Neural Networks for Risk Management in Life insurance

Insurance Data Science Conference 2019

Lucio Fernandez-Arjona

University of Zurich and Zurich Insurance Company

June 2019

Starting point

Research question

Can neural networks improve proxy modelling for risk management in Life insurance?

- We will approach this question with a machine-learning engineering mindset, looking for "what works" and focusing on measuring results for a true predictive model.
- For a more developed and robust mathematical framework, look for the upcoming paper "Machine learning for pricing and risk management", joint work with Prof. Damir Filipovic (EPFL & SFI).

Outline

1. Problem definition
2. Current models in use in the industry
3. Proposed Neural Network model
4. Numerical example

A problem of nested calculations

- In Life insurance the value of the asset-liability portfolio is calculated using option pricing theory.
- Due to the complexity of the derivative, no closed formulas are available.

$$V_t = E_t^{\mathbb{Q}}\big[\sum_{\tau>t} CF_\tau(X)\big] \approx V_{MC} = \frac{1}{N}\sum_{j=1}^{N}\sum_{\tau>t} CF_\tau(X_{t:T}^{(j)}|X_{0:t})$$

X: interest rates, equity markets, mortality rates, etc.

- X depends on a smaller set of normal random drivers ξ, ie $X = X(\xi)$.
- $X_{0:t}$ is known at t, and what is simulated after t is called $X_{t:T}$.

A problem of nested calculations

- What happens when attempting to calculate complex risk metrics?
- For example: Value at Risk or Expected Shortfall

$$VaR_\alpha(V) = -\inf\left\{v : F_V(v) > \alpha\right\} = F^{-1}_{-V}(1-\alpha)$$
$$ES_\alpha(V) = -\frac{1}{\alpha}\int_0^\alpha \mathrm{VaR}_\gamma(V)d\gamma$$

- if F_V^{-1} is not known, then we must simulate $\{V_t^{(i)}\}_{i=1:M}$ and then calculate the risk metric on the empirical (simulated) distribution.

When V_t is not known in closed form, $\{V_t^{(i)}\}_{i=1:M}$ must be approximated by some $\{\hat{V}_t^{(i)}\}_{i=1:M}$

Approximating the value function

- Nested Monte Carlo

$$\hat{V}_t^{(i)} = V_{MC}^{(i)} = \frac{1}{N} \sum_{j=1}^{N} \sum_{\tau > t} CF_\tau(X_{t:T}^{(j)} | X_{0:t}^{(i)}) \quad i = 1, 2, ..., M$$

  This approach is not feasible when $CF_t(\cdot)$ is slow to calculate as it's usually the case with complex products.

- Proxy model (regress-later type - cash flows function approximation)

$$\hat{V}_t^{(i)} = V_{pxy}^{(i)} = E_t^{\mathbb{Q}}[\sum_{\tau > t} \widehat{CF}_\tau(X)]$$

Polynomial curve-fitting approach

- The polynomial curve-fitting approach uses

$$V_t^{(i)}(X) \approx \widehat{V}_t^{(i)}(X) = \sum_k w_k \phi_k(X_{0:t}^{(i)})$$

- The approximation is based on a linear regression of $\tilde{V}_{MC}^{(i)}$ against $\{\phi_k(\cdot)\}$, a polynomial basis. $\tilde{V}_{MC}^{(i)}$ differs from $V_{MC}^{(i)}$ in that it is calculated with a very low number of inner simulations, N.
- $\widehat{V}_t$ is an estimator of $E_t^{\mathbb{Q}}\left[\sum_{\tau>t} \widehat{CF}_\tau(X)\right]$ directly, not of $CF_\tau(X)$.
- This approach is also called Least-Squares Monte-Carlo and it is an example of a regress-now estimator (Pelsser and Schweizer, 2016).

Replicating Portfolio approach

- The replicating portfolio approach uses

$$CF_\tau(X) \approx \widehat{CF}_\tau(X) = \sum_k w_k \phi_k(X_{0:\tau})$$

- The approximation is based on a linear regression at each τ of $CF_\tau(\cdot)$ against $\{\phi_{k,\tau}(\cdot)\}$, the cash functions of a set of financial instruments (bonds, swaps, equity options).
- $E_t^{\mathbb{Q}}[\phi_k(\cdot)]$ is known in closed-form or can be easily calculated.

$$\hat{V}_t^{(i)} = V_{pxy}^{(i)} = \sum_{\tau > t} \sum_k w_{k,\tau} E_t^{\mathbb{Q}} \left[\phi_{k,\tau}(X^{(i)}) \right]$$

A neural network approach

- The proposed neural network approach uses

$$\widehat{CF}_\tau(X) = f_\tau(X_{0:\tau}; W_\tau, \theta) \quad \textbf{or} \quad \widehat{CF}_\tau(X) = f_\tau(\xi_{0:\tau}; W_\tau, \theta)$$

  f_τ is a neural network with parameters W_τ and hyper-parameters θ

- For normally-distributed ξ, we can calculate $E_t^{\mathbb{Q}}$ for a single layer network with a ReLu activation function.

