ディープニューラルネットワークとGPUを用いた無裁定SABR

株式会社アイゲンヴュー

熊谷雅年

INDEX

当社について

- 当社

  株式会社アイゲンヴュー

  2007年設立

  - 代表取締役

    熊谷雅年

    略歴 – 金融系ソフトウェア会社数社, 富士銀行, 慶応義塾大学理工学部卒

  - ビジネス領域

    市場/信用リスク管理、ALM、デリバティブ周りに特化したシステム開発

要約

- ディープニューラルネットワーク(DNN)とGPUを用いた次世代数値計算手法
  - ニューラルネットワークの普遍近似定理（万能近似定理）に基づき正確な数値を算出
- プロトタイプとして、無裁定なSABRモデルによるインプライドボラティリティの補間補外機能を試作
  - 通貨オプションが対象取引 (マイナス金利の考慮が不要であれば、市場慣行がBlack型のボラにも対応可)
- DNN用の訓練データはGPUによるモンテカルロ・シミュレーションで生成
  - 市場で観測されるボラティリティの疑似データとして使用
  - 効率的なコードを書くことよりGPUは高速なモンテカルロ・シミュレーションを実行可能
- DNNによる訓練により、SABRモデルのキャリブレーションと無裁定となるSABRの関数近似が可能
  - 数値計算の結果はこのドキュメントの後半に提示
- この仕組みを用いれば、ユーザは市場データから容易にキャリブレーションし、SABRモデルによる無裁定なインプライドボラティリティを算出することが可能

どの様にスマイルを描くか（1/15）

- 概要
  - ニューラルネットワークの普遍近似定理に基づきSABRの関数近似を作成
  - DNNによる手法の簡潔な説明
    1. 訓練データ生成
       - *GPU* によるモンテカルロシミュレーション[1]で*SABR*モデルに従うボラティリティデータを大量生成
       - 多量のSABRの変数、パラメータの組み合わせにより訓練データを構成
       - 各変数、パラメータの範囲は市場で観測されるボラティリティをカバーするように作成
    2. 学習

       DNN は特徴量データと教師データ間[2]の関係を学習し、回帰機能によりSABRモデルの関数近似式を生成

       SABRモデルの関数近似の場合、
       - 特徴量：$\alpha_0, \beta, \rho, \nu$, *strike, tenor, forward rate*
       - 教師データ：σ_{black}

    3. 推計
       - 訓練に用いたものとは別のデータでボラティリティの推計を行い教師データと一致するかを確認

    充分な検証の上、<u>問題が無ければ実務にて使用可能でしょう！</u>

SABR model

$$dF_t = \alpha_t F_t^{\beta} dW_1$$
$$d\alpha_t = \nu \alpha_t dW_2$$
$$\rho = dW_1 dW_2$$

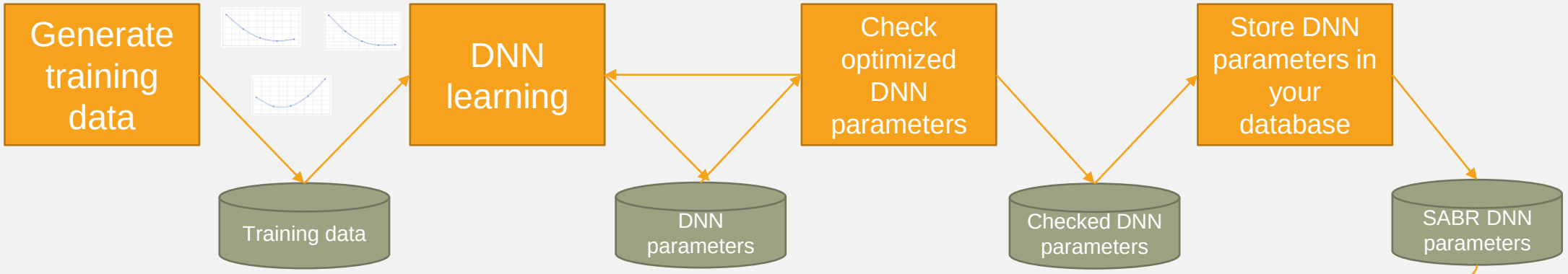
<u>*Approximation formula f()*</u>

f($\alpha_0, \beta, \rho, \nu$, *forward rate, strike, tenor*)
= σ_{black}

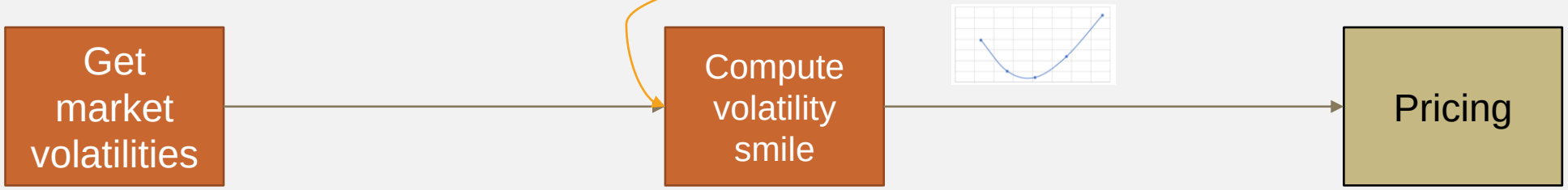
1 もっと早く正確な手段があればモンテカルロにこだわる必要はない。GPUを用いた有限差分法は有効か？
2 このドキュメントでは、訓練データは特徴量データと学習データの二種類とする

どの様にスマイルを描くか(2/15)

- 事前準備



- 実務



どの様にスマイルを描くか(3/15)

- 通貨オプションのインプライド・ボラティリティ補間 / 補外

1. キャリブレーション

   2つの手法

   I. SABR 関数近似 (DNN近似)

   市場のボラティリティ[3]に対し、DNNにより生成されたSABR関数近似式とLevenberg−Marquardtの様な準ニュートン法を組み合わせSABRのパラメータを推計

   II. ダイレクトキャリブレーション (DNNキャリブレーション)

   DNNにより５個の市場ボラティリティ[3]より直接SABRパラメータを推計

   このパラメータは上記のDNN近似の初期値としても使用可能

2. 補間/補外

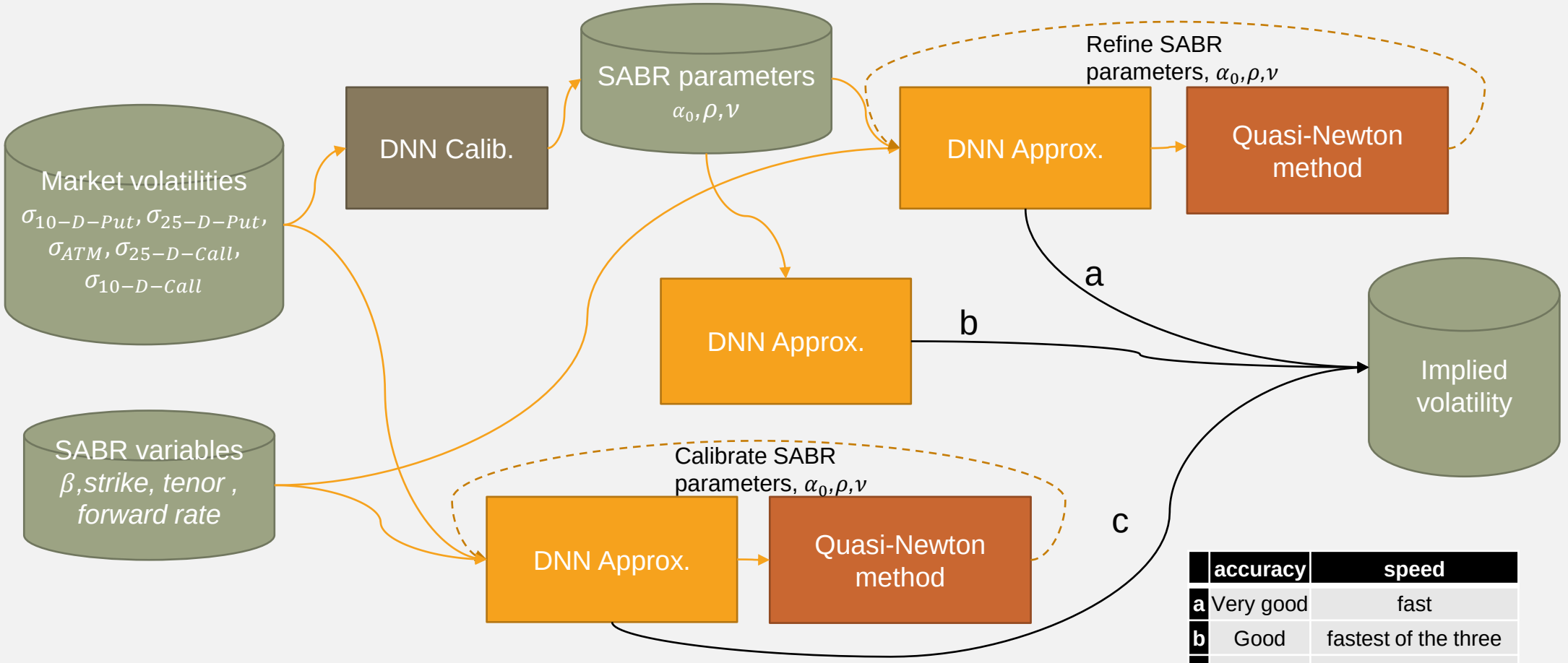
   I. 推計されたパラメータと行使価格をDNN近似式に代入.

   II. 当該行使価格におけるインプライドボラティリティが計算される

3 10%-delta-put, 25%-del-put, ATM, 25%-delta-call, and 10%-delta-callの５種類

どの様にスマイルを描くか(4/15)

- DNN近似と and DNNキャリブレーションの関係図.



	accuracy	speed
a	Very good	fast
b	Good	fastest of the three
c	Good?	slow

どの様にスマイルを描くか(5/15)

- 訓練データの生成 (feature and label) (1/2)
  - GPUの使用
    - GPUによりSABRモデルに従うBlackのインプライド・ボラティリティ (IV) をモンテカルロ・シミュレーションにて高速に生成
    - SABRパラメータに相応な現実的な行使価格を算出するため,ボラと同時にデルタの計算行う
  - 訓練データの分類

SABR model

$$dF_t = \alpha_t F_t^{\beta} dW_1$$
$$d\alpha_t = \nu \alpha_t dW_2$$
$$\rho = dW_1 dW_2$$

手法	Feature data	Label data	Remarks
DNN近似	• SABR parameters α_0, β(fixed)[4], ρ, ν • Strike • Tenor • Forward rate	Implied volatility	➢ Tenorsは市場慣行と一致させる ➢ Strike と IVはGPUにより生成
DNNキャリブレーション	• Five imitated market volatilities $\sigma_{10-D-Put}, \sigma_{25-D-Put}, \sigma_{ATM}, \sigma_{25-D-Call}, \sigma_{10-D-Call}$ • SABR parameter β(fixed)[4] • Tenor • Forward rate	SABR parameters α_0, ρ, ν	➢ 5個のボラティリティデータはDNN近似のキャリブレーションテストにも使用可能

