Deep hedging

Application of deep learning to hedge financial derivatives

mazars

Contents

04 Executive summary

04 Introduction

04 Methodology

- A. P&L modelling
- B. Role of neural networks
- C. Notion of risk measure and final formulation of the problem

06 Results under Black-Scholes Model

- A. Model assumptions
- B. Numerical results

07 Results under Heston Model

- A. Model assumptions
- B. Numerical results

08 Main limits of the deep hedging algorithm

09 Conclusion

09 References

Executive summary

The recent breakthrough of data science and deep learning make a model independent approach for hedging possible. This hedging approach known as deep hedging is a robust data-driven method able to consider market frictions as well as trading constraints without using model-computed greeks. This article gives the main theoretical tools to understand the methodology and presents examples of applications in different frameworks (Black-Scholes, Heston and a back-test on real data). The results of those applications show that deep hedging works well with data generated by complex models and can provide a relevant hedging strategy taking into account market constraints.

Keywords: *Deep Hedging, Deep Learning, Quantitative Finance, Greeks, Hedging strategy, Neural Networks, Heston and Black-Scholes models*

1. Introduction

Over the last years, quantitative finance has become a privileged data science application field. The increasing computing power along with the fast-growing volumetry of data available has made it possible to apply time-consuming and complex algorithms. Anomaly and fraud detections, predictive modelling for stocks and investment strategies, derivatives pricing are just a few examples of current applications.

Recently, banks set out to automate the hedging of financial derivatives. The objective is to replace classical hedging strategies that rely on the computation of risk sensitivities, known as greeks, by deep learning algorithms.

The idea of this approach is to no longer depend on those greeks or even models themselves based on *a priori* assumptions (*e.g.* absence of transaction costs). This new method known as deep hedging is theoretically entirely based on data and models hedging strategies with the use of neural networks. Training of the networks is performed with data input known as the training dataset. This dataset may contain classical market information such as prices of hedging instruments, bid-ask spreads or liquidity constraints, as well as other information like news analytics. The goal of the algorithm is to provide the best hedging strategy given the optimization of a risk metric such as the Value at Risk (VaR) or the Expected Shortfall (ES).

In this article, the theoretical tools are first presented (see references (1) and (2) for further details) in order to set a general framework of the deep hedging approach. Different applications and illustrations are then considered: Black-Scholes model without transaction costs, Heston model considering transaction costs, and a back-test approach on real market data. Finally, the pros and cons of deep hedging and the challenges ahead are discussed.

2. Methodology

A. P&L modelling

Assume that d hedging instruments (*e.g.* stocks but also vanilla options like call and put options, or even any other type of financial instrument) are available on the market. Those instruments are denoted by the stochastic process:

$$S \coloneqq (S_t)_{t\geq 0} \coloneqq \left(S_t^{(1)}, \dots, S_t^{(d)}\right)_{t\geq 0}$$

The objective is to hedge against a given liability/contingent claim Z which is completely known at a given time T. To do so, the hedger sets discrete times $0 = t_0 < t_1 < \cdots < t_n = T$ and looks for the best discrete stochastic process called hedging strategy and defined by:

$$\delta \coloneqq (\delta_k)_{0\leq k\leq n-1} \coloneqq \left(\delta_k^{(1)}, \dots, \delta_k^{(d)}\right)_{0\leq k\leq n-1}$$

Where δ_k is chosen knowing all market information up to time t_k. $\delta_{k-1}^{(j)}$ represents the number of hedging instruments $S^{(j)}$ owned by the hedger between times t_{k-1} and t_k. Between those two times, owning those instruments makes the portfolio value vary by:

$$\delta_{k-1} \cdot \left(S_{t_k} - S_{t_{k-1}}\right) \coloneqq \sum_{j=1}^{d} \delta_k^{(j)} \times \left(S_{t_k}^{(j)} - S_{t_{k-1}}^{(j)}\right)$$

