Determining SUSY Parameters at the LHC

N. Bornhauser, MD

Bonn University & Bethe Center for Theoretical Physics

Contents

Contents

Contents

Contents

The LHC Inverse Problem

- Usual problem (for theorists): Have a theory $\Longrightarrow$ make predictions, i.e. compute observables.
  In principle straightforward, at least for weakly interacting theories; is being automatized.

The LHC Inverse Problem

- Usual problem (for theorists): Have a theory $\Longrightarrow$ make predictions, i.e. compute observables.
  In principle straightforward, at least for weakly interacting theories; is being automatized.
- Inverse problem: Suppose we see some signal(s) that cannot be accommodated in the SM. Can we:

The LHC Inverse Problem

- Usual problem (for theorists): Have a theory $\Longrightarrow$ make predictions, i.e. compute observables.
  In principle straightforward, at least for weakly interacting theories; is being automatized.
- Inverse problem: Suppose we see some signal(s) that cannot be accommodated in the SM. Can we:
  - Find the right theory responsible for the signal(s)?

The LHC Inverse Problem

- Usual problem (for theorists): Have a theory $\Longrightarrow$ make predictions, i.e. compute observables.
  In principle straightforward, at least for weakly interacting theories; is being automatized.
- Inverse problem: Suppose we see some signal(s) that cannot be accommodated in the SM. Can we:
  - Find the right theory responsible for the signal(s)?
  - Determine the values of the free parameters of this theory?

Application to MSSM

- Usual SUSY propaganda . . .

Application to MSSM

- Usual SUSY propaganda . . .
- Predicts many new particles $\Longrightarrow$

Application to MSSM

- Usual SUSY propaganda . . .
- Predicts many new particles $\Longrightarrow$
  - Many different final states, depending on values of parameters

Application to MSSM

- Usual SUSY propaganda ...
- Predicts many new particles $\Longrightarrow$
  - Many different final states, depending on values of parameters
  - Possibly long decay chains $\Longrightarrow$ numerous kinematical features in distributions

Application to MSSM

- Usual SUSY propaganda . . .
- Predicts many new particles $\Longrightarrow$
  - Many different final states, depending on values of parameters
  - Possibly long decay chains $\Longrightarrow$ numerous kinematical features in distributions

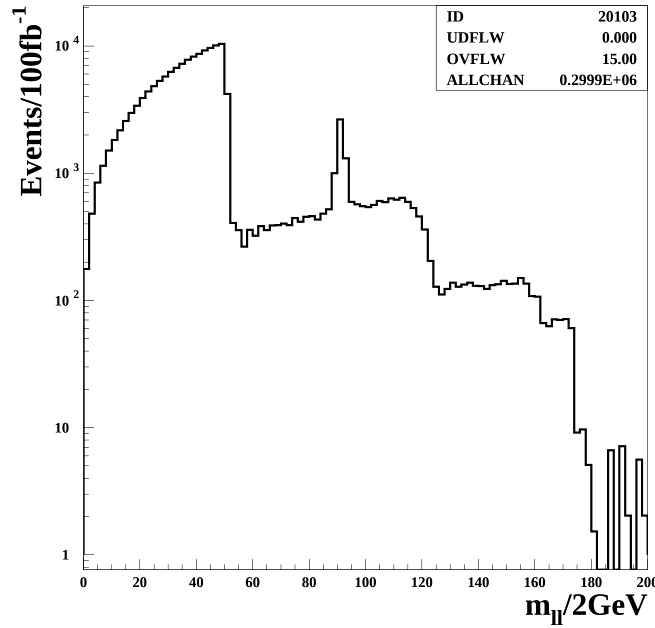
Good theory for trying out methods of parameter determination!

Traditional Approach to Parameter Determination

Determine (differences of) masses directly from kinematic features, e.g. $m_{\ell^+\ell^-}$ edges from $\tilde{\chi}_i^0 \to \tilde{\chi}_j^0 \ell^+ \ell^-$

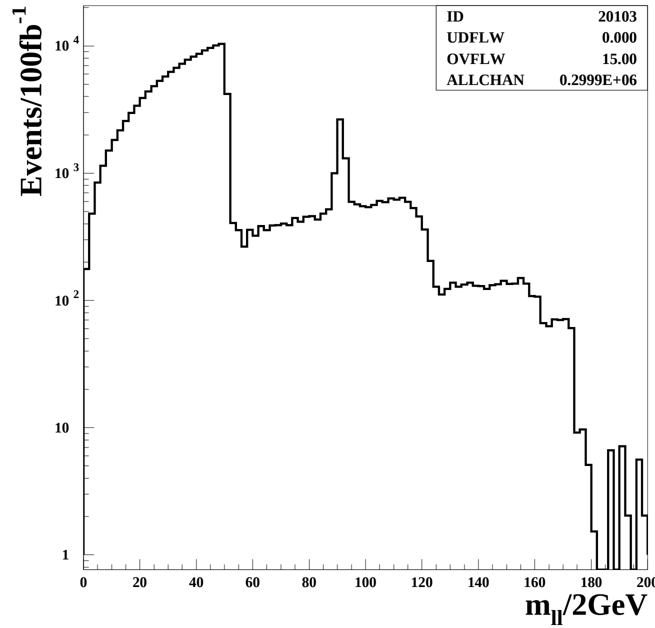
Traditional Approach to Parameter Determination

Determine (differences of) masses directly from kinematic features, e.g. $m_{\ell^+\ell^-}$ edges from $\tilde{\chi}_i^0 \to \tilde{\chi}_j^0 \ell^+ \ell^-$



Traditional Approach to Parameter Determination

Determine (differences of) masses directly from kinematic features, e.g. $m_{\ell^+\ell^-}$ edges from $\tilde{\chi}_i^0 \to \tilde{\chi}_j^0 \ell^+ \ell^-$



Once sufficiently many (differences of) masses are known: fit parameters in Lagrangian ($\overline{\mathrm{DR}}$ renormalized soft breaking parameters). Hinchliffe et al. 1996, . . .

