We thank the reviewers for their very helpful comments and suggestions. We will first address the primary concerns about real-world experiments and deep density models and then answer detailed comments and questions.

More experiments on real world data (R1). We present more experiments on both the UCI energy dataset and a UCI gas sensor dataset. Given the complexities of data with strong time dependencies, we walk through a few new real data experiments using a time-dependent bootstrap and a shuffled time axis (see Table A). First, we provide results for our unknown sensor experiment (the submission only contained results for a fixed single sensor) via the original bootstrap (denoted "MB-SM, Unshuffled"). The 100% recall but very poor precision (i.e., it always predicts a shift) is expected because the real data exhibits strong time dependencies, and thus *natural* (or "benign") distribution shifts exist across time. This highlights the inherent difficulty of detecting *adversarial* shifts in the presence of natural shifts: it is difficult (if not impossible in certain cases) to determine if a shift is benign or adversarial without further assumptions. As a first attempt to overcome this issue, we modify our bootstrapping method to account for natural shifts; specifically, we sample chunks of time jointly together (rather than completely random time points) from a held-out clean dataset. This causes our detection threshold to be much higher, and thus our time-dependent bootstrap method ("Time-Boot") rarely detects (very low recall) the adversarial shifts because they are hidden among natural shifts. Fundamentally, this is because our current density models assume no time dependency. One possible solution is to estimate time-dependent density models (e.g., autoregressive time models); however, exploration of time-dependent density models is outside the scope of this paper which focuses on the localization aspect. Thus, as a final experiment, we shuffle the dataset along the time axis so that all time dependencies are broken; this creates a semi-synthetic dataset because it retains the feature dependencies but removes time dependencies. In this semi-synthetic setting (denoted "Shuffled"), our method performs much better and similar to the simulation results. We hope these results and discussion bring insight into the challenges of time series data and encourage further work in this area.

Experiments with deep density models (R1, R2, R3, R4). In this experiment, we demonstrate that using a deep density model can improve the performance of our method. We fit a normalizing flow using iterative Gaussianization[1] which is fast and stable because it only requires iteratively estimating a PCA projections and univariate histograms and thus can be carefully controlled to help avoid overfitting. While recall is slightly reduced, our deep density method ("Deep-SM") significantly improves the precision of both detection and localization compared to the Gaussian-based method ("MB-SM") even when the time axis is unshuffled. Clearly, other deep density models including more general normalizing flows or autoregressive models could be used but we leave extensive comparisons to future work.

Table A: Unknown single sensor experiment with UCI appliance energy dataset (left) and UCI gas sensor dataset (right).

Time Axis	MB-SM		MB-SM-Time-Boot		Deep-SM-Time-Boot		MB-SM		MB-SM-Time-Boot		Deep-SM-Time-Boot	
	Prec	Recall	Prec	Recall	Prec	Recall	Prec	Recall	Prec	Recall	Prec	Recall
Feature shift detection (1st stage)												
Unshuffled	50.00%	**100%**	16.67%	8.86%	**55.00%**	62.62%	**50.00%**	**100%**	11.11%	2.27%	22.22%	4.55%
Shuffled	74.57%	**97.74%**	75.25%	96.20%	**77.89%**	93.67%	97.30%	**100%**	75.86%	**100%**	**97.78%**	**100%**
Feature shift localization (2nd stage)												
Unshuffled	1.92%	**100%**	2.38%	1.27%	**3.00%**	3.80%	8.85%	**100%**	**11.11%**	2.27%	**11.11%**	2.27%
Shuffled	2.87%	**97.74%**	**75.25%**	96.20%	64.21%	77.22%	4.52%	**100%**	56.90%	75.00%	**68.90%**	70.45%

R1. *Table 1 clarifications.* There was a typo and the first column should match that of the Table 2 which represents the difficulty of the attack based on mutual information between variables, see L232 - L245 on "Attack Strength and Difficulty". Marginal-KS is very bad because the attack model is very strong, i.e., it mimics the marginal distribution of the sensor. Thus, marginal KS will naturally fail—highlighting the limitation of prior work for this adversarial attack.

R2. *For bootstrapping, does the model need to be fit multiple times?* Yes, we refit the model for every bootstrap iteration. For Gaussian, this is fairly simple. For deep density models, we could train one model on all the data first, and then update the model slightly (1-2 epochs) for each bootstrap (similar to transfer learning). *How does bootstrapping perform at controlling the False Discovery Rate?* For the detection stage, the FDR was controlled below 0.05 in all but the hardest cases, see Table 1 (note that FDR is 1 - Precision). See also Table 6 and 7 in appendix. For the more challenging localization stage, we do not explicitly control the FDR.

R3. *Deep density with limited samples.* Thanks for the comment. Our results above demonstrate that deep models can indeed be helpful though we will add some discussion regarding this challenge. *Choice of window size.* Thanks for the comment. We highlight that the choice of window size is a trade-off between the delay in detecting a shift (Table 4) and the error of the sensor localization (Table 3). Also, the particular application may have resource constraints.

R4. *Motivation for KNN approach.* We wanted a method that could compute a "conditional" KS statistic since the marginal KS statistic is well-known for detecting 1D shifts. We are unsure what is meant by "K-D trees"; could you point to a reference for this? *Neural-kernelized and other conditional density estimation.* Thanks for the pointers. We note that using conditional density estimation would require estimating a different conditional density models for every feature. In contrast, a *single* joint density model can be used to compute all conditional statistics via the score function.

[1] [1] V. Laparra, G. Camps-Valls, and J. Malo. Iterative Gaussianization: From ICA to random rotations. *IEEE Tran. on Neural Networks*, 2011. [2] D. I. Inouye and P. Ravikumar. Deep density destructors. In *ICML*, 2018.