Option pricing and model calibration with neural networks

Halász Kristóf, ELTE

Thesis advisor: Dr. Fáth Gábor

2025.06.17.

Introduction: derivative pricing

- Derivatives
  - Option, barrier option
- Price: infimum of the prices of hedge strategies
  - Dynamic delta hedging
- In the options market prices are quoted in implied volatility
  - Convention to which implied volatility

> *"Implied volatility is the wrong number to put into the wrong formula to get the right price.„*
>
> *Riccardo Rebonato*

Introduction: derivative pricing

- We are also interested in the sensitivites or risks
  - Partial derivatives of input parameters (underliers)
  - „option greeks”

- Black-Scholes-Merton model, 1973
  - First consistent option pricing model
  - Closed formulas for vanilla options
  - Constant volatility

Introduction: derivative pricing

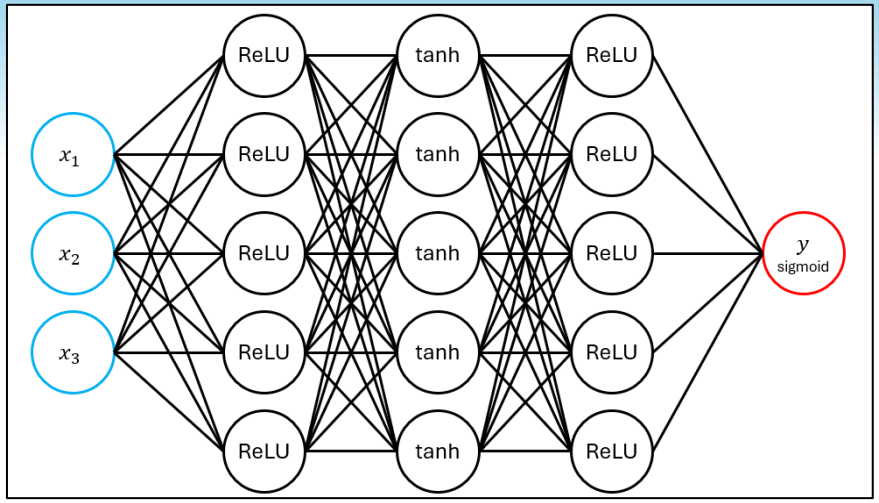
- Problems with the BS model
  - Volatilities calculated from prices observed in the market do not reflect the constant volatility assumption
  - Implied volatility is dependent on expiry and strike
  - Skews and smiles of the implied volatility surface

- More complex dynamics are able to capture the properties of observed implied volatility surface
  - Local volatility models
  - Stochastic volatility models
  - Jump models

Introduction: derivative pricing

- Basic payoffs in most of the cases have fast (semi-)analytical solution for the price
- Exotic payoffs often require numerical algorithms
  - Analytical formula is not known
  - PDE solvers, tree based methods, Monte-Carlo simulation
- Solving for the BS implied volatility is also an iterative numerical algorithm
- Stochastic yield curve modelling, including jump processes, stochastic time-change, …?

Introduction: neural networks



- Class of machine learning algorithms
  - Learn complex relationships or patterns in the data
  - Prediction
- Feedforward neural networks
  - Composition affine transformations and non-linear activation functions
- Convolutional DNN, Resnet, LSTM, …
- Black box
- In most real world applications, DNNs are used to approximate an unknown function in a data driven fashion.

Neural networks for derivative pricing

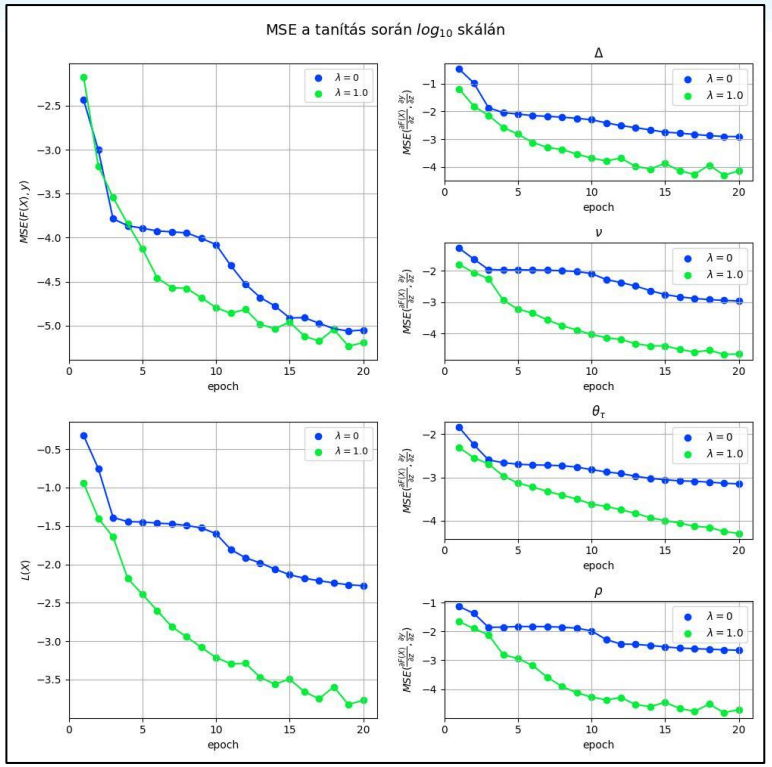
- DNNs can be used to approximate known functions
- Why?
  - Calculation leads to inefficient algorithm
  - Partial derivatives require numerical approximation
- Evaluation of DNNs is efficient
  - O(size of computational graph)
  - Backpropagation algorithm provides the partial derivates in the same complexity
  - All of them at the same time
- In derivative pricing, these properties can lead to significant computational speedup
  - Approximation ↯ arbitrage free