Advantages of Traditional Approach

- First step (kinematic fitting) is quite model independent.
  (Effects of additional hard radiation?)

Advantages of Traditional Approach

- First step (kinematic fitting) is quite model independent. (Effects of additional hard radiation?)
- Second step, and in simple cases also first step, does not need event generation / simulation
  $\Longrightarrow$ Small numerical effort needed

Disadvantages of Traditional Approach

- Works very well only for final states containing ≥ 2 charged leptons $\ell^\pm = e^\pm,\ \mu^\pm$; in large parts of (C)MSSM parameter space, these are very rare.

Disadvantages of Traditional Approach

- Works very well only for final states containing ≥ 2 charged leptons $\ell^\pm = e^\pm, \mu^\pm$; in large parts of (C)MSSM parameter space, these are very rare.
- Uses only small fraction of available information: wasteful!

Brute–Force Approach

Arkani-Hamed, Kane, Thaler, Wang 2005

- Consider $1,808$ observables:

Brute–Force Approach

Arkani-Hamed, Kane, Thaler, Wang 2005

- Consider $1,808$ observables:
  - Counting rates (few)

Brute–Force Approach

Arkani-Hamed, Kane, Thaler, Wang 2005

- Consider $1,808$ observables:
  - Counting rates (few)
  - Distributions: Divide in $n \in \{10,\ 20\}$ bins such that each bin contains same no. of events; observables are bin boundaries

Brute–Force Approach

Arkani-Hamed, Kane, Thaler, Wang 2005

- Consider $1,808$ observables:
  - Counting rates (few)
  - Distributions: Divide in $n \in \{10,\ 20\}$ bins such that each bin contains same no. of events; observables are bin boundaries
- Randomly generated $43,026$ MSSM spectra in $15-$parameter MSSM; found 283 “degenerate pairs”

Brute–Force Approach

Arkani-Hamed, Kane, Thaler, Wang 2005

- Consider $1,808$ observables:
  - Counting rates (few)
  - Distributions: Divide in $n \in \{10,\ 20\}$ bins such that each bin contains same no. of events; observables are bin boundaries
- Randomly generated $43,026$ MSSM spectra in $15-$parameter MSSM; found 283 “degenerate pairs”
- Statistics employed highly questionable!

Counting Observables

- Number / fraction of events in different final states have been used to distinguish between benchmark scenarios at least since 1992 (Baer, Tata, Woodside)

Counting Observables

- Number / fraction of events in different final states have been used to distinguish between benchmark scenarios at least since 1992 (Baer, Tata, Woodside)
- Total event rate recently used as additional input in CMSSM fits (Dreiner et al. 2010, 2011)

Counting Observables

- Number / fraction of events in different final states have been used to distinguish between benchmark scenarios at least since 1992 (Baer, Tata, Woodside)
- Total event rate recently used as additional input in CMSSM fits (Dreiner et al. 2010, 2011)

Our goals:

- Check usefulness of counting observables for discriminating between scenarios, determining values of parameters

Counting Observables

- Number / fraction of events in different final states have been used to distinguish between benchmark scenarios at least since 1992 (Baer, Tata, Woodside)
- Total event rate recently used as additional input in CMSSM fits (Dreiner et al. 2010, 2011)

Our goals:

- Check usefulness of counting observables for discriminating between scenarios, determining values of parameters
- Keep control of statistical properties: include full covariance matrix

Choice of Observables

- Not too few! Want to have more observables than parameters, with reasonably small statistical errors, “everywhere” in parameter space!

Choice of Observables

- Not too few! Want to have more observables than parameters, with reasonably small statistical errors, “everywhere” in parameter space!
- Not too many! Complicates analysis; can dilute statistical power.

Choice of Observables

- Not too few! Want to have more observables than parameters, with reasonably small statistical errors, “everywhere” in parameter space!
- Not too many! Complicates analysis; can dilute statistical power.
- $\Longrightarrow$ take 84 observables

Construction of Observables

- Divide all events into 12 mutually exclusive classes: $(\ell = e, \mu)$
  1: 0ℓ; 2: $1\ell^-$; 3: $1\ell^+$; 4: $2\ell^-$; 5: $2\ell^+$; 6: $\ell_i^+\ell_i^-$; 7: $\ell_i^+\ell_{j\neq i}^-$;
  8: $\ell_i^-\ell_j^-\ell_j^+$; 9: $\ell_i^+\ell_j^-\ell_j^+$; 10: $\ell_i^-\ell_j^-\ell^\pm_{k\neq i,j \text{ for } +}$; 11: $\ell_i^+\ell_j^+\ell^\pm_{k\neq i,j \text{ for } -}$;
  12: $\geq 4\ell$.

