We wish to thank all three referees for their encouraging and helpful reports. We think all the suggestions and comments are excellent, and plan to exploit the space available thanks to the additional 9th page to expand and improve the paper to address them as described below.

**Reproducibility and code release (reviewers 1 & 2):** We will significantly expand the description of the method to make it easier to reproduce. To ensure reproducibility, we commit to publicly releasing our code on GitHub and via the Python Package Index PyPI (so that it can be pip-installed) on acceptance.

**Clarification of relation to prior work (reviewers 1, 2 & 3):** We will expand the description of how our method differs from and improves upon prior work (aside from deploying normalizing flows flows to enable regression of probability distributions from samples). Although the original AI Feynman paper used a neural network approximation of the unknown function to discover simplifying function properties, it did so in an unprincipled and ad hoc way that we replace by a general, principled and much more effective method:

1. Instead of discovering merely two types of graph modularity (symmetry and separability) involving merely four particular bivariate functions ($+$, $-$, $\times$ and $\div$), our method has the potential to discover *any* graph modularity involving *any* functions of $n = 2, 3, ...$ variables. A key contribution is showing how this can be efficiently done by examining gradients of the neural network fit.

2. Instead of concluding that a candidate function or graph decomposition is good because the fitting accuracy exceeds arbitrary hyperparameter-determined thresholds, we eliminate these hyperparameters and use a Patero-frontier (of description-length complexity versus accuracy) to prune our search over candidate expressions by discarding all candidates not on the frontier, improving robustness to noise by 1-3 orders of magnitude.

3. Instead of simply rejecting formula candidates using $L_\infty$-norm (rejecting as soon as the error for a single data point crosses a threshold), we reject using statistical hypothesis testing, improving robustness to noise.

**Expanded discussion of NN architecture, training, overfitting (reviewers 1, 2 & 3):**

We will significantly expand the description of our neural network architecture and training, enabled by the expanded page limit. We will also add an extensive discussion of how overfitting impacts our method — here are the highlights:

1. To minimize neural network overfitting, we early-stop training when the validation loss starts increasing.

2. We avoid using our neural network (to guess symbolic functions) outside the domain where it was trained.

3. Overfitting noise would generally *reduce* apparent graph modularity, thus causing failure to discover formulas rather than discovery of spurious "overfit" formulas.

4. A key desirable feature of the minimum-description-length formalism (the information-theoretical inspiration for our method) is that it provably avoids overfitting as shown in refs [28,30].

**Expanded discussion of limitations, failure cases and areas for improvement (reviewer 1):** Text will be added clarifying that there is great room for further improvement, both to symbolic regression itself and to the problem of identifying a low-dimensional latent space from high-dimensional data whose coordinates are promising candidates for symbolic regression. To help readers identify such opportunities for future work, text will be added providing more details on failure modes of our method.

**Algorithm name (reviewer 1):** Clarification: The name "AI Feynman" originates from the benchmark dataset upon which our algorithm was originally trained (100 famous/complex equations from the Feynman Lectures on Physics trilogy).

**Quick fixes (reviewers 2 & 3):**

1. We will expand the labels in Figure 1.

2. We will improve the confusing notation for the action of generalized symmetry (Table 1).

3. We will fix the legend and the "??" in Figure 1 in the appendix.