Diffusion-Model SDME Extraction

A score-based posterior with a learned Rosenbluth

EMMI Rapid Reaction Task Force: Impact of Vector Mesons on the studies of the 3D structure of the nucleon

Bhawani(JLab) and Yaohang (ODU)

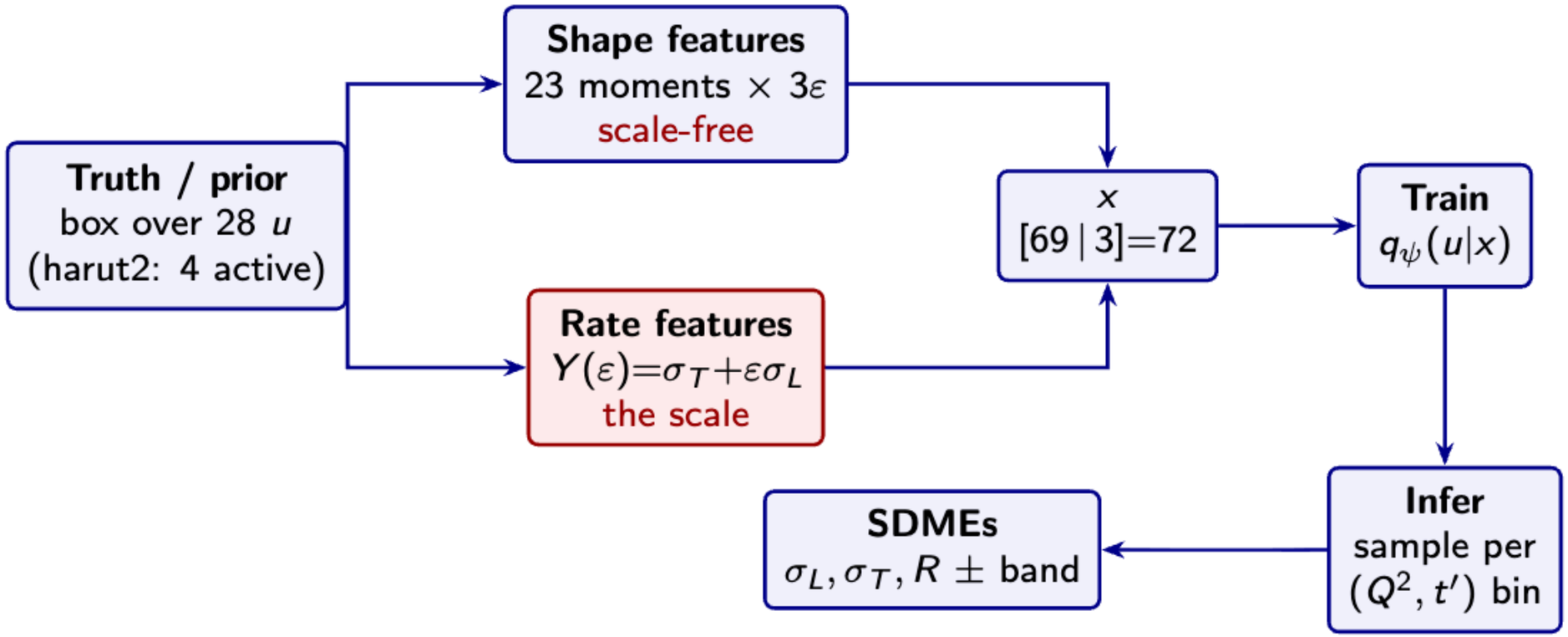
Thursday, June 18, 2026

* Y. Wang, **H. Avakian**, V.D. Burkert, **L. Elouadrhiri**, **D. Glazier**, **F.X. Girod**, R. Milner, I. Korover, V. Kubarovsky, and P. Moran

