SCYNet: a tool for fast global fits to collider searches

Abtin Narimani

University of Bonn

Terascale Meeting, Nov 2017

28.11.2017

Motivation

- Exploring more generic SUSY models (pMSSM-11)

  Global scan for SUSY

- Why pMSSM?

  1. $(g-2)_\mu$
  2. DM relic density
  3. LHC limits from direct searches

- Accurate and fast statistical evaluation of LHC search limits
- Calculate χ^2 from the event counts in the signal regions with CheckMATE
- χ^2 to train Neural Network

Motivation

- Exploring more generic SUSY models (pMSSM-11)

  Global scan for SUSY

- Why pMSSM?
  1. $(g-2)_\mu$
  2. DM relic density
  3. LHC limits from direct searches
- Accurate and fast statistical evaluation of LHC search limits
- Calculate χ^2 from the event counts in the signal regions with CheckMATE
- χ^2 to train Neural Network

Motivation

- Exploring more generic SUSY models (pMSSM-11)

  Global scan for SUSY

- Why pMSSM?

  1. $(g-2)_\mu$
  2. DM relic density
  3. LHC limits from direct searches

- Accurate and fast statistical evaluation of LHC search limits
- Calculate χ^2 from the event counts in the signal regions with CheckMATE
- χ^2 to train Neural Network

Motivation

- Exploring more generic SUSY models (pMSSM-11)

Global scan for SUSY

- Why pMSSM?

1. $(g-2)_\mu$
2. DM relic density
3. LHC limits from direct searches

- Accurate and fast statistical evaluation of LHC search limits
- Calculate χ^2 from the event counts in the signal regions with CheckMATE
- χ^2 to train Neural Network

Motivation

- Exploring more generic SUSY models (pMSSM-11)

  Global scan for SUSY

- Why pMSSM?

  1. $(g-2)_\mu$
  2. DM relic density
  3. LHC limits from direct searches

- Accurate and fast statistical evaluation of LHC search limits
- Calculate χ^2 from the event counts in the signal regions with CheckMATE
- χ^2 to train Neural Network

Motivation

- Exploring more generic SUSY models (pMSSM-11)

  Global scan for SUSY

- Why pMSSM?

  1. $(g-2)_\mu$
  2. DM relic density
  3. LHC limits from direct searches

- Accurate and fast statistical evaluation of LHC search limits
- Calculate χ^2 from the event counts in the signal regions with CheckMATE
- χ^2 to train Neural Network

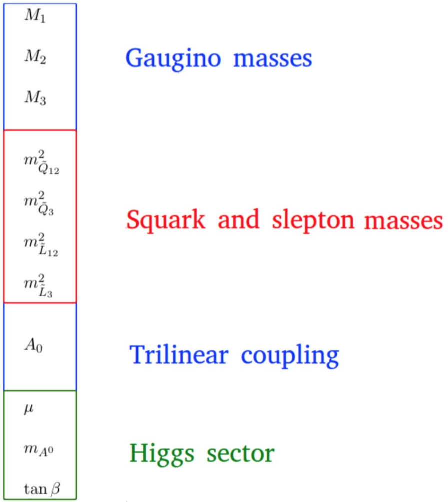
1 SUSY
2 Neural Network
3 Event Generation
4 Results

phenomenological MSSM (pMSSM)

- 105 parameters
- No R-parity violation

  no CP violation

  no FCNC
- 105 → 11 parameters

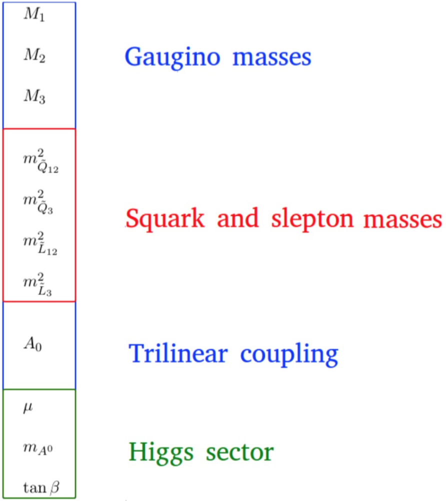


phenomenological MSSM (pMSSM)

- 105 parameters
- No R-parity violation

  no CP violation

  no FCNC
- $105 \rightarrow 11$ parameters



phenomenological MSSM (pMSSM)

- 105 parameters
- No R-parity violation

  no CP violation

  no FCNC
- $105 \rightarrow 11$ parameters

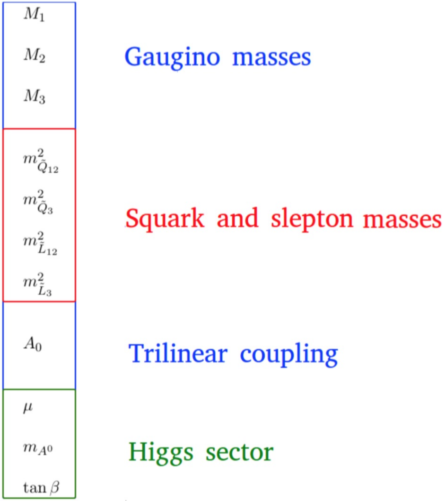
M_1, M_2, M_3	Gaugino masses
$m^2_{\tilde{Q}_{12}}$, $m^2_{\tilde{Q}_3}$, $m^2_{\tilde{L}_{12}}$, $m^2_{\tilde{L}_3}$	Squark and slepton masses
A_0	Trilinear coupling
μ, m_{A^0}, $\tan\beta$	Higgs sector

phenomenological MSSM (pMSSM)

- 105 parameters
- No R-parity violation

  no CP violation

  no FCNC
- 105 $\rightarrow$ 11 parameters



1. SUSY
2. Neural Network
3. Event Generation
4. Results

Interpolation

- Unknown function f: $f(x_i) = y_i, i = 1, 2, ..., n$
- Construct from $(x_1, y_1), ..., (x_n, y_n)$
  $\rightarrow$ Interpolated function g: $g(x_i) = y_i$
- Minimal disagreement between f & g

HOW?

