Flexible Multiple Imputation for Social Science using Generative Adversarial Nets

International Methods Colloquium: February 8, 2019.

Marcel Neunhoeffer

PhD Candidate, University of Mannheim, marcel-neunhoeffer.com

Table of contents

Introduction to Generative Adversarial Nets.

Generative Adversarial Nets Can Make You Rich.

Figure 1: Edmond de Belamy, painted by a GAN. Source: Christie's.

Generative Adversarial Nets Are Surprisingly Simple.

- Generative Adversarial Nets (GANs), introduced by Goodfellow et al. (2014), allow it to sample from arbitrary joint (continuous) distributions.
- At its core, a GAN is a minimax game with two competing actors–a discriminator (D) trying to tell real from synthetic samples and a generator (G) to produce realistic synthetic samples from random noise.
- Formally, this two-player minimax game can be written as:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}(x)}\Big[\log D(x)\Big] + \mathbb{E}_{z\sim p_z(z)}\Big[\log(1-D(G(z)))\Big]$$

where $p_{data}(x)$ is the distribution of the real data, X is a sample from $p_{data}(x)$. The generator network $G(z)$ takes as input z from $p(z)$, where z is a random sample from a probability distribution $p(z)$.

What Does A Schematic GAN Look Like?

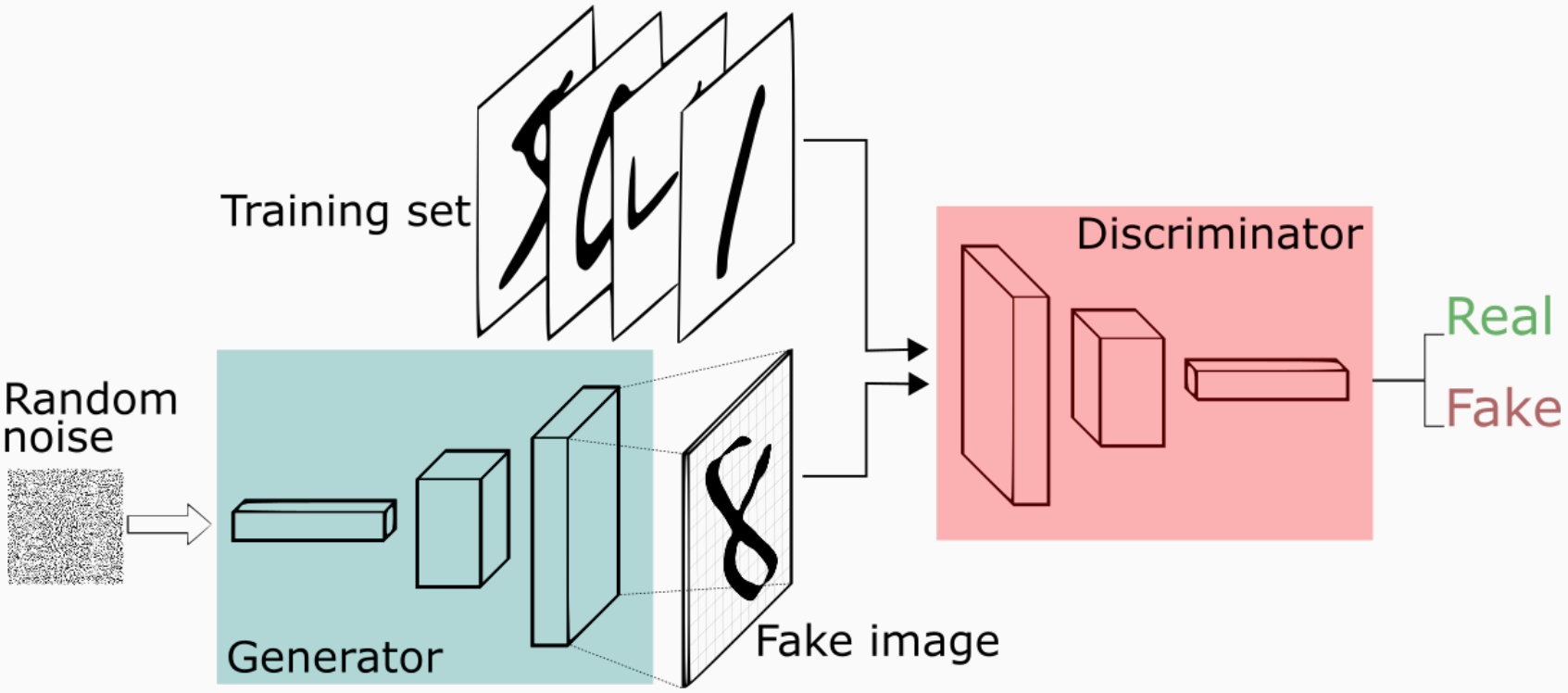

Figure 2: The architecture of a GAN. Source: Freecodecamp.org, Thalles Silva

A Closer Look at the Discriminator.

- The Discriminator is a binary classifier.
- An example for a binary classifier common in social sciences is a logit model.
- In GANs one will usually find neural nets.
- The Discriminator takes training examples and newly generated “fake” examples as input.
- The goal of the Discriminator is to tell real from “fake” examples.

A Closer Look at the Generator.

- The Generator is a model that learns parameters to generate real looking examples from random noise.
- It takes random noise as an input and outputs “fake” data of the dimension of the original data.
- In GANs one will usually find neural nets with an output layer of the dimensions of the training data as a Generator.

How Can We Train a GAN?

- A GAN is a dynamic system.
- Every update of the Discriminator or Generator changes the optimization landscape.
- The goal is to achieve an equilibrium between the Generator and the Discriminator.
- A GAN is usually trained iteratively (update D, then update G, then D, then G, ...) with some form of mini batch Stochastic Gradient Descent (SGD).
- In practice it is hard to determine when an equilibrium between D and G is reached. However, sufficiently good Generators have proven to be powerful enough to produce impressive results (like paintings).

How Could Multiple Imputation Profit from GANs?

GANs Can Learn Arbitrary Complex Joint Distributions.

- In theory, GANs can learn arbitrary complex joint distributions.
- A straightforward application of this capability to social science is Multiple Imputation.

A Brief (Re-)Introduction of Multiple Imputation.

- The idea of Multiple Imputation was first introduced in the late 1970s by Rubin (Rubin, 1978).
- It gained some traction in the last decade and is now widely applied in the social sciences (e.g. King et al., 2001; Honaker and King, 2010; Van Buuren, 2012; Kropko et al., 2014; Hollenbach et al., 2018).
- Still, there is some debate over the pros and cons of Multiple Imputation – mainly regarding the assumptions about the missingness mechanism (e.g. Lall, 2016; Arel-Bundock and Pelc, 2018; Pepinsky, 2018).

A Brief (Re-)Introduction of Multiple Imputation.

- The imputation model for multiple imputation could be any model to fill in missing values.
- To reflect imputation uncertainty we create m (multiple) completed data sets and analyze them separately.
- The results of the m analyses are then combined according to Rubin's rules:
  - $\bar{q} = \frac{1}{m} \sum_{j=1}^{m} q_j$
  - $SE(q)^2 = \frac{1}{m} \sum_{j=1}^{m} SE(q_j)^2 + S_q^2 * (1 + \frac{1}{m})$, where $S_q^2 = \sum_{j=1}^{m} \frac{(q_j - \bar{q})^2}{m-1}$

Preliminaries - Some recurring terms and Notation.

- **M**issing **C**ompletely **A**t **R**andom (MCAR).
- **M**issing **A**t **R**andom (MAR).
- **N**ot **M**issing **A**t **R**andom (NMAR).
- D: **D**ata matrix with observed and missing values.
- M: **M**issingness indicator matrix.
- Note that for any imputation approach to work the missing values have to be MAR (with MCAR being a special case of MAR).
- Formally, $p(M|D) = p(M|D^{obs})$

Preliminaries - Some Recurring Terms and notation.

$$D = \begin{bmatrix} 1 & 2 & NA \\ NA & 5 & 6 \\ NA & NA & 9 \end{bmatrix} \qquad M = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

How Do We Multiply Impute So Far?

- Current algorithms make assumptions either about the **joint distribution** or about the **conditional distributions** of columns with missing data:
- *Amelia II* for example assumes that the data is jointly multivariate normal.
- *mice* or *mi* model conditional distributions.

How Does Amelia II Work?

"The imputation model in Amelia II assumes that the complete data (that is, both observed and unobserved) are jointly multivariate normal. If we denote the $(n \times k)$ *dataset as* D *(with observed part* D^{obs} *and unobserved part* D^{mis}*), then this assumption is* $D \sim \mathcal{N}_k(\mu, \Sigma)$*, which states that* D *has a multivariate normal distribution with mean vector* μ *and covariance matrix* Σ*" (Honaker et al., 2011)*

- The mean vector and the covariance matrix of the multivariate normal distribution are estimated using the Expectation-Maximization with Bootstrapping (EMB) algorithm.

How Does mice Work?

"We assume that the multivariate distribution of Y is completely specified by θ, a vector of unknown parameters. The problem is how to get the multivariate distribution of θ, either explicitly or implicitly. The MICE algorithm obtains the posterior distribution of θ by sampling iteratively from conditional distributions of the form

$$P(Y_1|Y_{\neg 1}, \theta_1)$$

$$\ldots$$

$$P(Y_p|Y_{\neg p}, \theta_p)''$$

(Van Buuren and Groothuis-Oudshoorn, 2011).

- The vector of unknown parameters is estimated iteratively, usually starting with the column with the least missing values.