In practice, buying and selling assets in financial markets implies transactions costs due to liquidity constraints such as bid-ask spreads. These costs must be considered by the trader to limit his losses. The time t_k costs are denoted by $c_k\left(S_{t_k}, \delta_k - \delta_{k-1}\right)$. They may depend on instrument prices S_{t_k} and the quantity of instruments bought at time t_k, *i.e.* $\delta_k - \delta_{k-1}$. A possible example of transaction costs is the proportional cost defined as follows:

$$c_k\left(S_{t_k}, \delta_k - \delta_{k-1}\right) \coloneqq \sum_{j=1}^{d} c_k^{(j)} \times \left(\delta_k^{(j)} - \delta_{k-1}^{(j)}\right) \times S_{t_k}^{(j)}$$

Thus, the self-financing condition (*i.e.* no additional cash is included to the portfolio) along with transaction costs and the contingent claim Z to hedge implies the following profit and loss (P&L):

$$(\delta \cdot S)_T - C_T(\delta) - Z \coloneqq \sum_{k=0}^{n} \delta_{k-1} \cdot \left(S_{t_k} - S_{t_{k-1}}\right) - c_k\left(S_{t_k}, \delta_k - \delta_{k-1}\right) - Z$$
$$\delta_{-1} = \delta_n = 0$$

One concrete illustration of this problem is to consider a vanilla call option of maturity T and payoff $Z = (S_T - K)_+$. In this case, there is only $d = 1$ hedging instrument: the underlying itself $S = S^{(1)}$. A strategy commonly used by the trader is to calculate at each time step t_k the Black-Scholes (BS) price of the option and the delta, that is the derivative of the BS price with respect to the underlying. This output constitutes the hedging strategy and is plugged into δ_k. This allows the trader to mitigate his directional risk and limit his possible losses, which is the goal of hedging.

B. Role of neural networks

As explained in the previous sub-section, the goal of hedging is to find the best strategy δ in order to minimise the P&L of a given portfolio. Practitioners use models like Black-Scholes to find δ. At each rebalancing time t_k, they compute for each hedging instrument its corresponding greek, which is the derivative of the price of the contingent claim Z with respect to the instrument, and plug the result into δ_k. This method allows to offset the risk factor(s) of the contingent claim associated to the hedging instrument(s).

Even if this approach is commonly used on trading desks, it has several limits:

- Transaction costs, liquidity constraints such as bid-ask spreads or more generally market frictions remain difficult to model correctly and are generally not considered.
- Greek computations do not take into account trading constraints (*e.g.* a limit on delta) and the adjustments that a trader has to make to comply with his desk limits
- In case of a complex model with an exotic contingent claim, the Greek computation can be either time-consuming or inaccurate because based on Monte-Carlo simulations

Those limits can be addressed by considering a model-independent approach based on deep learning. More precisely, the hedging strategy δ consists of one or several neural network(s). This approach does not need greeks to choose an appropriate strategy and takes into account market frictions as well as trading constraints.

In addition, this data-driven approach does not need in theory a model to generate a relevant hedging strategy.

In this case, not only can input data consist of market indicators (*e.g.* prices of underlying assets, vanilla derivatives, volatility indicators, *etc.*) but also more qualitative information such as news analytics.

In the remainder of this article, for the sake of simplicity, input data, and in particular market data information, is only represented by instruments S. Similarly, trading restrictions can be added to the deep hedging algorithm but are not considered here.

Each δ_k is the output of a neural network able to capture market information up to time t_k. A neural network can be viewed as a parametrised function able to approximate any sensible function if the vector of parameter θ is large enough. Two types of neural networks are considered in this case study:

- **Feedforward neural networks**: for each time step t_k, a neural network $\delta_k \coloneqq F(S_{t_k}; \theta_k)$ is built with the instruments present value as inputs. In the case of a call option, the input is the underlying present value. The networks are mutually independent and the hedging strategy is only a function of the present and not of the past. The architecture is presented in the figure below:

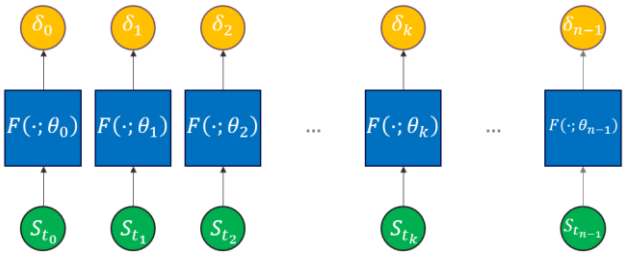

Figure 1: Overall architecture of the deep hedging strategy with n feedforward neural networks

