Introduction

本篇論文運用類神經網路（neural network）來做歐式選擇權（EU call option）定價，並透過模型設計來保證此模型符合一些經濟直覺。

選擇權定價已有一定的歷史，最早且簡單的Black-Scholes (Black and Scholes 1973)模型提供了歐式選擇權的定價方法，但這個經濟模型背後有許多假設，像是假設隱含波動率不變，與市場的波動度微笑曲線（volatility smile, 因為展望理論導致）不合，相對於透過經濟假設導出的公式，機器學習模型則是透過「大量資料」來學習做選擇權定價 (Malliaris and Salchenberger 1993)，且只要資料夠多，機器學習模型就可具有一般化的能力，某種程度上優於經濟學公式，可以透過增加經濟學上的數學限制來讓基於機器學習上的選擇權定價模型表現的合理 (Garcia, Ghysels, and Renault 2010)。

但機器學習模型在選擇權定價上有個缺點是，它往往想找「唯一」的定價方法來去涵蓋所有的選擇權，導致它會高估價外選擇權 (Ben- nell and Sutcliffe 2004)、低估快到期的的選擇權 (Dugas et al. 2000) ，為了解決這件事情，Gradojevic, Gencay, and Kukolj 於 2009 年提出了分治法（divide-and-conquer）來將每個選擇權分類到不同的子類別（time to maturity & moneyness），並為這些子類別來設置不同的定價方法。

但到底該如何分類呢？用一固定的理論來界定要不要分類（this categorisation is done by manually defined heuristics）可能會沒辦法考慮到市場會隨時變化的狀況，因此本篇論文提出了一個類神經網路的選擇權定價模型，它會根據從資料中學習到的資訊來「動態調整」要將選擇權分類到什麼樣的類別。

機器學習模型還有個缺點是，它往往不符合經濟直覺，導致在這樣的情況下有可能發生套利，沒辦法適用於一般的情況，所以有人就轉往整合了經濟上的限制至機器學習模型裡面的方向，也就是提供domain-specific inductive bias（有點類似於the prior assumption about the learner）來增加可以應用到的地方。

本研究則提出了具有經濟直覺的Gated neural network，像是合理的風險中立函數來避免套利。

The estimate of the option pricing model

在風險中立的情況下，在時間點T期望可能會獲得的選擇權收益

$$\hat{c}(K, S_t, \tau) = e^{-r\tau} \int_0^\infty max(0, S_T - K) f(S_T|S_t, \tau) dS_T$$

r is the risk free rate

K is the strike price

S_t is the underlying asset price

S_T is the underlying asset price at time T

τ is the time to maturity, $T - t$

e^{-rt} is the discount term

$f(x|S_t, \tau)$ is the conditional risk neutral prbability density function (probability density of the revenue)

因為無風險利率跟折現項都是獨立獲得的，所以真正的選擇權定價模型其實就是積分項

$$\tilde{c}(K,S_t,\tau)=\int_0^{\infty} max(0,S_T-K)f(S_T|S_t,\tau)dS_T$$

C1 : $P(S_T\le K)\le 1$

$$\frac{\partial \tilde{c}}{\partial K}\le 0, \frac{\partial \tilde{c}}{\partial K}=\int_0^{K} f(S_T|S_t,\tau)dS_T-1 \text{ (C1)}$$

$$\begin{aligned}
&\text{if } S_T-K>0:\\
\tilde{c}(K,S_t,\tau)&=\int_K^{\infty} S_T f(S_T|S_t,\tau)dS_T-K\int_K^{\infty} f(S_T|S_t,\tau)dS_T\\
\frac{\partial \tilde{c}}{\partial K}&=[-Kf(K|S_t,\tau)-\int_0^{\infty} f(S_T|S_t,\tau)dS_T]-K(0-f(K|S_t,\tau))\\
&=-\int_K^{\infty} f(S_T|S_t,\tau)dS_T\\
&=-(1-\int_0^{K} f(S_T|S_t,\tau)dS_T)\\
&=\int_0^{K} f(S_T|S_t,\tau)dS_T-1
\end{aligned}$$

