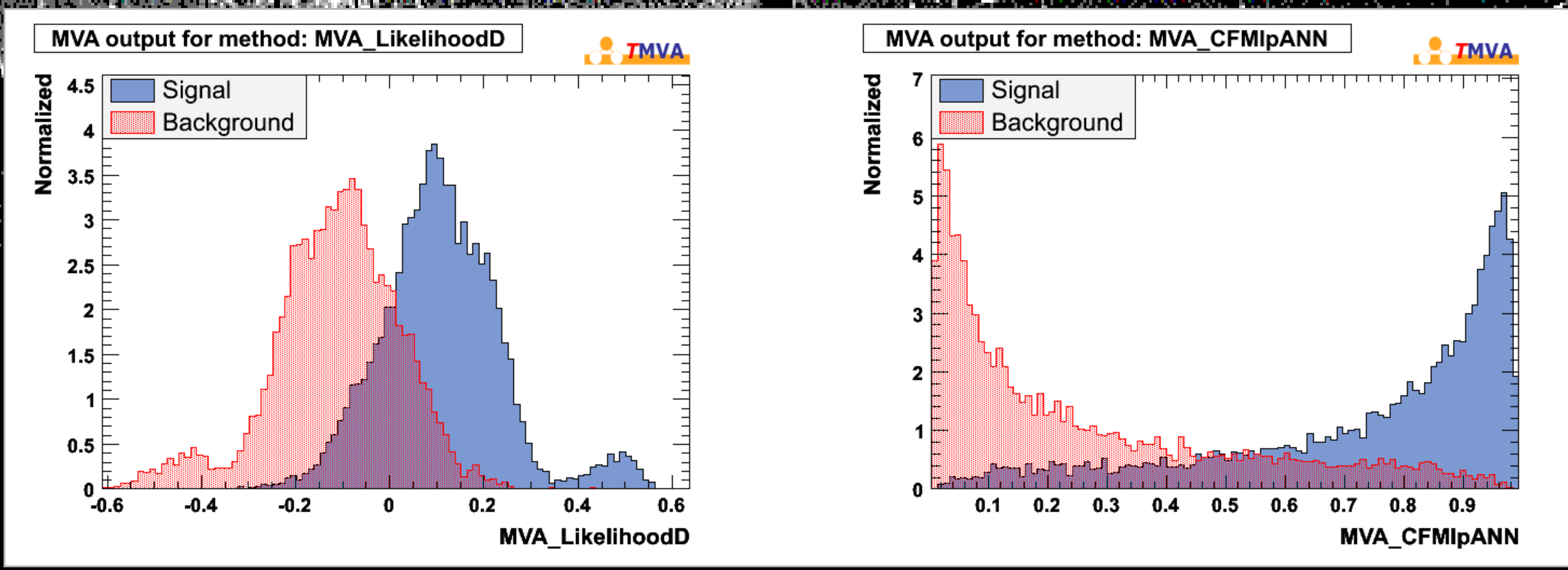
- The weight files are read by dedicated **reader class** for actual MVA analysis

- *T*MVA supports multiple MVAs as a function of up to two variables (*e.g.*, η, p_T)

*T*MVA Technicalities

- *T*MVA is a sourceforge (SF) package to accommodate world-wide access
  - code can be downloaded as *tar* file, or via anonymous cvs access
  - home page: http://tmva.sourceforge.net/
  - SF project page: http://sourceforge.net/projects/tmva
  - view CVS: http://cvs.sourceforge.net/viewcvs.py/tmva/TMVA/
  - mailing lists: http://sourceforge.net/mail/?group_id=152074
- *T*MVA is written in C++ and heavily uses ROOT functionality
  - Are in contact with ROOT developers (R. Brun et al.) for possible integration in ROOT
- *T*MVA is modular
  - training, testing and evaluation factory iterates over all available (and wanted) methods
  - though the current release is stable, we think that the improvement and extension of the methods is a continuous process
  - each method has specific options that can be set by the user for optimisation
  - ROOT scripts are provided for all relevant performance analysis
- *We enthusiastically welcome new users, testers and developers ☺*

*T*MVA Methods



Cut Optimisation

- Simplest method: cut in rectangular volume using N_{var} input variables

$$x_{\text{cut},i_{\text{event}}} \in (0,1) = \bigcap_{v \in \{\text{variables}\}} \left\{ x_{v,i_{\text{event}}} \subset \left[x_{v,\min}, x_{v,\max} \right] \right\}$$

- Usually training files in *T*MVA do not contain *realistic* signal and background abundance → cannot optimize for best significance
  - scan in signal efficiency [0 →1] and maximise background rejection
- Technical problem: how to perform maximisation
  - Minuit fit (SIMPLEX) found to be not reliable enough
  - use random sampling with uniform priors
  - not yet in release, but in preparation: *Genetics Algorithm* for maximisation (→ CMS)
- Huge speed improvement by sorting training events in N_{var}*-dim. Binary Trees*
  - for 4 variables: 41 times faster than simple volume cut
- Improvement (not yet in release): cut in de-correlated variable space

