# Supplementary Material: Deep Conditional Transformation Models

Philipp F. M. Baumann (✉)[1 [0000−0001−8066−1615]], Torsten Hothorn[2 [0000−0001−8301−0471]], and David Rügamer[3 [0000−0002−8772−9202]]

[1] KOF Swiss Economic Institute, ETH Zurich, Zurich, Switzerland
baumann@kof.ethz.ch
[2] Epidemiology, Biostatistics and Prevention Institute, University of Zurich, Zurich, Switzerland
torsten.hothorn@uzh.ch
[3] Department of Statistics, LMU Munich, Munich, Germany
david.ruegamer@stat.uni-muenchen.de

## 1 Proofs

The following Lemma proofs the monotonocity of $h_1$ when using Bernstein Polynomials, constrained coefficients in $\boldsymbol{\Gamma}$ and a positive matrix $\boldsymbol{\mathcal{B}}$.

**Lemma 1.** *Let $y_i > y_j$ for $i \neq j$ and $\boldsymbol{\mathcal{B}}_i, \boldsymbol{\mathcal{B}}_j \in \mathbb{R}^{1\times P}$ be the corresponding vector entries of the matrix $\boldsymbol{\mathcal{B}}$ with elements*

$$b_{i,k} = b_{j,k} \; \forall k \in \{1, \dots, P\}. \tag{1}$$

*That is, both outcome vectors $y_i, y_j$ have the same features. Then $h_1(y_i) > h_1(y_j)$ can be ensured if $b_{i,k} > 0 \; \forall k \in \{1, \dots, P\}$.*

*Proof.* Define the $l$th row of $\boldsymbol{\mathcal{A}}$ as $a(y_l)^\top = (a_{l,1}, \dots, a_{l,M+1}) \in \mathbb{R}^{1\times M+1}$ and entries in $\boldsymbol{\Gamma}$ as $\gamma_{m,k}, m = 1, \dots, M+1, k = 1, \dots, P$. Using Bernstein Polynomials and monotonically increasing coefficients $\gamma_{m,k} < \gamma_{m+1,k}$ for $1 \leq m \leq M$, we have

$$h_1(y_j) = \sum_{m=1}^{M+1} \sum_{k=1}^{P} a_{j,m} b_{j,k} \gamma_{m,k} = \sum_{k=1}^{P} \sum_{m=1}^{M+1} a_{j,m} b_{j,k} \gamma_{m,k} \tag{2}$$

$$= \sum_{k=1}^{P} b_{j,k} \sum_{m=1}^{M+1} a_{j,m} \gamma_{m,k} = \sum_{k=1}^{P} b_{i,k} \sum_{m=1}^{M+1} a_{j,m} \gamma_{m,k}, \tag{3}$$

where the last equivalence uses the assumption (1). Due to the property of Bernstein polynomials, it holds

$$h_{1,k}(y_j) := \sum_{m=1}^{M+1} a_{j,m} \gamma_{m,k} < \sum_{m=1}^{M+1} a_{i,m} \gamma_{m,k} =: h_{1,k}(y_i) \; \forall k \in \{1, \dots, P\} \tag{4}$$

by the assumption $y_i > y_j$ and monotonocity of $\gamma_{m,k}$. (4) also holds, if multiplied with $b_{i,k} > 0$ on both sides and finally also when summed over all elements $k \in \{1, \ldots, P\}$ as (4) holds for all $k$. Thus

$$h_1(y_j) = \sum_{k=1}^{P} b_{i,k} \sum_{m=1}^{M+1} a_{j,m}\gamma_{m,k} < \sum_{k=1}^{P} b_{i,k} \sum_{m=1}^{M+1} a_{i,m}\gamma_{m,k} = h_1(y_i).$$

