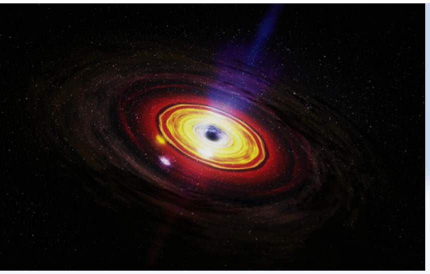
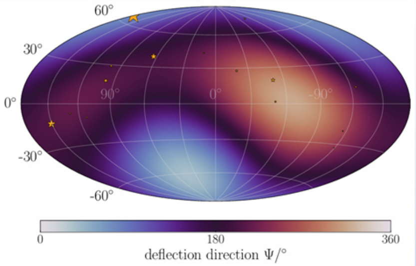
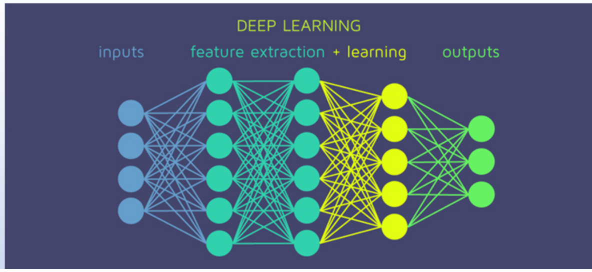


Coherent Galactic Magnetic Field Deflection Analysis with Conditional Invertible Neural Networks using an Active Galactic Nuclei Catalog

Dominik Wirtz, Martin Erdmann, Frederik Krieger, Josina Schulte





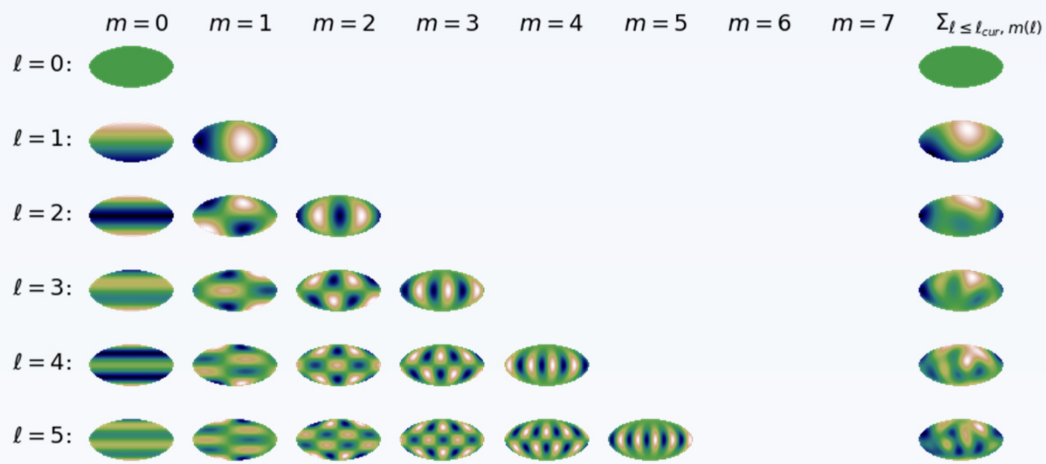
Ultra-high-energy cosmic rays (UHECRs)

- Can we identify coherently deflected (by toroidal and disk component of galactic magnetic field (GMF)) CRs originating from active galactic nuclei (AGN) in data?
- Current GMF-models are uncertain

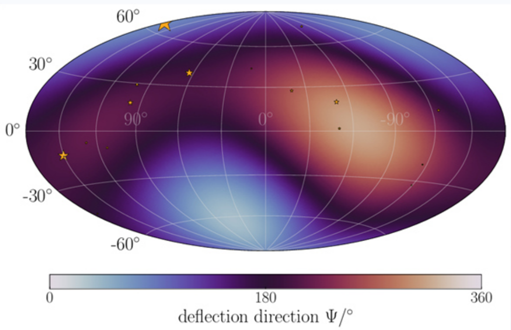
GMF Model

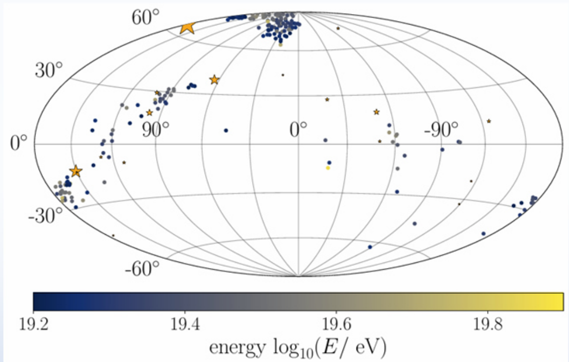
- GMF: coherent + turbulent deflections
- Turbulent deflections: Gaussian smearing
- GMF model consists of simulated spherical harmonic functions
- Spherical harmonics expansion resemble impact GMF has on deflections of CRs

$$\Psi = \sum_{l=1}^{l_{\max}} a_{lm} Y_{lm}$$



e.g. l_{max} = 2





- Observables: arrival direction **(AD)**, **energy** and $\boldsymbol{X_{\max}}$ of CRs

Method

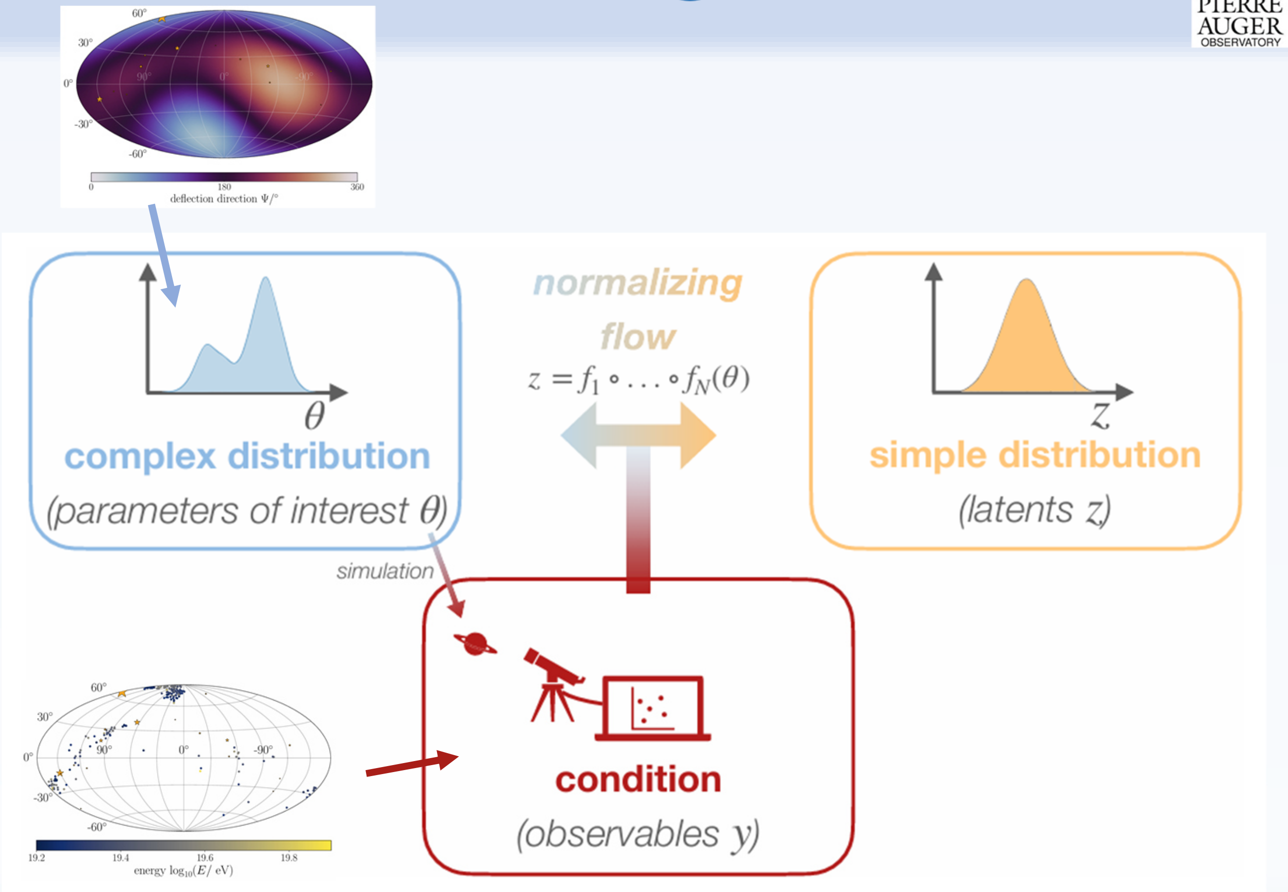
- Use **conditional invertible neural network (cINN)** to reconstruct spherical harmonics expansion coefficients