C2 : 機率≥0

$$\frac{\partial^2 \tilde{c}}{\partial^2 K}\ge 0, \frac{\partial^2 \tilde{c}}{\partial^2 K}=f(S_T|S_t,\tau) \text{ (C2)}$$

$$\begin{aligned}
\because \text{C1}: \frac{\partial \tilde{c}}{K}&=\int_0^{K} f(S_T|S_t,\tau)dS_T-1\\
\therefore \frac{\partial^2 \tilde{c}}{\partial^2 K}&=\frac{\partial}{K}[\int_0^{K} f(S_T|S_t,\tau)dS_T-1]=f(S_T|S_t,\tau)
\end{aligned}$$

C3 : 距離到期日越久（$\tau=T-t$越大），S_T>K的機率越大

$$\frac{\partial \tilde{c}}{\partial \tau}\ge 0 \text{ (C3)}$$

C4 : 若履約價 K $\to\infty$，則買權選擇權價格$\tilde{c}\to 0$，因為標的物價格永遠小於履約價$S_T<K$

$$lim_{K\to\infty}\tilde{c}(K,S_t,\tau)=\tilde{c}(\infty,S_t,\tau)=0 \text{ (C4)}$$

C5 : 當到期日是現在（$\tau=0$），買權選擇權價格可以馬上被執行且$S_t=S_T$，因此買權選擇權價格為$max(0,S_t-K)$

$$\tilde{c}(K,S_t,0)=max(0,S_t-K) \text{ when } \tau=0 \text{ (C5)}$$

C6 : 上界是因為買權選擇權價格 $\tilde{c}$ 不會超過標的物價格S_t，相當於履約價$K=0$；下界是因為買權選擇權具有時間價值

$$max(0,S_t-K)\le \tilde{c}(K,S_t,\tau)\le S_t \text{ (C6)}$$

> for upper bound :
>
> $$\text{when } K = 0$$
> $$\tilde{c}(0, S_t, \tau) = \int_0^{\infty} S_T f(S_T|S_t, \tau) dS_T = e^{r\tau} S_t$$
> $$\rightarrow \hat{c}(0, S_t, \tau) = e^{-r\tau} \tilde{c}(0, S_t, \tau) = S_t$$
>
> for lower bound :
>
> $$\text{買權選擇權價格 c} = \text{intrinsic value} + \text{time value}$$
> $$= max(0, S_t - K) + \text{time value}$$

Assume the pricing model is **rescalable**

$$\tilde{c}(\frac{K}{S_t}, 1, \tau) := \frac{\tilde{c}(K, S_t, \tau)}{S_t}$$
$$\text{let } m = \frac{K}{S_t}$$

Single Model

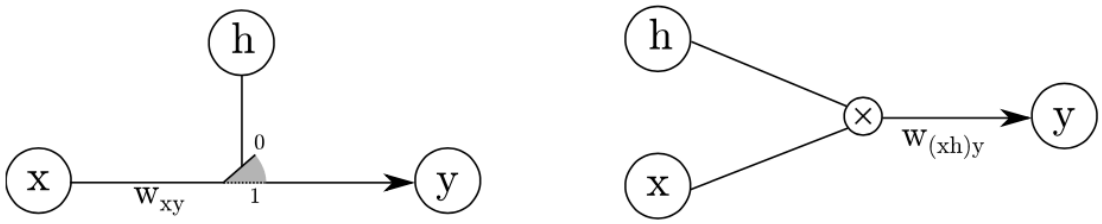


Figure 1: Two types of gating connections. On the left hand-side, the h neuron acts as a switch or gate that stops or not the flow of information between x and y. On the right hand-side, the connection implements a multiplicative relationships between the inputs x and h to provide the output y. Image reproduced from (Droniou, 2015).

Gated networks are extensions of the deep learning building blocks that are designed to learn relationships between at least two sources of input and at least one output with "gating connections". eg. LSTM family of networks uses a switch

The most general view is that neuron h (left-hand side is a switch, right-hand side is a multiplicative relation) modulates the signal between x and y. (cited from Gated networks an inventory)

所以使用gated neurl network就是為了控制moneyness跟到期日看誰可以通過、通過的機率要是多少（相乘起來就是在特定到期日與特定履約價下的機率）, 以此來自動分類選擇權

