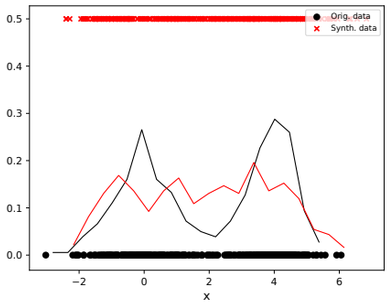
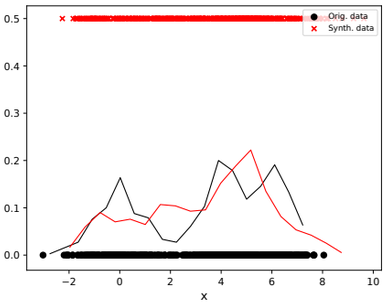
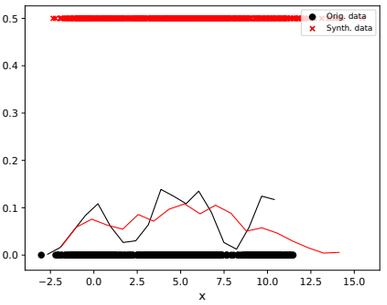
1 Simple GAN vs. Unsupervised Conditional GAN



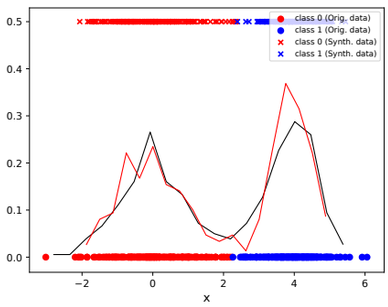
(a) Simple GAN (2 modes)



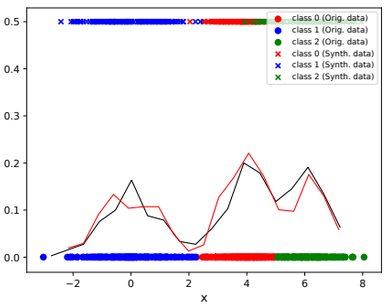
(b) Simple GAN (3 modes)



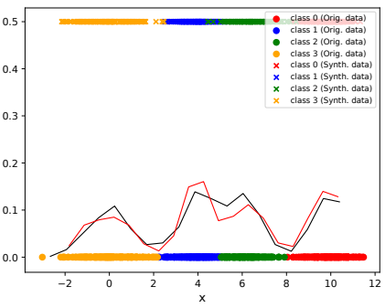
(c) Simple GAN (4 modes)



(d) Conditional GAN (2 modes)



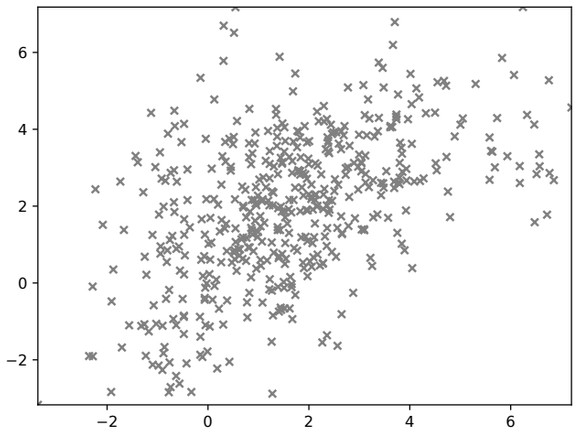
(e) Conditional GAN (3 modes)



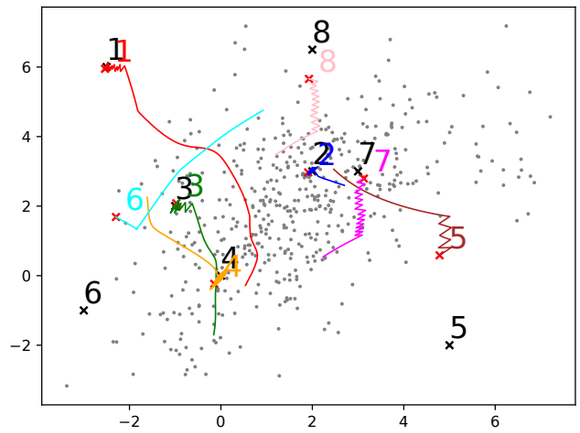
(f) Conditional GAN (4 modes)