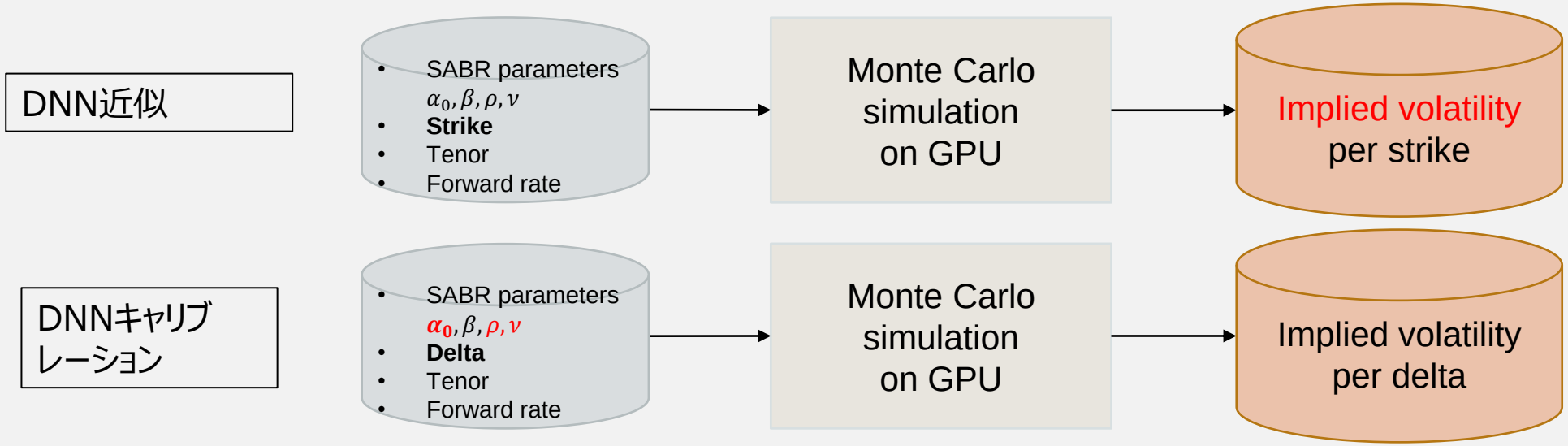
4 $\beta$is fixed according to market convention

どの様にスマイルを描くか(6/15)

- 訓練データの生成 (feature and label) (2/2)
  - Feature data
    - ランダム間隔 – 乱数により生成。後からデータの追加が容易。

      or

    - 等間隔 – 等間隔に生成するのでデータ生成が容易。一方後からのデータ追加には向かない。
  - Label data
    - インプライド・ボラティリティはモンテカルロシ・ミュレーションで生成



どの様にスマイルを描くか(7/15)

- SABRモデルのモンテカルロ・シミュレーション

$$dF_t = \alpha_t F_t^{\beta} dW_1$$
$$d\alpha_t = \nu \alpha_t dW_2$$
$$\rho = dW_1 dW_2$$

時間の離散化 … Euler-Maruyama method

$$F_{i+1} = F_i + \alpha_i F_i^{\beta} \sqrt{\Delta T} w_1$$

$$\alpha_{i+1} = \alpha_i + \nu \alpha_i \sqrt{\Delta T} \left(\rho w_1 + \sqrt{1-\rho^2} w_2 \right) \leftarrow \boldsymbol{fast, but\ not\ so\ accurate}$$

$$or$$

$$\begin{pmatrix} \alpha_{i+1} = \alpha_i \cdot exp\left[-\frac{1}{2}\nu^2 \Delta T + \nu\sqrt{\Delta T}\left(\rho w_1 + \sqrt{1-\rho^2} w_2 \right) \right] \\ or \\ log(\alpha_{i+1}) = log(\alpha_i) + \left[-\frac{1}{2}\nu^2 \Delta T + \nu\sqrt{\Delta T}\left(\rho w_1 + \sqrt{1-\rho^2} w_2 \right) \right] \end{pmatrix} \leftarrow \boldsymbol{accurate, but\ very\ slow}$$

$$ForwardValue_K = \frac{1}{M} \sum_j^M [max(F_{j,T} - K, 0)]$$

- デルタの計算

search $\sigma_{Black,delta} and\ K_{delta},\ delta = \{10\_D_{put}, 25\_D_{put}, ATM, 25\_D_{call}, 10\_D_{call}\}$

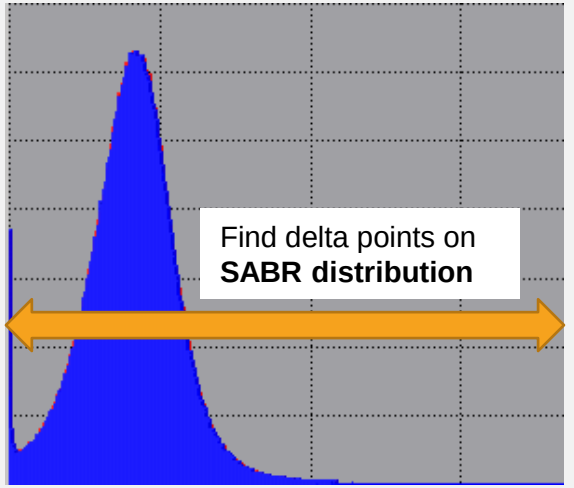
$that\ satisfy$

$$\boldsymbol{\Delta_{F;pips}} = \boldsymbol{N}\left(\frac{\boldsymbol{log}\left(\frac{F_0}{K_{delta}}\right) + \frac{\boldsymbol{1}}{\boldsymbol{2}}\sigma_{Black,delta}^2 T}{\sigma_{Black,delta}\sqrt{\boldsymbol{T}}} \right) = \{0.9, 0.75, 0.5, 0.25, 0.1\}$$

$subject\ to$

$$FV_{Black,delta}(F_0, K_{delta}, T, \sigma_{Black.delta}) = FV_{SABR,delta}(F_0, K_{delta}, T, \alpha_0, \beta, \rho, \nu)$$

on Monte Carlo paths.



どの様にスマイルを描くか(8/15)

- 生成するトレーニングデータの種類　…　通貨ペア毎に作成するか？
  - デルタの計算方法、テナー、β値が同じものは同一トレーニングデータを利用可能
    - 通貨オプションのクオートにおける市場慣行
      - Premium unadjusted … EUR/USD, GBP/USD, AUD/USD, 等々
      - Premium adjusted … その他大半の通貨ペア
    - 市場慣行のデルタ4種類

	delta	
Tenor	≤ 1 year	> 1 year
Premium unadjusted	$\phi e^{-r^f T} N(\phi d_1)$	$\phi N(\phi d_1)$
Premium adjusted	$\phi e^{-r^d T} \frac{K}{S_0} N(\phi d_2)$	$\phi \frac{K}{F_0} N(\phi d_2)$

	Formula
d_1	$= \frac{\boldsymbol{log}\left(\frac{F_0}{K}\right) + \frac{\mathbf{1}}{\mathbf{2}}\sigma^2 T}{\sigma\sqrt{\boldsymbol{T}}}$
d_2	$= d_1 - \sigma\sqrt{\boldsymbol{T}}$
S_0	$= F_0 e^{(r^f - r^d)T}$

- 効率化の為の変換 :$F_0^{training} = \frac{F_0}{F_0} = 1 \;\&\; K^{training} = \frac{K}{F_0}, S_0^{training} = \frac{S_0}{S_0} = 1 \;\&\; K^{training} = \frac{K}{S_0}, \alpha_0^{training} = \frac{\alpha_0}{F_0^{1-\beta}}$
- この変換により、上表の四つのタイプは同じタイプ且つ同じβの場合、同じ訓練データを使用可能。

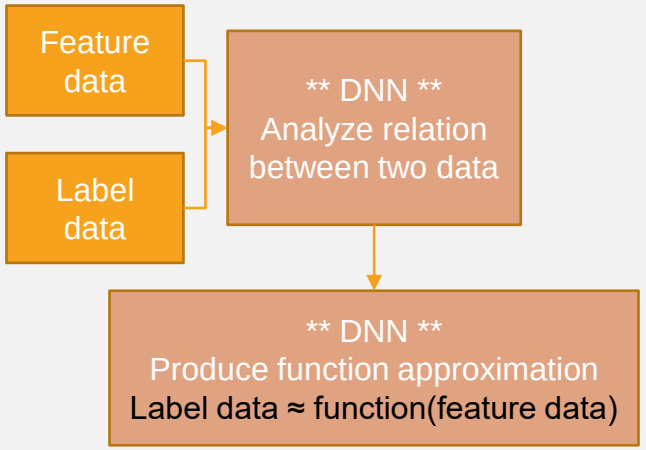
どの様にスマイルを描くか(9/15)

- DNN 学習設定 (1/2)
  - DNN 回帰分析

SABR parameters	DNN Approx.	DNN Calib.
α_0	Feature	Label
β	Fixed	Fixed
ρ	Feature	Label
ν	Feature	Label
Forward rate	Not used (due to all value is 1)	Not used
Strike	Feature	Not used
Tenor	Feature	Feature
Implied volatility	Label	Feature
Interest rate	Not used	Feature (≤ 1 year)
Remarks		Multiple outputs

  - データ件数は最低1万件以上？
  - 生成データの内、一定量はバリデーションデータに割り当て
  - 同様に生成データの内、一定量はテストデータに割り当て

“Label” データは回帰分析の従属変数に該当



どの様にスマイルを描くか(10/15)

- DNN学習設定 (2/2)
  - 機械学習設定

Machine learning parameters	Detail	Remarks
Loss function	Mean squared error	• パラメータの設定内容は多種有
Activation function	ReLU, linear, tanh, …	
Optimization	RMSProp, ADAM, …	
Number of layers	3 -	
Number of nodes	32 -	
Batch size	64 -	
Learning rate	0.001 -	
… … … … … … … … … …	… … … … … …	

どの様にスマイルを描くか(11/15)

- モンテカルロ・シミュレーションの精度

  SABRのIVを計算する他の手法と比較することにより、モンテカルロ手法の精度を確認

  - 比較対象
    - Hagan近似式
    - 有限差分法　Finite Difference Method (FDM)

FDM setting:

