國 立 高 雄 師 範 大 學
高雄師大學報 46, 2019. 6, 頁 47-68

Eikonal 方程式的快速數值解

黃耀新* 江仲驊**

摘 要

本文將提出一個描述相界面演遞的 Eikonal 方程式快速數值解法。在此方法中，配合模擬範圍內建置合宜的計算格點，並以界面到達計算點的時間作為計算變數。首先以界面位置預測界面到達計算點的時間，再根據此計算變數建構計算時間改變後所對應的界面位置，並以界面位置再修正界面到達計算點的時間，如此循環求解，直到界面抵達所有計算點。本文同時依據與界面的相對位置，將計算空間適當分類，使得在實際計算過程中，僅須針對界面附近的計算點，估算界面到達時間，從而得到相當快速、便捷的計算程序。經由若干案例計算，並與現有的計算方法比較驗證，說明此方法不僅可以非常快速地得到 Eikonal 方程式的數值解法，其模擬結果亦較現有的依點前進法或快速前進法精確。

關鍵詞：相界面問題、Eikonal 方程式、快速前進法、界面重構

投稿日期：2018/12/10；接受日期：2019/04/24

* 國立高雄海洋科技大學輪機工程系暨研究所教授
** 義守大學機械與自動化工程學系副教授(通訊作者)

A Fast Numerical Solution for Eikonal Equation

Yao-hsin Hwang* Chung-hua Chiang**

Abstract

In this paper, an efficient numerical method has been proposed to simulate internal ballistic of a solid motor with multiple propellants. The essential feature in this method can be depicted by transforming the original initial value problem for the solid propellant burning procedure into an associated boundary value problem, which can be described by the Eikonal equation. A fixed mesh covering simulation domain is established. The arrival time of the initial burning surface has been selected as the solution variable. At first, the arrival time is estimated by burn distance from the burning surface which is then reconstructed based on the arrival time as the simulation evolves. In this simple manner, all arrival time in the computational domain can be determined to serve as the essential element in the ballistic simulation. Meanwhile, special regions can be identified by the burning surface to enhance computational efficiency. By examining theoretical results from the physical model and computational results of test problems, the proposed numerical methodology in this research can be an effective tool for solid propellant motor design.

Keywords: Surface tracking, Fast marching methods, Eikonal equation, Surface reconstructions.

Submitted: 2018/12/10；Accepted: 2019/04/24

* Professor, Department of Marine Engineering, National Kaohsiung Marine University Kaohsiung , Taiwan
** Associate professor, Department of Mechanical and Automation Engineering, I-Shou University, Kaohsiung, Taiwan

壹、簡介

在相界面問題中，如果相界面沿著其本身的法線方向移動，且僅以單方向外擴或內縮，而不會出現來回移動現象，則其運動形式便可以 Eikonal 方程式描述[1]：

$$\left|\nabla T\right| F = 1 \tag{1}$$

其中對應的變數 T 即為相界面到達的時間，F 為界面移動速度，是空間位置的正函數，$F = F(\vec{r}) > 0$。使用 Eikonal 方程式最大的特點即是將原本必須以 Hamilton-Jacobi 方程式描述隨時間演遞的初始值(initial value problem)相界面移動問題，轉變為較為單純的邊界值問題(boundary value problem)，並得以在此基礎上發展較為有效、精確的數值解法。Eikonal 方程式所描述的相界面問題實際應用範圍非常廣泛，其中至少包括光線在不同介質間的傳播、電子元件製程中的刻蝕(etching)與沉積(deposition)過程、影像處理、固體燃料燃燒退化面形狀等[2、3]。

根據 Eikonal 方程式的特性所發展的數值模擬方法，基本上可分為兩大類，即界面追蹤法(front tracking method)與界面捕捉法(front capturing method)。其中界面追蹤法使用連結計算點標示界面位置，此即所謂的標誌粒子法(marker particle scheme)[4、5]，主要根據相界面的幾何變化決定標誌粒子的演變；另外，也可利用座標轉換原理，將 Eikonal 方程式置於射線座標系統(ray-coordinates system)下，求得相界前沿的分佈，如所謂的 Huygens 波前追蹤法(Huygens wavefront tracing method)[6]。此種方法的特點乃是以 Lagrangian 觀點直接界定相界面的確切位置，同時也不需要相界面本身以外的訊息便可模擬界面的演遞情形。至於在界面捕捉法方面，則是在模擬範圍內事先佈置若干計算點，並利用這些計算點上的訊息決定相界面位置。在此類方法中，較常被使用的有流體體積法(volume of fluid)[7]及水準組態法(level set method)[8]，其中流體體積法紀錄計算點上的不同相間的體積比率，而水準組態法則計算離相界面的符號距離函數(sign distance function)，兩者均可使用計算變數值而以內插方式重構相界面位置。使用 Eulerian 觀點界面捕捉法的最大優點，乃是可以避免在界面追蹤法中必須處理因相界面間相互交錯、合併所引致的幾何拓樸改變問題。另外，使用水準組態法除了可以界定相界面位置之外，其計算點訊息還提供相界面附近的幾何參數，亦可以相當簡便地得到諸如相界面法線方向及界面曲率等重要幾何量。然而，比較界面追蹤法僅需紀錄相界面位置，一般界面捕捉法必須考慮整個計算空間的變數值分佈情形，相對地會耗費較大量的儲存空間與計算量，因此，如何提高界面捕捉法的模擬效率並保持其計算上的便利性與精度，便為發展 Eikonal 方程式的計算模擬方法的主要重點。

在使用預設計算點模擬 Eikonal 方程式的數值方法中，Dijikstra[9]利用 Huygens 原理並配合 Eikonal 方程式的特點，將已知鄰近點視為新的傳播源而以空間前進方式得到邊界值問題的 Eikonal 方程式的數值解，從而得到相當快速的計算解。由於此方法的計算過程乃經由計算點逐步擴展，可視為依點前進法(point marching method)。雖然此方法可以得到相當快速的數值解，但其模擬精度卻並不令人滿意。Sethian[10]則以水準組態法為基礎，提出快速前進法(fast

marching method)，直接利用 Eikonal 方程式的差分表示式，使用上游差分(upwind difference)法則而以空間前進方式得到合理的模擬結果，另外再配合運用狹窄區間帶(narrow band)技巧[11]，更可進一步增加計算效率。如此不僅可以相當快速地完成模擬計算過程，同時也可避免水準組態法中必須設定的延伸速度(extended velocity)與計算變數(符號距離函數)再起始化(re-initialization)等問題。表面看來，雖然快速前進法直接利用Eikonal方程式的差分表示式求解，但實際上仍可將此處理方式置於 Huygens 原理架構下，得到其對應的物理基礎，此相關說明亦將於本文中討論。Hysing 與 Turek[12]比較多種 Eikonal 方程式的數值方法，除了快速前進法之外，亦比較包括快速掃描法(fast sweeping method)[13]、代數 Newton 法(algebraic Newton method)[14]等方法，經由計算案例結果的比較分析，發現快速前進法無論在模擬精度與計算效率上，均可作為 Eikonal 方程式的分析工具。另外由單點光源出發例如運用與幾何光學的傳遞波到達時間的計算，Treister 與 Haber[15]亦使用改進的快速前進法來解 factored Eikonal 方程式。