- Normalizing flows: Bijective functions

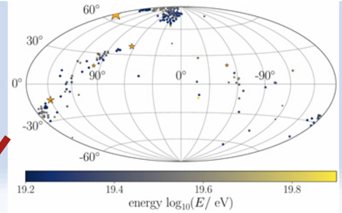
- Mapping between posteriors (complex distribution) and latents

- Training: forward →

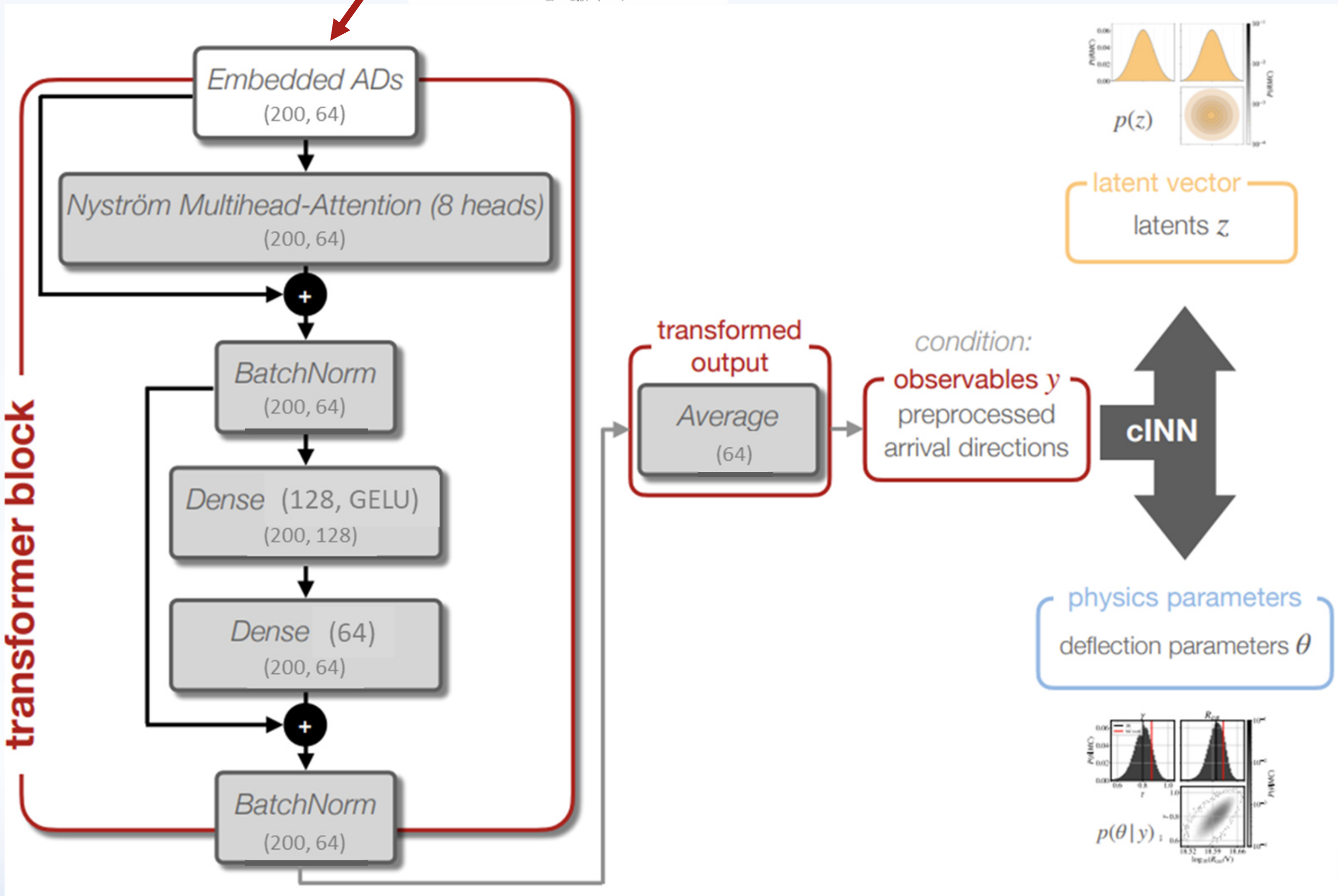
- Evaluation: backward ←



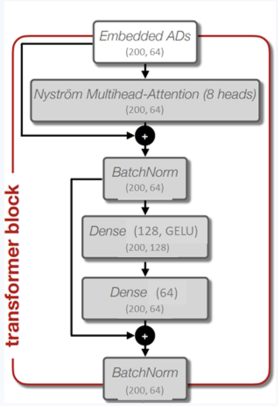
Transformer

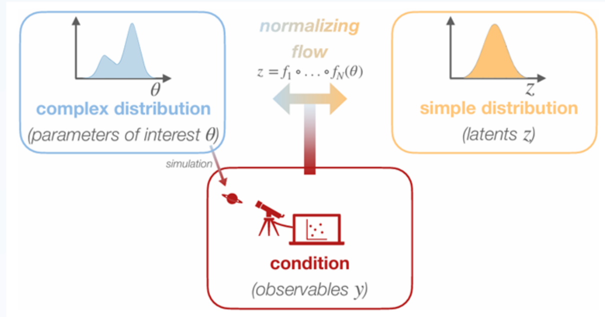


- 200 CRs with x,y,z (AD), energy, $X_{\max}$
- Attention module focuses on most prominent coherently deflected CRs
- helps preprocessing conditions for cINN



Goal





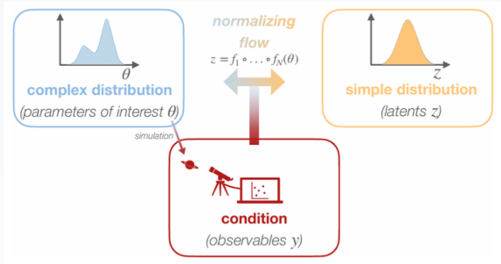
- cINNs have many hyperparameters which influence performance

- Which network hyperparameters suit best for fitting coefficients of spherical harmonic function (e.g. $l_{max} = 2$ → 8 parameters) for 200 CRs of 26 AGN sources?