Neural networks for derivative pricing

- Cons:
  - DNNs extrapolate poorly
  - Overfitting: accuracy is very high for the prices, but poor for the greeks
- Data?
  - Market observations
  - Synthetic data generation
- Training on synthetic datasets are pricing model dependent

Results: BS model

- Implementation in Python
  - Perfomance limitations
  - PyTorch, Tensorflow
- Greek regularization
  - Derivative informed learning
  - MSE of the neural network to the exact greeks are added to the price-MSE in the loss function
  - Less prone to overfitting
  - Accelerate training process
  - Allow simpler structures
- Error in shape of the function vs error of the predictions
- Bias in the predictions, λ parameter to control this tradeoff



Results: BS model

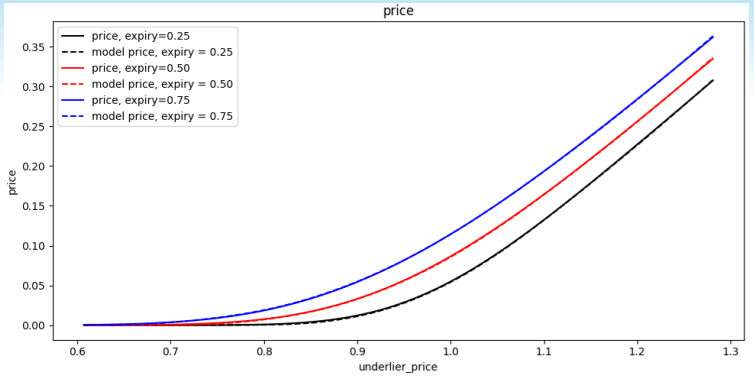
- Black-Scholes-Merton model
  - European call, put, and barrier options
  - 4 input variables, as the model is homogeneous in the strike

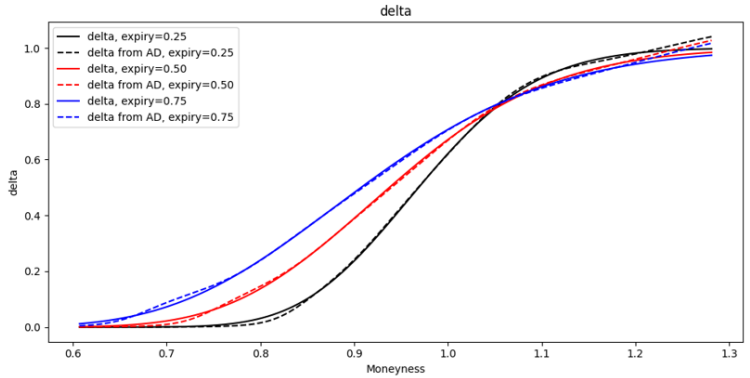
  - Vanilla options: data generation is fast due to analytical formulas to both price and greeks
  - Barrier options: Crank-Nicolson scheme for the BS PDE, greeks are numercially approximated

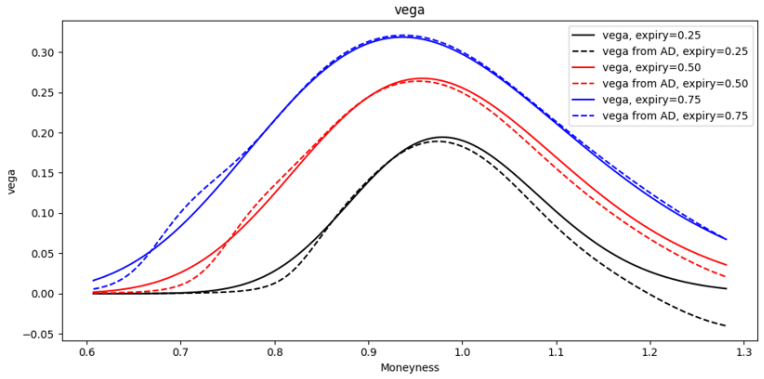
- $\sim 10^{-7}$ MSE in the prices, $\sim 10^{-6}$ MSE for the greeks