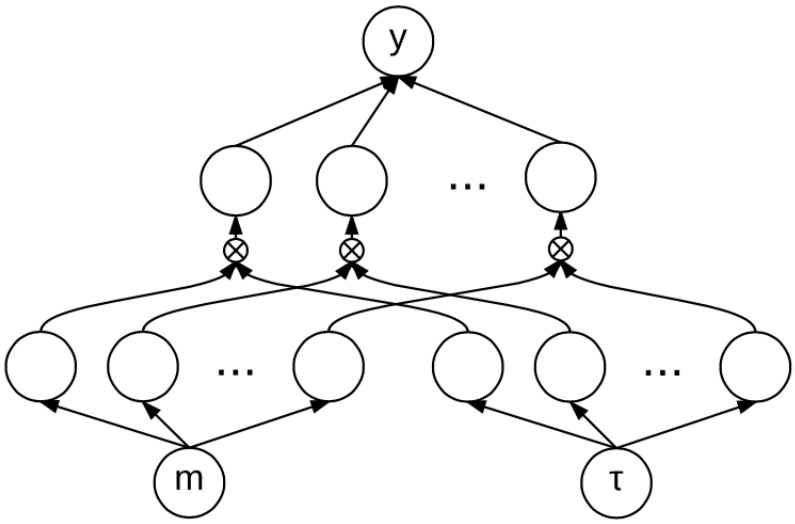


Figure 1: The proposed model (single). Note that bias terms exist, although they are omitted for neat appearance. $\otimes$ is the multiplication gate that outputs the product of the inputs.

$$y(m,\tau) \equiv \tilde{c}(m,1,\tau) := \frac{\tilde{c}(K,S_t,\tau)}{S_t}$$

$$y(m,\tau) = \sum_{j=1}^{J} \sigma_1(\tilde{b}_j - me^{\tilde{w}_j})\sigma_2(\bar{b}_j + \tau e^{\bar{w}_j})e^{\hat{w}_j}$$

$\sigma_1(x) = log(1+e^x)$ is a softplus function (between 0 and 1)

$\sigma_2(x) = \frac{1}{1+e^{-x}}$ is a sigmoid function (between 0 and 1)

Also,

$$\begin{aligned} \sigma_1'(x) &= \frac{e^x}{1+e^x} = \frac{1}{e^{-x}+1} = \sigma_2(x), \\ \rightarrow \sigma_1''(x) &= \sigma_2'(x) \\ \text{besides, } \sigma_2'(x) &= \frac{-1(-e^{-x})}{(1+e^{-x})^2} = \frac{1}{1+e^{-x}} * \frac{e^{-x}}{1+e^{-x}} = \sigma_2(x)(1-\sigma_2(x)) \end{aligned}$$

Hence, all $\sigma_1(x), \sigma_2(x), \sigma_1'(x)$, and $\sigma_2'(x) = \sigma_1''(x) \geq 0$

Rationality

C1($\frac{\partial \tilde{c}}{\partial K} \leq 0$), C2($\frac{\partial^2 \tilde{c}}{\partial^2 K} \geq 0$), C3($\frac{\partial \tilde{c}}{\partial \tau} \geq 0$) :

> 已知 $y(m,\tau) = \sum_{j=1}^{J} \sigma_1(\tilde{b}_j - me^{\tilde{w}_j})\sigma_2(\bar{b}_j + \tau e^{\bar{w}_j})e^{\hat{w}_j}$,
>
> $y(m,\tau) = \frac{\tilde{c}(K,S_t,\tau)}{S_t} \rightarrow \tilde{c}(K,S_t,\tau) = S_t y(m,\tau)$, $m = \frac{K}{S_t}$

$$\text{C1: } \frac{\partial y}{\partial m} = \sum_{j=1}^{J} -e^{\tilde{w}_j}\sigma_2(\tilde{b}_j - me^{\tilde{w}_j})\sigma_2(\bar{b}_j + \tau e^{\bar{w}_j})e^{\hat{w}_j} \leq 0$$

$$\rightarrow \frac{\partial \tilde{c}}{\partial K} = \frac{\partial S_t y}{\partial K} = S_t \frac{\partial y}{\partial m}\frac{\partial m}{\partial K} = S_t \frac{\partial y}{\partial m}\frac{1}{S_t} = \frac{\partial y}{\partial m} \leq 0$$

$$\text{C2: } \frac{\partial^2 y}{\partial^2 m} = \sum_{j=1}^{J} e^{2\tilde{w}_j}\sigma_2'(\tilde{b}_j - me^{\tilde{w}_j})\sigma_2(\bar{b}_j + \tau e^{\bar{w}_j})e^{\hat{w}_j} \geq 0$$