Construction of Observables

- Divide all events into 12 mutually exclusive classes: $(\ell = e, \mu)$
  1: 0ℓ; 2: $1\ell^-$; 3: $1\ell^+$; 4: $2\ell^-$; 5: $2\ell^+$; 6: $\ell_i^+\ell_i^-$; 7: $\ell_i^+\ell_{j\neq i}^-$;
  8: $\ell_i^-\ell_j^-\ell_j^+$; 9: $\ell_i^+\ell_j^-\ell_j^+$; 10: $\ell_i^-\ell_j^-\ell_{k\neq i,j \text{ for } +}^{\pm}$; 11: $\ell_i^+\ell_j^+\ell_{k\neq i,j \text{ for } -}^{\pm}$;
  12: $\geq 4\ell$.
- Define 7 observables within each class of events:
  $O_{1,c} = n_c/N;\ O_{2,c} = \langle n_{\tau^-}\rangle_c;\ O_{3,c} = \langle n_{\tau^+}\rangle_c;$
  $O_{4,c} = \langle n_b\rangle_c;\ O_{5,c} = \langle n_j\rangle_c;\ O_{6,c} = \langle n_j^2\rangle_c;\ O_{7,c} = \langle H_T\rangle_c$
  $\sum_{i=1}^{12} O_{i,c} = 1 \Longrightarrow$ replace $O_{1,1}$ by N.

Construction of Observables

- Divide all events into 12 mutually exclusive classes: ($\ell = e, \mu$)
  1: 0ℓ; 2: $1\ell^-$; 3: $1\ell^+$; 4: $2\ell^-$; 5: $2\ell^+$; 6: $\ell_i^+\ell_i^-$; 7: $\ell_i^+\ell_{j\neq i}^-$;
  8: $\ell_i^-\ell_j^-\ell_j^+$; 9: $\ell_i^+\ell_j^-\ell_j^+$; 10: $\ell_i^-\ell_j^-\ell^\pm_{k\neq i,j \text{ for } +}$; 11: $\ell_i^+\ell_j^+\ell^\pm_{k\neq i,j \text{ for } -}$;
  12: $\geq 4\ell$.
- Define 7 observables within each class of events:
  $O_{1,c} = n_c/N;\ O_{2,c} = \langle n_{\tau^-}\rangle_c;\ O_{3,c} = \langle n_{\tau^+}\rangle_c;$
  $O_{4,c} = \langle n_b\rangle_c;\ O_{5,c} = \langle n_j\rangle_c;\ O_{6,c} = \langle n_j^2\rangle_c;\ O_{7,c} = \langle H_T\rangle_c$
  $\sum_{i=1}^{12} O_{i,c} = 1 \Longrightarrow$ replace $O_{1,1}$ by N.
- Calculation of covariance matrix straightforward

Construction of Observables

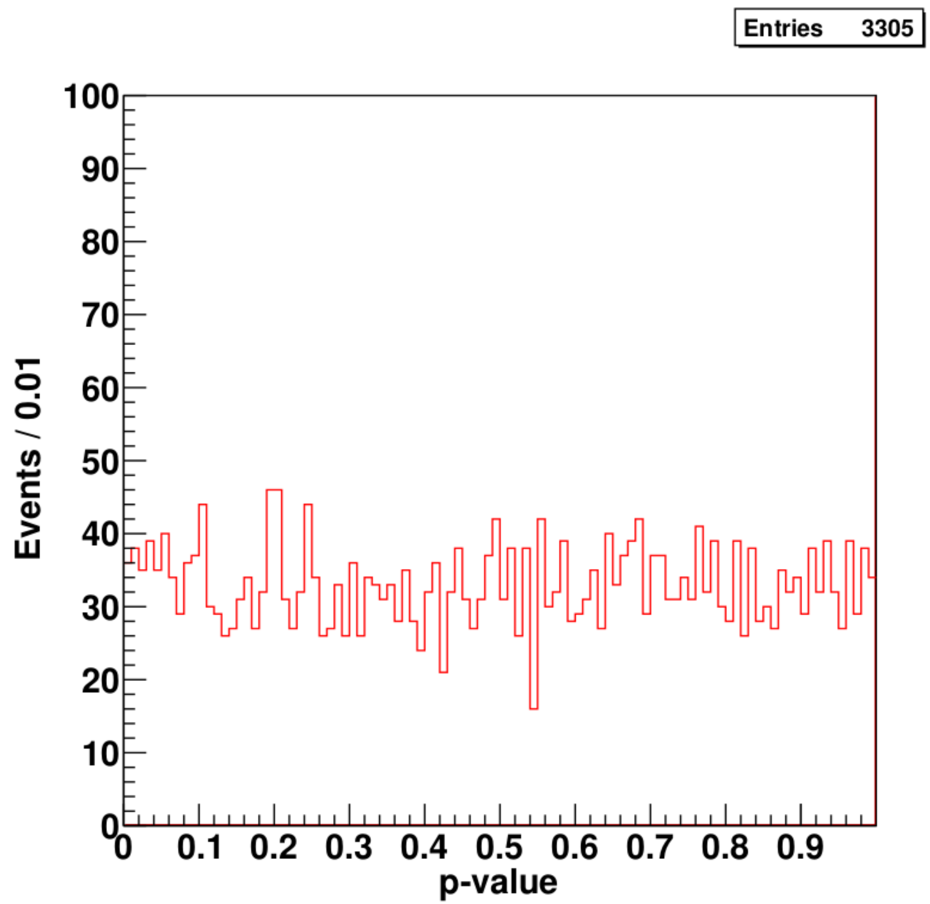
- Divide all events into 12 mutually exclusive classes: ($\ell = e, \mu$)
  1: 0ℓ; 2: $1\ell^-$; 3: $1\ell^+$; 4: $2\ell^-$; 5: $2\ell^+$; 6: $\ell_i^+\ell_i^-$; 7: $\ell_i^+\ell_{j\neq i}^-$;
  8: $\ell_i^-\ell_j^-\ell_j^+$; 9: $\ell_i^+\ell_j^-\ell_j^+$; 10: $\ell_i^-\ell_j^-\ell^{\pm}_{k\neq i,j \text{ for } +}$; 11: $\ell_i^+\ell_j^+\ell^{\pm}_{k\neq i,j \text{ for } -}$;
  12: $\geq 4\ell$.
- Define 7 observables within each class of events:
  $O_{1,c} = n_c/N;\ O_{2,c} = \langle n_{\tau^-}\rangle_c;\ O_{3,c} = \langle n_{\tau^+}\rangle_c;$
  $O_{4,c} = \langle n_b\rangle_c;\ O_{5,c} = \langle n_j\rangle_c;\ O_{6,c} = \langle n_j^2\rangle_c;\ O_{7,c} = \langle H_T\rangle_c$
  $\sum_{i=1}^{12} O_{i,c} = 1 \Longrightarrow$ replace $O_{1,1}$ by N.
- Calculation of covariance matrix straightforward
- Cuts depend on event class