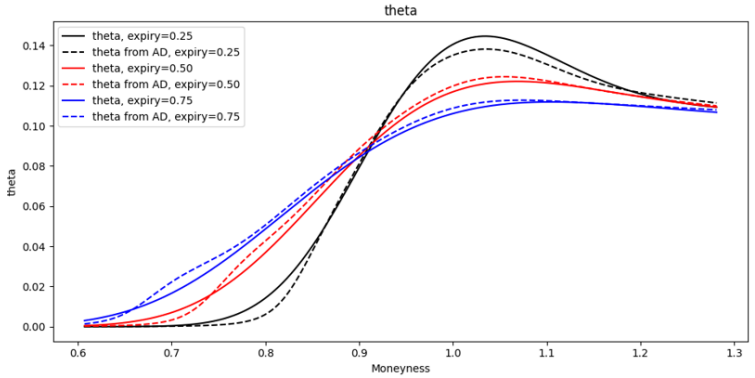
- Good visual alignment across the input parameter space

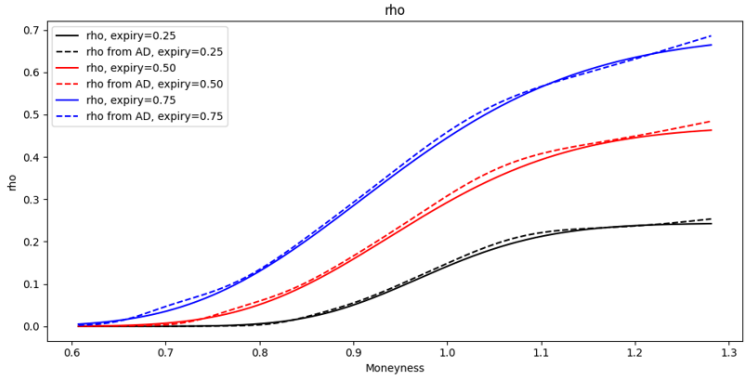
Results: BS model







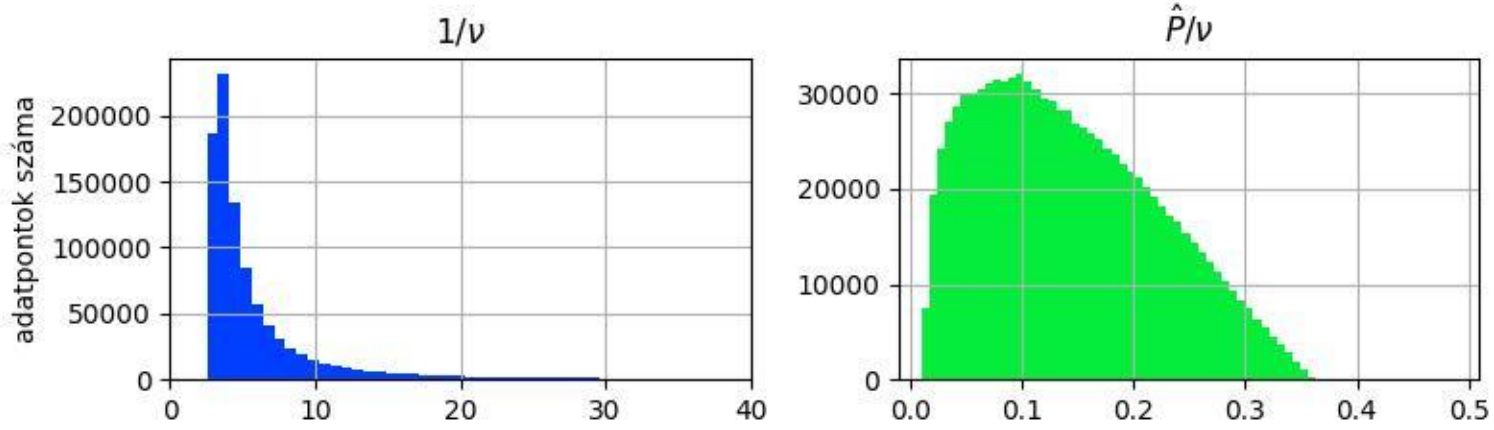




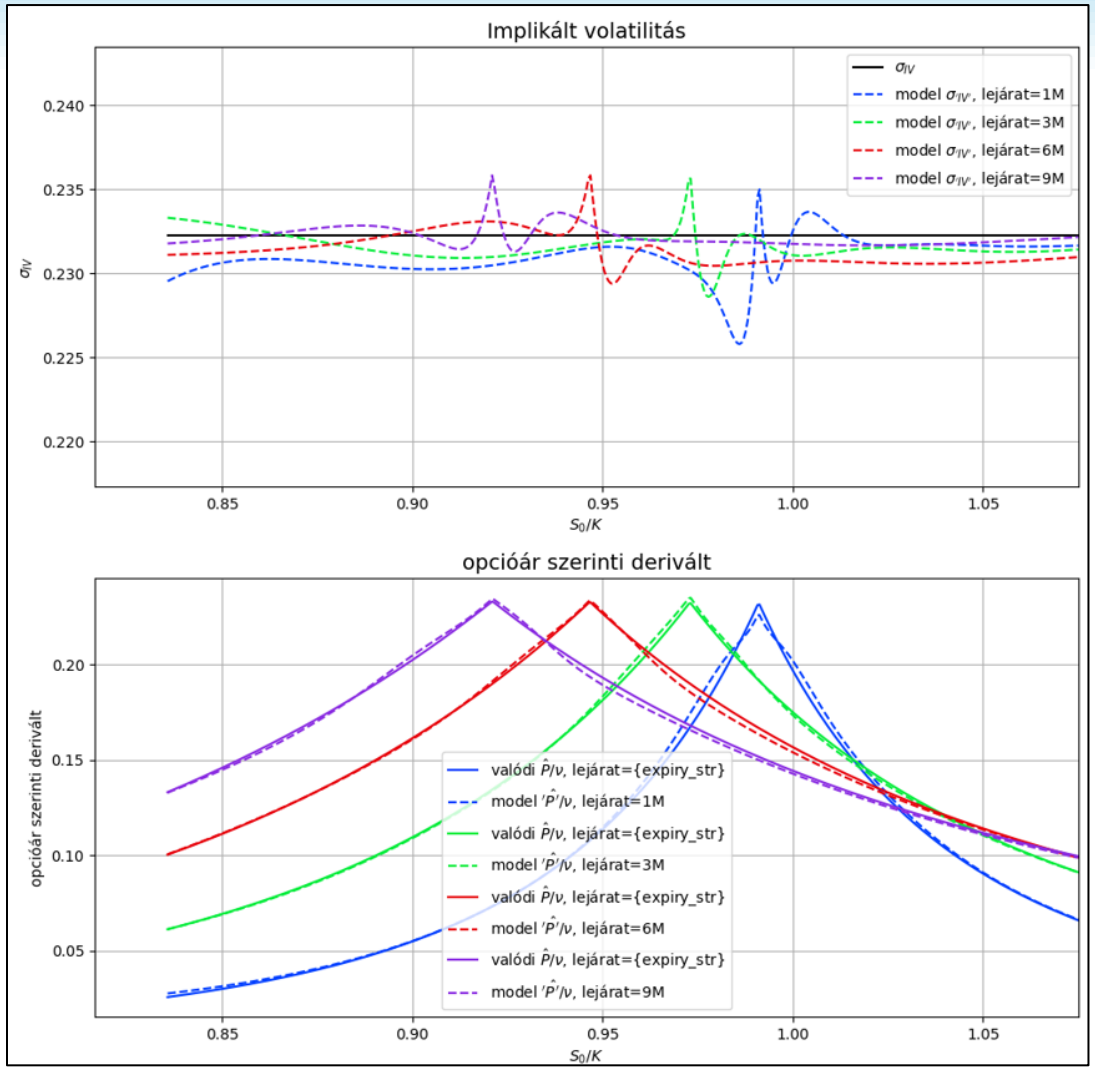
		call		put	
		64 | 8 | tanh | $\lambda = 0.5$		128 | 8 | tanh | $\lambda = 1.0$	
		tanító adathalmaz	teszt adathalmaz	tanító adathalmaz	teszt adathalmaz
ár	MSE	1.6674e-07	1.2237e-07	8.9815e-07	6.4966e-07
Δ	MSE	1.1981e-05	1.0109e-05	9.1642e-06	5.8899e-06
θ_τ	MSE	4.8399e-06	8.6058e-07	9.6393e-06	1.8052e-06
ν	MSE	4.6260e-06	2.7756e-06	6.1357e-06	4.1319e-06
ρ	MSE	6.4467e-06	2.6898e-06	7.3239e-06	3.1663e-06