- Divide the set into:

$$\underbrace{\{(x_1, y_1), ..., (x_m, y_m)\}}_{\text{interpolation set}}\underbrace{\{(x_{m+1}, y_{m+1}), ..., (x_n, y_n)\}}_{\text{validation set}}$$

- Interpolation set $\rightarrow$ construct g
  validation set $\rightarrow$ measure disagreement between f & g

Interpolation

- Unknown function f: $f(x_i) = y_i, i = 1, 2, ..., n$

- Construct from $(x_1, y_1), ..., (x_n, y_n)$
  $\rightarrow$ Interpolated function g: $g(x_i) = y_i$

- Minimal disagreement between f & g

HOW?

- Divide the set into:

Interpolation

- Unknown function f: $f(x_i) = y_i, i = 1, 2, ..., n$
- Construct from $(x_1, y_1), ..., (x_n, y_n)$
  $\rightarrow$ Interpolated function g: $g(x_i) = y_i$
- Minimal disagreement between f & g

HOW?

- Divide the set into:

$$\underbrace{\{(x_1, y_1), ..., (x_m, y_m)\}}_{\text{interpolation set}}\underbrace{\{(x_{m+1}, y_{m+1}), ..., (x_n, y_n)\}}_{\text{validation set}}$$

- Interpolation set $\rightarrow$ construct g
  validation set $\rightarrow$ measure disagreement between f & g

Interpolation

- Unknown function f: $f(x_i) = y_i, i = 1, 2, \ldots, n$
- Construct from $(x_1, y_1), \ldots, (x_n, y_n)$
  $\rightarrow$ Interpolated function g: $g(x_i) = y_i$
- Minimal disagreement between f & g

HOW?

- Divide the set into:

$$\underbrace{\{(x_1, y_1), \ldots, (x_m, y_m)\}}_{\text{interpolation set}} \cup \underbrace{\{(x_{m+1}, y_{m+1}), \ldots, (x_n, y_n)\}}_{\text{validation set}}$$

- Interpolation set $\rightarrow$ construct g
  validation set $\rightarrow$ measure disagreement between f & g

Interpolation

- Unknown function f: $f(x_i) = y_i, i = 1, 2, ..., n$
- Construct from $(x_1, y_1), ..., (x_n, y_n)$
  $\rightarrow$ Interpolated function g: $g(x_i) = y_i$
- Minimal disagreement between f & g

HOW?

- Divide the set into:

$$\underbrace{\{(x_1, y_1), ..., (x_m, y_m)\}}_{\text{interpolation set}} \underbrace{\{(x_{m+1}, y_{m+1}), ..., (x_n, y_n)\}}_{\text{validation set}}$$

- Interpolation set $\rightarrow$ construct g
  validation set $\rightarrow$ measure disagreement between f & g

Interpolation

- Unknown function f: $f(x_i) = y_i, i = 1, 2, ..., n$
- Construct from $(x_1, y_1), ..., (x_n, y_n)$
  $\rightarrow$ Interpolated function g: $g(x_i) = y_i$
- Minimal disagreement between f & g

HOW?

- Divide the set into:

$$\underbrace{\{(x_1, y_1), ..., (x_m, y_m)\}}_{\text{interpolation set}}\underbrace{\{(x_{m+1}, y_{m+1}), ..., (x_n, y_n)\}}_{\text{validation set}}$$

- Interpolation set $\rightarrow$ construct g
  validation set $\rightarrow$ measure disagreement between f & g

Interpolation

- Unknown function f: $f(x_i) = y_i, i = 1, 2, ..., n$
- Construct from $(x_1, y_1), ..., (x_n, y_n)$
  $\rightarrow$ Interpolated function g: $g(x_i) = y_i$
- Minimal disagreement between f & g

HOW?

- Divide the set into:

$$\underbrace{\{(x_1, y_1), ..., (x_m, y_m)\}}_{\text{interpolation set}} \underbrace{\{(x_{m+1}, y_{m+1}), ..., (x_n, y_n)\}}_{\text{validation set}}$$

- Interpolation set $\rightarrow$ construct g
  validation set $\rightarrow$ measure disagreement between f & g

Interpolation

- Unknown function f: $f(x_i) = y_i, i = 1, 2, ..., n$
- Construct from $(x_1, y_1), ..., (x_n, y_n)$
  $\rightarrow$ Interpolated function g: $g(x_i) = y_i$
- Minimal disagreement between f & g

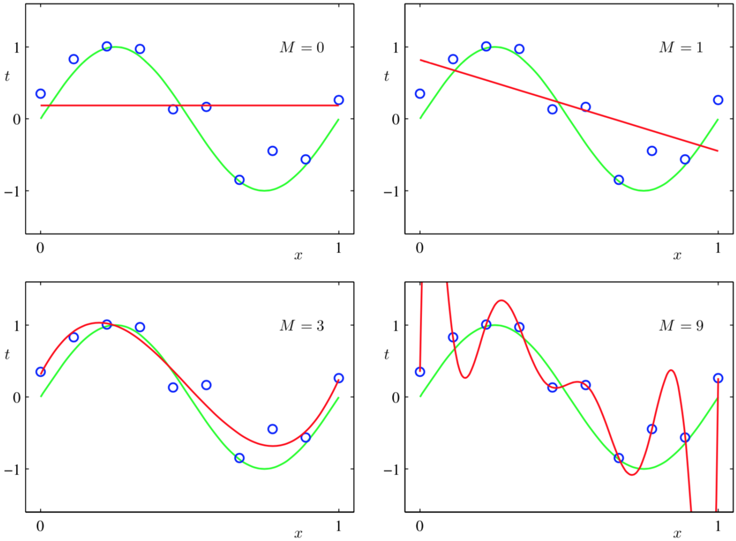
HOW?