Comparing the Two Approaches.

- For Amelia II, the choice of the distribution is a strong assumption. The standard implementations assume the data to be multivariate normal.
- For mice, the specification of the conditional distributions is key and requires a lot more input form the imputer than Amelia II.
- If the assumptions are met, both models provide accurate imputations.

Multiple Imputation with Generative Adversarial Nets.

Imputation with a GAN.

- For imputation I only want to sample imputations for missing values from the underlying joint distribution.
- Yoon et al. (2018) propose a straightforward extension to the basic GAN for imputation (GAIN).
- The Generator is trained to only generate imputations for missing values instead of generating entire observations.
- The Discriminator is trained to tell observed values from imputed values.

The GAIN architecture.

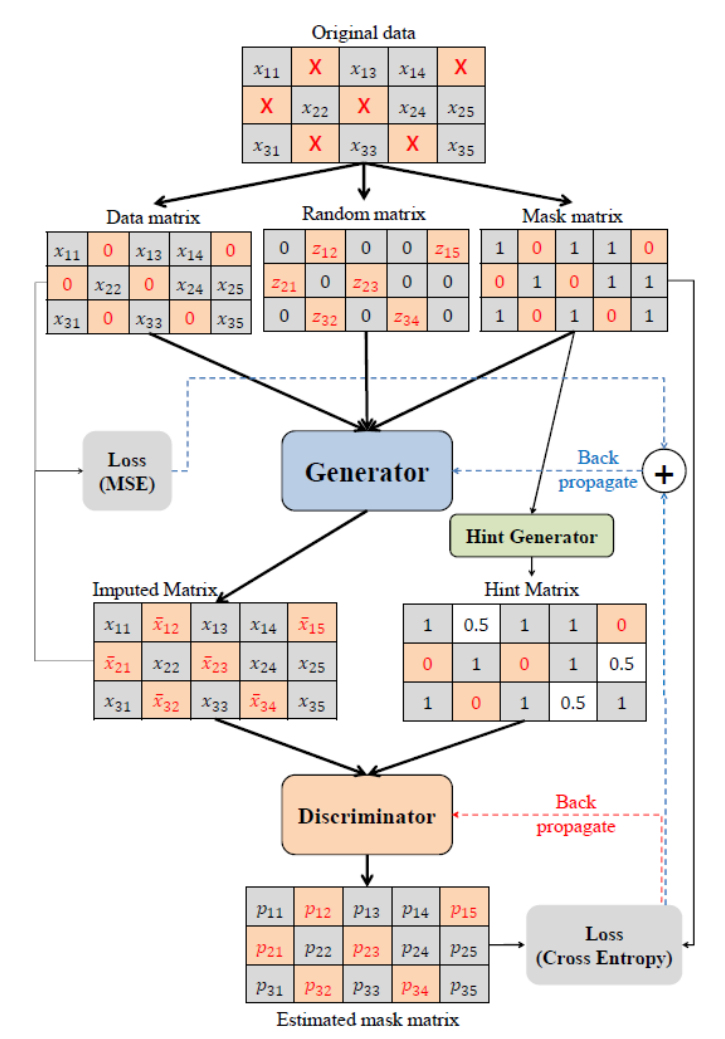

Figure 3: The architecture of GAIN. Source: Yoon et al., 2018

Metropolis Hastings Sampler for Multiple Imputations

- In practice it is very hard to train a Discriminator and Generator to equilibrium.
- Turner et al. (2018) observe that training a good Discriminator is a simpler task than training a good Generator.
- They propose a Metropolis Hastings Sampler to augment the Generator based on the Discriminator scores of generated samples.
- I implement the Metropolis Hastings Sampler together with GAIN (MH-GAIN).
- To achieve that I train a second Discriminator (MH Discriminator) that scores the quality of completed rows (with a score of 1 being a row without missing values).
- With the Metropolis Hastings Sampler it is possible to draw multiple imputations from the distribution of the data.

Putting it all together: MH-GAIN 1/4.

1. Initialize all weights of MH-GAIN.
2. Sample mini batch of n (e.g. 128) observations from data matrix D with k columns.
3. Fill in missing values with a draw from $p(z)$.
4. Bind the matrix from 3. to the missingness indicator matrix M.
5. Feed the matrix from 4. to the Generator (input dimension $n \times k * 2$, output dimension $n \times k$).
6. Generate the Hint matrix H. And bind it to the output of 5.
7. Feed the matrix from 6. to the Discriminator (input dimension $n \times k * 2$, output dimension $n \times k$).
8. Calculate the loss of the Discriminator and update the weights.
9. Calculate the loss of the Generator and update the weights.
10. Repeat 2 to 9 for a defined number of iterations (e.g. 20,000).

Putting it all together: MH-GAIN 2/4.

For the Metropolis Hastings Sampler the MH-Discriminator is trained parallel to the Discriminator:

1. ...
2. ...
3. ...
4. ...
5. ...
6. Generate the MH-Hint matrix $MH - H$. And bind it to the output of 5.
7. Feed the matrix from 6. to the MH-Discriminator (input dimension $n \times k + 1$, output dimension $n \times 1$).
8. Calculate the loss of the MH-Discriminator and update the weights.
9. ...
10. Repeat 2 to 9 for a defined number of iterations (e.g. 20,000).

Putting it all together: MH-GAIN 3/4.

After the last iteration sample m multiple imputations from D using the Metropolis Hastings sampler:

1. Fill in missing values in the original data with a draw from $p(z)$.
2. Bind the matrix from 2. to the missingness indicator matrix M.
3. Feed the matrix from 3. to the Generator (input dimension $N \times k * 2$, output dimension $N \times k$)–Current Sample.
4. Calculate the MH-Discriminator score for each observation in the Current Sample (Current Score).
5. Fill in missing values in the original data with a draw from $p(z)$.
6. Bind the matrix from 5. to the missingness indicator matrix M.
7. Feed the matrix from 6. to the Generator (input dimension $N \times k * 2$, output dimension $N \times k$)–Proposed Sample.
8. Calculate the MH-Discriminator score for each observation (Proposed Score).

Putting it all together: MH-GAIN 4/4.

9. Calculate Acceptance Ratio: $\frac{\text{Proposed Score}}{\text{Current Score}}$.
10. Draw Acceptance Probability for each observation from a uniform distribution.
11. Update rows in Current Sample with rows from Proposed Sample for which Acceptance Ratio $>$ Acceptance Probability.
12. Repeat 4 to 11 until for a defined number of burn-in $+ m$ iterations.
13. Return the m imputed data sets.

Experimental Evidence.

Imputation of Data From a Complex Distribution.

The experimental data set has five columns and 1,000 observations. It contains the following Variables:

- $Y = 0 + 0X_1 + 20X_2 + 10X_1X_2 + \epsilon$
- $X_1 \sim \mathcal{N}(0, 1)$
- $X_2 \sim \text{Binom}(p = 0.5)$
- $X_3 \sim \mathcal{N}(0, 1)$
- $X_4 \sim f(x, \mu_1 = -50, \mu_2 = 50, \sigma_1 = 1, \sigma_2 = 10) = \sum_i^2 0.5 * \mathcal{N}(x; \mu_i, \sigma_i)$

X_3 is fully observed, about 19% of Y and X_4 are MCAR, and about 7% of X_1 and X_2 are MAR depending on values of X_3.

The goal is to recover the coefficients in the regression of Y on X_1 and X_2 with the interaction term. As well as to recover the bimodal mixture of normals in X_4.

What would happen with (naive) Amelia II?