- **Recurrent neural networks**: This type of network is in theory more adapted to the deep hedging problem. Indeed, it consists of only one network such that the k^{th} output δ_k depends on the first k inputs $S_{t_1}, S_{t_2}, \dots, S_{t_k}$. In this case, the hedging strategy takes into account **both present and past information**, which is more appropriate to consider for instance transaction costs or path-dependent instruments. The architecture is displayed in the figure below:

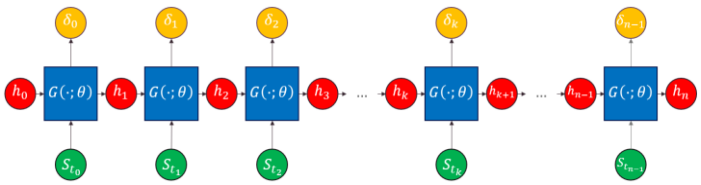

Figure 2: Overall architecture of the deep hedging strategy with one recurrent neural network

Where $G(\cdot;\theta)$ is a neural network (of parameter θ) with input the instruments S_{t_k} at a given time t_k plus

a previous state h summarising information up to t_{k-1}. In these two cases, the hedging strategy modelled as a neural network will be denoted by δ^θ.

C. Notion of risk measure and final formulation of the problem

As pointed out in the previous sub-sections, the goal of deep hedging is to minimise the losses of the P&L $(\delta^\theta \cdot S)_T - C_T(\delta^\theta) - Z$ by choosing the best hedging strategy δ^θ with neural networks. The P&L is a random variable and not only a function of the strategy δ^θ. It therefore cannot be optimised directly. This optimisation can only be achieved in the light of the scalar $\rho\left[Z + C_T(\delta^\theta) - (\delta^\theta \cdot S)_T\right]$ that focuses only on losses and that satisfies some mathematical properties.

A function satisfying these properties is called a convex risk measure. A possible risk measure is the Expected Shortfall ES_α of confidence level α and defined from the Value at Risk VaR_α of confidence level α according to the figure below:

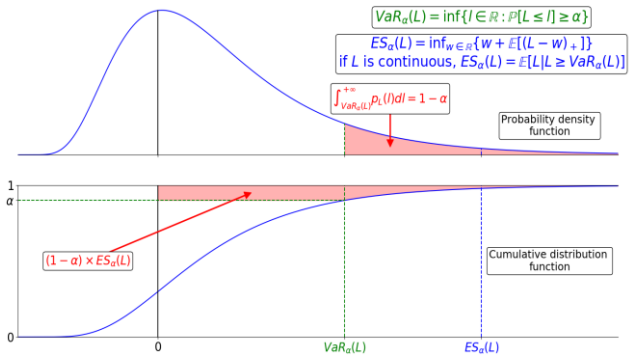

Figure 3: Graphical definition of VaR and ES with probability density and cumulative distribution functions of L

Where $L = Z + C_T(\delta) - (\delta \cdot S)_T$ refers to the reverse P&L and losses are counted positively. Even if trading desks rather privilege VaR over ES as a reference risk metric, ES is favoured in this context because VaR does not satisfy all the mathematical criteria of a convex risk measure and thus cannot be used to optimise the neural networks. From the ES definition, one can build an interesting convex risk measure called mixed expected shortfall and defined by:

$$\rho_\beta[L] := \frac{1}{1+\beta}\left(ES_{50\%}[L] + \beta \times ES_{99\%}[L]\right)$$

This measure enables to account for two different types of losses : $ES_{50\%}[L]$ represents overall losses while $ES_{99\%}[L]$ represents extreme losses. β is a hyper-parameter chosen arbitrarily. The higher the β parameter, the more the focus is on extreme losses as $\lim_{\beta \to +\infty} \rho_\beta[L] := ES_{99\%}[L]$. In this case study, tests showed that choosing $\beta = 1$ allows to consider extreme losses.