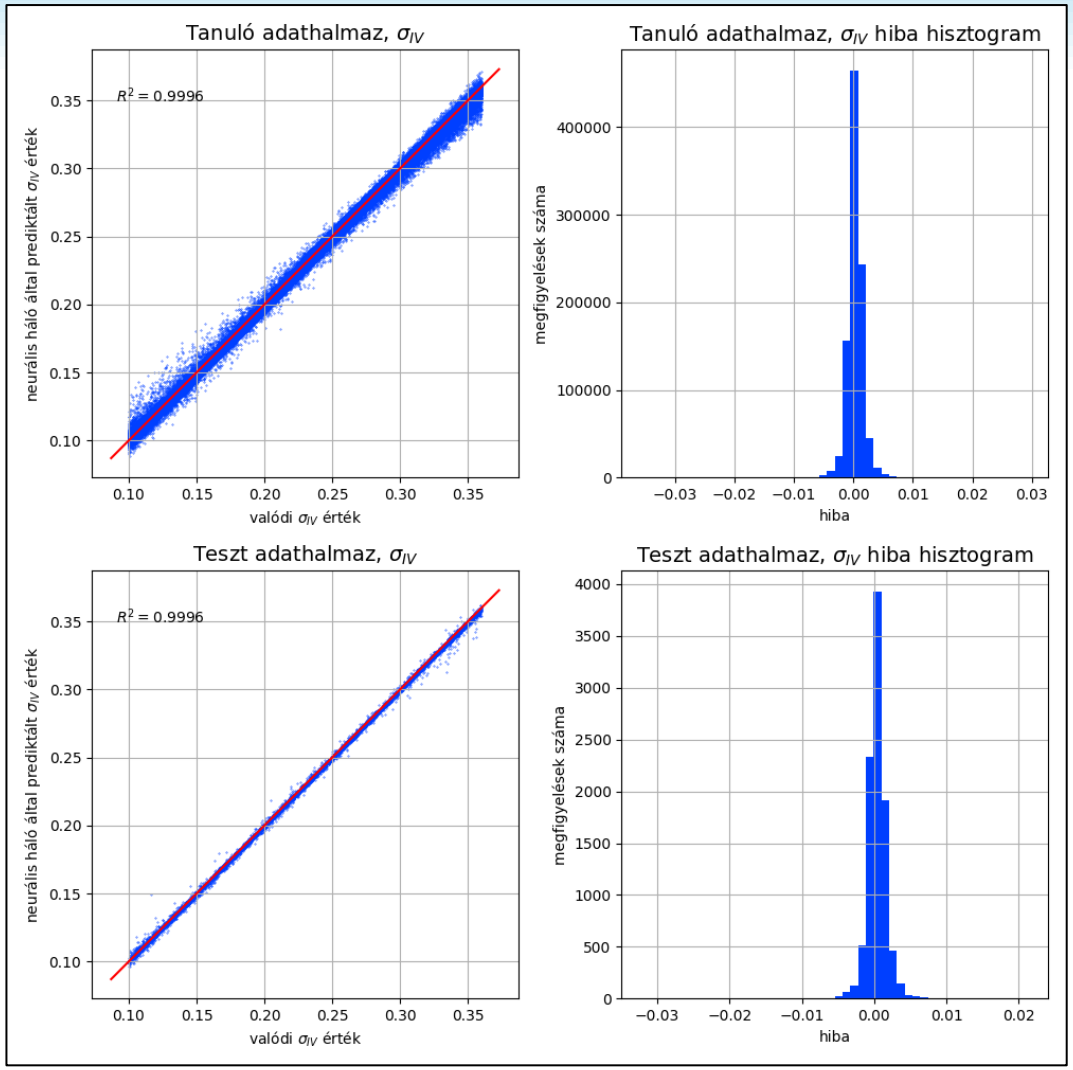
Results: BS model implied volatility

- Partial derivative wrt. the option price is $\frac{1}{\nu}$.
- Exploding gradient towards ITM or OTM
  - Instable learning
- Transforming moneyness to logarithm of the time values leads to much better performance