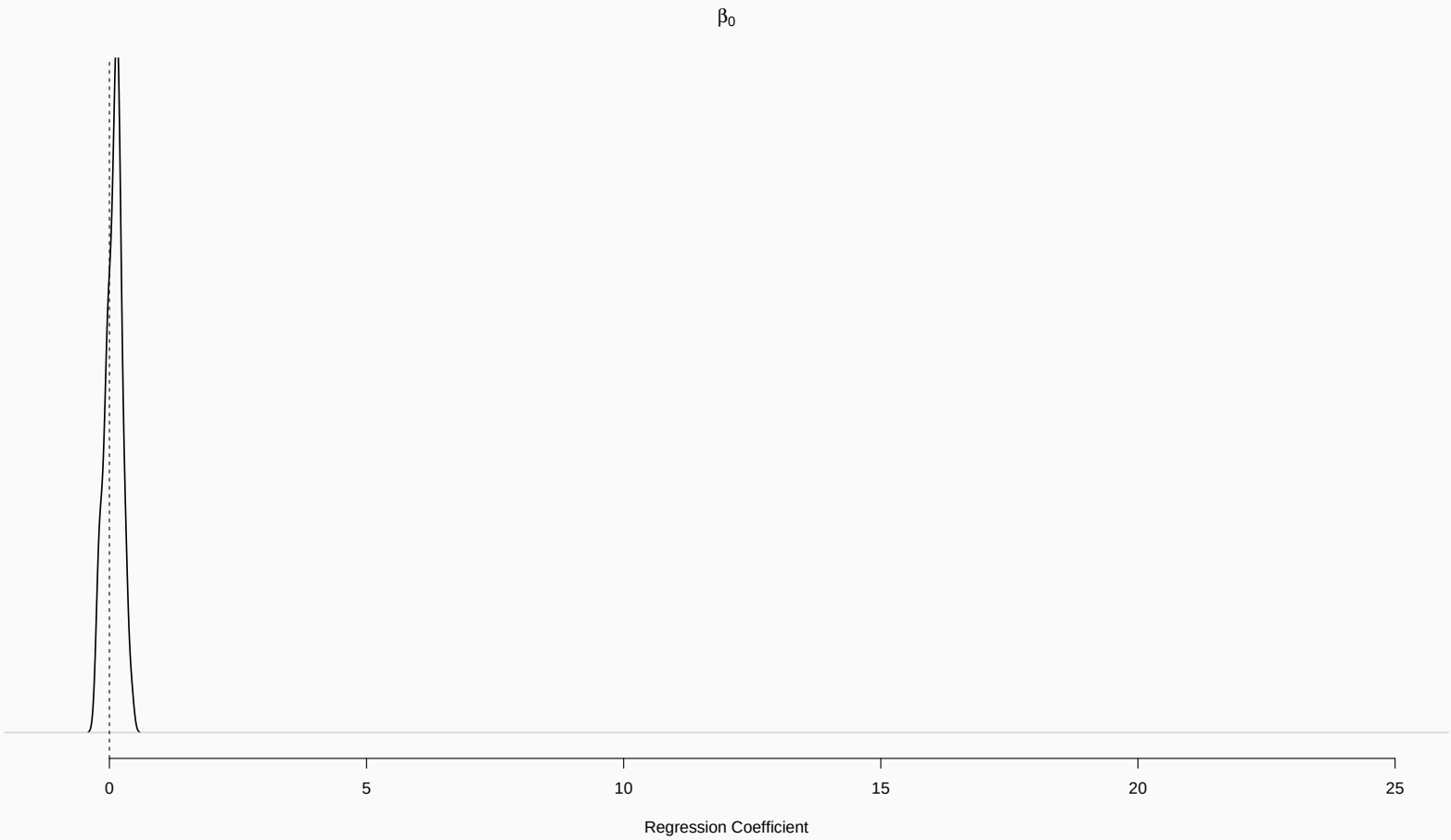

Figure 4: The distribution of β_0 after imputation with Amelia II.

What would happen with (naive) Amelia II?

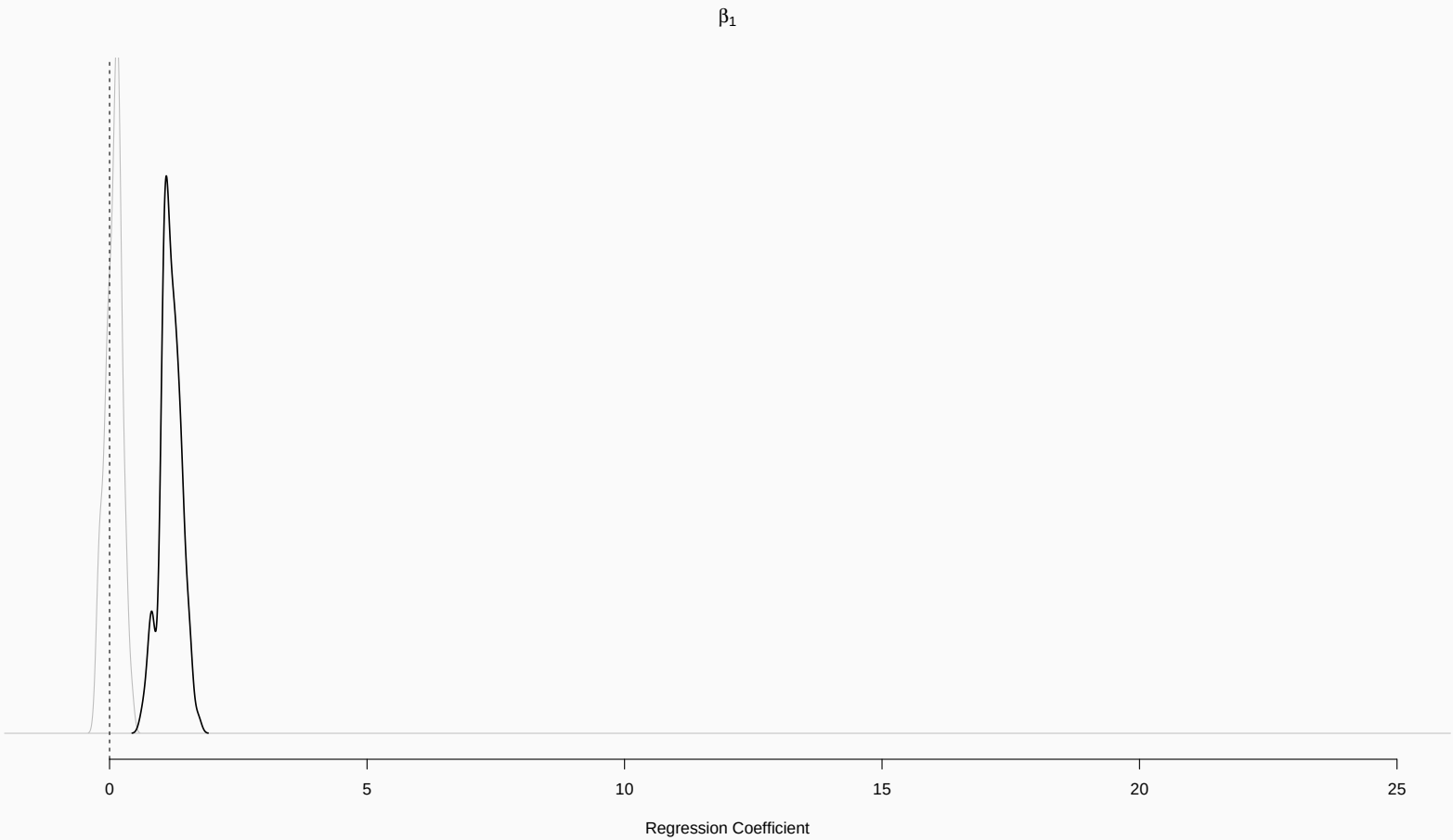

Figure 5: The distribution of β_1 after imputation with Amelia II.

What would happen with (naive) Amelia II?

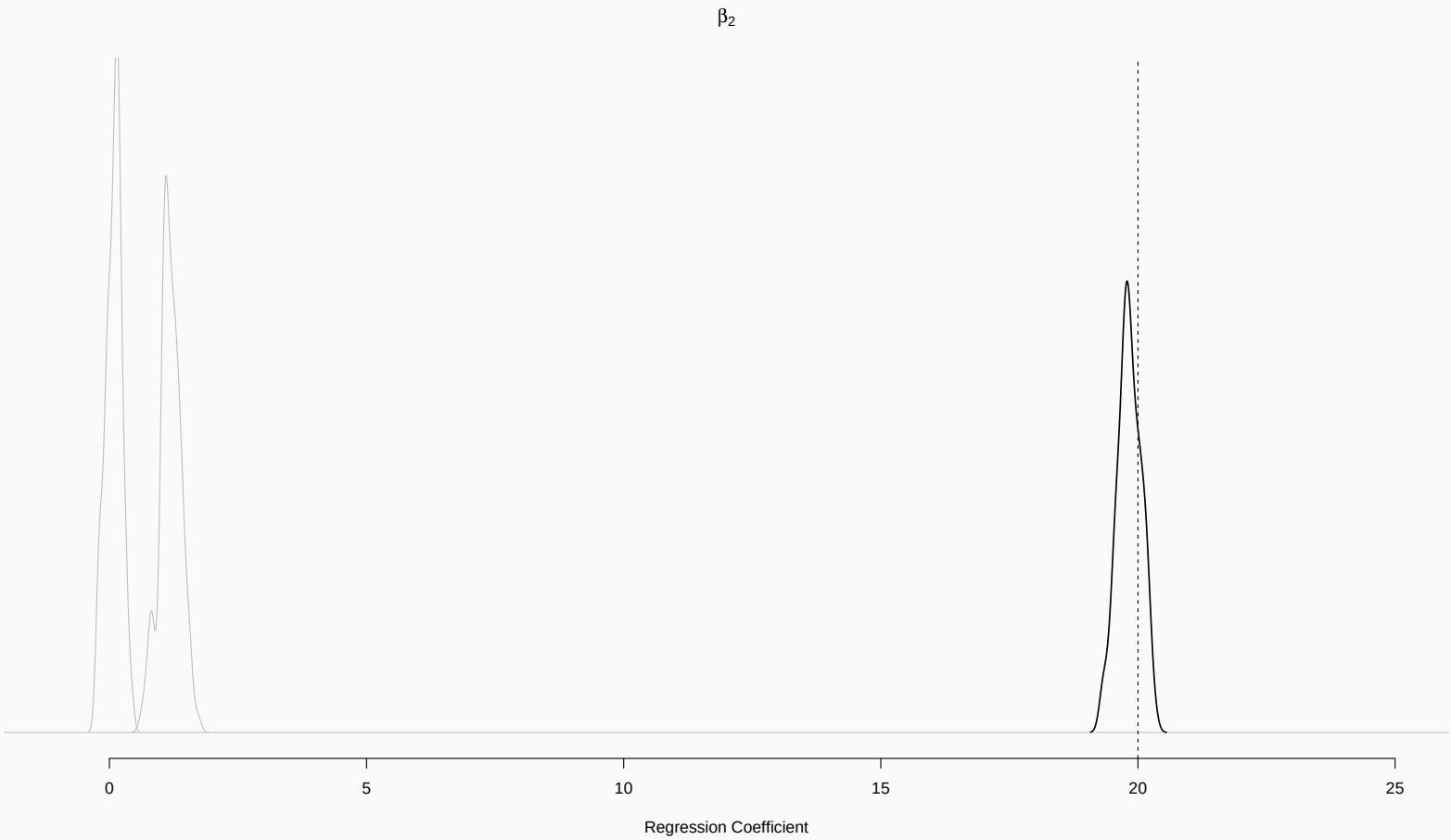

Figure 6: The distribution of β_2 after imputation with Amelia II.