Projected Likelihood Estimator (PDE Approach)

- Combine probability density distributions to likelihood estimator

Likelihood ratio for event i_{event}

PDFs

discriminating variables

$$x_{\text{PDE},i_{\text{event}}} = \frac{\prod_{v\in\{\text{variables}\}} p_v^{\text{signal}}(x_{v,i_{\text{event}}})}{\sum_{S\in\{\text{species}\}} \left(\prod_{v\in\{\text{variables}\}} p_v^{S}(x_{v,i_{\text{event}}}) \right)}$$

Species: signal, background types

- Assumes uncorrelated input variables
  - optimal MVA approach if *true*, since containing *all* the information
  - performance reduction if *not true* → reason for development of other methods!

- Technical problem: how to implement reference PDFs
  - 3 ways: function fitting, parametric fitting (splines, kernel est.), counting

difficult to automate

easy to automate, can create artefacts

automatic, unbiased, <u>but</u> suboptimal

"De-correlated" Likelihood Estimator

- Remove linear correlations by rotating variable space in which PDEs are applied

- Determine *square-root* C' of correlation matrix C, *i.e.*, $C = C'C'$
  - compute C' by diagonalising C: $D = S^T C S \quad \Rightarrow \quad C' = S\sqrt{D}S^T$
  - transformation from original ($\mathbf{x}$) in de-correlated variable space ($\mathbf{x}'$) by: $\mathbf{x}' = C'^{-1}\mathbf{x}$

- Separate transformation for signal and background

- Note that this "de-correlation" is only complete, if:
  - input variables are Gaussians
  - correlations linear only
  - in practise: gain form de-correlation often rather modest

---

- Output of likelihood estimators often strongly peaked at 0, 1 → *T*MVA applies inverse Fermi transformation to facilitate parameterisation:

$$x_{\mathrm{PDE},i_{\mathrm{event}}} \rightarrow x'_{\mathrm{PDE},i_{\mathrm{event}}} = -\tau^{-1}\ln\left(x^{-1}_{\mathrm{PDE},i_{\mathrm{event}}} - 1\right)$$

Multidimensional Likelihood Estimator

- Generalisation of 1D PDE approach to N_{var} dimensions

- Optimal method – in theory – since full information is used

- Practical challenges:
  - parameterisation of multi-dimensional phase space needs huge training samples
  - implementation of N_{var}-dim. reference PDF with kernel estimates or counting
  - for kernel estimates: difficult to control fidelity of parameterisation

- *T*MVA implementation following Carli-Koblitz *Range-Search* method
  - count number of signal and background events in "vicinity" of data event
  - "vicinity" defined by *fixed* or *adaptive* N_{var}-dim. volume size
  - *adaptive* means rescale volume size to achieve constant number of reference events
  - speed up range search by sorting training events in Binary Trees

Carli-Koblitz, NIM A501, 576 (2003)

Fisher Discriminant (and H-Matrix)

- Well-known, simple and elegant MVA method: event selection is performed in a transformed variable space with zero linear correlations, by distinguishing the mean values of the signal and background distributions

- Instead of equations, words:

> An axis is determined in the (correlated) hyperspace of the input variables such that, when projecting the output classes (signal and background) upon this axis, they are pushed as far as possible away from each other, while events of a same class are confined in a close vicinity. The linearity property of this method is reflected in the metric with which "far apart" and "close vicinity" are determined: the covariance matrix of the discriminant variable space.

  - optimal for linearly correlated Gaussians with equal RMS' and different means
  - no separation if equal means and different RMS (shapes)

- Computation of Fisher MVA couldn't be simpler:

$$x_{\mathrm{Fisher},i_{\mathrm{event}}} \propto \sum_{v\in\{\mathrm{variables}\}} \left\{ x_{v,i_{\mathrm{event}}} \cdot F_v \right\}$$

