UNITED NATIONS ECONOMIC COMMISSION FOR EUROPE

CONFERENCE OF EUROPEAN STATISTICIANS

Expert meeting on Statistical Data Confidentiality

26–28 September 2023, Wiesbaden

Experiments on Federated Data Synthesis

Claire Little, Mark Elliot, Richard Allmendinger (University of Manchester, UK)

claire.little@manchester.ac.uk

Abstract

Federated Learning (FL) is a decentralized approach to statistical model training in which training is performed across multiple clients to produce a global model. This approach can be used where multiple sites have data but do not have enough data to generate the required statistical power and cannot for legal, commercial or ethical reasons share their data. One paradigm case is randomized control trials for rare diseases. With FL, training data stays with each local client and is not shared or exchanged with other clients, so the use of FL can reduce privacy and security risks (compared to methods that pool multiple data sources) while addressing data access and heterogeneity problems. This study explores the feasibility of using FL to generate synthetic microdata, allowing multiple organizations to contribute to the construction of combined synthetic datasets (possibly for wider release) without the need to share or distribute their own data. The primary issue is whether it is possible in principle to produce good enough quality synthetic data and the study here focuses on this as a proof of concept before going on to discuss the issue of risk measurement. The results show that the approach is feasible and crucially in the main experiment the synthetic datasets better represented the full population than random samples of that population do. However the experiments are on toy datasets and the next step is to expand the dataset size.

1 Introduction

To enable the safe release of data, Statistical Disclosure Control (SDC) methods (Hundepool et al., 2012) can be applied to remove or alter disclosive information. Data synthesis (Rubin, 1993; Little, 1993) is an alternative to SDC which uses models of the original dataset to generate artificial data with the same structure and statistical properties as the original but (in the case of full synthesis) not containing any of the original data.
In this study, we explore the feasibility of federated synthesis, allowing multiple organizations to contribute to the construction of combined synthetic datasets (possibly for wider release) without the need to share or distribute their own data. The primary issue is whether it is possible in principle to produce good enough quality synthetic data and the study here focuses on this as a proof of concept before going on to discuss the issue of risk measurement.
The next section will present background information on Data Synthesis and Federated Learning, Section 3 outlines the methodology, Section 4 provides the results of our experiments, Section 5 discusses the results and their implications and final thoughts and ideas for future work an be found in Section 6.

2 Background

2.1 Data Synthesis

Data Synthesis (Rubin, 1993; Little, 1993) is an alternative to SDC and uses models built using the original dataset to generate artificial data with the same structure and statistical properties as the original but (in the case of full synthesis) not containing any of the original data. Synthetic data may be used where access to the original data is not possible or restricted due to privacy constraints. For example, the approval process to acquire access to safeguarded data can be lengthy, potentially delaying research; in these situations synthetic data can allow researchers to test code or plan analysis whilst awaiting access. Synthetic data may also be used to augment (add more records to) existing datasets.
There is an increasing number of techniques to generate synthetic data, including statistical methods (such as Nowok et al. (2016); Zhang et al. (2017)), and deep learning (DL) methods based on neural networks (NN) such as Generative Adversarial Networks (GANs) (Goodfellow et al., 2014), variational autoencoders (VAE) (Kingma and Welling, 2014), large language models (Radford et al., 2019), diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020), and genetic algorithms (GAs) (Chen et al., 2017, 2018).

2.2 Federated Learning

Federated Learning (FL) (McMahan et al., 2017) is a method that allows multiple clients (or devices) to collaboratively build a shared model without the clients transmitting or exchanging their raw data. In the context of synthetic data, this could allow multiple clients (organisations, users, etc.) to produce a shared synthetic dataset, without the need to share their own individual private data thereby minimising disclosure risk. It could allow the linkage of datasets that would otherwise be unlikely to be linked in the traditional sense, thereby producing opportunities to access unique synthetic data that is potentially more diverse, and richer, than each participant's synthetic dataset alone. This paper explores the feasibility of using FL together with a GA, to produce a combined synthetic dataset, which as far as we are aware has not been attempted so far.
The early focus of FL was its use on mobile and edge devices (e.g. Bonawitz et al. (2016); Konecny et al. (2016)), where an FL model could have many massively distributed clients, each with potentially different computational capabilities, limited communication and unbalanced data. An example of its usage is Google's Gboard (keyboard) application which trains a model on each mobile device (when it is idle) using the local data and then sends only model updates (parameters) to the server; this allows it to predict the next word when typing, suggest emojis and discover new words (McMahan and Thakurta, 2022). As described by Kairouz

et al. (2021) interest has increased in the use of FL for other (non-mobile) applications, such as allowing cross-organisational collaboration to train models. For example, in healthcare, sensitive data is difficult to access and tightly regulated, making sharing/pooling data (between institutions) prohibitive – FL can allow the creation of more robust models, trained on a larger and more diverse pool of data (than a single institution could provide), without the need to exchange or centralise sensitive medical data (Rieke et al., 2020; Kumar and Singla, 2021). FL has generally been used to produce shared models (such as predictive models) collectively trained on each clients data. A central server controls the process but does not access any of the client data. NN based methods are typically used, where each client receives the current model weights from the central server, trains the model on their own data and then sends the model weights (or parameters) back to the server. All the clients' weights are then aggregated (typically using the FedAvg, or Federated Averaging algorithm (McMahan et al., 2017)) by the server which updates the global shared model. The model is then sent back to the clients and the process continues until some stopping condition is met.
There is a small body of research into the use of FL to generate synthetic data. We use microdata for this study and therefore focus on methods designed for tabular data (i.e. structured data comprising rows and columns containing mixed-type features, such as categorical and numerical). Duan et al. (2023); Fang et al. (2022); Zhao et al. (2021) use GAN-based methods to generate synthetic data, with a GAN training on each client and each sending the model weights to the server to aggregate, etc. (each client generates the final synthetic data individually using the shared model). Weldon et al. (2021) use a GAN on the clients and on the server, but differs in that the server GAN generates the final synthetic dataset. Lomurno et al. (2023) present a different method, using VAE, with each client training a data generator locally. The clients send their models (generators) to the server, but they are not aggregated or combined (as is typical in FL). In the final phase, each client can access the set of generators (from all clients) stored on the server and use some or all of these to generate their own synthetic data. Here we use a GA to generate synthetic data on the server, which is then sent to the clients who each calculate the fitness (utility) score, then send it back to the server where all client scores are combined and used to create the next generation of synthetic datasets. Qu et al. (2020) also generate synthetic data on the server, which is sent to the clients to evaluate, but this employs a GAN-based method, uses image data and focusses on the use-case where clients are temporary (i.e. they may not be available for the whole process).

