2006

Theory of Curvature-Dependent Disjoining Pressure and Its Application to Liquid Films

Jin Zhang
Louisiana State University and Agricultural and Mechanical College

THEORY OF CURVATURE-DEPENDENT DISJOINING PRESSURE AND ITS APPLICATION TO LIQUID FILMS

A Dissertation
Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

in

The Department of Mechanical Engineering

by
Jin Zhang
B. S. Shanghai Jiao Tong University, 1997
M.S. Shanghai Jiao Tong University, 2000
M.S., Louisiana State University, 2002

August 2006

ACKNOWLEDGEMENTS

First of all, the author likes to thank her major professor, Dr. Harris Wong for his constant encouragement, insightful suggestions, and many devoted guidance and help throughout the research.

The author also likes to thank Dr. T. Charalampopoulos, Dr. G. Sinclair, and Dr. A. Houston for their time and energy to serve on her examination committee.

The author appreciates her colleagues in lab: T. Xin, D. Min, W. Kan, Q. Wu, T. Yi, and P. Du for their help and a great time they spent together.

Finally, the author would like to express gratefulness to her parents, her sister, her husband and son, for their continuous love and support.

TABLE OF CONTENTS

ABSTRACT

A liquid film of thickness h < 100 nm is subject to additional intermolecular forces, which are collectively called disjoining pressure Π. Since Π dominates at small film thicknesses, it determines the stability and wettability of thin films. Current theory for uniform films gives $\Pi = \Pi(h)$. It becomes unbounded as $h \to 0$. We present a theory of curvature-dependent disjoining pressure. The new Π depends on the curvature h_{xx}, slope h_x, and h. When this theory is implemented for Lennard-Jones liquid films, the new Π is bounded as $h \to 0$. We show that this Π captures three regimes of drop behavior (complete wetting, partial wetting, and pseudo partial wetting) without altering the signs of the long and short-range interactions. We also find that a drop with a uniform film is linearly stable, whereas a drop without a uniform film is unstable.

Evaporating thin films is important in solvent coating and thin-film heat transfer. In some experiments, satellite liquid drops were observed when an evaporating film retracts. We model the evolution of a two-dimensional evaporating thin film on a heated substrate. The results show that the film thins and the film edge retracts. The thinning film forms a ridge at the edge followed by a thin neck before it returns to the uniform thickness. This profile is maintained till dry-out. The ridge breaks away to form a droplet and the remaining film keeps evolve. In this way, a series of droplets is formed with decreasing volumes. We find that the drop volume depends on the evaporation parameters and the drops have similar profiles.

We explore the reasons for the similar profiles of satellite droplets. We study self-similar retraction of a step liquid film pinned at the contact line. We find a self-similar solution in which the x-coordinate is scaled with time t. We also simulate the evolution of

a pinned step film and find that the film profile always approaches the self-similar solution as $t \to \infty$. We study the linear stability of the self-similar solution and find that the self-similar solution is stable.

CHAPTER 1. INTRODUCTION

Disjoining pressure Π becomes dominant in a liquid film when the thickness h < 100nm. Therefore, it is important to determine Π accurately because it governs stability and wettability of thin films. However, the commonly used Π becomes unbounded when $h \to 0$ and cannot be applied to films that end at the substrate with non-zero contact angles. Recently, a slope-dependent disjoining pressure was proposed [1, 2]. The slope-dependent Π also depends on the curvature $h_{xx.}$ However, it is not clear whether all the curvature effects have been included because during their derivation, a liquid wedge, which has no curvature information, has been employed. We present a theory of curvature-dependent disjoining pressure, and derive a new disjoining pressure that depends on the curvature h_{xx}, the slope h_x, and the film height h. When this theory is implemented for Lennard-Jones liquid films, the new Π is bounded near a contact line as $h \to 0$. When this new Π is applied to the augmented Young-Laplace equation, we find a set of equilibrium film profiles including pancakes or drops (partial wetting), periodic drops connected with infinite or finite precursor film (pesudo partial wetting), periodic drops connected without uniform film, a uniform film (complete wetting), or a meniscus. This part of the work is presented in Chapter 2.