□

## 2 Hyperparameter and Network Settings

### 2.1 Results of the Numerical Experiments

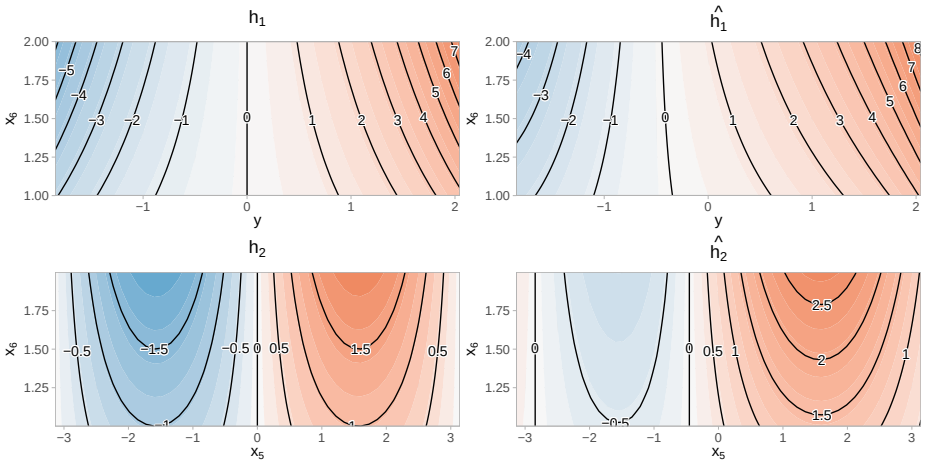


**Fig. 1.** Exemplary visualization of the learned feature-driven interaction term $h_1$ (upper row) as well as shift term $h_2$ (lower row). Plots on the left show the data generating surface $h_1, h_2$, plots on the right the estimated surface $\hat{h}_1, \hat{h}_2$ for different values of the feature inputs. The plots correspond to the DGP setting $g_1, \eta_1, \sigma_2$.

### 2.2 Specifications for the Numerical Experiments

For $g$, we considered two choices. First, the sinus hyperbolicus $g_1(y) = \sinh(y)$ and, second $g_2(y) = \log(y) + 1$ if $0 < y \leq 1$, $g_2(y) = y$ if $1 < y < 2$ and $g_2(y) = 1 + \exp(y - 2)$ otherwise.

$$g_2(y) = \begin{cases} \log(y) + 1 & \text{if } 0 < y \leq 1 \\ y & \text{if } 1 < y < 2 \\ 1 + \exp(y - 2) & \text{otherwise.} \end{cases}$$

For $\eta$ we either use $\eta_1(x) = \sin(x_5)$ with $x_5 \sim U[-\pi, \pi]$ or $\eta_2(x) = \sin(x_5) + \beta_1 x_1$ with $x_1 \sim U[-1, 1]$ and $\beta_1 = 1$. For $\sigma$ we define either no interaction by setting $\sigma = \sigma_1 := 1$ or using a data-dependent standard deviation by setting $\sigma = \sigma_2 := 1/x_6$ for $x_6 \sim U[1, 2]$. We generate 20 data sets for each combination of all possible specifications of $g$, $\eta$, $\sigma$ for $n = 500$ and $n = 3,000$. In addition to these configurations we also specify two further experiments where $\eta_3 = \exp(\sum_{j=1}^{4} x_j)(1 + \sin(x_5))$ for $x_1, \ldots, x_4 \sim U[-1, 1]$ and $\sigma = \sigma_2$ for both transformations $g_1$ and $g_2$.

For model estimation, $\boldsymbol{\mathcal{A}}$ contains the evaluated basis of a Bernstein polynomial of degree $M \in \{15, 25\}$ and $\boldsymbol{\mathcal{B}}$ either contains a vector of ones for the case of $\sigma_1$ or additionally $x_6$ for the case of $\sigma_2$. For the distribution shift $h_2$, we specify $\boldsymbol{\mathcal{C}}$ as the design matrix of a linear predictor function containing the feature $x_5$ encoded as a B-spline, for the case of $\eta_1$. For $\eta_2$, $\boldsymbol{\mathcal{C}}$ further includes the feature $x_1$ and, when using $\sigma_2$ for the DGP, also $x_6$.