- Divide the set into:

$$\underbrace{\{(x_1, y_1), ..., (x_m, y_m)\}}_{\text{interpolation set}} \underbrace{\{(x_{m+1}, y_{m+1}), ..., (x_n, y_n)\}}_{\text{validation set}}$$

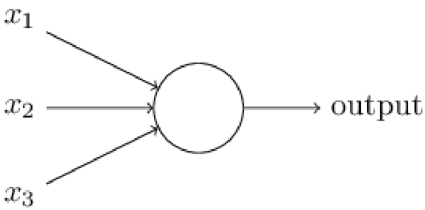
- Interpolation set $\rightarrow$ construct g
  validation set $\rightarrow$ measure disagreement between f & g

Simplified Example for Interpolation



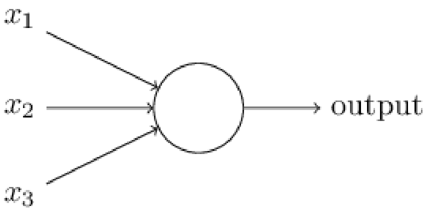
Source: Pattern Recognition and Machine Learning [Bishop]

Neural Network



- Inputs: $x_i, i = 1, 2, ..., n$
- Weights: $w_i, i = 1, 2, ..., n$
- Bais: b
- Activation function:

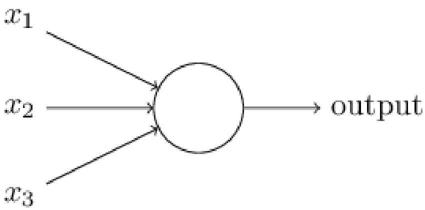
Neural Network



- Inputs: $x_i, i = 1, 2, ..., n$
- Weights: $w_i, i = 1, 2, ..., n$
- Bais: b
- Activation function:

$$a(\sum_{i=1}^{n} w_i x_i + b) = a(w.x + b) = \text{output}$$

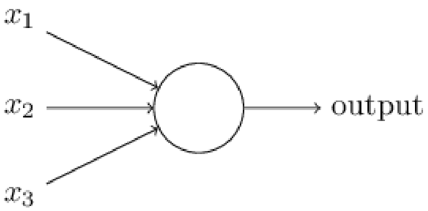
Neural Network



- Inputs: $x_i, i = 1, 2, ..., n$
- Weights: $w_i, i = 1, 2, ..., n$
- Bais: b
- Activation function:

$$a(\sum_{i=1}^{n} w_i x_i + b) = a(w \cdot x + b) = output$$

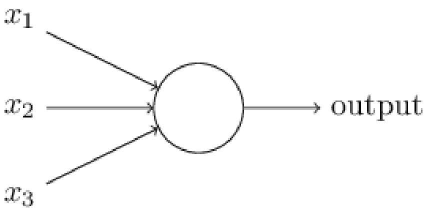
Neural Network



- Inputs: $x_i, i = 1, 2, ..., n$
- Weights: $w_i, i = 1, 2, ..., n$
- Bais: b
- Activation function:

$$a(\sum_{i=1}^{n} w_i x_i + b) = a(w.x + b) = \text{output}$$

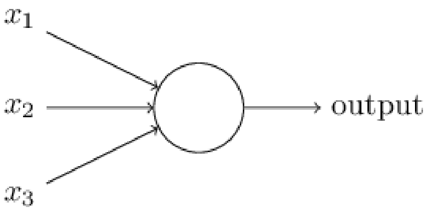
Neural Network



- Inputs: $x_i, i = 1, 2, ..., n$
- Weights: $w_i, i = 1, 2, ..., n$
- Bais: b
- Activation function:

$$a(\sum_{i=1}^{n} w_i x_i + b) = a(w.x + b) = \text{output}$$

Neural Network



- Inputs: $x_i, i = 1, 2, ..., n$
- Weights: $w_i, i = 1, 2, ..., n$
- Bais: b
- Activation function:

$$a(\sum_{i=1}^{n} w_i x_i + b) = a(w.x + b) = \text{output}$$

Cost function

- How can we judge if the parameters are chosen well?

Cost function

Cost function

- How can we judge if the parameters are chosen well?

Cost function

- Quadratic cost function: $C(w,b) = \frac{1}{2n}\sum_x ||y(x) - a||^2$
- $C = C(w,b) + \frac{\lambda}{2n}|w|^2$
- If $y(x) \simeq a \rightarrow C(w,b) \simeq 0$ (for all training inputs)

Aim of training algorithm is minimising $C(w,b)$

Cost function

- How can we judge if the parameters are chosen well?

Cost function

- Quadratic cost function: $C(w,b) = \frac{1}{2n}\sum_x ||y(x) - a||^2$
- $C = C(w,b) + \frac{\lambda}{2n}|w|^2$
- If $y(x) \approx a \to C(w,b) \approx 0$ (for all training inputs)

Aim of training algorithm is minimising $C(w,b)$

Cost function

- How can we judge if the parameters are chosen well?

Cost function

- Quadratic cost function: $C(w, b) = \frac{1}{2n}\sum_x ||y(x) - a||^2$
- $C = C(w, b) + \frac{\lambda}{2n}|w|^2$
- If $y(x) \approx a \rightarrow C(w, b) \approx 0$ (for all training inputs)

Aim of training algorithm is minimising $C(w, b)$

Cost function

- How can we judge if the parameters are chosen well?

Cost function

- Quadratic cost function: $C(w, b) = \frac{1}{2n} \sum_x ||y(x) - a||^2$
- $C = C(w, b) + \frac{\lambda}{2n} |w|^2$
- If $y(x) \approx a \rightarrow C(w, b) \approx 0$ (for all training inputs)

Aim of training algorithm is minimising $C(w, b)$

Cost function

- How can we judge if the parameters are chosen well?

Cost function

- Quadratic cost function: $C(w,b) = \frac{1}{2n}\sum_x ||y(x) - a||^2$
- $C = C(w,b) + \frac{\lambda}{2n}|w|^2$
- If $y(x) \approx a \rightarrow C(w,b) \approx 0$ (for all training inputs)

Aim of training algorithm is minimising $C(w,b)$

Cost function

- How can we judge if the parameters are chosen well?