singh@jlab.org

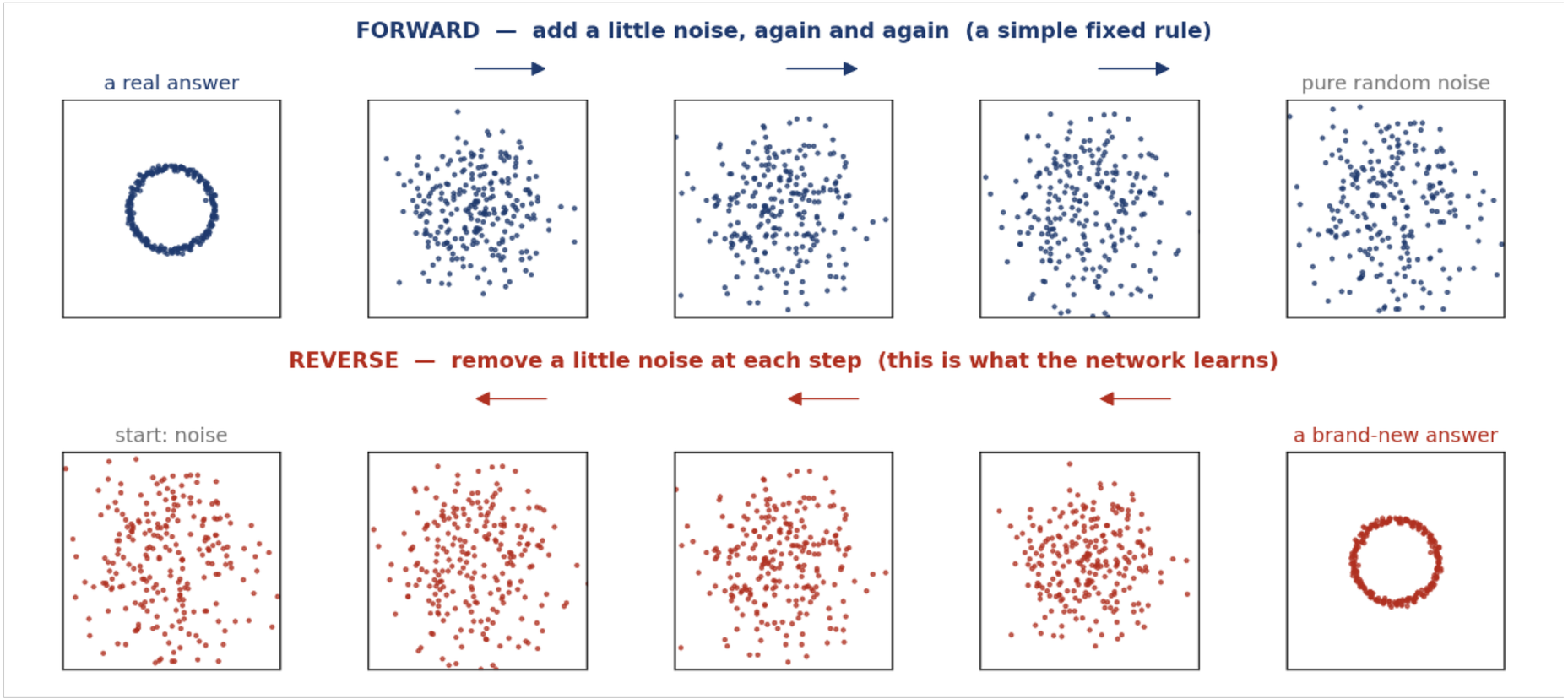
Work flow for extraction

- **A conditional diffusion model:** $q_\psi(u \mid x)$ — the third neural route, twin of the normalizing-flow NPE.
- Different engine: iterative denoising instead of one invertible map (no Jacobian constraint ⇒ flexible, correlated posteriors).



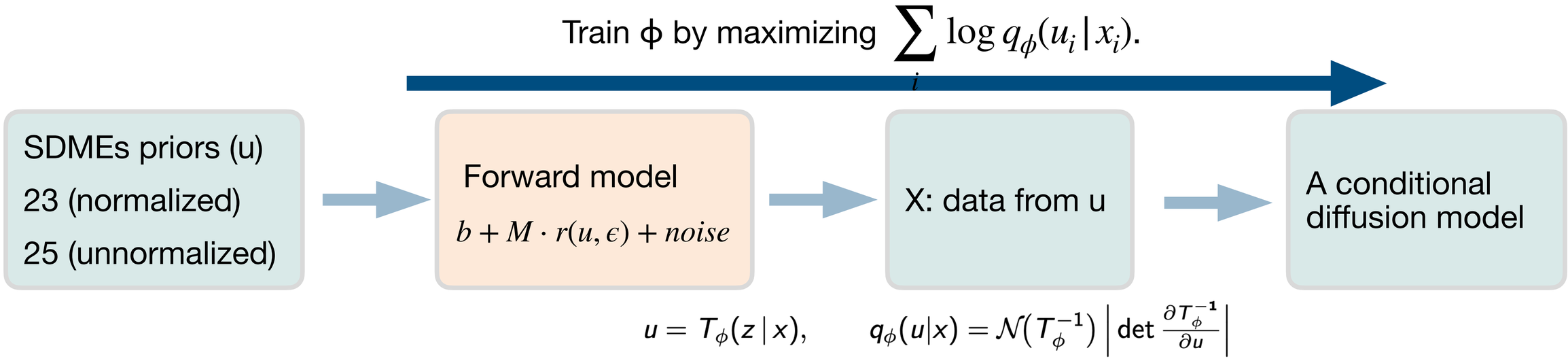
The problem and how diffusion model can help