本研究的初衷主要是發展一個可以模擬具不均勻燃燒速率固體火箭燃料的燃燒退化界面，作為火箭內彈道分析的基礎，並希望未來能與流場分析程式配合，直接模擬固體燃料退化面與燃燒室流場的交互作用。職是之故，選用事先設置格點的 Eulerian 方式，以使退化後的格點配置亦可直接使用於流場分析。本研究便根據 Eikonal 方程式，以相界面到達的時間作為計算變數，但與快速前進法不同的是，本研究所提出的計算方法並不直接針對 Eikonal 方程式求其對應離散方程式的數值解，而是採取與依點前進法相同的觀點，使用 Huygens 原理將相界面作為新的傳播發射源。但是，依點前進法將每個計算點視為新的可能傳播源，並不考慮相界面實際位置的效應，因此無法得到精準的模擬結果。有鑑於此，本文乃根據已給定的相界面到達時間重新建構在特定計算時間時所對應的相界面位置，並據此更新計算點上的相界面到達時間。也就是說，不再僅以計算點為新的傳播源，而是以計算相界面為新的傳播源，從而提高模擬精度。除了本節的簡介之外，本篇論文的架構依序為計算方法、案例驗證及結論。在計算方法中，將說明本文所提出計算方法的物理模式及相對應的數值方法；在案例驗證中，藉由若干實際案例的計算結果，並與其他現有的計算方法比較，同時探討某些重要數值參數的效應。

貳、計算方法

相較於依點前進法或快速前進法，本研究所使用的 Eikonal 方程式計算方法具有兩個特點：

1. 根據計算點上的界面到達時間，估算界面位置。
2. 由界面位置計算計算點上的界面到達時間。

也就是說，當計算時間為計算點上的界面到達時間時，界面應該也將移動至計算點；而當計算時間大於計算點上的界面到達時間，表示界面已通過此計算點；同理，當計算時間小於計算點上的界面到達時間，表示界面尚未到達此計算點。這樣的處理方式，相當類似於水準組態法決定界面位置的方式，但由於界面的移動速度並非均質化分佈(homogeneous)，因此離界面位置的距離函數無法直接反應界面的可能移動，必須選用離界面位置的時間來更新界面的位置。根據這樣的理解，以下便詳細說明本研究所採用的物理模式及其對應的數值方法。

一、物理模式

選擇在一般化的基礎上，討論數值模擬方法的特性。當給定起始界面位置 $\vec{r}_o$ 及其傳播速率 F 時，在模擬範圍內界面到達某一計算點位置 $\vec{r}$ 的時間便可表示為：

$$T(\vec{r}) = \min_{\vec{r}_o,s} \int_0^{s(\vec{r},\vec{r}_o)} \frac{dr}{F} \tag{2a}$$

其中 $s(\vec{r},\vec{r}_o)$ 表示由起始界面位置至計算點的路徑。換言之，界面到達計算點的時間即為所有起始界面位置經所有可能路徑至計算點時間中的最小值，此亦為所謂的 Fermat 最小時間原理[16]。然而，在實際計算中，即使給定界面移動速率分佈 $F(\vec{r})$ 的條件下，也無法逐一檢驗所有可能路徑並測試其達到計算點所需的時間。因此，必須尋求此方程式的等價表示式。根據 Huygens 原理，在任何時間所對應的相界面點，均可以視為新的傳播發射源，而得到相界面位置的演變。因此，可將上式改寫為：

$$T(\vec{r}) = \min_{\vec{r}_c}(T(\vec{r}_c) + \min_s \int_0^{s(\vec{r},\vec{r}_c)} \frac{dr}{F}) \tag{2b}$$

其中，$\vec{r}_c$ 為計算時間 t_c 時所對應的界面位置：

$$T(\vec{r}_c) = t_c \tag{3a}$$

在方程式(2b)中，$s(\vec{r},\vec{r}_c)$ 便為界面位置點 $\vec{r}_c$ 至計算點 $\vec{r}$ 的路徑。如果計算點與已知相界面點間的距離不是很大，便可以使用近似表示式來估算上式右邊的第二項，即為界面到達計算點的時間差距。這樣的處理方式，便是一般界面追蹤法中所採用的物理模式。也就是說，在計算模擬時間為 $t_c + \Delta t$ 時的界面位置便可由以下關係式求得：

$$\Delta t = \min_{\vec{r}_c,s} \int_0^{s(\vec{r},\vec{r}_c)} \frac{dr}{F} \tag{3b}$$

當計算時間間距 Δt 夠小時，便可以假設移動速率的分佈(最簡化的方式乃假設為定值，$F(\vec{r}) \approx F(\vec{r}_c)$)，從而估算滿足式(3b)的可能位置，即對應為模擬時間為 $t_c + \Delta t$ 時的界面估算位置。

至於以 Eulerian 描述觀點的界面捕捉法也是同樣以方程式(2b)作為數值模擬的基礎，所不同的是，界面捕捉法事先排置計算點位置，再利用此式決定界面到達計算點的時間。譬如依點前進法在已知界面到達時間點 $\vec{r}_c$，計算鄰近點的到達時間。當計算點相當靠近已知點時，其到達時間可估算為：

$$T(\vec{r}) \approx \min_{\vec{r}_c}(T(\vec{r}_c) + \frac{|\vec{r} - \vec{r}_c|}{F(\vec{r})}) \tag{4}$$

此計算方法亦假設移動速率的分佈為定值，且兩點間的可能路徑為直線。不過，需特別注意的是，此處已知計算點的界面到達時間並非對應於某一相界面，此現象並不符合 Huygens 原理的要求，也是導致計算誤差的主要原因。至於快速前進法則是以已知鄰近點的界面到達時間，使用 Eikonal 方程式的差分表示式估算計算點的界面到達時間。這兩種方法均不需設定計算過程中，相界面的可能位置，但必須選擇適當已知界面到達時間的鄰近點，方可得到合理的計算結果。

本研究所發展的模擬方法，也同樣建立在方程式(2b)的基礎上，同時也以計算事先設置計算點上的界面到達時間，但在估算兩點間的界面到達時間差距時，亦建構計算時間所對應的界面位置，以符合 Huygens 原理的要求。也就是說，本文所提出的計算方法，同時兼具界面追蹤法與界面捕捉法特點。因此，式(2b)可改寫為：

$$T(\vec{r}) = t_c + \min_{\vec{r}_c, s} \int_0^{s(\vec{r},\vec{r}_c)} \frac{dr}{F} \qquad (5)$$

此處 $\vec{r}_c$ 即界面到達時間為 t_c 時所對應的界面位置，滿足式(3a)的條件。然而，一般而言，界面位置 $\vec{r}_c$ 並非恰好為事先設置計算點位置，必須由已知計算點上界面到達時間來決定，因此出現計算上必須適當處理的因果關係：由界面位置估算計算點上界面到達時間，及由計算點上界面到達時間決定界面位置。此問題的處理方式將在下節計算方法中詳細說明。

針對本文目前處理的二維相界面移動問題，以及配合所使用的計算模擬方法，歸納以下相關物理模式的假設：

1. 以連結線段近似相界面位置。
   這樣的處理方式與界面追蹤法相同，其間的差異乃在於界面追蹤法直接計算連結線段位置，而本方法則以：
2. 相界面到達時間函數重新建構界面位置。
   當計算時間介於兩計算點的界面到達時間之間，便可以內插方式獲得其所對應的界面位置。至於相界面的移動則考慮：
3. 在數值計算間距內，相界面以沿其法線方向的直線方式移動。
   在實際計算中，並不直接移動界面，而是以不同計算時間所架構出的對應相界面位置，來決定界面的移動。至於相界面未到達區域內計算點的界面到達時間函數，則修正為：
4. 所有相界面線段移動至此點所需時間中的最小值。
   基於以上相界面移動方式的假設，方程式(5)可改寫為：

$$T(\vec{r}) = t_c + \min_{\vec{r}_c} \int_0^{|\vec{r}-\vec{r}_c|} \frac{dr}{F}$$

不過，即使採用此種簡化的物理模式，由於傳播速度並非定值，在數值上仍很難決定計算點上的界面到達時間，因此，當計算點與界面點相當接近時，進一步假設：

5. 界面到達計算點的最小時間路徑即為其至計算點的最短路徑。

配合這樣的物理模式，本文所採用數值方法的基礎便為以下方程式：

$$T(\vec{r}) = t_c + \int_0^{\min_{\vec{r}_c}|\vec{r}-\vec{r}_c|} \frac{dr}{F} \qquad (6)$$