Discriminating between Scenarios

arXiv:1206.6080 (hep-ph)

Based on $\chi^2_{AB} = \sum_{m,n=1}^{N_{O,\max}} \left(O_m^A - O_m^B\right) \left(V^{-1}\right)_{mn} \left(O_n^A - O_n^B\right)$
Include only "significant" observables ($\Longrightarrow N_{O,\max} < 84$ usually), based on ≥ 10 events

Distribution of total $p-$value



Based on self-comparison of $3,305$ scenarios of Arkani-Hamed et al.

Results

- No sys. errors, no backgrounds: *all* 283 "degenerate pairs" have $p < 0.05$, i.e. are distinguishable; $\bar{p} = 7 \cdot 10^{-5}$.

Results

- No sys. errors, no backgrounds: *all* 283 "degenerate pairs" have $p < 0.05$, i.e. are distinguishable; $\bar{p} = 7 \cdot 10^{-5}$.
- Including syst. errors (15% on N, 1% on all other observables, no correlations): 260 "degenerate pairs" have $p < 0.05$

Results

- No sys. errors, no backgrounds: *all* 283 “degenerate pairs” have $p < 0.05$, i.e. are distinguishable; $\bar{p} = 7 \cdot 10^{-5}$.
- Including syst. errors (15% on N, 1% on all other observables, no correlations): 260 “degenerate pairs” have $p < 0.05$
- With SM background, no syst. error: 282 “degenerate pairs” have $p < 0.05$.

Results

- No sys. errors, no backgrounds: *all* 283 "degenerate pairs" have $p < 0.05$, i.e. are distinguishable; $\bar{p} = 7 \cdot 10^{-5}$.
- Including syst. errors (15% on N, 1% on all other observables, no correlations): 260 "degenerate pairs" have $p < 0.05$
- With SM background, no syst. error: 282 "degenerate pairs" have $p < 0.05$.
- With SM background and syst. errors: 237 "degenerate pairs" have $p < 0.05$

Conclusions of Part I

- MSSM inverse problem not as difficult as claimed

Conclusions of Part I

- MSSM inverse problem not as difficult as claimed
- Counting observables are useful!

Conclusions of Part I

- MSSM inverse problem not as difficult as claimed
- Counting observables are useful!
- Scenarios considered here have (almost?) all been excluded already ...

Determining Parameters

Here: use CMSSM (4 parameters), for simplicity

Determining Parameters

Here: use CMSSM (4 parameters), for simplicity

First attempt: by minimizing χ^2: Did not work!

Determining Parameters

Here: use CMSSM (4 parameters), for simplicity

First attempt: by minimizing χ^2: Did not work!

Basic problem: routine minimizing χ^2 wants to compute $\delta\chi^2$ when some parameter is changed by small amount $\Longrightarrow$ need huge statistics for $\delta\chi^2$ *not* to be distorted by statistical fluctuations!