2.3 Study Aims

In this study, our objective is to assess the feasibility of using a federated learning to generate a combined synthetic dataset. The research questions are as follows.

RQ1: Can a federated synthesis model reproduce the joint distribution of combined distributed datasets?
RQ1.1: What information does the server need to be able to reproduce that joint distribution?
RQ2: Is the utility of a synthesised combined dataset at least as good as that of the samples held by each client.

3 Methodology

The study is simulation of a server and two clients. The basic simulation scenario is that the sever generates synthetic data, which is then sent to the clients, who each calculate the similarity of the synthetic data to the sample that they hold, then send those similarity scores back to the server where all client scores are combined and used to create the next generation of synthetic datasets.
Our machine learning model of choice is genetic algorithms. In Section 3.1, we describe GAs and motivate this choice, then in Section 3.2 we describe the data that we use and how it was set up for the simulation.

TABLE 1. Simple binary original dataset with ten rows, sampled from UK 1991 Census data, which was split into two five-row datasets, for clients A and B.

AGE	MSTATUS	SEX	LTILL	TENURE	client
1	2	2	2	2	A
1	1	1	2	2	A
1	1	2	2	2	A
2	2	2	2	1	A
1	1	1	2	1	A
2	2	2	2	1	B
1	2	1	2	1	B
1	1	2	2	1	B
1	1	2	1	2	B
1	1	1	2	1	B

3.1 Genetic Algorithms

Genetic Algorithms (GAs) (Holland, 1992) perform iterative optimisation. There are three main (biologically inspired) operators: selection (parental and environmental), crossover, and mutation. Broadly speaking, an initial population of candidate solutions is specified (in this case, a candidate solution is a synthetic dataset), and the fitness (the utility) of the candidates is calculated. The parental selection operator is used to select candidates (parents) to reproduce for a new population, with fitter candidates more likely to be selected. A crossover operator combines some of the parents (there are a variety of methods for this) to produce new candidate solutions (children). A mutation operator then mutates some of the candidates (i.e. randomly changes some of the features). The children or a combination of children and parents form the population of the next generation (this step is called environmental selection). This process is repeated multiple times (generations), using the fitness to guide it, with ideally fitter solutions produced with each generation. Commonly, the process terminates when a specified number of generations has been produced or a particular fitness level has been reached.

GAs are flexible in that there are many parameters that can be changed or set, and the fitness function can be designed for the specific purpose. Work by Chen et al. (2017, 2018) has shown the feasibility of using GAs to generate synthetic microdata, and demonstrated the viability of using risk and utility as conflicting objectives (Chen et al., 2019). More recently, Thogarchety and Das (2023) used a GA approach to produce synthetic data to augment class imbalanced datasets and Liu et al. (2023) presented a GA method that generates synthetic data capable of approximating a range of statistical queries.

3.2 Data

A (very) small binary dataset was used, which was randomly split into two datasets (of equal sizes) to represent two clients named client A and client B. The UK 1991 Census (University of Manchester and ONS, 2013) microdata was used, with 10 rows randomly sampled (from the same geographic area). Table 1 displays the data, with five variables (respectively: age, marital status, sex, long-term illness, and housing tenure) which were all converted to binary (using values of 1 and 2). This is called the original data set. It was randomly split into two five-row datasets, one representing client A and one client B, these are identified in Table 1.

TABLE 2. Parameters that were fixed in the experiments.

Parameter	Type	Value chosen	Further details
No. of clients	Simulation	VARIES	-
Initial Metadata sent by clients	Simulation	Univariates	-
Combination of client scores	Simulation	VARIES	-
No. of objectives for GA	Simulation	1	Similarity (utility)
SDC applied to the output sent to server	Simulation	None	-
Output passed to client by server	Simulation	VARIES	-
Population size	Model	50	-
Parental selection	Model	Binary tournament	k=2
Mutation rate	Model	0.05	-
Crossover Operator	Model	None	-
Environmental selection	Model	Elitism	-
No. of generations	Experiment	150	-
Choice of Dataset	Experiment	UK Census microdata	1991
No. of rows (per client)	Experiment	VARIES	-
No. of variables	Experiment	5	-
Type of variables	Experiment	Binary	-
No. of runs	Experiment	5	-

TABLE 3. Parameters varied by experiment.

Parameter	Value		
	Experiment 1	Experiment 2	Experiment 3
No. of clients	1	2	2
Combination of client scores	N/A	None	Mean
Output passed to client by server	Synthetic clients dataset	Synthetic clients dataset	Synthetic combined dataset
No. of rows (per client)	10	5	5

3.3 Method and Parameters

The potential range of variation in the simulation is huge. There are three types of parameters that could be varied in the study design:

Model Parameters: These are changeable settings for the GA (e.g., mutation rate)
Simulation Parameters: These are variations in the scenario being presented (e.g., number of clients)
Experimental Parameters: These are elements of the study design that are not part of the simulation itself (e.g., number of runs, data choices).

A set of these is shown in Table 2. For proof-of-concept experiments we have chosen one value of most of these parameters; a much simpler set than might be used in practice. As well as using a very small sample (of real data), we kept the model complexity low. This simplicity assists us with the interpretation of the results. We have varied four of the parameters across three experiments. These are shown in Table 3.

The first two experiments are used to establish a baseline. In experiment 1 we have just a single client. In effect, this tests whether a GA can reproduce the original data when unencumbered by the distributed data. In experiment 2 we split the data across two clients, but the server has a separate interaction with each client and then is deemed to combine the data at the end. This is in effect a minor variation on experiment 1. Experiment 3 is the main experiment, and we now describe what is simulated in more detail.

The experiment 3 simulation is represented graphically in Figure 1. The GA runs on the central server, and at the start of the process (labeled Initialisation, in the figure) each of the clients sends metadata about their individual data to the server. At the most basic, the server would need to know the variable names and the size (how many records) of the data. It is expected that the clients will agree in advance on the variables to be included. For this experiment, the clients send the univariate distributions (this information is used by the GA to mutate the data) and the number of records in each dataset.