利用以上物理模式的假設，便可以很快速的決定界面位置與更新界面到達計算點的時間。也就是說，只要能夠合理地決定在計算時間 t_c 的界面位置，便可以估算界面到達其附近計算點的時間，反之亦然。

二、數值方法

如前所述，本文所提出計算方法的特點乃在於由已知計算點上的相界面到達時間建構界面位置及由已知相界面位置估算計算點上的界面到達時間。首先考慮如何在已知計算點上的相界面到達時間建構界面位置，為方便說明起見，考慮圖 1(a)的一維情形，圖中 r_A 與 r_B 分別為計算點 A 與 B 的位置座標，如果其對應的相界面到達時間分別為 T_A 與 T_B ($T_B > T_A$)，便可以建構在此計算空間內相界面到達時間的分佈函數，如圖中通過 T_A 與 T_B 的斜線。當計算時間 t_c 介於 T_A 與 T_B 之間($T_B > t_c > T_A$)，根據到達時間分佈函數以內差方式可以得到對應的位置座標，如圖中 r_c 所示。同理，計算時間增加間距 Δt 之後，亦可得到其對應的界面位置 $r_{c+\Delta t}$ ，從而顯示了界面的移動。相同的原則亦可應用於二維的情況，考慮如圖 1(b)所示，如果計算時間介於 T_A 與 T_B 以及 T_A 與 T_C 之間，可以依據前述內差原則，得到對應的位置座標 r_{c1} 與 r_{c2} ，並在計算空間內而以線段方式顯示的界面位置，即為圖中顯示的虛線。也就是說，當計算時間介於任意兩相臨計算點所對應的界面到達時間之間，即以線性內插方式得到界面點位置：

$$\vec{r}_c = \frac{t_c - T_A}{T_B - T_A}\vec{r}_A + \frac{T_B - t_c}{T_B - T_A}\vec{r}_B$$

由以上說明可知，只要能夠得到計算點上的相界面到達時間，便可很便捷地得對應計算時間的界面位置。利用建構界面的處理方式，基本上可以得到二階精度的界面位置表示式，其誤差量(估算值與實際位置的差距)為：

$$\Delta r = \frac{1}{2} F \frac{dF}{dr}(T_B - t_c)(t_c - T_A) + HOT \qquad (7)$$

其中 HOT 表示高階項(higher order term)。當然，依此內插原則也可以推導出更高階精度的界面點位置表示式，但必須牽涉到更多計算點，同時界面到達時間函數也可能會出現不符合實際物理現象的震盪分佈現象，可以造成界面位置不再唯一。

再來考慮在已知的相界面位置時，計算點上界面到達時間的表示方式。根據所採用物理模式而得到的操作方程式(6)，必須決定計算點至已知相界面的最短距離。由於相界面可用連結線段表示，在不失一般性的情況下，考慮圖 2(a)所示計算點 P_1 及 P_2 至相界面線段最短距離的對應

點必定為垂直投影點或是兩端點，也就是說，如果垂直投影點位於界面線段內，則投影點至計算點的路徑便為最短距離，如圖中的$P_1P_1^*$所示；另一方面，如果垂直投影點位於界面線段之外，則計算點至某一端點的路徑便為最短距離，如圖中的$P_2P_2^*$所示。如此，便可相當便捷地決定計算點至相界面的最短路徑，而其隱含的相界面移動方式便如圖 2(b)所示，這也是以 Lagrangian 觀點為基礎的界面追蹤法所採用的相界面移動方式。當決定式(6)的積分路徑，其值便可利用 Gauss 積分方式得到計算點上的界面到達時間[17]。然而很明顯地，當相界面移動之後，計算點上的新界面到達時間極可能與之前的估算值並不相符。因此，在實際計算中，如果相界面已通過計算點 $P(T_P<t_C)$)，才認定其為計算點上的界面到達時間；而當相界面未通過計算點 $P(T_P>t_C)$ 時，此時的界面到達時間均為暫時估算值，必須隨計算時間或相界面位置修正、更改。換言之，界面未通過的預估到達時間，主要是用來決定相界面位置，而這樣的處理可歸納為預測──修正(predictor-corrector)的形式。事實上，依點前進法與快速前進法亦採用當類似的處理方式[9、10]。

由以上計算方法的說明可知，無論要決定相界面位置或估算計算點上的界面到達時間，均不需要針對所有計算範圍，而僅需考慮相界面附近的計算點。因此，在節省計算機運算量及提高分析效益的考量下，可依據與界面相對位置的關係，將計算空間標示為界面已通過區(arrived cell)、界面到達區(arriving cell)、活躍區(activated cell)及閒置區(inactivated cell)四種，相關示意圖可見於圖 3。其中界面已通過區(標示值為−1)為所有構成頂點(計算點)的界面到達時間均小於目前計算時間，表示界面已通過此區域；在界面到達區，計算時間則介於構成頂點的界面到達時間之間(標示斜線區域)；活躍區(標示值為 1)與閒置區(標示值為 2)所有構成頂點的界面到達時間均大於目前計算時間，表示界面尚未到達此區，而此兩者的差異乃在於活躍區與到達區相臨(具有相同的構成計算點)，閒置區則表示界面亦尚未到達其相臨區域。根據這樣的分類，如果再進一步設定計算時間間距使得界面移動不會跨越一個完整的計算區域(活躍區)，則各計算區域隨計算時間的演變，將由閒置區依序轉變為活躍區、界面到達區及界面已通過區，而閒置區的構成計算點並不影響相界面的建構位置及其移動。基於這樣的觀察，便可將計算點分類為存活點(alive node)、活躍點(activated node)及閒置點(inactivated node)三種，其示意亦可見於圖 3。在此圖中，實心圓點即表示存活點；粗邊圓點為活躍點；而閒置點便以細邊圓點表示。由圖 3 可明白看出，存活點與活躍點分別為已通過區與活躍區的構成頂點，而閒置點則為存活點與活躍點之外的計算點。因此，存活點上界面到達時間小於計算時間，表示界面已通過此點，其界面到達時間將不再修正；至於活躍點或閒置點上的界面到達時間均大於計算時間，其值將可能隨界面的移動而更新。另外，配合計算時間間距的設定，使得閒置點的界面到達時間將不影響相界面的建構位置及其移動，在實際計算上不必考慮其變數值，進而可以很大程度地減少計算量及提高計算效率。

要避免相界面移動不會跨越一個完整計算區域的最大可容許時間間距，可由構成活躍區的計算點(活躍點)上的界面到達時間決定：

$$\Delta t_{max} = \min_{fl(ic)=1} \{\max_{k(ic)}[T(\vec{r}_k; ic)]\} - t_c \qquad (8a)$$

其中 fl(ic)表示編號 ic 區域的標記值，fl(ic)=1 即此區域為活躍區；k(ic)表示構成此區域計算點的編號。計算時間間距便可設定為：

$$\Delta t = Cr\Delta t_{max} \tag{8b}$$

此處 Cr 為 Courant 數。在本文案例計算中，採用 Cr=0.5。

由於存活點的界面到達時間不必再更新，且可不必考慮閒置點上的界面到達時間，因此僅須更新活躍點上的界面到達時間，使得界面演變的計算局限在界面附近，有效地降低數值計算量。另外，在本文所採用的計算方法中，為了進一步降低計算量且簡化計算過程，當界面移動後仍保持在相同的區域中，亦不考慮此計算區域內移動後的界面對各活躍點的影響，也就是說，只有當活躍區轉變為界面到達區時，才考慮其臨近活躍點上界面到達時間的更新。因此，在實際計算模擬中，僅須標記活躍區，而當計算時間增加使得活躍區轉變為界面到達區時，利用其構成計算點上的界面到達時間建構相界面位置，同時修正其附近非存活點的界面到達時間。最後，將此區域移出活躍區，並設定其臨近的閒置區為活躍區。至於計算區域的臨近點，即為構成其臨近區域的所有計算點。