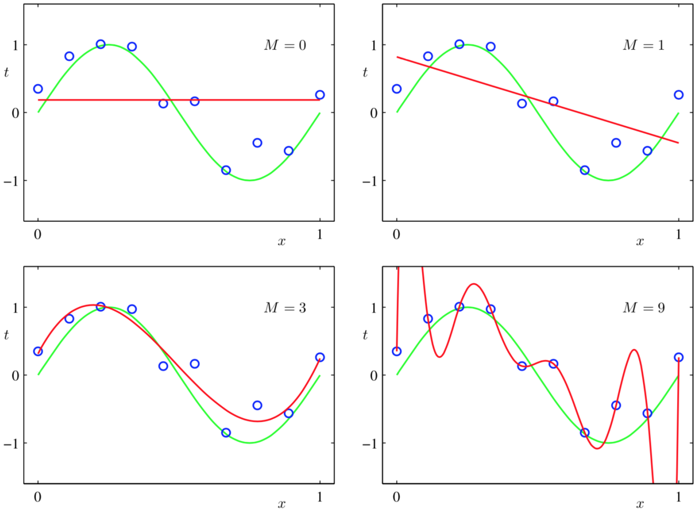
Cost function

- Quadratic cost function: $C(w,b) = \frac{1}{2n}\sum_x ||y(x) - a||^2$
- $C = C(w,b) + \frac{\lambda}{2n}|w|^2$
- If $y(x) \approx a \rightarrow C(w,b) \approx 0$ (for all training inputs)

Aim of training algorithm is minimising $C(w,b)$

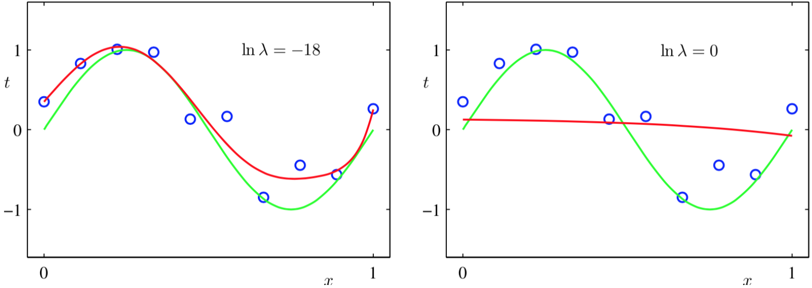
Overfitting

Total error on the validation data does not decrease (even increase)



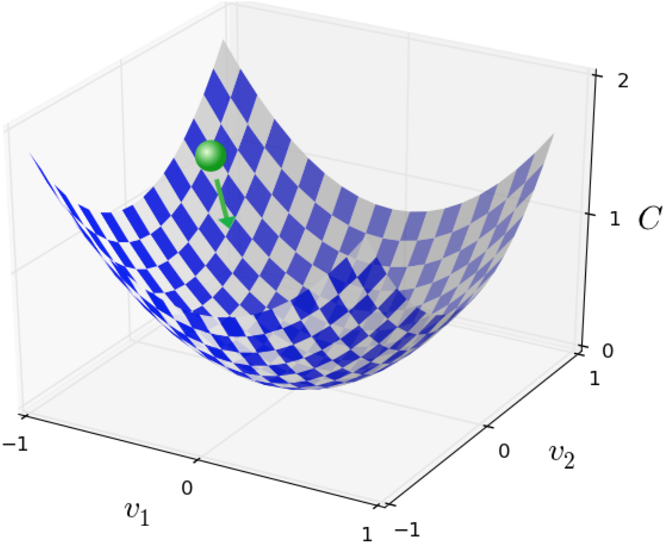
Source: Pattern Recognition and Machine Learning [Bishop]

Regularization



Source: Pattern Recognition and Machine Learning [Bishop]

Gradient Descent



Source: Neural Networks and Deep Learning [Nielsen]

1. SUSY
2. Neural Network
3. Event Generation
4. Results

2 Neural Network approaches

1. Direct: Train NN using parameters of pMSSM-11 as an input $\rightarrow$ global pMSSM-11 fits $\Rightarrow$ fast

2. Indirect: Train NN using model-independent signature-related objects $\rightarrow$ slower than direct approach

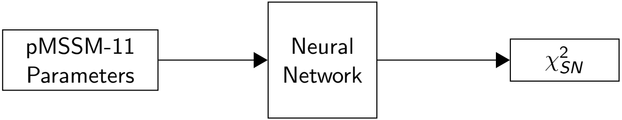
2 Neural Network approaches

1. Direct: Train NN using parameters of pMSSM-11 as an input $\rightarrow$ global pMSSM-11 fits $\Rightarrow$ fast

2. Indirect: Train NN using model-independent signature-related objects $\Rightarrow$ slower than direct approach

2 Neural Network approaches

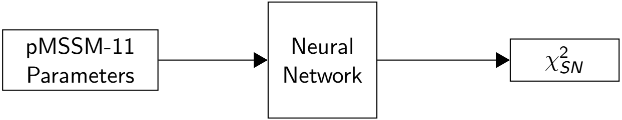
1. Direct: Train NN using parameters of pMSSM-11 as an input $\rightarrow$ global pMSSM-11 fits $\Rightarrow$ fast



2. Indirect: Train NN using model-independent signature-related objects $\Rightarrow$ slower than direct approach

2 Neural Network approaches

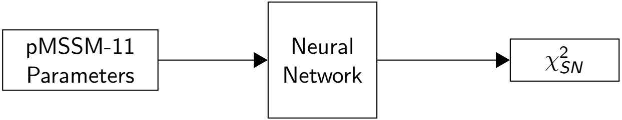
1. Direct: Train NN using parameters of pMSSM-11 as an input $\rightarrow$ global pMSSM-11 fits $\Rightarrow$ fast



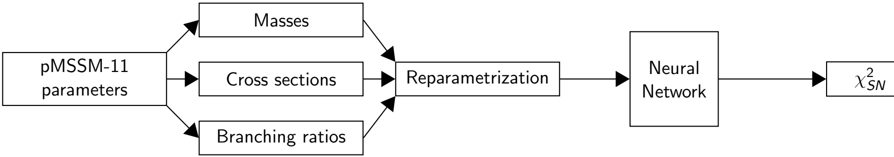
2. Indirect: Train NN using model-independent signature-related objects $\Rightarrow$ slower than direct approach

2 Neural Network approaches

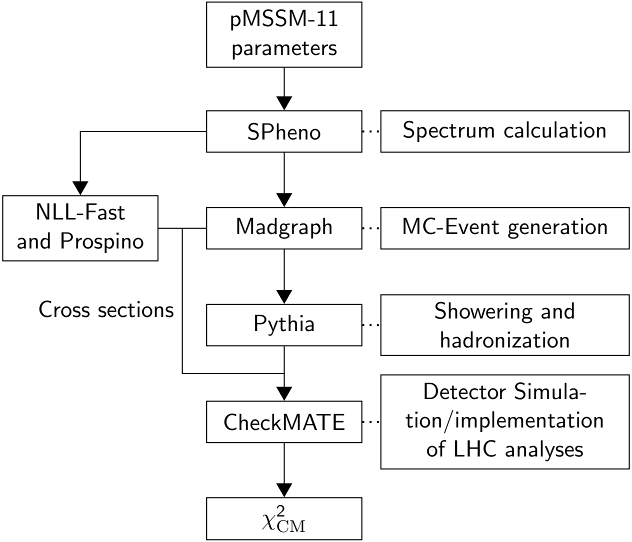
1. Direct: Train NN using parameters of pMSSM-11 as an input $\rightarrow$ global pMSSM-11 fits $\Rightarrow$ fast