What would happen with (naive) Amelia II?

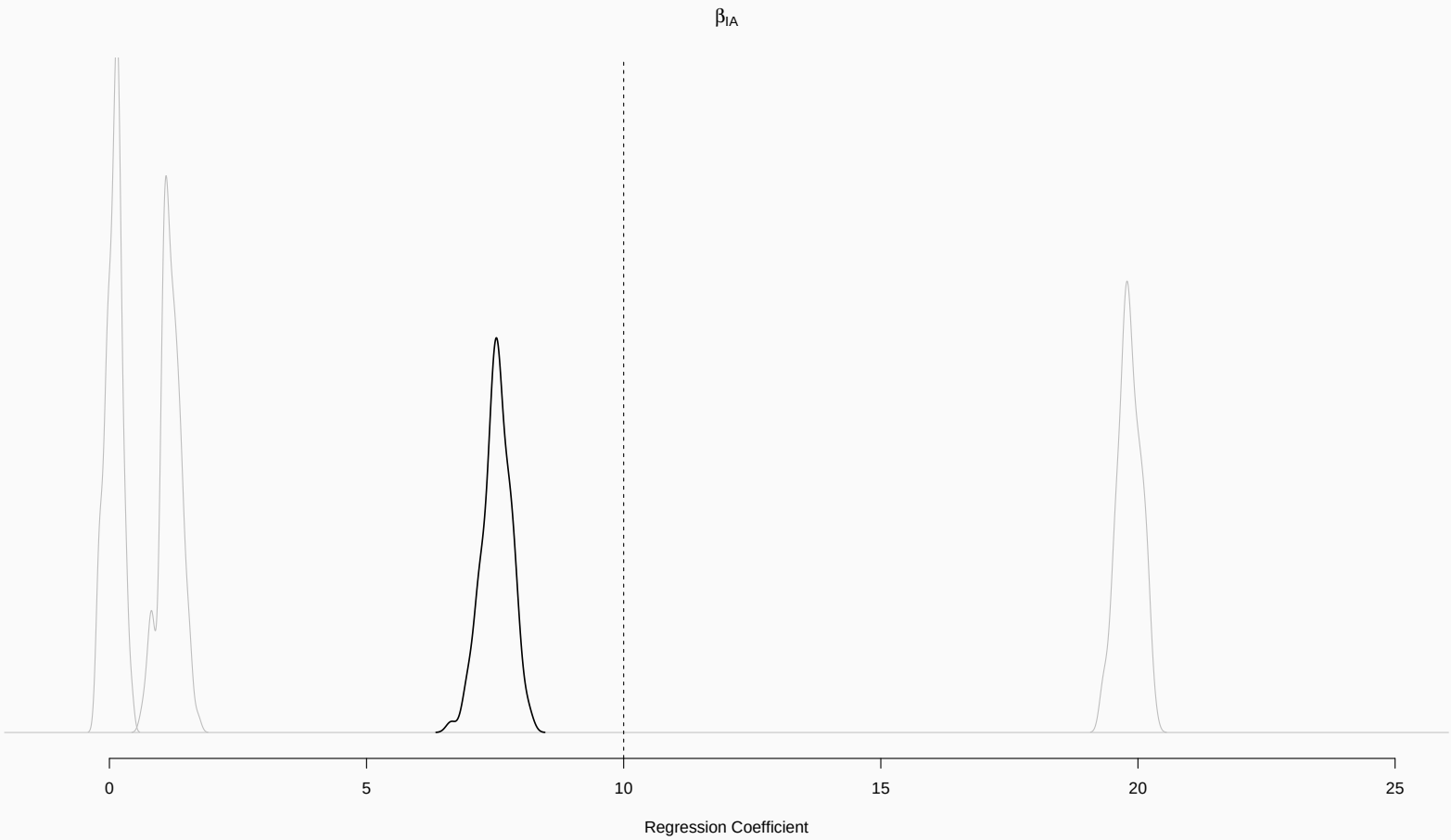

Figure 7: The distribution of β_{IA} after imputation with Amelia II.

What would happen with (naive) Amelia II?

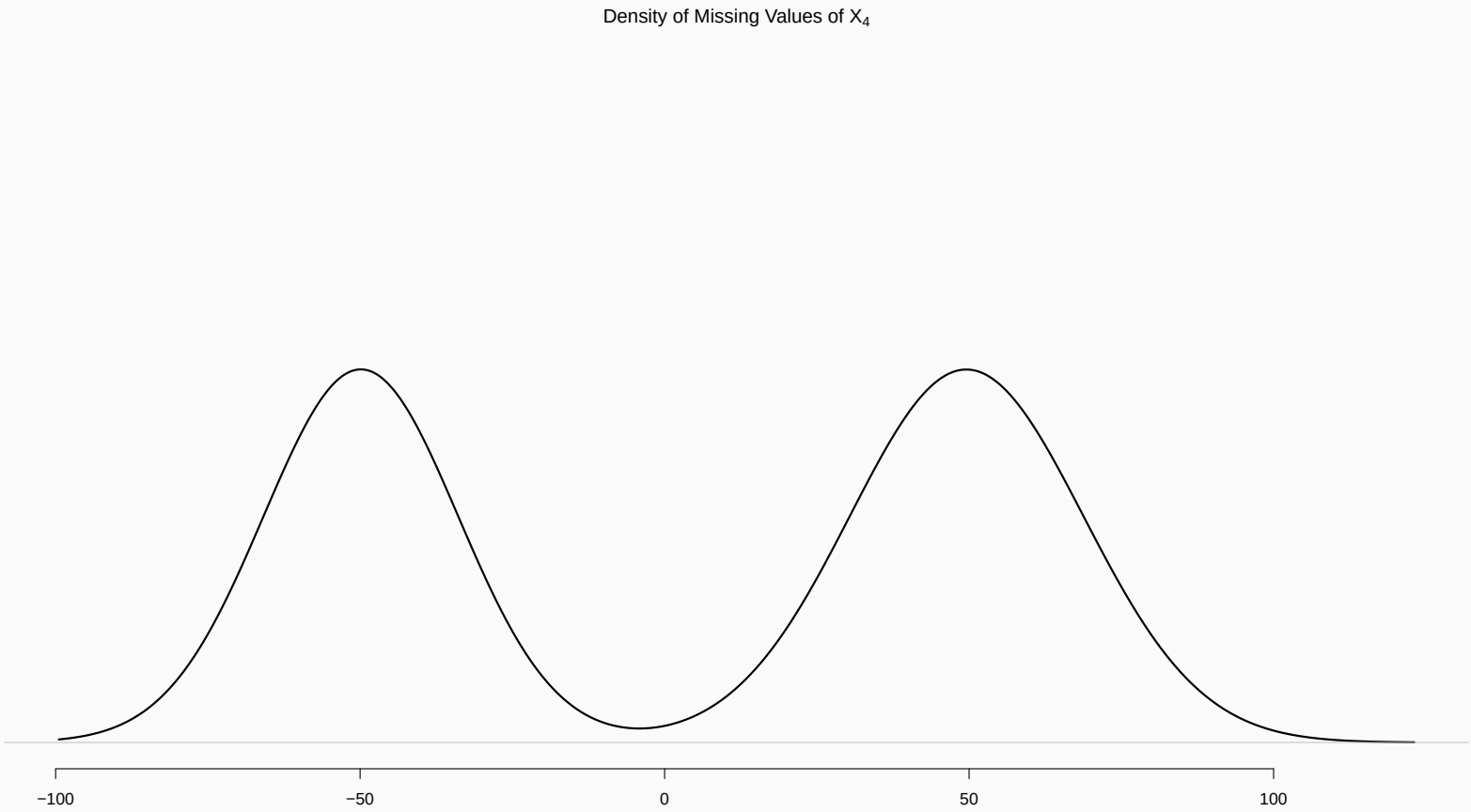

Figure 8: The distribution of missing values of X_4.

What would happen with (naive) Amelia II?

