Neural Causal Models with TRAM-DAGs:
A framework for modelling causal effects in a flexible and interpretable way and for making subsequent causal queries.

Master Thesis in Biostatistics (STA495)

by

Mike Krähenbühl

Matriculation number: 18-652-149

supervised by

Prof. Dr. Beate Sick

Prof. Dr. Oliver Dürr, HTWG Konstanz

Zurich, July 2025

Neural Causal Models with TRAM-DAGs:

Applied on real-world data and used for ITE estimation.

Mike Krähenbühl

Version May 24, 2025

Contents

Preface

In the introduction part, my aim is to give a summary of important concepts of causal inference and causal models. Further, I motivate the importance for methods that allow to draw causal conclusions from observational data in contrast to randomized controlled trials (RCTs) and that the proposed framework of tram-dags (cite sick duerr) can be used as such a tool.

In the methods section, I give a detailed description of the tram-dag framework and how it works by illustrating it on a simple simulated example. I also show for what kind of causal queries the model can be used for. Although it is not a topic for observations data, I discuss how the model can be used for estimating the individualized treatment effect (ITE).

In the results section, I will first show results from simulation studies where the ground truth is known. Second, I will show results on a real-world example in the setting of climate data. Third, I present the results of the ITE estimation.

In the final section, I will discuss the results and give a conclusion about the advantages and limitation of this framework and provide an outlook.

Mike Krähenbühl
July 2025

Chapter 1

Introduction

The important questions that studies want to answer are usually not associational but causal (Pearl, 2009a). For example, these are questions that ask for effects when making a certain intervention, like the effect of a treatment. They can ask for reasons that lead to the observed outcome, like which disease caused the given symptoms. Or what would have been different if another action was taken, like what would the gdp have benn, if the interest rates were incereased only by 25 instead of 75 Bps. To answer such questions, a causal reasoning must be applied that aims to understand the underlying data genearting mechanism, sole associational reasoning directly from the data is not sufficient.

Importance of Causal Inference on Observational Data

The gold standard to measure causal relationships between an intervention and an outcome is the randomized controlled trial (RCT) (Hariton and Locascio, 2018). The key concept of this prospective study design, is that the participants are randomly allocated due either the treatment or control group. Due to this randomization, the influence of potential confounding variables is eliminated and study groups are balanced with respect to baseline characteristics allowing for an unbiased cause-effect estimation. Disadvantages of RCTs include but are not limited to often high cost, the time for planning and executing the trial, and generalisability to the population of interest. Furthermore, RCTs typically aim to estimate an average treatment effect on a sample, which is the difference in the averages accross the treatment groups (Nichols, 2007). However, patients have individual responses to the treatment, depending on their characteristics. In personalized medicine, such individual treatment effects are crucial. Another central limitation of RCTs is that in many scenarios they can not be conducted due to ethical or practical reasons. For example, an RCT is only ethical in the case of clinical equipoise, which means that there is uncertainty about the (superiority) of one of the two treatment arms (Freedman, 1987). It is not acceptable to treat one group with the assumed inferior treatment. The same is true for obviously harmful interventions, like smoking or drinking alcohol. In these cases, it is not possible to conduct an RCT to estimate the causal effect of smoking on lung cancer.

Therefore, much of the research aims to make causal inference from observational data in a non-experimental or quasi-experimental design. In an observational setting, there are usually confounding variables that make it challenging to measure the effect between exposure and outcome. Methods for causal infreence on observational data for example aims to correctly adjust or control for confounders. Sick and Dürr (2025) proposed the framework of TRAM-DAGs to estimate the causal relationships in an observational setting and make subsequent queries. The aim of this thesis is to further analyze this method and apply it in a real-world scenario.

General Causality review (DAG, SCM, Pearls ladder, rubins rules?)

Causal relationships can be represented by a directed acyclic graph (DAG) as, for example, shown in Figure 1.1(a). The variables, or nodes, are connected by directed edges which indicate the path of causal dependence.