using Quantlib　FdSabrVanillaEngine

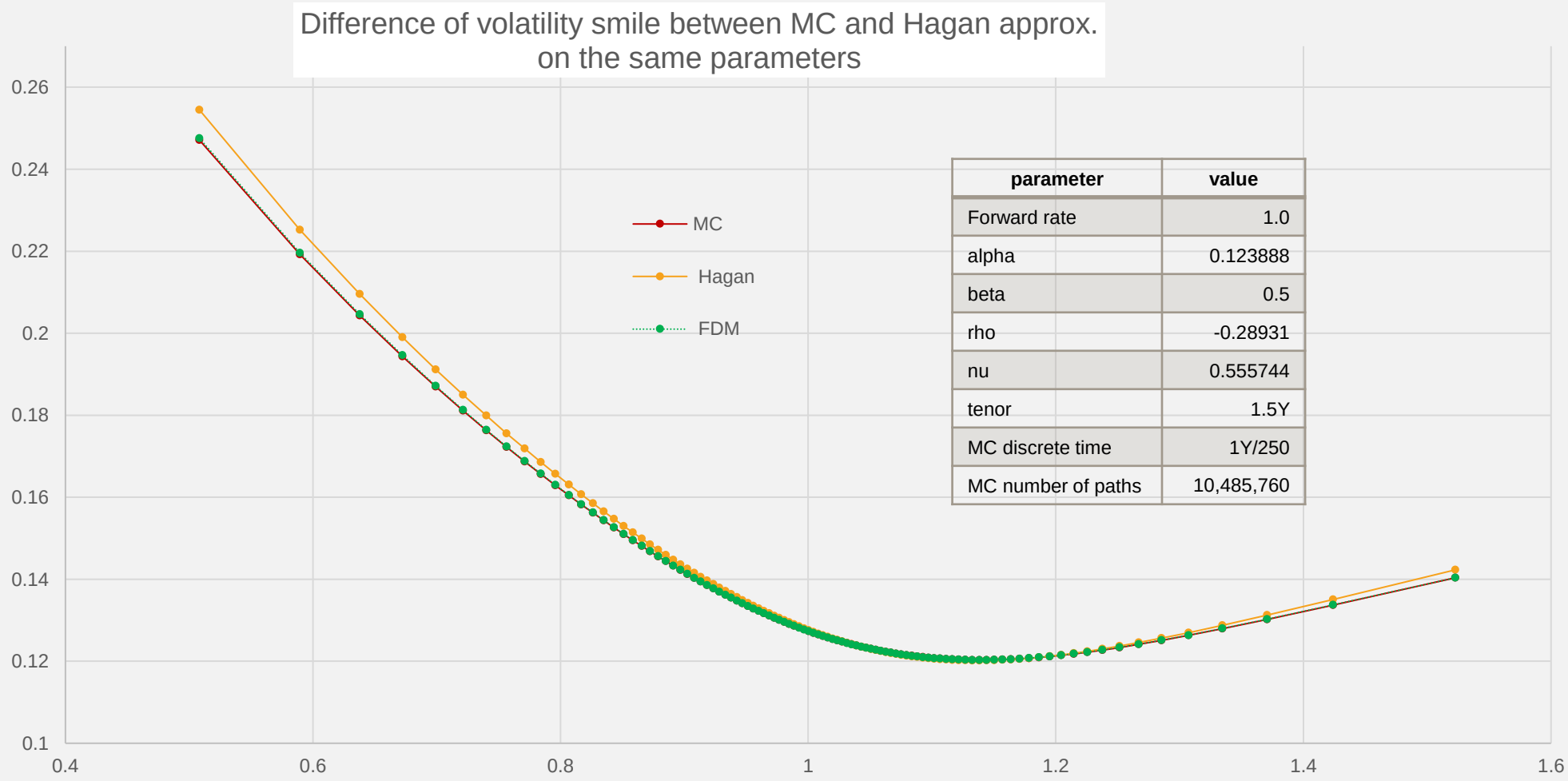
Parameter	Value	Remarks
tGrid	100	Number of time discretization
fGrid	400	Number of underlying price discretization
xGrid	100	Number of volatility of volatility discretization

他のパラメータは変更せず

またHagan近似はオリジナルのまま使用

どの様にスマイルを描くか(12/15)

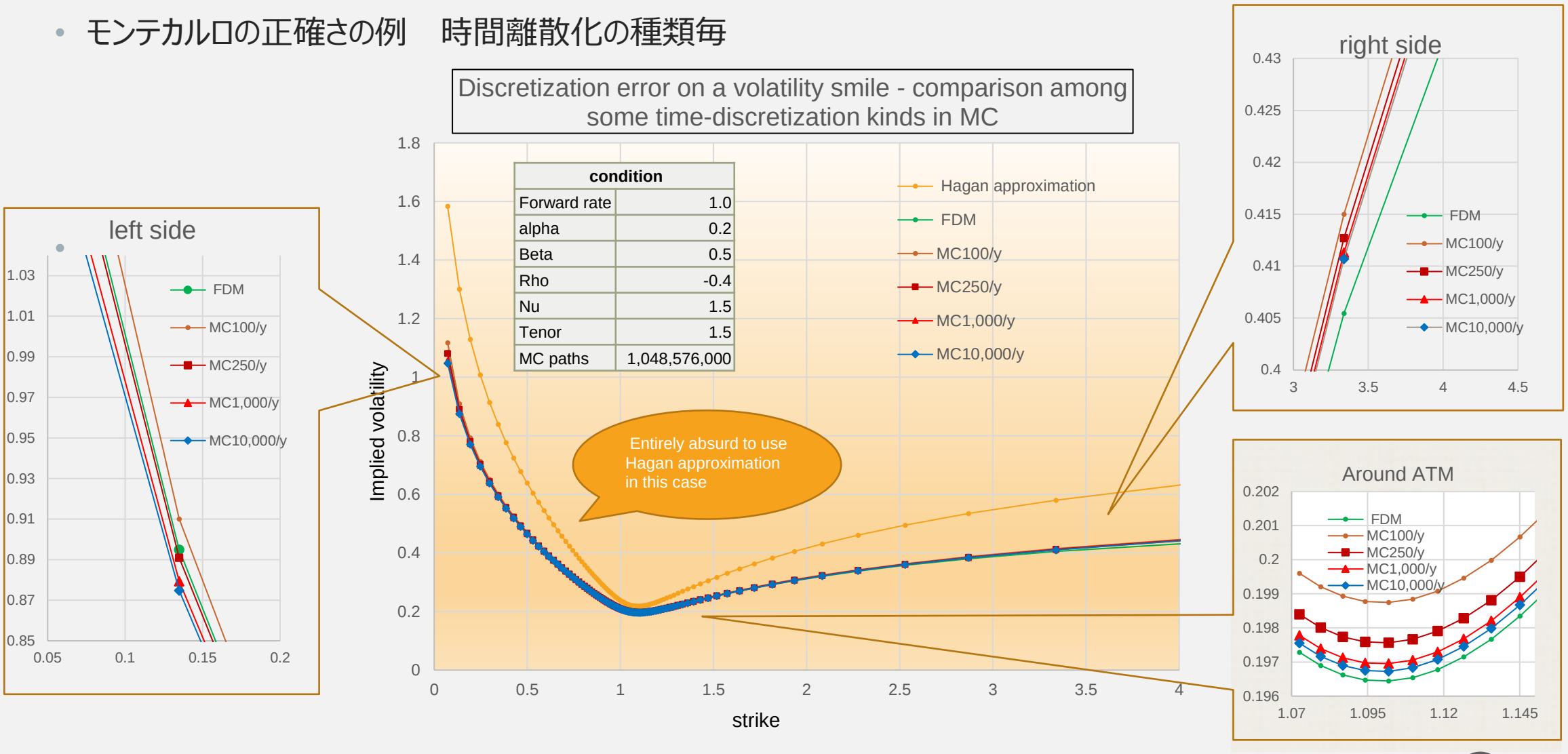
- スマイルの例



parameter	value
Forward rate	1.0
alpha	0.123888
beta	0.5
rho	-0.28931
nu	0.555744
tenor	1.5Y
MC discrete time	1Y/250
MC number of paths	10,485,760

どの様にスマイルを描くか(13/15)

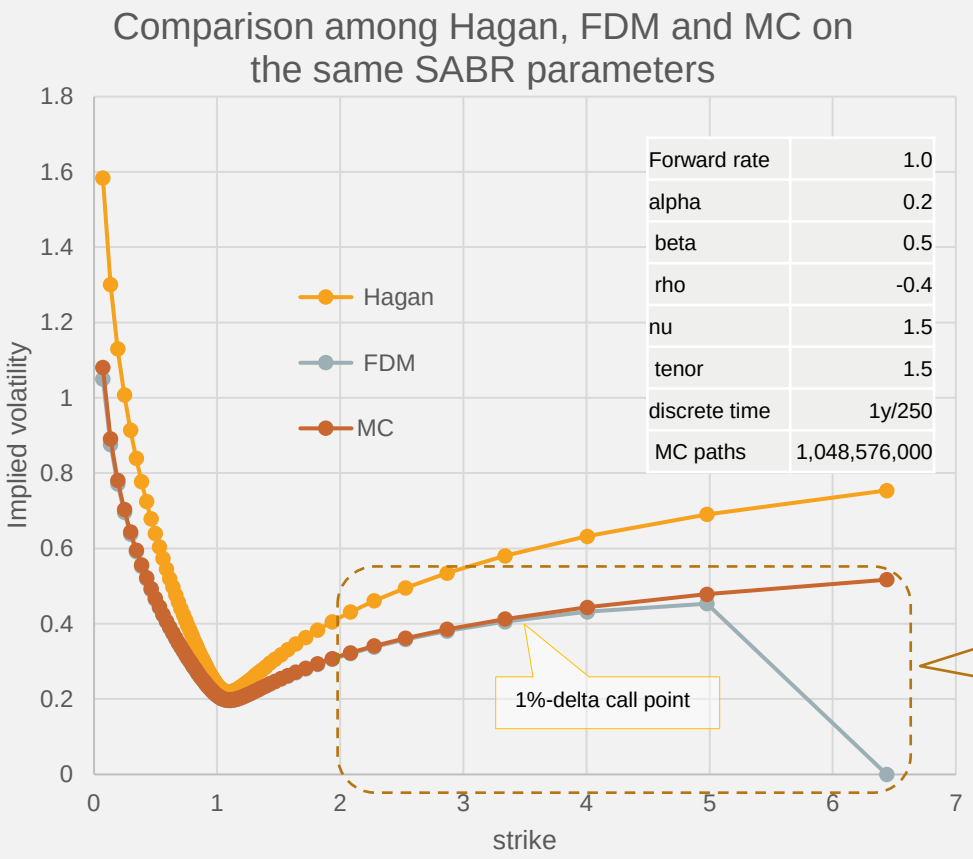
- モンテカルロの正確さの例　時間離散化の種類毎

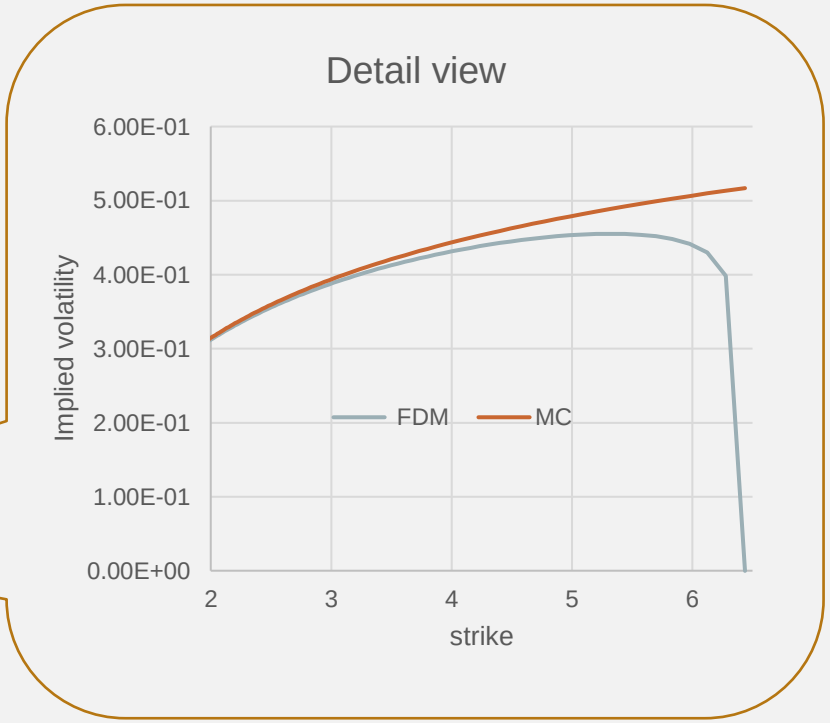


どの様にスマイルを描くか(14/15)