綜合以上所採用的數值處理方式，在實際操作上便可以歸納為以下的演算步驟：

1. 在模擬計算範圍建構不疊錯也不留空的非內凹格點系統。
2. 標示所有計算區域及計算點分別為閒置區(標記值為 2)與閒置點(界面到達時間為無窮大)。
3. 根據起始界面位置，決定界面已通過區(標記值為−1)、界面到達區(標記值為 0)及活躍區(標記值為 1)，並計算各區對應存活點與活躍點上的界面到達時間。設定起始計算時間 $t_c = 0$ 。
4. 根據活躍點上的界面到達時間，決定計算時間間距 Δt ，並更新計算時間 $t_c = t_c + \Delta t$ 。
5. 由計算時間逐一檢視各活躍區是否轉變為界面到達區。如果沒有出現轉變區，回到步驟 4。
6. 由轉變區計算點上的界面到達時間，建構此區內相界面位置，並更新其非存活臨近點的界面到達時間。
7. 將轉變區臨近閒置區設定為活躍區，並將此轉變區移出活躍區而設定為界面到達區。
8. 回到步驟 4，直到已不存在活躍區。

相較於快速前進法[2、10]，本方法並不直接對 Eikonal 方程式求解，而是以代數方式決定界面的移動。

參、案例驗證

為驗證本文所提出計算方法的可行性並探討其數值特性，考慮若干二維計算案例，其中包括均勻與非均勻傳播速度及不同方式的起始界面。在不特別言明的情況下，採用均勻格點分佈，但亦考慮格點變動扭曲的影響，因此，計算格點的座標可表示為：

$$x_i = (i-1)\Delta x + \alpha(\gamma_{x,ij} - 0.5)\Delta x \tag{9a}$$

$$y_j = (j-1)\Delta y + \alpha(\gamma_{y,ij} - 0.5)\Delta y \tag{9b}$$

其中 i 與 j 分別為 x 與 y 方向的格點標示值，Δx 與 Δy 分別為 x 與 y 方向的格點間距，α 為格點設定變動扭曲量，$\gamma_{x,ij}$ 與 $\gamma_{y,ij}$ 為介於 0 與 1 的隨機亂數值(random number)。當 α=0 時，即表示使用均勻格點。

首先考慮單一源點在均勻速度場(F=1)所發展出來的相界面演遞問題。計算模擬範圍為 $-1 \le x, y \le 1$，計算參考格點為 40x40，但亦衡量改變格點密度的影響。相界面發展源點置於座標原點處(0, 0)。計算結果除了與正解比較外，亦與依點前進法及快速前進法所獲得的數值結果比較。圖 4(a)-(c)即為分別使用依點前進法、快速前進法及本文所提出的方法所獲得界面到達時間與誤差分佈情形，上圖所顯示的界面到達時間等值線，其間差值為 0.1；下圖的誤差乃為計算值與正解的差異量。由圖可以看出，依點前進法得到的相界面呈八邊形，顯示迥異於正解的圓形分佈；快速前進法雖然可以得到較合理的分佈形態，但其誤差值仍與依點前進法相當。另外，依點前進法在傳播速度方向與格點線垂直或呈 45°夾角時，均可得到正解值，而快速前進法在 45°夾角時則反而呈現較大的誤差量。至於本文所提出的計算方法，無論在相界面形態或誤差量上，均較此兩種方法更接近正解，其誤差量約比依點前進法或快速前進法低一個量階。圖 5(a)與(b)分別表示在不同切面上，界面到達時間與相對誤差的分佈情形。相對誤差定義為誤差量與正解的比值。此兩圖更進一步說明本文所提供的方法確實可以得到較準確的數值模擬結果。另外，圖 5(b)亦顯示依點前進法的相對誤差在 y=0.6 與 y=1.0 處幾乎具有相同的分佈，說明此方法雖然在計算上較為簡便，但可能無法以增加計算點的密度來提高其模擬精度。這樣的猜測可由圖 6 所顯示的在改變格點密度後，相對誤差在 y=1.0 處分佈可看出。無論使用快速前進法或本文提出的新方法，當格點密度增加後，均可以提高模擬精度，反觀依點前進法的精度幾乎與採用格點數目無關，表現出零階精度的數值特性。為了說明計算方法的精度階數，考慮三種不同的整體誤差度量：

$$\varepsilon_1 = \frac{1}{n_p}\sum_{k=1}^{n_p}\left|T(\vec{r}_k) - T_{ex}(\vec{r}_k)\right| \tag{10a}$$

$$\varepsilon_2 = \sqrt{\frac{1}{n_p}\sum_{k=1}^{n_p}[T(\vec{r}_k) - T_{ex}(\vec{r}_k)]^2} \qquad (10b)$$

$$\varepsilon_{max} = \max_{k=1,n_p}\left|T(\vec{r}_k) - T_{ex}(\vec{r}_k)\right| \qquad (10c)$$

其中，$T(\vec{r}_k)$ 與 $T_{ex}(\vec{r}_k)$ 分別表示在格點上的計算值與正解，n_p 為在模擬範圍內所有計算點數目。探討由於燃面折曲所造成燃畢時間的高估現象，如果使用正方形格點，可以進一步推導出最大可能誤差量。由於任何燃面可以視為無窮燃點所構成，因此在不失一般性的原則下，考慮位於原點(0, 0)處的燃面發展情形。在二維正方形網格中，其對應的計算點座標為：

$$x_i = (i-1)\Delta \;;\; y_j = (j-1)\Delta$$

其中 Δ 為網格尺寸。如此，計算點上的燃畢時間正解為：

$$T_{ex} = \frac{\Delta}{v_B}\sqrt{(i-1)^2 + (j-1)^2}$$

同時，定義計算收斂階數：

$$n_{conv} = \frac{\ell n(\varepsilon_{(1)} / \varepsilon_{(2)})}{\ell n(\Delta_{(1)} / \Delta_{(2)})} \qquad (11)$$

下標(1)與(2)表示不同格點密度的兩次計算。表 1 列出在不同格點密度下的誤差值及收斂階數。由此表可看出，在所考慮的計算格點密度中，本文所提出的計算方法無論以何種誤差度量，均遠較依點前進法與快速前進法為低，而依點前進法法在使用較少格點數目時可以得到比快速前進法較低的誤差量，但當格點數目增多後，其模擬精度則較快速前進法為差。事實上，如圖 6 所觀察，依點前進法不會因格點變密而提高模擬精度，顯現出零階收斂特性。另一方面，雖然快速前進法的收斂階數略高於本文提出的計算方法，但由於其誤差值遠大於本文提出的計算方法，即使在所考慮最細的格點中，本文提出的計算方法仍可以得到遠較快速前進法為準確的模擬結果。

為了評估數值模擬方法的計算效率，圖 7 顯示所採用格點數目與計算機時間關係，很明顯地，依點前進法所需的計算時間均小於快速前進法與本文提出的計算方法，而本文提出的計算方法在計算點數目小時需要比快速前進法較多的計算時間，然而在使用細密格點的情況下，則所需的計算量則較快速前進法節省。不過，必須值得一提的是，本計算是基於 Cartesian 正交座標系統上，使用差分方程式求解的快速前進法具有相當大的便利性。雖然依點前進法完成計算需要較少的計算量，但如前所述，其計算精度無法隨格點數目(或計算量)增加而提高。因此，在相同計算量的情況下，可能無法得到較精確的模擬結果，此分析說明可由圖 8 所顯示的計算

量與計算誤差比較圖得知。圖 8 明白表示，在相同的計算量的情況下，本文提出的計算方法可以得到遠較依點前進法與快速前進法更為精確的模擬結果。最後，考慮計算點座標隨機變動而造成的扭曲對計算結果的影響。圖 9(a)與圖 9(b)分別表示扭曲格點分佈在不同切面，使用本文提出的計算方法所得計算結果與相對誤差比較圖。由此圖可看出，即使使用相當的扭曲格點分佈(α=0.8)，計算結果亦與正交格點(α=0)相當吻合，其所對應的誤差改變量亦不顯著，清楚說明使用扭曲格點分佈並不會降低本文提出的計算方法的模擬精度，顯示本文所提出計算方法的強固性。