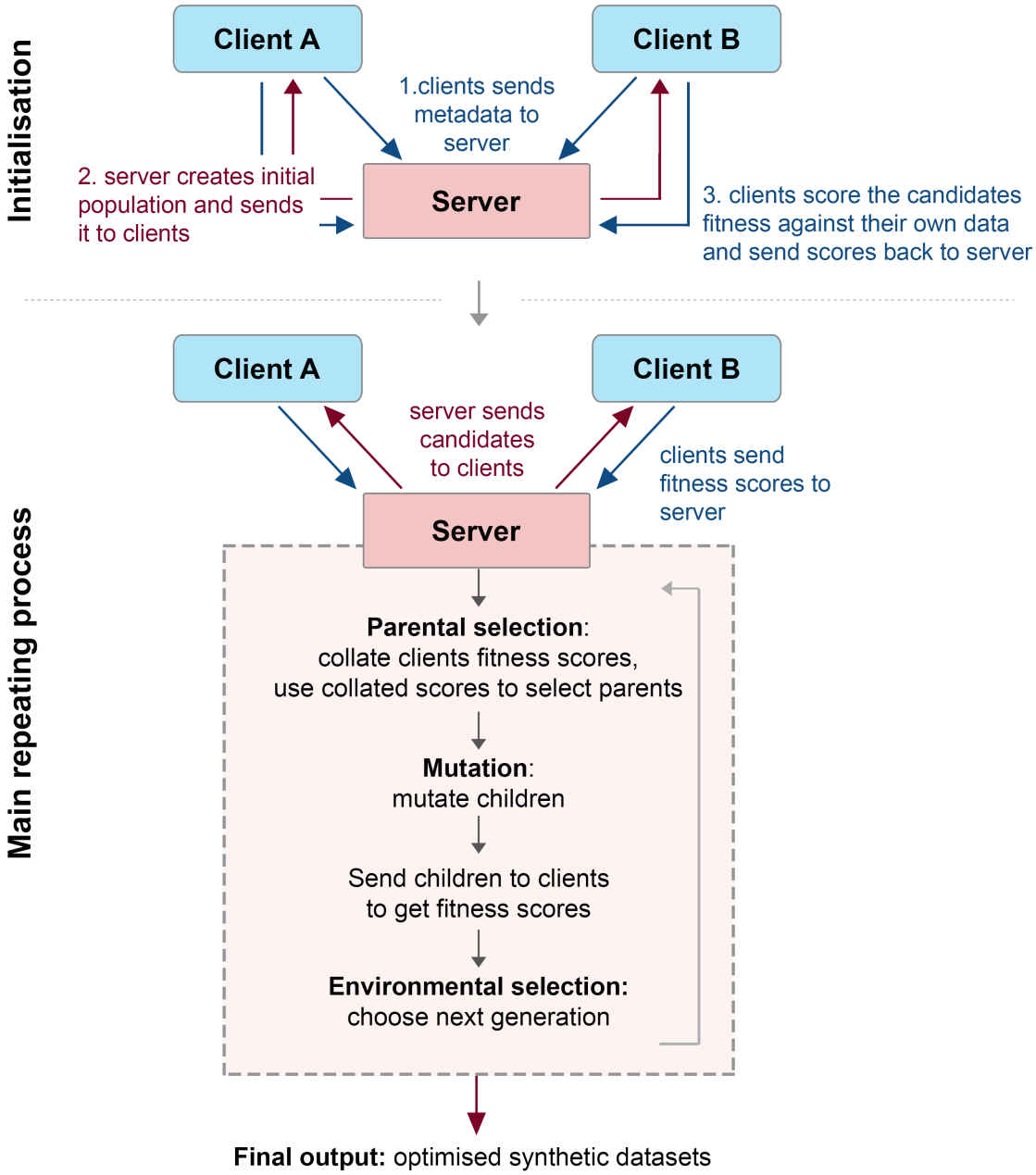

FIGURE 1. An illustration of the federated synthesis simulation used for Experiment 3, with a server and two clients.

The server then combines the distributions of each client by taking the average to calculate a combined distribution. An initial population of synthetic datasets (candidates) is generated; these are drawn from the

uniform distributions of the five variables. The datasets in the initial population have the same number of records as both combined client data would have and the same variables.
For this simple model, only one objective is assessed by the client, which is the similarity of the multivariate distribution of the clients data and each of the synthetic datasets passed by the server to the client. They then send those similarity scores back to the server. In detail the similarity measure calculates the proportion of every combination of values in the synthetic (candidates) and client data, then takes the mean of the absolute differences. This is then subtracted from 1 so that the similarity score takes a value between zero and one (where 0 indicates no similarity and 1 indicates an identical distribution).
Once the server receives each of the client scores, it calculates the mean to produce an overall score for each synthetic dataset. In experiment 3 the server simply averages the clients scores, but they could be combined in other ways (e.g. using the lowest or the highest, or weighted by how similar each of the clients scores are to each other). This completes the initialisation phase.
The main repeating process involves selection and mutation, but not crossover; this was excluded to reduce complexity. Firstly, parents are selected from the population using tournament selection (two synthetic datasets are randomly selected and the one with the highest similarity score wins). Two parents produce two children (i.e., two datasets), which are simply copies of themselves (where crossover is used, the children may be a combination of the parents), and the same amount of children are produced as the population size. Then, each child is mutated with a probability equal to the mutation rate (0.05), with the replacement value being drawn from the relevant univariate distribution. The children are then sent to the clients, who score the similarity and send the scores back to the server. These scores are then aggregated by the server. Finally elitism is used to select the next generation, that is the best (those with highest similarity) of the parents and the children are retained for the next generation (e.g. if a parent has higher similarity than the child, the parent is retained). This process is repeated for a set number of generations.