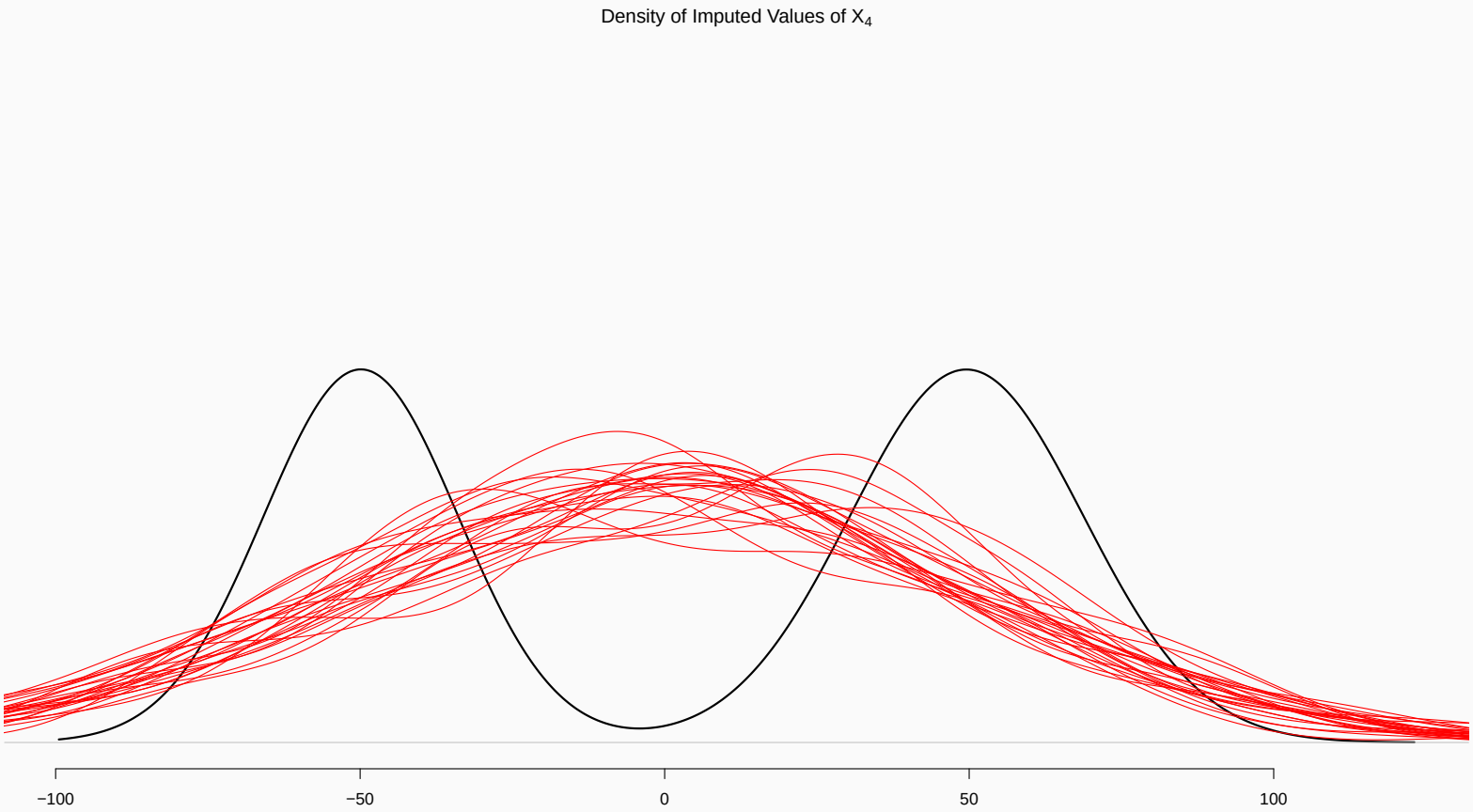

Figure 9: The distribution of imputed values of X_4 from Amelia II.

Imputation of Data From a Complex Distribution.

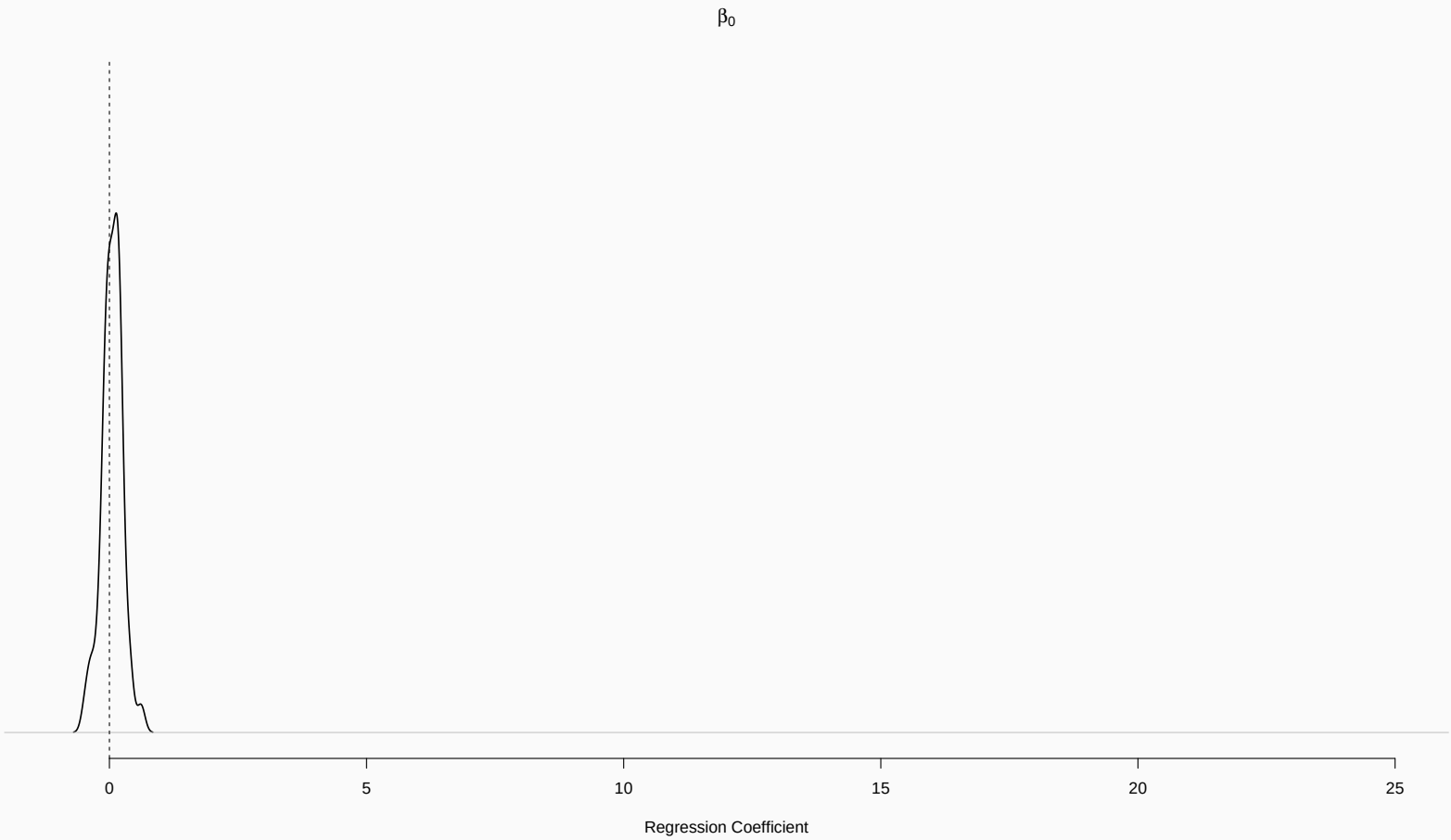

Figure 10: The distribution of β_0 after imputation with MH-GAIN.

Imputation of Data From a Complex Distribution.

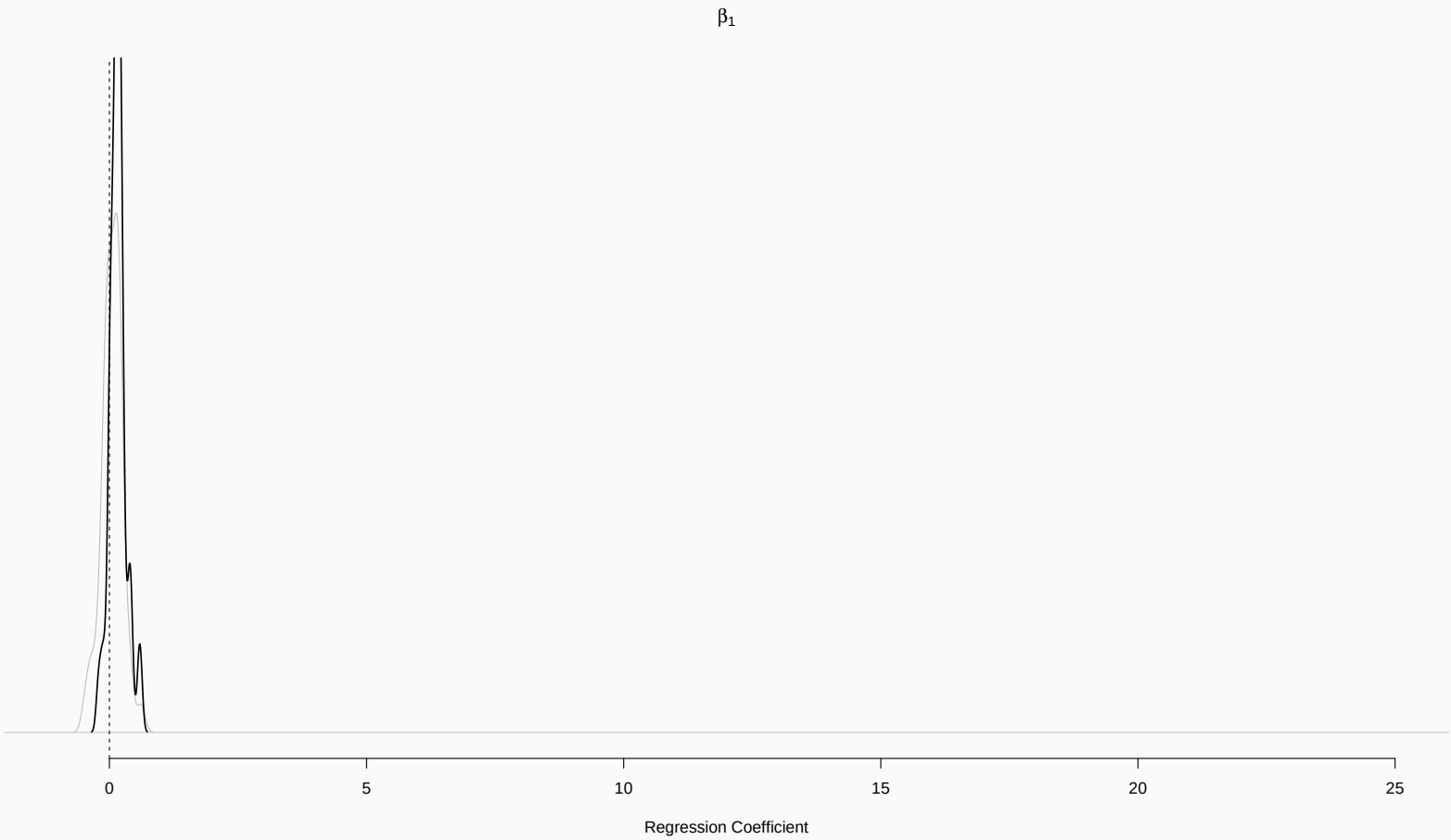

Figure 11: The distribution of β_1 after imputation with MH-GAIN.