Determining Parameters

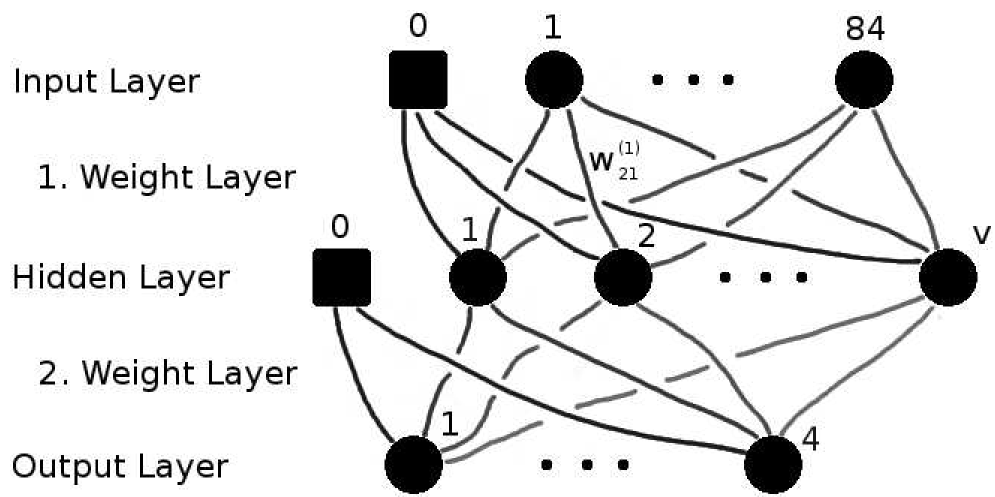
Here: use CMSSM (4 parameters), for simplicity

First attempt: by minimizing χ^2: Did not work!

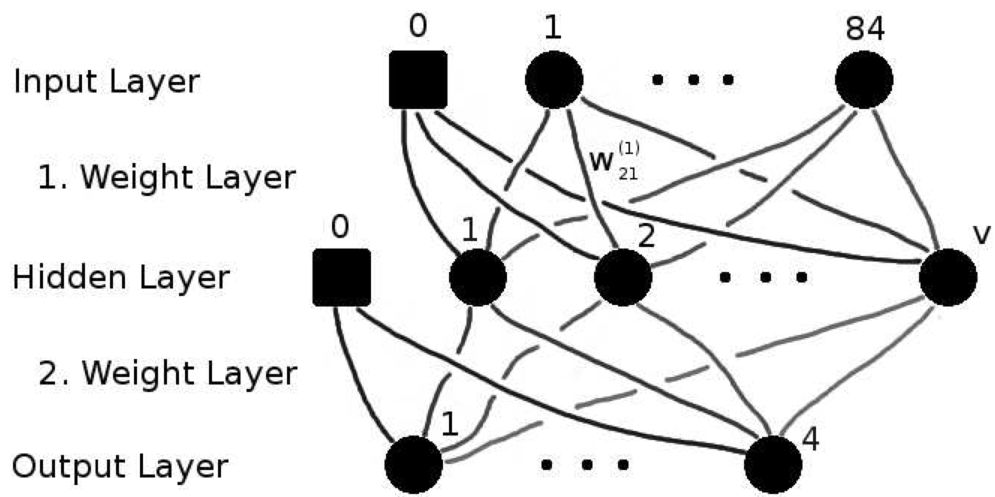
Basic problem: routine minimizing χ^2 wants to compute $\delta\chi^2$ when some parameter is changed by small amount $\Longrightarrow$ need huge statistics for $\delta\chi^2$ *not* to be distorted by statistical fluctuations!

Best effort gave estimated errors two times larger than better method.

Artificial Neural Networks

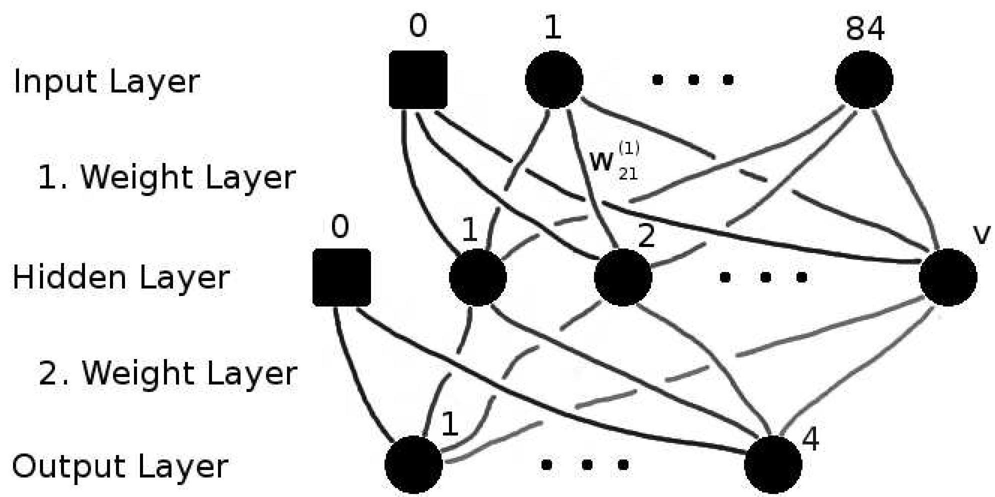


Artificial Neural Networks



$84(+1)$ input neurons: pass on normalized observables $x_i \in [-1, 1]$. $0-$th input neuron always gives 1.