4 Results

Each experiment was repeated five times (using different random seeds). The plots in Figures 2-4 give the mean similarity score across the population for each of the generations for which the GA was run.
Figure 2 shows the results for experiment 1. For all runs the GA converged (that is, the synthetic datasets reproduced the original). The goal would not generally be to reproduce the original dataset, but this demonstrates that the GA works (albeit on a very small dataset). In experiment 2, the GA was run separately on both clients five-row datasets, with the results plotted in Figure 3. Each of the five runs converged to one (that is, all runs reproduced the clients data), and so each of the clients resulting data could be combined to reproduce the original dataset.
The results for experiment 3 (as described in Figure 1) are illustrated in Figure 4. Panels 1 and 2 illustrate the scores calculated by clients A and B, these individual scores are sent to the server which aggregates them, as displayed in panel 3. The aggregated score is what drives the GA (and clients A and B do not see each other's scores, they only communicate with the server).
The plot highlights that the synthetic datasets generated in run 3.3 scored highly with client A, but scored poorly with client B, however, when aggregated by the server, all five of the runs look remarkably similar. Run 3.3 is interesting as until about generation 20 the gradient is very similar on both clients to the other runs but around that generation a bifurcation happens. This appears to be the result of the process falling into a local optimum in which client A's dataset was optimised at the expense of client B's. This was the result of some mutation around generation 20 (subsequent test runs with the same starting seed failed to reproduce this result).
Panel 4 shows the similarity scores of the synthetic datasets produced at each generation against the real combined data – by definition this would not be possible in a real-life scenario since the original data would not be available, but this is calculated post hoc to evaluate how the overall model is working (i.e., we are more interested in whether the server is reproducing the overall dataset than whether it replicates individual client

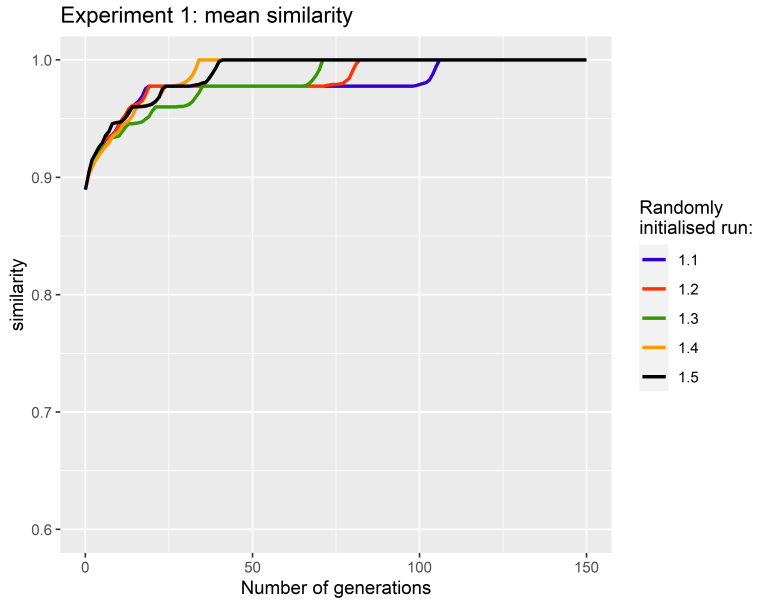

FIGURE 2. Experiment 1, the mean (n=50) similarity of five randomly initialised runs of the GA on the original ten-row dataset. Note the truncated y axis.

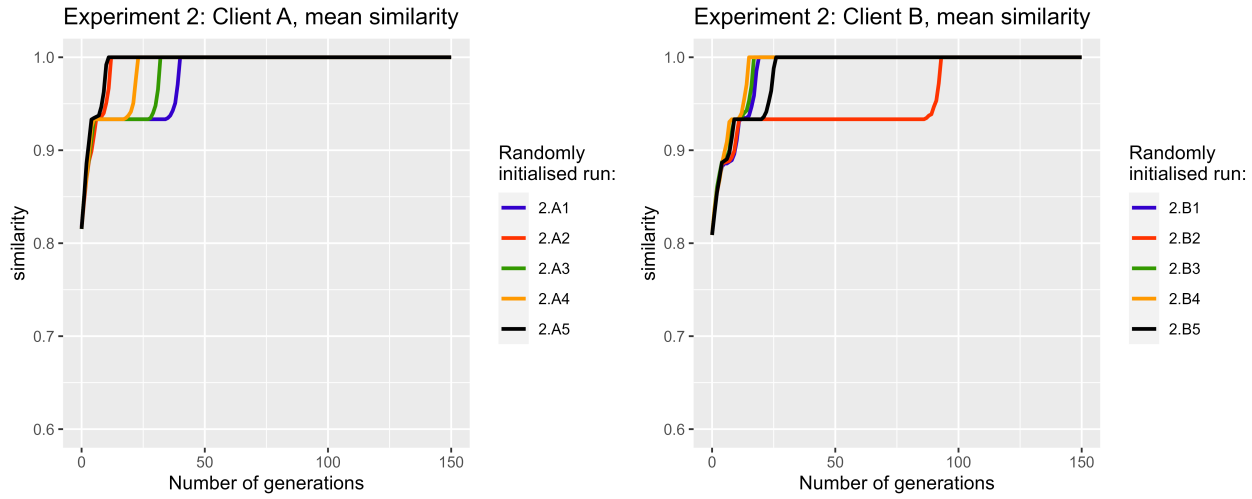

FIGURE 3. Experiment 2, the mean (n=50) similarity of five randomly initialised runs of the GA on the five-row datasets of client A and client B. Note the truncated y axis.

distributions). Panel 4 shows that for all but run 3.3 the model converges on the original data, that is, each run reproduces the original dataset. This is a particularly fascinating finding as it has done this despite the evaluations from the clients indicating sub-optimality. The baseline is included to indicate the combined client to server data similarity.

Panel 3 (Figure 4) illustrates that (at least in this example) it may be difficult for the server to determine how well the overall model is performing. Other methods of combining the client scores as variations on experiment 3 were also considered (minimum, weighted, and alternating). The results are shown in the Appendices.

