We thank the reviewers for their feedback, which will help improve writing clarity and notation consistency.

[Reveiwer #1] **Algorithm Descriptions:** As suggested, we will replace the numerical example with an extended discussion of recursive ExactLine application that improves upon Supplemental Section 3.

Extending lines beyond the adversarial example in Figure 4: We have found some images where the classification changes multiple times close to the adversarial example indicating that the adversarial example lies in a "peninsula". We can quantify the size of such peninsulas using ExactLine, which we can include in the paper.

[Reviewer #2] **Impact of exact IG on saliency maps:** Saliency maps produced using exact IG are noticeably more accurate compared to those generated from approximate IG. We will include these in the paper and code release.

Regarding limited applications of proposed method: The paper already demonstrates the varied types of applications for ExactLine, ranging from the first method for exact computation of IG to an empirical falsification of the Linear Explanation for Adversarial Examples. Furthermore, using exact IG computed using ExactLine as the baseline, better *non-uniform* sampling methods for approximate IG could be devised.

Using ExactLine to detect adversarial images: Initial experiments show that densities (as defined on line 232) computed using random ExactLine "probes" around natural images differ significantly from those around adversarial images. Thus, ExactLine could use such densities to distinguish between natural and adversarial images.

[Reviewer #4] **Theoretical significance of ExactLine algorithms:** The "most straightforward way" to approach the algorithm would be that of [34], which requires exponential time to enumerate all possible orthants. A key observation underlying our work is that we get a significantly faster algorithm (worst-case polynomial time for a fixed number of layers) by restricting the input to be one dimensional (a line). This insight opens a new direction of research for network-analysis tools: approaches in [34,12] are precise but exponential time, approaches such as [39] are overapproximations but polynomial time; in contrast, we show that restricting the input dimensionality allows precise *and* efficient algorithms. Furthermore, we believe that the efficient handling of MaxPool and MaxPool+ReLU is a substantial theoretical contribution, which we will include in the main paper in lieu of the example in Section 2.

Usefulness to practitioners: After presenting our quantitative results to the authors of the IG paper, they have agreed to switch to trapezoidal sampling in their implementation. Our falsification of the linearity hypothesis directly contributes to the community's knowledge, and our initial observation about the relative-linearity of robust networks opens up future work to developing a better understanding of adversarial examples. Furthermore, to enable practitioners to use our ExactLine primitive, we provide a well-documented and tested, open-source library for computing ExactLine in the form of a gRPC C++ server parallelized with the Intel TBB library with Python frontend. Our optimized implementation of ExactLine can handle large ImageNet-scale networks without running out of memory.

What is $\overleftrightarrow{QR}$ in Theorem 1? The *unbounded* line incident to Q and R (which line *segment* $\overline{QR}$ lies in).

Usefulness of visualizing ACAS Xu: Visualizing their decision boundaries is a common way to understand networks in practice. Prior approaches would use sampling when visualizing; e.g., Figure 7 in [12], and Figures 2, 8, 10, 12–15 in [5]. Use of ExactLine avoids the use of sampling along one dimension. Probing the behavior of networks could be used to synthesize candidate whole-network specifications, which can then be verified by tools like [12,39].

Regarding consistency of number of IG samples within a network: Line 194 should actually read "the *density* of samples needed is relatively consistent within each network"; i.e., number of samples divided by distance between image and baseline. We will correct this sentence along with Table 2 in the final paper; we thank the reviewer for bringing this to our attention. In fact, the variance of the number of samples is high, while the variance for density is low. Standard deviation of the number of samples, eg., convsmall-left in Table 2 was approximately 77.6 vs. convsmall-trap having 46.97. This high variance is primarily due to some images being further from the baseline than others and some skew in the distribution of the densities. However, the quartiles for *density* for convsmall-left are 3.23/4.36/5.56 while the corresponding values for convsmall-trap are 2.26/2.82/3.94. We will update Table 2 with such statistics. Note that the suggestion on lines 195–198 can likewise be updated; viz., we can compute high-percentile density using a set of training inputs, and use this density to compute the number of samples required for a given test image.

Aren't trapezoidal approximations always better? This is not true in general, and surprisingly not true in practice either. Consider $\int_0^1 f(x)dx$ for $f(x) = 0$ when $x \in [0, 0.99]$ and $f(x) = 1$ otherwise. With two samples, the left-approximation is 0, while the trapezoidal-approximation is 0.5. The true integral (0.01) is closer to the left-approximation than the trapezoidal. In fact, for all models reported in Table 2, there were images for which trapezoidal sampling needed *more* samples than left sampling to get below 5% error, indicating that left-approximation was sometimes better than trapezoidal.

Explanation of Figure 4: Counting the number of vertical lines in the top and second-to-bottom lines of Figure 4 shows how the density of the FGSM is higher (more non-linear) than that of the random direction for the same network (normal training), falsifying the fundamental assumption behind the linear explanation of adversarial examples. Comparing the number of vertical lines on all of the black (normally-trained) lines with that of their green (DiffAI-trained) counterparts, we can see that the DiffAI model is significantly less-dense than the normal one. We will separate Figure 4 into two figures and explain each individually.