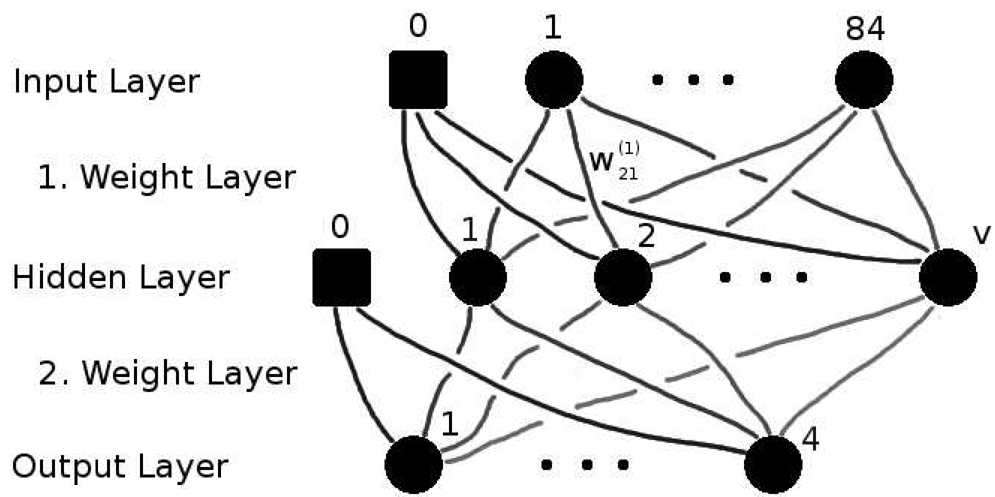
Artificial Neural Networks



$84(+1)$ input neurons: pass on normalized observables $x_i \in [-1, 1]$. $0-$th input neuron always gives 1.

$v \simeq 20(+1)$ "hidden neurons" in single layer

Artificial Neural Networks



$84(+1)$ input neurons: pass on normalized observables $x_i \in [-1, 1]$. $0-$th input neuron always gives 1.

$v \simeq 20(+1)$ "hidden neurons" in single layer

1 output neuron: outputs normalized CMSSM parameter $y \in [-1, +1]$.

ANN set–up (cont'd)

- Layers of neurons are connected; each connection is assigned a weight (ideally, are all $\leq \mathcal{O}(1)$ in magnitude)

ANN set–up (cont'd)

- Layers of neurons are connected; each connection is assigned a weight (ideally, are all $\leq \mathcal{O}(1)$ in magnitude)
- Input and output neurons just sum all their inputs as output

ANN set–up (cont'd)

- Layers of neurons are connected; each connection is assigned a weight (ideally, are all $\leq \mathcal{O}(1)$ in magnitude)
- Input and output neurons just sum all their inputs as output
- Hidden neurons: apply $\tanh$ function to sum of inputs

ANN set–up (cont'd)

- Layers of neurons are connected; each connection is assigned a weight (ideally, are all $\leq \mathcal{O}(1)$ in magnitude)
- Input and output neurons just sum all their inputs as output
- Hidden neurons: apply $\tanh$ function to sum of inputs

Altogether: ANN output

$$y = \sum_{a=0}^{v} w_a^{(2)} \tanh\left(\sum_{i=0}^{84} w_{ai}^{(1)} x_i\right)$$

ANN set–up (cont'd)

- Layers of neurons are connected; each connection is assigned a weight (ideally, are all $\leq \mathcal{O}(1)$ in magnitude)
- Input and output neurons just sum all their inputs as output
- Hidden neurons: apply $\tanh$ function to sum of inputs

Altogether: ANN output

$$y = \sum_{a=0}^{v} w_a^{(2)} \tanh\left(\sum_{i=0}^{84} w_{ai}^{(1)} x_i \right)$$

Form of $\tanh \Longrightarrow$ ANN is good at interpolating, not necessarily at extrapolating

Training the ANN

Determine "optimal" values of weights $w_{ai}^{(1)}$, $w_{a}^{(2)}$:
$86 \cdot v + 1 \gtrsim 2,000$ variables in total!

Training the ANN

Determine "optimal" values of weights $w_{ai}^{(1)}$, $w_a^{(2)}$:
$86 \cdot v + 1 \gtrsim 2,000$ variables in total!

To that end: need training sets and control sets

Training the ANN

Determine "optimal" values of weights $w_{ai}^{(1)}$, $w_a^{(2)}$:
$86 \cdot v + 1 \gtrsim 2,000$ variables in total!

To that end: need training sets and control sets

Considered 4 benchmark points just beyond current limits; used reduced parameter range for m_0, $m_{1/2}$ when constructing training and control sets:
$m_{1/2} \in [m_{1/2}^{\rm true} - 40 \text{ GeV}, m_{1/2}^{\rm true} + 40 \text{ GeV}]$
$m_0 \in [m_0^{\rm true} - 150 \text{ GeV}, m_0^{\rm true} + 150 \text{ GeV}]$ (roughly).
$A_0 \in [-2m_0, 2m_0]; \quad \tan\beta \in [5, 50]$.

Training the ANN

Determine "optimal" values of weights $w_{ai}^{(1)}$, $w_a^{(2)}$:
$86 \cdot v + 1 \gtrsim 2,000$ variables in total!

To that end: need training sets and control sets

Considered 4 benchmark points just beyond current limits; used reduced parameter range for m_0, $m_{1/2}$ when constructing training and control sets:
$m_{1/2} \in [m_{1/2}^{\rm true} - 40 \text{ GeV}, m_{1/2}^{\rm true} + 40 \text{ GeV}]$
$m_0 \in [m_0^{\rm true} - 150 \text{ GeV}, m_0^{\rm true} + 150 \text{ GeV}]$ (roughly).
$A_0 \in [-2m_0, 2m_0]; \quad \tan\beta \in [5, 50]$.

Generated $\sim 1,000$ (primary) training sets, 300 control sets for each benchmark point

Training the ANN (cont'd)

Problem: ANNs are only fed observables, *not* their errors!