$$\rightarrow \frac{\partial^2 \tilde{c}}{\partial K^2} = \frac{\partial^2 y}{\partial m \partial K} = \frac{\partial^2 y}{\partial m^2}\frac{\partial m}{\partial K} = \frac{1}{S_t}\frac{\partial^2 y}{\partial m^2} \geq 0$$

$$\text{C3: } \frac{\partial y}{\partial \tau} = \sum_{j=1}^{J} e^{\bar{w}_j}\sigma_1(\tilde{b}_j - me^{\tilde{w}_j})\sigma_2'(\bar{b}_j + \tau e^{\bar{w}_j})e^{\hat{w}_j} \geq 0$$

$$\rightarrow \frac{\partial \tilde{c}}{\partial \tau} = \frac{\partial S_t y}{\partial \tau} = S_t \frac{\partial y}{\partial \tau} \geq 0$$

C4 : $lim_{K\rightarrow\infty}\tilde{c}(K, S_t, \tau) = 0$

> 已知 $m = \frac{K}{S_t}$, $\sigma_1(x) = log(1 + e^x)$, $y(m, \tau) = \sum_{j=1}^{J}\sigma_1(\tilde{b}_j - me^{\tilde{w}_j})\sigma_2(\bar{b}_j + \tau e^{\bar{w}_j})e^{\hat{w}_j}$, $\tilde{c}(K, S_t, \tau) = S_t y(m, \tau)$
>
> 當 $K \rightarrow \infty$
>
> $\Rightarrow m = \frac{K}{S_t} \rightarrow \infty$
>
> $\Rightarrow \sigma_1(\tilde{b}_j - me^{\tilde{w}j}) = log(1 + e^{\tilde{b}_j - me^{\tilde{w}j}}) = 0$
>
> $\Rightarrow y(m, \tau) = \sum_{j=1}^{J}\sigma_1(\tilde{b}_j - me^{\tilde{w}_j})\sigma_2(\bar{b}_j + \tau e^{\bar{w}_j})e^{\hat{w}_j} = 0$
>
> $\Rightarrow \tilde{c} = S_t y(m, \tau) = 0$
>
> This also explains **why there is no bias term for the top layer.**

C5 & C6 因為難以直接透過single model的設計來證明，所以改以透過產生符合條件的合成買權選擇權合約（不存在於現實的選擇權市場裡）來滿足這兩個限制式，並假設真實的選擇權價格c 就是理論上的選擇權價格 $\hat{c} = e^{-r\tau}\int_0^\infty max(0, S_T - K)f(S_T|S_t, \tau)dS_T$，而model $y(m, \tau)$的職責就是要去學出合適的weight w和bias b來逼近選擇權價格

C5 : $\text{when } \tau = 0, \tilde{c}(K, S_t, 0) = max(0, S_t - K)$

> **針對不同的現貨價格S_t，固定到期日$\tau = 0$ 並均勻取樣(uniformly sample)在$[0, S_t]$範圍內的履約價格K**，這樣的話買權選擇權價格$\tilde{c}$就會是$S_t - K$

C6 : $max(0, S_t - K) \leq \tilde{c}(K, S_t, \tau) \leq S_t$

> 思考方向：因為是看上下界，所以直接看極端值的地方
>
> for upper bound：$\tilde{c}(K, S_t, \tau) \leq S_t$
>
> > 極端值發生在最貴的買權選擇權合約，也就是履約價$K = 0$的買權選擇權合約（因為最容易有價內價格），因此**針對不同的到期日τ（相較於C5，這次C6產生的τ不限於0），取出履約價$K = 0$的買權選擇權合約**，這樣的話買權選擇權價格$\tilde{c}$就會是S_t

> 至此合成買權選擇權合約已產生完畢，產生依據來自C5跟C6 upper bound
>
> for lower bound：$max(0, S_t - K) \leq \tilde{c}(K, S_t, \tau)$
>
> > (i) when $K \geq S_t$: 極端值發生在最便宜的買權選擇權，也就是履約價$K \to \infty$的買權選擇權合約（因為最容易有價外價格），此時買權選擇權價格$\tilde{c}$便為0（亦即當$K \geq S_t$，lower bound會是0），參考C5跟C6 upper bound產生的買權選擇權合約，確實可以看到當履約價K越大，買權選擇權價格$\tilde{c}$越小
> >
> > (ii) when $K \leq S_t$：參考C5產生的買權選擇權合約，買權選擇權合約的價格$\tilde{c}$本身就是$S_t - K$了，所以透過C5跟C6 upper bound產生的買權選擇權合約不容易違反此限制式
>
> **直覺的想法可以想成 : psudo data其實就是在講邊界條件**