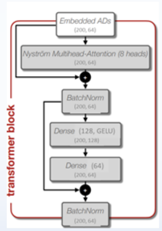
Network hyperparameters

- Data
  - l_{max}
  - **Number of training data**
  - **Batch size**

- cINN
  - **Number of invertible blocks**
  - **Internal size**

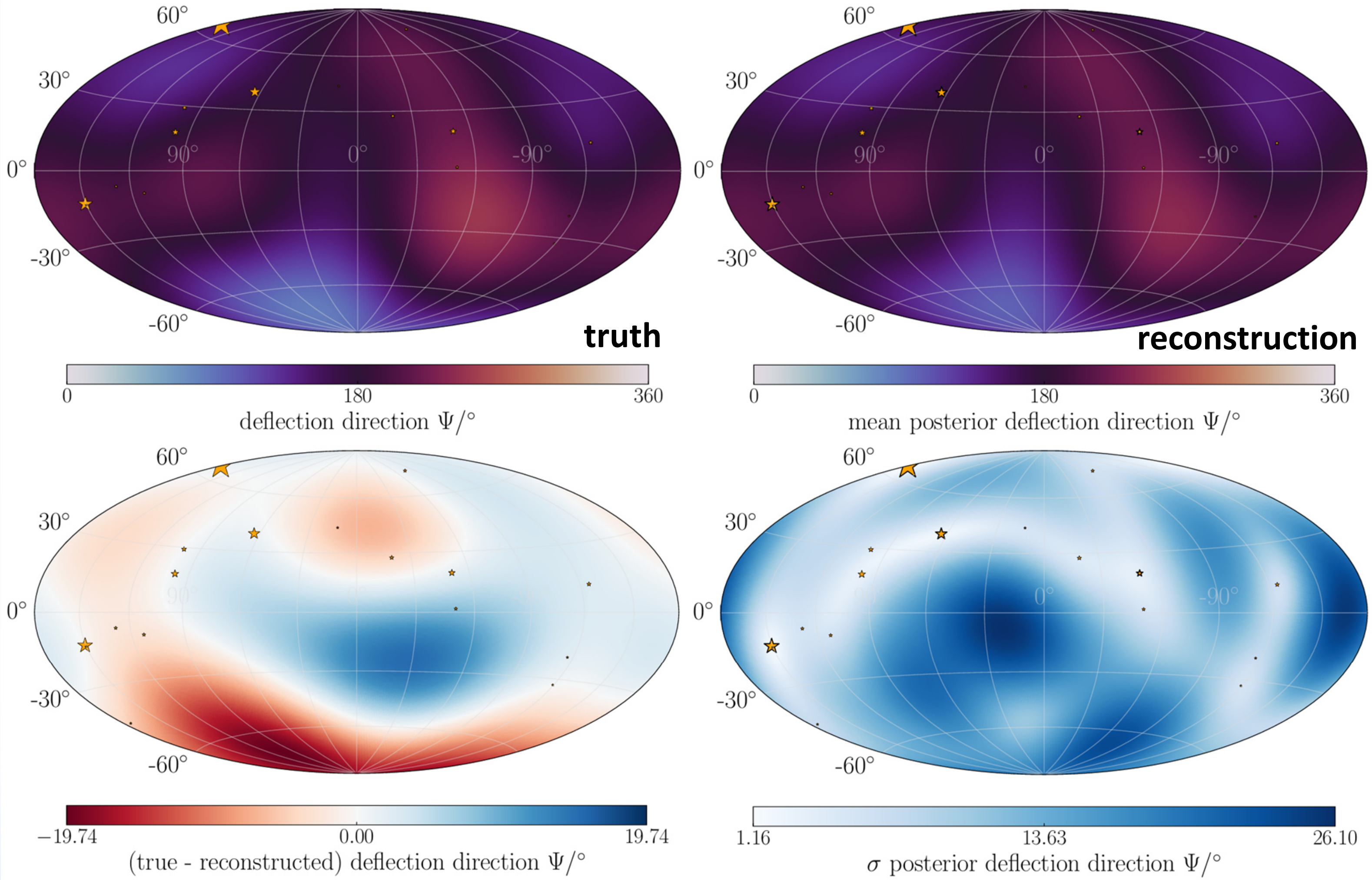


- Transformer
  - **Transformer dimension** (= dimension of condition)
  - **Number of layers**