Keeping the same notations as before, the final problem of P&L minimisation with neural networks can be formulated as the following stochastic optimisation problem:

$$\text{find } \theta^* := \underset{\theta}{\operatorname{argmin}}\, \rho_\beta\left[Z + C_T(\delta^\theta) - (\delta^\theta \cdot S)_T\right]$$
$$= \underset{\theta}{\operatorname{argmin}} \min_{w_1, w_2} \frac{1}{1+\beta}\begin{pmatrix} w_1 + \frac{1}{1-50\%}\mathbb{E}\left[\left(Z + C_T(\delta^\theta) - (\delta^\theta \cdot S)_T - w_1\right)_+\right] \\ +\beta \times w_2 + \frac{1}{1-99\%}\mathbb{E}\left[\left(Z + C_T(\delta^\theta) - (\delta^\theta \cdot S)_T - w_2\right)_+\right] \end{pmatrix}$$

This optimisation problem can be solved by usual stochastic optimisation methods such as stochastic gradient descent or Adam.

3. Results under Black-Scholes Model

A. Model assumptions

In this case study, the deep hedging approach is first tested under the Black-Scholes model. This model is still an industry widespread model used by traders to hedge vanilla options. In this section, the goal is to hedge a short call option position (*i.e.* $Z = (S_T - K)_+$) of maturity 30 days with daily rebalancing where S_T was simulated under the Black-Scholes framework:

$$S_t = S_0 e^{\sigma W_t - \frac{\sigma^2}{2}t}$$

Where S_0 is the initial underlying price, W_t is a Brownian motion and σ is the volatility parameter. To hedge against the risk of loss, only $d = 1$ instrument is considered in the market: the underlying price S_t. In the results detailed below, no transaction cost is considered.

In theory, it is possible to perfectly replicate a short call option position with only the underlying and without considering transaction costs. This can be done by buying the quantity delta of the underlying, where delta is the derivative of the option price with respect to S_t.

B. Numerical results

In this framework, only overall losses are considered (*i.e.* $\beta = 0$) in the deep hedging algorithm. As the P&L histogram and its associated numerical results show in figure 4 and table 1, the deep hedging strategy with feedforward neural networks is close to the Black-Scholes strategy and produces similar results in terms of P&L distribution and risk metrics (VaR and ES).

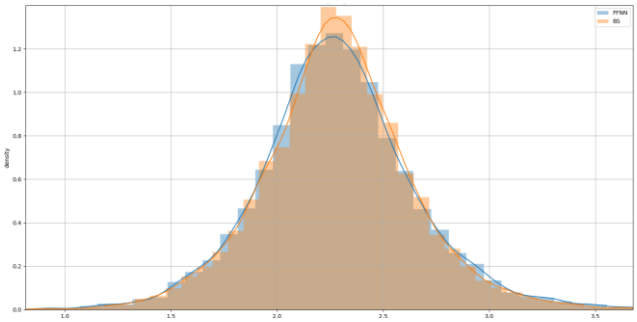

Figure 4: Reverse P&L histogram of the BS strategy in orange and the deep hedging strategy with feedforward neural networks in blue

Hedging strategy	50% VaR	50% ES	99% VaR	99% ES
Black-Scholes	2.28	2.56	3.25	3.50
deep hedging	2.27	2.57	3.34	3.65

Table 1 : Numerical results associated to figure 4

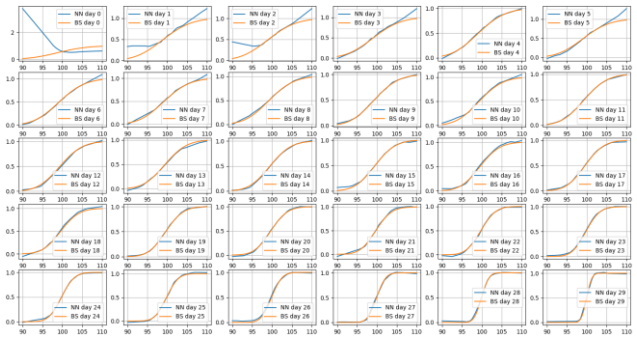

Figure 5: Comparison for each rebalancing day between the BS strategy in orange and the deep hedging strategy with feedforward neural networks in blue

The BS and deep hedging strategies δ_k as a function of the underlying price S_{t_k} for each day t_k are displayed in figure 5. The two strategies are very similar since the two curves match each other for each rebalancing day. The algorithm provides relevant results with this model.

