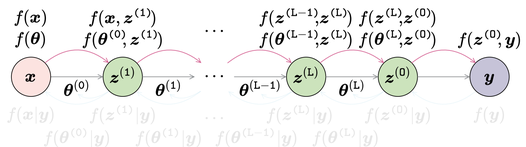
Implementation of a probabilistic Softmax for BNNs (Group 11)

David Wardan, Khalil Sabri, and Miquel Florensa – Polytechnique Montréal, Canada

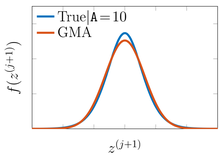
(Adapted from the work proposed by Nguyen & Goulet)

Tractable Approximate Gaussian Inference (TAGI)

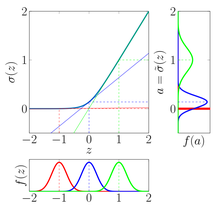
TAGI - Forward uncertainty propagation



Challenge 1: GMA



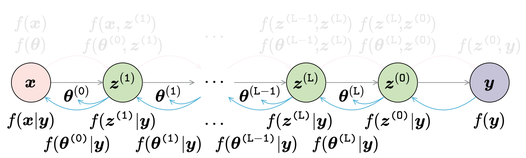
Challenge 2: Locally Linearize



$$
\begin{aligned}
\boldsymbol{z}^{(1)} &= \boldsymbol{w}^{(0)}\boldsymbol{x} + \boldsymbol{b}^{(0)}, \\
\boldsymbol{a}^{(i)} &= \bar{\sigma}(\boldsymbol{z}^{(i)}), \\
\boldsymbol{z}^{(i)} &= \boldsymbol{w}^{(i)}\boldsymbol{a}^{(i-1)} + \boldsymbol{b}^{(i)}, \\
\boldsymbol{z}^{(0)} &= \boldsymbol{w}^{(\mathrm{L})}\boldsymbol{a}^{(\mathrm{L})} + \boldsymbol{w}^{(\mathrm{L})} \\
\boldsymbol{y} &= \boldsymbol{z}^{(0)} + \boldsymbol{v}.
\end{aligned}
$$

where $\boldsymbol{v}$ are the observation noise.

TAGI - Backward parameter and hidden state inference



First, given the observations $\boldsymbol{y}$, the output layer is updated as

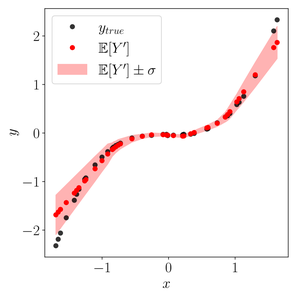
$$
\begin{aligned}
f\left(\boldsymbol{z}^{(0)} \mid \boldsymbol{y}\right) &= \mathcal{N}\left(\boldsymbol{z}^{(0)}; \boldsymbol{\mu}_{Z^{(0)}|y}, \boldsymbol{\Sigma}_{Z^{(0)}|y}\right) \\
\boldsymbol{\mu}_{Z^{(0)}|y} &= \boldsymbol{\mu}_{Z^{(0)}} + \boldsymbol{\Sigma}_{YZ^{(0)}}^{\top}\boldsymbol{\Sigma}_{Y}^{-1}\left(\boldsymbol{y} - \boldsymbol{\mu}_{Y}\right) \\
\boldsymbol{\Sigma}_{Z^{(0)}|y} &= \boldsymbol{\Sigma}_{Z^{(0)}} - \boldsymbol{\Sigma}_{YZ^{(0)}}^{\top}\boldsymbol{\Sigma}_{Y}^{-1}\boldsymbol{\Sigma}_{YZ^{(0)}}.
\end{aligned}
$$

Next, the hidden units and parameters of the layer i^{th} are updated using a layer-wise procedure as

$$
\begin{aligned}
f(\boldsymbol{z} \mid \boldsymbol{y}) &= \mathcal{N}\left(\boldsymbol{z}; \boldsymbol{\mu}_{Z|y}, \boldsymbol{\Sigma}_{Z|y}\right) \\
\boldsymbol{\mu}_{Z|y} &= \boldsymbol{\mu}_{Z} + \mathbf{J}_{Z}\left(\boldsymbol{\mu}_{Z^{+}|y} - \boldsymbol{\mu}_{Z^{+}}\right) \\
\boldsymbol{\Sigma}_{Z|y} &= \boldsymbol{\Sigma}_{Z} + \mathbf{J}_{Z}\left(\boldsymbol{\Sigma}_{Z^{+}|y} - \boldsymbol{\Sigma}_{Z^{+}}\right)\mathbf{J}_{Z}^{\top} \\
\mathbf{J}_{Z} &= \boldsymbol{\Sigma}_{ZZ^{+}}\boldsymbol{\Sigma}_{Z^{+}}^{-1},
\end{aligned}
\qquad
\begin{aligned}
f(\boldsymbol{\theta} \mid \boldsymbol{y}) &= \mathcal{N}\left(\boldsymbol{\theta}; \boldsymbol{\mu}_{\theta|y}, \boldsymbol{\Sigma}_{\theta|y}\right) \\
\boldsymbol{\mu}_{\theta|y} &= \boldsymbol{\mu}_{\theta} + \mathbf{J}_{\theta}\left(\boldsymbol{\mu}_{Z^{+}|y} - \boldsymbol{\mu}_{Z^{+}}\right) \\
\boldsymbol{\Sigma}_{\theta|y} &= \boldsymbol{\Sigma}_{\theta} + \mathbf{J}_{\theta}\left(\boldsymbol{\Sigma}_{Z^{+}|y} - \boldsymbol{\Sigma}_{Z^{+}}\right)\mathbf{J}_{\theta}^{\top} \\
\mathbf{J}_{\theta} &= \boldsymbol{\Sigma}_{\theta Z^{+}}\boldsymbol{\Sigma}_{Z^{+}}^{-1},
\end{aligned}
$$