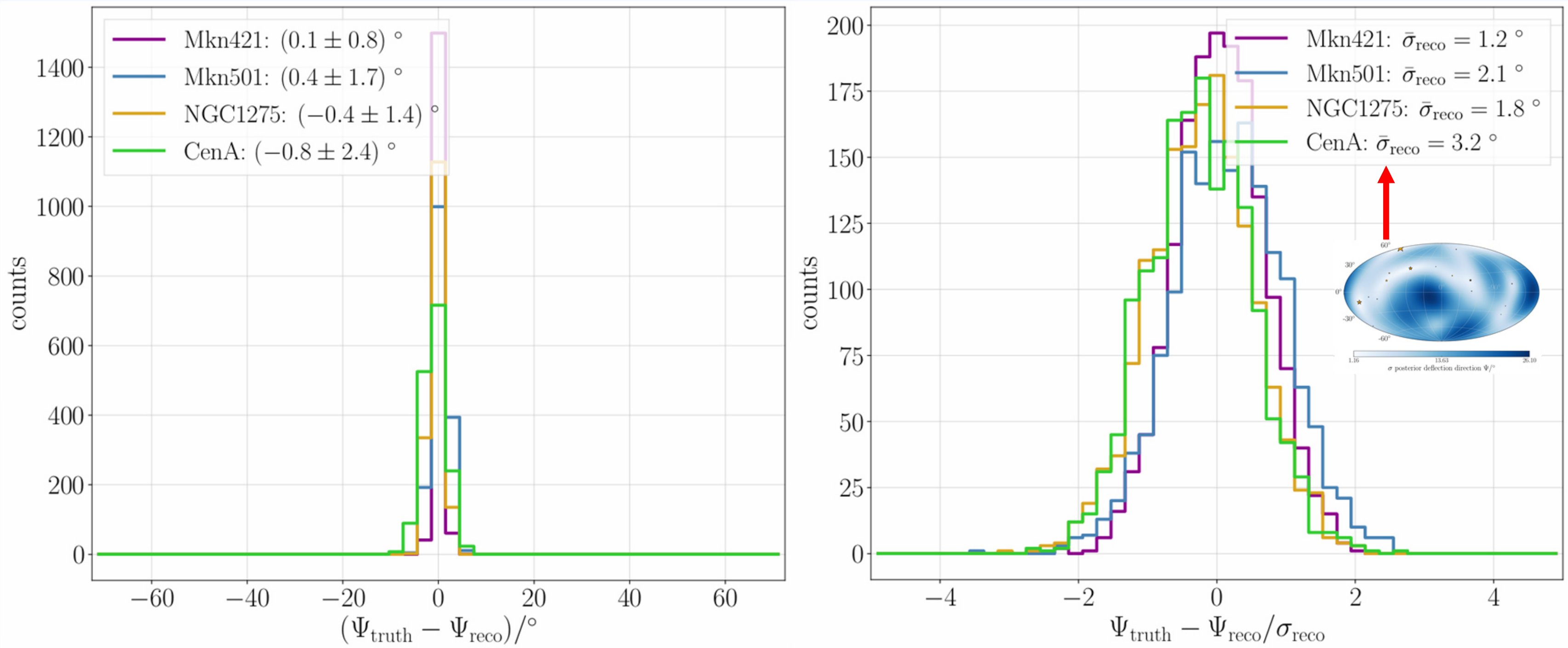
Example reconstruction

- Reconstruction works fine
- Uncertainties are small in regions with sources

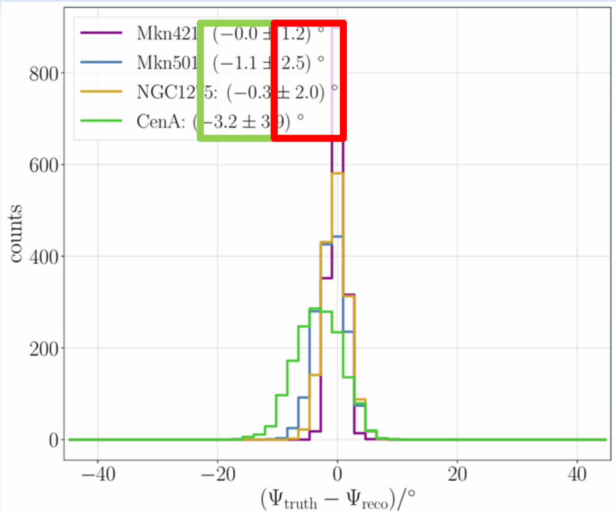


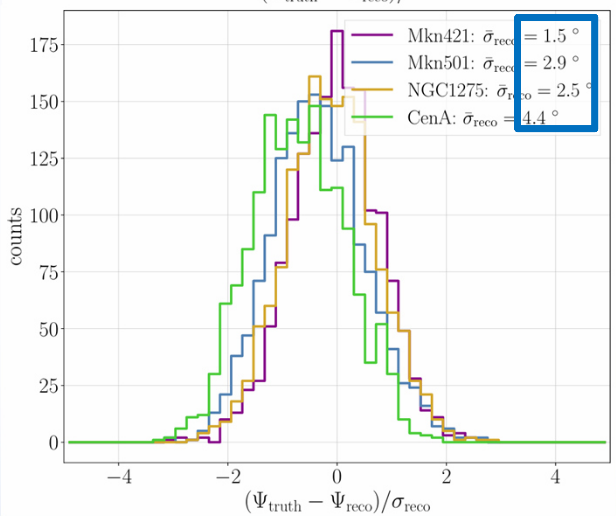
Sensitivity of the method

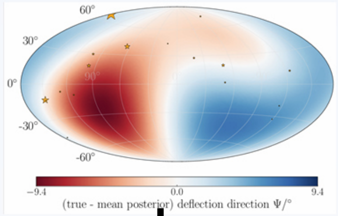
- Evaluating test data of size 1600 at 4 source positions
- Histogram values of $\delta = \Psi_{truth} - \Psi_{reco}$
- δ should be close to 0

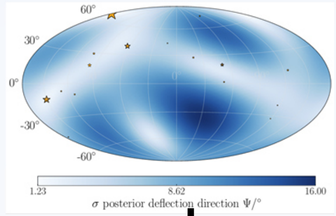


A measure for fitness of networks









$\langle \delta \rangle$ /°	$\langle \sigma_\delta \rangle$ /°	$\langle \sigma_{\mathrm{reco}} \rangle$ /°
0.56 ± 0.38	1.52 ± 0.85	1.90 ± 0.87

→ Averages over 4 source positions

$$\langle \widetilde{\mathrm{fit}}_{1\ldots4,1\ldots1600} \rangle = \left\langle \delta_{1\ldots1600} + \left(1 - \frac{\delta_{1\ldots1600}}{\sigma_{reco,1\ldots1600}}\right) \right\rangle_{1\ldots4}$$

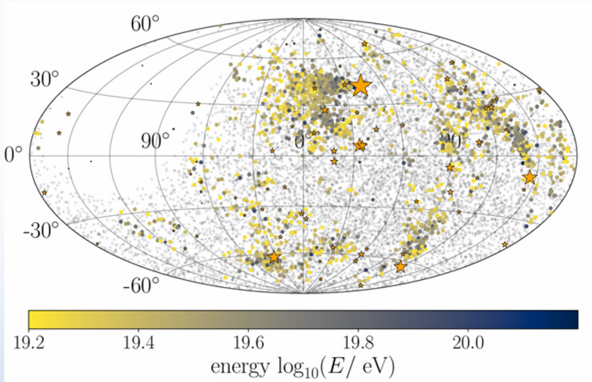
Results & Outlook

Results:

- Fitness parameter allows to identify best-performing network
- Networks are stable to variations of hyperparameters
- ~1 million trainable parameters are needed for good reconstruction
- Trend: Larger cINN and larger transformer work best for 200 CRs with 5 parameters each (w.r.t. tested parameter space)

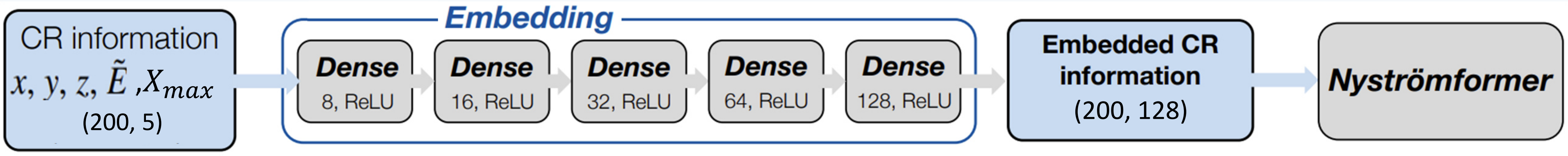
Outlook:

- Use Pierre Auger Observatory‘s exposure
- Include background CRs



Backup

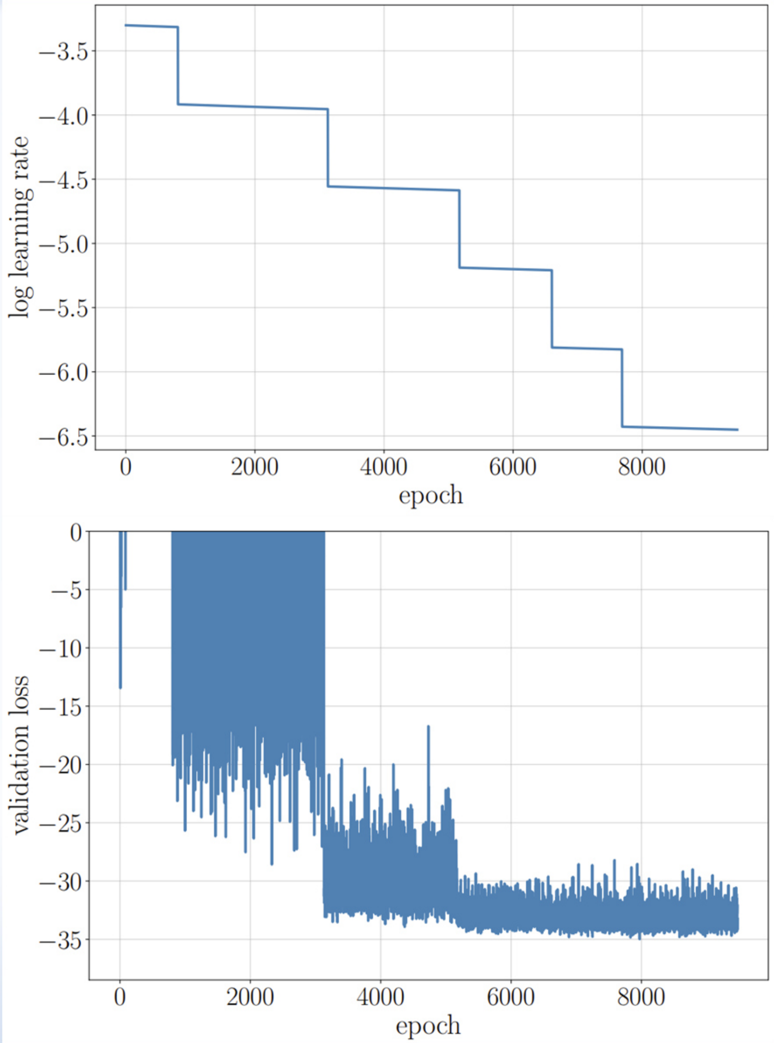
Network architecture - Embedding



Reduce learning rate on plateau + iteration decay

- **Reduce learning rate on plateau:**
  - Takes:
    - patience
    - reduction Factor
    - threshold (0.0)
  - if validation loss does not decrease after **patience**, reduces learning rate by a **factor**
  - allows for a **threshold** of deviations between validation losses of each epoch