2. Indirect: Train NN using model-independent signature-related objects $\Rightarrow$ slower than direct approach



Flow Chart of Simulation Chain



Selecting non-Overlapping Signal Regions

Calculation of total χ^2

- All selected SRs: $\chi_j^2 = \sum_k \chi_{jk}^2$
- Overall: $\chi_{tot}^2 = \sum_j \chi_j^2$

$$\text{Orthogonality Matrix} = \begin{array}{c|cccc} & A & B & C & D \\ \hline A & 0 & 1 & 0 & 0 \\ B & 1 & 0 & 1 & 0 \\ C & 0 & 1 & 0 & 0 \\ D & 0 & 0 & 0 & 0 \end{array}$$

Selecting non-Overlapping Signal Regions

Calculation of total χ^2

- All selected SRs: $\chi_j^2 = \sum_k \chi_{jk}^2$
- Overall: $\chi_{tot}^2 = \sum_j \chi_j^2$

$$\text{Orthogonality Matrix} = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Selecting non-Overlapping Signal Regions

Calculation of total χ^2

- All selected SRs: $\chi^2_j = \sum_k \chi^2_{jk}$
- Overall: $\chi^2_{tot} = \sum_j \chi^2_j$

$$\text{Orthogonality Matrix} = \begin{array}{c c} & \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{array}{c} A \\ B \\ C \\ D \end{array} & \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Selecting non-Overlapping Signal Regions

Calculation of total χ^2

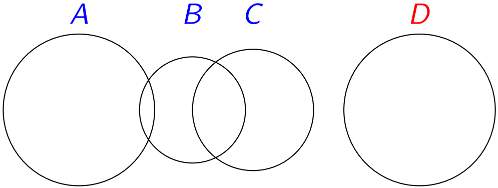
- All selected SRs: $\chi_j^2 = \sum_k \chi_{jk}^2$
- Overall: $\chi_{tot}^2 = \sum_j \chi_j^2$

$$\text{Orthogonality Matrix} = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Selecting non-Overlapping Signal Regions

Calculation of total χ^2

- All selected SRs: $\chi^2_j = \sum_k \chi^2_{jk}$
- Overall: $\chi^2_{tot} = \sum_j \chi^2_j$

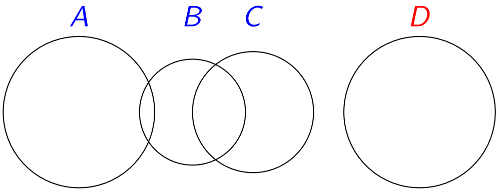


$$\text{Orthogonality Matrix} = \begin{array}{c c} & \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{array}{c} A \\ B \\ C \\ D \end{array} & \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Selecting non-Overlapping Signal Regions

Calculation of total χ^2

- All selected SRs: $\chi_j^2 = \sum_k \chi_{jk}^2$
- Overall: $\chi_{tot}^2 = \sum_j \chi_j^2$



$$\text{Orthogonality Matrix} = \begin{array}{c c} & \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{array}{c} A \\ B \\ C \\ D \end{array} & \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Profile Likelihood Ratio

$$\mathscr{L}(N_{obs}|\mu, \nu_S, \nu_{SM}) = \frac{e^{-\lambda}}{N_{obs}!}\,\lambda^{N_{obs}}\,\frac{e^{-\frac{\nu_S^2}{2}}}{\sqrt{2\pi}}\,\frac{e^{-\frac{\nu_{SM}^2}{2}}}{\sqrt{2\pi}}$$

$$\lambda(\nu_S, \nu_{SM}, \mu) = \mu\, N_S\, e^{\frac{\sigma_{N_S}}{N_S}\nu_S} + N_{SM}\, e^{\frac{\sigma_{N_{SM}}}{N_{SM}}\nu_{SM}}$$

$$PLR = \frac{\mathscr{L}_C}{\mathscr{L}_G} = \frac{\max_{\nu_S, \nu_{SM} \in R}\mathscr{L}(\mu = 1, \nu_S, \nu_{SM})}{\max_{\mu, \nu_S, \nu_{SM} \in R}\mathscr{L}(\mu, \nu_S, \nu_{SM})}$$

$$q_\mu = -2\log PRL$$

q_μ is asymptotically χ^2 distributed in the limit of large N_{obs}

Profile Likelihood Ratio

$$\mathscr{L}(N_{obs}|\mu,\nu_S,\nu_{SM}) = \frac{e^{-\lambda}}{N_{obs}!}\,\lambda^{N_{obs}}\,\frac{e^{-\frac{\nu_S^2}{2}}}{\sqrt{2\pi}}\,\frac{e^{-\frac{\nu_{SM}^2}{2}}}{\sqrt{2\pi}}$$

$$\lambda(\nu_S,\nu_{SM},\mu) = \mu\, N_S\, e^{\frac{\sigma_{N_S}}{N_S}\nu_S} + N_{SM}\, e^{\frac{\sigma_{N_{SM}}}{N_{SM}}\nu_{SM}}$$

$$PLR = \frac{\mathscr{L}_C}{\mathscr{L}_G} = \frac{\max_{\nu_S,\nu_{SM}\in R}\mathscr{L}(\mu=1,\nu_S,\nu_{SM})}{\max_{\mu,\nu_S,\nu_{SM}\in R}\mathscr{L}(\mu,\nu_S,\nu_{SM})}$$

$$q_\mu = -2\log PRL$$

q_μ is asymptotically χ^2 distributed in the limit of large N_{obs}

Profile Likelihood Ratio

$$\mathscr{L}(N_{obs}|\mu,\nu_S,\nu_{SM}) = \frac{e^{-\lambda}}{N_{obs}!}\,\lambda^{N_{obs}}\,\frac{e^{-\frac{\nu_S^2}{2}}}{\sqrt{2\pi}}\,\frac{e^{-\frac{\nu_{SM}^2}{2}}}{\sqrt{2\pi}}$$

