Xintong Li
xintong.li1@uwaterloo.ca

Dept. Statistics and Actuarial Science
University of Waterloo

Resilient Machine Learning Approaches for Fast Risk Evaluation and Management of Financial Portfolios and Variable Annuities

Supervised by Prof. Tony Wirjanto and Prof. Mingbin Feng

Thesis Defense
University of Waterloo
2025

Outline

Nested Simulation Procedures

Nested simulation procedures are necessary for **complex** financial derivatives and insurance products.

$$\rho(L) = \rho(L(X)), \quad L(X) = \mathbb{E}\left[Y|X=x\right]|_{x=X}.$$

Involves two levels of Monte Carlo simulations:

- Outer: underlying risk factors, $X_i \sim F_X$
- Inner: scenario-wise losses, $Y_{ij} \sim F_{Y|X_i}$

With an expensive total simulation budget $\Gamma = M \cdot N$:

$$\hat{L}_{N,i} = \frac{1}{N}\sum_{j=1}^{N} Y_{ij}; \quad Y_{ij} \sim F_{Y|X_i}$$

- Uses inner sample mean to estimate $L(X_i)$.

Metamodeling Approach

We focus on procedures that **pool** with **supervised learning metamodels**.

- Treat inner simulation as a black-box function
- Approximate $L(\cdot)$ with $\hat{L}^{\mathsf{SL}}_{M,N}(\cdot)$
- Train with feature-label pairs generated by simulation:

$$\{(X_i, \hat{L}_{N,i}) | i = 1, \ldots, M, j = 1, \ldots, N\}$$

- Use metamodel predictions to estimate risk measures

There are **computational costs** associated with pooling inner replications.

Asymptotic Convergence Rates of Different Procedures

$$\min_{M,N} \quad \mathbb{E}\left[(\hat{\rho}_{M,N} - \rho)^2\right]$$

$$\text{subject to } M \cdot N = \Gamma$$

Procedures	Smooth h	Hockey-Stick h	Indicator h
Standard Procedure	$\mathcal{O}(\Gamma^{-2/3})$	$\mathcal{O}(\Gamma^{-2/3})$	$\mathcal{O}(\Gamma^{-2/3})$
Regression	$\mathcal{O}(\Gamma^{-1})$	$\mathcal{O}(\Gamma^{-1+\delta})$	No Result
Kernel Smoothing	$\mathcal{O}(\Gamma^{-\min(1,4/(d+2))})$		
Kernel Ridge Regression[1]	$\mathcal{O}(\Gamma^{-1})$		
Likelihood Ratio	$\mathcal{O}(\Gamma^{-1})$		

Key Theoretical Results

Contribution: bridging the gap between MSE and absolute error convergence.

- Convergence in MSE:

$$\mathbb{E}\left[(\hat{\rho}_\Gamma - \rho)^2\right] = \mathcal{O}\left(\Gamma^{-\xi}\right)$$

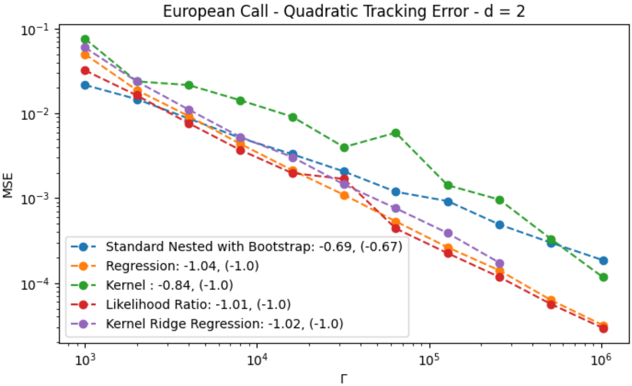
- Convergence in Probabilistic Order:

$$|\hat{\rho}_\Gamma - \rho| = \mathcal{O}_{\mathbb{P}}(\Gamma^{-\xi})$$

Theorem

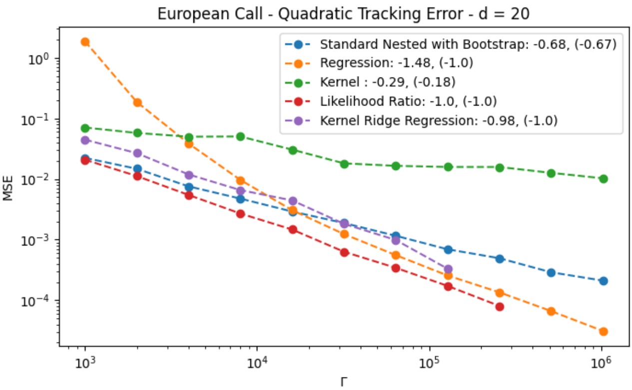
If $\hat{\rho}_\Gamma$ converges in MSE to ρ in order ξ, then $\hat{\rho}_\Gamma$ converges in probabilistic order to ρ in order $\frac{\xi}{2}$.

Finite-Sample Performance



- Standard, KRR, and likelihood ratio procedures are dimension-independent
- Kernel smoothing and regression show sensitivity to dimension, but in different ways

Sensitivity to Asset Dimension



- Regression and kernel converge faster than their asymptotic rates
- Results of other experiments are in the thesis (Figure 2.3 - Figure 2.11)

Additional Computational Costs

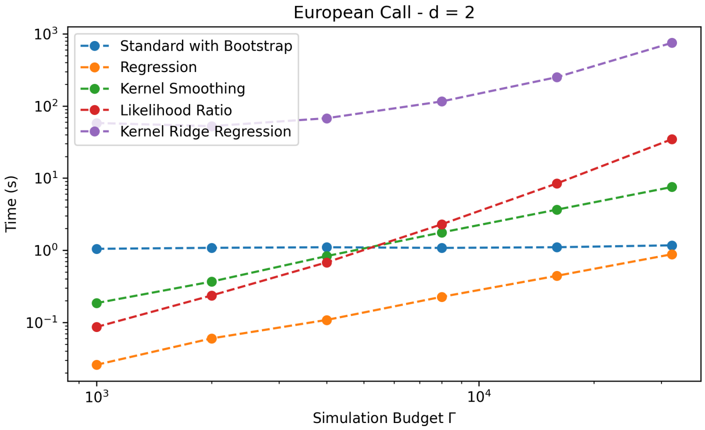


Figure: Additional computation time for different procedures

Conclusion

Regression-based nested simulation procedure:

- Most robust and stable for limited budgets
- Efficient to implement
- Fast empirical convergence for option portfolios

For high-dimensional or complex payoffs:

- Difficult to find a good regression basis
- Neural network-based procedures may be more suitable

Next project: examining performance of metamodel-based simulation procedures for variable annuities

Nested Simulation for Risk Management of VAs

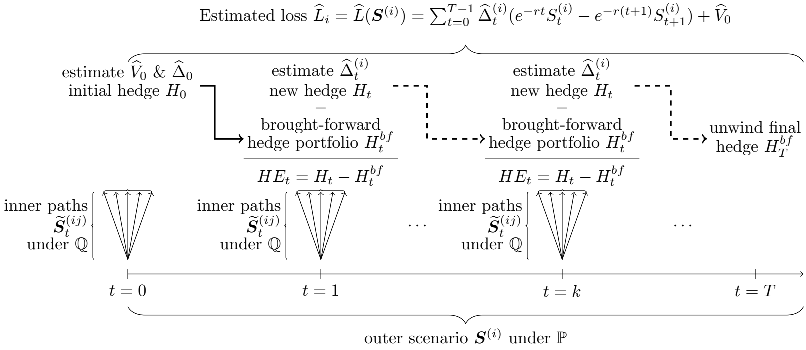


Figure: Illustration of nested simulation that estimates the P&L for one outer scenario

Metamodel-based Nested Simulation

We use deep neural networks (DNNs) as metamodels

- Use LSTMs for sequential data
- **Challenge**: lack of transparency and interpretability

Research Contributions:

1. Propose two generic DNN-based nested simulation procedures
   - Accurate tail scenario identification
   - Significant computational savings by **budget concentration**
2. Study noise tolerance of DNNs using simulated data
   - **Control noise levels** by adjusting simulation parameters
   - Provide direct evidence on transparency and interpretability

Two-Stage Metamodel-based Nested Simulation

Algorithm Two-Stage Metamodel-based Nested Simulation for VAs

1: **Generate training data for metamodels**
2: **Train metamodels**
3: **Estimate α-CVaR with extensive simulation on predicted tail scenarios**
- Concentrate simulation on predicted tails

Key Findings:

- Substantial computational savings ($70\% - 85\%$ reduction)
- Some DNN metamodels make **accurate loss predictions**

Single-Stage Metamodel-based Nested Simulation

Algorithm Single-Stage Metamodel-based Nested Simulation for VAs

1: **Generate training data for metamodels**
2: **Train metamodels**
3: **Estimate α-CVaR with metamodel predictions**
- Entirely avoids extensive simulation

Key advantages:

- more efficient than a two-stage procedure;
- avoids specifying a safety margin m.