4. Results under Heston Model

A. Model assumptions

The deep hedging approach was also tested in a more complex framework. The goal is still to hedge a short call option position (*i.e.* $Z = \left(S_T^{(1)} - K\right)_+$) of maturity 30 days with daily rebalancing, but here, $S_T^{(1)}$ is simulated under the Heston framework:

$$dV_t = a(b - V_t)dt + \sigma\sqrt{V_t}dW_t^{(1)}$$
$$dS_t^{(1)} = \sqrt{V_t}S_t^{(1)}\left(\rho dW_t^{(1)} + \sqrt{1-\rho^2}dW_t^{(2)}\right)$$

Where $W_t^{(1)}$ and $W_t^{(2)}$ are two independent Brownian motions, and V_t and $S_t^{(1)}$ denote respectively the squared stochastic volatility and the underlying price. ρ is the correlation parameter between underlying and volatility. a is the mean reversion speed of V_t, b is the long term mean reversion level, and σ is the noise parameter.

To hedge against the risk of loss, $d = 2$ instruments are available:

- The underlying asset of price $S_t^{(1)}$
- A variance swap of maturity $T = 30$ days and price $S_t^{(2)} = \int_0^t V_u du + \frac{1-e^{-a(T-t)}}{a}(V_t - b) + b(T-t)$

In theory, it is possible to perfectly replicate a short call option position with only those two instruments and without considering transaction costs. Furthermore, unlike the BS case, it is assumed that transaction costs are proportional with $c_k^{(j)} = 0{,}01$.

B. Numerical results

To assess the efficiency of the deep hedging algorithm in the Heston framework, the reverse P&Ls (losses are counted positively) histograms of 4 strategies are compared:

- No hedge (red): no hedge has been performed *i.e.* $\delta_k = 0$ at each time step
- Heston (green): δ_k is calculated according to the Heston model hedging strategy
- FFNN (orange): the δ_k are the outputs of independent feedforward neural networks as explained in section 3
- RNN (blue): δ is the output of one recurrent neural network as explained in section 3

The choice of $\beta = 0$ (only overall losses are taken into account) produces the following results:

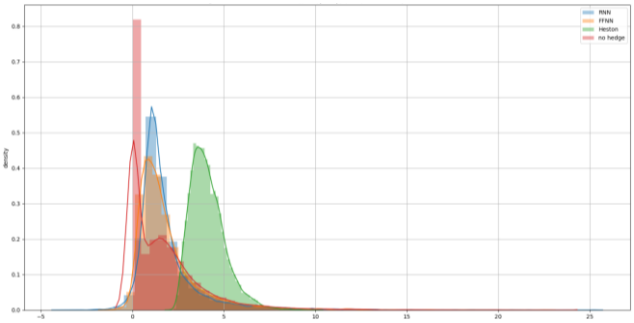

Figure 6: Reverse P&L histograms of the Heston strategy in green, the deep hedging strategies with feedforward neural networks in orange and recurrent neural network in blue. The red histogram represents the no hedge case. In this case, $\beta = 0$.

The choice of $\beta = 1$, *i.e.* both overall and extreme losses are taken into account, gives the reverse P&L histogram in figure 7.