$$\lambda(\nu_S,\nu_{SM},\mu) = \mu\,N_S\,e^{\frac{\sigma_{N_S}}{N_S}\nu_S} + N_{SM}\,e^{\frac{\sigma_{N_{SM}}}{N_{SM}}\nu_{SM}}$$

$$PLR = \frac{\mathscr{L}_C}{\mathscr{L}_G} = \frac{\max_{\nu_S,\nu_{SM}\in R}\mathscr{L}(\mu=1,\nu_S,\nu_{SM})}{\max_{\mu,\nu_S,\nu_{SM}\in R}\mathscr{L}(\mu,\nu_S,\nu_{SM})}$$

$$q_\mu = -2\log PRL$$

q_μ is asymptotically χ^2 distributed in the limit of large N_{obs}

Profile Likelihood Ratio

$$\mathscr{L}(N_{obs}|\mu, \nu_S, \nu_{SM}) = \frac{e^{-\lambda}}{N_{obs}!} \lambda^{N_{obs}} \frac{e^{-\frac{\nu_S^2}{2}}}{\sqrt{2\pi}} \frac{e^{-\frac{\nu_{SM}^2}{2}}}{\sqrt{2\pi}}$$

$$\lambda(\nu_S, \nu_{SM}, \mu) = \mu\, N_S\, e^{\frac{\sigma_{N_S}}{N_S}\nu_S} + N_{SM}\, e^{\frac{\sigma_{N_{SM}}}{N_{SM}}\nu_{SM}}$$

$$PLR = \frac{\mathscr{L}_C}{\mathscr{L}_G} = \frac{\max_{\nu_S,\nu_{SM}\in R} \mathscr{L}(\mu = 1, \nu_S, \nu_{SM})}{\max_{\mu,\nu_S,\nu_{SM}\in R} \mathscr{L}(\mu, \nu_S, \nu_{SM})}$$

$$q_\mu = -2 \log PRL$$

q_μ is asymptotically χ^2 distributed in the limit of large N_{obs}

Profile Likelihood Ratio

$$\mathscr{L}(N_{obs}|\mu,\nu_S,\nu_{SM}) = \frac{e^{-\lambda}}{N_{obs}!}\,\lambda^{N_{obs}}\,\frac{e^{-\frac{\nu_S^2}{2}}}{\sqrt{2\pi}}\,\frac{e^{-\frac{\nu_{SM}^2}{2}}}{\sqrt{2\pi}}$$

$$\lambda(\nu_S,\nu_{SM},\mu) = \mu\,N_S\,e^{\frac{\sigma_{N_S}}{N_S}\nu_S} + N_{SM}\,e^{\frac{\sigma_{N_{SM}}}{N_{SM}}\nu_{SM}}$$

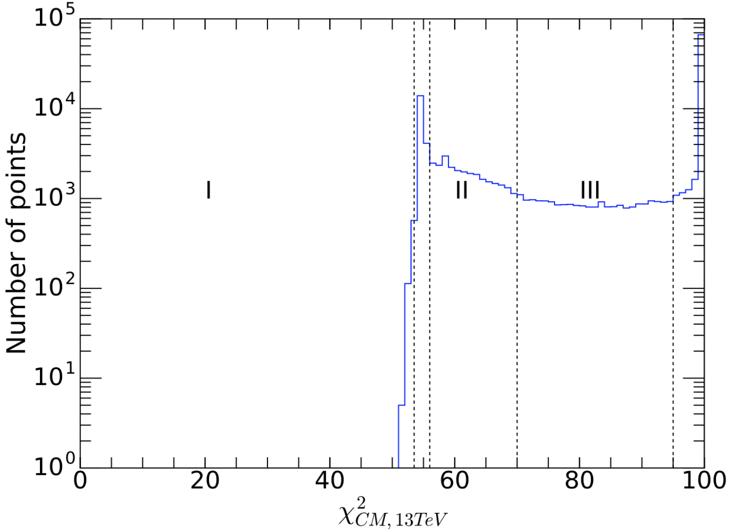
$$PLR = \frac{\mathscr{L}_C}{\mathscr{L}_G} = \frac{\max_{\nu_S,\nu_{SM}\in R}\mathscr{L}(\mu=1,\nu_S,\nu_{SM})}{\max_{\mu,\nu_S,\nu_{SM}\in R}\mathscr{L}(\mu,\nu_S,\nu_{SM})}$$

$$q_\mu = -2\log PRL$$

q_μ is asymptotically χ^2 distributed in the limit of large N_{obs}

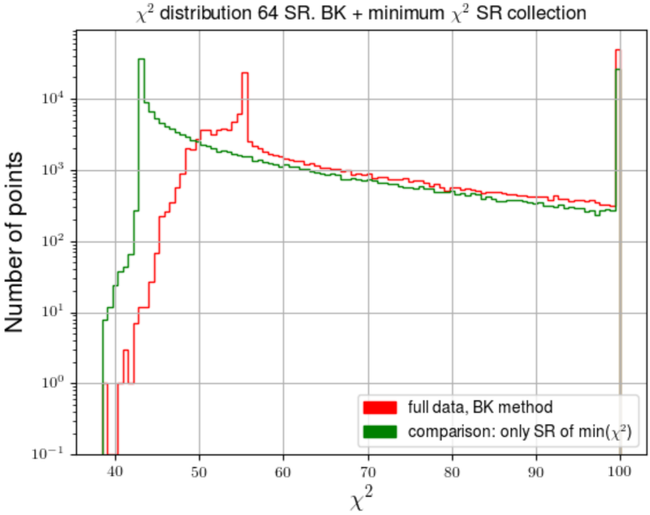
1. SUSY
2. Neural Network
3. Event Generation
4. Results

Distribution of χ^2 (Old result)



Source: arXiv 1703.01309

Distribution of χ^2 (New result)