- From those angles we want 23/25 numbers — the SDMEs — that describe the ϕ's spin state
- Going SDMEs → data is easy: we have the formula.
- Going data → SDMEs is the hard INVERSE problem! A diffusion model is a tool that learns this inverse — and gives honest error bars.



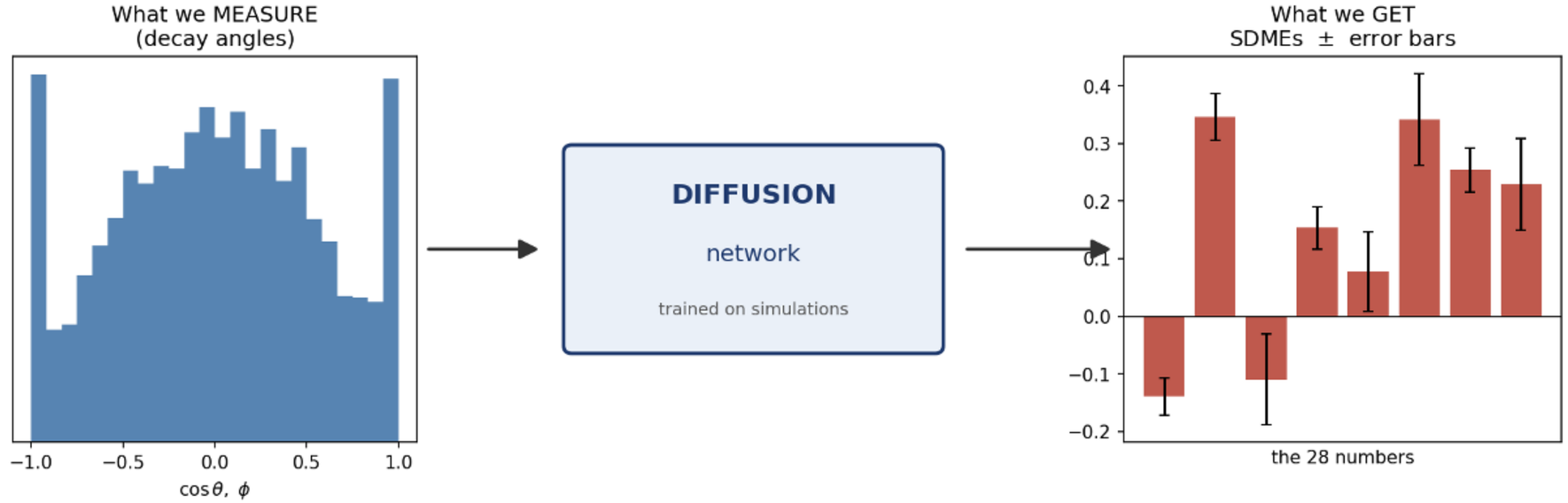
Forward Simulator

- For each random SDME set u's we compute the 23/25 moments ANALYTICALLY:
  - moments = b(ε) + M(ε) · r(u, ε) (a known linear formula).
  - r(u,ε) are the normalized SDME ratios; b, M are fixed pre-computed tables.
- We add realistic statistical scatter (the noise a for finite event sample): Gaussian, size ~1/√(#evts)



Inference stage!