- FDM error の例

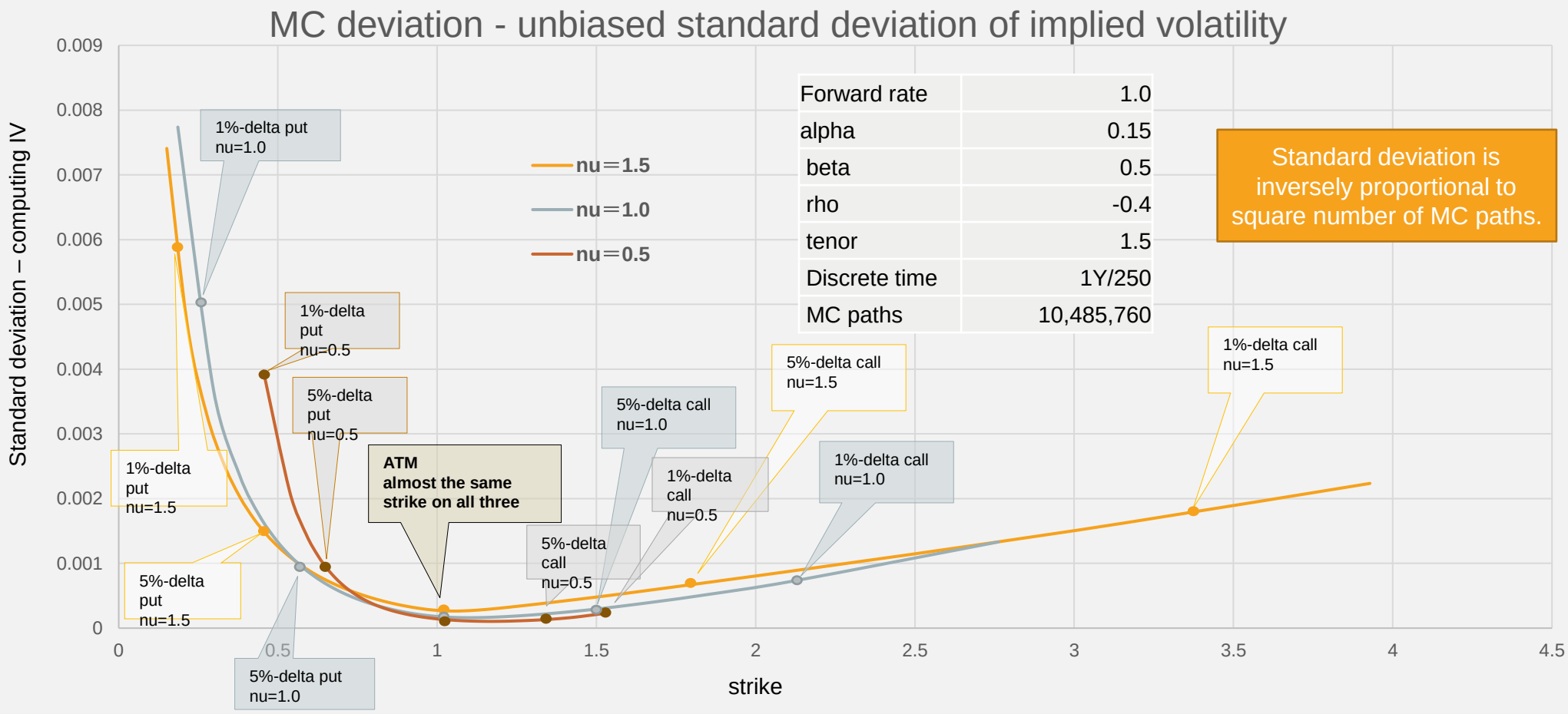
FDMではパラメータの上下限の設定が必須
このため、行使価格が大きい（小さい）とこの上限（下限）に達し数値が狂う





どの様にスマイルを描くか(15/15)

- モンテカルロの精度例



プロトタイプテスト（1/19）

- 訓練データの設定 (1/2)

	Prototype setting	Remarks
PGM language	CUDA v.11.7	
Tenor	1.5 year	For 1 or less year, interest rate is needed to compute delta. It is troublesome for this prototype. This value means **375/250, business day base.**[5]
Currency pair	EUR/USD	
Delta	7 quoted deltas	Adding 5%-delta put and 5%-delta call as loqer/upper limits of strikes
Strike range	5%-delta put ≤ strike ≤ 5%-delta call	Extremely small or big strike value is omitted. Most users mightn't prefer extrapolation in far wings.
α	0.05 – 0.3	Using random number generator
β	Fixing to 0.5	
ρ	-0.9 – 0.9	Using random number generator
ν	0.05 – 1.5	Using random number generator
MC Discrete tool	Euler-Maruyama	
MC Discrete time	1y/250	
MC Number of paths	10 million	
Data precision	Float32 (single precision)	Significant digits of market volatility is covered enough by float32.

5 実務では休日調整を考慮し、営業日ベースまたは暦日ベースのデータを作成するか、もしくはテナーも教師データとして学習する必要あり。

プロトタイプテスト(2/19)

- 訓練データ生成の設定(2/2)

Data image

Input data example

fwdrate	tenor	alpha	beta	rho	volvol
1	1.5	0.065315357	0.5	-0.35811	0.525657
1	1.5	0.242011988	0.5	0.863899	1.260464
1	1.5	0.165297476	0.5	0.180486	0.989391
1	1.5	0.156737171	0.5	-0.03611	0.054857
…	…	…	…	…	…
1	1.5	0.284696963	0.5	0.183503	0.858997
1	1.5	0.243779974	0.5	-0.71787	1.42026

同一行のデータはすべて同時に算出

output data example

σ5dp	σ10dp	σ25dp	σATM	σ25dc	σ10dc	σ5dc	K5dp	K10dp	K25dp	KATM	K25dc	K10dc	K5dc	Strike	Volatility
0.095434	0.085922	0.074367	0.066575	0.063093	0.063264	0.064822	0.830754	0.878689	0.944325	1.00333	1.056652	1.107715	1.143092	1.063823	0.062926
0.25222	0.213786	0.212114	0.289299	0.467208	0.754917	0.97634	0.631036	0.739874	0.868074	1.064782	1.73267	5.0145	14.61181	13.2204	0.957622
0.285801	0.23774	0.193168	0.184423	0.213578	0.272548	0.323987	0.597805	0.718378	0.876704	1.025837	1.234473	1.621734	2.078002	1.836106	0.298777
0.16994	0.166625	0.161354	0.156006	0.151172	0.147252	0.14508	0.72565	0.786072	0.892468	1.018421	1.1526	1.280673	1.360776	1.219458	0.149054
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
0.447552	0.382823	0.322718	0.309517	0.341956	0.404549	0.455178	0.471716	0.612042	0.828217	1.074495	1.447999	2.133395	2.922271	2.669083	0.440883
0.605392	0.466604	0.308858	0.214163	0.180222	0.190915	0.210938	0.388796	0.566046	0.832272	1.034998	1.189148	1.386789	1.581401	1.293205	0.18271

Output data description

Output	Description	Usage
σ5dp - σ5dc	Seven volatilities, $\sigma_{5\%d\ put}, \sigma_{10\%d\ put}, \sigma_{25\%d\ put}, \sigma_{ATM}, \sigma_{25\%d\ call}, \sigma_{10\%d\ call}, and\ \sigma_{5\%d\ call}$	• As training data for DNN Approx. and DNN-calibration except both 5% put and call • For calibration test for DNN Approx.
K5dp - K5dc	Seven strikes, $K_{5\%d\ put}, K_{10\%d\ put}, K_{25\%d\ put}, K_{ATM}, K_{25\%d\ call}, K_{10\%d\ call}, and\ K_{5\%d\ call}$	For checking smile curve to draw
Strike	Chosen any value from $K_{5\%d\ put}$ to $K_{5\%d\ call}$	As feature data for DNN Approx.
Volatility	Black implied volatility coincided with the strike above.	As label data for DNN Approx.

プロトタイプテスト(3/19)

- DNN学習設定

	DNN Approx.	DNN Calib.	Remarks
Machine learning library	TensorFlow 2.8.0	TensorFlow 2.8.0	
Loss function	MSE	MSE	
Activation function	ReLU	ReLU, linear,…	
Optimization	ADAM	ADAM	
Epochs	5,000	5,000	
Bach size	8,192	4096	Using tf.data.Dataset
Number of training data	720,000	144,000	
Number of validation data	80,000	16,000	
Number of test data	200,000	40,000	
Number of layers	6	9	Details are confidential matter
Number of nodes	Max 256	Max 512	Details are confidential matter
Number of inputs	4	5	Tenor, β, and forward rate are fixed.
Number of outputs	1 (implied volatility)	3 (α_0, ρ, ν)	

プロトタイプテスト(4/19)