Training the ANN (cont'd)

Problem: ANNs are only fed observables, *not* their errors!

Include entries of covariance matrix as inputs: would need too many input neurons $\Longrightarrow$ too many weights: numerical effort for training $\tau \propto (\mathrm{no.\ of\ weights})^2$!

Training the ANN (cont'd)

Problem: ANNs are only fed observables, *not* their errors!

Include entries of covariance matrix as inputs: would need too many input neurons $\Longrightarrow$ too many weights: numerical effort for training $\tau \propto (\text{no. of weights})^2$!

Instead: Generate 100 "satellite sets" for each primary training set! Use central values, covariance matrix of original training set to generate 100 Gaussian distributed variants of this "measurement". Does *not* need additional event generation, but $\tau \propto \text{no. of training sets}$!

Training the ANN (cont'd)

Problem: ANNs are only fed observables, *not* their errors!

Include entries of covariance matrix as inputs: would need too many input neurons $\Longrightarrow$ too many weights: numerical effort for training $\tau \propto (\text{no. of weights})^2$!

Instead: Generate 100 "satellite sets" for each primary training set! Use central values, covariance matrix of original training set to generate 100 Gaussian distributed variants of this "measurement". Does *not* need additional event generation, but $\tau \propto \text{no. of training sets}$!

This implies:
Different inputs (observables) can give same output (CMSSM parameter);
Same inputs can give different outputs, in training sets!

Effect of “satellite sets”

- Totality of training sets can*not* be described by a function, but ANN is still a function (“fuzzy logic”)

Effect of “satellite sets”

- Totality of training sets can*not* be described by a function, but ANN is still a function (“fuzzy logic”)
- Improves ANN performance by 15 to 30%

Effect of “satellite sets”

- Totality of training sets can*not* be described by a function, but ANN is still a function (“fuzzy logic”)
- Improves ANN performance by 15 to 30%
- Improves statistical properties of output: can derive reliable Gaussian error estimates!

ANN training: iterative

- In each step, change weights such that totality of *training sets* is described better, i.e.

$$\sum_{i=1}^{N_{\rm train}} \left[y(O_i) - y_i \right]^2$$

is reduced (“conjugate gradient algorithm”)

ANN training: iterative

- In each step, change weights such that totality of *training sets* is described better, i.e.

$$\sum_{i=1}^{N_{\text{train}}} \left[y(O_i) - y_i\right]^2$$

is reduced ("conjugate gradient algorithm")

- Stop iteration when normalized control error has reached minimum:

$$\bar{F}^2 = \frac{\sum_{i=1}^{N_{\text{control}}} \left[y(O_i) - y_i\right]^2}{\sum_{i=1}^{N_{\text{control}}} \left[y_i - \bar{y}\right]^2}$$

Recall: control sets not used for modifying weights!

Estimating the error on the final ANN output

Used two methods:

- Gaussian error propagation

Estimating the error on the final ANN output

Used two methods:

- Gaussian error propagation
- By generating 10^5 to 10^6 Gaussian distributed “satellite measurements” from central value, covariance matrix of original “measurement”
  $\Longrightarrow$ fit (2–dim.) Gaussian to distribution of ANN output

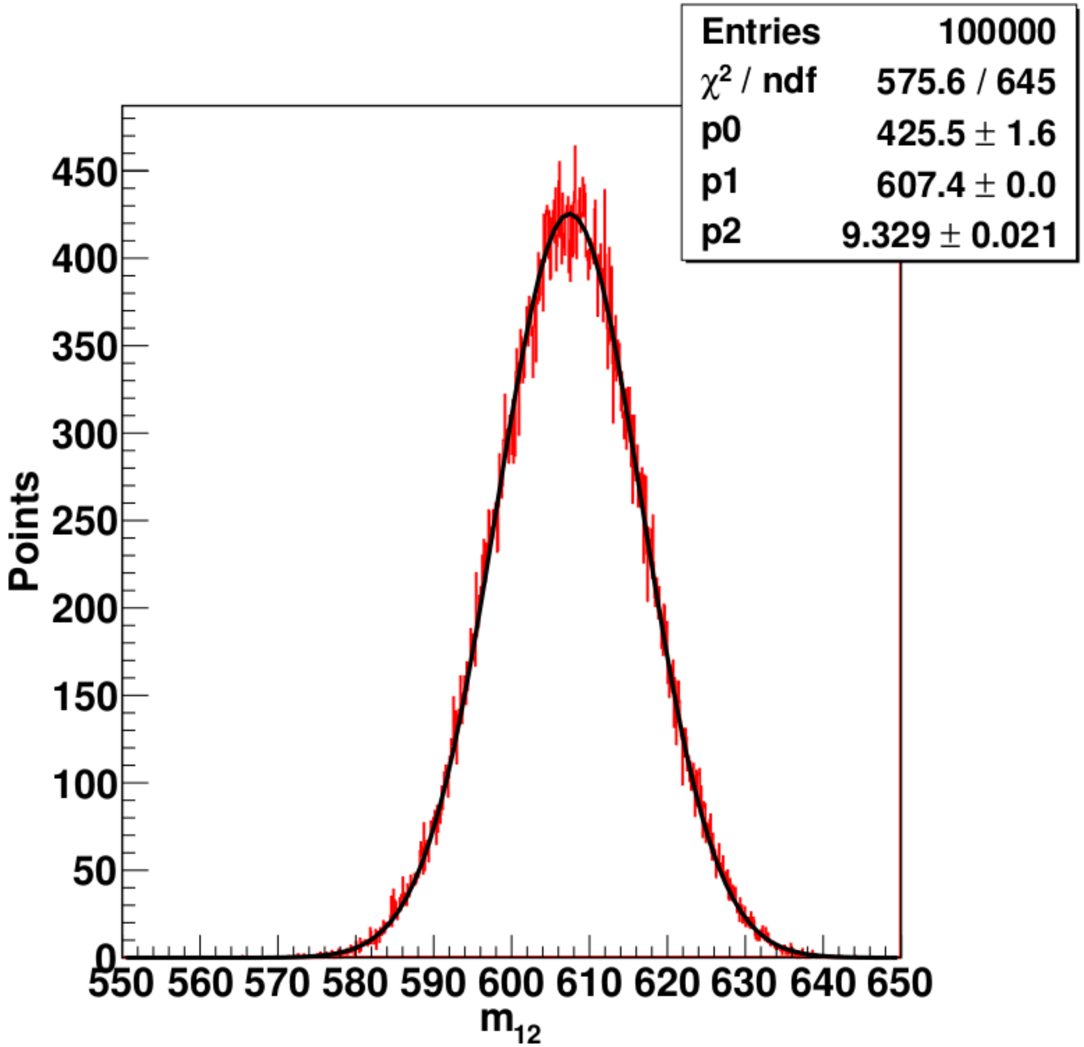
Estimating the error on the final ANN output

