Imputing Censored Patient-Level Data With Neural ODEs

Areeb Gani *
Undergraduate, Class of 2027
Yale University
New Haven, CT 06520
areeb.gani@yale.edu

Abstract

Researchers in survival analysis have long grappled with the problem of censoring, which is when data is not available for certain patients throughout the full duration of the study. In this project, I used Neural ODEs to model the dynamics of patients throughout time in an attempt to impute data when they were missing from follow-ups. The results show that for this selection of data, the Neural ODE had questionable efficacy at imputing covariates — however, future enhancements may show promise in modeling the change in patient features throughout time.

1 Introduction/Motivation

1.1 Clinical Trial Data and Censoring

Researchers conduct clinical trials to monitor patients and their biological features in response to a certain drug or treatment, in the hopes of uncovering an effect for a given target outcome. Most papers regarding clinical trials deal with *survival analysis*, a branch of statistics that attempts to predict the relationship between a patient's primary outcome and time.

A major problem in this field of research, however, is *censoring*. Censoring occurs whenever the data (e.g. survival time) is not available for all the patients for the full duration of the study. This may be because the patient withdrew from the study, or perhaps experienced an adverse outcome after the conclusion of the study. While survival analysis techniques use conditional probability formulas to alleviate this issue, covariates such as biological measurements may be lost.

1.2 Neural ODEs

In the context of a clinical trial, a patient's biological features are often represented as a tensor of biological measurements and other baseline characteristics. If we view the patient's characteristics as a function, we can interpret the change in features over time as a dynamic process that can be modeled through an ordinary differential equation (ODE); namely,

$$f(x, t_0) + \int_{t_0}^{t_e} f'(x, t)dt = f(x, t_e)$$

where $f(x, t_0)$ represents the patient's features at time step t_0, the baseline, and $f(x, t_e)$ represents the patient's features at time step t_e, which is some follow-up in the future. This is the basis behind the Neural ODE (Chen et al., 2019).

*The code can be found at this link. The video can be found here.

For patients who were present at the baseline but not for a certain follow-up in time, if we can train a neural network to compute $f'(x,t)$ for any time t as mentioned above, we can theoretically obtain $f(x,t_e)$ at any time step and explain a patient's characteristics at a follow-up which they missed. This can provide crucial missing data to aid researchers in their survival analyses.

1.3 TOPCAT Trial

The Treatment of Preserved Cardiac Function Heart Failure With an Aldosterone Antagonist (TOPCAT) trial was a multicenter, double-blind trial which randomized 3,445 patients with HFpEF to spironolactone or placebo therapy (Pitt et al., 2014). The data for the TOPCAT trial was obtained from the National Heart, Lung, and Blood Institute through the BioLINCC data repository.

In this trial, patients are monitored through several follow ups (up to 72 months after). However, there was a major dropoff in how many patients are present at each followup. As such, the study is a great candidate to try the dynamic-based imputation of patient data.

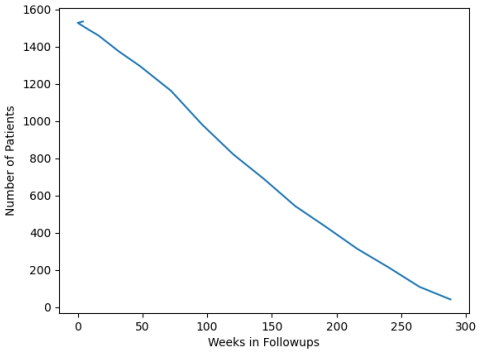


Figure 1: Number of patients present at each followup over time in the TOPCAT dataset.

2 Background/Related Work

Many survival analyses use "single imputation," which replaces a single value. For example, "lost observation carried forward" (LOCF) imputation simply fills in data for later time steps by the final observation. However, this could become problematic for a patient who was only present at the baseline, but then misses the rest of the follow-ups; this imputation method will claim that there were no effects on the patient whatsoever, which could be very false. Even new methods such as missForest, CALIBERrfimpute, and MICE-based imputation have their own downsides as well; they can produce biased regression models and are not robust to skewed data (Stekhoven et al., 2011; Hong et al., 2020).

Importantly, however, none of these methods treat the patient as a variable that dynamically evolves throughout time. Additionally, since Neural ODEs calculate continuous change, they can be used to impute at *any* time step, not only those for which there was prior data. This massively expands the potential of the clinical trial data, if it can be successfully applied.

3 Model

Latent Space Embedding

The first part of my model will reduce the data into a latent space. This is because clinical trial data can be high-dimensional, so using a latent space embedding can be more effective for the NeuralODE. Various techniques can be used, ranging from Principal Component Analysis (PCA), Potential of

Heat Diffusion for Affinity-Based Transition Embedding (PHATE), or even autoencoders such as the Geodesic Autoencoder (Moon et al. 2019; Huguet et al. 2022).

