Using Wasserstein generative adversarial networks for the design of Monte Carlo simulations

by Susan Athey, Guido W. Imbens, Jonas Metzger, Evan Munro (2021)

Vsevolod Iakovlev

Economics Reading Group
Heriot-Watt University

November 17, 2023

Motivation I

- It is common to conduct **Monte Carlo** studies to assess the performance of estimators
- Artificial data is generated using a **distribution specified by the researcher**
- Selected distributions often have a high degree of **smoothness and limited dependence** between the variables
- It can be argued that the performance of an estimator in a Monte Carlo simulation is **not representative** of the performance in the real world

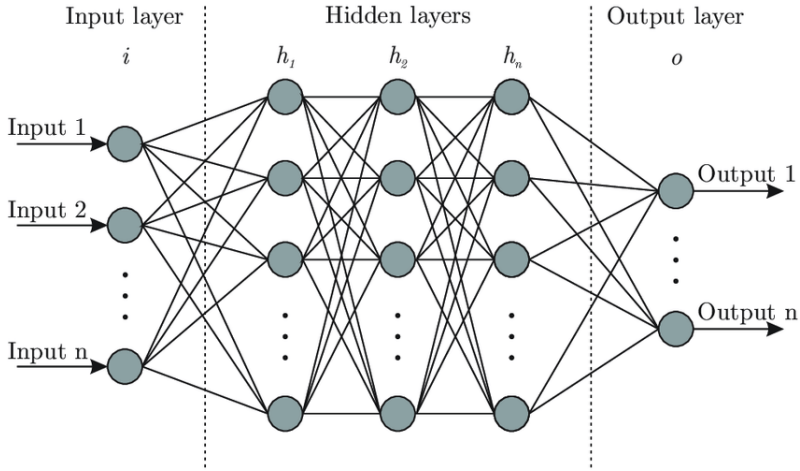
Motivation II

- Applied researchers have access to a **wide range of estimators** that could be used on a specific dataset
- It is often **unclear which estimator should be preferred** and it may not be feasible to analyse the results for all of them
- One way to deal with it is **stacked generalisation** (Wolpert, 1992)
  Shameless self-promotion: *"An introduction to stacking regression for economists"* (Ahrens et al., 2022)
- Alternatively, the **performance of potential estimators can be compared** using an artificial dataset featuring similar characteristics as the real dataset at hand

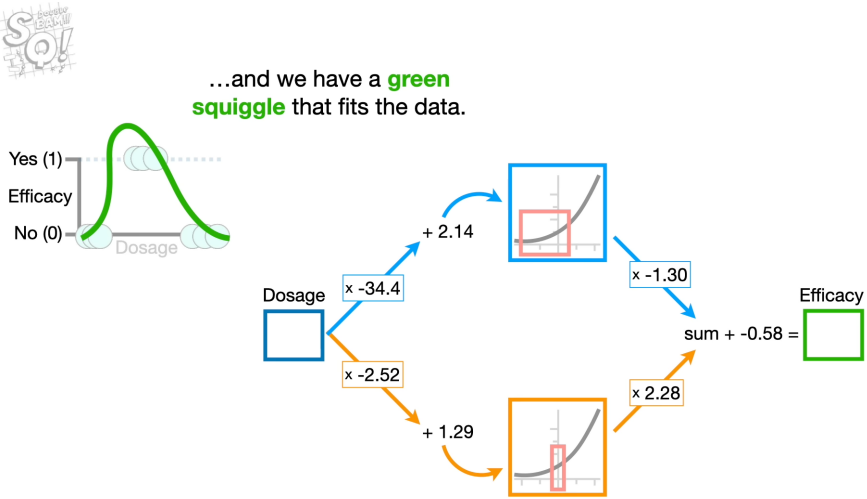
GANs I: Set-up

- Generative Adversarial Nets (GANs) was originally developed by Goodfellow et al. (2014)
- Suppose we have a **real-wrold** i.i.d. sample $\mathbf{X}_R \sim p_R(\cdot)$
- We want to generate an **artificial** sample $\mathbf{X}_A \sim p_A(\cdot)$ that is distributed similarly to $\mathbf{X}_R$
- The two *neural networks*:
  1. **Generator** $\mathcal{G}(\mathbf{Z}; \boldsymbol{\theta}_{\mathcal{G}})$
     - Transforms noise $\mathbf{Z} \sim p_Z(\cdot)$ into realistic $\mathbf{X}_R$
  2. **Discriminator** $\mathcal{D}(\mathbf{X}; \boldsymbol{\theta}_{\mathcal{D}})$
     - Analyses and classifies the datapoints in $\mathbf{X}$ as either real or artificial
- GANs training can be thought of as a **zero-sum game** with two players where parameters $\boldsymbol{\theta}_{\mathcal{G}}$ and $\boldsymbol{\theta}_{\mathcal{D}}$ determine their strategies