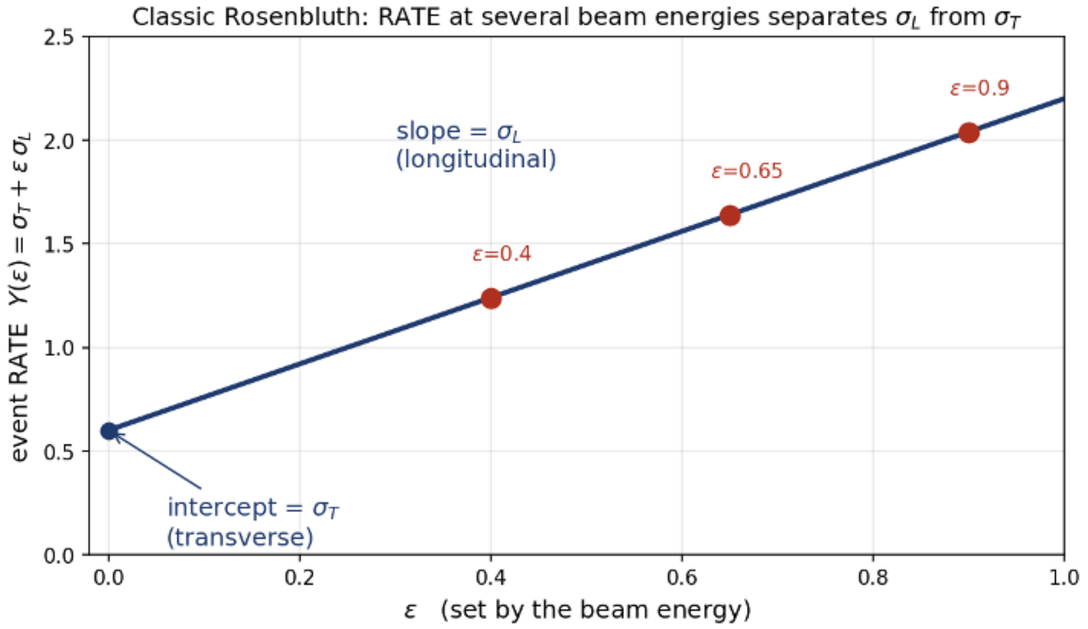
- Inference (Diffusion approach): It learns q(u | x): given the data x, the probability distribution of the SDMEs u that could have produced it
- ‘Distribution’, not a single answer → we get a best value AND an error bar
- It is a SAMPLER: we never need a formula for the probability, we only need to draw answers from it — which is exactly what a diffusion model does well
- Trained once, used on every kinematic bin



Run the network many times → many plausible answers → their spread IS the uncertainty.

Physics catch

- For Key fact: the angular pattern is ‘scale-free’. If you multiply ALL the SDMEs by the same factor, every moment stays identical
- So the moments fix only the SHAPE (ratios), never the absolute size of longitudinal (σ_L) vs transverse (σ_T)
- A shape-only network just guesses the size from the prior → it gets σ_L wrong (overshoots)
- The absolute size lives in the event RATE, not the angles. At each beam energy ε the reduced yield is Y(ε) = σ_T + ε·σ_L.