Trajectory Mapping

From here, a Neural ODE will be trained to predict the evolution of a patient throughout follow-up times in the latent space.

```
Sequential(
  (0): Linear(in_features=2, out_features=64, bias=True)
  (1): Softplus(beta=1.0, threshold=20.0)
  (2): Linear(in_features=64, out_features=64, bias=True)
  (3): Softplus(beta=1.0, threshold=20.0)
  (4): Linear(in_features=64, out_features=64, bias=True)
  (5): Softplus(beta=1.0, threshold=20.0)
  (6): Linear(in_features=64, out_features=2, bias=True)
)
```

(a) Architecture of neural network in Neural ODE.

```
Neural ODE:
        - order: 1
        - solver: Tsitouras45()
        - adjoint solver: Tsitouras45()
        - tolerances: relative 0.001 absolute 0.001
        - adjoint tolerances: relative 0.0001 absolute 0.0001
        - num_parameters: 8642
        - NFE: 0.0
```

(b) Neural ODESolver.

Figure 2: Overview of Neural ODE model.

This solver will compute trajectories for each input point (a patient at the baseline), which can then be used to get the interpolated measurements for the patients at a desired time step. I will use Mean Squared Error (MSE) to determine how well the NeuralODE performed in predicting the trajectory of a point from one time step to the next (not using a probabilistic divergence metric, because I am doing loss sample-by-sample rather than throughout the entire distribution).

3.1 Pairwise Time Spans

Since each patient meets at the baseline and then for some number of followups afterwards (up to 14 followups), we train the NeuralODE in steps to accomodate for this variety in time spans. For each patient, we train the NeuralODE on the data going from their baseline to *each* of the followups they had, with different time spans for the different durations. This is required because some patients don't stay for all the followups (which is the entire purpose of this project). In this way, we obtain even more information about a wider expanse of followup times. Note that this still only requires $O(n)$ training steps (where n = length of the dataset), as we can imagine counting only the end of each time interval.

4 Empirical/Theoretical Results

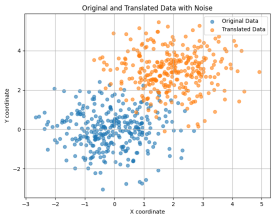
4.1 Model Generalization

To prove the efficacy of this approach, I generated 2D data and translated it (along with some noise) to show how the Neural ODE would be able to extract such dynamics. One can imagine this data as a 2D latent space representation of patients at a baseline, versus the 2D latent space representation of the same patients at a later point in time.

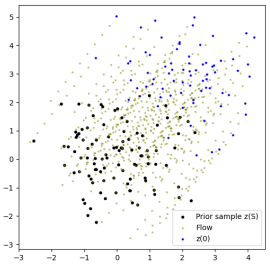
As we can see in Figure 3 above, the NeuralODE does a good job matching the trajectories for each datapoint from the original data to the translated data. Note that we are doing a sample-specific translation, not a distribution-level one. In part (b), we can see the literal trajectories computed for each element in the test set. And in part (c), we see the vector field which tells us the overall shift of the data. The mean squared error for the Neural ODE's prediction in this dataset was 0.01.

4.2 Latent Space Embedding

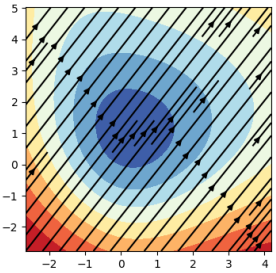
For the TOPCAT data, only a subsection of continuous measurements taken at every follow-up were used in the expression vector; this is because a lot of the data is information that doesn't change over time, and thus does not need dynamic systems modeling. The best embedding results were obtained using PCA, which is likely because the data is low-dimensional to begin with.



(a) Visualization of toy 2D data.



(b) Trajectories computed by Neural ODE.



(c) Vector field of derivatives computed by neural network.

Figure 3: Toy data generalization analysis.

Table 1: Mean Squared Error in Latent Space Projections

Time (in weeks)	MSE in Latent Space
4	0.95
16	1.05
96	1.57
144	1.78
264	1.76
288	2.76

As shown in Figure 4 part (a), PCA did not uncover a significant difference in distribution between patients at the baseline and patients at the later time step.

4.3 Neural ODE Performance

The mean squared error for test data in each of the time frames can be found in Table 1. As expected, the MSE increases over time, although it seems to drop at around 264 weeks.

4.4 Vector Field and Trajectories

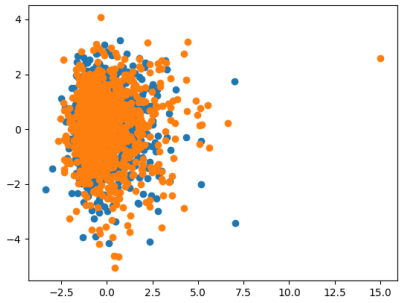
Figure 4b shows the vector field computed by the NeuralODE for shifting the latent space embeddings over time. This corroborates the earlier figure, showing how our latent space gets pushed more inwards over time.

Below, we can see an example of trajectories computed by the NeuralODE. As we might expect, since the latent spaces are hardly separable, it was tough to get solid trajectory mappings; but some of them steer in the right direction, while others are wrong altogether. The start and end positions are often off, too, although this is likely because there are so many other datapoints that are skewing the NN; it is not overfitting and learning the specific mappings for individual points in space.