In Chapter 3, we study the satellite droplet formation during the evaporation of a liquid step film. The dynamics of evaporating thin liquid films play an important role in many scientific and engineering applications such as tear film drainage [3, 4], solvent coating [5], and thin-film heat transfer. Evolution of an evaporating liquid film is crucial to solvent coating because coating quality highly depends on the evaporation rate of the solvent. Liquid film evaporation is also important to understand thin film heat transfer, which is a fundamental phenomenon in many heat transfer devices. We consider a two-

dimensional evaporating thin film on a heated horizontal substrate. The film initially forms a small contact angle with the substrate at one end and extends uniformly to infinity at the other end. We solve the evolution equation numerically and find that the resulting satellite droplets have similar profiles.

We explore the reasons for the similar profiles of the droplets formed during evaporation of a step liquid film in Chapter 4. We find a self-similar solution of film retraction in which the film thickness stays constant, but the film profile spreads with time as $t^{1/4}$. Any initial film profile will eventually approach this self-similar solution. This solution describe the nature of the solution to the evolution equation. We study the linear stability of the self-similar solution and find it is stable. We conclude the dissertation in Chapter 5.

CHAPTER 2. A THEORY OF CURVATURE-DEPENDENT DISJOINING PRESSURE WITH APPLICATION TO LENNARD-JONES LIQUID FILMS

2.1. Review

A macroscopic liquid film on a solid substrate is governed by the usual forces such as gravity and capillarity. However, additional forces must be considered if the range of film thickness is under 100 nm. These forces come from inter-atomic interactions between the liquid molecules and the solid and vapor molecules surrounding the thin film. For example, two bubbles pressed against each other in a surfactant solution forms a stable thin film, despite that the liquid pressure gradient tends to derive the liquid out of the film. This implies that a force exists to prevent the film from thinning. This repulsive force per unit area is called disjoining pressure Π.

Disjoining pressure has been measured for different liquid-solid and liquid-liquid systems [6-8]. The common measurement method uses a disk made of a porous material [9-11]. The disk has a small hole in the center and is filled with a liquid to form a film in the hole. The liquid pressure in the film is varied and the film thickness measured by interferometry. This yields Π as a function of film thickness. This method can be applied to freely-suspended films as well as to films on solid substrates. Disjoining pressure also exists between solid-solid surfaces and has been studied by atomic force microscopy [6, 12].

Disjoining pressure dominates at small film thicknesses and therefore governs the stability and wettability of thin films. It determines the breakup time of the tear film covering the eyeball [13], and the continuity of the liquid lining in the lung during aerosol treatment of surfactant deficiency [14]. It controls the stability of thin polymer films on

silicon wafers [15] and can explain the development of mixed wettability in oil-reservoir rock [16]. Given the importance of thin liquid films in biological and industrial processes, it is critical to model Π accurately.

Current theory assumes that the film is uniform with thickness h and yields $\Pi = \Pi(h)$. For van der Waals forces, this leads to $\Pi = A/6\pi h^3$, where A is the Hamaker constant [6]. This expression has been applied to simulate equilibrium profiles [17-19] and evolution of non-uniform films [20-24]. Recently, short-range intermolecular forces have been included in $\Pi(h)$ to explain thin-film instability and dewetting on solid substrates [15, 25-32]. The purpose is to understand pattern formation of polymer films with the ultimate goal of controlling the patterns to form regular microstructures. The instability is commonly referred to as spinodal decomposition because the thin film is unstable in the range of h where $d\Pi/dh > 0$, similar to spinodal phase separation. A uniform film with a film thickness in the unstable range will separate into a thicker film connected smoothly to a thinner film. All the disjoining pressure isotherms used in the above studies contain the van der Waals term $A/6\pi h^3$ and thus they all become unbounded as $h \to 0$. To avoid this singularity, a thin precursor film must always exist on the substrate to prevent h from going to zero.

Since Π results from the net effect of inter-molecular interactions and is determined by summing intermolecular potentials between different phases, Π should depend on the domain shape of the liquid phase, i.e., Π should depend on the entire film profile. This means that Π should depend not only on the local film thickness h, but also should contain information about the neighboring film profile. Therefore, $\Pi = \Pi(h, h_x, h_{xx}, \ldots)$. This dependence reduces to $\Pi = \Pi(h)$ if the film is uniform with slope $h_x = 0$.