- **Iteration decay:**
  - Takes:
    - iteration decay parameter (2.5e-6)
  - decreases initial learning rate with every epoch by a factor of 1/(1+**iteration decay parameter***epoch)



Loss function for cINNs

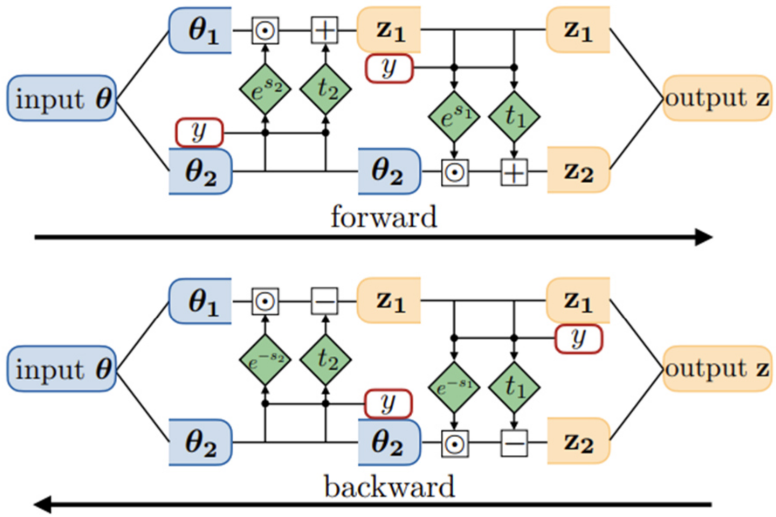


Fig. 4: Structure of the reversible block used for the conditional invertible neural network. It can be evaluated in two directions. The upper part shows the training mode or forward direction, the lower part displays the evaluation mode or backward direction.

<u>forward</u>

$$z_1 = \theta_1 \odot \exp(s_2(\theta_2)) + t_2(\theta_2)$$
$$z_2 = \theta_2 \odot \exp(s_1(z_1)) + t_1(z_1)$$

<u>backward</u>

$$\theta_2 = (z_2 - t_1(z_1)) \odot \exp(-s_1(z_1))$$
$$\theta_1 = (z_1 - t_2(\theta_2)) \odot \exp(-s_2(\theta_2))$$

Loss function for cINNs

θ is input
$\mathcal{Y}$ is condition

Kullback-Leibler Divergenz KL

$$
\begin{aligned}
L &= \mathbb{KL}\big(p(\theta|y) \parallel p_\phi(\theta|y)\big) \\
&= \mathbb{E}_{\theta\sim p(\theta|y)}\big(\log p(\theta|y) - \log p_\phi(\theta|y)\big) \\
&= \text{const.} + \mathbb{E}_{\theta\sim p(\theta|y)}\big(-\log p_\phi(\theta|y)\big)
\end{aligned}
$$

← provides a measure on the difference of two probability distributions

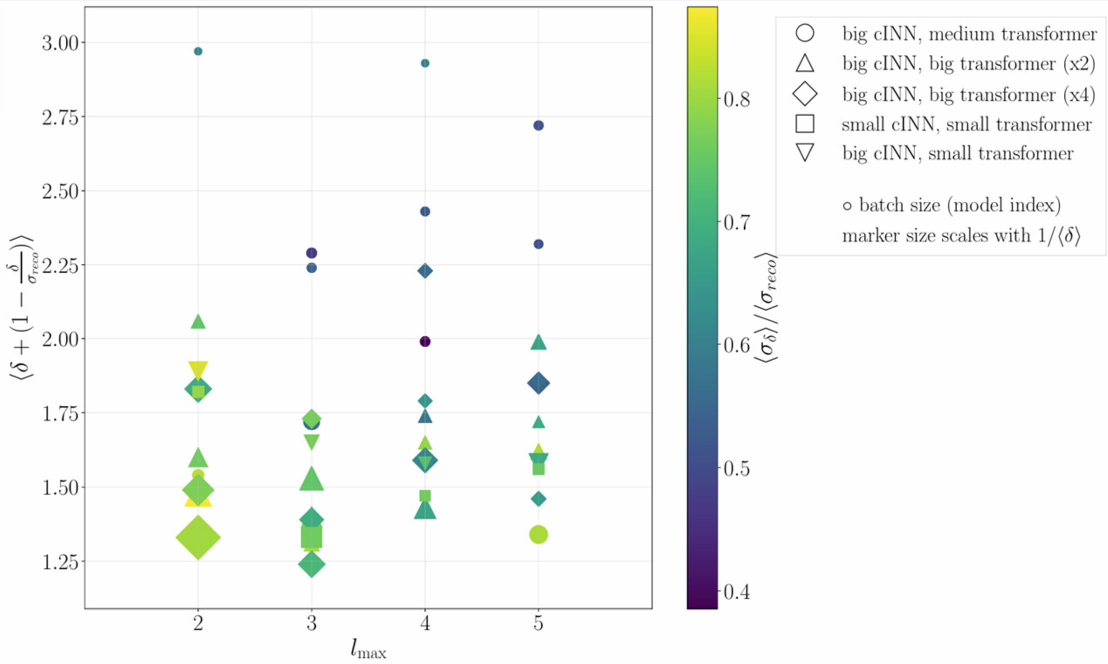
← true posterior not dependent on network parameters

$$
\begin{aligned}
L &= \mathbb{E}_{\theta\sim p(\theta|y)}\big(-\log p_\phi(\theta|y)\big) \\
&= \mathbb{E}_{\theta\sim p(\theta|y)}\left(-\log \left(p(z) \cdot \left|\det\left(\frac{\partial z}{\partial \theta}\right)\right|\right)\right) \\
&= \mathbb{E}_{\theta\sim p(\theta|y)}\left(-\log \left(p(z)\right) - \log\left(\left|\det\left(\frac{\partial z}{\partial \theta}\right)\right|\right)\right)
\end{aligned}
$$