$$E_t^{\mathbb{Q}}[\max(\sum w_i \xi_i + b, 0)] = \frac{1}{2}\sigma\sqrt{\frac{2}{\pi}}e^{\mu^2/2\sigma^2} + \mu(1 - \Phi(-\frac{\mu}{\sigma}))$$

$$\mu = E_t^{\mathbb{Q}}[\sum w_i \xi_i + b]; \sigma = \sigma_t^{\mathbb{Q}}[\sum w_i \xi_i + b]$$

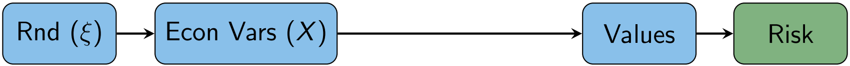
Machine learning meets financial engineering

Using this formula we can transform a cash-flow predictive model into a price predictive model

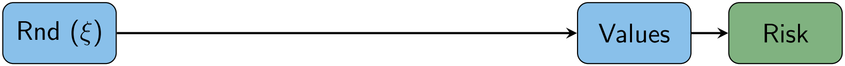
A neural network model is simpler than an RP model

This is the replicating portfolio prediction phase

This is the neural network on X ("nn econ") prediction phase



This is the neural network on ξ ("nn rand") prediction phase



Experimental set-up

- Typical Life liability model setup:
  - ESG
  - Insurance cash flows model
  - RP instruments cash flows and pricing functions
- Example based on portfolio with a "return premium on death" guarantee
- Scenario generator available open-source at https://gitlab.com/luk-f-a/EsgLiL
- All datasets used for this presentation are freely available in Mendeley Data
- Entirely written in Python. Using NumPy for array operations, pandas for data aggregation, scikit-learn for regressions and neural networks, joblib and Dask for parallelization.

Measuring quality

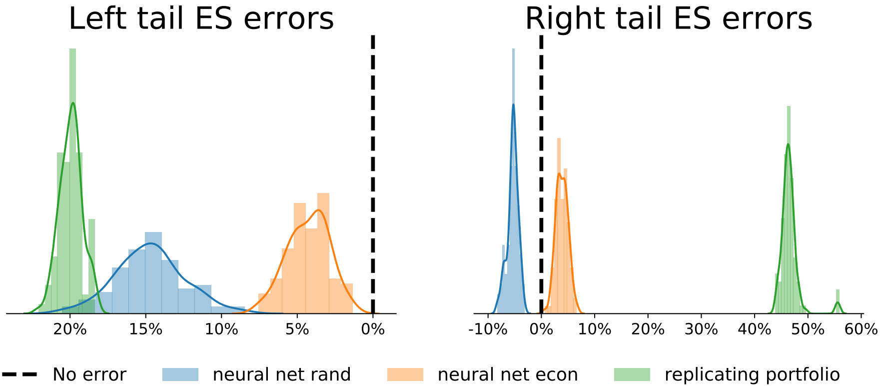
- **Mean Absolute Percentage Error (MApE) on risk metric**

$$\frac{1}{R}\sum_{i}^{R}\left|\frac{\hat{\rho}_i}{\rho}-1\right|$$

$\hat{\rho}_i$ is the proxy model estimation of ρ, the risk metric of concern (ρ is either the true ES or true VaR).
Model results are mean-centered before calculating risk metrics.
In all cases, the results presented are calculated using $R = 100$ macro-repetitions on the estimator.

Results of quality comparison

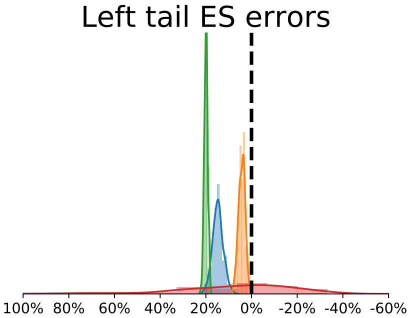
	Left ES	Left VaR	Mean	Right VaR	Right ES
Rep. Portfolio MApE	20%	23%	4%	46%	47%
Neural net (econ) MApE	4%	2%	2%	7%	4%
Neural net (rand) MApE	15%	11%	5%	4%	5%

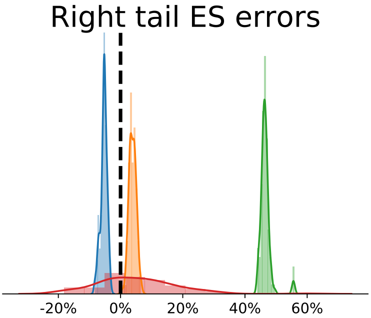


Neural networks used data more efficiently than nested Monte Carlo

When given a fixed "simulation budget", neural networks deliver more accurate results than using that budget for nested Monte Carlo.

	Left ES	Left VaR	Mean	Right VaR	Right ES
Nested MC MApE	19%	14%	2%	8%	10%
Neural net (econ) MApE	4%	2%	2%	7%	4%
Neural net (rand) MApE	15%	11%	5%	4%	5%





Conclusions

- We described the current state of proxy modelling focusing on a particular widely-used technique, replicating portfolios.
- We presented an alternative model based on a neural network approach and showed that
  - this model can be simpler than existing ones,
  - and the quality of risk calculations higher.
- Caveats: the comparison focused on one specific ESG, one specific insurance product and one specific implementation of the replicating portfolio technique.

Thank You