Usually we want to answer questions that can be assigned to one of the groups in pearls hierarchy of causation Pearl (2009b). Visual examples are presented in Figure 1.1(a)-(c). Level 1 are observational queries which are conditional probabilities $P(Y \mid X)$ and can be answered directly from the joint distribution $P(Y \cap X)$. Level 2 are interventional queries which are probabilities $P(Y \mid do(X))$ that result by a taken action do(X). Where observational queries only require to know the joint distribution, for interventional queries an additional understanding of the causal mechanism is necessary. Level 3, the analysis of counterfactuals, poses the biggest challenge. These are what-if questions. An example of this would be if a sick patient was treated with a certain treatment and then died. Death would therefore be the observed and therefore factual outcome. The counterfactual outcome is the outcome that would have occurred if the patient had received a different medication. Such counterfactual questions are often labelled as metaphysical because they can never be tested directly. However, there are important practical questions that require the analysis of such counterfactuals.

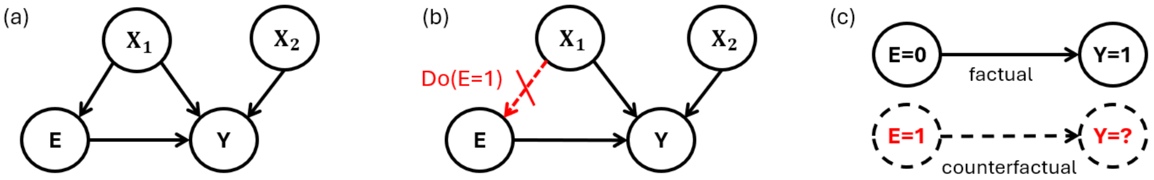

Figure 1.1: Example for the three levels of Pearl's hierarchy of causation. (a) DAG for observational data. (b) DAG when making a do-intervention by fixing the variable E at a certain value. c) Observed factual outcome and corresponding counterfactual query.

To illustrate Pearl's three levels of causality, I consider a simplified example involving the exposure Exercise (E), the outcome Heart Disease (Y), the confounder Age (X_1) and the additional covariate Smoking (X_2). I assume that exercise reduces the risk of heart disease, but both variables are also influenced by age. Figure 1.1(a)-(c) illustrates the corresponding scenarios.

Level 1: Observational ("seeing"). We observe the joint distribution of variables without intervention. Example: What is the probability of heart disease given that a person exercises?

$$P(Y \mid E = 1)$$

This can be estimated directly from data (by filtering for E=1 and calculating the fraction of Y), but does not account for confounding.

Level 2: Interventional ("doing"). We consider the effect of an intervention on the system. Example: What is the probability of heart disease if everyone were made to exercise, regardless of age or smoking?

$$P(Y \mid \mathrm{do}(E = 1))$$

This requires assumptions about the causal structure.

Level 3: Counterfactual ("imagining"). We ask what would have happened under different circumstances, which means imagine an alternative reality. Example: For a person who does not exercise and has heart disease, would they still have heart disease if they had exercised?

$$P(Y_{E=1} \mid E = 0, Y = 1)$$

Here $Y_{E=1}$ is the outcome under the positive exposure. Counterfactual queries require a structural causal model and cannot be answered from data alone.

What is a Structural Causal Model? To answer questions from Pearl's ladder of causation, the concept of DAG's can be extended to structural causal model (SCM). A set of structural equations of the from $x_i = f_i(pa(x_i), z_i)$, $i = 1, ..., n$ forms a structural causal model

(Pearl, 2009b). $pa(x_i)$ are the direct causal parents of X_i and therefore directly determe its value. Z_i are errors that follow exogeneous noise distributions $P(Z_i)$. They can be understood as latent variables that represent unmodeled factors which can not be observed or measured directly. By convention, the Z_i are assumed to be mutually independent. The potentially non-linear function f_i determines the functional form of the parents and the noise that represents the data generating mechanism of the dependent variable X_i. Hence, in a SCM a source node X_j can be represented as $X_j = f_j(z_j)$ since it does not depend on any other variables in the system. Once all components of the structural equations are known (or assumed to be known), it is fully deterministic.