Multi-Model

市場上有許多不同類型的option，這些不同類型的option可能就會有不同的 m 與 τ ，若是只訓練single model，那就只會專注訓練到某個特定類型的option，在這個特性類型上的option導致了over-fitting，所以才會透過multi-model來增加參數把模型變大，盡可能的多樣容納並平衡(softmax)不同種類的option。

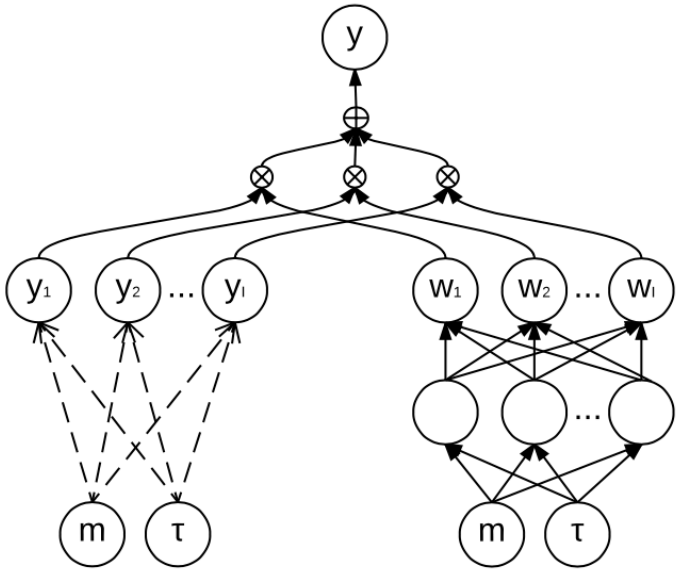


Figure 2: The proposed model (multi): The right side is the weight generating model, and the left side is a set of single models. Note that the left side is not a single layer. Each $(m, \tau) \to y_i$ (linked by two dashed arrows) is realised by a full-sized single model. $\oplus$ is the addition gate that outputs the sum of the inputs.

$$y_i(m,\tau) = \sum_{j=1}^{J} \sigma_1(\tilde{b}_j^{(i)} - me^{\tilde{w}_j^{(i)}})\sigma_2(\bar{b}_j^{(i)} + \tau e^{\bar{w}_j^{(i)}})e^{\hat{w}_j^{(i)}}$$

$$w_i(m,\tau) = \frac{e^{\sum_{k=1}^{K}\sigma_2(m\dot{W}_{1,k}+\tau\dot{W}_{2,k}+\dot{b}_k)\ddot{W}_{k,i}+\ddot{b}_i}}{\sum_{i=1}^{I} e^{\sum_{k=1}^{K}\sigma_2(m\dot{W}_{1,k}+\tau\dot{W}_{2,k}+\dot{b}_k)\ddot{W}_{k,i}+\ddot{b}_i}}, \text{ softmax function}$$

$$\Rightarrow y(m,\tau) = \sum_{i=1}^{I} y_i(m,\tau)w_i(m,\tau),$$

$$\text{softmax weighter average of the I local option pricing models'outputs}$$

remark : sum of the weights w_i is one

Rationality

Multi-model 仍然符合條件 C1($\frac{\partial \tilde{c}}{\partial K} \leq 0$)、C3($\frac{\partial \tilde{c}}{\partial \tau} \geq 0$) 、C4($lim_{K\to\infty}\tilde{c}(K, S_t, \tau) = 0$)

而條件C5(when $\tau = 0, \tilde{c}(K, S_t, 0) = max(0, S_t - K)$)、C6($max(0, S_t - K) \leq \tilde{c}(K, S_t, \tau) \leq S_t$)也一樣透過餵進合成選擇權合約成training data來滿足條件

但 C2 （$\frac{\partial^2 \tilde{c}}{\partial^2 K} \geq 0$） 這個條件卻沒有被滿足，為了解決這個問題，這篇論文為Multi-model引進了extra loss function