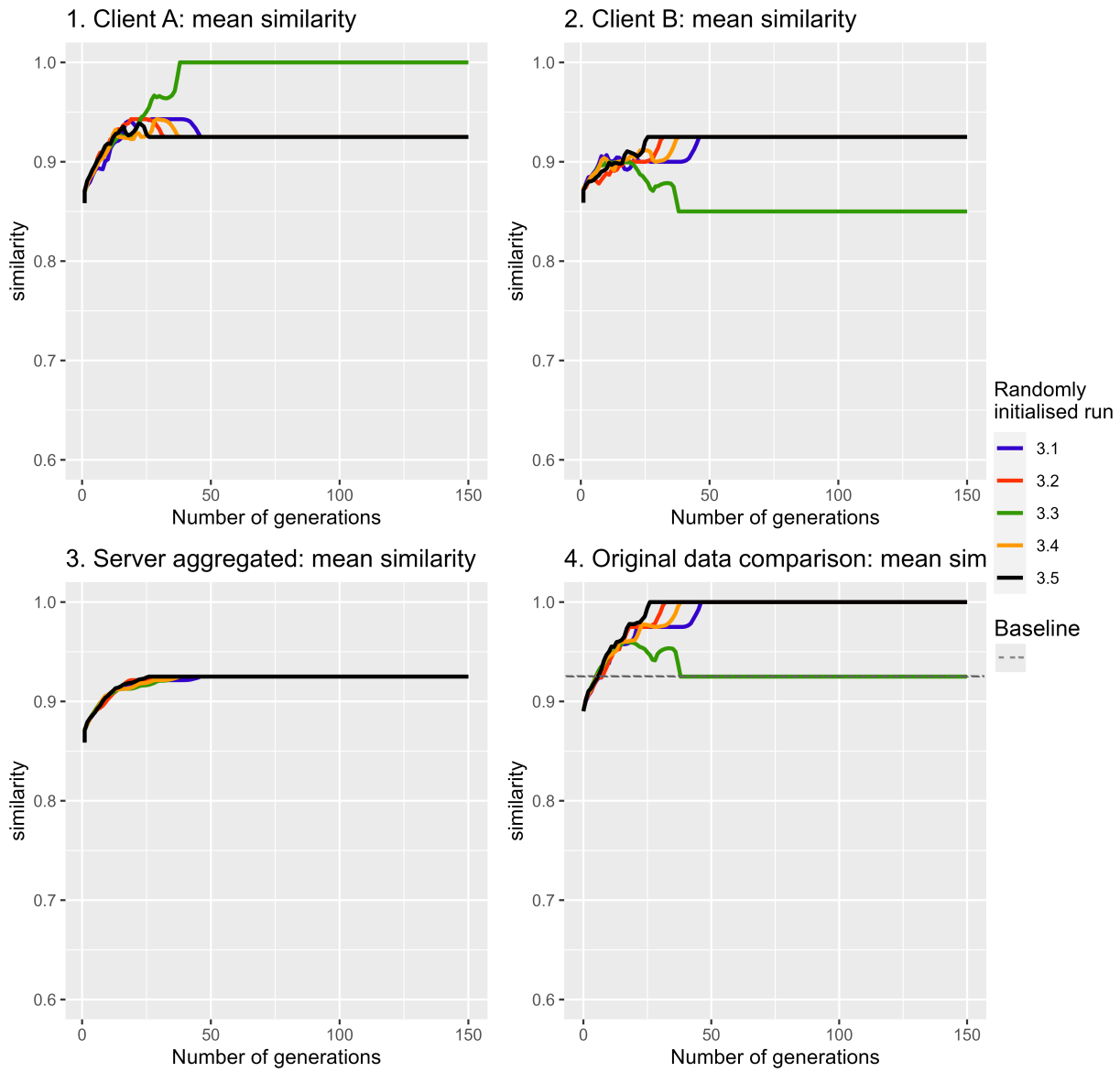

FIGURE 4. Experiment 3, the mean (n=50) similarity scores of five randomly initialised runs of the server GA, showing client A (1), client B (2), the server aggregated scores (3) and the comparison against the original data (4). Note the truncated y axes.

5 Discussion

The results of experiment 3 demonstrates our proof of concept. 4 out of 5 of the runs reproduced the original data. Fascinatingly they did this despite the mean evaluations scores from the client indicating that the operation had not achieved unity. This however was simple a reflection of the clients own sample not fully representing the combined datasets structure. Thus the synthetic datasets were a better representation of the 'real' combined dataset than the 'samples' held by each client. This *emergent reproducibility* shows how the approach could deliver the desired outcome to produce analytically useful datasets synthesised across distributed datasets.
The experiments reported here focused on the single objective of utility, and in this case the goal was to reproduce the original data. In a real-life scenario, there would also be a consideration of risk – reproducing

the original data would not be desirable. A way to incorporate risk would be to use a multi-objective approach within the GA, and explore options such as Pareto optimality. The flexibility of GAs means that different utility and risk measures could be easily added. Another angle would be using deep learning methods (such as GANs and diffusion models) and adapting them to multi-objective optimisation (GANs are already used widely within FL).
The fact that in our experiments it was not clear on the server that the original data had been reproduced might be thought of as useful, in terms of disclosure risk, but it would also mean that in this mechanism we could not rely on severe side restraint to manage risk.
An obvious important expansion of these experiments is to test the method on larger and more complex datasets. Firstly, we need to establish if the emergent reproducibility effect scales. Also, for very large datasets, it simply may not be practical to send the entire population of datasets at each generation, and so alternatives may need to be explored. Another important expansion is to examine the effect of having more than two clients. The flexibility of the method also means that there are many parameters that can be experimented with.

6 Conclusion

The purpose of this study was as a proof of concept to determine whether using FL together with a GA to produce synthetic data was feasible. We have shown that it is feasible, albeit with a very small dataset, and with the focus being only synthetic data utility. The results are promising and there are many areas of future work including testing this on larger, more complex datasets, using a multiobjective approach that incorporates risk, and experimenting more generally with the various parameters.

References

Bonawitz, K. A., V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth (2016). Practical secure aggregation for federated learning on user-held data. In *NIPS Workshop on Private Multi-Party Machine Learning*. https://arxiv.org/abs/1611.04482.

Chen, Y., M. Elliot, and D. Smith (2018). The application of genetic algorithms to data synthesis: a comparison of three crossover methods. In *Privacy in Statistical Databases. PSD 2018*, pp. 160–171. Springer.

Chen, Y., M. J. Elliot, and J. W. Sakshaug (2017). Genetic algorithms in matrix representation and its application in synthetic data. In *UNECE Worksession on Statistical Confidentiality*. https://unece.org/fileadmin/DAM/stats/documents/ece/ces/ge.46/2017/2_Genetic_algorithms.pdf.