By the way: Neural Networks I



By the way: Neural Networks II



GANs II: Payoffs

- The players' payoffs can be summarised in the following value function

$$\min_{\mathcal{G}} \max_{\mathcal{D}} V(\mathcal{D}, \mathcal{G}) = \underbrace{\mathbb{E}_{\mathbf{X}_R \sim p_R(\mathbf{X})}[\ln \mathcal{D}(\mathbf{X}_R)]}_{\substack{\text{Expectation of } \mathcal{D} \text{ correctly} \\ \text{classifying } \textbf{real} \text{ data}}} + \underbrace{\mathbb{E}_{\mathbf{Z} \sim p_Z(\mathbf{Z})}[\ln(1 - \mathcal{D}(\mathcal{G}(\mathbf{Z})))]}_{\substack{\text{Expectation of } \mathcal{D} \text{ correctly} \\ \text{classifying } \textbf{artifitial} \text{ data}}} \tag{1}$$

where $\mathcal{D}(\mathbf{X}) \equiv$ probability of $\mathbf{X}$ coming from p_R as opposed to p_A

- for any given $\mathcal{G}$, $\mathcal{D}$ achieves optimality at $\mathcal{D}^*_{\mathcal{G}}(\mathbf{X}) = \frac{p_R(\mathbf{X})}{p_R(\mathbf{X}) + p_A(\mathbf{X})}$
  - Note: $\mathcal{D}$ is trained in its inner loop for T repetitions (large T may cause overfitting)
- $\mathcal{G}$ then aims to come up with such mapping $\mathbb{Z} \xrightarrow{\mathcal{G}} \mathbb{X}$ that $p_A = p_R$ and $\mathcal{D}(\mathbf{X}) = 0.5$

GANs III: Updating the parameters

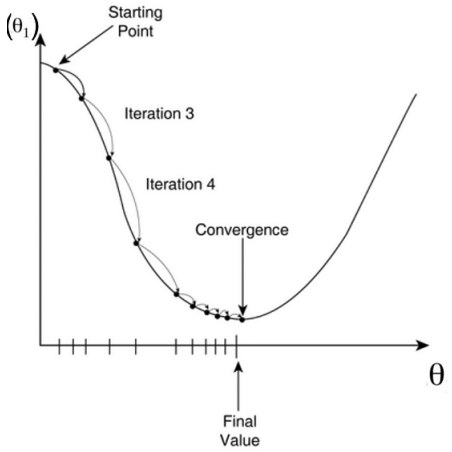
The parameters are updated using **stochastic gradient descent/ascend**. At the $(r+1)$th training iteration:

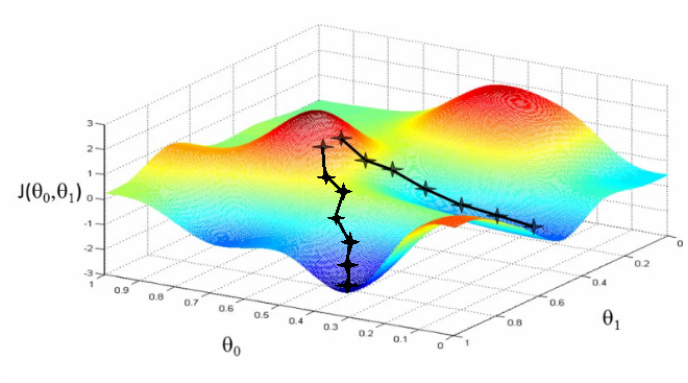
$$\boldsymbol{\theta}_{\mathcal{G},r+1} = \boldsymbol{\theta}_{\mathcal{G},r} - \lambda_r \nabla_{\boldsymbol{\theta}_{\mathcal{G}}} \frac{1}{M} \sum_{i=1}^{M} \ln(1 - \mathcal{D}(\mathcal{G}(\mathbf{z}_i)) \tag{2}$$

$$\underbrace{\boldsymbol{\theta}_{\mathcal{D},r+1}}_{\text{New}} = \underbrace{\boldsymbol{\theta}_{\mathcal{D},r}}_{\text{Old}} + \underbrace{\lambda_r \underbrace{\nabla_{\boldsymbol{\theta}_{\mathcal{D}}} \frac{1}{M} \sum_{i=1}^{M} [\ln \mathcal{D}(\mathbf{x}_i) + \ln(1 - \mathcal{D}(\mathcal{G}(\mathbf{z}_i)))]}_{\left[\frac{\partial V}{\partial \theta_1}, \frac{\partial V}{\partial \theta_2}, \frac{\partial V}{\partial \theta_3}, \ldots\right]'}}_{\text{Step size}} \tag{3}$$