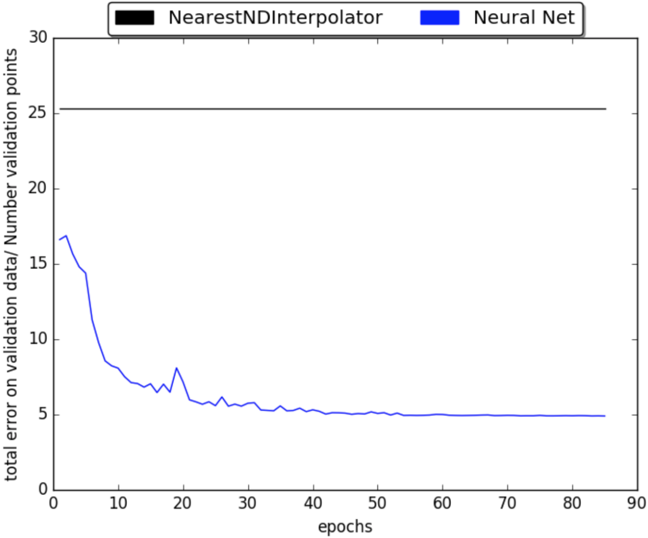
Source: Ferdinand Eiteneuer

Results

13 TeV (Old)						
χ^2 range	0 – 100	0 – 53.5	53.5 – 56	56 – 70	70 – 95	95 – 100
	1.45	1.78	0.92	2.35	3.95	0.49
13 TeV (New)						
χ^2 range	0 – 100	0 – 53.5	53.5 – 56	56 – 70	70 – 95	95 – 100
	1.49	1.97	0.57	1.46	3.01	0.66

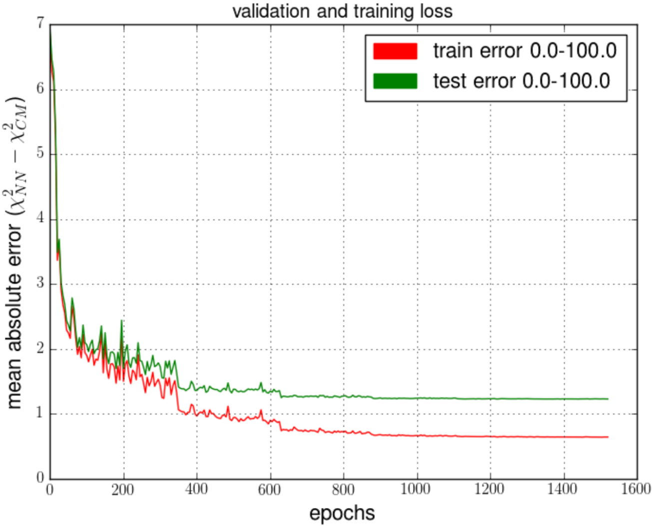
Table: Mean errors of NN for LHC χ^2

Evolution of the mean error (Old)



Source: Jan Schütte-Engel

Evolution of the mean error (New)



Source: Ferdinand Eiteneuer

Old Results from CheckMATE and SCYNet (8 TeV)

Project the 11-d pMSSM parameter space onto the masses of $\tilde{g}$ and $\tilde{\chi}_1^0$

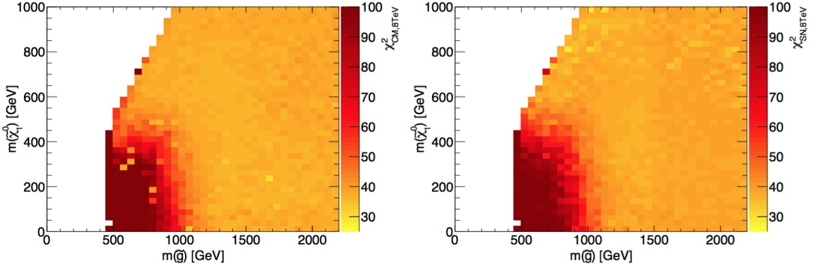


Figure: Minimum pMSSM-11 χ^2 in the $\tilde{g}$-$\tilde{\chi}_1^0$ mass plane

Source: arXiv 1703.01309

Comparison CM - SN

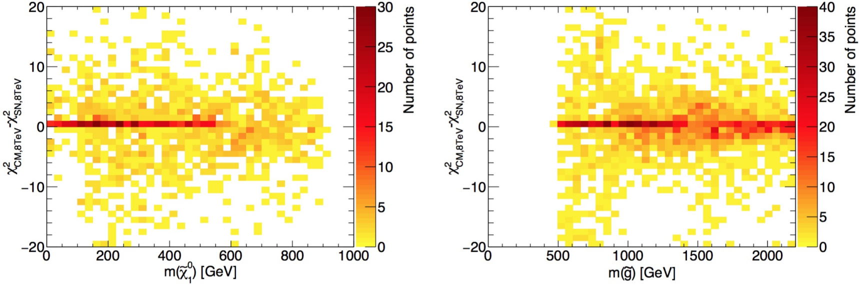


Figure: Difference between the CheckMATE and SCYNet χ^2

Source: arXiv 1703.01309

Summary

- Test SUSY against LHC data with a proper tool
  $\rightarrow$ SCYNet (SUSY Calculating Yield Net)
- 11 parameters as input, χ^2 as output
- Use Neural Network for reducing time.
- 13 TeV analyses (14 ATLAS + 1 CMS) for strong and electroweak processes
- Still need more statistics in RTLP regions to improve χ^2

Summary

- Test SUSY against LHC data with a proper tool
  $\rightarrow$ SCYNet (SUSY Calculating Yield Net)
- 11 parameters as input, χ^2 as output
- Use Neural Network for reducing time.
- 13 TeV analyses (14 ATLAS + 1 CMS) for strong and electroweak processes
- Still need more statistics in RTLP regions to improve χ^2

Summary

- Test SUSY against LHC data with a proper tool
  $\rightarrow$ SCYNet (SUSY Calculating Yield Net)
- 11 parameters as input, χ^2 as output
- Use Neural Network for reducing time.
- 13 TeV analyses (14 ATLAS + 1 CMS) for strong and electroweak processes
- Still need more statistics in RTLP regions to improve χ^2

Summary

- Test SUSY against LHC data with a proper tool
  $\rightarrow$ SCYNet (SUSY Calculating Yield Net)
- 11 parameters as input, χ^2 as output
- Use Neural Network for reducing time.
- 13 TeV analyses (14 ATLAS + 1 CMS) for strong and electroweak processes
- Still need more statistics in RTLP regions to improve χ^2