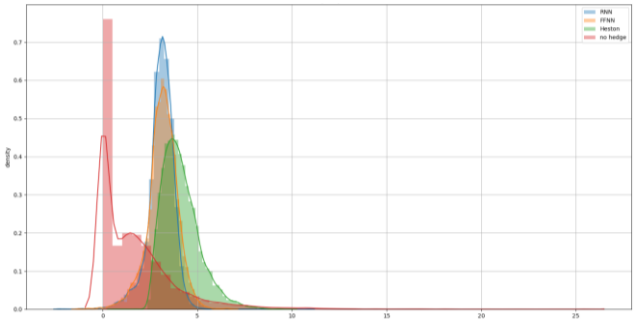

Figure 7: Reverse P&L histograms of the Heston strategy in green, the deep hedging strategies with feedforward neural networks in orange and recurrent neural network in blue. The red histogram represents the no hedge case. In this case, $\beta = 1$.

Hedging strategy	50% VaR	50% ES	99% VaR	99% ES
Heston	3.97	4.88	7.09	7.79
RNN ($\beta = 0$)	1.35	2.95	10.21	13.38
FFNN ($\beta = 0$)	1.38	3.04	9.72	12.25
RNN ($\beta = 1$)	3.26	3.67	4.53	4.93
FFNN ($\beta = 1$)	3.16	3.77	5.11	5.57
No Hedge	1.16	3.16	10.15	12.91

Table 2 : Numerical results associated to figures 6 and 7

The numerical results of the histograms can be found in Table 2.

These results show the crucial influence of β. They point out that the RNN framework performs better than the FFNN in this specific case. When $\beta = 0$, even if overall losses are minimised by the two algorithms (the RNN being better than the FFNN), almost no hedge is performed by the neural networks, and the hedging performance is so poor that extreme losses are of the same order of magnitude as the no hedge case. On the contrary, if $\beta = 1$, the overall losses are again slightly larger but extreme losses are much smaller. Moreover, the two deep hedging strategies provide better results than the Heston strategy in all situations. Finally, and as expected, the RNN strategy is the best hedging strategy both in terms of overall losses and extreme losses and manages satisfactorily transaction costs.

5. Main limits of the deep hedging algorithm

Even if the previous numerical results show that the deep hedging algorithm has a huge potential, it has non-negligible drawbacks. To illustrate that, a numerical example based on real data is given in this section.

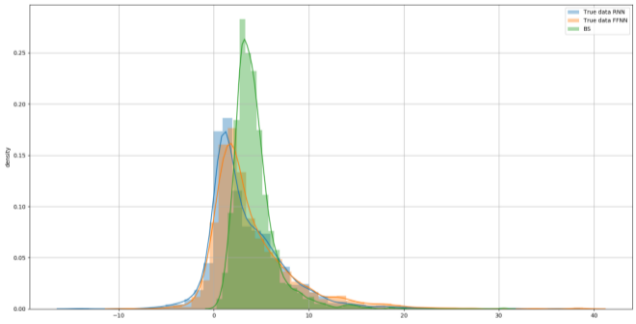

Figure 8 : Reverse P&L histograms of the BS strategy in green, the deep hedging strategies with feedforward neural networks in orange and recurrent neural network in blue. In this case, $\beta = 1$ and deep hedging strategies are trained with CAC40 stocks sample trajectories

In this example, the dataset (for both training and testing) is composed of all the constituents of the CAC 40 index ranging from 1990 to 2020. Each sample path used is a normalised 31-days stock price process of one given stock such that the sample trajectories do not overlap. Similarly to the previous numerical examples, the goal is to hedge against a short call option position of payoff $Z = (S_T - K)_+$.

The deep hedging algorithm (with FFNNs and RNNs) was performed with only $d = 1$ hedging instrument, namely the stock itself. To limit losses as much as possible, $\beta := 1$ was chosen, and no transaction cost was considered. To have a relevant benchmark, a BS strategy was performed as well. BS volatility σ was estimated for each stock of the training dataset. Hence, the benchmark knows which stock is used for hedging, which is not the case for the deep hedging algorithm.

As the results in figure 8 point out, the BS hedging method is significantly more efficient than the deep hedging ones both in terms of overall and extreme losses, even though it has the advantage to be specific to each stock.

This highlights two points:

- Contrary to the model approaches detailed in previous sections, the back-test on real data has much less sample trajectories. The lack of data has then a negative impact on the hedging strategy efficiency
- The algorithm seems to lack robustness if the input information is not diverse enough.