Denoting the first-order derivative of $y(m, \tau)$ w.r.t. m as $g(m, \tau) = \frac{\partial y}{\partial m}$

$$\sum_{p=1}^{P}\sum_{q=1}^{Q} max(0, g(m_{p,q}, \tau_q) - g(m_{p,q} + \Delta, \tau_q))$$

Δ is an small number

P is the number of pseudo data generated for every unique time- to-maturity

Q is the number of unique time-to-maturity τ in the training set

the equation will push $g(m, \tau)$ to a monotonically increasing function w.r.t. m

thus $\frac{\partial g}{\partial m} (\text{equivalently } \frac{\partial^2 y}{\partial m^2}) \geq 0$

$\Rightarrow \frac{\partial^2 \tilde{c}}{\partial K^2} = \frac{1}{S_t}\frac{\partial^2 y}{\partial m^2} \geq 0$

Comparison

Loss function : MSE(Mean Square Error), MAPE(Mean Absolute Percentage Error)

> the difference between $(c, e^{-r\tau}S_t y = \hat{c})$
>
> for numerical stability, equivalently the difference between $(e^{r\tau}\frac{c}{S_t}, y)$

Data : the option data for S&P500 index from Option Metric & Bloomberg

> closing price = mid-point of bid-ask price

Period : 1996/04/01 - 2016/05/31

Compared target : **PSSF**(Dugas et al. 2000), **Modular Neural Networks (MNN)** (Gradoje- vic, Gencay, and Kukolj 2009), **Black–Scholes (BS)** (Black and Scholes 1973), **Variance Gamma (VG)** (Madan, Carr, and Chang 1998), and **Kou Jump** (Kou 2002)

Preprocessing :

> 1. 捨棄價內選擇權，因為他們的交易非常不活躍以致難有參考價值
> 2. 捨棄到期日只剩下兩天的選擇權
> 3. 透過平價理論（put-call parity）將賣權選擇權轉換至買權選擇權來彌補第一點所失去的資料
> 4. 標準化（normalize）到期日τ至以年為單位 eg. $\tau = 7$, actual input$= \frac{7}{365}$

Training & Testing :

> 1. five continuous trading days data for training, and the following one day for testing
> 2. BS, VG, and Kou Jump only use the last training day's data (∵ Marcov process) to calibrate their parameters
> 3. For Single and PSSF, the number of hidden layer neurons is J = 5
> 4. The number of pricing models in both Multi and MNN is I = 9
> 5. The number of neurons in hidden layer for the right-branch weighting network of Multi is K = 5