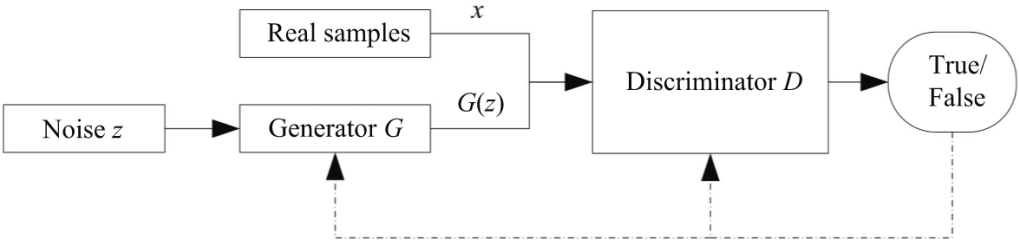
where $\lambda_r \in \mathbb{R}_+$ is the *learning rate* and M is the *minibatch size* (both hyperparameters)

By the way: Gradient Descen





GANs IV: Structure



Source: Wang et al. (2017)

Vanishing Gradient Problem I

- GANs **does not require a closed-form expression** for the distribution of interest p_R (helpful for dealing with complex data)
- However, $\mathcal{D}$ has to use some **measure of statistical distance** to compare p_R and p_A
- Specifically, *Jensen–Shannon divergence* (JS), which is an improved version of *Kullback–Leibler divergence* (KL)

$$\begin{aligned}
\text{Training criterion } C(\mathcal{G}) &= \max_{\mathcal{D}} V(\mathcal{G}, \mathcal{D}) && (4)\\
&= \mathbb{E}_{\mathbf{X}\sim p_R(\mathbf{X})}[\ln \mathcal{D}^*_{\mathcal{G}}(\mathbf{X})] + \mathbb{E}_{\mathbf{Z}\sim p(\mathbf{Z})}[\ln(1 - \mathcal{D}^*_{\mathcal{G}}(\mathcal{G}(\mathbf{Z})))] && (5)\\
&= \dots && (6)\\
&= KL\left(p_R \middle|\middle| \frac{p_R + p_A}{2}\right) + KL\left(p_A \middle|\middle| \frac{p_R + p_A}{2}\right) - \ln 4 && (7)\\
&= 2 \cdot JS(p_R || p_A) - \ln 4. && (8)
\end{aligned}$$

Vanishing Gradient Problem II

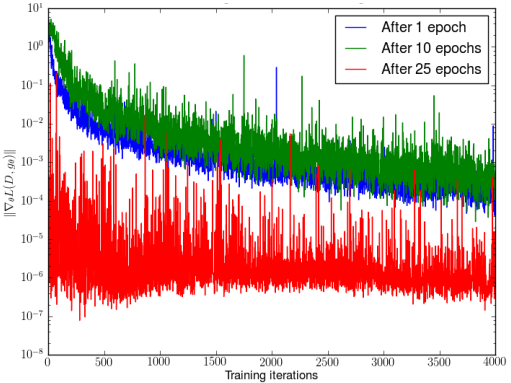
- Recall

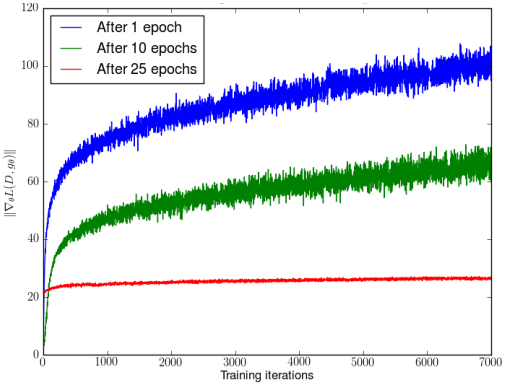
$$\boldsymbol{\theta}_{\mathcal{G},r+1} = \boldsymbol{\theta}_{\mathcal{G},r} - \lambda \nabla_{\boldsymbol{\theta}_{\mathcal{G}}} \frac{1}{M} \sum_{i=1}^{M} \ln(1 - \mathcal{D}(\mathcal{G}(\mathbf{z}_i))$$