Imputation of Data From a Complex Distribution.

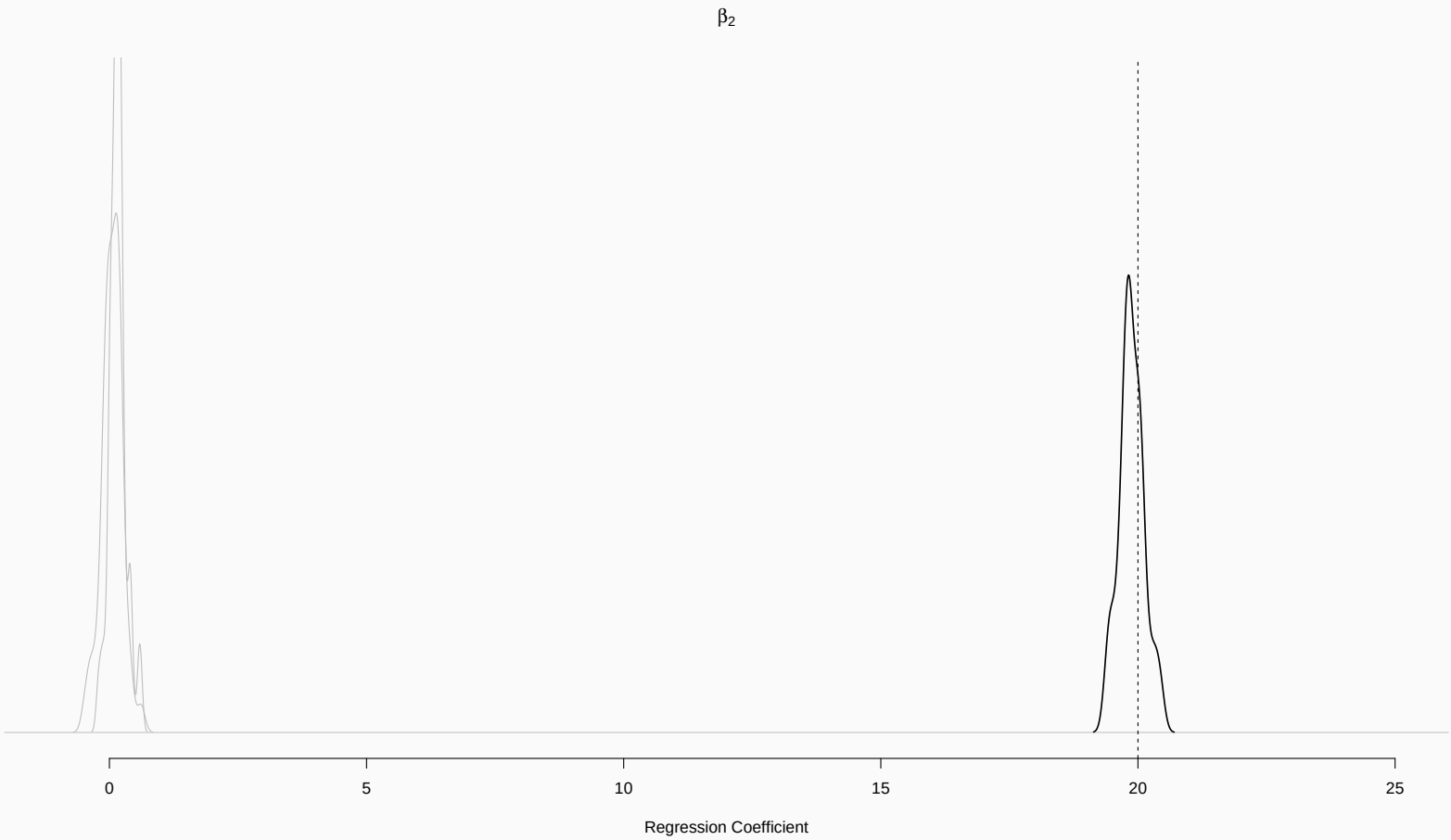

Figure 12: The distribution of β_2 after imputation with MH-GAIN.

Imputation of Data From a Complex Distribution.

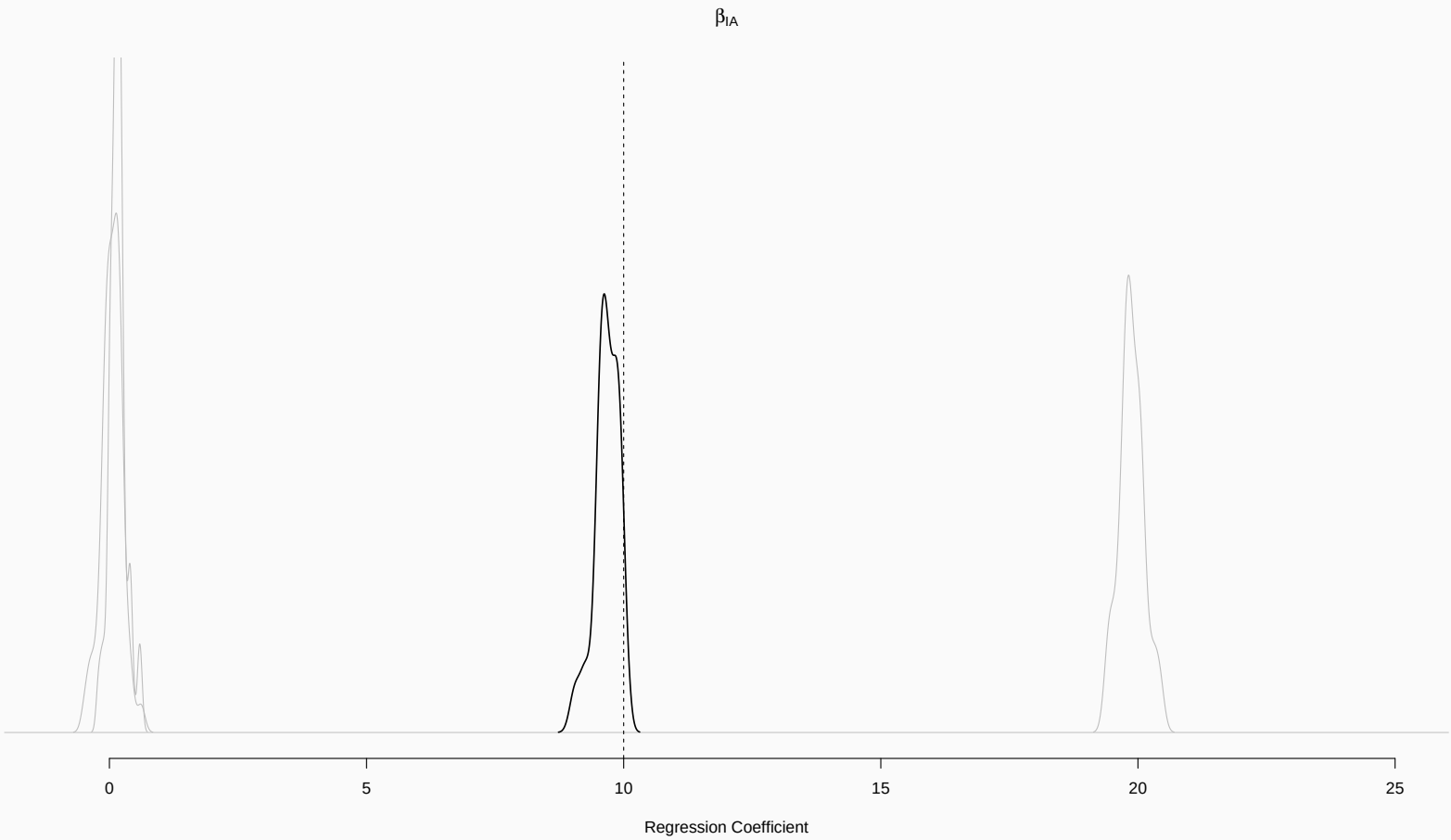

Figure 13: The distribution of β_{IA} after imputation with MH-GAIN.

Imputation of Data From a Complex Distribution.