本研究所考慮的第二個計算案例為四源點界面在均勻速度場的演變問題，界面速度與計算範圍均與案例一相同，而四個起始源點的位置座標則為：(0, ±0.5)與(±0.5, 0)。此案例主要說明本文所提出的計算方法處理界面交錯的能力。圖 10(a)表明相界面在不同時間的分佈情形及其對應誤差；圖 10(b)則顯示不同計算方法在 y=0.25 及 y=0.75 處界面到達時間的分佈比較圖。由計算結果可知，即使出現多個相界面的交錯、合併現象，本文所提出的計算方法仍可以得到相當準確、合理的模擬結果，其精度遠較依點前進法與快速前進法為佳。

再來考慮單一源點在不同傳播速度場所發展出來的相界面演遞問題。計算範圍為一矩形：$-2 \le x \le 2$ ； $-1 \le y \le 1$，計算格點為 80×40 的正交網格，起始源點位於座標(0, −0.5)處，傳播速度為：

$$y<0，F=1；y\ge 0，F=1.5 \qquad (12)$$

此案例等同於光線通過不同介質的問題。使用本文所提出的計算方法所獲得的界面分佈情形及其對應誤差可見於圖 11(a)；而圖 11(b)則顯示在 y=−0.25 及 y=0.5 處界面到達時間的分佈圖，其中亦包括正解及使用依點前進法與快速前進法的計算結果。由模擬結果可知，具有不連續分佈的界面傳播速度，亦不會降低本文所提出的計算方法的精度，同樣地，其模擬結果亦較依點前進法與快速前進法為佳。

由以上案例的計算結果可知，本文所提出的計算方法可以得到較依點前進法與快速前進法精確的結果。其中與依點前進法的比較結果很容易理解，因為此法與文中所提出的方法均建立在 Huygens 原理的近似解上(見式(6))，然而依點前進法僅考慮若干鄰近計算點做為新的傳播源點，而非對應於計算時間的相界面點。參考圖 12，如果計算點在 x-與 y-方向的間距均為 Δ，而點 W、S 與 SW 均為已知界面到達時間，根據依點前進法的計算方式，點 P 的界面到達時間便為：

$$T_{P,point} = \min\left(T_W + \frac{\Delta}{F}, T_S + \frac{\Delta}{F}, T_{SW} + \frac{\sqrt{2}\Delta}{F}\right) \qquad (13)$$

相較於本文所提出的界面建構方式，此法顯然僅考慮有限的計算點，而且這些已知界面到達時間的計算點亦並非在同一時間的界面上，因此導致較大的計算誤差。至於快速前進法直接使用 Eikonal 方程式的差分式[2、10]：

$$(\frac{T_{P,fast}-T_W}{\Delta})^2+(\frac{T_{P,fast}-T_S}{\Delta})^2=\frac{1}{F^2} \qquad (14a)$$

或

$$T_{P,fast}=\frac{T_W+T_S}{2}+\frac{1}{2}\sqrt{\frac{2\Delta^2}{F^2}-(T_W-T_S)^2} \qquad (14b)$$

然而，此表示法也可同樣使用 Huygens 原理加以理解，而得到其對應的幾何意涵。在圖 12 中，如果點 W 與點 S 間的界面到達時間假設為線性分佈，則以任何一點作為發射源而至點 P 的界面到達時間便可整理為：

$$v(\xi)=(1-\xi)T_W+\xi T_S+\frac{\sqrt{2}\Delta}{F}\sqrt{\xi^2-\xi+\frac{1}{2}} \qquad (15a)$$

其中，$\xi=\frac{s}{\sqrt{2}\Delta}$，為無單位弧長座標量，而快速前進法的表達式(14b)，即為：

$$T_{P,fast}=\min_{0\le\xi\le1} v(\xi) \qquad (15b)$$

也就是說，快速前進法乃是以點 W 與點 S 間的線段為發射源所得到的界面到達時間。這樣的處理方式，相當接近本文所提出的方法，所不同的是，後者以計算時間所對應的建構界面為發射源，而快速前進法的近似發射源並非在同一界面位置上(點 W 與點 S 可能具有不同的界面到達時間，$T_W \neq T_S$)，因此無法得到與本文計算方法同樣精確的計算結果。

在驗證若干案例之後，可以明顯看出本文所提出計算方法具有運算快速、結果精確的優點。以下便再模擬兩個實際工程案例，以說明本方法的應用性。首先考慮半導體製程中的沉積(deposition)模擬計算，計算模擬範圍為 $-1\le x, y\le 1$，計算參考格點為 40×40，假設固定沉積速度(F=1)，起始堆積面的連結座標為(-1,0)-(-0.35,0)-(-0.35,-0.8)-(0.35,0.8)-(0.35,0)-(1,0)。圖 13 顯示此模擬過程堆積面的形狀，將此結果與文獻中使用快速前進法的結果比較[10]，明顯說明本文所提出計算方法的準確性。第二應用案例則為半導體製程中的刻蝕(lithograghic)模擬計算，計算模擬範圍為 $-1\le x, y\le 1$，計算參考格點為 50×50，由起始面 y=1 向下進行刻蝕，刻蝕速度場的分佈為：

$$F=2e^{-8x^2}\{\cos^2[6(y+1)]+0.01\} \qquad (16)$$

計算結果顯示於圖 14(a)與圖 14(b)。圖 14(a)顯示不同時間的刻蝕界面分佈，此分佈型態與文獻[10]中的計算結果相當接近；至於圖 14(b)則比較扭曲格點在 y=−0.5、0 及 0.5 處的界面到達時間，說明即使使用相當程度的扭曲格點分佈，其結果亦非常接近使用正交無扭曲格點的分佈。

肆、結論

本研究針對 Eikonal 方程式的物理特性，以相界面演變問題推導其快速數值解法。其中以相界面到達時間作為計算變數，求得在事先給定計算格點上的相界面到達時間分佈，根據此分佈便可進一步修正界面尚未到達處的預計到達時間，進而以循環更新的方式逐步地得到合理的數值解。為了增加計算效率，特別根據相界面的相對位置，將計算空間分類為界面已通過區、界面到達區、活躍區及閒置區，在實際計算中，僅需更新組成活躍區計算格點上的變數值，而不必計算在界面已通過區及閒置區內的記算點。由於活躍區的大小對應於現時相界面範圍，因此可以相當有效地提高計算速率。在案例驗證計算中，經由與解析正解、依點前進法及快速前進法的數值計算結果比較，明白地顯示本文所提出的計算方法在相同的計算時間量下，可以得到相當精確的模擬結果；同時，即使使用相當扭曲的計算格點，也不會明顯地降低計算模擬精度。

參考資料

[1] Lions, P. L., *Generalization Solution of Hamilton-Jacobi Equations*, Pitman, London, 1982.

[2] Sethian, J. A., *Level Set Methods and Fast Marching Methods*, Cambridge University Press, Cambridge, England, 1999.

[3] Sutton, G. P., *Rocket Propulsion Elements: An Introduction to the Engineering of Rockets*, 6th ed. New York, John Wiley & Sons, 1992.

[4] Qian, J., Tryggvason, G. and Law C. K., "A Front Tracking Method for the Motion of Premixed Flames," *Journal of Computational Physics*, Vol., p.52-69, 1998.

[5] Udaykumar, H. S., Mittal, R. and Shyy, W., "Computation of Solid-Liquid Phase Fronts in the Sharp Interface Limit on Fixed Grid," *Journal of Computational Physics*, Vol., p.201-225, 1981.