- Since $\mathcal{G}$ **is poor** early in the process, $\mathcal{D}$ can easily spot $\mathbf{X}_A$, i.e. $\mathcal{D}(\mathbf{X})$ is close to 1
- Hence, $\ln(1 - \mathcal{D}(\mathcal{G}(\mathbf{z}_i)))$ is low and the **gradient vanishes** early in the training (before $\mathcal{G}$ has received sufficient information about p_R)
- Goodfellow et al. (2014) **solution**: train $\mathcal{G}$ to minimise the probability of $\mathcal{D}$ correctly identifying $\mathbf{X}_A$, i.e. replace $\ln(1 - \mathcal{D}(\mathcal{G}(\mathbf{z}_i)))$ with $-\ln(\mathcal{D}(G(\mathbf{z}_i))$
- But Arjovsky and Bottou (2017) show that distribution $\mathbb{E}_{\mathbf{Z}\sim p(\mathbf{Z})}[\nabla_{\boldsymbol{\theta}_{\mathcal{G}}} \ln -\mathcal{D}(\mathcal{G}(\mathbf{z}_i))]$ is **centred** and features **infinite expectation and variance** (no feedback from the gradient)

Vanishing Gradient Problem III

Figure: Gradient for $\mathcal{G}$ with $\ln(1 - \mathcal{D}(\mathcal{G}(\mathbf{z}_i)))$ and $-\ln(\mathcal{D}(G(\mathbf{z}_i))$ costs (Arjovsky and Bottou, 2017)





Wasserstein GANs I

- Arjovsky and Bottou (2017) notice that for p_R and p_A that are close in terms of manifolds, $JS(p_R||p_A)$ is considerably higher than JS between the noisier versions of p_R and p_A, $JS(p_{R+\epsilon}||p_{A+\epsilon})$, that appear to overlap
- The vanishing gradient problem can be addressed by **adding a continuous noise term** ϵ to the list of $\mathcal{D}$'s inputs
- Wouldn't work with JS divergence due to its sensitivity to the presence of noise rather than the amount of it $\implies$ **need a new measure of distance**

Wasserstein GANs II

- *Wasserstein metric* (Earth mover's distance)

$$W(p_R, p_A) = \inf_{\gamma \in \Gamma} \int_{\mathbb{X} \times \mathbb{X}} ||x_R - x_A||_2 d\gamma(x_R, x_A) \tag{9}$$

  where Γ is a set of all possible joints on $\mathbb{X} \times \mathbb{X}$ that have margins p_R and p_A.
- Arjovsky and Bottou (2017) show that $W(p, p_{+\epsilon}) \leq Var(\epsilon)^{\frac{1}{2}}$
- The distance between p_R and p_A is then

$$W(p_R, p_A) \leq W(p_R, p_{R+\epsilon}) + W(p_{R+\epsilon}, p_{A+\epsilon}) + W(p_{A+\epsilon}, p_A) \tag{10}$$

$$\leq \underbrace{2Var(\epsilon)^{\frac{1}{2}}}_{\substack{\textit{decreased by} \\ \textit{annealing } \epsilon}} + \underbrace{2C(JS(p_{R+\epsilon}||p_{A+\epsilon}))}_{\text{minimized by GANs}} \tag{11}$$

where C is the diameter of a ball containing the support of $p_{R+\epsilon}$ and $p_{A+\epsilon}$

Wasserstein GANs III

- Arjovsky et al. (2017) use the Kantorovich-Rubinstein duality to rewrite W:

$$W(p_R, p_A) = \sup_{||\mathcal{D}||_L \leq 1} \mathbb{E}_{\mathbf{X} \sim p_R}[\mathcal{D}(\mathbf{X})] - \mathbb{E}_{\mathbf{X} \sim p_A}[\mathcal{D}(\mathbf{X})], \tag{12}$$

  where $\mathcal{D} : \mathbb{X} \to \mathbb{R}$ (a.k.a. "the Critic") is Lipschitz continuous with the constant of 1
- WGANs training criterion:

$$C(\mathcal{G}) = \max_{||\mathcal{D}||_L \leq 1} V(\mathcal{D}, \mathcal{G}) = \mathbb{E}_{\mathbf{X}_R \sim p_R(\mathbf{X})}[\mathcal{D}^*_{\mathcal{G}}(\mathbf{X}_R)] - \mathbb{E}_{\mathbf{Z} \sim p(\mathbf{Z})}[\mathcal{D}^*_{\mathcal{G}}(\mathcal{G}(\mathbf{Z}))] \tag{13}$$