Experiment Setting

We consider the following metamodel architectures:

Metamodel	Abbreviation	Capacity
Multiple Linear Regression	MLR	241
Quadratic Polynomial Regression	QPR	481
Feedforward Neural Network	FNN	35,009
Recurrent Neural Network	RNN	32,021
Long Short-Term Memory	LSTM	35,729

Table: Metamodel architectures for GMWB inner simulation model

In our experiment:

- 95%-CVaR of loss for a GMWB with a 240-month maturity
- 240-dimensional feature vector and 1-dimensional loss

Metamodel Performance

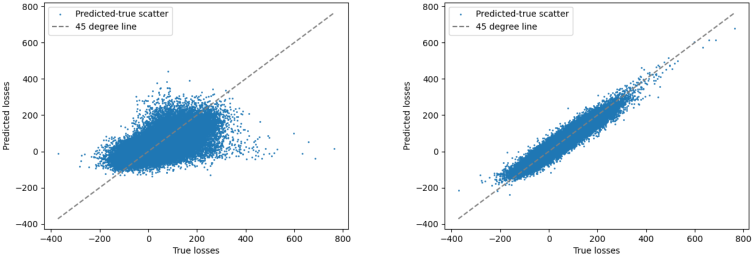
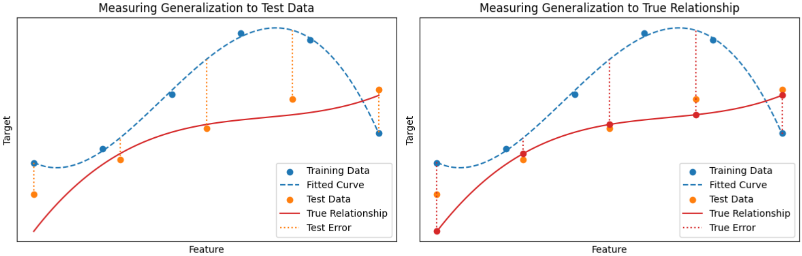


Figure: QQ plots between true and predicted loss labels for QPR and LSTM metamodels

- QPR make **inaccurate** loss predictions
- DNN metamodels are more flexible.

Research Questions

- What do DNNs learn from noisy data?
- How well do DNNs learn from noisy data?



Three datasets are considered:

- Training: 90,000 scenarios (100 inner replications).
- Test: 10,000 scenarios (100 inner replications).
- True: 100,000 scenarios (100,000 inner replications).

Metamodel Performance on Different Datasets

Metamodel	Training error	Test error	True error
MLR	0.706 $(\pm 8.3 \times 10^{-4})$	0.713 $(\pm 2.7 \times 10^{-2})$	0.706 $(\pm 3.4 \times 10^{-4})$
QPR	0.543 $(\pm 8.3 \times 10^{-4})$	0.554 $(\pm 2.7 \times 10^{-2})$	0.544 $(\pm 4.1 \times 10^{-4})$
FNN	0.129 $(\pm 6.0 \times 10^{-3})$	0.240 $(\pm 9.8 \times 10^{-3})$	0.132 $(\pm 5.8 \times 10^{-3})$
RNN	0.132 $(\pm 7.5 \times 10^{-3})$	0.137 $(\pm 7.6 \times 10^{-3})$	0.119 $(\pm 7.5 \times 10^{-3})$
LSTM	0.075 $(\pm 4.5 \times 10^{-3})$	0.079 $(\pm 5.4 \times 10^{-3})$	0.063 $(\pm 4.4 \times 10^{-3})$
RNN*[2]	0.109 $(\pm 5.2 \times 10^{-3})$	0.128 $(\pm 5.2 \times 10^{-3})$	0.109 $(\pm 5.2 \times 10^{-3})$

Table: Average MSEs and 95% confidence bands of metamodels for GMWB.