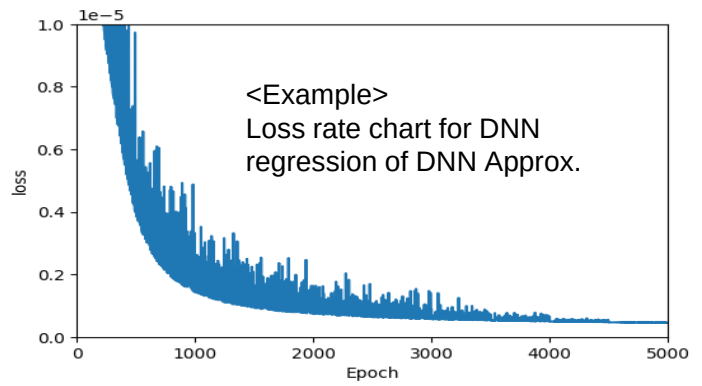
- TensorFlow実行

  損失関数が納得のいく数値になるまで、Tensorflowで使用する各種ハイパーパラメータを繰り返し変更し調整

  経験の蓄積が必要

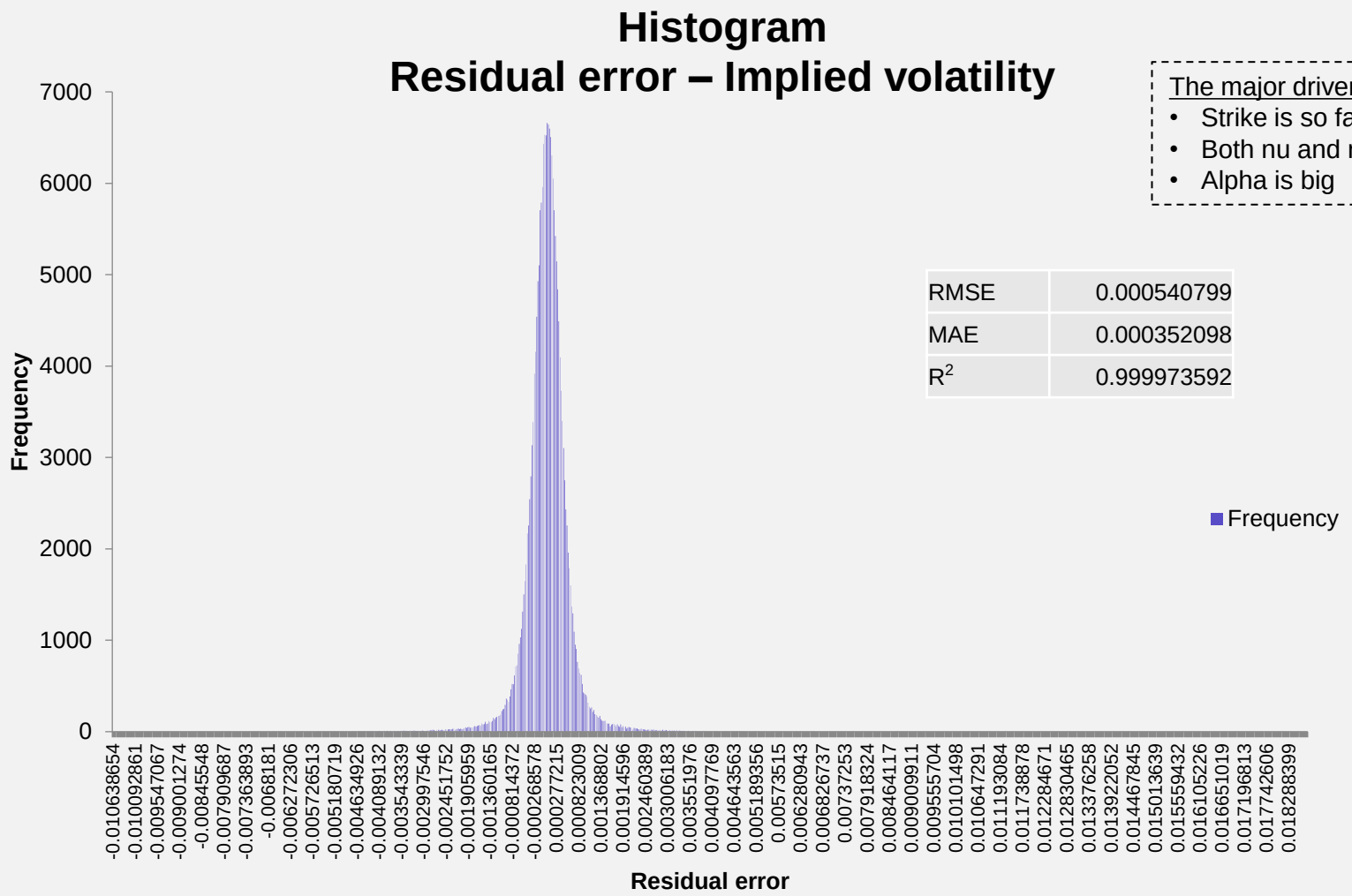
<tensorFlow log example - learning DNN regression of calibration>

```
Epoch 1/5000
INFO:tensorflow:batch_all_reduce: 30 all-reduces with algorithm = nccl, num_packs = 1
INFO:tensorflow:Reduce to /job:localhost/replica:0/task:0/device:CPU:0 then broadcast to ('/job:localhost/replica:0/task:0/device:CPU:0',).
INFO:tensorflow:Reduce to /job:localhost/replica:0/task:0/device:CPU:0 then broadcast to ('/job:localhost/replica:0/task:0/device:CPU:0',).
INFO:tensorflow:Reduce to /job:localhost/replica:0/task:0/device:CPU:0 then broadcast to ('/job:localhost/replica:0/task:0/device:CPU:0',).
INFO:tensorflow:batch_all_reduce: 30 all-reduces with algorithm = nccl, num_packs = 1
6/6 [==============================] - 9s 429ms/step - loss: 0.8472 - alpha_loss: 1.7812 - rho_loss: 0.2859 - volvol_loss: 0.3798 - alpha_mae: 1.3328 - alpha_mse: 1.7812 - rho_mae:
0.4624 - rho_mse: 0.2859 - volvol_mae: 0.5365 - volvol_mse: 0.3798 - val_loss: 0.8261 - val_alpha_loss: 1.7473 - val_rho_loss: 0.2529 - val_volvol_loss: 0.3741 - val_alpha_mae: 1.3200 -
val_alpha_mse: 1.7473 - val_rho_mae: 0.4353 - val_rho_mse: 0.2529 - val_volvol_mae: 0.5321 - val_volvol_mse: 0.3741
Epoch 2/5000
6/6 [==============================] - 1s 194ms/step - loss: 0.8108 - alpha_loss: 1.7230 - rho_loss: 0.2328 - volvol_loss: 0.3660 - alpha_mae: 1.3108 - alpha_mse: 1.7230 - rho_mae:
0.4166 - rho_mse: 0.2328 - volvol_mae: 0.5236 - volvol_mse: 0.3660 - val_loss: 0.7932 - val_alpha_loss: 1.6897 - val_rho_loss: 0.2075 - val_volvol_loss: 0.3601 - val_alpha_mae: 1.2981 -
val_alpha_mse: 1.6897 - val_rho_mae: 0.3918 - val_rho_mse: 0.2075 - val_volvol_mae: 0.5188 - val_volvol_mse: 0.3601
Epoch 3/5000
```



プロトタイプテスト(5/19)

- DNN近似の学習結果

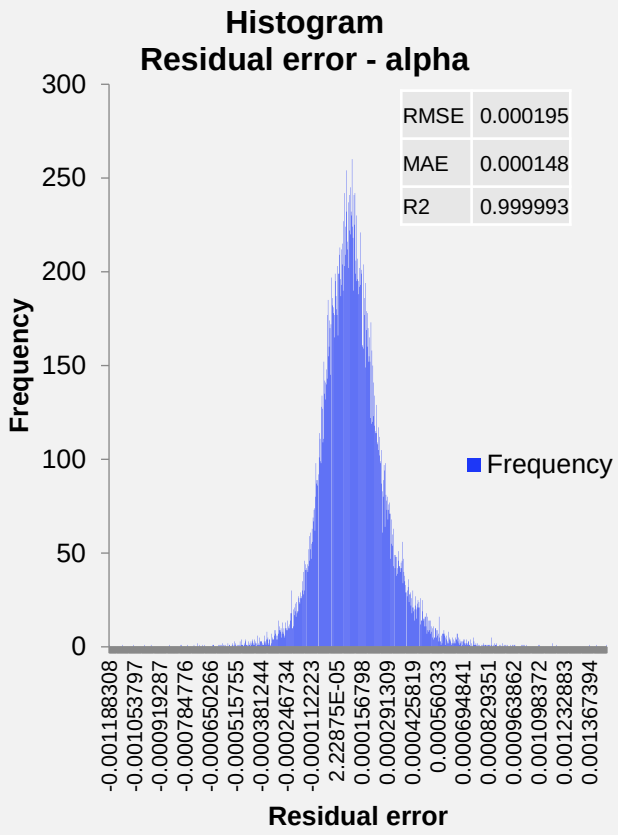


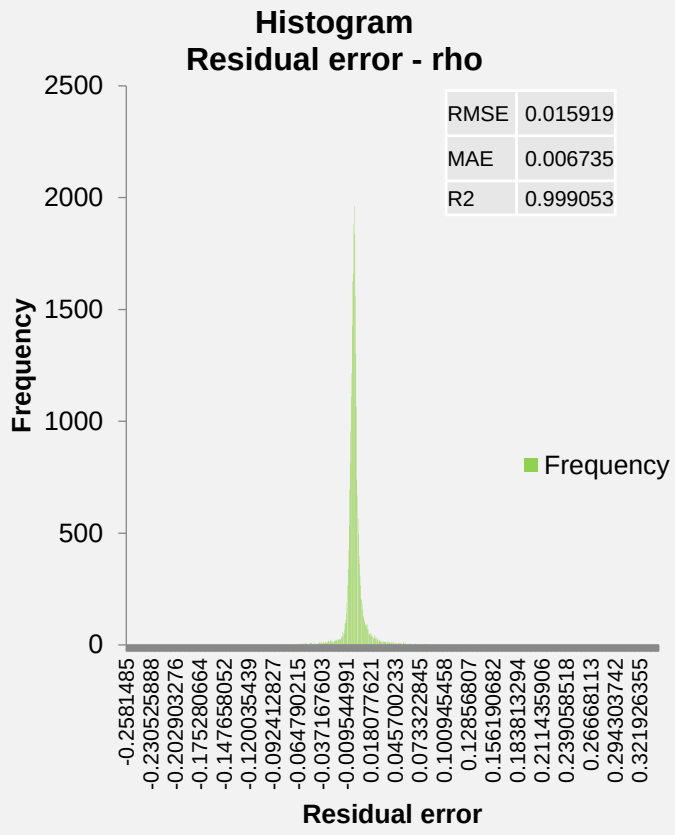
The major driver of error
- Strike is so far from ATM
- Both nu and rho are big
- Alpha is big

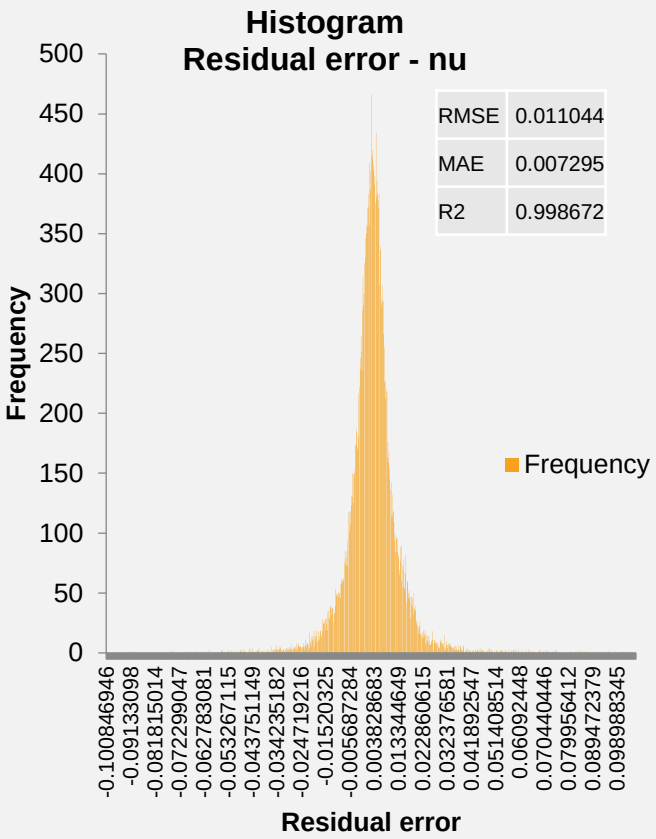
RMSE	0.000540799
MAE	0.000352098
R^2	0.999973592

プロトタイプテスト(6/19)

- DNNキャリブレーションの学習結果







The causes of error are the same as DNN approx. in the last page.

プロトタイプテスト(7/19)

- 数値検証 –DNN近似によるスマイル (1/2)
- 例
  - 設定

	Case 1	Case 2	Case 3	Case 4
Forward rate	1.0	1.0	1.0	1.0
tenor	1.5	1.5	1.5	1.5
α	0.1	0.1	0.2	0.2
β	0.5	0.5	0.5	0.5
ρ	-0.5	0	-0.5	0.5
ν	0.5	0.5	1.5	1.5

  - 結果 --- 次ページにグラフ有

Case 1			
strike	FDM	DNN Approx	difference
0.744988	0.15273	0.153583	0.000853345
0.817327	0.135825	0.135751	-7.39E-05
0.860714	0.12647	0.12625	-0.000220031
0.892419	0.120033	0.119972	-6.19E-05
0.917862	0.115139	0.115228	8.99E-05
0.93947	0.111193	0.111505	0.000311844
0.958557	0.10789	0.10805	0.000160024
0.97593	0.105047	0.105023	-2.45E-05
0.992136	0.102552	0.102595	4.30E-05
1.007584	0.100328	0.100494	0.000165835
1.022611	0.098322	0.098631	0.000308983
1.037527	0.096496	0.096737	0.000240341
1.052655	0.094825	0.094995	0.000169508
1.068381	0.093291	0.093606	0.000314832
1.085222	0.091886	0.092194	0.000307918
1.103983	0.090616	0.090717	0.000100926
1.126118	0.089512	0.089784	0.000271902
1.154866	0.08868	0.08888	0.000199206
1.200745	0.088558	0.088498	-6.05E-05