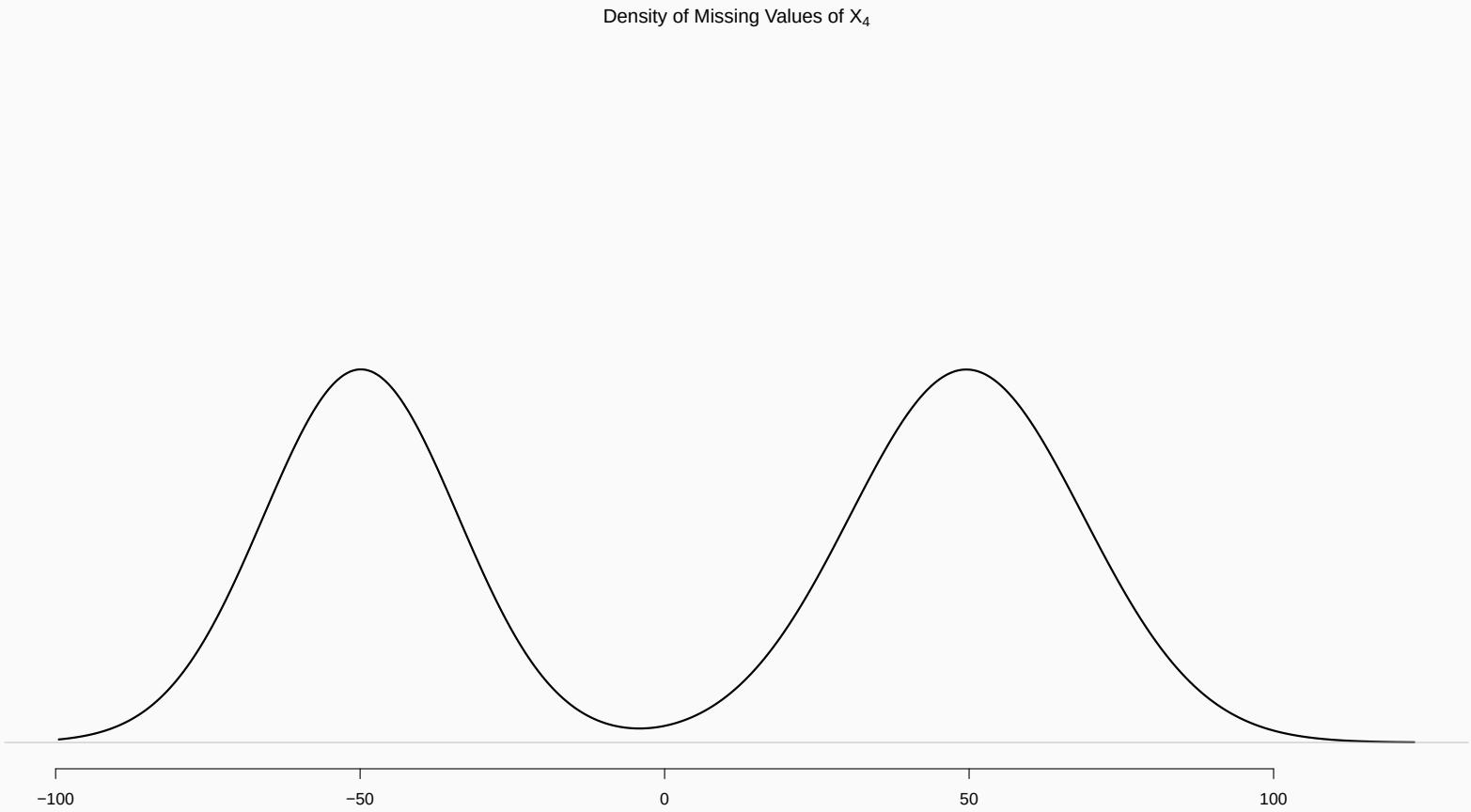

Figure 14: The distribution of missing values of X_4.

Imputation of Data From a Complex Distribution.

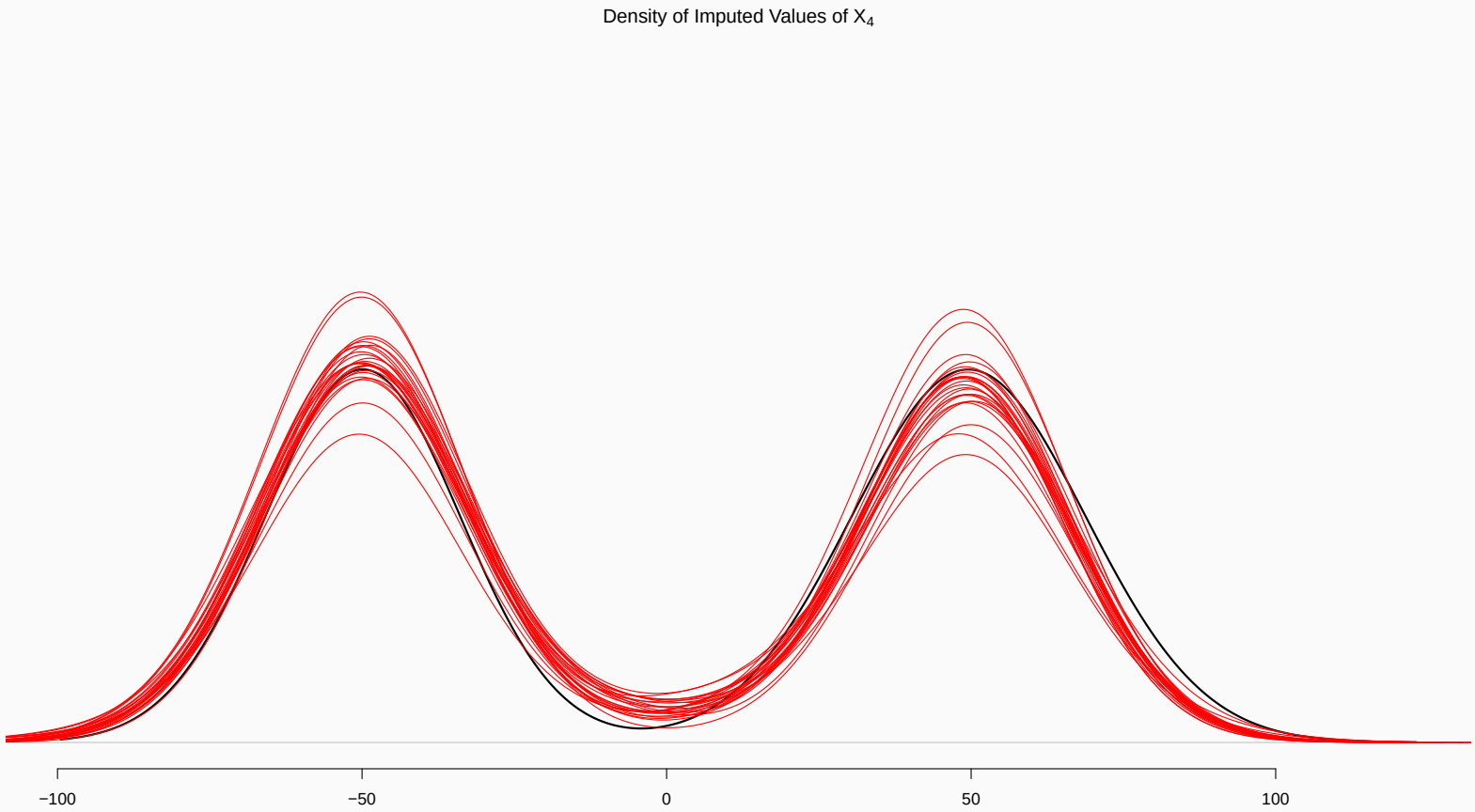

Figure 15: The distribution of imputed values of X_4 from MH-GAIN.

Application of MH-GAIN to Actual Data.

Application of MH-GAIN to Actual Data.

- Finally, I apply MH-GAIN to the study “Economic Inequality and Democratic Support” by Krieckhaus et al. (2014).
- Hollenbach et al. (2018) used the same study to benchmark the copula approach for multiple imputation.
- The data is drawn from the World Values Survey with a high share of missing values.
- The authors want to explain “Democratic Support” using Inequality at the country level and Income at the individual level as main explanatory variables.

Application of MH-GAIN to Actual Data.