以下這張圖可以看出Multi-model的loss function通常都是最小的

三段灰色部分loss function突高分別是因為1998 網路泡沫、2008金融危機、2011歐債危機

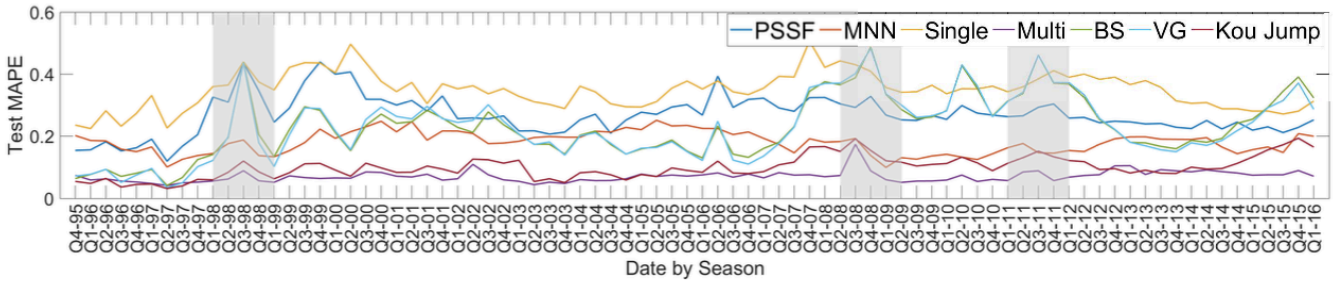
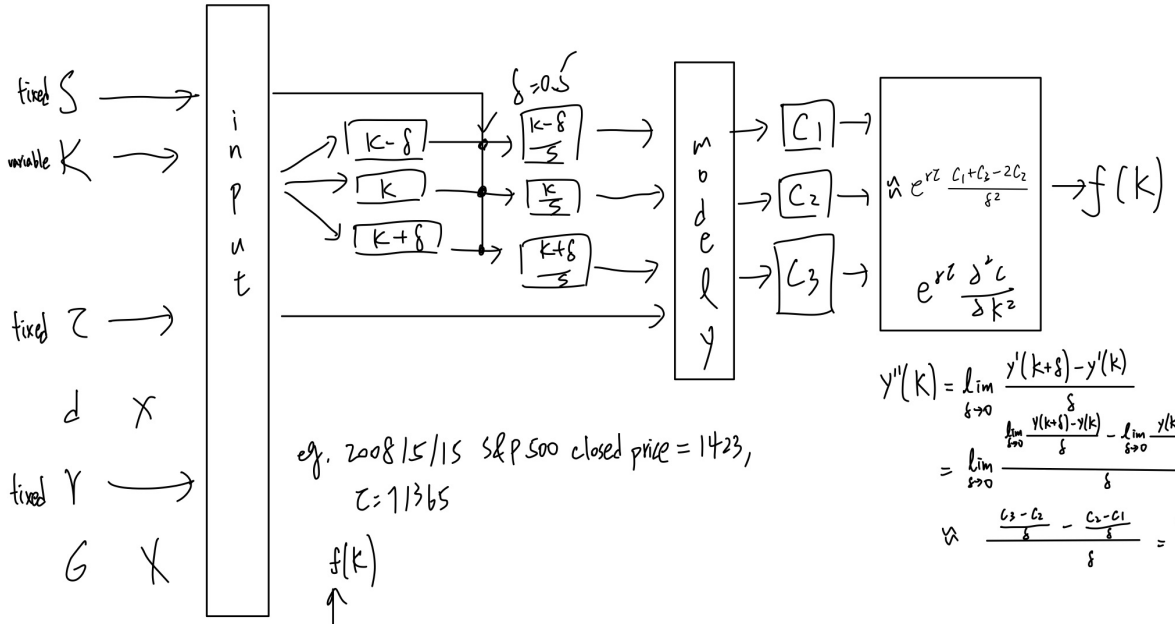


Figure 3: Test MAPE by Seasons: The shadowed parts correspond to the following events: Dot-com bubble (1998), global financial crisis (2008), and European debt crisis (2011).

	Train		Test	
	MSE	MAPE (%)	MSE	MAPE (%)
PSSF	267.48	25.77	269.56	26.25
MNN	50.08	16.89	63.16	18.22
Single	579.74	34.74	580.47	34.99
Multi	**9.91**	**5.75**	**12.11**	**6.84**
BS	63.73	21.64	64.71	22.42
VG	55.40	18.42	61.57	22.64
Kou Jump	18.37	8.69	20.13	9.90

Table 4: Quantitative comparison of pricing on 3M contracts.

以下這些例子取自testing day 2008/05/15，當天 S&P Index為1423，並繪製7天之後 (i.e., $\tau = 7$)的risk neutral density of the S&P Index，用來說明C1~C6的重要性

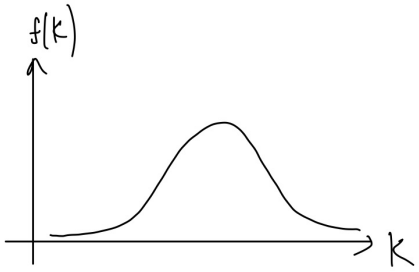


$$y''(K) = \lim_{\delta \to 0} \frac{y'(k+\delta) - y'(k)}{\delta}$$

$$= \lim_{\delta \to 0} \frac{\lim_{\delta \to 0} \frac{y(k+\delta)-y(k)}{\delta} - \lim_{\delta \to 0} \frac{y(k)-y(k-\delta)}{\delta}}{\delta}$$

$$\approx \frac{\frac{C_3 - C_2}{\delta} - \frac{C_2 - C_1}{\delta}}{\delta} = \frac{C_3 + C_1 - 2C_2}{\delta^2}$$