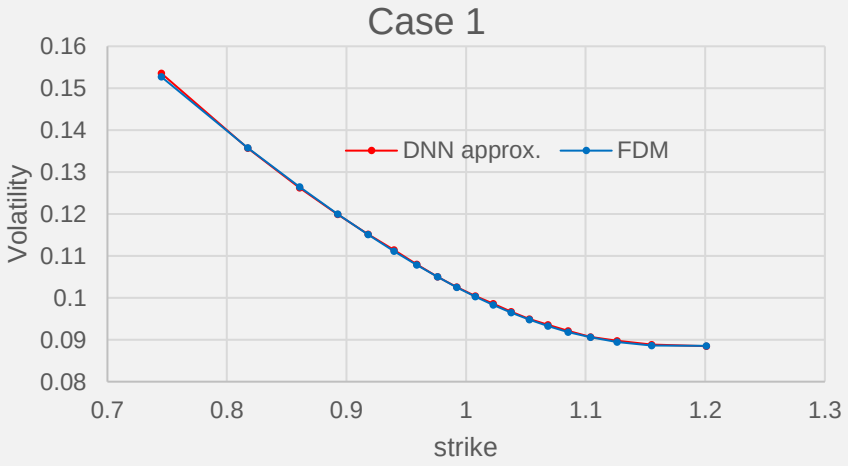
Case 2			
strike	FDM	DNN Approx	difference
0.781724	0.127177	0.127106	-7.17E-05
0.839064	0.117714	0.117754	4.07E-05
0.874329	0.112842	0.112663	-0.00017932
0.900848	0.10973	0.109248	-0.00048204
0.922784	0.107551	0.107548	-2.73E-06
0.942018	0.105956	0.106147	0.00019082
0.959581	0.104769	0.104866	9.72E-05
0.97613	0.103891	0.104042	0.00015128
0.992136	0.103264	0.103422	0.00015758
1.007983	0.102855	0.10281	-4.55E-05
1.024023	0.102648	0.1025	-0.0001476
1.040625	0.102641	0.10245	-0.00019132
1.058229	0.102846	0.10278	-6.53E-05
1.077415	0.103294	0.103285	-9.09E-06
1.099035	0.104043	0.104378	0.0003352
1.124495	0.105203	0.105413	0.00021067
1.156453	0.106989	0.106679	-0.00031014
1.201044	0.10991	0.109848	-6.23E-05
1.278944	0.115646	0.11543	-0.00021568

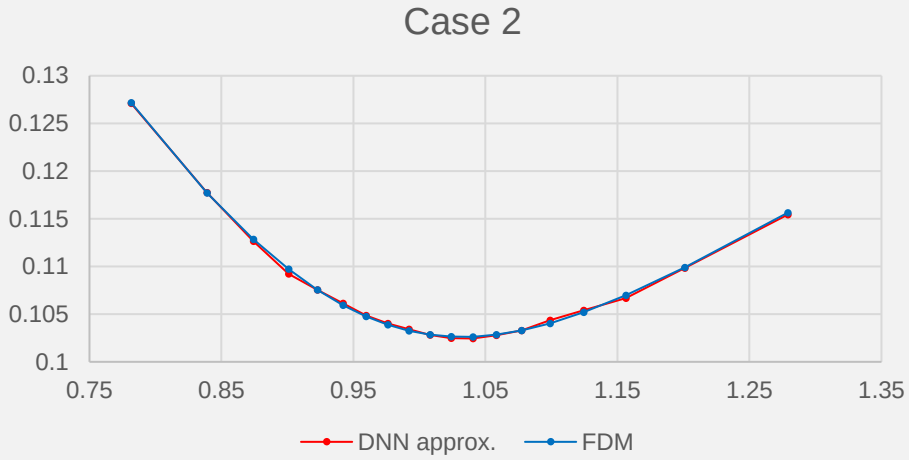
Case 3			
strike	FDM	DNN Approx	difference
0.30069	0.639747	0.621811	-0.01793599
0.501761	0.466258	0.470156	0.00389779
0.642868	0.374972	0.376667	0.00169501
0.745378	0.317491	0.319065	0.00157374
0.822366	0.278126	0.279773	0.00164709
0.882052	0.249899	0.250678	0.00077982
0.929849	0.229145	0.229794	0.00064841
0.96952	0.213738	0.21421	0.00047167
1.003868	0.202357	0.203814	0.00145756
1.035196	0.194167	0.195308	0.00114141
1.065665	0.188678	0.189899	0.0012209
1.097674	0.185705	0.18688	0.00117552
1.134424	0.185409	0.186307	0.00089839
1.180995	0.188433	0.189262	0.00082877
1.246842	0.196195	0.197011	0.00081591
1.352948	0.211606	0.213127	0.00152114
1.557525	0.241021	0.24142	0.00039996
2.091997	0.300515	0.30145	0.00093496

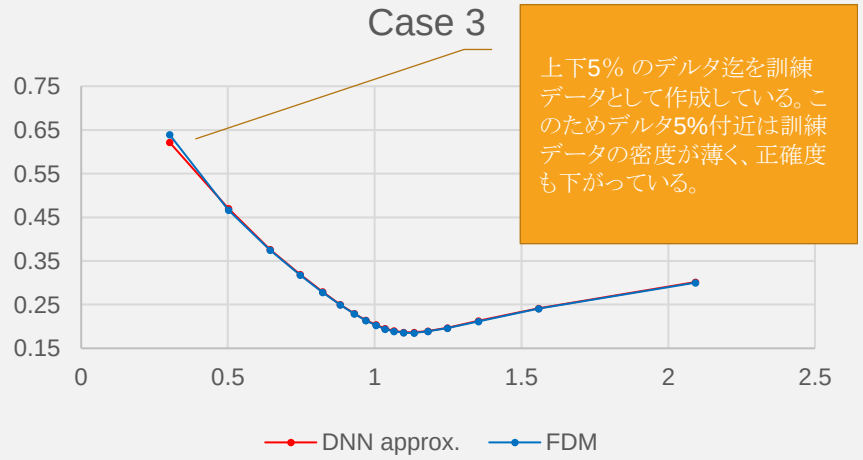
Case 4			
strike	FDM	DNN Approx	difference
0.429082	0.434533	0.432679	-0.00185373
0.631139	0.314932	0.321948	0.00701639
0.741561	0.264958	0.271092	0.00613421
0.811164	0.239924	0.244146	0.00422134
0.86051	0.227082	0.232413	0.00533143
0.899608	0.221371	0.227595	0.00622414
0.934303	0.220558	0.226351	0.00579239
0.968813	0.223799	0.229645	0.00584607
1.007065	0.231095	0.236724	0.0056289
1.053788	0.243097	0.248412	0.00531507
1.116006	0.261097	0.264906	0.00380808
1.205923	0.28721	0.29144	0.00423074
1.34777	0.324801	0.331852	0.00705174
1.59667	0.379176	0.388411	0.009235
2.09807	0.457561	0.472812	0.01525077
3.286737	0.559591	0.595245	0.03565383

プロトタイプテスト(8/19)

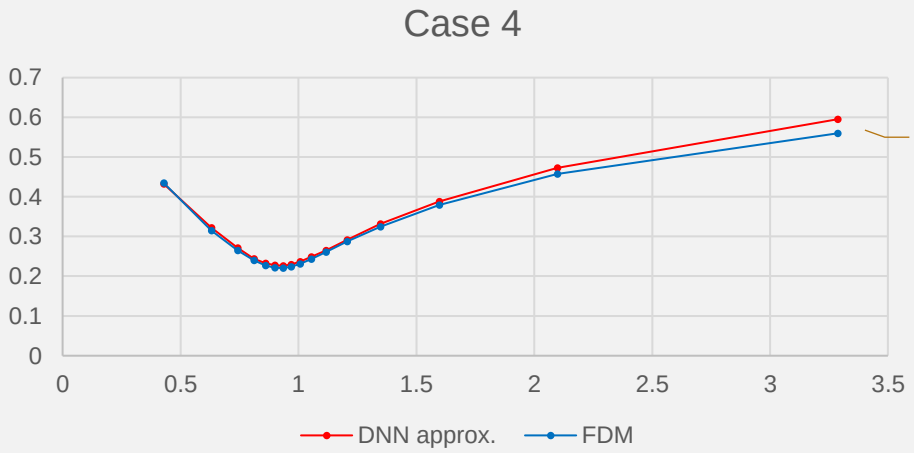
- 数値結果 –DNN近似によるスマイル (2/2)







上下5% のデルタ迄を訓練データとして作成している。このためデルタ5%付近は訓練データの密度が薄く、正確度も下がっている。



この差はFDMの正確度と、DNN近似の端っこでの正確度、双方の問題に起因。

プロトタイプテスト(9/19)

- 数値結果 – キャリブレーションとスマイルカーブ作成 (1/7)
  - α_0, ρ, νを実マーケットのボラティリティにキャリブレーションしスマイルカーブを作成
  - 他の手法と結果比較

補間/補外　手法	設定 / 使用ツール
Vanna volga	• Strike < 10% delta put - using 10% delta put/call volatility • 10%delta put ≤ strike < 25% delta put - using proportion of both 10% delta put volatility and 25% delta put volatility • 25%delta put ≤ strike ≤ 25% delta call - using 25% delta put/call volatility • 25%delta call < strike ≤ 10% delta put - using proportion of both 10% delta call volatility and 25% delta call volatility
Hagan近似	• Calibration: least_square() function in scipy • Interpolation/extrapolation: Hagan' approximation
DNN近似	• Calibration: least_square() function in scipy • Interpolation/extrapolation: DNN Approx.
DNNキャリブレーション	• Calibration: keras model.predict_on_batch function • Interpolation/extrapolation: DNN Approx.

プロトタイプテスト(10/19)

- 数値結果 – キャリブレーションとスマイルカーブ作成(2/7)
  - 実際の市場データ
    - 通貨ペア EUR/USD
    - テナー 18M

Data date	Feb. 19, 2010		Mar. 13, 2013		Nov. 26, 2014		Sep. 17, 2018	
Forward rate	1.3366		1.3099		1.2544		1.2194	
Strike/IV	**strike**	**IV**	**strike**	**IV**	**strike**	**IV**	**strike**	**IV**
Extrapolation 1	1.002445		1.040789		1.003509		1.036479	
10-delta put	1.062461	0.157805	1.091882	0.118129	1.072153	0.105208	1.056535	0.095456
Interpolation 1	1.069274		1.170887		1.128947		1.097448	
25-delta put	1.20922	0.136976	1.204	0.102263	1.169026	0.091841	1.14335	0.083218
Interpolation 2	1.269763		1.235937		1.191667		1.158418	
ATM	1.352565	0.125513	1.309334	0.091422	1.261046	0.082993	1.224765	0.075346
Interpolation 3	1.470252		1.366035		1.317105		1.280356	
25-delta call	1.497181	0.122882	1.407418	0.086945	1.348958	0.080457	1.303164	0.073859
Interpolation 4	1.603912		1.431085		1.379825		1.341326	
10-delta call	1.654783	0.126944	1.504751	0.087059	1.440318	0.082585	1.38775	0.07725
Extrapolation 2	1.670741		1.561183		1.505263		1.463265	

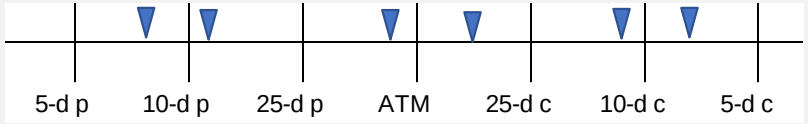
Volatilities are calculated using simple arithmetic, "market strangle."

$$\sigma^{C}_{delta} = \sigma_{ATM} + \frac{1}{2} RR_{delta} + BF_{delta}$$

$$\sigma^{P}_{delta} = \sigma_{ATM} - \frac{1}{2} RR_{delta} + BF_{delta}$$

- 補間 / 補外

5つの市場ボラティリティを補完補外する6つの行使価格を任意に選択。補外分はデルタ5%put/call内に調整。



プロトタイプテスト(11/19)