Used two methods:

- Gaussian error propagation
- By generating 10^5 to 10^6 Gaussian distributed “satellite measurements” from central value, covariance matrix of original “measurement”
  $\Longrightarrow$ fit (2–dim.) Gaussian to distribution of ANN output

Both methods give consistent results!

Example: BP3 with $10\ \text{fb}^{-1}$



From error propagation: $m_{1/2} = 607 \pm 11$ GeV.

Results

	Point 1	Point 2	Point 3	Point 4
	10/fb			
m_0	171.47 ± 34.53	1998.93 ± 92.20	1055.97 ± 47.26	482.94 ± 61.23
$m_{1/2}$	697.41 ± 7.76	446.55 ± 11.30	607.45 ± 11.53	695.48 ± 7.88
$\tan\beta$	21.35 ± 5.96	12.34 ± 20.17	23.41 ± 37.42	26.00 ± 10.66
A_0	463.43 ± 326.00	1406.37 ± 2898.67	1453.49 ± 1891.58	-73.52 ± 628.16
	500/fb			
m_0	156.40 ± 4.88	2004.05 ± 10.61	1015.66 ± 4.49	391.86 ± 7.70
$m_{1/2}$	701.17 ± 0.87	451.15 ± 1.00	598.75 ± 1.21	700.71 ± 0.88
$\tan\beta$	9.39 ± 0.70	14.30 ± 2.12	20.79 ± 5.20	30.54 ± 1.23
A_0	-43.61 ± 55.37	261.47 ± 474.68	774.39 ± 295.63	183.05 ± 101.48

All correlations are quite weak

Summary

- Counting observables proved useful for

Summary

- Counting observables proved useful for
  - Discriminating between MSSM parameter sets using true χ^2: LHC inverse problem not as difficult as claimed

Summary

- Counting observables proved useful for
  - Discriminating between MSSM parameter sets using true χ^2: LHC inverse problem not as difficult as claimed
  - Determining CMSSM parameters, and reliable errors, using artificial neural networks

Summary

- Counting observables proved useful for
  - Discriminating between MSSM parameter sets using true χ^2: LHC inverse problem not as difficult as claimed
  - Determining CMSSM parameters, and reliable errors, using artificial neural networks
- To do for ANN analysis:

In principle straightforward!

Summary

- Counting observables proved useful for
  - Discriminating between MSSM parameter sets using true χ^2: LHC inverse problem not as difficult as claimed
  - Determining CMSSM parameters, and reliable errors, using artificial neural networks
- To do for ANN analysis:
  - Include backgrounds, syst. errors

In principle straightforward!

Summary

- Counting observables proved useful for
  - Discriminating between MSSM parameter sets using true χ^2: LHC inverse problem not as difficult as claimed
  - Determining CMSSM parameters, and reliable errors, using artificial neural networks
- To do for ANN analysis:
  - Include backgrounds, syst. errors
  - Include other observables, including kinematical ones

In principle straightforward!

Summary

- Counting observables proved useful for
  - Discriminating between MSSM parameter sets using true χ^2: LHC inverse problem not as difficult as claimed
  - Determining CMSSM parameters, and reliable errors, using artificial neural networks
- To do for ANN analysis:
  - Include backgrounds, syst. errors
  - Include other observables, including kinematical ones

  In principle straightforward!
- Method should be applicable to *any* BSM framework!

Summary

- Counting observables proved useful for
  - Discriminating between MSSM parameter sets using true χ^2: LHC inverse problem not as difficult as claimed
  - Determining CMSSM parameters, and reliable errors, using artificial neural networks
- To do for ANN analysis:
  - Include backgrounds, syst. errors
  - Include other observables, including kinematical ones

  In principle straightforward!
- Method should be applicable to *any* BSM framework!