The inclusion of the gradient dependence is common in modeling and has led to many successful theories.[33, 34]

The commonly used $\Pi(h)$ was derived for uniform films, but is being applied to non-uniform films. Is this the origin of the singularity near a contact line as $h \to 0$? To answer this question, we need to formulate a theory of Π that depends on the film slope h_x as well as the film height h. Previous attempts to derive $\Pi = \Pi(h, h_x, h_{xx})$ do not contain a proper equilibrium condition [35, 36]. As a result, there is ambiguity in the relation between Π and the intermolecular potential. Here, we present a theory in which equilibrium conditions are obtained from energy minimization. We apply this theory to Lennard-Jones liquid films and find that the resulting Π can be bounded near a contact line as $h \to 0$. Thus, the singularity in $\Pi = \Pi(h)$ near a contact line is not physical but artificial because it is created by applying a simple theory beyond its range of validity.

Using the framework $\Pi = \Pi(h)$, Brochard-Wyart et al. [25] classified drop shapes into three categories depending on two factors: the Hamaker constant *A*, and the spreading coefficient *S* that is affected by short-range interactions. By varying the signs of *A* and *S*, they found three regimes of drop behavior: complete wetting, partial wetting, and pseudo partial wetting. In complete wetting, a small drop spreads to become a flat "pancake" surrounded by a dry solid. In partial wetting, a drop forms a spherical cap with a finite contact angle α and the surrounding solid is dry. In pseudo partial wetting, a drop maintains a finite macroscopic contact angle α but is surrounded by a precursor film.[32] Recently, Yeh et al. [37] confirmed the predictions of Brochard-Wyart et al. by constructing a $\Pi = \Pi(h)$ with eight fitting parameters. Here, our slope-dependent Π contains a long-range attractive component and a short-range repulsive component. The

new Π can capture all three regimes of drop behavior without altering the signs of the long- and short-range components.

This chapter is organized as the following. The Helmholtz free energy of a drop on a substrate is minimized to yield three equilibrium condition (§2.2). The intermolecular potential is presented in §2.3. One condition relates Φ to an excess interaction energy E, which is found by summing Lennard-Jones potentials. The other two give boundary conditions (§2.4). The new disjoining pressure is incorporated into the augmented Young-Laplace equation and non-dimensionalized in section §2.5. The dimensionless equation is solved for uniform films in section §2.6 and for equilibrium drop profiles in section §2.7. The linear stability of a drop with or without a uniform film is studied in section §2.8. We discuss the implications in section §2.9 and conclude this chapter in section §2.10.

2.2. Minimization of the Total Energy

Consider a mono-atomic liquid drop without charge on a smooth solid surface and in thermo- dynamic equilibrium with its own vapor at constant temperature, as illustrated in Fig. 2.1. The drop size is assumed small to neglect gravitational effects and the interface is located following the Gibbs model. The system is located in a closed box and the drop is assumed to be two dimensional and symmetric. In this system, the total energy consists of surface energies and an excess interaction potential energy E. The mass of the liquid drop and the temperature of the system are constant. The total energy of the system is minimized when the system reaches equilibrium [1, 38]. At the minimum energy state, variation of this total energy is null [39, 40]:

$$\delta\int_0^{x_0}\left[\sigma\left(1+h_x^2\right)^{1/2}+\sigma_{fs}-\sigma_{sg}+E+P_c h\right]dx=0, \qquad (2.2.1)$$

where δ represents the variation of a functional, h is the film height, x is a horizontal coordinate starting at the center of the drop, h_x is the film slope, and σ is the liquid-vapor interfacial surface tension. The arc length of a liquid-vapor interfacial element is $(1+h_x^2)^{1/2}dx$, the half width of the drop is x_0, σ_{fs} and σ_{sg} are surface tensions between liquid and solid and between solid and gas, E = E(h, h_x) is an excess energy due to thin film forces and depends on the film thickness and slope, and p_c is a Lagrange multiplier. The term $p_c h$ imposes mass conservation. Eq. (2.2.1) can be derived rigorously using thermodynamics [38]. To include the curvature effect, we take E = E(h, h_x, h_{xx}), where $h_{xx} = \partial^2 h / \partial x^2$ is the curvature of the liquid-vapor interface.

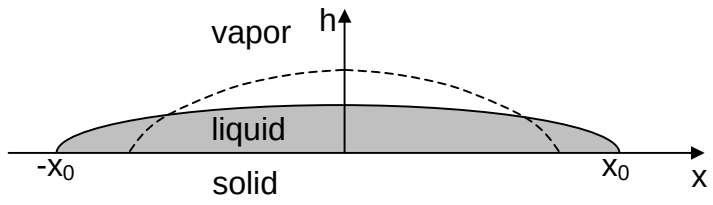


Figure 2.1. A two-dimensional drop on a solid substrate. The drop is in thermodynamic equilibrium with its vapor and is symmetric. A horizontal coordinate x starts from the center of the drop. The drop width is $2x_0$ and the drop height is denoted by h = h(x). The dashed line represents a perturbed drop shape.