where $\{\boldsymbol{\theta}^{+}, \boldsymbol{Z}^{+}\}$ is the short-hand notation of $\{\boldsymbol{\theta}^{(j+1)}, \boldsymbol{Z}^{(j+1)}\}$, and $\{\boldsymbol{\theta}, \boldsymbol{Z}\}$ is the short-hand notation of $\{\boldsymbol{\theta}^{(j)}, \boldsymbol{Z}^{(j)}\}$.

Regression toy example



- We re-implement the toy example presented by the authors using the numpy library in Python.
- This includes defining a fully connected neural network with one layer and 40 hidden units.
- The figure shows the output of the model on the test set with the epistemic uncertainty.

COLUMN 2 (Remax)

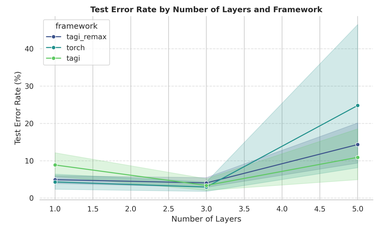
Key Results: TAGI-HRC vs. TAGI-Remax vs. Torch

Framework	Avg. Training Error	Avg. Test Error
Torch	10.05	10.69
TAGI-HRC	7.07	**7.73**
TAGI-Remax	7.14	**7.78**

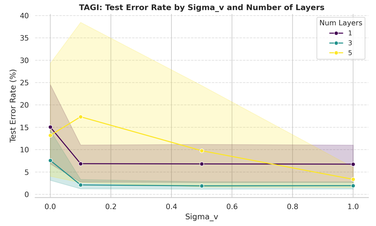
Important Findings

- **Batch Size:** TAGI excels with smaller batches, while Torch performs better with larger batches.
- **Network Depth:** TAGI maintains stability as depth increases, unlike Torch, which struggles without batch normalization.
- **Effect of σ_v:** Higher σ_v values enhance TAGI's performance, especially in deeper networks.
- **Capacity Effect:** TAGI benefits more from increasing neurons/channels, showing improved generalization, while Torch fails to utilize additional capacity effectively.
- **Comparison of Implementations:** TAGI-HRC achieves better error rates overall, while TAGI-Remax shows higher stability but less frequent best-case performance.
- **Batch Normalization:** TAGI performs well even without batch normalization, handling deeper architectures (5+ layers) where Torch often diverges.

Ablation Study: Performance Comparisons and Capacity Effects



Test Error vs Network Depth: TAGI maintains low error rates, while Torch degrades significantly.



Impact of σ_v: Higher σ_v values improve TAGI's test error.

Framework	Small Batch (16) With BN	Small Batch (16) W.o BN	Large Batch (512) With BN	Large Batch (512) W.o BN
Torch	**1.35**	31.20	**1.06**	16.22
TAGI-HRC	1.97	**1.74**	4.11	15.99
TAGI-Remax	12.05	7.30	5.75	**5.43**

Framework	Small (32) Neurons/layer	Large (512) Neurons/layer
Torch	7.03	7.32
TAGI-HRC	13.58	**8.98**
TAGI-Remax	9.25	**6.91**

py/cuTAGI – Open-source Bayesian deep-learning framework

cuTAGI

`github.com/lhnguyen102/cuTAGI`

`pip install pytagi`

- **Performance-Oriented Kernels** written in C++/CUDA from scratch, with pybind11 for a seamless integration. It allows running on CPU & CUDA devices through a Python API.
- **Broad Architecture Support** of the basic DNN layer including *Linear*, *CNNs*, *Transposed CNNs*, *LSTM*, *Average pooling*, *Batch* and *Layer normalization*, enabling the building of mainstream architectures such as *Autoencoders*, *Transformers*, *Diffusion Models*, and *GANs*.
- **Model Building and Execution** currently supports sequential model building, with plans to introduce Eager Execution
- **Open Platform** providing access to its entire codebase.

TAGI-related references

- *Coupling LSTM Neural Networks and SSM through Analytically Tractable Inference*, (Vuong, Nguyen & Goulet, International Journal of Forecasting, 2024)
- ***Analytically tractable hidden-states inference in Bayesian neural networks*** (Nguyen and Goulet. Journal-to-conference track, ICLR 2024)
- *Analytically tractable heteroscedastic uncertainty quantification in Bayesian neural networks for regression tasks* (Deka, Nguyen & Goulet. Neurocomputing, 2024)
- ***Tractable approximate Gaussian inference for Bayesian neural networks*** (Goulet, Nguyen, & Amiri, JMLR, 2021)