This representation makes it practical to determine interventional distributios and counterfactual queries. This will be discussed in detail in Section XXX.

In this thesis, the focus is not on finding the causal structure of a system. Such a structure can be found by structure finding algorithms, or determided by expert knowledge, for example. Instead, I assume the DAG to be known and focus on the estimation of the functional form of the relationships.

There is a variety of methods that are applied to quantify these structural equations that form the resulting SCM.

The simplest method might be the linear regression which assumes gaussian error terms Z_i and a linear f_i. Similarly other classical statistical methods can be applied they have the advantage of being typically well defined and having interpretable parameters. However, they require to make strong assumptions about the underlying data generating mechanism which can make them susceptible to bias if these assumptions are violated.

Then there is also the possibility to model the relationships with various methods based on neural networks. Because they can be very flexible, they can also model complex underlying distributions with practically no bias. But this comes at the cost of less interpretability. And often they are limited to continuous data.

The framework of TRAM-DAGs proposed by Sick and Dürr (2025) builds a bridge between these two approaches of classical statistical and deep learning methods. It allows us to model the causal relationships with interpretable or fully-flexible parts, depending on what is regarded more important at the moment. The basis of this model is to construct the structural equations as transformation models as introduced by Hothorn *et al.* (2014), which is a flexible distributional regression method. To make them even more customizable, these transformation models (TRAMS) were extended to deep TRAMS (Sick *et al.*, 2021). Applied in a causal context, these deep TRAMS form the framework of TRAM-DAGs that can be fitted on observational data and allow to tackle causal queries on all three levels of Pearl's causal hierarchy. This framework will be explained in detail in the Section XXX.

The primary goal of this thesis is to apply TRAM-DAGs on real world data where the underlying data generating process is unknown.

An application where causal inference is of particular importance is the estimation of individualized treatment effects (ITE). Chen *et al.* (2025) showed that all causal machine learning models that were trained on a train set failed to generalize to a test set.

The estimation of this with the help of TRAM-DAGs constitutes the second objective of this thesis. Although I analyze it in the setting of RCT data and not observational.

Chapter 2

Methods

In this section I will explain the necessary background needed to understand the TRAM-DAGs. Once the framework of tram dags is explained, I will present how the experiments of the simulation, the application on real data and the ITE estimation are conducted.

The goal of TRAM-DAGs is to estimate the structural equations according to the causal order in a given DAG in a flexible and possibly still interpretable way in order to sample observational and interventional distributions and to make counterfactual statements. The estimation requires data and a DAG that describes the causal structure. It must be assumed that there are no hidden confounders. TRAM-DAGs estimate for each variable X_i a transformation function $Z_i = h_i(X_i \mid pa(X_i))$, where Z_i is the noise value and $pa(X_i)$ are the causal parents of X_i. The important part here is that we can rearrange this equation to $X_i = h_i^{-1}(Z_i \mid pa(x_i))$ to get to the structural equation. The transformation functions h are monotonically increasing functions that are a representation of the conditional distribution of X_i on a latent scale. They are based on the idea of transformation models as introduced by Hothorn *et al.* (2014) but were extended to deep trams by Sick *et al.* (2021). In the following sections I review the most important ideas of these methods as they are the essential components of TRAM-DAGs.

2.1 Transformation Models

Transformation models are a flexible distributional regression method for various data types. They can be for example specified as ordinary linear regression, logistic regression or proportional odds logistic regression. But Transformation models further allow to model conditional outcome distributions that do not even need to belong to a known distribution family of distributions by model it in parts flexibly. This reduces the strength of the assumptions that have to be made.