Chen, Y., J. Taub, and M. J. Elliot (2019). Trade-off between information utility and disclosure risk in ga synthetic data generator. In *Joint UNECE/Eurostat Expert Meeting on Statistical Data Confidentiality*. https://unece.org/fileadmin/DAM/stats/documents/ece/ces/ge.46/2019/mtg1/SDC2019_S3_UK_Chen_Taub_Elliot_AD.pdf.

Duan, S., C. Liu, P. Han, X. Jin, X. Zhang, T. He, H. Pan, and X. Xiang (2023). Ht-fed-gan: Federated generative model for decentralized tabular data synthesis. *Entropy 25*(1). DOI: 10.3390/e25010088.

Fang, M. L., D. S. Dhami, and K. Kersting (2022). Dp-ctgan: Differentially private medical data generation using ctgans. In M. Michalowski, S. S. R. Abidi, and S. Abidi (Eds.), *Artificial Intelligence in Medicine*, pp. 178–188. Springer International Publishing. DOI: 2022.10.1007/978-3-031-09342-5_17.

Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio (2014). Generative Adversarial Nets. In *Proceedings of the Advances in Neural Information Processing Systems*, Volume 27. https://papers.nips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

Ho, J., A. Jain, and P. Abbeel (2020). Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems 33*, 6840–6851. https://proceedings.neurips.cc/paper/2020/file/

4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.
Holland, J. H. (1992). *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press.
Hundepool, A., J. Domingo-Ferrer, L. Franconi, S. Giessing, E. Schulte Nordholt, K. Spicer, and P.-P. de Wolf (2012). *Statistical Disclosure Control*. Wiley series in Survey Methodology. John Wiley & Sons, Ltd. ISBN: 978-1-119-97815-2.
Kairouz, P., H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, et al. (2021). Advances and open problems in federated learning. *Foundations and Trends in Machine Learning 14*(1–2), 1–210. DOI: 10.1561/2200000083.
Kingma, D. and M. Welling (2014). Auto-encoding variational bayes. DOI: 10.48550/ARXIV.1312.6114.
Konecny, J., H. B. McMahan, F. X. Yu, P. Richtarik, A. T. Suresh, and D. Bacon (2016). Federated learning: Strategies for improving communication efficiency. In *NIPS Workshop on Private Multi-Party Machine Learning*. https://arxiv.org/abs/1610.05492.
Kumar, Y. and R. Singla (2021). *Federated Learning Systems for Healthcare: Perspective and Recent Progress*, pp. 141–156. Cham: Springer International Publishing. DOI:10.1007/978-3-030-70604-3_6.
Little, R. (1993). Statistical Analysis of Masked Data. *Journal of Official Statistics 9*(2), 407–426. https://www.scb.se/contentassets/ca21efb41fee47d293bbee5bf7be7fb3/statistical-analysis-of-masked-data.pdf.
Liu, T., J. Tang, G. Vietri, and Z. S. Wu (2023). Generating private synthetic data with genetic algorithms. DOI: 10.48550/arXiv.2306.03257.
Lomurno, E., A. Archetti, L. Cazzella, S. Samele, L. Di Perna, and M. Matteucci (2023). Sgde: Secure generative data exchange for cross-silo federated learning. In *Proceedings of the 2022 5th International Conference on Artificial Intelligence and Pattern Recognition*, pp. 205–214. Association for Computing Machinery. DOI: 10.1145/3573942.3573974.
McMahan, B., E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas (2017). Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pp. 1273–1282. PMLR. http://proceedings.mlr.press/v54/mcmahan17a/mcmahan17a.pdf.
McMahan, B. and A. Thakurta (2022). Federated learning with formal differential privacy guarantees. https://ai.googleblog.com/2022/02/federated-learning-with-formal.html, accessed 2023-05-24.
Nowok, B., G. Raab, and C. Dibben (2016). Synthpop: Bespoke creation of synthetic data in R. *Journal of Statistical Software 74*(11). DOI: 10.18637/jss.v074.i11.
Qu, H., Y. Zhang, Q. Chang, Z. Yan, C. Chen, and D. Metaxas (2020). Learn distributed gan with temporary discriminators. In A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm (Eds.), *Computer Vision – ECCV 2020*, pp. 175–192. Springer International Publishing. DOI: 10.1007/978-3-030-58583-9_11.
Radford, A., J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever (2019). Language models are unsupervised multitask learners. *OpenAI blog 1*(8), 9. https://d4mucfpksywv.cloudfront.net/better-language-models/language-models.pdf.
Rieke, N., J. Hancox, W. Li, F. Milletari, H. R. Roth, S. Albarqouni, S. Bakas, M. N. Galtier, B. A. Landman, K. Maier-Hein, et al. (2020). The future of digital health with federated learning. *NPJ digital medicine 3*(1), 119. DOI:10.1038/s41746-020-00323-1.
Rubin, D. B. (1993). Statistical Disclosure Limitation. *Journal of Official Statistics 9*(2), 461–468. https://ecommons.cornell.edu/bitstream/handle/1813/23033/rubin-1993.pdf?sequence=7.
Sohl-Dickstein, J., E. Weiss, N. Maheswaranathan, and S. Ganguli (2015). Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning*, Volume 37, pp. 2256–2265. https://proceedings.mlr.press/v37/sohl-dickstein15.html.
Thogarchety, P. and K. Das (2023). Synthetic data generation using genetic algorithm. In *2023 2nd International Conference for Innovation in Technology (INOCON)*, pp. 1–6. DOI: 10.1109/INOCON57975.2023.10101072.
University of Manchester and ONS (2013). Census 1991: Individual Sample of Anonymised Records for Great Britain (SARs). http://doi.org/10.5255/UKDA-SN-7210-1.

Weldon, J., T. Ward, and E. Brophy (2021). Generation of synthetic electronic health records using a federated gan. DOI: 10.48550/arXiv.2109.02543.

Zhang, J., G. Cormode, C. Procopiuc, D. Srivastava, and X. Xiao (2017). PrivBayes: Private data release via Bayesian networks. *ACM Transactions on Database Systems 42*(4). DOI: 10.1145/2588555.2588573.

Zhao, Z., R. Birke, A. Kunar, and L. Y. Chen (2021). Fed-tgan: Federated learning framework for synthesizing tabular data. DOI: 10.48550/arXiv.2108.07927.