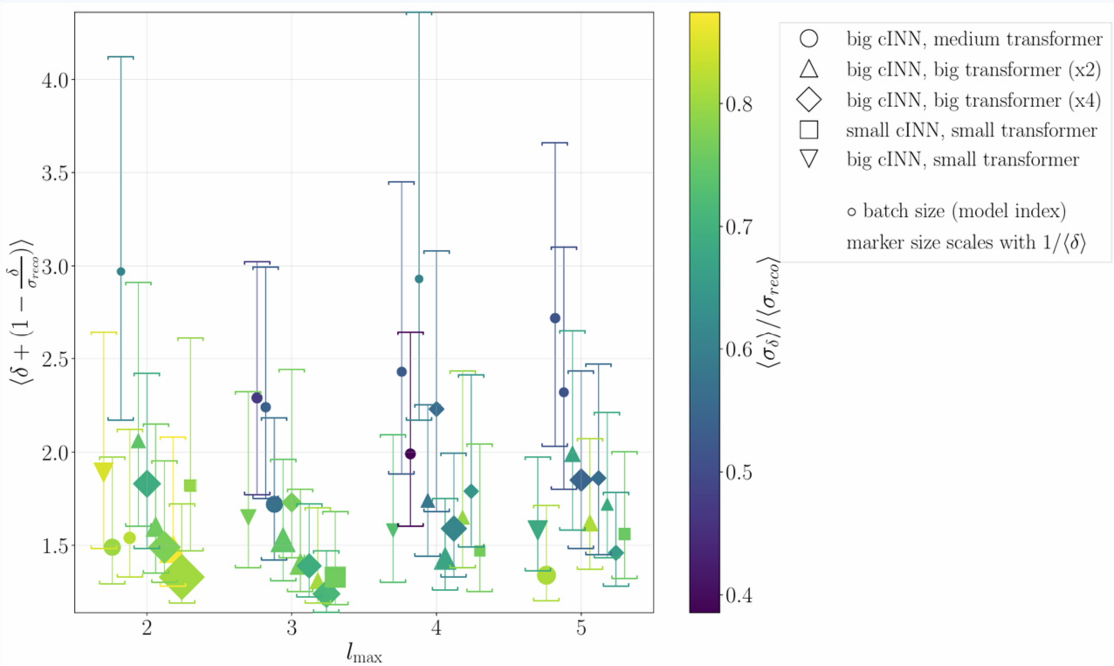
← omit constant

← change of variables

Results

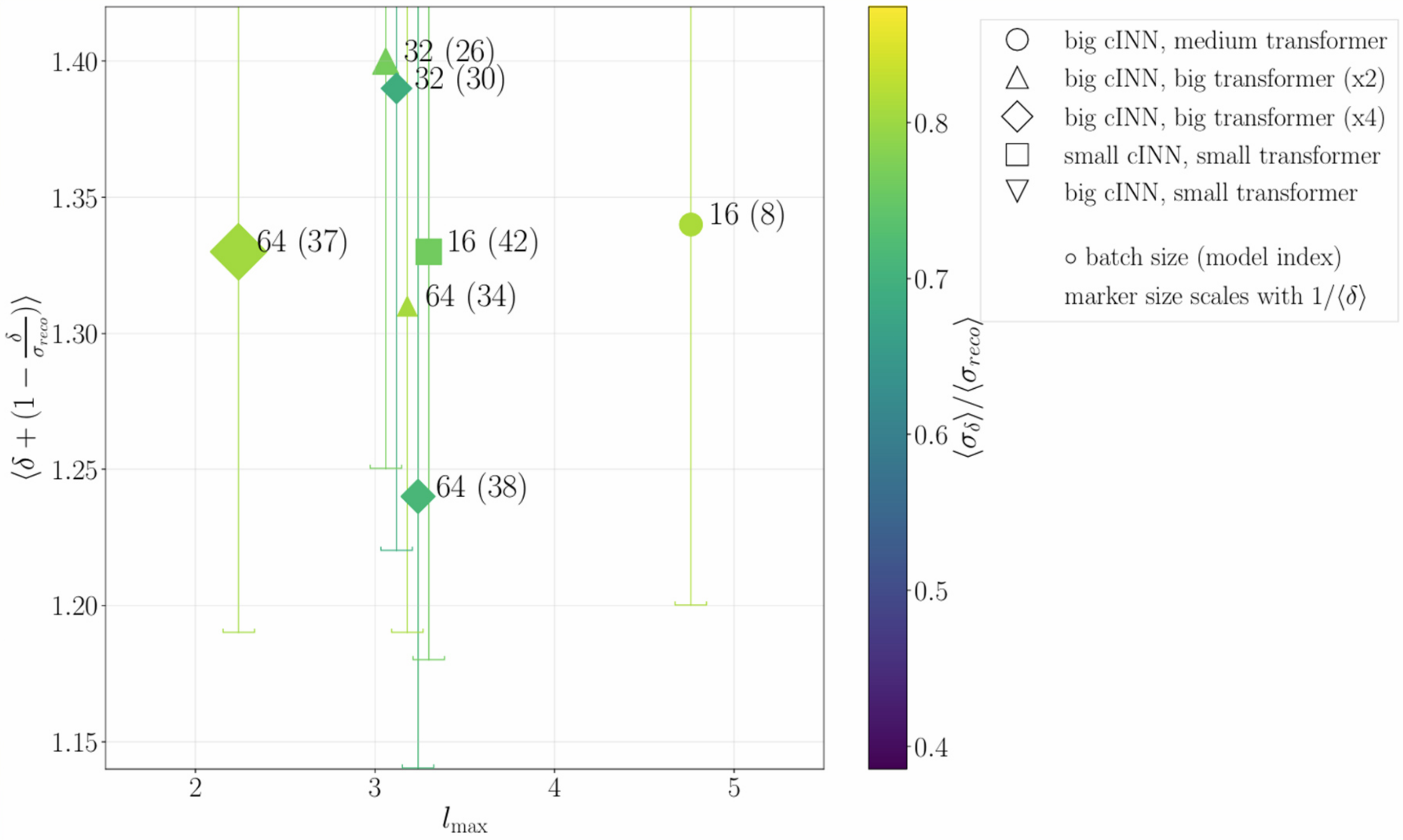


$\langle \delta + (1 - \frac{\delta}{\sigma_{reco}}) \rangle$
$l_{\max}$
$\langle \sigma_\delta \rangle / \langle \sigma_{reco} \rangle$
big cINN, medium transformer
big cINN, big transformer (x2)
big cINN, big transformer (x4)
small cINN, small transformer
big cINN, small transformer
o batch size (model index)
marker size scales with $1/\langle \delta \rangle$



$\langle \delta + (1 - \frac{\delta}{\sigma_{reco}}) \rangle$
$l_{\max}$
$\langle \sigma_\delta \rangle / \langle \sigma_{reco} \rangle$
big cINN, medium transformer
big cINN, big transformer (x2)
big cINN, big transformer (x4)
small cINN, small transformer
big cINN, small transformer
o batch size (model index)
marker size scales with $1/\langle \delta \rangle$

Results



1.40
1.35
1.30
1.25
1.20
1.15
$\langle \delta + (1 - \frac{\delta}{\sigma_{reco}}) \rangle$
2
3
4
5
$l_{\max}$
32 (26)
32 (30)
64 (37)
16 (42)
64 (34)
16 (8)
64 (38)
0.8
0.7
0.6
0.5
0.4
$\langle \sigma_\delta \rangle / \langle \sigma_{reco} \rangle$
big cINN, medium transformer
big cINN, big transformer (x2)
big cINN, big transformer (x4)
small cINN, small transformer
big cINN, small transformer
o batch size (model index)
marker size scales with $1/\langle \delta \rangle$

Loss function for cINNs

How to calculate Jacobian?

$$f_1(\theta) = \begin{cases} z_1 = \theta_1 \odot \exp(s_2(\theta_2)) + t_2(\theta_2) \\ \theta_2 = \theta_2 \ , \end{cases}$$

$$\begin{aligned}\det \frac{\partial f_1(\theta)}{\partial \theta} &= \det \begin{pmatrix} \frac{\partial z_1}{\partial \theta_1} & \frac{\partial z_1}{\partial \theta_2} \\ \frac{\partial \theta_2}{\partial \theta_1} & \frac{\partial \theta_2}{\partial \theta_2} \end{pmatrix} \\ &= \det \begin{pmatrix} \mathrm{diag}(\exp(s_2(\theta_2))) & \frac{\partial z_1}{\partial \theta_2} \\ 0 & \mathbb{I} \end{pmatrix} \\ &= \prod_j \exp(s_{2,j}(\theta_2))\end{aligned}$$