Results: BS model implied volatility



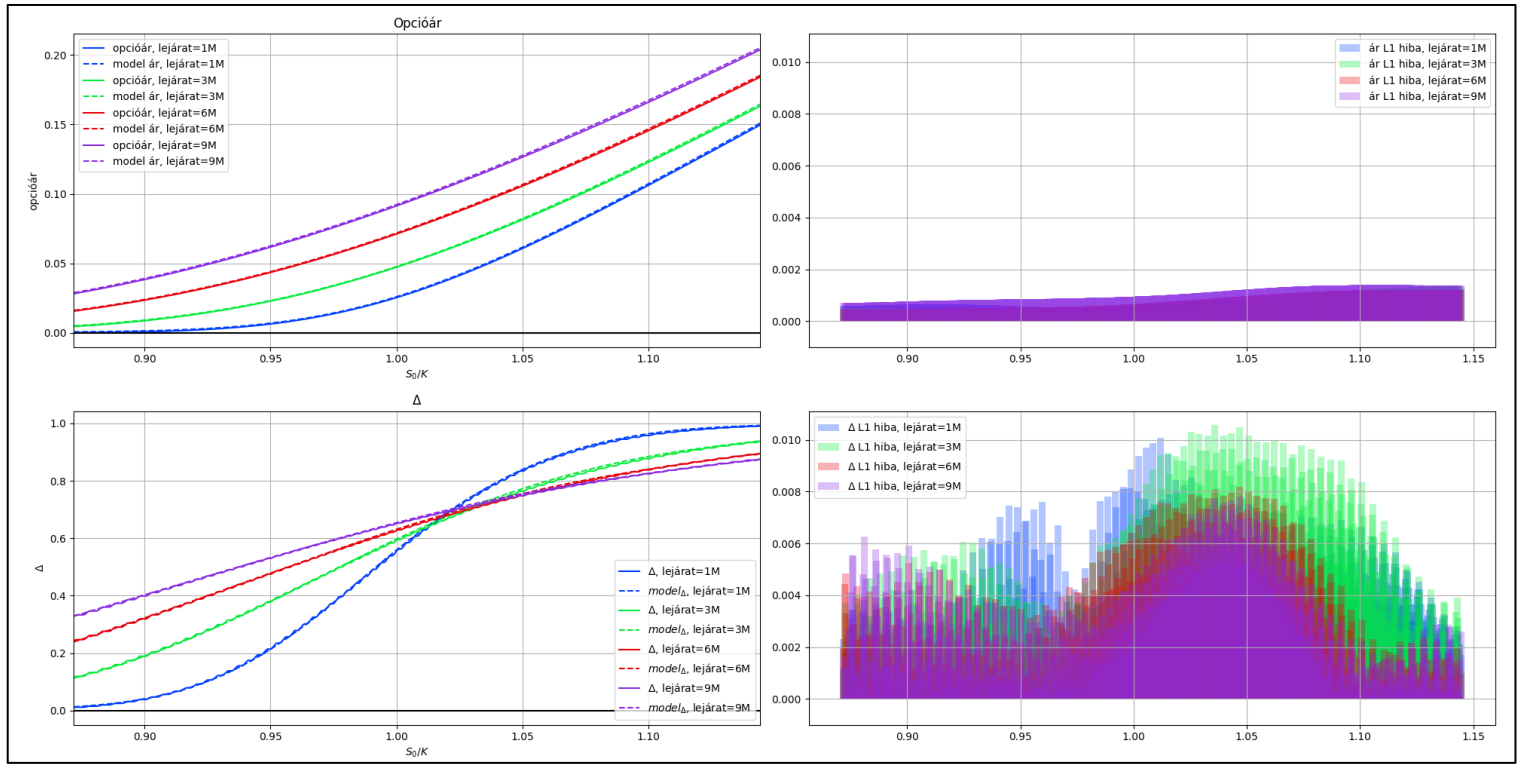


Results: Heston model

- Heston model
  - European call, put and barrier options
  - 8 input variables, as the model is homogeneous in the strike
  - Semi-analytical formula is available for vanilla options

  - For barrier options, solving the Heston PDE in three dimensions
  - Craig-Sneyd scheme to account for the cross partial derivative term
- Data generation is much slower
- Higher errors compared to the Black-Scholes model

Results: Heston model



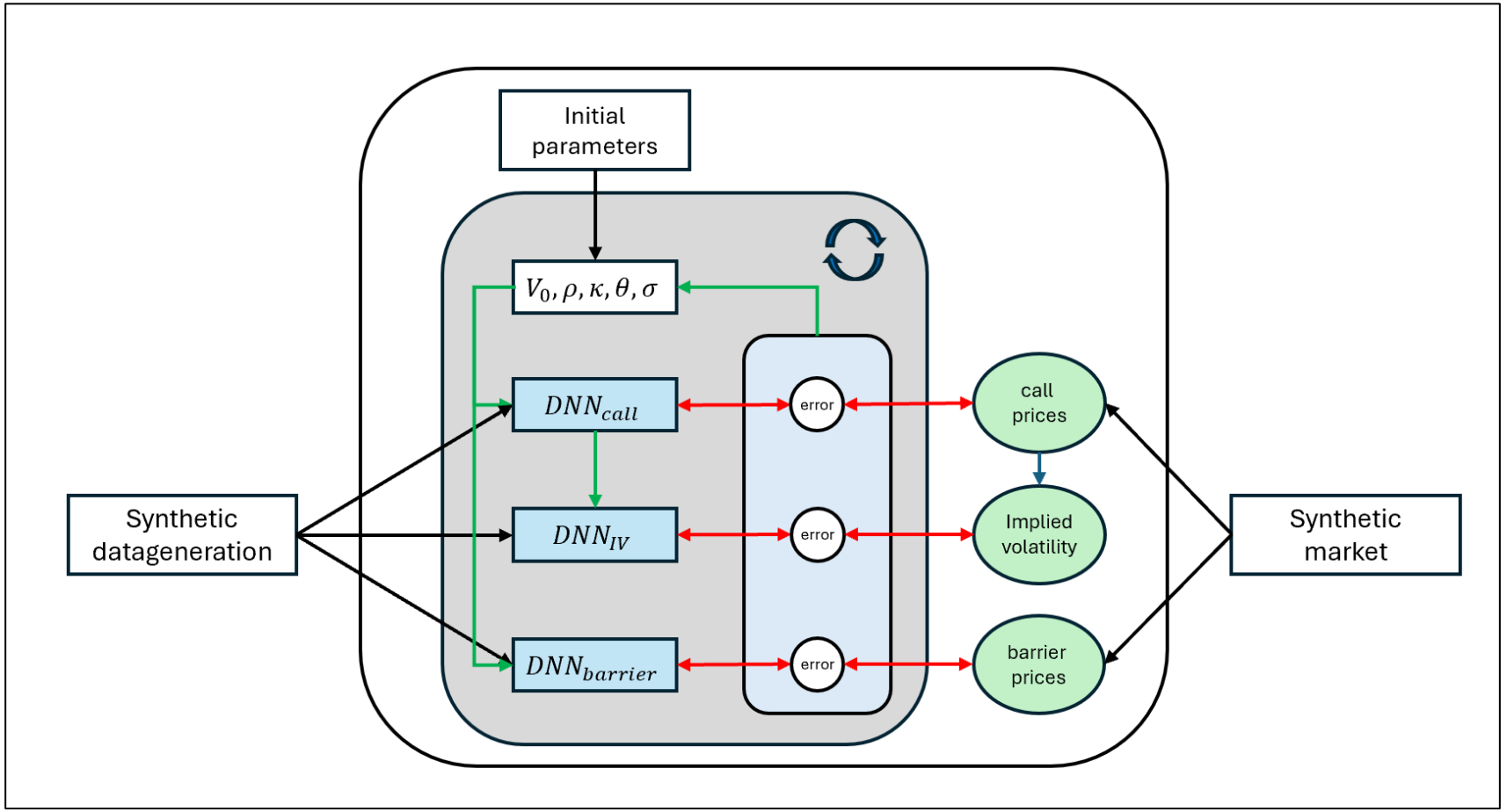
		Heston call		Heston up-and-out call	
		64 | 8 | tanh | $\lambda = 1.0$		**128 | 2 | GeLU | $\lambda = 0.1$**	
		tanító adathalmaz	**teszt adathalmaz**	**tanító adathalmaz**	**teszt adathalmaz**
ár	MSE	5.4437e-05	4.8597e-05	1.1150e-03	1.2622e-03
Δ	MSE	7.4685e-04	7.8749e-04	3.4645e-02	2.6031e-02
θ_τ	MSE	1.5911e-04	1.5379e-04	1.8282e-03	1.6843e-03
ν	MSE	1.5990e-03	1.4738e-03	4.4343e-02	4.4459e-02
ρ	MSE	2.3128e-04	2.3322e-04	6.0044e-03	6.1415e-03
	$\frac{\partial f}{\partial \mathrm{corr}}$	1.2530e-06	1.1452e-06	9.5922e-05	1.1442e-04
	$\frac{\partial f}{\partial \kappa}$	3.9682e-07	4.0224e-07	6.7190e-06	9.5121e-06
	$\frac{\partial f}{\partial \mathrm{theta}}$	1.7193e-03	1.5500e-03	3.5093e-02	4.0878e-02
	$\frac{\partial f}{\partial \sigma}$	4.5582e-06	4.2644e-06	2.4680e-04	3.0416e-04