After applying the Leibniz rule, Eq. (2.2.1) is expanded as

$$\left[\sigma\left(1+h_x^2\right)^{1/2}+\sigma_{fs}-\sigma_{sg}+E\right]_{x_0}\delta x_0+\int_0^{x_0}\left(\frac{\partial E}{\partial h}+P_c\right)(\delta h)dx$$
$$+\int_0^{x_0}\left(\frac{\sigma_{fg}h_x}{\left(1+h_x^2\right)^{1/2}}+\frac{\partial E}{\partial h_x}\right)(\delta h_x)dx+\int_0^{x_0}\left(\frac{\partial E}{\partial h_{xx}}\right)(\delta h_{xx})dx=0. \qquad (2.2.2)$$

Since $\delta h_x = d(\delta h) / dx$, and $\delta h_{xx} = d^2(\delta h) / dx^2$, Eqt. (2.2.2) can be rearranged using integration by parts to

$$\int_0^{x_0}\left[P_c-\frac{\sigma h_{xx}}{\left(1+h_x^2\right)^{3/2}}+\frac{\partial E}{\partial h}-\frac{d}{dx}\left(\frac{\partial E}{\partial h_x}\right)+\frac{d^2}{dx^2}\left(\frac{\partial E}{\partial h_{xx}}\right)\right](\delta h)dx$$
$$+\left[\frac{\sigma}{\left(1+h_x^2\right)^{1/2}}+\sigma_{fs}-\sigma_{sg}+E-h_x\frac{\partial E}{\partial h_x}+h_x\frac{d}{dx}\left(\frac{\partial E}{\partial h_{xx}}\right)-h_{xx}\frac{\partial E}{\partial h_{xx}}\right]_{x_0}\delta x_0$$
$$+\left[-\frac{\sigma h_x}{\left(1+h_x^2\right)^{1/2}}-\frac{\partial E}{\partial h_x}+\frac{d}{dx}\left(\frac{\partial E}{\partial h_{xx}}\right)\right](\delta h)\Bigg|_{x=0}=0, \tag{2.2.3}$$

where we have applied the following relations: $\delta h|_{x=x_0}=-h_x\delta x_0|_{x=x_0}$, $\delta(h_x)|_{x=x_0}=-h_{xx}\delta x_0$ from the geometry, and $\delta(h_x)|_{x=0}=0$ from the symmetry of the drop, and from the assumption that the slope at the contact line is constant.

Because δx_0 and δh are arbitrary, the three coefficients in (2.2.3) must be zero. The first coefficient leads to

$$\frac{\sigma h_{xx}}{\left(1+h_x^2\right)^{3/2}}-\frac{\partial E}{\partial h}+\frac{d}{dx}\left(\frac{\partial E}{\partial h_x}\right)-\frac{d^2}{dx^2}\left(\frac{\partial E}{\partial h_{xx}}\right)=P_c. \tag{2.2.4}$$

Compare this with the augmented Young-Laplace equation [19], we find the disjoining pressure Π as

$$\Pi=-\frac{\partial E}{\partial h}+\frac{d}{dx}\left(\frac{\partial E}{\partial h_x}\right)-\frac{d^2}{dx^2}\left(\frac{\partial E}{\partial h_{xx}}\right), \tag{2.2.5}$$

and

$$P_c=p_g-p_f, \tag{2.2.6}$$

where p_g and p_f are the vapor and liquid pressures. The first two terms in Π are the same as previous models [1]. The last term in Π contains the curvature dependence.

The other two coefficients in (2.2.3) give the boundary conditions for the augmented Young-Laplace equation. At the edge of the drop $x = x_0$,

$$\frac{\sigma}{\left(1+h_x^2\right)^{1/2}} + \sigma_{fs} - \sigma_{sg} + E - h_x \frac{\partial E}{\partial h_x} + h_x \frac{d}{dx}\left(\frac{\partial E}{\partial h_{xx}}\right) - h_{xx} \frac{\partial E}{\partial h_{xx}} = 0. \qquad (2.2.7)$$

And at the center of the drop $x = 0$,

$$\frac{\sigma h_x}{\left(1+h_x^2\right)^{1/2}} + \frac{\partial E}{\partial h_x} - \frac{d}{dx}\left(\frac{\partial E}{\partial h_{xx}}\right) = 0. \qquad (2.2.8)$$

These conditions contain the new derivative with respect to curvature.