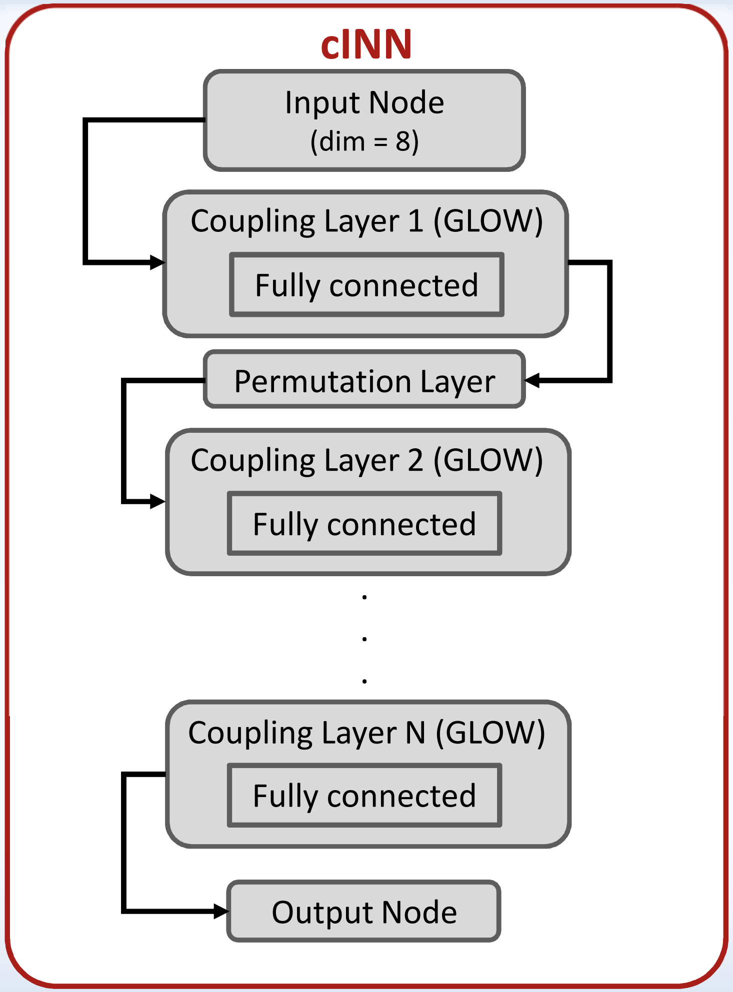
$$\begin{aligned}|\det\left(\frac{\partial z}{\partial \theta}\right)| &= |\frac{\partial f_1(\theta)}{\partial \theta} \frac{\partial f_2(\theta)}{\partial \theta}| \\ &= \prod_j \exp(s_{2,j}(\theta_2)) \cdot \exp(s_{1,j}(z_1)) \\ &= \exp\left(\sum_j s_{2,j}(\theta_2) + s_{1,j}(z_1)\right)\end{aligned}$$

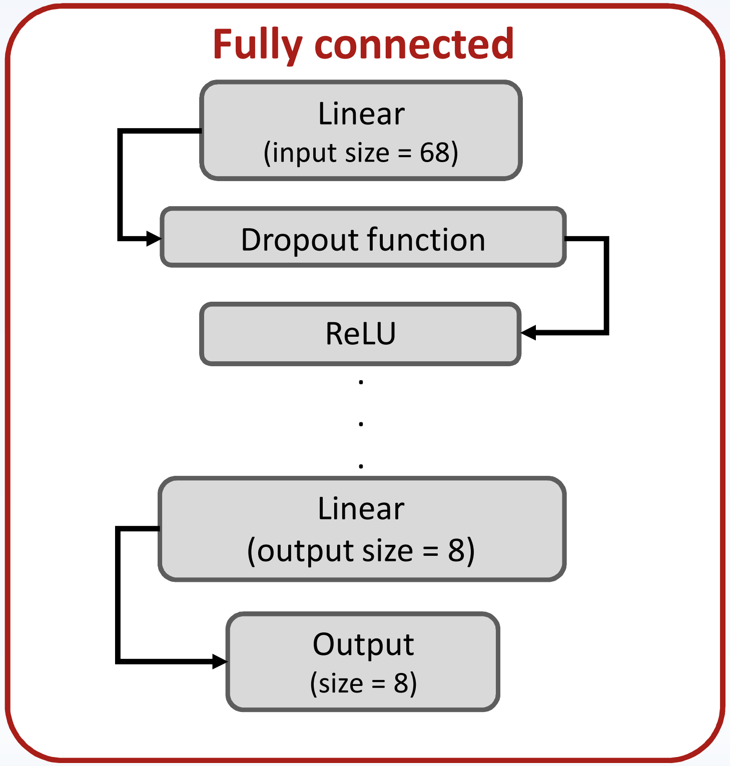
$$L = \mathbb{E}_{\theta \sim p(\theta|y)}\left(-\log\left(p(z)\right) - \log\left(|\det\left(\frac{\partial z}{\partial \theta}\right)|\right)\right)$$

can become negative → $L = \frac{1}{m} \sum_{i=1}^{m} \left(\frac{1}{2} \|f(\theta_i)\|^2 - \sum_{l=1}^{2} \sum_j s_{l,j}\right)$

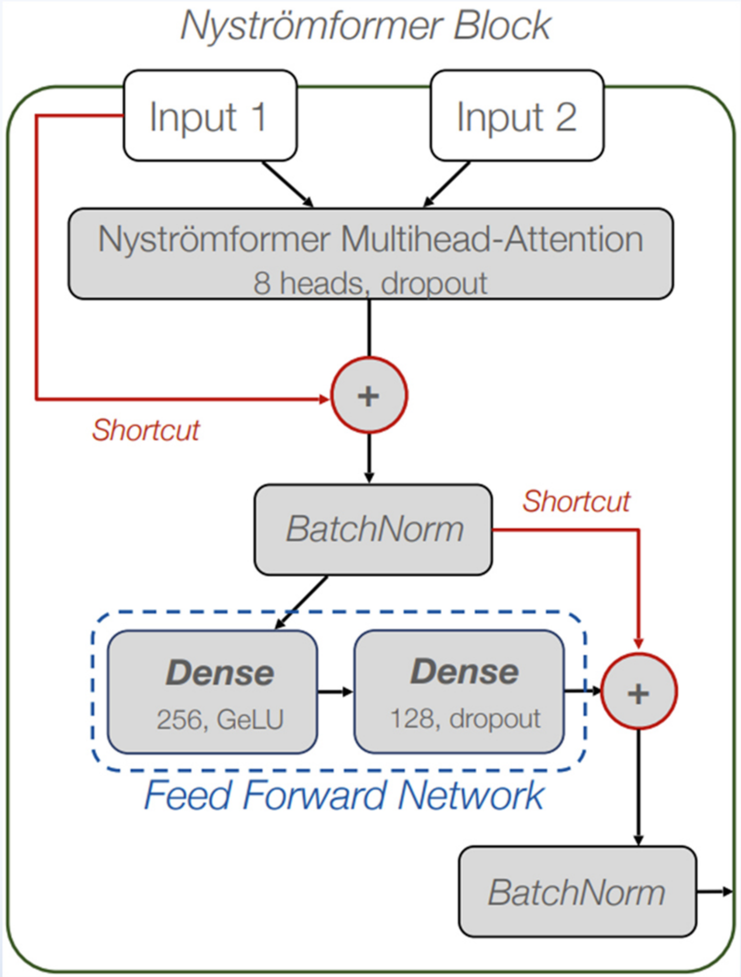
```
zz = torch.sum(z**2, dim=-1)  # si

neg_log_likeli = 0.5 * zz - jac  #
loss = torch.mean(neg_log_likeli)
loss.backward()
```

Network architecture - cINN





Transformer



cINN	training data	N_{blocks}	internal size	T_{dim}	batch size	N_{layers}	l_{max}	$\langle\delta\rangle$ /°	$\langle\sigma_\delta\rangle$ /°	$\langle\sigma_{\text{reco}}\rangle$ /°	fit	per_{16}	per_{84}
8	5870080	5	256	64	16	1	5	0.46 ± 0.40	1.34 ± 0.84	1.65 ± 0.77	1.34	-0.14	+0.37
34	2621440	5	128	128	64	2	3	0.62 ± 0.27	1.26 ± 0.41	1.56 ± 0.44	1.31	-0.12	+0.39
37	2621440	5	128	128	64	4	2	0.15 ± 0.09	1.32 ± 0.67	1.64 ± 0.62	1.33	-0.14	+0.39
38	2621440	5	128	128	64	4	3	0.41 ± 0.26	1.07 ± 0.56	1.50 ± 0.58	1.24	-0.10	+0.23

A measure for fitness of networks

$$\text{fit}_1 = \delta_1 + \left(1 - \frac{\delta_1}{\sigma_{reco,1}}\right)$$

.