- 数値結果 – キャリブレーションとスマイルカーブ作成 (3/7)
  - 計算されたパラメータ

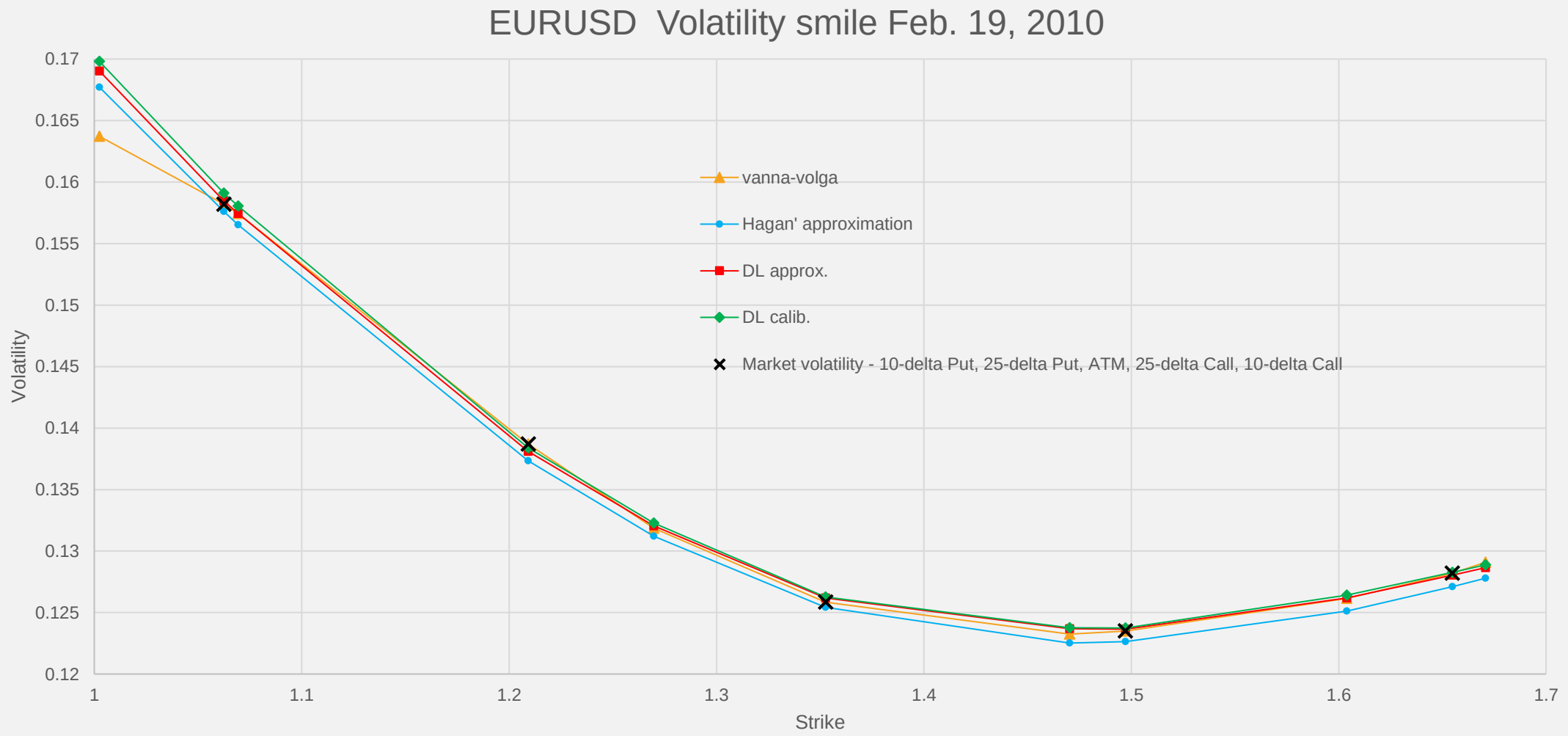
Data date	Feb. 19, 2010			Mar. 13, 2013			Nov. 26, 2014			Sep. 17, 2018		
	α	ρ	ν	α	ρ	ν	α	ρ	ν	α	ρ	ν
Hagan' approximation	0.124432	-0.17889	0.537617	0.07532101	-0.30901	0.506137	0.091739	-0.22419	0.524591	0.083126	-0.18343	0.553797
DNN Approx.	0.122935	-0.17374	0.556933	0.07415797	-0.31312	0.521621	0.090861	-0.23239	0.552005	0.081958	-0.18969	0.584813
DNN Calib.	0.12295	-0.17647	0.564586	0.074331	-0.31199	0.541931	0.091146	-0.22764	0.565128	0.082298	-0.18846	0.596481

  - 5つの市場ボラティリティに対する残差平方和

Data date	Feb. 19, 2010	Mar. 13, 2013	Nov. 26, 2014	Sep. 17, 2018	Remarks
Vanna-volga	3.11E-17	1.06E-17	1.07E-16	5.24E-17	Pass surely on the delta points
Hagan' approximation	4.31E-06	4.42E-06	5.19E-06	5.39E-06	Calibration hard
DNN Approx.	5.83E-07	2.85E-07	7.64E-07	3.25E-07	Pretty good?
DNN Calib.	1.14E-06	3.56E-06	2.89E-06	1.5E-06	Not bad?

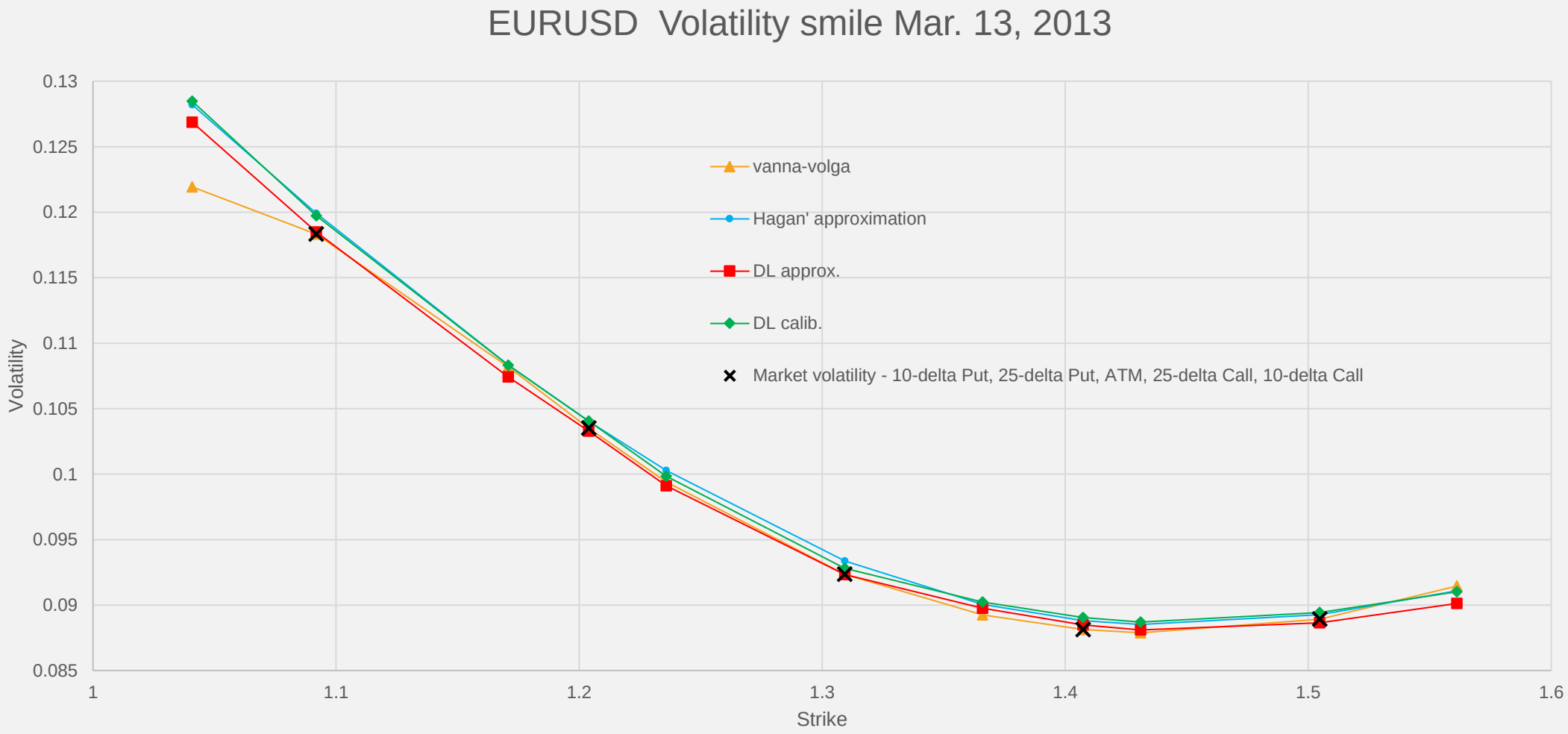
プロトタイプテスト(12/19)

- 数値結果 – キャリブレーションとスマイルカーブ作成 (4/7)



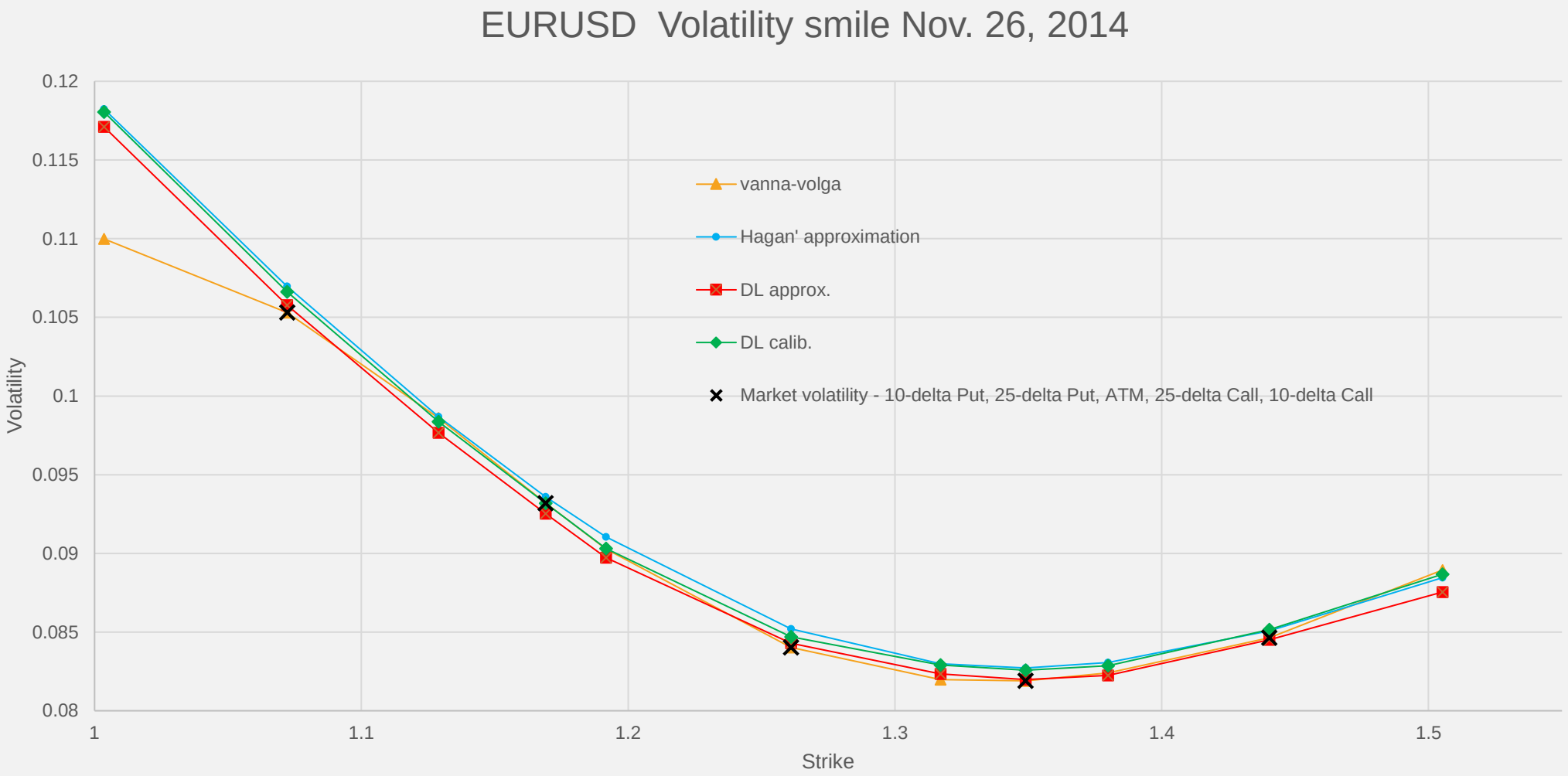
プロトタイプテスト(13/19)