- RNN-based metamodels have lower true errors than their training errors.
- DNN metamodels with suitable architectures **cut through the noise** in training labels.

[2] This row summarizes the results of the well-trained RNNs.

Safety Margin

Consider estimating the 95% CVaR with 100,000 outer scenarios.

Figure: Illustration: a safety margin of 4% ($m = 9000$)

Trade-off between accuracy and efficiency

- Lower margin: not enough tail identified
- Higher margin: more budget spent on extensive simulation

Metamodel Performance

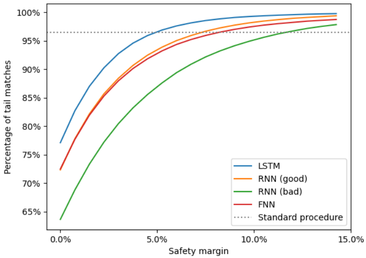


Figure: Tail Matches and CVaR Predictions for DNN Metamodels

- Traditional regression metamodels are **unable** to accurately identify tail scenarios even with high safety margins.
- LSTM metamodels surpasses the standard procedure with 5% safety margin.

Sensitivity Testing for DNNs

Simulation controls **noise level in training labels** and **number of training samples**.

N' varies the noise level.

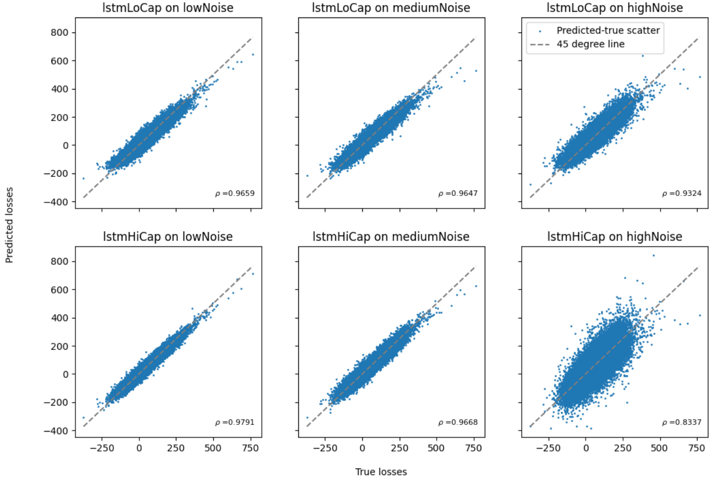
- **Low noise labels**: $N' = 100$
- **Medium noise labels**: $N' = 10$
- **High noise labels**: $N' = 1$

M varies the number of training samples.

- $M \in \{10^2, 10^3, 10^4, 10^5\}$

2 LSTMs of **different capacities** are examined based on their MSEs.

Noise Tolerance of DNNs



Sensitivity of High-capacity LSTM

	$N' = 1$	$N' = 10$	$N' = 100$	$N' = 1000$
$M = 100$	0.764	0.408	0.131	0.087
$M = 1000$	0.878	0.367	0.156	0.087
$M = 10000$	0.351	0.147	0.064	0.063
$M = 100000$	0.149	0.065	0.060	0.038

Table: MSE between high-capacity LSTM's predicted losses and true losses.

- Same color $\rightarrow$ same total simulation budget.
- $N' = 10$ is a reasonable budget allocation for LSTM metamodels.

Single-Stage Procedure

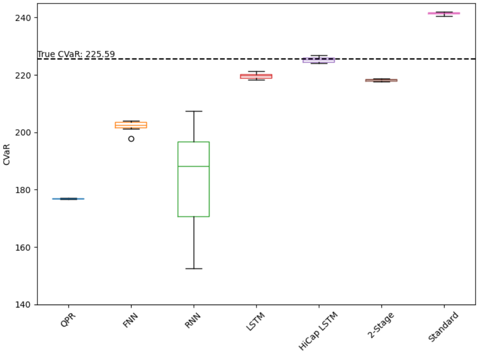


Figure: CVaR estimates of single-stage procedures with $N' = 10$.

- The single-stage procedure outperforms the two-stage procedure.
- $N' = 10$ is a reasonable budget allocation.