To find $E = E(h, h_x, h_{xx})$ an intermolecular potential model must be selected. Here, we use the Lennard-Jones potential. Given the potential, we still need to determine the functional dependence of E on h, h_x, and h_{xx}. A liquid drop in the shape of a part circle with radius S is used to find E. The part circle is the simplest geometry that can yield h, h_x, and h_{xx} dependence.

2.3. Intermolecular Potential

We consider a two-dimensional liquid domain in the shape of a part circle with radius S (Fig. 2.2). Unlike the previous derivations in which a wedge liquid domain was used, we need a curved interface to capture the curvature effect. We calculate the intermolecular potential energy between a liquid molecule M and all other molecules. The liquid molecule M interacts with another molecule N in solid (s), liquid (f), or vapor (g) through the Lennard-Jones potential:

$$\phi_{fs} = \frac{L_{fs}}{MN^{12}} - \frac{\beta_{fs}}{MN^6}, \phi_{ff} = \frac{L_{ff}}{MN^{12}} - \frac{\beta_{ff}}{MN^6}, \phi_{fg} = \frac{L_{fg}}{MN^{12}} - \frac{\beta_{fg}}{MN^6}, \qquad (2.3.1a, b, c)$$

where L and β are the strengths of the repulsive and attractive components, and MN is the distance between M and N. Subscripts s, f, and g denote solid, liquid, and vapor.

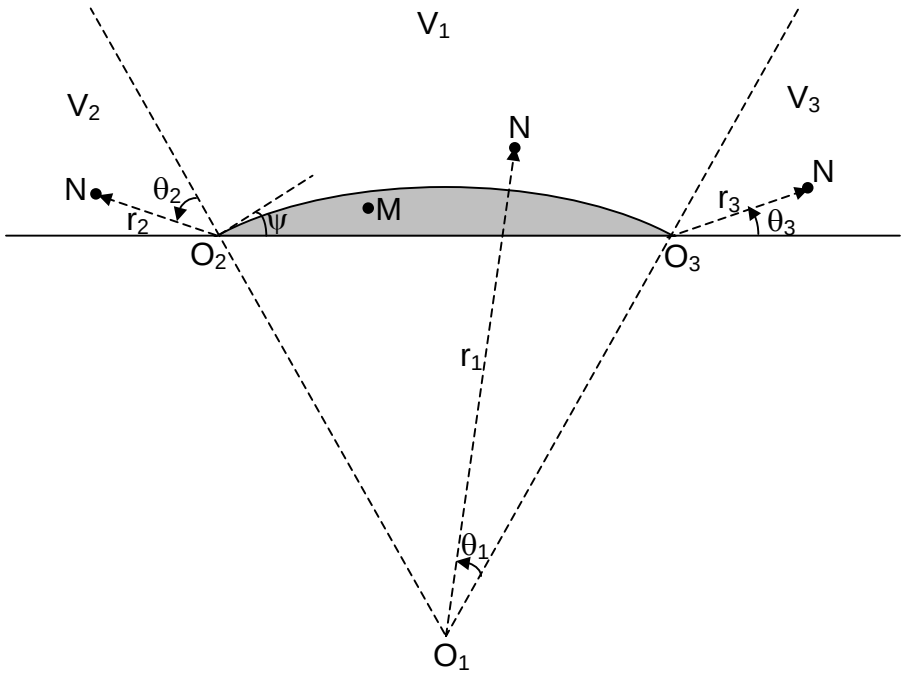


Figure 2.2. A liquid drop on a solid substrate in equilibrium with its vapor. The drop is in the shape of a part circle with radius a_0. Cartesian coordinates (x, y, z) are chosen so that the x-axis is long the liquid-solid interface; the drop is symmetric about y-axis, and z-axis pointing out of the paper. The drop contacts the substrate with an angle ψ. M is a liquid molecule, and N is a molecule in liquid, vapor, or solid.