Conditional WGANs I

- **Causality**: generate potential treated and control outcomes given a common set of pre-treatment variables $\Longrightarrow$ generate data from a **conditional distribution**
- Mirza and Osindero (2014): feed the information (labels) in a conditioning variable set $\mathbf{V} \in \mathbb{V}$ to both $\mathcal{G}$ and $\mathcal{D}$ as an additional input

$$C(\mathcal{G}) = \max_{||\mathcal{D}||_L \leq 1} V(\mathcal{D}, \mathcal{G}) = \mathbb{E}_{\mathbf{X}_R \sim p_R(\mathbf{X})}[\mathcal{D}^*_{\mathcal{G}}(\mathbf{X}_R|\mathbf{V})] - \mathbb{E}_{\mathbf{Z} \sim p(\mathbf{Z})}[\mathcal{D}^*_{\mathcal{G}}(\mathcal{G}(\mathbf{Z}|\mathbf{V}))]. \tag{14}$$

Conditional WGANs II

- Athey et al. (2021): the infinite-sample version of the CWGANs objective is **equivalent** to the one of the independent WGANs fitted at $\mathbf{V} = v$ for every $v \in \mathbb{V}$
- In finite-sample case, the optimisation is not performed separately for different $\mathbf{v}_i$, so the equivalent disappears but the **intuition is the same**
- Updated parameters for the $(r+1)$th training iteration:

$$\boldsymbol{\theta}_{\mathcal{G},r+1} = \boldsymbol{\theta}_{\mathcal{G},r} - \lambda \nabla_{\boldsymbol{\theta}_{\mathcal{G}}} \frac{1}{M} \sum_{i=1}^{M} \mathcal{D}(\mathcal{G}(\mathbf{z}_i|\mathbf{v}_i)) \tag{15}$$

$$\boldsymbol{\theta}_{\mathcal{D},r+1} = \boldsymbol{\theta}_{\mathcal{D},r} + \lambda \nabla_{\boldsymbol{\theta}_{\mathcal{D}}} \left[\frac{1}{M} \sum_{i=1}^{M} \mathcal{D}(\mathbf{x}_i|\mathbf{v}_i) - \frac{1}{M} \sum_{i=1}^{M} \mathcal{D}(\mathcal{G}(\mathbf{z}_i|\mathbf{v}_i)) \right] \tag{16}$$

Data I

- Field experiment data used by LaLonde (1986) and Dehejia and Wahba (1999)
  - + Current Population Survey (CPS) data
  - + Panel Survey of Income Dynamics (PSID) data
- Variables:
  - Outcome $\mathbf{y}(\mathbf{w})$: earnings in 1978
  - Treatment $\mathbf{w}$: National Supported Work (NSW) Demonstration (a training programme)
  - Pre-treatment variables $\mathbf{X}$: earnings in '74 and '75, black, hispanic, age, marital status, and two education measures

Data II

Table 1
Summary statistics for Lalonde–Dehejia–Wahba data.

	Experimental trainees (185)		Experimental controls (260)		CPS controls (15,992)		PSID controls (2490)	
	mean	s.d.	mean	s.d.	mean	s.d.	mean	s.d.
black	0.84	(0.36)	0.83	(0.38)	0.07	(0.26)	0.25	(0.43)
hispanic	0.06	(0.24)	0.11	(0.31)	0.07	(0.26)	0.03	(0.18)
age	25.82	(7.16)	25.05	(7.06)	33.23	(11.05)	34.85	(10.44)
married	0.19	(0.39)	0.15	(0.36)	0.71	(0.45)	0.87	(0.34)
nodegree	0.71	(0.46)	0.83	(0.37)	0.3	(0.46)	0.31	(0.46)
education	10.35	(2.01)	10.09	(1.61)	12.03	(2.87)	12.12	(3.08)
earn '74	2.1	(4.89)	2.11	(5.69)	14.02	(9.57)	19.43	(13.41)
earn '75	1.53	(3.22)	1.27	(3.1)	13.65	(9.27)	19.06	(13.6)
earn '78	6.35	(7.87)	4.55	(5.48)	14.85	(9.65)	21.55	(15.56)

CWGANs Settings

Generator

- 3 hidden layers + an output layer where $(\cdot, \cdot)$ denotes $(\#inputs, \#outputs)$:
  $(d_{modeled} + d_{condition}, 128)$, $(128, 128)$, $(128, 128)$, $(128, d_{modeled})$
  - Dimensions for generating $\mathbf{X}|\mathbf{w}$: $d_{modeled} = 8$, $d_{condition} = 1$
  - Dimensions for generating $\mathbf{y}|\mathbf{X}, \mathbf{w}$: $d_{modeled} = 1$, $d_{condition} = 9$
- Activation functions:
  - Hidden layers: Rectified Linear Unit (ReLU) $a(z) = z\mathbf{1}_{z>0} = \max(0, z) = \frac{z+|z|}{2}$
  - Output layer: varies depending on the variable type
- Minimabts size: 128 for experimental sample but larger for CPS and PSID