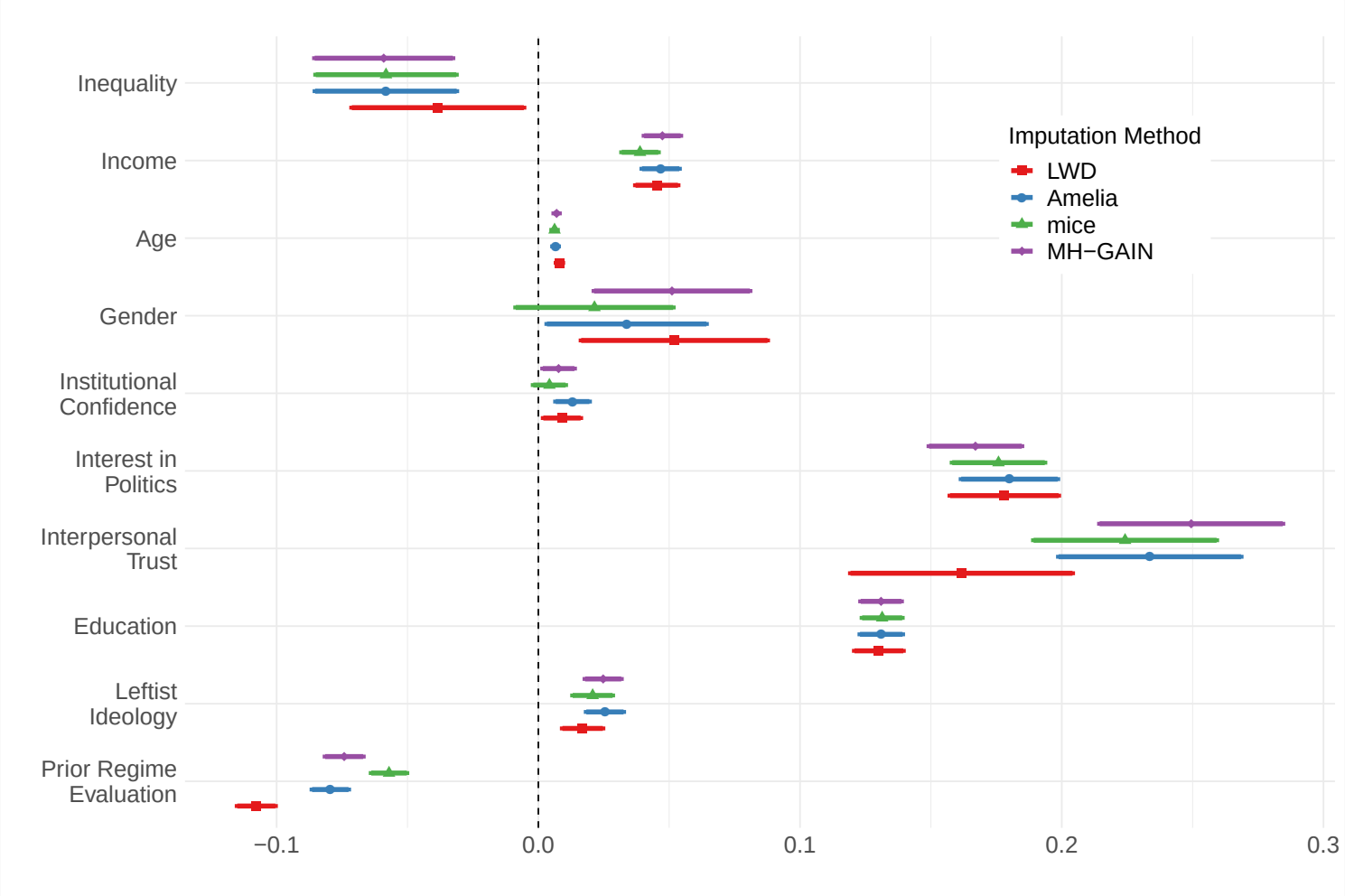

Figure 16: Comparison for the Krieckhaus et al. 2014 data.

Application of MH-GAIN to Actual Data.

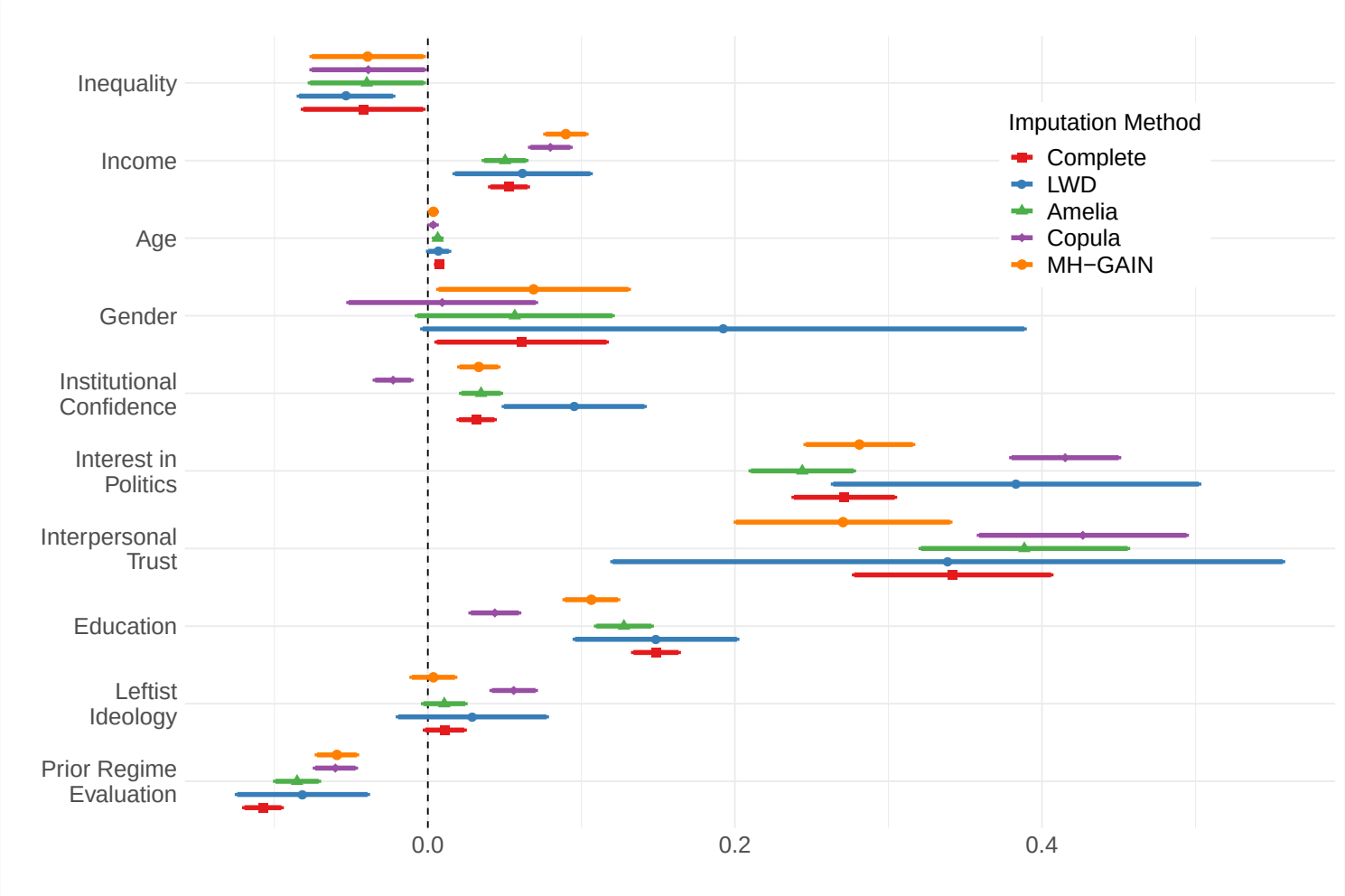

Figure 17: Comparison for the Krieckhaus et al. 2014 data.

Wrap-up and Outlook.

How Multiple Imputation Can Profit from Deep Learning.

- I implemented and extended GANs for multiple imputation.
- MH-GAIN is less restrictive in its assumptions than current Multiple Imputation approaches.
- MH-GAIN decreases Researcher Degrees of Freedom, e.g. interactions do not need to be specified ahead of the imputation procedure.

MH-GAIN for Time-Series Data.

- To make MH-GAIN work for time-series data the neural networks in the Generator and Discriminator(s) have to account for the time-dependent structure of the data.
- Neural network architectures for time-series data include so called recurrent neural networks, or long short-term memory modules (LSTM), that are also popular in Natural Language Processing.
- For time-series data I changed the networks in the Generator and Discriminators to include bi-directional LSTM modules.
- First experimental results are very promising.

Open Questions.

- When to stop GAIN training?
- Can we learn something from how other iterative Multiple Imputation algorithms stop early?
- Your questions!

Thank you for your attention!

Questions?

References

Arel-Bundock, V. and Pelc, K. J. (2018). When Can Multiple Imputation Improve Regression Estimates? *Political Analysis*, pages 1–6.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative Adversarial Nets. *Advances in Neural Information Processing Systems 27*, pages 2672–2680.

Hollenbach, F. M., Bojinov, I., Minhas, S., Metternich, N. W., Minhas, S., Ward, M. D., and Volfovsky, A. (2018). Principled Imputation Made Simple: Multiple Imputation Using Gaussian Copulas. *Sociological Methods & Research*, pages 1–25.

References ii

Honaker, J. and King, G. (2010). What to Do About Missing Values in Time-Series Cross-Section Data. *American Journal of Polictical Science*, 54(2):561–581.

Honaker, J., King, G., and Blackwell, M. (2011). AMELIA II : A Program for Missing Data. *Journal Of Statistical Software*, 45(7):1–54.

King, G., Honaker, J., Joseph, A., and Scheve, K. (2001). Analyzing Incomplete Political Science Data: An Alternative Algorithm for Multiple Imputation. *American Political Science Review*, 95(1):49–69.

Krieckhaus, J., Son, B., Bellinger, N. M., and Wells, J. M. (2014). Economic inequality and democratic support. *Journal of Politics*, 76(1):139–151.

References iii

Kropko, J., Goodrich, B., Gelman, A., and Hill, J. (2014). Multiple imputation for continuous and categorical data: Comparing joint multivariate normal and conditional approaches. *Political Analysis*, 22(4):497–519.

Lall, R. (2016). How multiple imputation makes a difference. *Political Analysis*, 24(4):414–433.

Pepinsky, T. B. (2018). A Note on Listwise Deletion versus Multiple Imputation. *Political Analysis*.

Rubin, D. B. (1978). Multiple imputations in sample surveys - A phenomenological Bayesian approach to nonresponse. In *Proceedings of the Survey Research Methods Section of the American Statistical Association*, pages 20–28.

References iv

Turner, R., Hung, J., Saatci, Y., and Yosinski, J. (2018). Metropolis-Hastings Generative Adversarial Networks. (NeurIPS):1–10.

Van Buuren, S. (2012). *Flexible imputation of missing data*. Chapman and Hall/CRC.

Van Buuren, S. and Groothuis-Oudshoorn, K. (2011). Multivariate Imputation by Chained Equations. *Journal Of Statistical Software*, 45(3):1–67.

Yoon, J., Jordon, J., and van der Schaar, M. (2018). GAIN: Missing Data Imputation using Generative Adversarial Nets.