.

.

.

.

$$\text{fit}_{1600} = \delta_{1600} + \left(1 - \frac{\delta_{1600}}{\sigma_{reco,1600}}\right)$$

$$\widetilde{\text{fit}}_{1\ldots1600}$$

Source position:

1. $\widetilde{\text{fit}}_{1,1\ldots1600}$
2. $\widetilde{\text{fit}}_{2,1\ldots1600}$
3. $\widetilde{\text{fit}}_{3,1\ldots1600}$
4. $\widetilde{\text{fit}}_{4,1\ldots1600}$

$$\langle\widetilde{\text{fit}}_{1\ldots4,1\ldots1600}\rangle$$

Normalizing flows

- Bijective functions
- Map between posteriors (input) and latents (output)
- Training: forward
- Evaluation: backward

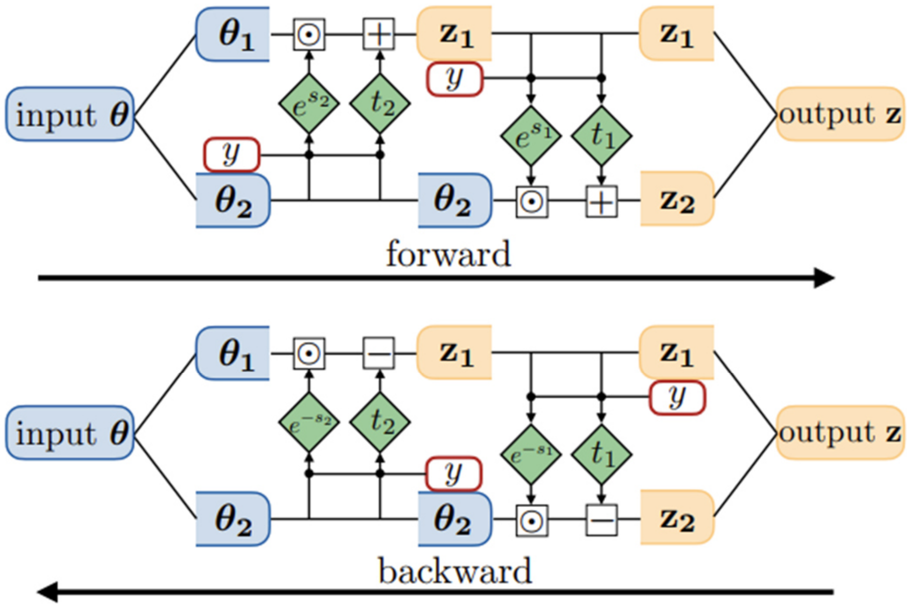
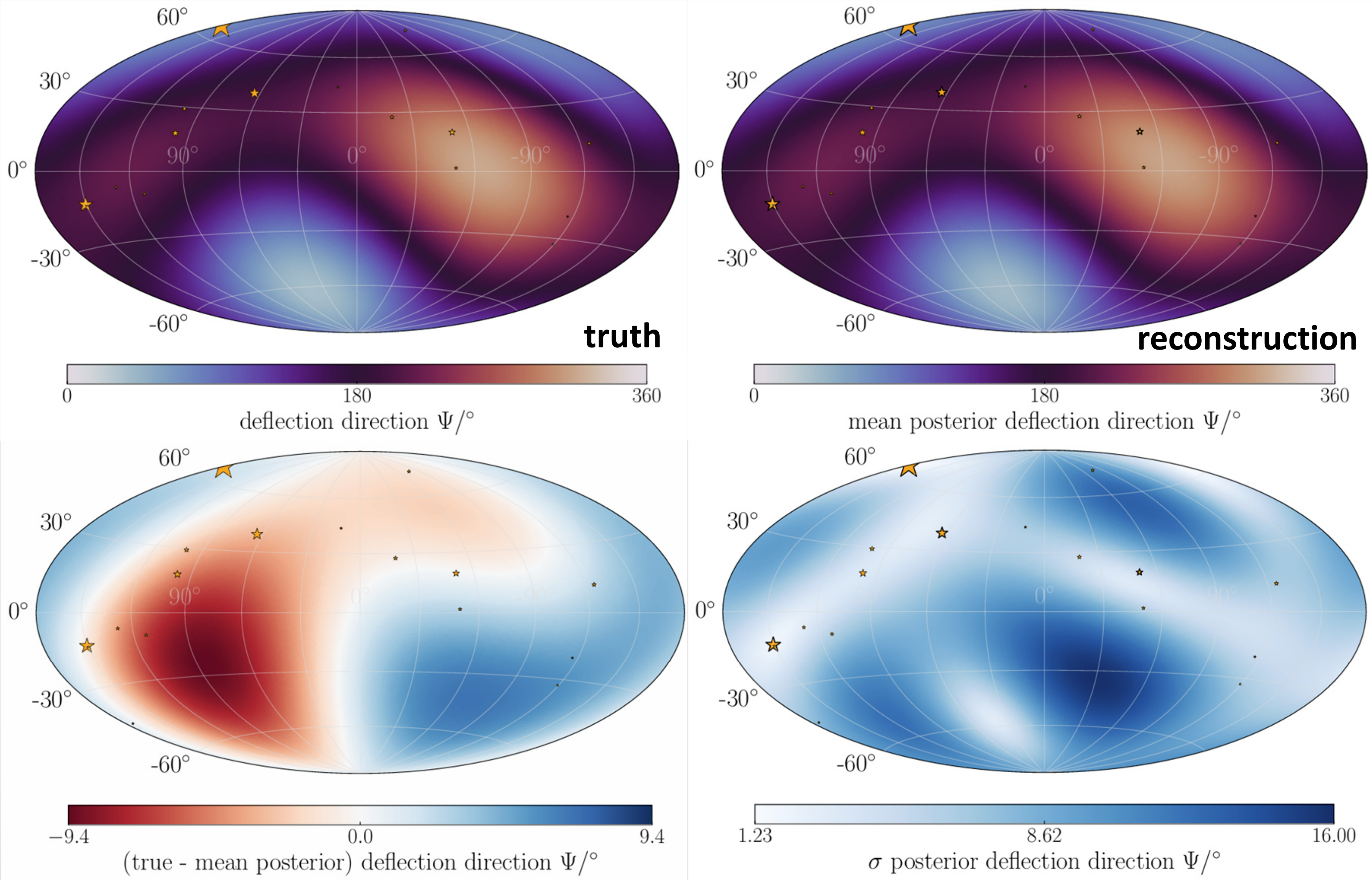


Fig. 4: Structure of the reversible block used for the conditional invertible neural network. It can be evaluated in two directions. The upper part shows the training mode or forward direction, the lower part displays the evaluation mode or backward direction.

Example reconstruction

- Reconstruction works fine
- Uncertainties are small in regions with sources



Sensitivity of the method

- Evaluating test data of size 1600 at 4 source positions
- Histogram values of $\delta = \Psi_{truth} - \Psi_{reco}$
- δ should be close to 0