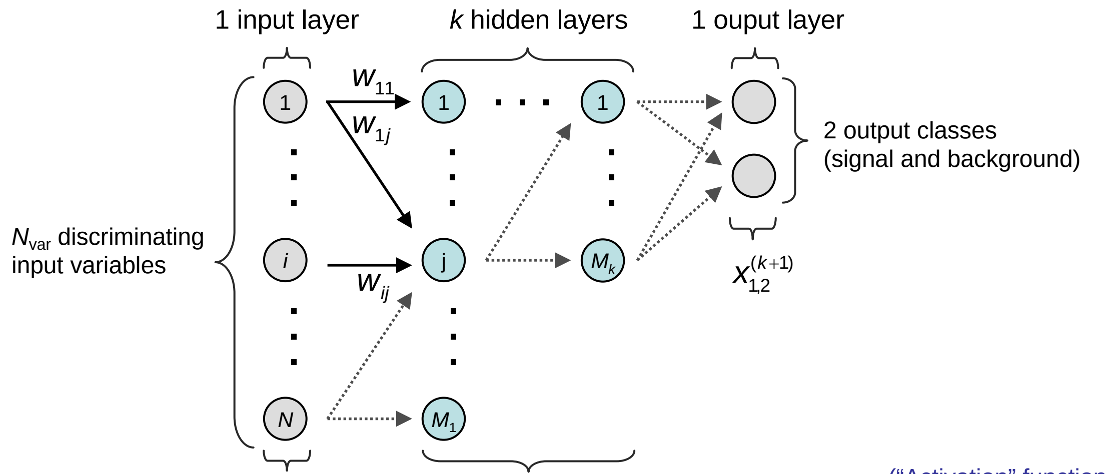
"Fisher coefficents"

- H-Matrix estimator: correlated χ^2 – poor man's variation of Fisher discriminant

Artificial Neural Network (ANN)

- ANNs are non-linear discriminants: Fisher = ANN without hidden layer
  - ANNs are now extensively used in HEP due to their performance and robustness
  - they seem to be better adapted to realistic use cases than Fisher and Likelihood
- TMVA has two different ANN implementations – both are *Multilayer Perceptrons*
  1. Clermont-Ferrand ANN: used for ALEPH Higgs analysis; translated from FORTRAN
  2. TMultiLayerPerceptron interface: ANN implemented in ROOT

Feed-forward Multilayer Perceptron



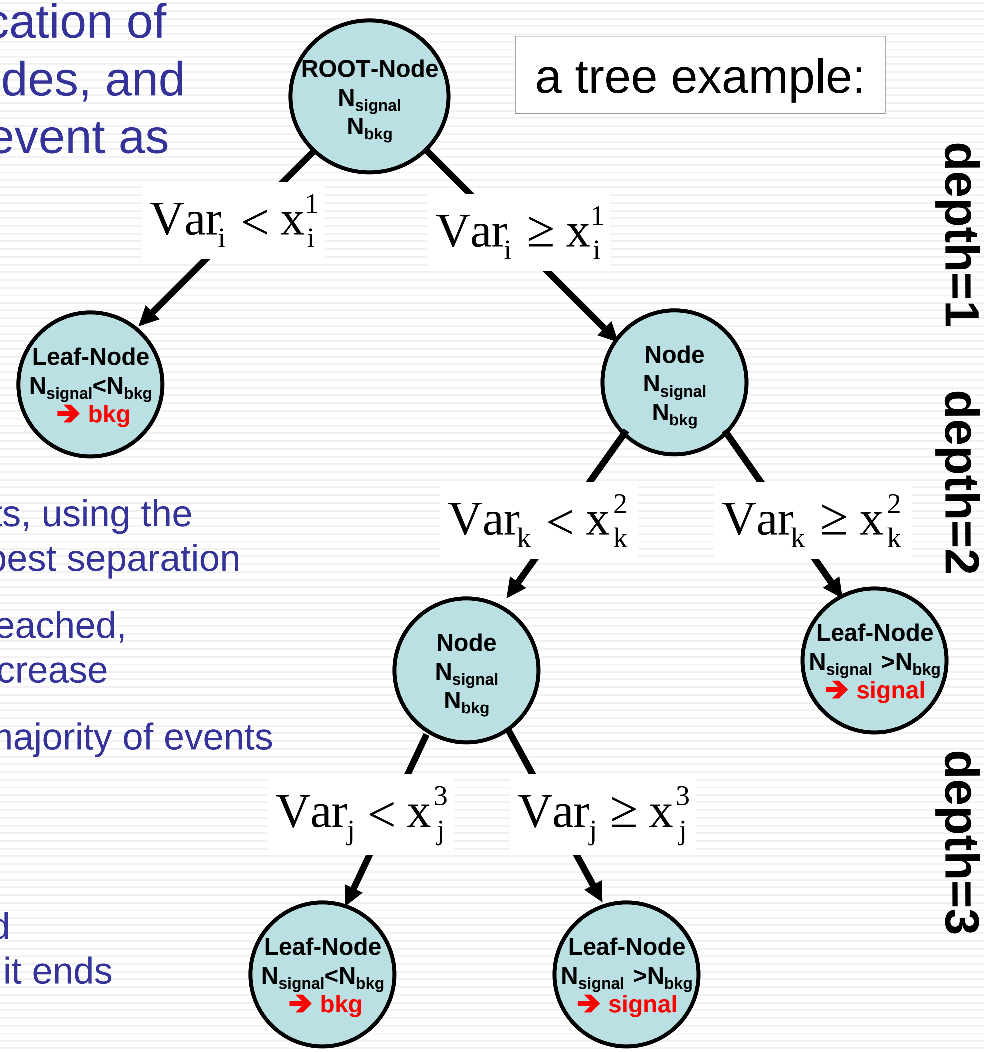
$$x_j^{(k)} = A\left(w_{0j}^{(k)} + \sum_{i=1}^{M_{k-1}} w_{ij}^{(k)} \cdot x_i^{(k-1)} \right)$$