The circular-arc liquid surface has radius S and contacts the substrate with angle ψ (Fig. 2.2). The liquid molecule M is at a height y from the substrate. Thus, we show in later sub-sections that the total interaction potential Φ between M and other molecules depends on S, ψ, and y. In the next section, Φ is integrated with respect to y to yield E = E(S, ψ, h), where h is the height of the liquid surface at M. Since $S = -1/h_{xx}$, and $\psi = \tan^{-1} h_x$, we find $E = E(h_{xx}, h_x, h)$. For this functional dependence to work, we need to be able

to vary S, ψ, and h independently. This is indeed the case for the circular-arc geometry. For example, we can fix S and ψ, and vary h. However, the range of variation for each parameter is limited.

We can separate Φ into two parts: bulk and thin-film. The thin-film part of Φ will be integrated to yield E, which is an excess interaction energy that vanishes in the bulk. Thus, the bulk part of Φ can be identified by letting $S \to \infty$ and $y \to \infty$ (same as $h \to \infty$).

2.3.1. Solid Substrate

The interactive potential between the liquid molecule M and the solid substrate is

$$\Phi_{fs} = \iiint_{V_s} n_s \phi_{fs} dV, \tag{2.3.2}$$

where n_s is the number density of the solid molecules, and V_s is the lower half plane occupied by the solid. This integration is the same as that in Yi & Wong [2]:

$$\Phi_{fs} = \frac{\pi n_s L_{fs}}{45 y^9} - \frac{\pi n_s \beta_{fs}}{6 y^3}, \tag{2.3.3}$$

where y is the vertical height of M from the solid surface (Appendix A).

2.3.2. Vapor

To calculate the intermolecular potential Φ_{fg} between the liquid molecule M and vapor molecules, it is convenient to divide the vapor domain into three sub-domains (Fig. 2.2):

$$\Phi_{fg} = \sum_{i=1}^{3} \Phi_{fgi} = \sum_{i=1}^{3} \iiint_{V_i} n_g \phi_{fg} dV, \tag{2.3.4}$$

where V_i is the volume of the sub-domain i, and n_g is the number density of vapor. Sub-domain 1 is the vapor above the circular liquid surface contained between two straight lines normal to the liquid surface at the contact lines (Fig. 2.2). Sub-domain 2 is the vapor

wedge on the left of sub-domain 1, and sub-domain 3 is the vapor wedge on the right. A cylindrical coordinate system (r_i, θ_i, z) is defined for each sub-domain i with origin at O_i, i = 1, 2, 3, as depicted in Fig. 2.2.

In sub-domain 1, N is located at (r_1, θ_1, z) and M is at (R_1, Γ_1, 0). The potential is

$$\Phi_{fg1} = n_g \int_0^{2\psi} \int_S^{+\infty} \int_{-\infty}^{+\infty} \left(\frac{L_{fg}}{MN^{12}} - \frac{\beta_{fg}}{MN^6} \right) dz r_1 dr_1 d\theta_1, \tag{2.3.5}$$

where S is the radius of the circular liquid surface, ψ is the contact angle, and

$$MN^2 = R_1^2 + r_1^2 + z^2 - 2\rho r_1 \cos(\theta_1 - \Gamma_1). \tag{2.3.6}$$

Appendix A gives in the limit $\psi \to 0$,

$$\Phi_{fg1} = \frac{63\pi n_g L_{fg}}{256} \left[\frac{(10S - R_1)\psi}{45(S - R_1)^{10}} - \frac{R_1\left(R_1^2 - 12SR_1 + 66S^2\right)\psi\left(4\psi^2 + 3\Gamma_1^2 - 6\psi\Gamma_1\right)}{180(S - R_1)^{12}} \right]$$
$$- \frac{3\pi n_g \beta_{fg}}{8} \left[\frac{(4S - R_1)\psi}{6(S - R_1)^4} - \frac{R_1\left(R_1^2 - 6SR_1 + 15S^2\right)\psi\left(4\psi^2 + 3\Gamma_1^2 - 6\psi\Gamma_1\right)}{36(S - R_1)^6} \right]. \tag{2.3.7}$$

If $S \to \infty$ and $R_1 \to \infty$, but (S - ρ) and ψ are kept fixed, then $\Phi_{fg1} \to \infty$. Thus, Φ_{fg1} belongs to the bulk contribution and plays no part in the thin-film excess interaction energy E.