Results: Calibration

- Calibration
  - Optimizing the model parameters on the distance between model and observed prices
  - Non-convex optimization
  - Fast if we can calculate the prices efficiently

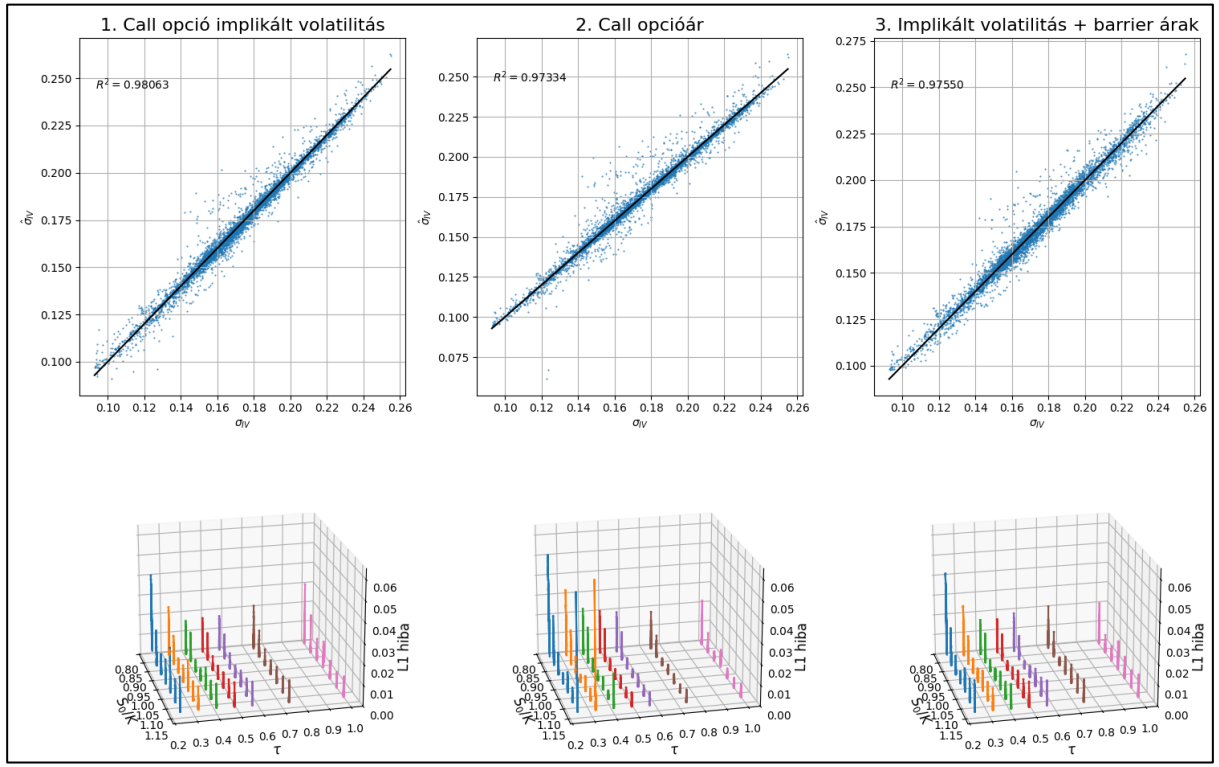
- Inclusion of exotic payoff leads to better capture of forward skew and smile
  - Increase in the calibration time