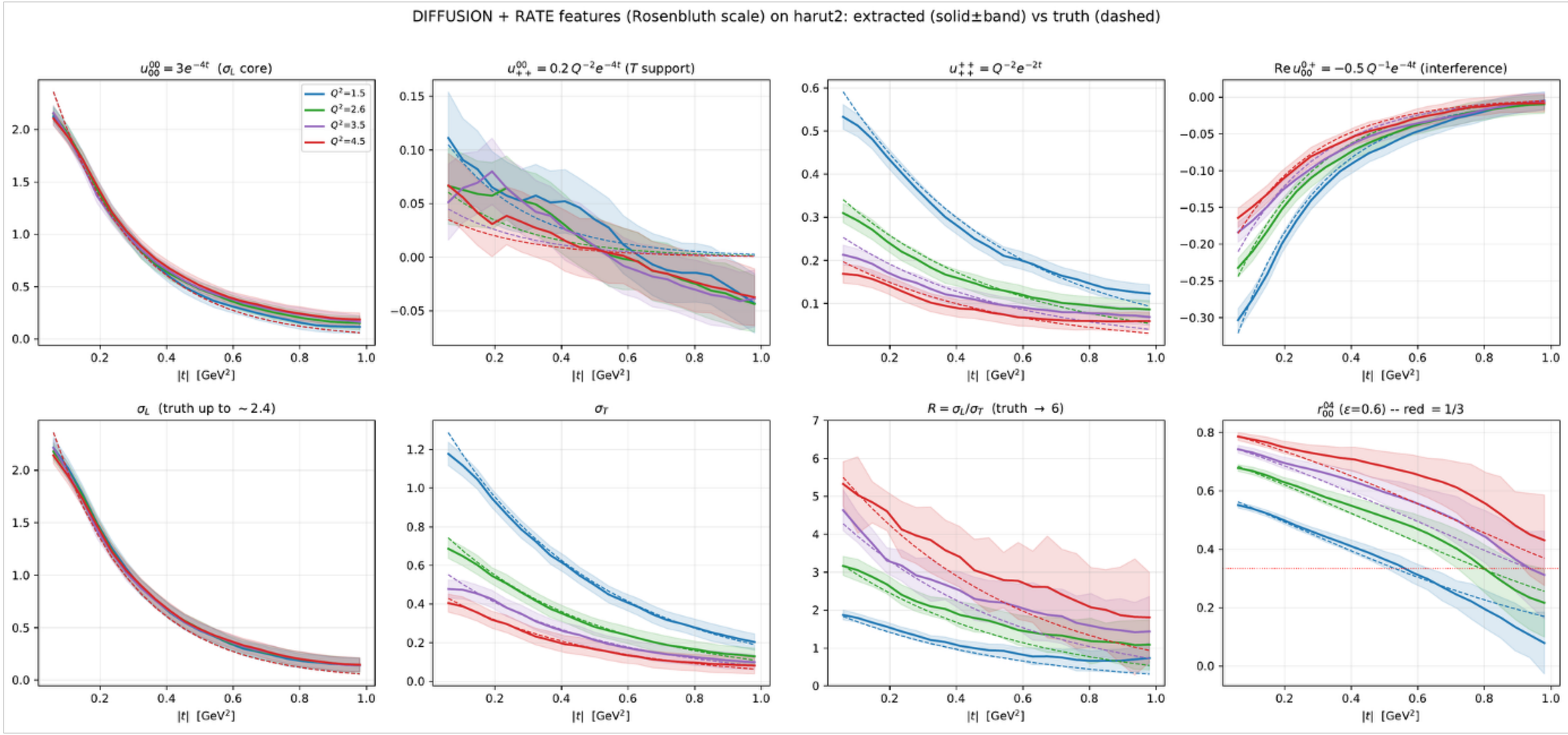


The stress test for model

- We invent a demanding answer where the SDMEs are smooth FUNCTIONS of the kinematics (Q^2, t) and the longitudinal piece is LARGE:
  - u00_00 = 3 e^(–4t), u11_pp = Q^{-2} e^(–2t),
  - Re u00_0p = –0.5 Q^{-1} e^(–4t), u00_pp = 0.2 Q^{-2} e^(–4t).
- This gives R = σ_L/σ_T up to ~6 — exactly the regime that defeats simple per-bin fits. Only 4 SDMEs are non-zero; the rest must come out ~0.
- We simulate its data, run the trained network per (Q^2, t) point, and compare to the known truth.

Results

- Solid = network (± band), dashed = truth, per Q^2 ring.
- All four active SDMEs recovered, incl. the interference Re u00_0p; σ_L RMS 0.08, σ_T RMS 0.02, R up to ~6