("Activation" function)

with: $A(x) = \left(1 + e^{-x}\right)^{-1}$

Decision Trees

- Decision Trees: a sequential application of "cuts" which splits the data into nodes, and the final nodes (leaf) classifies an event as signal or background



- Training:
  - start with the root-node
  - split training sample at node into two parts, using the variable and cut, which at this stage gives best separation
  - continue splitting until: minimal #events reached, or further split would not yield separation increase
  - leaf-nodes classified (S/B) according to majority of events

- Testing:
  - a test event is "filled" at the root-node and classified according to the leaf-node where it ends up after the "cut"-sequence

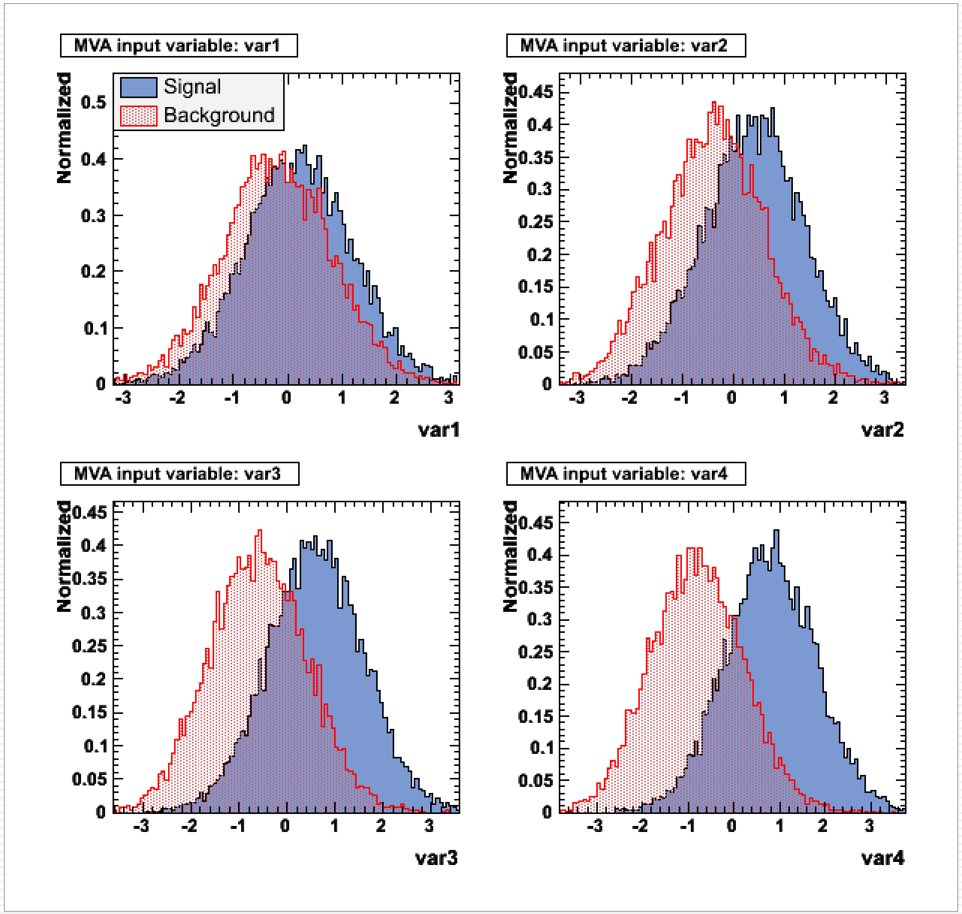
Boosted Decision Trees

- Decision Trees: used since a long time in general “data-mining” applications, less known in HEP (but very similar to “simple Cuts”)
- Advantages:
  - easy to interpret: independently of N_{var}, can always be visualised in a 2D tree
  - independent of monotone variable transformation: rather immune against outliers
  - immune against addition of weak variables
- Disadvatages:
  - instability: small changes in training sample can give large changes in tree structure

- **Boosted** Decision Trees: appeared in 1996, and overcame the disadvantages of the Decision Tree by combining several decision trees (forest) derived from one training sample via the application of event weights into ONE mulitvariate event classifier by performing “majority vote”::
  - *e.g.* AdaBoost: wrong classified training events are given a larger weight

Academic Examples (I)