Comparison between simple GAN and **unsupervised** Conditional GAN (cGAN) for $1D$ data with multiple modes. The samples at $y = 0$ are the real samples on which the GAN was trained. The samples at $y = 0.5$ are synthetic samples generated by the GANs after training for 1000 epochs. The *black* lines show the histogram of the original data. The *red* lines show the histogram of the synthetic samples generated by the GAN. For the unsupervised cGAN, we first perform *unsupervised* clustering with Gaussian Mixture Models. The number of components were selected with *BIC*. The cluster labels are then assigned as the class labels. For the 1D data shown here, cluster information seems to help.

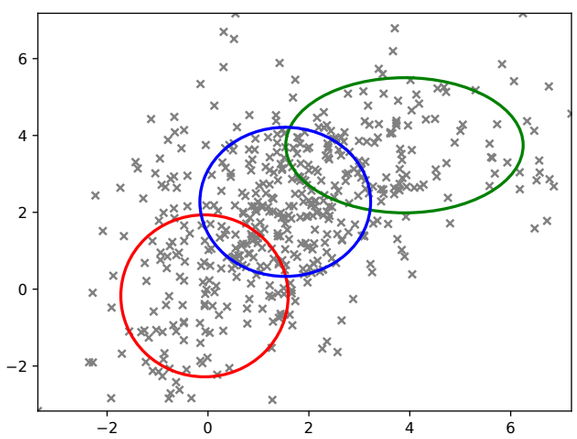
2 AnoGAN



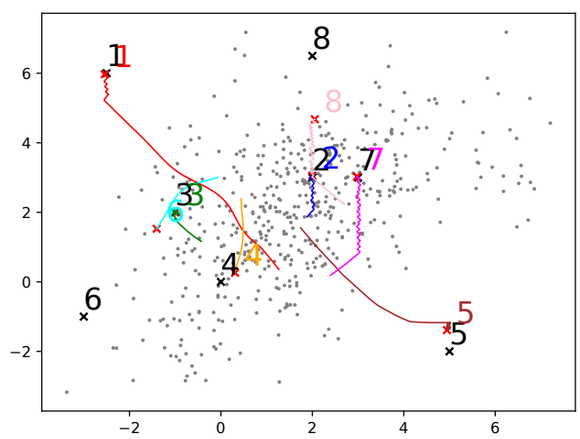
(a) Toy data samples



(b) *AnoGAN* with simple GAN



(c) Clusters for cGAN



(d) *AnoGAN* with cGAN

Illustration of **AnoGAN**. **(a)** shows the $2D$ *Toy* data. All GANs were trained for 2000 epochs. **(b)** shows the AnoGAN results with a simple GAN. **(c)** shows the unsupervised clustering of the *Toy* dataset. The number of components in the Gaussian Mixture Model was found to be 3 using the *BIC* model selection criteria. **(d)** shows the AnoGAN results with a cGAN where the labels were assigned according to the clusters shown in **(c)**. In **(b)** and **(d)** the points marked in *black* are test data points which were not available during training. Each test point has a corresponding colored point (marked with a *red* '$\times$') that represents its synthetic image reconstructed by AnoGAN. In simplistic terms, the distance between a test point and its reconstructed image can be considered a measure of anomalousness. **(b)** and **(d)** show that the points in the dense regions (e.g., $2, 3, 4, 7$) generally have low reconstruction loss. However, points in low-density regions (e.g., $1, 5, 6, 8$) give mixed results: while the reconstruction loss for 1 is low, it is high for $5, 6, 8$. **(b)** and **(d)** also show the trace of reconstruction for each data point. AnoGAN computes the reconstructed image iteratively. For example, with test point 5, the initial random latent representation z_0 corresponds to a point in the middle. After optimizing the AnoGAN loss, a new latent representation z_1 is found which corresponds to a point slightly closer to 5. Then, using z_1 and optimizing, the next latent representation z_2 is found. Over the several iterations, the reconstructed image for 5 moved along the brown line until it finally reached the point marked with $\times$. The traces are useful for debug: we can see that all reconstructed points move closer to the target test points irrespective of where they start. It is hard to see whether a cGAN performs better than a simple GAN in this case. Moreover, the AnoGAN loss should be averaged over multiple runs for each test point because there is stochasticity in the initial latent representation.