[6] Sava, P. and Fomel, S., 3-D Traveltime Computation Using Huygens Wavefront Tracing, Geophysics, Vol.66, p.883-889, 2001.

[7] Hirt C. W. and Nichols, "Volume of Fluid (VOF) Method for the Dynamics of Free Boundaries," *Journal of Computational Physics*, Vol., p.52-69, 1998.

[8] Osher, S. and Sethian, J. A., Fronts Propagating with Curvature-Dependent Speed: Algorithms based on Hamilton-Jacobi Formulation, *Journal of Computational Physics*, Vol.79, p.12-,1988.

[9] Dijkstran, E. W., Numerische Mathematik, Vol.1, p.269-271, 1959.

[10] Sethian, J. A., "A Fast Marching Level Set method for Monotonically Advancing Fronts," *Proc. Natl. Acad. Sci. USA*, Vol.93, p.1591-1595, 1996.

[11] Chopp, D. L., "Computing Minimal Surfaces via Level Set Curvature Flow," *Journal of Computational Physics*, Vol.106, p.77-, 1993.

[12] Hysing, S.-R. and Turek, S., "The Eikonal Equation: Numerical Efficiency vs. Algorithmic Complexity on Quadrilateral Grids, *Proceedings of ALGORITHMY*, p.22-31, 2005.

[13] Kao, C. Y., Osher, S. and Qian, J., "Lax-Friedrichs Sweeping Scheme for Static Hamilton-Jacobi

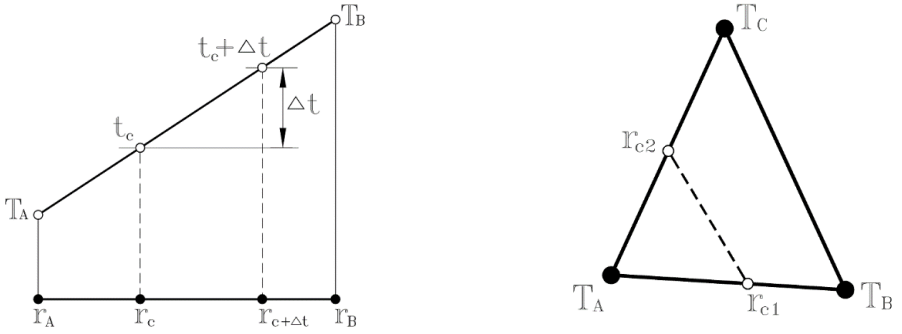
Equations,” *Journal of Computational Physics*, Vol.196, p.367-391, 2004.

[14] Persson, P.-O. and Strang, G., “A Simple Mesh Generator in Matlab,” *SIAM Review*, Vol.46, p.329-345, 2004.

[15] Treister, Eran; Haber, Eldad. A fast marching algorithm for the factored eikonal equation. *Journal of Computational Physics*, 2016, 324, p.210-225.

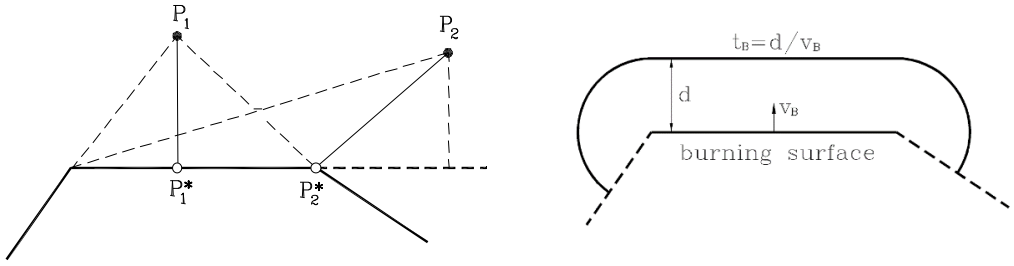
[16] Feynman, R. P., Leighton, R. B. and Sands, M., The Feynman Lectures on Physics, Vol. I, Ch. 26, New York, Addison-Wesley Publishing Co., 1989.

[17] Carnahan, B., Luther, H. A. And Wilkes, J. O., Applied Numerical Methods, New York, John Wiley & Sons Inc., 1969.



(a)一維情形 (b)二維情形

圖 1 計算點界面到達時間估算界面位置



(a)界面距離 (b)界面演遞

圖 2 相界面移動模式

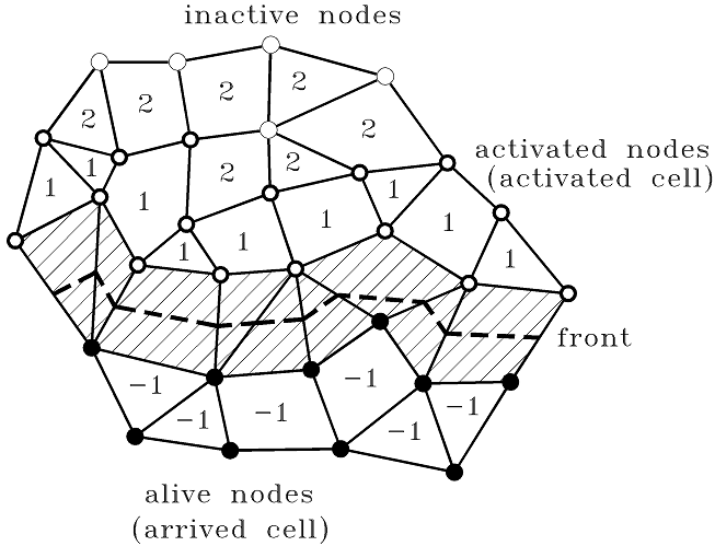
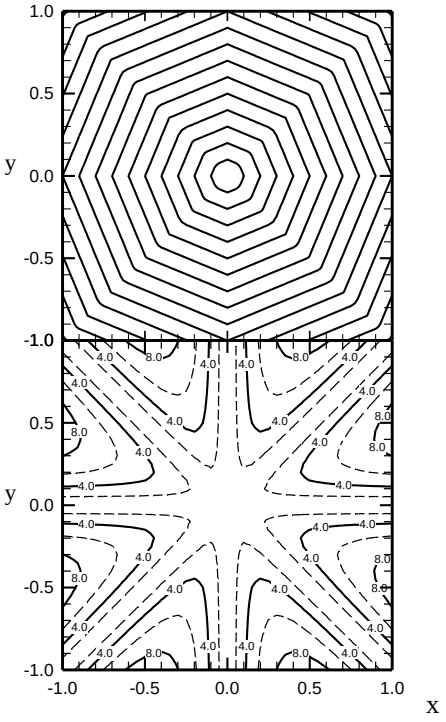
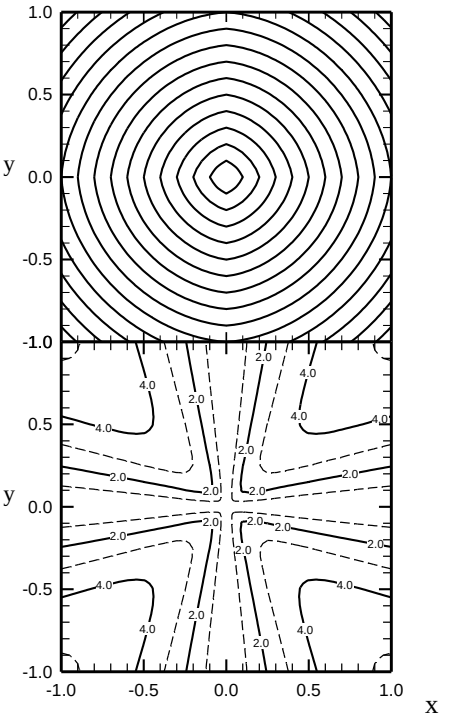