### 2.3 Hyperparameters for the Numerical Experiments

All structured model terms in $h_2$ were specified as B-splines with 15 knots in the univariate case and with 50 knots in the bivariate case. The unstructured model components consist of a deep neural net with 4 fully-connected layers with tanh activation and dropout rates of 0.1. Each layer consists of 20 units.

### 2.4 Application

**Movie Reviews** For the TBMs as well as the DCTMs we specified a Bernstein Polynomial of degree $M = 25$. Further, for TBM-Shift we used a 10-fold Bootstrap with 300 iterations and learning rate 0.1 to find the optimal stopping iteration by choosing the minimum out-of-sample risks averaged over all folds. For TBM-Distribution we used a 10-fold Bootstrap with 700 iterations and learning rate 0.1. In the neural network layers were specified with ReLU activation.

**UTKFace** The CNN blocks consist of a 2D convolution, batch normalization, max pooling and dropout with 16,32,32 filters, kernel sizes of (3,3), ReLU activations and dropout rates of 0.25. The first block has a pool size of (3,3), the second and third of (2,2).

**Benchmark Study** For all 4 data sets, the unstructured neural network predictors takes all tabular features as inputs and, if not specified otherwise, tanh activation is used. The optimal number of epochs for each training is found by 5-fold cross-validation. For the normalizing flows we take a (non-trainable) standard Gaussian base distribution with a varying number of radial flows [1]. The parameters of the flows are learned with tanh activations and are parameterized as in [4]. Further details for each data set are as follows:

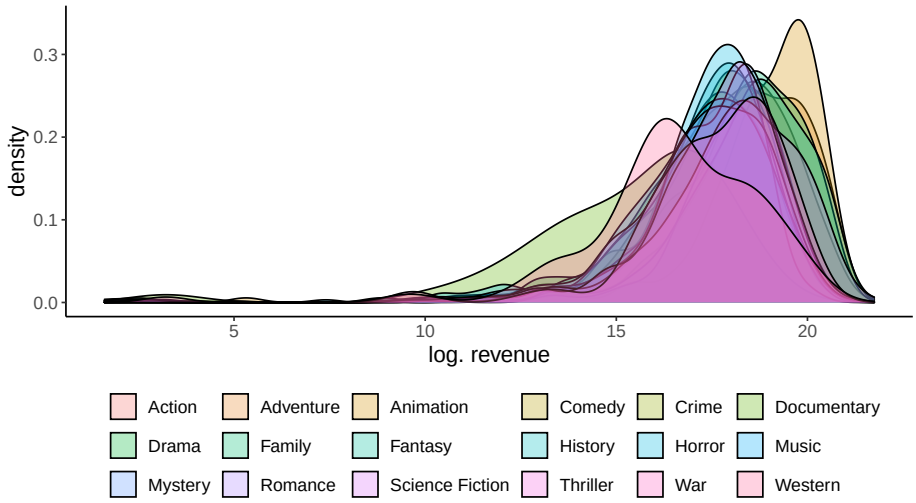


**Fig. 2.** Distribution of the logarithmic revenue for different genres in the movie data set.

*Forest Fire* Only an unstructured and no structured predictor for $h_1$ as well as $h_2$ are specified. The unstructured predictor is a deep neural network consisting of 2 fully-connected layers with 14 and 4 units.

*Diabetes* An unstructured predictor is specified for $h_1$ as well as $h_2$, while $h_2$ is additionally equipped with a structured term (univariate thin-plate regression spline) for a single tabular feature. The unstructured predictor is a shallow neural network consisting of 1 fully-connected layers with 4 units.