Convergence Analysis

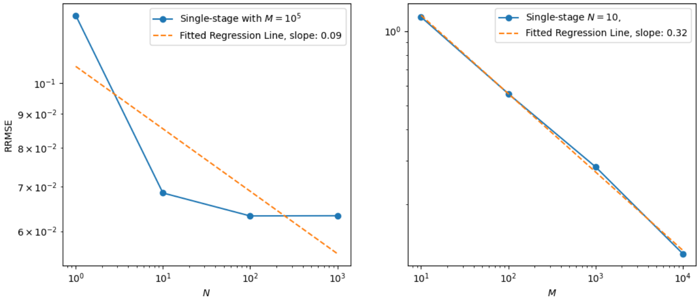


Figure: Empirical convergence of the single-stage procedure with a LSTM metamodel.

- Minimal effect of increasing N' on CVaR estimation.
- For a given Γ, set N' constant and allocate budget to outer simulations.

Conclusion

Key Findings:

- LSTMs are **resilient** to moderate levels of noise in training labels.
- DNNs can learn **true** complex dynamic hedging model.
- Two-stage procedure addresses regulatory concerns.
- Single-stage procedure is **efficient**.

Future Directions:

- Apply DNNs to other risk management tasks.
- Fast adaptation to new contracts/market conditions.

Transfer Learning for Rapid Adaptation of DNN Metamodels

Problem: Updating models for new conditions is expensive.

- Full retraining takes too much time.
- New VA contracts need quick model updates.
- Need balance between speed and accuracy.

Solution: Transfer learning for faster model adaptation.

- Train first on existing contract data.
- Update with small amount of new contract data.
- Reuse knowledge between similar contracts.
- Benefits:
  - Faster training.
  - Less data needed.

Transfer Learning Framework

Key Components:

- **Domain** $\mathcal{D}$: feature space $\mathcal{X}$ + probability distribution F
- **Task** $\mathcal{T}$: label space $\mathcal{Y}$ + predictive function $f : \mathcal{X} \to \mathcal{Y}$

Source vs. Target:

- **Source:** $\mathcal{D}_{\mathsf{So}} = \{\mathcal{X}_{\mathsf{So}}, F_{\mathsf{So}}(X)\}$
- **Target:** $\mathcal{D}_{\mathsf{Ta}} = \{\mathcal{X}_{\mathsf{Ta}}, F_{\mathsf{Ta}}(X)\}$

Our Goal:

- **Input features** X: risk factors from outer simulation
- **Output labels** L: contract losses
- **Source and target**: from VAs with abundant simulation data to new VAs with limited data
- **Goal**: improve $f_{\mathsf{Ta}}(\cdot)$ using knowledge from $\mathcal{D}_{\mathsf{So}}$ and $f_{\mathsf{So}}(\cdot)$

Transfer Learning Techniques

Common Techniques:

- **Fine-tuning:** a model pre-trained on a source task is used as a starting point for a target task.
- **Layer freezing:** only part of the model is fine-tuned.

Key considerations: similarity between source and target tasks

Experiment Design:

- Source domain (50000 samples): GMMB with no lapse and GMMB with static lapse
- Target domain (2000 samples): GMMB and GMWB with dynamic lapse

Results: Accuracy Comparison

Method and Setting	MSE
Extensive Training (Dynamic)	**0.0587**
Fine-tuning (Static)	0.0794
Layer Freezing (Static)	0.0763
Layer Freezing (No Lapse)	0.3361
Fine-tuning (No Lapse)	0.4894
Without TL (Dynamic)	0.2950

Table: Comparison of different TL methods on GMMB contracts (best MSE values)