Summary

- Test SUSY against LHC data with a proper tool
  $\rightarrow$ SCYNet (SUSY Calculating Yield Net)
- 11 parameters as input, χ^2 as output
- Use Neural Network for reducing time.
- 13 TeV analyses (14 ATLAS + 1 CMS) for strong and electroweak processes
- Still need more statistics in RTLP regions to improve χ^2

Summary

- Test SUSY against LHC data with a proper tool
  $\rightarrow$ SCYNet (SUSY Calculating Yield Net)
- 11 parameters as input, χ^2 as output
- Use Neural Network for reducing time.
- 13 TeV analyses (14 ATLAS + 1 CMS) for strong and electroweak processes
- Still need more statistics in RTLP regions to improve χ^2

Thank you for your attention

References

- Philip Bechtle, Sebastian Belkner, Daniel Dercks, Matthias Hamer, Tim Keller, Michael Krämer, Björn Sarrazin, Jan Schütte-Engel, Jamie Tattersall, *SCYNet: Testing supersymmetric models at the LHC with neural networks*, arXiv:1703.01309v2 [hep-ph] 14 Mar 2017.
- Michael A. Nielsen, *Neural Networks and Deep Learning*, Determination Press, 2015.
- Christopher M. Bishop, *Pattern Recognition and Machine Learning*, Springer , 2006.
- Jan Schütte-Engel, *Searching for physics beyond the Standard Model at the LHC in the framework of the pMSSM-11 by using a neural network approach*, Master thesis, 2016.
- Ferdinand Eiteneuer, *Improving Supersymmetric Model Fits with Neural Networks*, Master thesis, 2017.

Preselection Criteria for LHC Event Generation

Use pMSSM-11 as model of interest

- No tachyons in the spectrum
- χ_1^0 is LSP
- $m_{h^0},\ m_{H^0} > 110\ GeV$
- $m_{\chi_1^\pm} > 103.5\ GeV$
- $5 \times \sqrt{\sigma_{theory}^2 + \sigma_{exp.}^2} > |\text{theory value} - \text{exp. value}|$
  for the EW precision observables
  $m_W,\quad \Delta m_s,\quad \text{BR}(B_s \to \mu\mu),\quad \text{BR}(b \to s\gamma),\quad \text{BR}(B_u \to \tau\nu)$

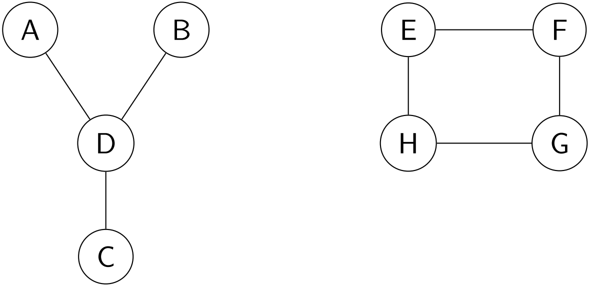
Restrict the pMSSM-11 parameter space to phenomenologically viable regions.

CheckMATE (Check Models At Terascale Energies)

- Test SUSY model against LHC data
- Check if the model is excluded or not at 95% C.L.

- Input
  Cross sections and their errors for all processes
- Output
  $\hat{N}_{i,j,k}$: number of signal events from process i in SR k of analysis j
  $N_{i,j,k}$: normalised number of signal events from process i in SR k of analysis j
  $N_{jk} = \sum_{i=1}^{n} N_{i,j,k}$: normalized number of events in SR k of analysis j for all processes + statistical error + systematical error
  O_{jk} number of observed events in SR k of analysis j (experiment)
  Logarithm of the profile likelihood ratio (PLR) for each SR k of each analysis j

Bron-Kerbosch Algorithm for Selecting disjoint SRs



- Orthogonal group 1: $\{A, B, C\}$, $\{D\}$
- Orthogonal group 2: $\{E, G\}$, $\{H, F\}$

More on Neural Network

- Sigmoid function:

$$a(w.x+b) = \frac{1}{1+\exp^{-(w.x+b)}}$$

- Output depends on all weights and biases

- Construction of g: training phase
  Interpolation set: training set

- Evaluation of the total error on validation set after training

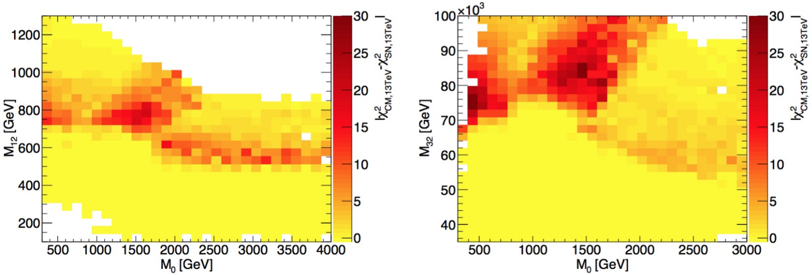
Adam Optimizer (Adaptive Moment estimation)

- Minimization algorithm which relies only on first order information
- 4 hyperparameters (chosen at the beginning but can be changed after each training epoch)

pMSSM-11 parameters and scan ranges

parameter	scan range
M_1	[-4000,4000] GeV
M_2	[100,4000] GeV
M_3	[-4000,-400]∪[400,4000] GeV
$m_{\tilde{q}_{12}}$	[300,5000] GeV
$m_{\tilde{q}_3}$	[100,5000] GeV
$m_{\tilde{l}_{12}}$	[100,3000] GeV
$m_{\tilde{l}_3}$	[100,4000] GeV
m_{A^0}	[0,4000] GeV
A^0	[-5000,5000] GeV
μ	[-5000,-100]∪[100,5000] GeV GeV
$\tan\beta$	[1,60]

Indirect approach



(a) (b)

Figure: Performance of the neural network trained on the reparametrized pMSSM-11 points on the cMSSM (a) and AMSB (b). In each bin the mean difference was calculated for all validation points.

Source: arXiv 1703.01309

Information of event generation

scanned points:

$\sqrt{s}$	**scanned points**
13 TeV (old)	140000
13 TeV (new)	170000

Disjoint signal regions:

$\sqrt{s}$	**disjoint SRs**
13 TeV (old)	65
13 TeV (new)	64