*Boston* Two different unstructured predictor were specified for $h_1$ and $h_2$. The shift term, $h_2$, is additionally equipped with structured terms taking all tabular features together with 11 features which additionally enter as univariate thin-plate regression splines. The unstructured predictor for $h_1$ is a shallow neural network consisting of 1 fully-connected layer with 2 units. For $h_2$, a deep neural network with 3 fully-connected layers (32,16 and 4 units) is specified.

*Airfoil* An unstructured predictor is specified for $h_1$ as well as $h_2$ together with an univariate thin-plate regression spline for 3 tabular features. The unstructured predictor is a deep neural network consisting of 2 fully-connected layers with 16 and 4 units.

## 3 Additional Experiments

The following plot contains the RIMSE results for 6 out of 10 DGPs which were not reported in the main text. Note that the results now also comprise DGPs

that were solely learned by unstructured model terms. These are DGPs where $\eta_3$ is contained (i.e., row 3 and 6).

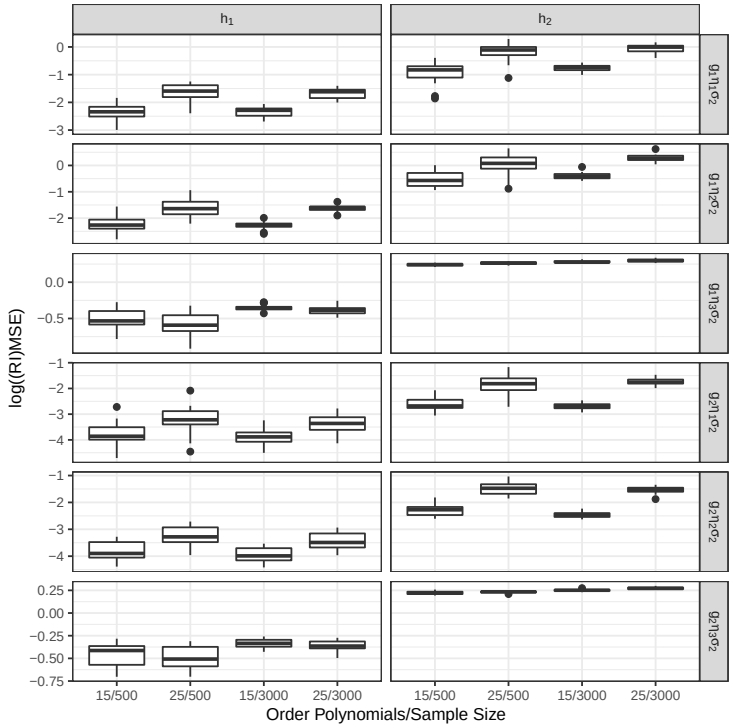


**Fig. 3.** The log(RIMSE) for DCTMs for the 6 remaining DGPs (DGPs in rows), different orders of the Bernstein polynomial and different sample sizes for 20 runs.

## 4 Statistical Inference

Obtaining statistical inference (p-values, confidence intervals) for estimated regression coefficients is not straightforward for neural-based model estimation. Current Bayesian approaches suffer proper coverage (see, e.g., [3]) and for techniques like dropout, the overoptimism of post-selection inference (see, e.g., [2]) would have to be taken into account. This is an interesting topic for future research, in particular because TBMs also face the same challenges.

## References

1. Rezende, D., Mohamed, S.: Variational inference with normalizing flows. Proceedings of Machine Learning Research **37**, 1530–1538 (2015)

2. Rügamer, D., Greven, S.: Selective inference after likelihood- or test-based model selection in linear models. Statistics & Probability Letters **140**, 7–12 (2018)
3. Rügamer, D., Kolb, C., Klein, N.: A Unified Network Architecture for Semi-Structured Deep Distributional Regression. arXiv preprint arXiv:2002.05777 (2020)
4. Trippe, B.L., Turner, R.E.: Conditional density estimation with bayesian normalising flows (2018)