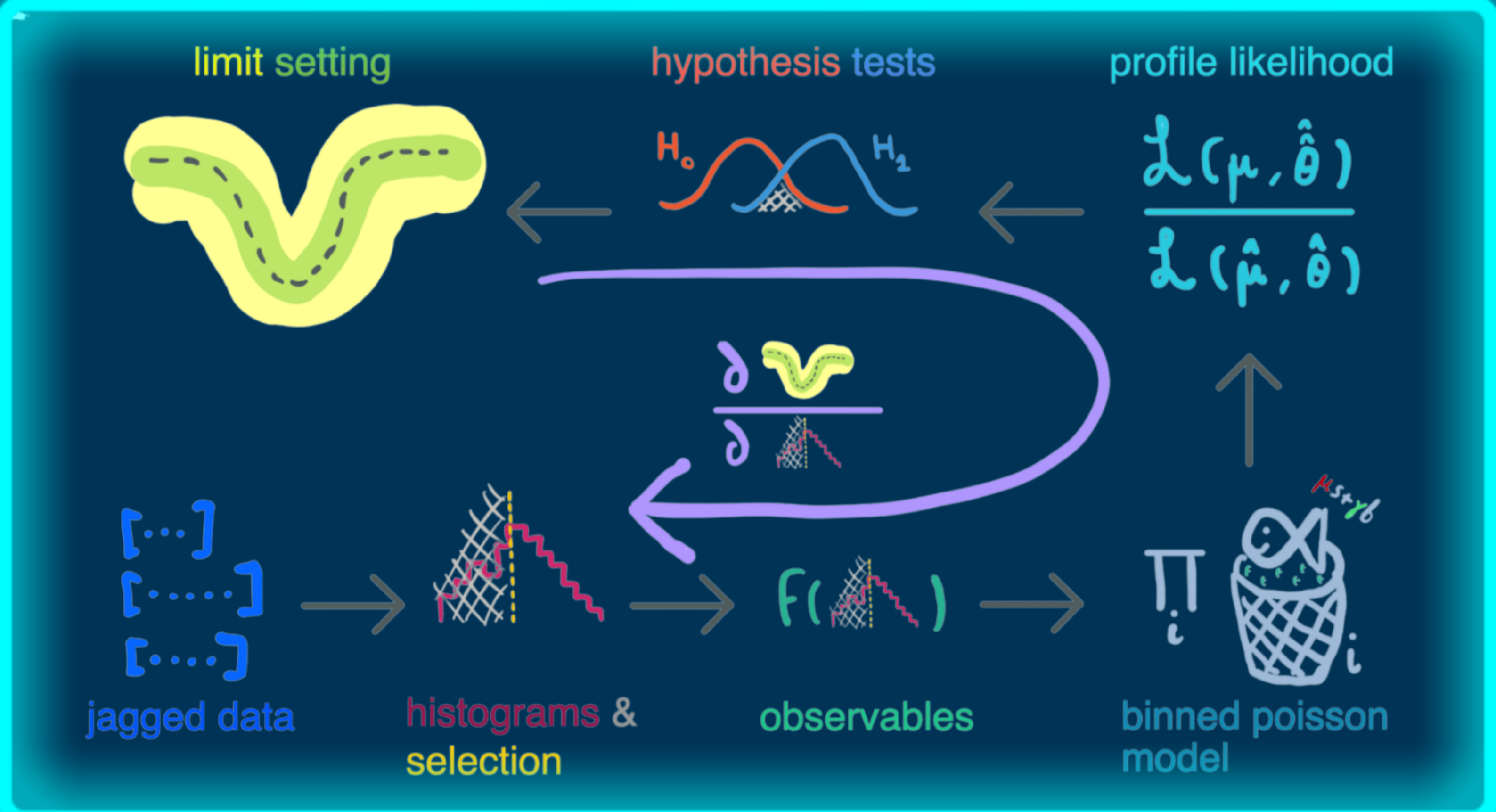
neos: neural end-to-end optimised statistics