A Using the worst client scores to drive the GA, rather than averaging

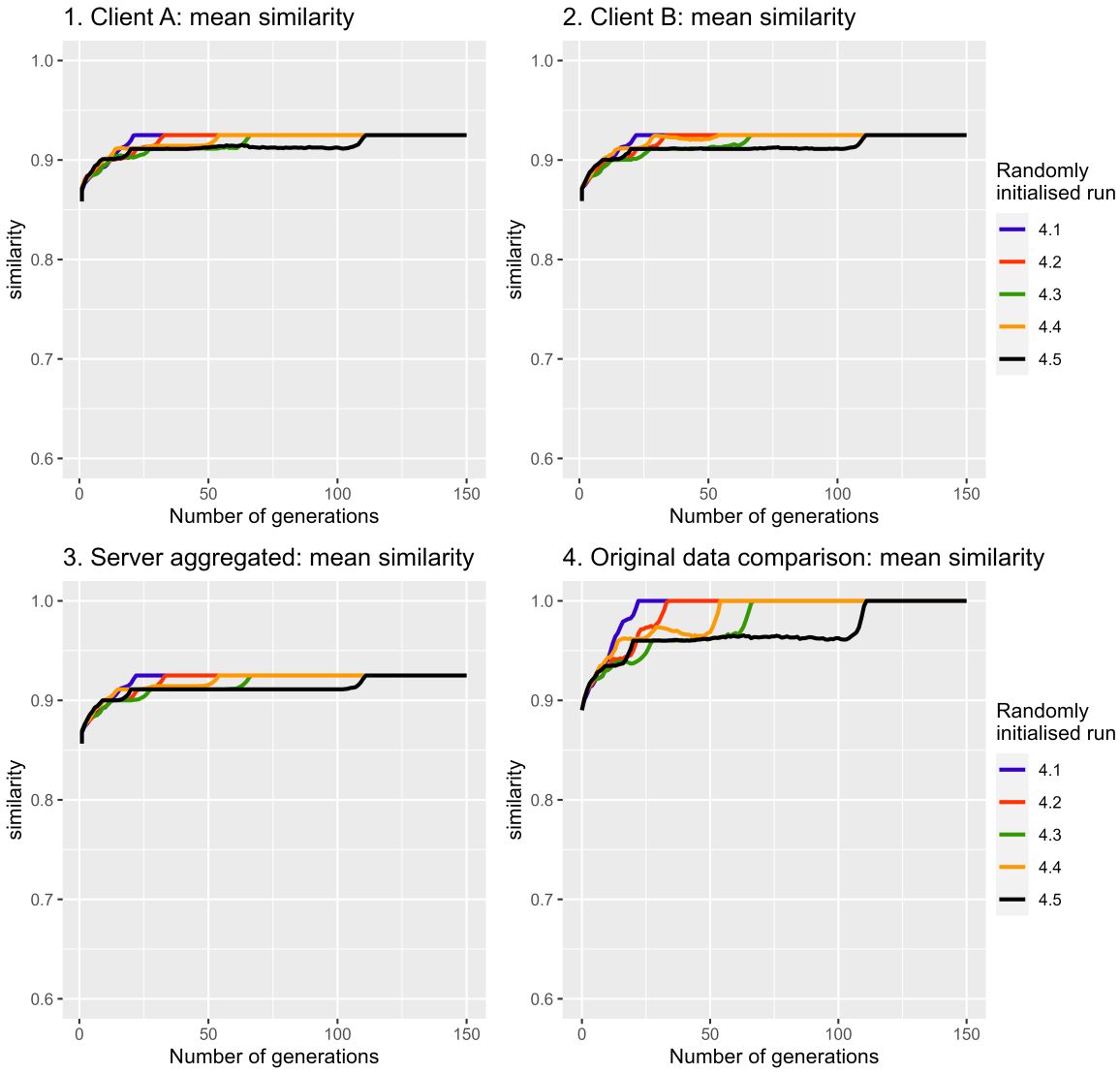

FIGURE 5. Mean (n=50) similarity scores of five randomly initialised runs of the server GA, where only the worst (lowest) client score is used to drive the GA (rather than averaging both client scores). Showing client A (1), client B (2), the worst scores (3) and the comparison against the original data (4). Note the truncated y axes.