Results: Calibration

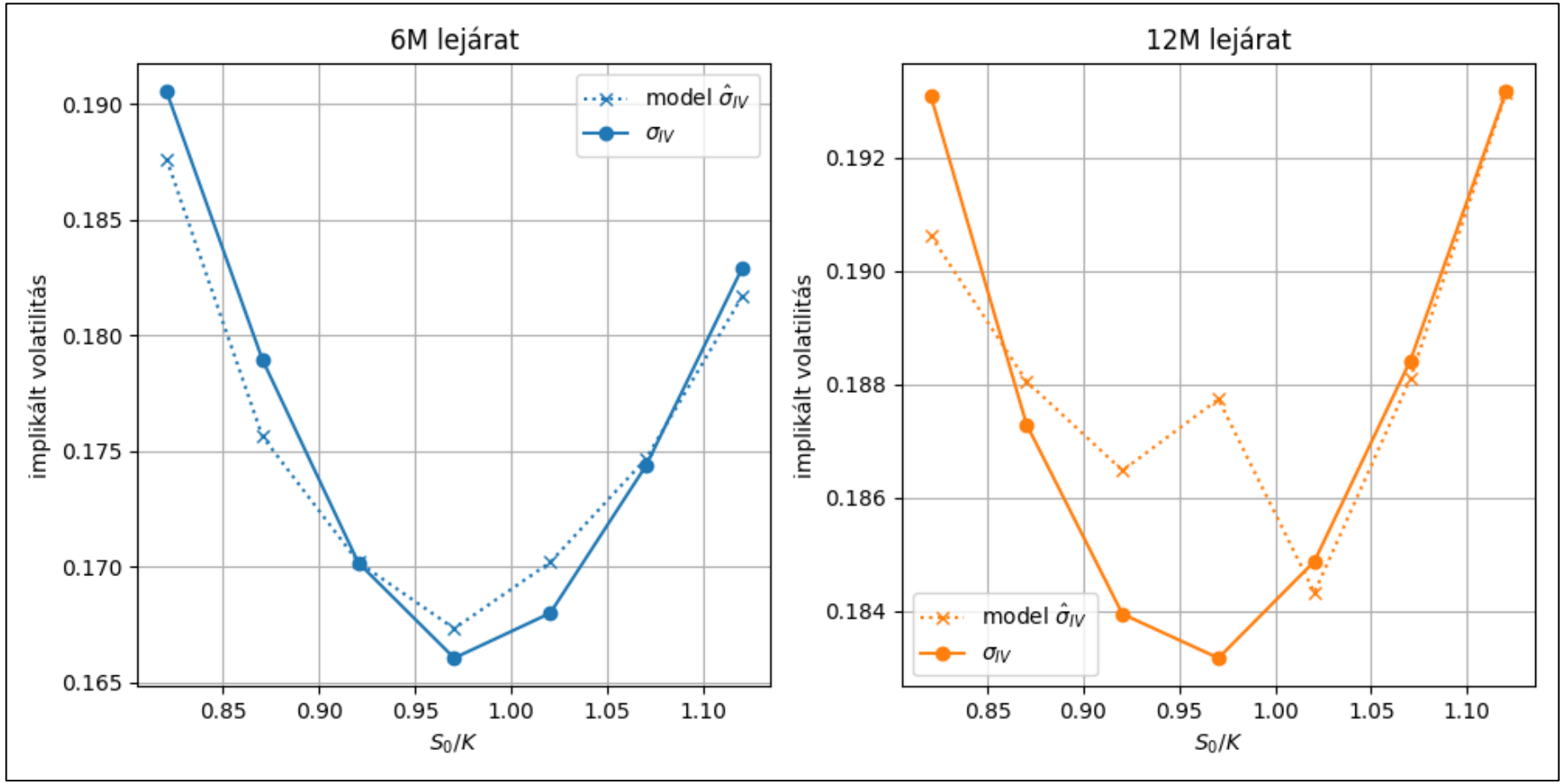


Results: Calibration

- Calibrating to synthetic market data
  - Acceptable visual fit
  - Large error in parameters, due to the DNNs error in the prices



Results: Calibration



Possible improvements

- Implementation in a low-level programming language (C++)
- Computational graph for the PDE solver algorithm to reduce the data generation runtime
  - Memory issues, as sparse matricies are not supported
  - Interpolation on the grid is not possible (easily)
- Fourier network operators
  - DNNs are learning the PDE itself

References

- https://www.math.elte.hu/thesisupload/thesisfiles/2025msc_actfinmat2y-cj49ch.pdf

- https://github.com/HKristof136/msc_thesis_2025