- Simple toy to illustrate the strength of the de-correlation technique
  - 4 linearly correlated Gaussians, with equal RMS and shifted means between S and B



*T*MVA output :

--- TMVA_Factory: correlation matrix (signal):

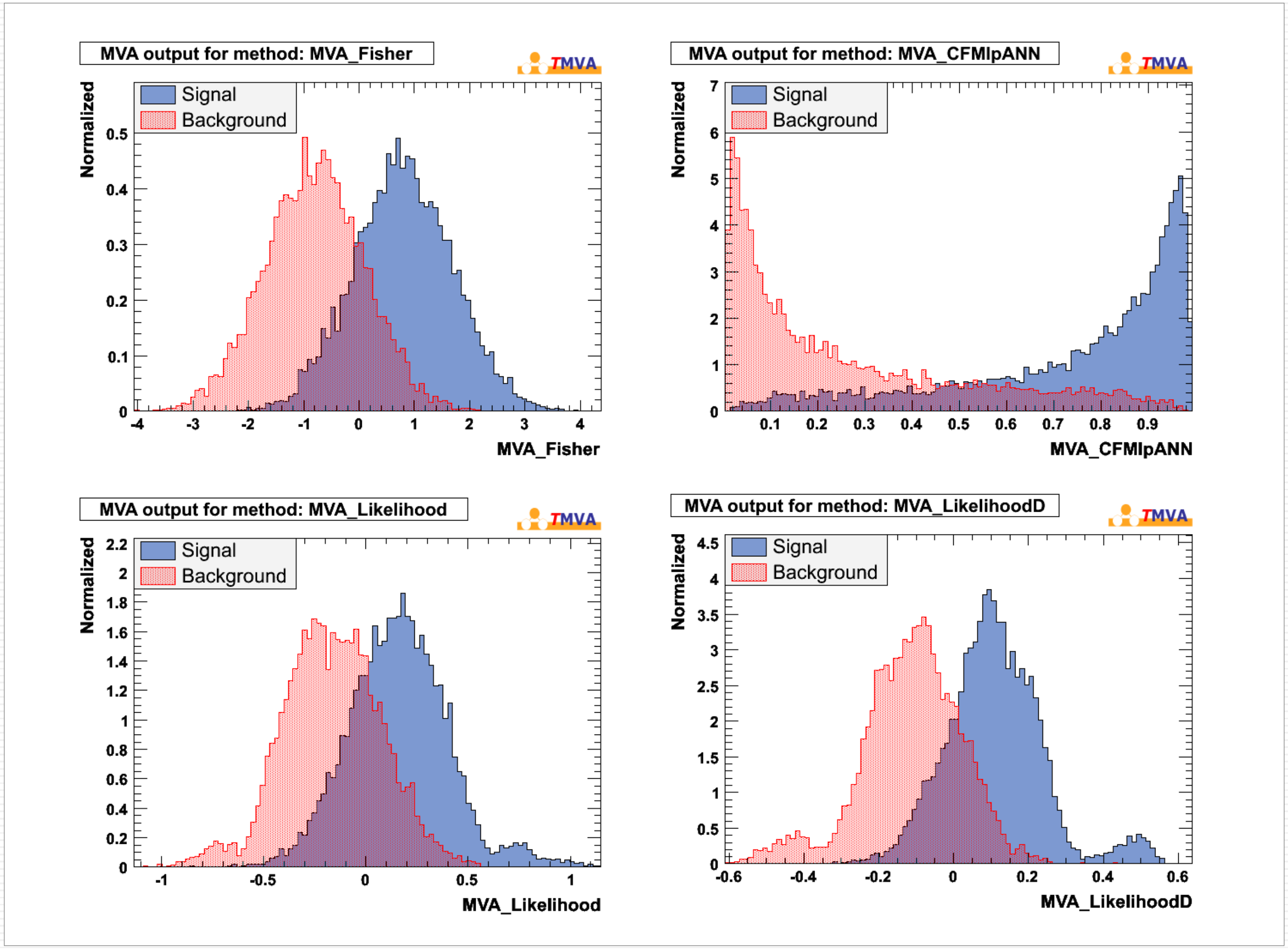
---		var1	var2	var3	var4
---	var1:	+1.000	+0.336	+0.393	+0.447
---	var2:	+0.336	+1.000	+0.613	+0.668
---	var3:	+0.393	+0.613	+1.000	+0.907
---	var4:	+0.447	+0.668	+0.907	+1.000

--- TMVA_MethodFisher: ranked output (top
--- variable is best ranked)

--- Variable	:	Coefficient:	Discr. power:
--- var4	:	+8.077	0.3888
--- var3	:	–3.417	0.2629
--- var2	:	–0.982	0.1394
--- var1	:	–0.812	0.0391