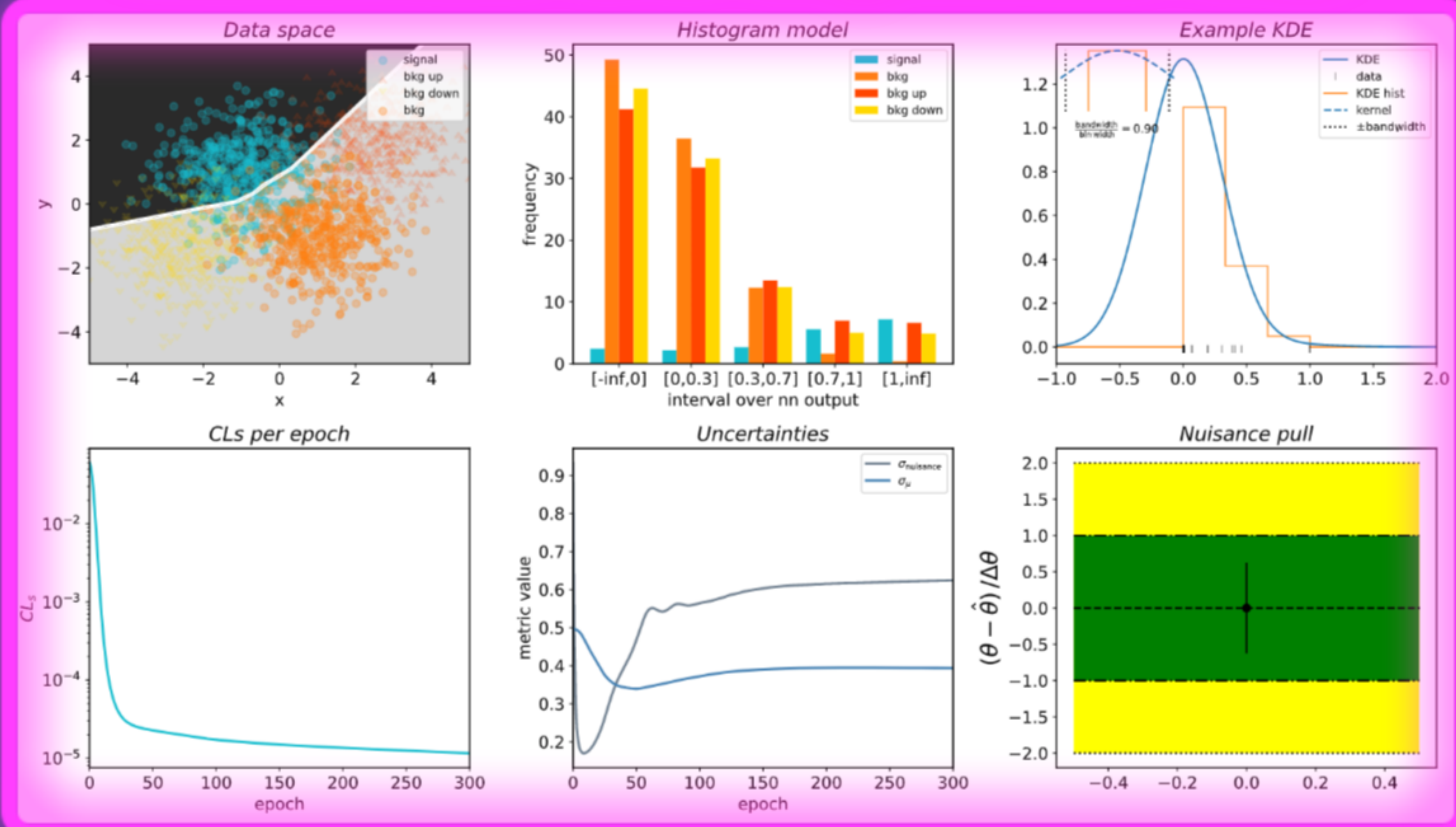
NATHAN SIMPSON LUKAS HEINRICH

WOW! *Machine learning!*

By making an analysis workflow that's fully differentiable...



...we train a NN observable to minimise the expected CL_s!



How? Automatic differentiation:

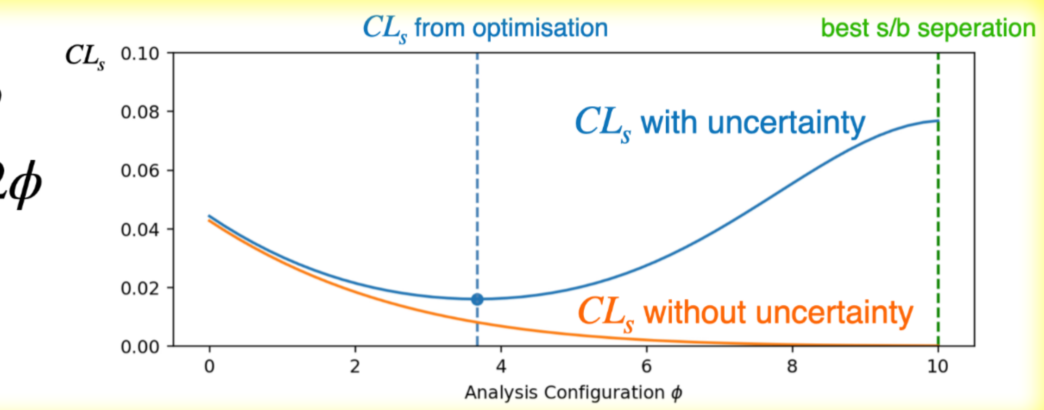
We train the neural network via gradient descent, i.e.:
$\text{weights}_{n+1} \approx \text{weights}_n - \nabla \text{loss}(\text{weights}_n) * \text{lr}$.
Crucially, this requires the calculation of $\nabla \text{loss}(\text{weights}_n)$: the loss ($CL_s$) must be *differentiable with respect to the weights.*

We achieved this by coding an analysis using automatic differentiation software (JAX), using a trick to differentiate through a fit, and by approximating a histogram differentiably.

Why? sig/bkg seperation may not be enough:

An illustrative one bin example with a single parameter ϕ: Optimising for s/b separation gives us the *best stat-only CL_s*, but the *worst CL_s when including background uncertainty*!

$$s = s + \phi$$
$$b = b - 2\phi$$
$$\sigma_b \propto \phi^2$$



WANT TO KNOW MORE? VISIT *GITHUB.COM/GRADHEP/NEOS*