The basic form of transformation models can be described by

$$F(y|\mathbf{x}) = F_Z(h(y \mid \mathbf{x}) = F_Z(h_I(y) - \mathbf{x}^\top \boldsymbol{\beta}) \tag{2.1}$$

, where $F(y|\mathbf{x})$ is the conditional cumulative distribution function of the outcome variable Y given the predictors $\mathbf{x}$. $h(y \mid \mathbf{x})$ is a transformation function that maps the outcome variable y onto the latent scale of Z. F_Z is the cumulative distribution function of a latent variable Z, the so-called inverse-link function that maps $h(y \mid \mathbf{x})$ to probabilities. In this basic version, the transformation function can be split into an intercept part $h_I(y)$ and a linear shift part $\mathbf{x}^\top \boldsymbol{\beta}$, where the vector $\mathbf{x}$ are the predictors and $\boldsymbol{\beta}$ are the corresponding coefficients.

If the latent distribution Z is chosen to be the standard logistic distribution, then the coefficient β_i can be interpreted as log-odds ratios when increasing the predictor x_i by one unit, holding all other predictors unchanged. This means that an increase of one unit in the predictor x_i leads to an increase of the log-odds of the outcome Y by $\boldsymbol{\beta}$. The additive shift of the transformation function means a linear shift on the latent scale (herer log-odds). The following transformation to probabilities by F_Z potentially leads to a non-linear change in the conditional outcome distribution on the original scale. This means not only is the distribution shifted, also its shape can change to some degree based on the covariates. More details about the choice of the latent distribution and the interpretation of the coefficients are provided in the appendix XXX.

For a continuous outcome Y the intercept h_I is represented by a bernstein polynomial, which is a flexible and monotonically increasing function

$$h_I(y) = \frac{1}{M+1} \sum_{k=0}^{M} \vartheta_k \, \mathrm{B}_{k,M}(y) \tag{2.2}$$

, where ϑ_k are the coefficients of the bernstein polynomial and $\mathrm{B}_{k,M}(y)$ are the Bernstein basis polynomials. More details about the technical implementation of the bernstein polynomial in the context of TRAM-DAGs is given in the appendix XXX.

For a discrete outcome Y the intercept h_I is represented by cut-points, which are the thresholds that separate the different levels of the outcome. For example, for a binary outcome Y there is one cut-point and for an ordinal outcome with K levels there are $K-1$ cut-points. The transformation model is given by

$$P(Y \le y_k \mid \mathbf{X} = \mathbf{x}) = F_Z(\vartheta_k + \mathbf{x}^\top \boldsymbol{\beta}), \quad k = 1, 2, \ldots, K-1 \tag{2.3}$$

A visual representation for a continuous and discrete (ordinal) outcome is provided in Figure 2.1.

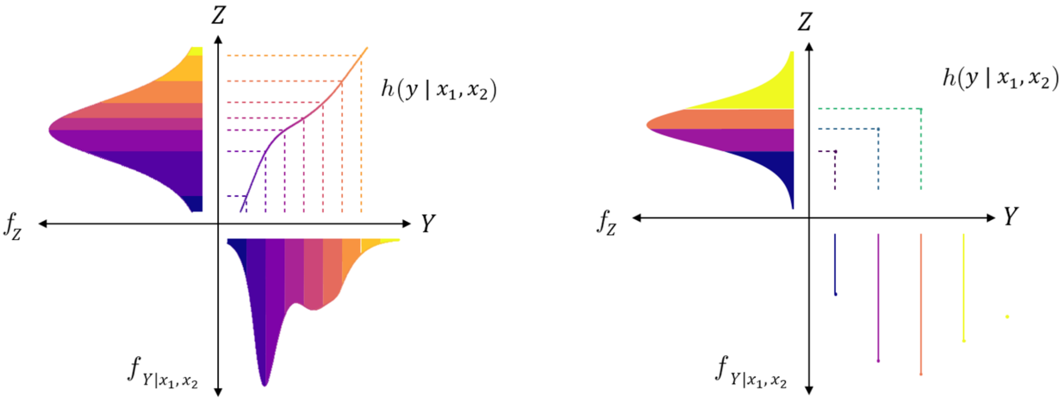

Figure 2.1: Left: Example of a transformation model for a continuous outcome Y with a smooth transformation function. **Right:** Example of a transformation model for an ordinal outcome Y with 5 levels. The transformation function consists of cut-points that separate the probabilities for the levels of the outcome. In both cases the latent distribution Z is the standard logistic and the predictors $\mathbf{x}$ induce a linear (vertical) shift of the transformation function.