Discriminator (Critic)

- Same structure but the output layer has (128, 1)

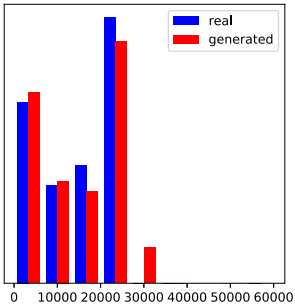
Post-training

- Generate a large sample with $N = 10^6$ to be used as simulated population
  1. Draw the covariates using trained $\mathcal{G}$ with $\theta_{\mathcal{G},\mathbf{X}|\mathbf{w}}$ for the treated and control units
  2. Draw $\mathbf{y}(\mathbf{w} = \mathbf{0})$ and $\mathbf{y}(\mathbf{w} = \mathbf{1})$ independently using trained $\mathcal{G}$ with $\theta_{\mathcal{G},\mathbf{y}|\mathbf{X},\mathbf{w}}$
- Since in simulation we observe both $y_i(w_i = 0)$ and $y_i(w_i = 1)$ for each i, the approximate true ATT is given by

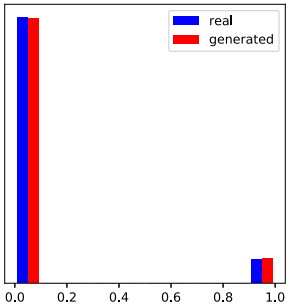
$$\tau = \frac{1}{N_1} \sum_{i:w_i=1}^{N_1} (y_i(1) - y_i(0)) \tag{17}$$

- The **first two moments** of the simulated data are similar to the ones of the real data, however, it **can be achieved** with the bootstrap or a multivariate normal distribution.
- GANs allows to generate artificial samples that contain observations **not featured** in the original dataset and with **no duplicates**