Sub-domain 2 is a wedge and a cylindrical coordinate system is defined with origin O_2 at the wedge tip, as shown in Fig. 2.2. In this system, N is located at (r_2, θ_2, z) and M at (R_2, Γ_2, 0) and. The potential is

$$\Phi_{fg2} = n_g \int_0^{\frac{\pi}{2} - \psi} \int_0^{+\infty} \int_{-\infty}^{+\infty} \left(\frac{L_{fg}}{MN^{12}} - \frac{\beta_{fg}}{MN^6} \right) dz r_2 dr_2 d\theta_2, \tag{2.3.8}$$

This integration can be evaluated exactly (Appendix A):

$$\Phi_{fg2} = \frac{\pi n_g L_{fg}}{11520}\left(\frac{b_1}{y^9} + \frac{b_2}{v_2^9}\right) - \frac{\pi n_g \beta_{fg}}{6}\left(\frac{a_1}{y^3} + \frac{a_2}{v_2^3}\right), \tag{2.3.9a}$$

where

$$a_1 = -\frac{1}{2} - \frac{3}{4}\sin(\psi + \Gamma_2) + \frac{1}{4}\sin^3(\psi + \Gamma_2), \tag{2.3.9b}$$

$$a_2 = \frac{1}{2} + \frac{3}{4}\cos\Gamma_2 - \frac{1}{4}\cos^3\Gamma_2, \tag{2.3.9c}$$

$$\begin{aligned} b_1 = {} & -128 - 315\sin(\psi + \Gamma_2) + 420\sin^3(\psi + \Gamma_2) \\ & - 378\sin^5(\psi + \Gamma_2) + 180\sin^7(\psi + \Gamma_2) - 35\sin^9(\psi + \Gamma_2). \end{aligned} \tag{2.3.9d}$$

$$\begin{aligned} b_2 = {} & 128 + 315\cos\Gamma_2 - 420\cos^3\Gamma_2 \\ & + 378\cos^5\Gamma_2 - 180\cos^7\Gamma_2 + 35\cos^9\Gamma_2. \end{aligned} \tag{2.3.9e}$$

$$v_2 = -R_2\sin\Gamma_2. \tag{2.3.9f}$$

There are two integrated groups in Φ_{fg2}, one from each wedge surface. The terms containing y comes from evaluating the integral at the vapor-solid interface, whereas the terms containing v_2 are evaluated at the inclined surface (v_2 is the normal distance between M and the inclined surface). We treat the terms $-\pi n_g \beta_{fg}/12v_2^3$ and $\pi n_g L_{fg}/90v_2^9$ the bulk terms because as $S \to \infty$ whereas v_2 keeps fixed, these two terms are unbounded. However, the other terms containing v_2 are thin-film terms (Appendix A).

Sub-domain 3 is another wedge and a cylindrical coordinate system is defined with origin O_3 at the wedge tip (Fig. 2.2). In this system, N is located at (r_3, θ_3, z) and M at (R_3, Γ_3, 0). The potential Φ_{fg3} has the same form as Φ_{fg2}, because sub-domain shapes are mirror images. Therefore,

$$\Phi_{fg3} = \frac{\pi n_g L_{fg}}{11520}\left(\frac{b_3}{y^9} + \frac{b_4}{v_3^9}\right) - \frac{\pi n_g \beta_{fg}}{6}\left(\frac{a_3}{y^3} + \frac{a_4}{v_3^3}\right), \tag{2.3.10a}$$

where

$$a_3 = -\frac{1}{2} - \frac{3}{4}\cos\Gamma_3 + \frac{1}{4}\sin^3\Gamma_3, \tag{2.3.10b}$$

$$a_4 = \frac{1}{2} + \frac{3}{4}\sin(\psi + \Gamma_3) - \frac{1}{4}\sin^3(\psi + \Gamma_3), \tag{2.3.10c}$$

$$\begin{aligned} b_3 = &-128 - 315\cos\Gamma_3 + 420\cos^3\Gamma_3 \\ &- 378\cos^5\Gamma_3 + 180\cos^7\Gamma_3 - 35\cos^9\Gamma_3. \end{aligned} \tag{2.3.10d}$$

$$\begin{aligned} b_4 = &+128 + 315\sin(\psi + \Gamma_3) - 420\sin^3(\psi + \Gamma_3) \\ &+ 378\sin^5(\psi + \Gamma_3) - 180\sin^7(\psi + \Gamma_3) + 35\sin^9(\psi + \Gamma_3). \end{aligned} \tag{2.3.10e}$$

$$v_3 = -R_3\cos(\psi + \Gamma_3). \tag{2.3.10f}$$

By the same reasoning as in sub-domain 2, we treat the terms $-\pi n_g \beta_{fg}/12v_3^2$ and $\pi n_g L_{fg}/90v_3^9$ as bulk terms and drop them. And keep the other terms containing v_3 because they are thin-film terms (Appendix A).