For the remainder of this thesis, I rely on the idea of these transformation models to model the conditional distribution functions represented by the transformation functions of the respective variables. The standard logistic distribution is used as F_Z, which results in a logistic transformation model.

2.2 Deep TRAMs

The transformation models as discussed before were extended to deep TRAMs using neural networks. The goal is to get a parametrized transformation function by training a modular neural network and thereby minimizing the NLL. This minimization is done with Deep learning optimization methods.

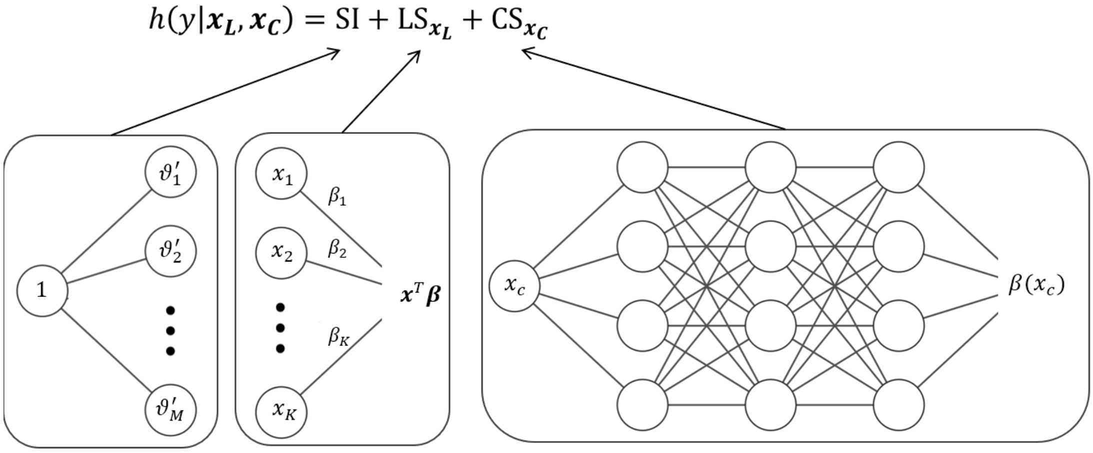
$h(y|\boldsymbol{x_L}, \boldsymbol{x_C}) = \mathrm{SI} + \mathrm{LS}_{\boldsymbol{x_L}} + \mathrm{CS}_{\boldsymbol{x_C}}$
ϑ'_1
ϑ'_2
1
ϑ'_M
x_1
β_1
x_2
β_2
$\boldsymbol{x}^T\boldsymbol{\beta}$
β_K
x_K
x_c
$\beta(x_c)$

Figure 2.2: Modular deep transformation model. The transformation function $h(y \mid \mathbf{x})$ is constructed by the outputs of three neural networks.

So on the slides you can see again the transformation function for the outcome Y that depends on the covariates X. Here, XL should stand for the predictors that shift the transformation function in a linear manner (as in the examples on the previous slides) and XC stands for Predictors that have a non linear influence. And the cool thing here is that these Complex Predictors could also include something like an Image.

The first part of the transformation function is a Simple intercept which is responsible for the baseline shape of the distribution. This intercept is constructed by a neural network, that takes no predictors as input and just outputs the parameters for the Bernstein polynomial or for the cut points in the discrete case. If we wanted to allow for more flexibility, we could give a predictor as input instead of just a constant, but in this presentation I will only show the simple case.

Next there is a Linear Shift which is basically a linear combination of the linear predictors. We can obtain this by creating another neural network with no hidden layers taking only the predictors as input and producing the linear shift as output.