B Using weighted averaged score to drive the GA

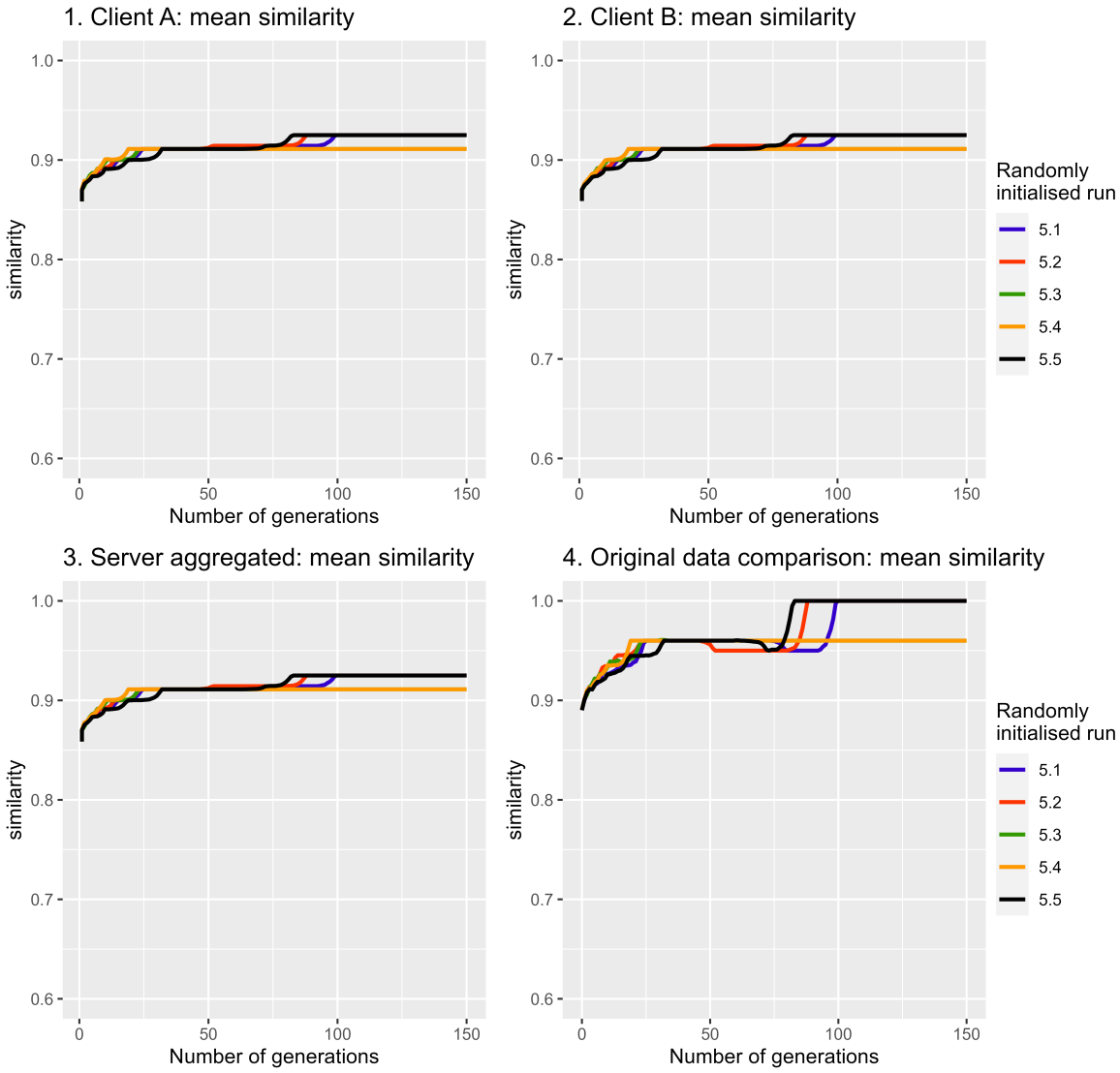

FIGURE 6. Mean (n=50) similarity scores of five randomly initialised runs of the server GA, where a weighted averaged score is used to drive the GA. Where the client scores are close (the clients agree) the scores are weighted higher, where they are far apart (the clients disagree) the scores are weighted lower. Showing client A (1), client B (2), the server weighted averaged scores (3) and the comparison against the original data (4). Note the truncated y axes.

C Alternating the client scores to drive the GA, rather than averaging

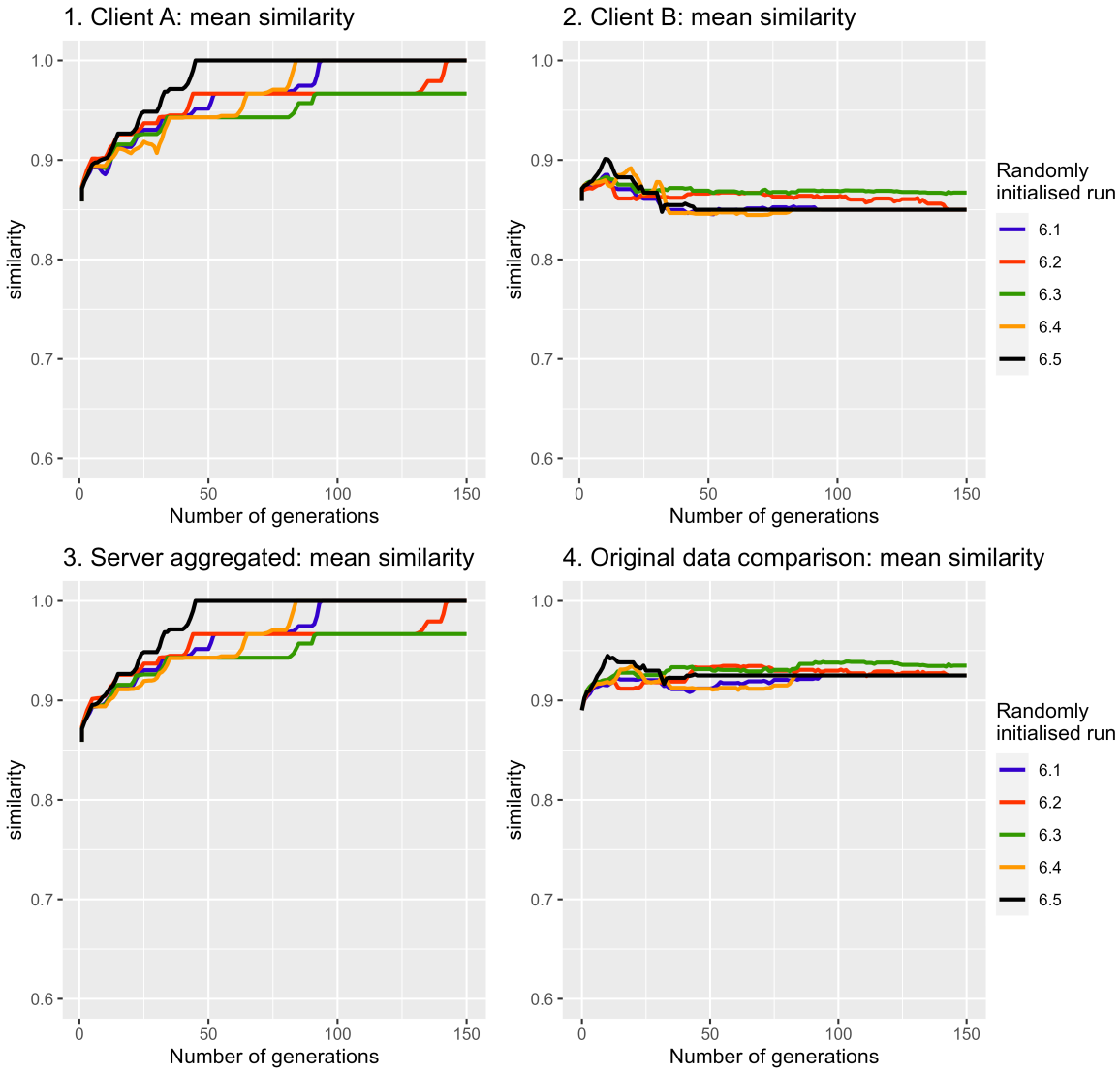

FIGURE 7. Mean (n=50) similarity scores of five randomly initialised runs of the server GA, where the alternating client score is used to drive the GA (five generations using client A, five using client B, etc.). Showing client A (1), client B (2), the server mean alternating scores (3) and the comparison against the original data (4). Note the truncated y axes.