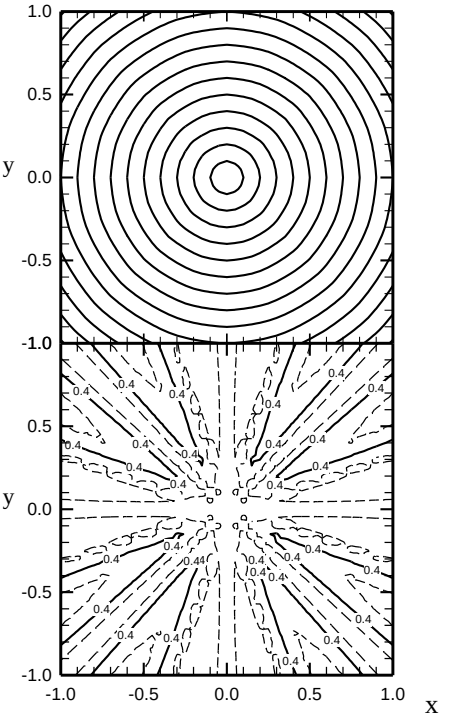
圖 3 界面區域及計算點分類示意圖



(a)依點前進法

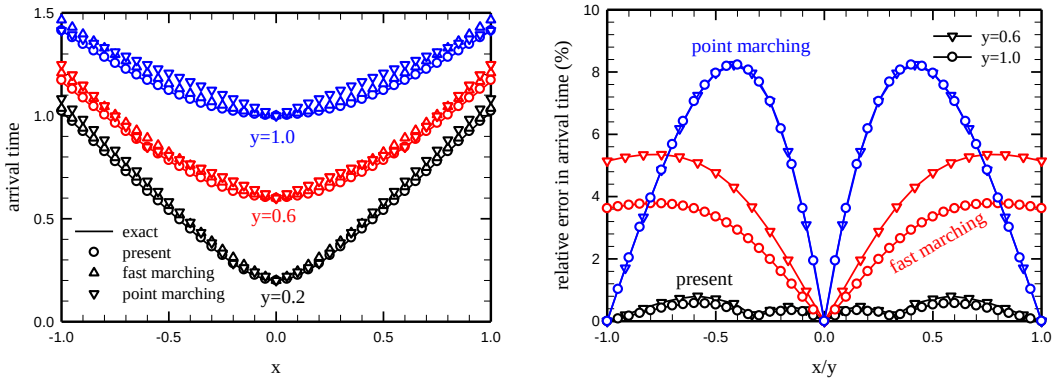


(b)快速前進法



(c)目前方法

圖 4　單一源點發展界面及其誤差



(a)到達時間 (b)相對誤差

圖 5 單一源點發展界面在不同切面比較結果

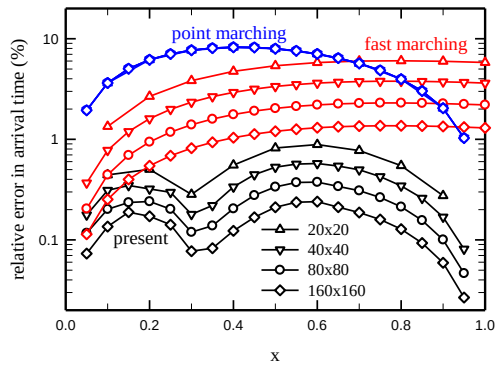


圖 6 單一源點發展界面在不同格點密度下的相對誤差在 y=1.0 處分佈圖

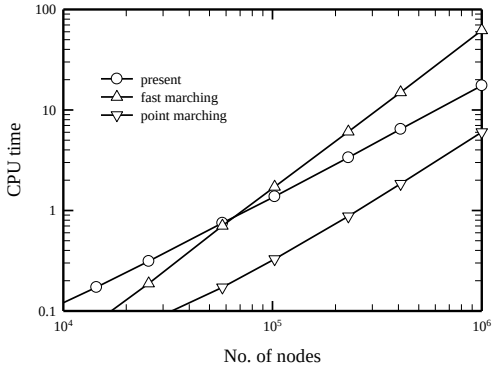


圖 7 格點數目與計算機時間關係

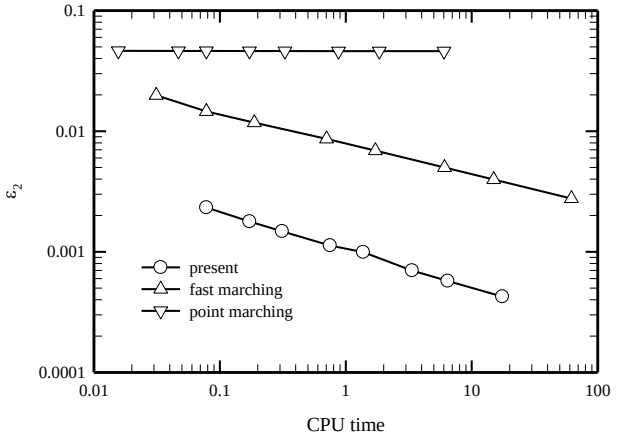
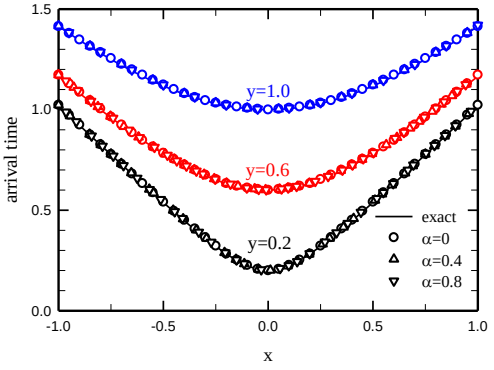
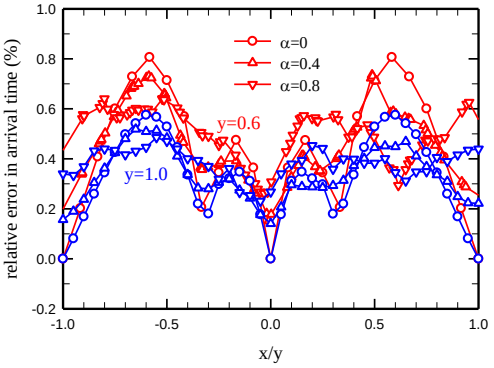