The second issue can be fixed by adding more information. The first issue remains more complicated to solve and sets a paradigm problem : the only way to have a sufficient amount of data is to generate it with a model, which makes the deep hedging approach no longer model-independent.

Advantages	Limits
• **Greek independence:** No need to compute Greeks especially for a complex model and a portfolio of exotic and illiquid derivatives • **Market constraints modelling**: Ability to model market frictions such as transaction costs and trading constraints • **Choice of risk aversion:** Possibility to privilege overall losses over extreme losses and vice-versa • **Diversity of input information**: Ability to consider almost as many risk factors as possible and in case of complete model independence, any possible quantitative or qualitative piece of information • **Model independent in theory**: No need of pricing models	• **Lack of data**: Need a huge amount of data to be trained correctly • **Lack of robustness**: If not enough information is included, the deep hedging algorithm fails to provide a relevant hedging strategy • **Neural Network recalibration**: Necessity of model recalibration in the case of a market shift regime • **Compliance with regulatory constraints and Validation**: Difficulty to explain the outputs of the deep hedging framework especially for complex derivatives

Table 3: Main advantages and limits of deep hedging

6. Conclusion

In this article, we presented a data-driven and model-independent method with huge potential to hedge financial derivatives. This deep hedging approach is able to provide a relevant hedging strategy with three main inputs: the derivative to hedge, the hedging instruments, and the aversion against potential extreme losses, *i.e.* the choice of β value. In addition, this algorithm is flexible since it can include any relevant input information such as transaction costs, trading constraints, market data information or even qualitative information. The capacity of the algorithm to include these new elements is difficult to obtain from a model approach and allows to refine the hedging strategy.

The results obtained in the case of data simulated from existing models such as Black-Scholes or Heston are promising. Indeed, the hedging performance of the algorithm is either as good as the model-based strategy if there is no transaction costs, or better than these strategies in the presence of those costs. In the two models, the deep learning approach has not used greeks to obtain such results.

Nevertheless, a simple back-test on real data shows two important drawbacks of deep hedging: it needs a huge amount of data to be trained and tested correctly and is not robust when it does not have enough information. The different advantages and limits are summarised in table 3.

Although the second drawback can be easily fixed by adding more input information to the algorithm, the first one is more complicated to solve. A possible way to get around that issue is to reproduce similar artificial data as the existing market data in terms of distribution. To do so, several methods are possible: GAN, ARCH, and ARMA-GARCH or quantile regression. Still in the perspective of being completely model independent, quant practitioners such as Hans Buehler (see reference (3)) explained that pure data-driven approaches already exist in several banks for liquid derivatives books thanks to the data availability but not yet for illiquid products because of a lack of data.

The deep hedging approach for illiquid derivatives seems still relevant for hedging without considering greeks. Such products need a more complex model like a local or stochastic volatility model or the combination of the two. In this case, greeks computations are achieved by Monte-Carlo simulation and can be time consuming. Deep hedging can return in a similar amount of time a more relevant hedging strategy considering both transaction costs and trading constraints.

References

1. Hans Buehler et al. “Deep Hedging”. arXiv: Computational Finance. 2018.
2. Michal Kozrya. “Deep Learning approach to Hedging”. University of Oxford. 2018. MA Thesis
3. Nazneen Sherif and Mauro Cesa. “Hans Buehler on deep hedging and the advantages of data-driven approaches”. Risk.net. 2019. URL:https://www.risk.net/derivatives/6705012/podcast-hans-buehler-on-deep-hedging-and-the-advantages-of-data-driven-approaches

Contact

Christophe Bonnefoy
Partner
Head of Quantitative Finance
Christophe.bonnefoy@mazars.fr

Contributors: Thomas Leygonie, David Dagane

Mazars is an internationally integrated partnership, specialising in audit, accountancy, advisory, tax and legal services*. Operating in over 90 countries and territories around the world, we draw on the expertise of more than 42,000 professionals – 26,000+ in Mazars' integrated partnership and 16,000+ via the Mazars North America Alliance – to assist clients of all sizes at every stage in their development.

*where permitted under applicable country laws.

www.mazars.com

mazars