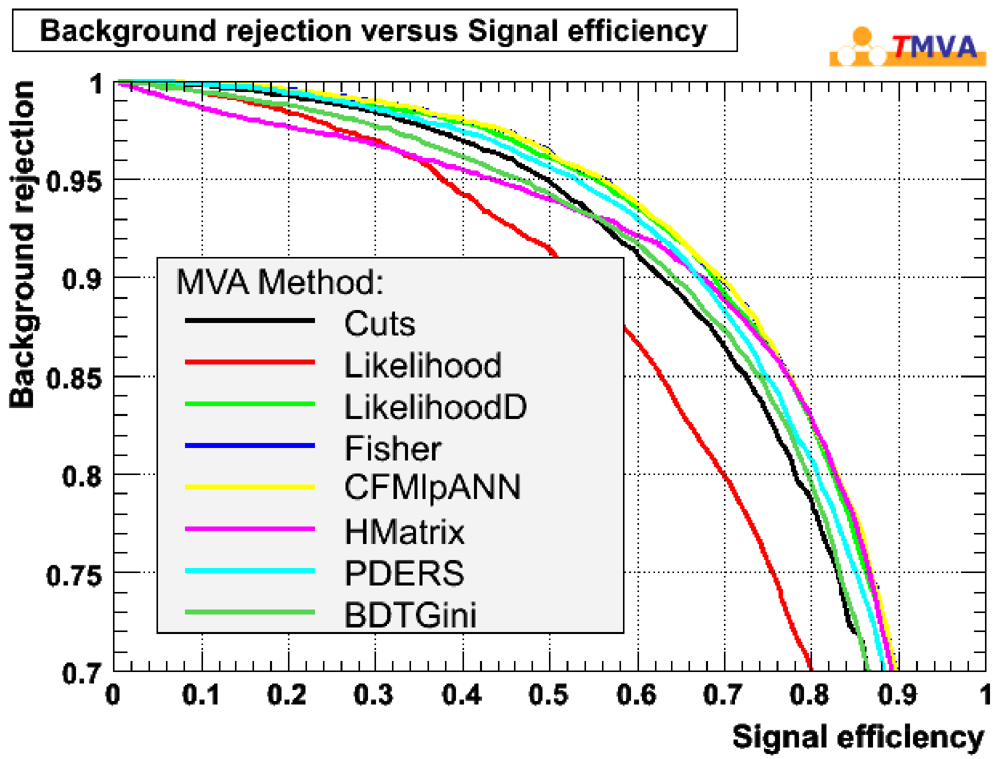
Academic Examples (I) *...continued*

- MVA output distributions for Fisher, (CF)ANN, Likelihood and de-corr. Likelihood