圖 8 計算機時間與計算誤差關係

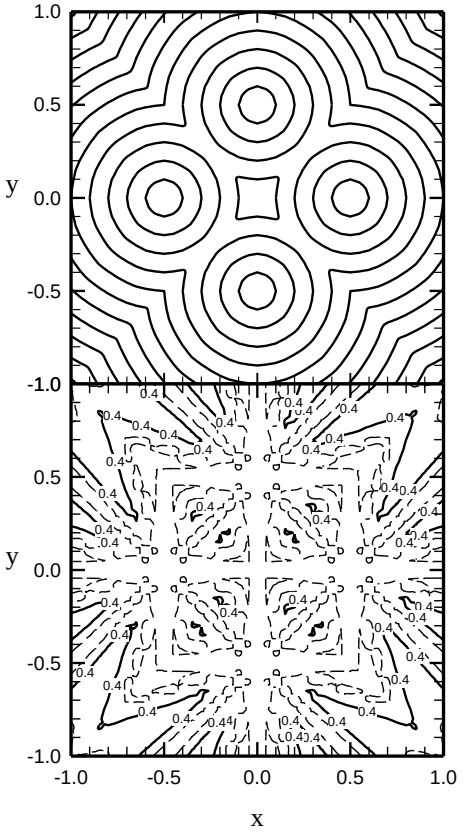


(a)到達時間

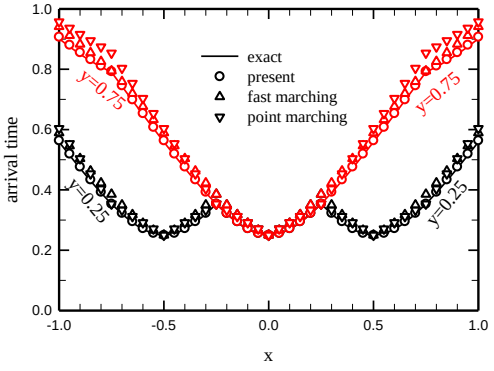


(b)相對誤差

圖 9　扭曲格點的效應

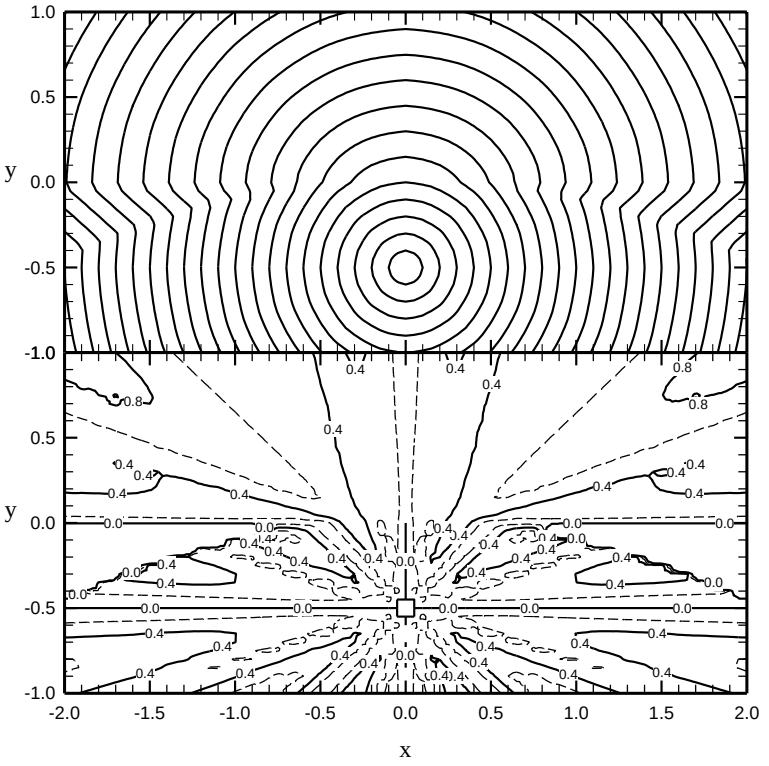


(a)界面分佈與誤差

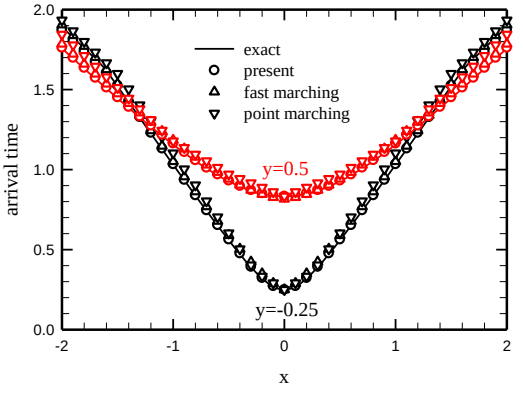


(b)不同切面結果比較

圖 10　四源點界面發展問題



(a)界面分佈與誤差



(b)不同切面結果比較

圖 11 單一源點在不同傳播速率區域界面發展問題

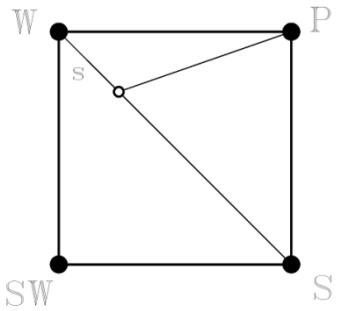


圖 12 點前進法與快速前進法發射源點示意圖

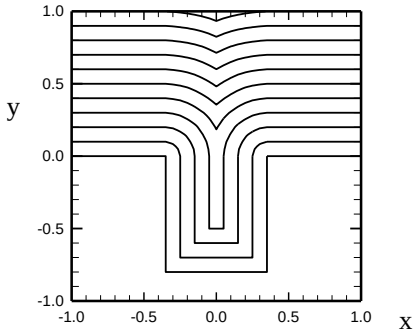
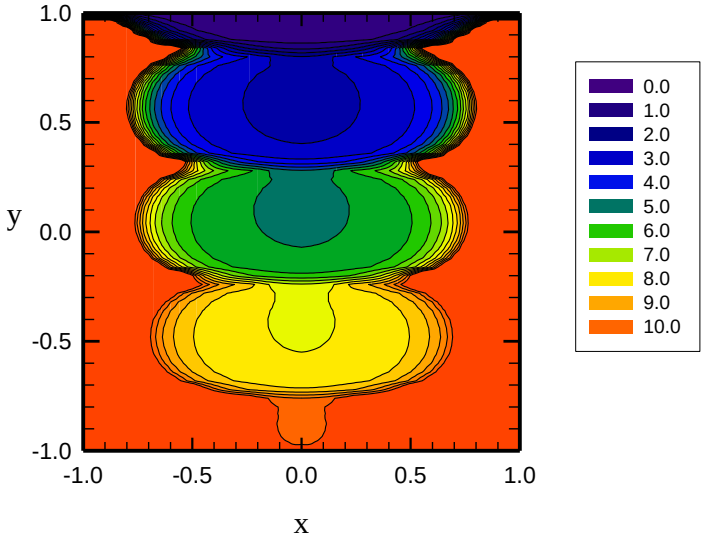
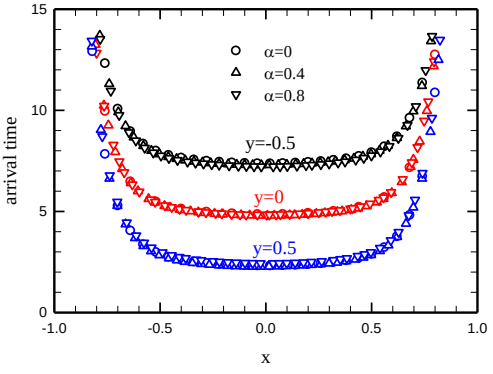


圖 13 沉積問題的界面演遞



(a)界面演遞



(b)扭曲格點效應

圖 14 刻蝕問題模擬結果

表 1 單一源點發展界面在不同格點密度下的誤差值及收斂階數

Method	Present			
Grid	20x20	40x40	80x80	160x160
ε_1 / n_{conv}	4.76×10^{-3} / – – –	3.18×10^{-3} / 0.582	2.06×10^{-3} / 0.626	1.31×10^{-3} / 0.653
ε_2 / n_{conv}	5.61×10^{-3} / – – –	3.63×10^{-3} / 0.628	2.33×10^{-3} / 0.640	1.48×10^{-3} / 0.655
ε_{max} / n_{conv}	1.03×10^{-2} / – – –	7.20×10^{-3} / 0.517	4.43×10^{-3} / 0.701	2.88×10^{-3} / 0.621
Method	Fast Marching			
ε_1 / n_{conv}	4.53×10^{-2} / – – –	2.87×10^{-2} / 0.658	1.76×10^{-2} / 0.705	1.05×10^{-2} / 0.745
ε_2 / n_{conv}	5.10×10^{-2} / – – –	3.21×10^{-2} / 0.668	1.96×10^{-2} / 0.712	1.18×10^{-2} / 0.732
ε_{max} / n_{conv}	8.21×10^{-2} / – – –	5.12×10^{-2} / 0.681	3.11×10^{-2} / 0.719	1.84×10^{-2} / 0.757
Method	Point Marching			
ε_1 / n_{conv}	3.98×10^{-2} / – – –	3.98×10^{-2} / 0	3.97×10^{-2} / 0.004	3.97×10^{-2} / 0
ε_2 / n_{conv}	4.72×10^{-2} / – – –	4.66×10^{-2} / 0.018	4.64×10^{-2} / 0.006	4.62×10^{-2} / 0.006
ε_{max} / n_{conv}	8.91×10^{-2} / – – –	8.98×10^{-2} / –0.011	8.98×10^{-2} / 0	8.98×10^{-2} / 0