Finally there is the complex shift which we obtain by giving the complex predictors into a neural network with some hidden layers and getting a single number as output.

The Neural Network training happens iteratively by starting with a (random) parameter configuration and then using the outputted intercepts and shift to calculate the loss, which is the negative log likelihood of our transformation model. To improve the loss, the parameters are then slightly adjusted by the adam optimizer. Then the process is repeated until the parameter estimates converge. So this means we are optimizing the negative log likelihood of the logistic transformation model, which represents the conditional distribution function of our outcome variable.

The nice thing of these deep transformation models is that we can specify, whether we want a linear shift bx or a complex shift b(x) or even a complex intercept and thereby controlling the flexibility.

TRAM-DAGs

And here you can see how this looks in our case, where we apply these deep transformation models in a causal setting. So we assume a pre-specified DAG which defines the Causal dependence. And then we model each node by a transformation model that is conditional on its parents.

In this Example dag we have 3 variables:

X1 is continuous and a source node, meaning it has no parents that it depends on, hence the transformation function h only represents the intercept.

Then X2 depends on X1, so this means that the transformation function changes with X1. In what way the transformation function is allowed to change, has to be specified. This could either be a linear or complex shift or even a complex intercept.

And finally there is the ordinal variable X3 which has 4 levels. The Transformation function consists of 3 cut points that depend on X1 and X2. These cut-points represent the Probabilities of the 4 levels of X3.

Choice of link function does not matter if fully flexible (CI) but it puts some assumptions if not fully flexible. Same as in the sense of the SCM where the choice of the Noise distribution can matter (depending on how flexible the equations are).

Ok so lets make a simulation example and put this all together.

First of all we have observational data that follows a pre-defined DAG without hidden confounders. In practice such a dag can be defined by expert knowledge or by some sort of structure finding algorithm.

Then we want to estimate the conditional distribution function of each variable so that we can sample from the distributions and make causal queries.

So in this example we assume the same dag as on the previous slide with the 3 variables. X1 and X2 are continuous and X3 is ordinal. Now we also specify how these variables are related. So X1 is the source node. X2 depends on X1 through a linear shift and X3 depends on X1 by a linear shift and on X2 by a complex shift.

This model structure can also be presented by an adjacency matrix where the rows indicate the source of effect and the columns are the target of the effect. In our algorithm we use this Meta-adjacency matrix to control the information flow.

Construct Modular Neural network

Now we set up a modular neural network which produces the components of the transformation functions as outputs. The Adjacency Matrix thereby controls the information flow. Here you also can see the number of parameters. In total there are 281 Parameters in this model, but not all of them are used I will not get into the details here. On the right side you can also see the neural network that produces the complex shift from X2 to X3. We chose to model it by 4 hidden layers with 2 nodes each. That should allow for high enough flexibility.

Finally the outputs of the different Neural network parts are combined to the transformation functions for each node, from which we then derive the negative log likelihood.

Optimizers and hyperparameters etc

Adam, Batchnormalization, activation functions (relu, sigmoid) Impact of Scaling (maybe in Appendix) Neural network works best if inputs are scaled. Proof that we can do that, it just changes the interpretation. For structure finding algorithms, this might be problematic, because increasing variance along the causal order would be destroyed. (why, how, interpretation change etc. check meeting notes 22.04.2025)

Different Intercepts, and Shifts

and show, describe how the transformation function and hence the conditional distribution will change in each scenario. Also show in detail how the neural networks would be set up, how the information flow is controlled and what kind of outputs are produced and how they further have to be transformed.

Describe interpretation quickly and refer to formal proof in the Appendix.

Fitting Betas Interpretable

The two parameters for our linear shift terms are plotted here. We can see that they converge quickly to the same values as we used in the DGP. We can interpret these parameters as log-odds ratios if changing the value of the parent by one unit.