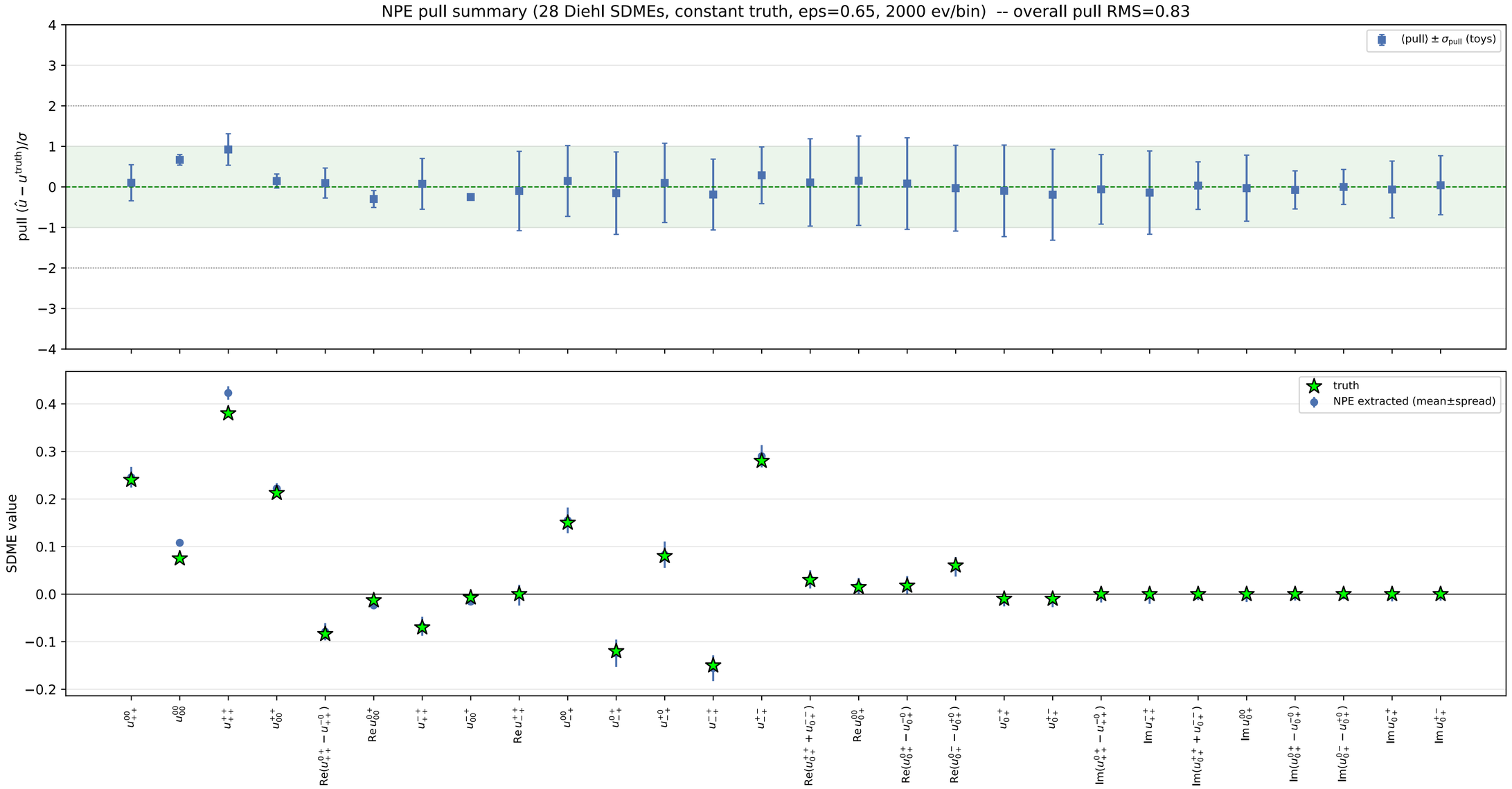


How we train it

- Show it an example: take a known u, pick a random step t, noise it to u_t, ask the net to predict the noise
- Loss = mean-squared error between predicted and true noise — one number saying how wrong it is
- ‘Adam’ nudges the weights downhill to reduce the loss; ‘learning rate’ (0.0002) is the step size
- ‘Batch’ = 1024 examples processed at once; ‘epoch’ = one pass through all 60,000; we do 400 epochs (~few minutes on a laptop CPU).
- Bookkeeping: u is rescaled to [–1,1]; x is standardized — so all numbers are comparable in size (helps training).

.

Extraction (sim only)



Event by event extractions

- Kinematics dependencies, event-by-event extraction, with stress functions given by Harut

$$u_{00}^{00} = 3\,e^{-4t}, \quad u_{++}^{++} = Q^{-2}e^{-2t}, \quad u_{0+}^{00} = -0.5\,Q^{-1}e^{-4t}, \quad u_{++}^{00} = 0.2\,Q^{-2}e^{-4t},$$

EVENT-LEVEL fit on harut2 truth (large sigma_L): extracted (solid) vs truth (dashed). Per-bin MCMC gave sigma_L 1.9 → 0.36; does the rate term hold it up?