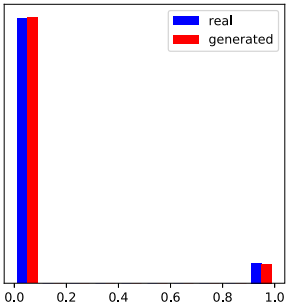
Real vs Generated CPS Data I



(a) Earnings 1978

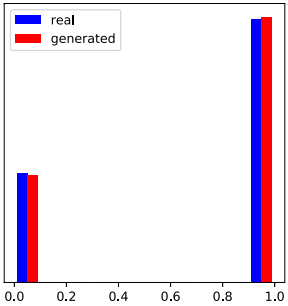


(b) Black

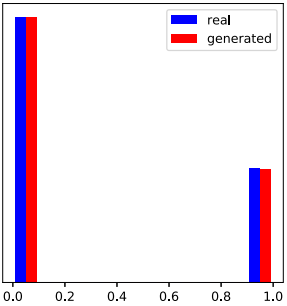


(c) Hispanic

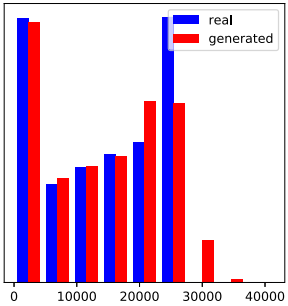
Real vs Generated CPS Data II



(d) Married

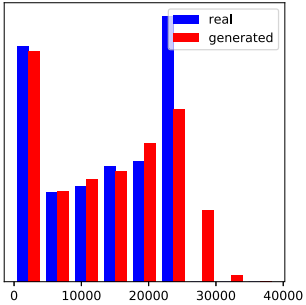


(e) No Degree

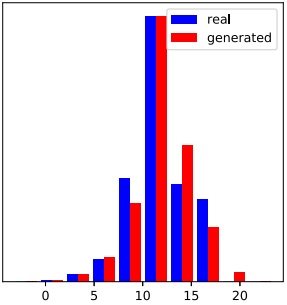


(f) Earnings 1974

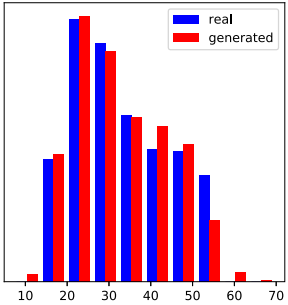
Real vs Generated CPS Data III



(g) Earnings 1975

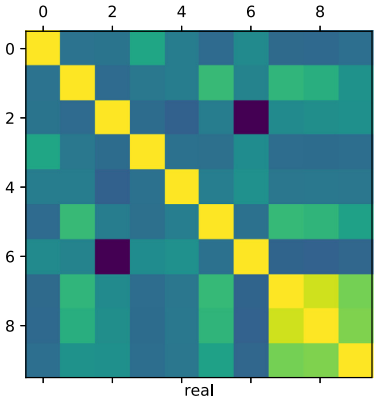


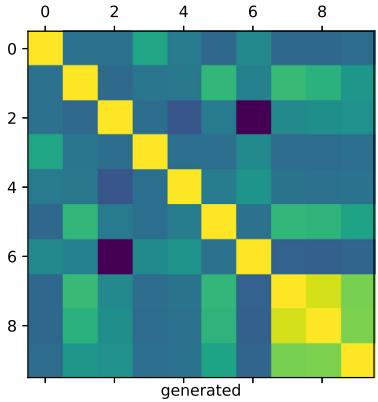
(h) Education



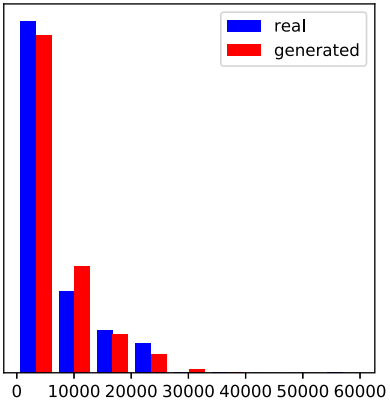
(i) Age

Real vs Generated CPS Data IV: Correlations

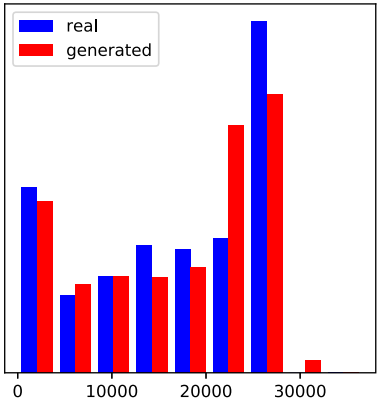




Real vs Generated CPS Data V: Conditional Distributions



(a) Earnings 1978 | Earnings 1974= 0



(b) Earnings 1978 | Earnings 1974 > 0

Comparing 13 ATT Estimators I

- 1. Difference in Means (DIFF)
- 2. Bias-Adjusted Matching (BCM)
- Conditional Outcome Model
  - 3. Linear Model (LIN)
  - 4. Random Forest (RF)
  - 5. Neural Nets (NN)
- The Horowitz–Thompson Inverse Propensity Weighting (IPWE)
  - 6. Logit Model (LIN)
  - 7. Random Forest (RF)
  - 8. Neural Nets (NN)
- Double Robust Estimator
  - 9. Linear and Logit Model (LIN)
  - 10. Random Forest (RF)
  - 11. Neural Nets (NN)
- 12. Causal Forest (CF)
- 13. Residual Balancing (RB)

Comparing 13 ATT Estimators II

Table 5
Estimates based on LDW data.

	Experimental		CPS		PSID	
	Estimate	s.e.	Estimate	s.e.	Estimate	s.e.
Baselines						
DIFF	1.79	0.63	−8.50	0.71	−15.20	1.15
BCM	2.12	0.88	2.15	0.87	0.57	1.47
Outcome models						
L	1.79	0.57	0.69	0.60	0.79	0.60
RF	1.69	0.58	0.85	0.60	−0.20	0.56
NN	1.49	0.59	1.70	0.60	1.47	0.60
Propensity score models						
L	1.81	0.83	1.18	0.77	1.26	1.13
RF	1.90	0.86	0.73	0.82	0.24	1.00
NN	1.69	0.86	1.38	0.77	0.42	1.45
Doubly robust methods						
L	1.80	0.67	1.27	0.65	1.50	0.97
RF	1.93	0.70	1.63	0.76	0.98	0.83
NN	1.90	0.75	1.63	0.72	1.56	0.76
CF	1.72	0.68	1.58	0.67	0.59	0.78
RB	1.73	0.70	0.93	0.62	0.72	0.79

Comparing 13 ATT Estimators III

Table 6
Estimates based on LDW experimental data (2000 Replications).