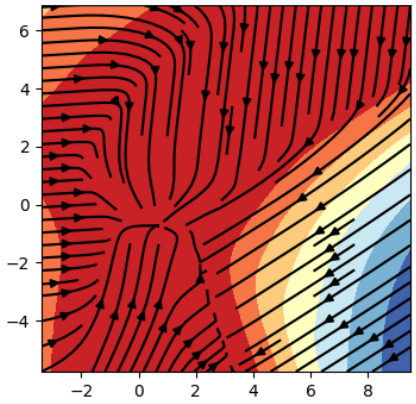
As a final example, we can see how our model reconstructed patient data by inverting from the PCA space. Below is a random patient with only a baseline visit, and applied the NeuralODE and reconstructed their predicted covariates for $t = 96$ weeks in the future. The results can be seen in Table 2.

Table 2: Predicted Covariates

	NA (mmolL)	K (mmolL)	CL (mmolL)	CO2 (mmolL)	BUN (mgdL)
Baseline	139.0	3.3	98.0	29.0	13.0
Week 12 (predicted)	138.28	4.03	99.24	29.71	19.66

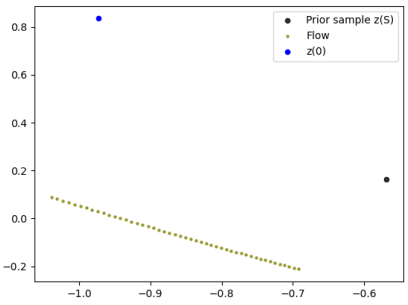


(a) Latent Space Embedding with PCA. Points in blue indicate patients at the baseline, and points in orange indicate patients after 96 weeks.

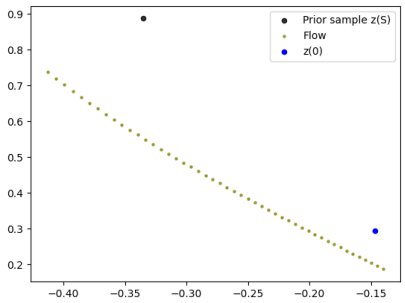


(b) Vector field visualization from Neural ODE.

Figure 4: Latent space of patient data.



(a) Trajectory going the wrong direction.



(b) Trajectory going the right direction.

Figure 5: Examples of trajectories in the TOPCAT data.

5 Conclusions and Future Work

Including more data would give a more robust expression of patient covariates throughout time. If more data was included, an alternative dimensionality reduction than PCA might be required, especially if the data consisted mixed categorical and continuous data. An autoencoder into a higher-dimensional latent space may be the best way to preserve meaning for the vector representing each patient throughout time.

Another idea to note is why the trajectory mapping was poor; this is mainly because our latent space at different time steps had significant overlap with other data. This is why the NeuralODE was trying to push points close together in the vector field (Figure 4, part b), although this may not have always been the best trajectory for each point.

Future analyses can also integrate more advanced methods such as TrajectoryNet and MIOFlow, which make adjustments to the model that can augment the flows. I did implement magnitude regularization (part of TrajectoryNet), but it did not aid much in this project specifically.

However, this concept does have promise. Effective dynamic modeling can cut costs in clinical trials by limiting the amount of data required, especially if meaningful dynamics can be learned by these solvers.

References

[1] Chen, R. T. Q., Rubanova, Y., Bettencourt, J., & Duvenaud, D. (2019, December 14). Neural ordinary differential equations. arXiv.org. https://arxiv.org/abs/1806.07366

[2] Hong, S., & Lynn, H. S. (2020, July 25). Accuracy of random-forest-based imputation of missing data in the presence of non-normality, non-linearity, and interaction - BMC Medical Research methodology. BioMed Central. https://bmcmedresmethodol.biomedcentral.com/articles/10.1186/s12874-020-01080-1

[3] Huguet, G., Magruder, D. S., Tong, A., Fasina, O., Kuchroo, M., Wolf, G., & Krishnaswamy, S. (2022, November 3). Manifold interpolating optimal-transport flows for trajectory inference. arXiv.org. https://arxiv.org/abs/2206.14928

[4] Moon, K. R., van Dijk, D., Wang, Z., Gigante, S., Burkhardt, D. B., Chen, W. S., Yim, K., Elzen, A. van den, Hirn, M. J., Coifman, R. R., Ivanova, N. B., Wolf, G., & Krishnaswamy, S. (2019, December 3). Visualizing structure and transitions in high-dimensional biological data. Nature News. https://www.nature.com/articles/s41587-019-0336-3

[5] Pitt, B. (n.d.). Spironolactone for heart failure with preserved ejection fraction | Nejm. https://www.nejm.org/doi/full/10.1056/NEJMoa1313731

[6] Stekhoven, D. J., & Bühlmann, P. (2011, October 28). Missforest-non-parametric missing value imputation for mixed-type data. OUP Academic. https://academic.oup.com/bioinformatics/article/28/1/112/219101