Intercepts

Show the Discrete case with just cutpints (only K-1 parameters of outputs are used) Show the continuous case where the outputs are transformed to monotonically increasing betas for the bernstein polynomial. Also describe Bernstein polynomial construction in detail with scaling and linear extrapolation.

Here I plotted the intercepts of the 3 transformation functions. They also resemble the DGP very nicely.

Linear and complex shifts

Here in the first two plots we can see the linear shifts. And in the right plot we have the complex shift of X2 on X3. The estimated shifts match quite well with the DGP.

Complex shift (Interaction example) to show what is also possible

Here I just want to make a short input from another example. So there the true model was that of a logistic regression with the binary outcome Y and 3 predictors. The binary treatment T and the two continuous predictors X1 and X2. There was also an interaction effect assumed between treatment and X1. So this basically means that the effect of X1 on the outcome is different for the two treatment groups.

And here we can show that our TRAM-DAG specified by a complex shift of T and X1 can also capture this interaction effect quite well.

Loss function

And this is how the negative log likelihood looks like for a continuous outcome. It is derived from the CDF based on the logistic transformation model. A special thing here is that the outcome variable has to be scaled to the range between 0 and 1 first, and this scaling also has to be considered when calculating the NLL. But I will not go through this now. For final Loss, the individual losses of the nodes are added together. (only in R framework, in Python they are fitted individually?)

NN Training

Now we have everything in place to train the neural network. Here I run the model for 400 Epochs (which means that the model has seen each sample 400 times). On the right you can see the loss of the training set and also the validation loss as comparison. The NLL quickly dropped to around 1.1 and then didnt change much anymore, which indicates that the parameter estimates have probably converged to a good state.

discrete predictors

sampling from the tram dag Each variable Xi in the DAG can be described by a structural equation Xi = f(Zi, pa(Xi)). For a continous outvome x1, in TRAM-DAGs this structural equation is the inverse of the conditional transformation function Xi = h-1(Zi | pa(Xi)). for a discrete outcome it is defined as... show sampling.

Refer back to the SCM, that we basically can obtain the structural equations from our model. Okay so now we have estimates for the conditional transformation functions of our 3 variables. To generate a sample for a node, we first sample a random value from the latent distribution. In our case from the standard logistic. We denote this sample as Z.

Next we want to determine the value of the Node X. If X is continuous, we can apply the inverse of the transformation function evaluated at Z to find X. If X is ordinal, we just select the corresponding category that belongs to the next bigger cut-point.

If we want to make a do-intervention, we just fix a node at the desired value.

counterfactuals

Describe how to do it, limitations etc.

see pearl book causality: 1.4.4 Counterfactuals in Functional Models

ITE how it is applied in our model?

RCTs only measure the average treatment effect. There will be patients who respond better or worse to the treatment because patient specific characteristcs. In personalized medicine however, the aim is to find the optimal treatment for a specific individual. Such a measure that can help in decision making is the ITE.

Rubins potential outcomes framework.

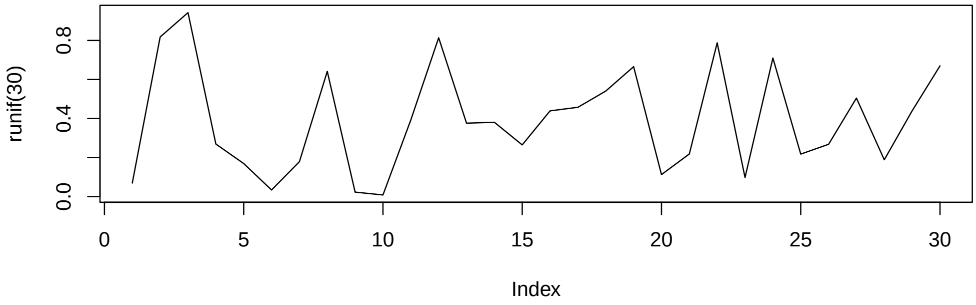

Figure 2.3: Test figure to illustrate figure options used by knitr.