Method	rmse	bias	sdev	Coverage
Baselines				
DIFF	0.49	0.06	0.48	0.94
BCM	0.58	0.00	0.58	0.96
Outcome models				
L	0.52	−0.06	0.51	0.88
RF	0.51	−0.07	0.50	0.88
NN	1.32	0.04	1.32	0.75
Propensity score models				
L	0.52	−0.08	0.52	0.99
RF	0.52	−0.06	0.51	0.99
NN	0.52	0.01	0.52	0.99
Doubly robust methods				
L	0.51	−0.08	0.51	0.95
RF	0.52	−0.04	0.52	0.95
NN	0.79	−0.05	0.79	0.95
CF	0.50	−0.09	0.49	0.94
RB	0.52	−0.09	0.51	0.95

Comparing 13 ATT Estimators IV

Table 7
Estimates based on LDW-CPS data (2000 Replications).

Method	rmse	bias	sdev	Coverage
Baselines				
DIFF	11.12	−11.11	0.45	0.00
BCM	0.73	0.07	0.73	0.96
Outcome models				
L	2.14	−2.08	0.51	0.02
RF	1.00	−0.87	0.51	0.54
NN	0.63	0.14	0.61	0.88
Propensity score models				
L	0.51	0.00	0.51	0.98
RF	1.00	−0.87	0.50	0.73
NN	0.65	0.23	0.61	0.94
Doubly robust methods				
L	0.53	0.03	0.53	0.96
RF	0.54	−0.05	0.54	0.93
NN	0.62	0.20	0.58	0.94
CF	0.55	0.11	0.53	0.91
RB	0.57	−0.22	0.52	0.89

Comparing 13 ATT Estimators V

Table 8
Estimates based on LDW-PSID data (2000 Replications).

Method	rmse	bias	sdev	Coverage
Baselines				
DIFF	18.81	−18.81	0.53	0.00
BCM	0.98	−0.02	0.98	0.98
Outcome models				
L	1.95	−1.82	0.72	0.12
RF	2.30	−2.22	0.62	0.02
NN	2.97	−0.93	2.82	0.59
Propensity score models				
L	1.11	−0.64	0.91	0.96
RF	2.21	−2.05	0.82	0.32
NN	1.82	−1.43	1.11	0.69
Doubly robust methods				
L	0.98	−0.35	0.92	0.94
RF	0.98	−0.57	0.80	0.84
NN	0.98	−0.38	0.90	0.92
CF	1.13	−0.89	0.69	0.73
RB	1.06	0.33	1.01	0.75

More comparisons from Athey et al. (2021)

- Comparing RMSEs of estimators for a **given dataset** yields a **unique ranking**, so there is a need for a measure of the robustness of that ranking
  - *Robustness to sample* – draw different samples; see what happens
  - *Robustness to model architecture* – change the hyperparameters of $\mathcal{G}$ and $\mathcal{D}$; see what happens
- One may need to generate data from a **restricted distribution**, e.g. estimating a demand function that is typically assumed to be monotone in prices
  - They compare plugging the restriction into the WGANs objective to alternative ways of imposing the restriction
  - WGANs appear to **remain stable** and **fit the unpenalised aspects** of the data better than the alternatives

Conclusions

- WGANs can be an effective way of generating **artifitial data** that **resambles real data**
- **Different estimators** emerge at the top **depending on the datasets** used (experimental, CPS, PSID)
- Changes **within** a specific dataset (e.g. changing the sample size) have **little effect**
- RF or NN-based **double robust** estimators feature the lowest overall loss in RMSE, although they get outperformed in some settings
- WGANs can be used to **impose restrictions** on the simulated distribution

References I

Susan Athey, Guido W Imbens, Jonas Metzger, and Evan Munro. Using wasserstein generative adversarial networks for the design of monte carlo simulations. *Journal of Econometrics*, 2021.

David H Wolpert. Stacked generalization. *Neural networks*, 5(2):241–259, 1992.

Achim Ahrens, Erkal Ersoy, Vsevolod Iakovlev, Haoyang Li, and Mark E Schaffer. An introduction to stacking regression for economists. In *International Conference of the Thailand Econometrics Society*, pages 7–29. Springer, 2022.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

Kunfeng Wang, Chao Gou, Yanjie Duan, Yilun Lin, Xinhu Zheng, and Fei-Yue Wang. Generative adversarial networks: introduction and outlook. *IEEE/CAA Journal of Automatica Sinica*, 4(4):588–598, 2017.

References II

Martin Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862*, 2017.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

Robert J LaLonde. Evaluating the econometric evaluations of training programs with experimental data. *The American economic review*, pages 604–620, 1986.

Rajeev H Dehejia and Sadek Wahba. Causal effects in nonexperimental studies: Reevaluating the evaluation of training programs. *Journal of the American statistical Association*, 94 (448):1053–1062, 1999.