- 数値結果 – キャリブレーションとスマイルカーブ作成 (5/7)



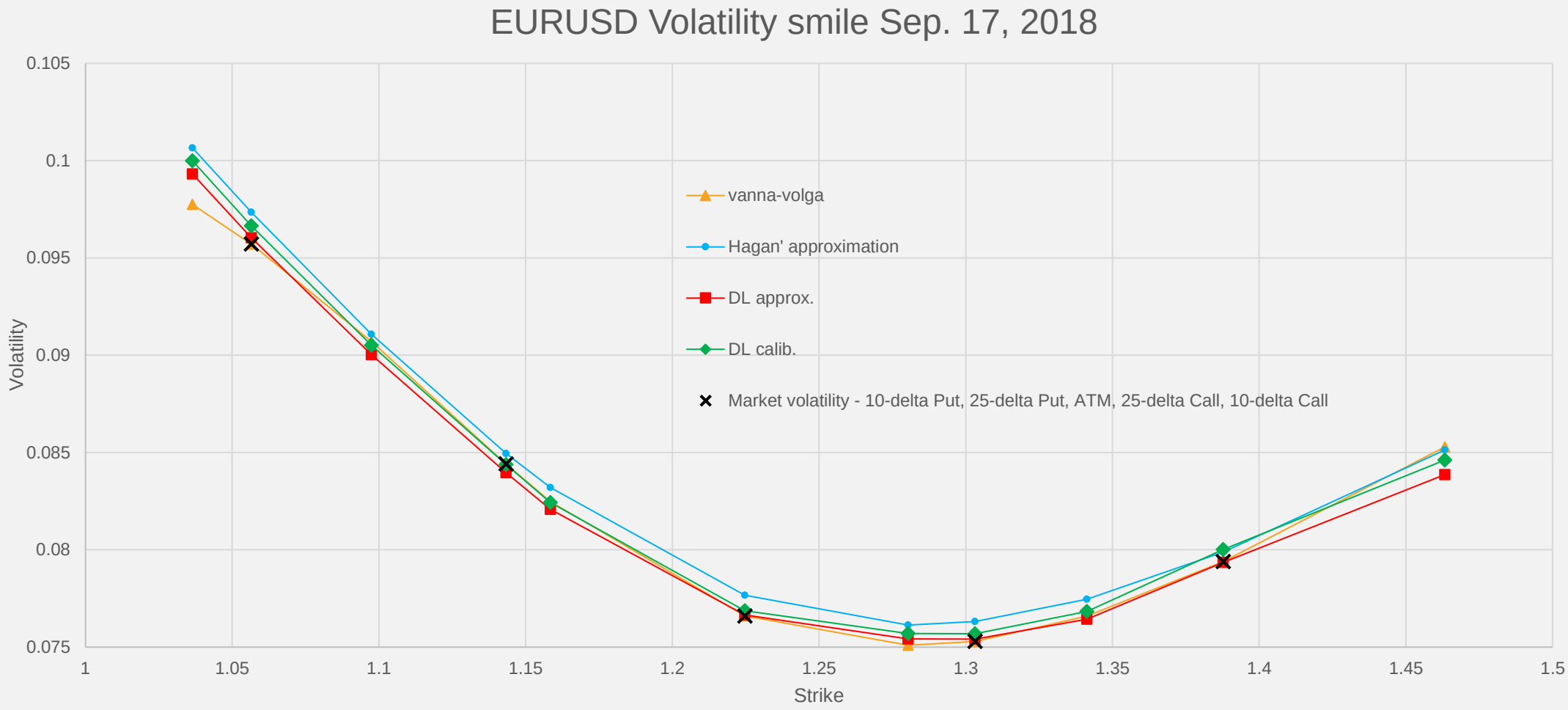
プロトタイプテスト(14/19)

- 数値結果 – キャリブレーションとスマイルカーブ作成 (6/7)



プロトタイプテスト(15/19)

- 数値結果 – キャリブレーションとスマイルカーブ作成 (7/7)



プロトタイプテスト(16/19)

- パフォーマンス (1/2)
  - 訓練データ生成

	Value / number
MC condition	
Number of MC path	10,000,000
Number of time discretization	250/y
Output data	σ5dp, σ10dp, σ25dp, σATM, σ25dc, σ10dc, σ5dc, K5dp, K10dp, K25dp, KATM, K25dc, K10dc, K5dc, Arbitrary Strike, Volatility … 16 data
Elapsed time a MC simulation	3.25 seconds
Elapsed time for generating 1 million volatilities	3.25 x 1,000,000 / 8 ≈ 4.70 days

  - DNN学習[6]

	Time
DNN Approx. 1 epoch	1.34 seconds
DNN Calib. 1epoch	2.16 seconds

6 tensorflow付属のtf.data.Dataset機能を使用することで, 5-10倍高速化

プロトタイプテスト(17/19)

- パフォーマンス (2/2)
  - キャリブレーション[7]

	Time
DNN Approx. using scypy' least_square() and tf.model.predit_on_batch()	キャリブレーション1件あたり　About 1 seconds
DNN Calib.	キャリブレーション1件あたり　About 0.1 seconds

7 推計に使用しているtf.model.predict_on_batch はPythonで作成されている為、計算速度が遅い。

プロトタイプテスト(18/19)

- Machine environment for prototyping
  - 訓練データ生成
    - OS　　Windows 11 Professional
    - Memory　　32GB
    - CPU　　Core i7-10750H
    - GPU　　RTX2070 Super x 1
  - DNN
    - OS　　Ubuntu 18.04.6
    - Memory　　64GB
    - CPU　　Xeon(R) CPU E5-2650 v2 x 2
    - GPU　　Tesla K20C x 2

プロトタイプテスト(19/19)

- プロトタイプで使用したGPUと最新のGPUのパフォーマンス比較

  最新のGPUはプロトタイプで使用したGPUを圧倒的に上回るパフォーマンス。最新のGPUを使用すれば訓練データ生成もDNN学習劇的にパフォーマンスが改善！

2022年10月末現在

	Prototyping	Latest	Prototyping	Latest
Model	RTX2070Super	RTX4090	Tesla K20C	Tesla H100
FP32	7.066 TFLOPS	73.073 TFLOPS	3.5 TFLOPS	66.9 TFLOPS
FP64	0.2 TFLOPS	1.142 TFLOPS	1.2 TFLOPS	33.5 TFLOPS
Remarks	• Sale date Jan. 2019 • Used for prototyping to generate training data • For consumer	• Sale date Sep. 2022 • For consumer	• Sale date Nov. 2012 • Used for prototyping to execute TensorFlow • For business	• Sale date Mar. 2022 • For business

業務での使用例（1/2）

- リスク計量
  - FRTB-IMAのボラティリティ・サーフェス推移計算
  - グリークス計算
- 会計
  - 財務会計、管理会計でのオプション価格算出
  - xVA計算
- フロント
  - vanna-volgaの代替
- R&D
  - 既存のSABR計算機能の検証用または代替手段
  - 新規の補完補外機能の較検証

等々

業務での使用例(2/2)

- <使用例> IMA-FRTBにおけるボラティリティ・サーフェス
  a. 市場慣行に対応したテナー毎(1D, 1W,…,1y, 2y, ….)の市場ボラティリティについて、ヒストリカル分析に基づき10営業日後のシナリオを12か月分(250パターン)生成する
  b. 生成された各シナリオにつきDNNキャリブレーションにてパラメータ算出
  c. 上記b.で求まったパラメータを初期値にDNN近似を実行し、ポートフォリオ内の各取引に必要なインプライド・ボラティリティを算出する (時間方向の補間については割愛)

- その他SABR以外の普遍近似定理適用例
  - 多変量正規分布の計算
  - 資源配分等ポートフォリオマネジメント
  - ＸVA計算
  - ヘッジ額計算

  等々

結論

- 数値結果は悪くはないし、貧弱なハードウェアのわりにパフォーマンスもそれほど悪くはない。
- DNNとGPUを使うことで、これまで充分な計算が行えなかった金融計算が正確に計算できるようになる可能性大
- 今回の手法の場合、自ら計算し入力にした値を元に回帰した結果を使用しているので、所謂AIのブラックボックス問題には該当せず
- また、理論的考察では、
  - ATMから遠い行使価格、大きなρ、大きなν　等で比較的大きめの残差も散見されたが、行使価格の値の範囲拡大、モンテカルロの試行回数を増やしたり、時間離散化の数を増やすことが有効。
  - 上記問題は最新のハードウェアを使用すると、容易に改善可能。
  - さらに、トレーニングデータ量を増やすことで、DNNの推計精度の向上が期待できる。

終わりに

- 最後までお読み頂きまして、ありがとうございます。
- GPUとDNNの可能性を感じて頂けましたでしょうか？ 当社は今後の貴社のこれらツール導入検討のお手伝いをさせて頂きたいと考えております。
- もう少し詳しい内容を知りたい等、質問がございましたら下記メールアドレスへご連絡ください。
- 実際にボラティリティを計算する所をオンラインミーティングでお見せすることも可能です。
- また、計算結果を試したい市場ボラティリティ・データと行使価格をご提供頂きましたら、その場で直ちにSABR補間補外による計算結果をお返しします。

宜しくお願いします。

熊谷雅年

kumagai.masatoshi@eigenview.co.jp

株式会社アイゲンヴュー

参考文献

- Alan Hicks. 通貨オプション入門. *Foreign exchange options: An international guide to currency options, trading and practice, Japanese edition*. Kinzai. 2018
- Iain J. Clark. *Foreign Exchange Option Pricing*. John Wiley & Sons Ltd. 2011
- Ignacio Ruiz , Mariano Zeron. *Machine Learning for Risk Calculations: A Practitioner's View (The Wiley Finance Series)*. John Wiley & Sons Ltd.2021
- Quantifi. *Intel and Quantifi Accelerate Derivative Valuations by 700x Using AI on Intel Processors*. https://www.intel.com › central-libraries › documents. 2021
- William A McGhee. *An Artificial Neural Network Representation of the SABR Stochastic Volatility Model*. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3288882. 2018
- 篠崎 裕司　深層学習によるファイナンスの新展開 ディープ・ヘッジングの紹介 https://www.imes.boj.or.jp/jp/conference/finance/2022_slides/1111finws_slide3.pdf　2022