Learning Curves: Effect of Similarity

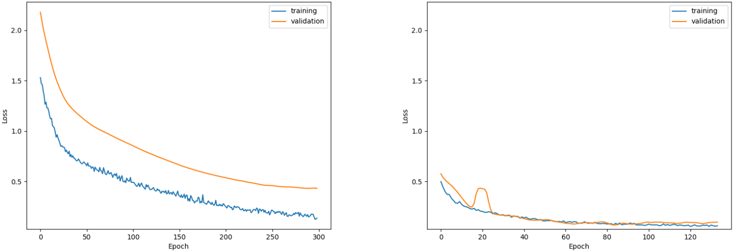
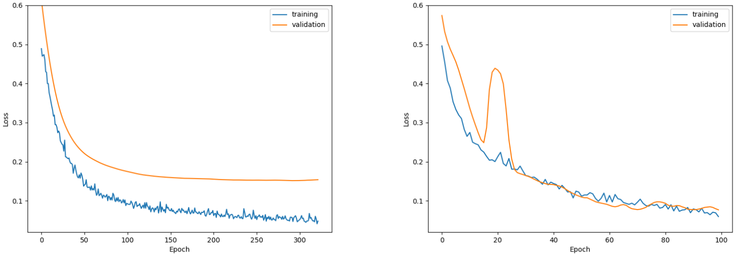


Figure: Fine-tuning performance on GMMB with dynamic lapse

- Similarity between source and target domain matters
- Negative transfer can happen for dissimilar tasks

Learning Curves: Effect of Freezing Layers



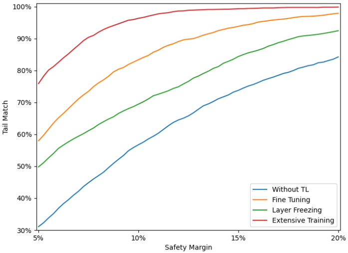
(a) Freezing LSTM layers

(b) Freezing fully connected layers

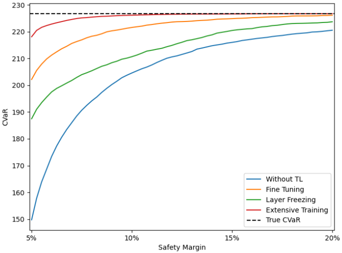
Figure: Learning dynamic lapse from a GMMB with static lapse

- When learning dynamic lapse, freezing the LSTM layers is suboptimal.
- Freezing the fully connected layers leads to better performance.

Metamodel Performance



(a) Identifying tail scenarios



(b) CVaR estimation

Figure: Comparison of different TL methods on transferring to GMWB contracts

Conclusions

- Transfer learning significantly improves metamodeling for VA contracts:
  - Faster training convergence
  - Better prediction accuracy
  - Reduced computational requirements
- Enables more frequent risk assessments and faster decision-making

Future Work:

- Incorporating domain knowledge into transfer process
- Extension to other insurance and financial products
- Multi-task learning

References I

Broadie, M., Du, Y., and Moallemi, C. C. (2015). Risk estimation via regression. *Operations Research*, 63(5):1077–1097.

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*. `https://arxiv.org/abs/1412.3555`. Accessed 7$^{\mathrm{th}}$ Sep 2023.

Dang, O. (2021). *Efficient Nested Simulation of Tail Risk Measures for Variable Annuities*. Ph.D. thesis, Department of Statistics and Actuarial Science, University of Waterloo, Waterloo, ON, Canada. `https://uwspace.uwaterloo.ca/handle/10012/17084`.

Feng, M. and Song, E. (2020). Optimal nested simulation experiment design via likelihood ratio method. *arXiv preprint arXiv:2008.13087*.

Gordy, M. B. and Juneja, S. (2010). Nested simulation in portfolio risk measurement. *Management Science*, 56(10):1833–1848.

Hardy, M. R. (2001). A regime-switching model of long-term stock returns. *North American Actuarial Journal*, 5(2):41–53.

References II

Hardy, M. R. (2003). *Investment Guarantees: Modeling and Risk Management for Equity-Linked Life Insurance*, volume 168. John Wiley & Sons.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Hong, J. L., Juneja, S., and Liu, G. (2017). Kernel smoothing for nested estimation with application to portfolio risk measurement. *Operations Research*, 65(3):657–673.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K., editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Wang, W., Wang, Y., and Zhang, X. (2022). Smooth nested simulation: bridging cubic and square root convergence rates in high dimensions. *arXiv preprint arXiv:2201.02958*.

Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014). How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27.

References III

Zhang, K., Feng, M., Liu, G., and Wang, S. (2022). Sample recycling for nested simulation with application in portfolio risk measurement. *arXiv preprint arXiv:2203.15929*.

Zhang, K., Liu, G., and Wang, S. (2021). Bootstrap-based budget allocation for nested simulation. *Operations Research*.