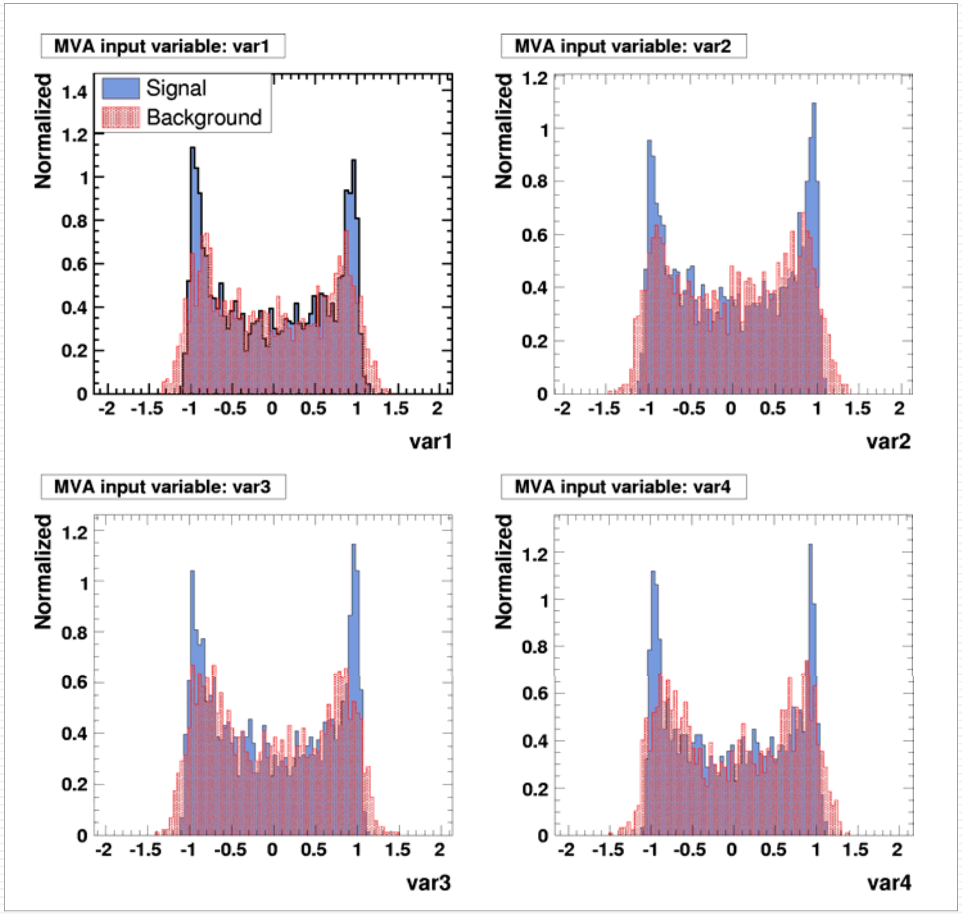
Academic Examples (I) *...continued*

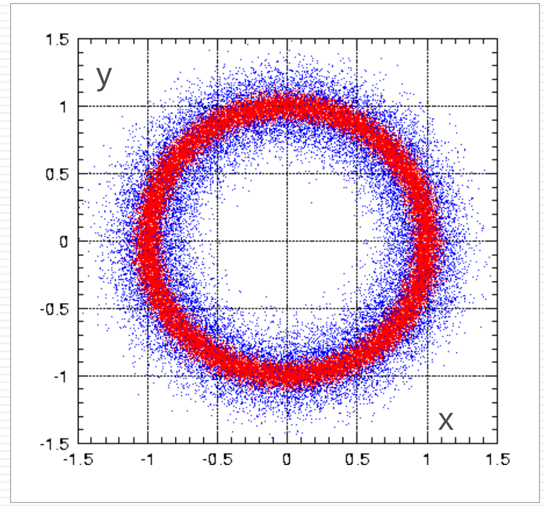
- MVA output distributions for Fisher, (CF)ANN, Likelihood and de-corr. Likelihood



Academic Examples (II)

- Simple toy to illustrate the shortcomings of the de-correlation technique
  - 2x2 variables with circular correlations for each set, equal means and different RMS’





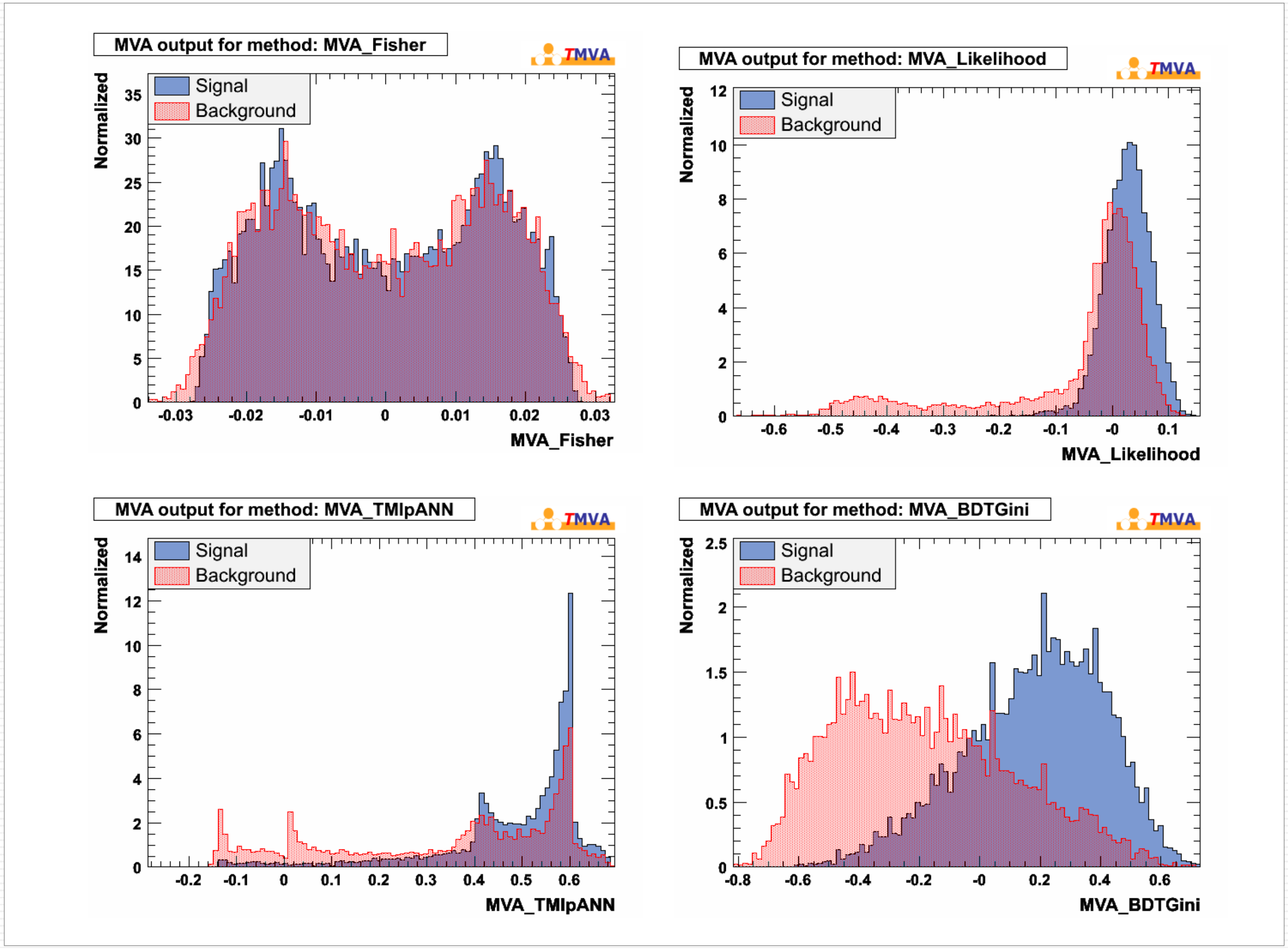
*T*MVA output :