eg. 2008/5/15 S&P 500 closed price = 1423,
τ = 1/365



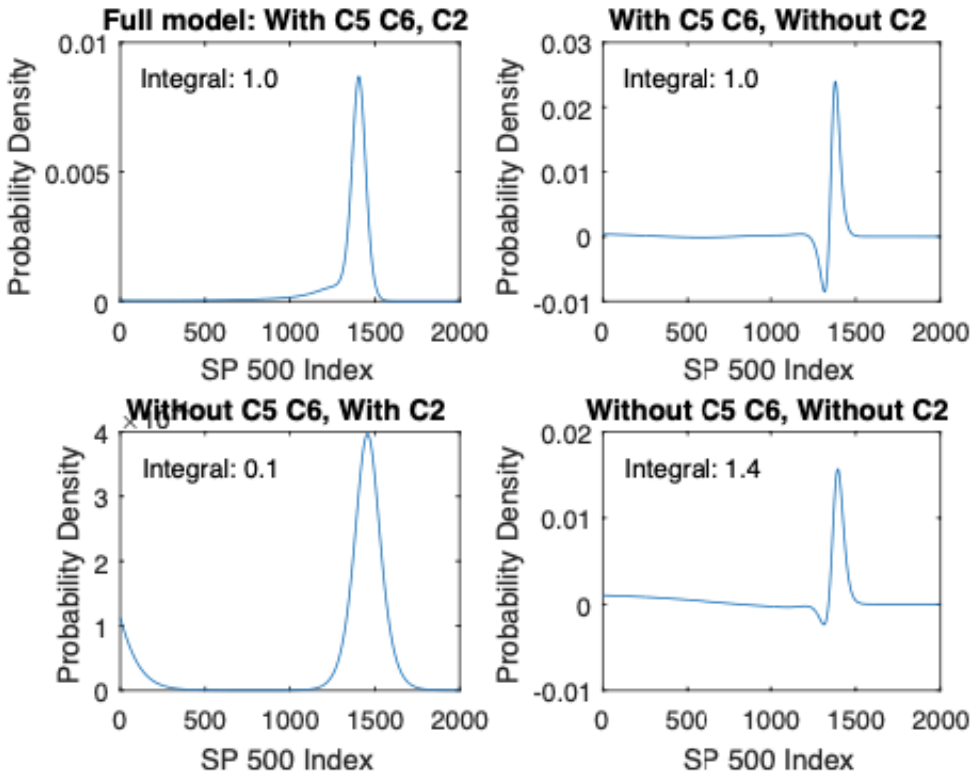


Figure 4: Implied distribution over future asset price. Top Left: Our multi model. Top Right: Without second derivative constraint (C2), we observe invalid negative values. Bottom Left: Without virtual options (conditions C5 and C6): we see density around zero which is senseless. Bottom Right: No derivative constraint or virtual options gives invalid and meaningless density.

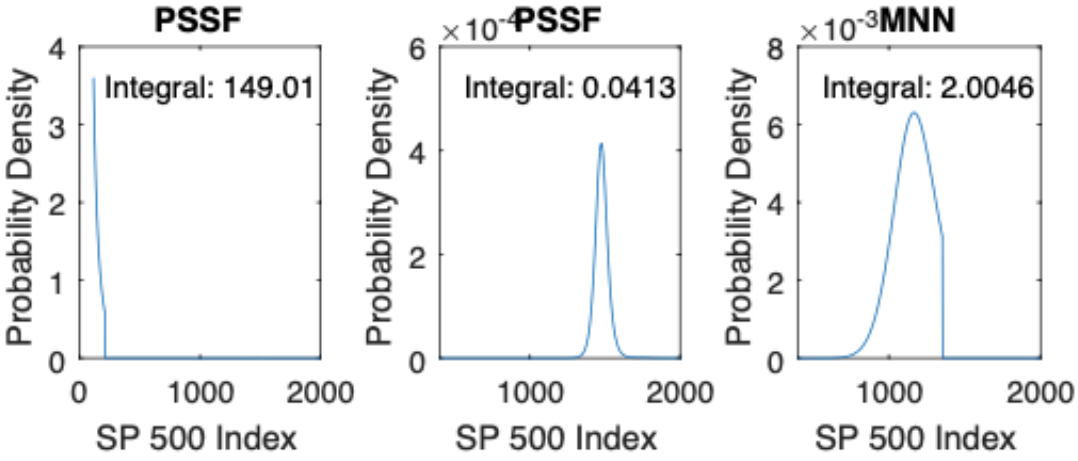


Figure 5: Neither PSSF nor MNN produces a valid distribution. Left: PSSF risk neural density for X-axis range $[0, 2000]$. Middle: PSSF risk neural density for X-axis range $[400, 2000]$ (note the difference on Y-axis scale). Right: Risk neural density of MNN.