Maybe it is the methods section. Here however, we give a couple hints. Note that you can wisely use *preamble*-chunks. Minimal, is likely:

```
library(knitr)
opts_chunk$set(
    fig.path='figure/ch02_fig',
    self.contained=FALSE,
    cache=TRUE
)
```

Defining figure options is very helpful:

```
library(knitr)
opts_chunk$set(fig.path='figure/ch02_fig',
               echo=TRUE, message=FALSE,
               fig.width=8, fig.height=2.5,
               out.width='\\textwidth-3cm',
               message=FALSE, fig.align='center',
               background="gray98", tidy=FALSE, #tidy.opts=list(width.cutoff=60),
               cache=TRUE
)
options(width=74)
```

This options are best placed in the main document at the beginning. Otherwise a `cache=FALSE` as knitr option is necessary to overrule a possible `cache=TRUE` flag.

Notice how in Figure 2.3 everything is properly scaled.

2.3 Citations

Recall the difference between `\citet{}` (e.g., Chu and George (1999)), `\citep{}` (e.g., (Chu and George, 1999)) and `\citealp{}` (e.g., Chu and George, 1999). For simplicity, we include here all references in the file `biblio.bib` with the command `\nocite{*}`.

Chapter 3

Results

Chapter 4

Discussion and Outlook

Chapter 5

Conclusions

Bibliography

Chen, H., Aebersold, H., Puhan, M. A., and Serra-Burriel, M. (2025). Causal machine learning methods for estimating personalised treatment effects – insights on validity from two large trials. 3

Chu, E. and George, A. (1999). *Inside the FFT Black Box: Serial and Parallel Fast Fourier Transform Algorithms*. CRC Press. 10

Collett, D. (2014). *Modelling Survival Data in Medical Research*. Chapman & Hall/CRC Texts in Statistical Science. Taylor & Francis, third edition.

Freedman, B. (1987). Equipoise and the ethics of clinical research. *The New England journal of medicine*, **317**, 141–145. 1

Hariton, E. and Locascio, J. J. (2018). Randomised controlled trials - the gold standard for effectiveness research. *BJOG: An International Journal of Obstetrics & Gynaecology*, **125**, 1716 – 1716. 1

Hothorn, T., Kneib, T., and Bühlmann, P. (2014). Conditional transformation models. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, **76**, 3–27. 3, 5

Hurrell, J. W., Kushnir, Y., Ottersen, G., and Visbeck, M. (2013). *An Overview of the North Atlantic Oscillation*, 1–35. American Geophysical Union.

Kratzer, G. and Furrer, R. (2018). *varrank: an R package for variable ranking based on mutual information with applications to observed systemic datasets*. R package version 0.3.

Nichols, A. (2007). Causal inference with observational data. *The Stata Journal*, **7**, 507 – 541. 1

Pearl, J. (2009a). Causal inference in statistics: An overview. *Statistics Surveys*, **3**, 96 – 146. 1

Pearl, J. (2009b). *Causality: Models, Reasoning and Inference*. Cambridge University Press, USA, 2nd edition. 2, 3

Porcu, E., Furrer, R., and Nychka, D. (2020). 30 years of space-time covariance functions. *Wiley Interdisciplinary Reviews. Computational Statistics*, **13:e1512**, 1–24.

Sick, B. and Dürr, O. (2025). Interpretable neural causal models with tram-dags. Accepted at the CLeaR 2025 Conference. 1, 3

Sick, B., Hathorn, T., and Dürr, O. (2021). Deep transformation models: Tackling complex regression problems with neural network based transformation models. In *2020 25th International Conference on Pattern Recognition (ICPR)*, 2476–2481. 3, 5

Wang, C. and Furrer, R. (2020). Monte Carlo permutation tests for assessing spatial dependence at different scales. In La Rocca, M., Liseo, B., and Salmaso, L., editors, *Nonparametric Statistics. ISNPS 2018. Springer Proceedings in Mathematics & Statistics*, volume 339, 503–511. Springer.