--- TMVA_Factory: correlation matrix (signal):

---		var1	var2	var3	var4
---	var1:	+1.000	+0.001	–0.004	–0.012
---	var2:	+0.001	+1.000	–0.020	+0.001
---	var3:	–0.004	–0.020	+1.000	+0.012
---	var4:	–0.012	+0.001	+0.012	+1.000

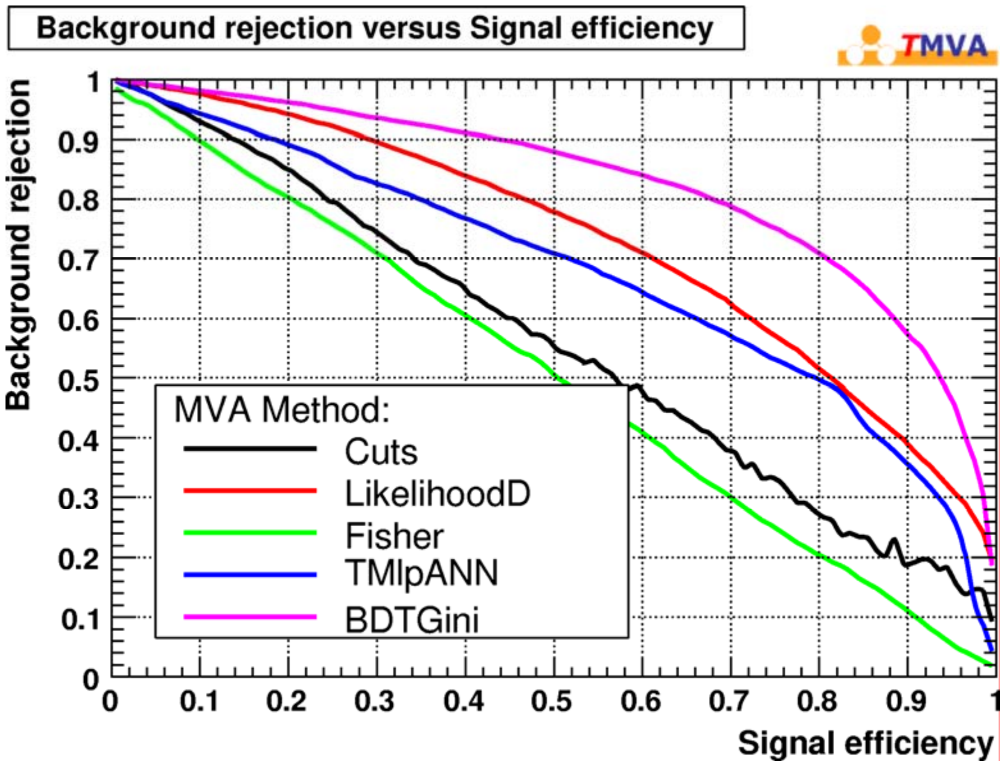
Academic Examples (II)

- MVA output distributions for Fisher, Likelihood, (ROOT)ANN, Boosted DT



Academic Examples (II)

- MVA output distributions for Fisher, Likelihood, (ROOT)ANN, Boosted DT



Concluding Remarks

- First stable *T*MVA release available at sourceforge since March 8, 2006
- ATHENA implementation ongoing – fully integrate in ROOT ?
- Compact Reader class for immediate use in ATHENA/ROOT analysis provided

- *T*MVA provides the training and evaluation tools, but the decision which method is the best is heavily dependent on the use case
- Most methods can be improved over default by optimising the training options

- Tools are developed, but now need to gain realistic experience with them !
- Starting realistic analysis with *T*MVA (Jet calibration, e-id, Trigger, LHCb, …)