Summing up Φ_{fg1}, Φ_{fg2}, and Φ_{fg3} and dropping the bulk terms gives the interaction potential between the liquid molecule M and vapor:

$$\begin{aligned} \Phi_{fg} = &\frac{\pi n_g L_{fg}}{11520}\left(\frac{b_1 + b_3}{y^9} + \frac{b_2 - 128}{v_2^9} + \frac{b_4 - 128}{v_3^9}\right) \\ &- \frac{\pi n_g \beta_{fg}}{6}\left(\frac{a_1 + a_3}{y^3} + \frac{a_2 - \frac{1}{2}}{v_2^3} + \frac{a_4 - \frac{1}{2}}{v_3^3}\right). \end{aligned} \tag{2.3.11}$$

2.3.3. Liquid

The potential Φ_{ff} is easily found by using the expression of Φ_{fs} and Φ_{fg} in (2.3.3) and (2.3.11). Let Φ_{-D} be the intermolecular potential between M and an infinite body of liquid V_{-D} outside a sphere of radius D surrounding M:

$$\Phi_{-D} = \iiint_{V_{-D}} n_f \phi_{ff} dV == \int_D^{+\infty} \left(\frac{n_f L_{ff}}{r^{12}} - \frac{n_f \beta_{ff}}{r^6} \right) \left(4\pi r^2 \right) dr = \frac{4\pi n_f L_{ff}}{9D^9} - \frac{4\pi n_f \beta_{ff}}{3D^3}. \quad (2.3.12)$$

Let Φ_C be the intermolecular potential between M and a liquid occupying the vapor and solid regions:

$$\Phi_c = \frac{\pi n_f L_{ff}}{11520} \left(\frac{b_1 + b_3 + 256}{y^9} \right) - \frac{\pi n_f \beta_{ff}}{6} \left(\frac{a_1 + a_3 + 1}{y^3} \right). \quad (2.3.13)$$

Since the interaction potential is the integral over space occupied by the materials,

$$\begin{aligned} \Phi_{ff} &= \Phi_{-D} - \Phi_C \\ &= \frac{4\pi n_f L_{ff}}{9D^9} - \frac{\pi n_f L_{ff}}{11520} \left(\frac{b_1 + b_3 + 256}{y^9} + \frac{b_2 - 128}{v_2^9} + \frac{b_4 - 128}{v_3^9} \right) \\ &- \frac{4\pi n_f \beta_{ff}}{3D^3} + \frac{\pi n_f \beta_{ff}}{6} \left(\frac{a_1 + a_3 + 1}{y^3} + \frac{a_2 - \frac{1}{2}}{v_2^3} + \frac{a_4 - \frac{1}{2}}{v_3^3} \right). \end{aligned} \quad (2.3.14)$$

Because this potential is relative, we can neglect the constant terms in (3.3.14).

2.3.4. Total Potential

The total interaction thin-film potential between point M in liquid and other material points is found by summing (2.3.3), (2.3.11) and (2.3.14):

$$\Phi = n_f\left(\Phi_{fs} + \Phi_{fg} + \Phi_{ff}\right)$$
$$= -\frac{\pi n_f^2 L_{ff}}{11520}\left[\frac{256(1-\lambda_1)}{y^9} + \left(\frac{b_1+b_3}{y^9} + \frac{b_2-128}{v_2^9} + \frac{b_4-128}{v_3^9}\right)(1-\lambda_2)\right] \quad (2.3.15)$$
$$+ \frac{\pi n_f^2 \beta_{ff}}{6}\left[\frac{(1-\rho_1)}{y^3} + \left(\frac{a_1+a_3}{y^3} + \frac{a_2-\frac{1}{2}}{v_2^3} + \frac{a_4-\frac{1}{2}}{v_3^3}\right)(1-\rho_2)\right].$$

$$\lambda_1 = \frac{n_s L_{fs}}{n_f L_{ff}}, \quad \lambda_2 = \frac{n_g L_{fg}}{n_f L_{ff}}, \quad \rho_1 = \frac{n_s \beta_{fs}}{n_f \beta_{ff}}, \quad \rho_2 = \frac{n_g \lambda_{fg}}{n_f \lambda_{ff}}. \quad (2.3.16)$$

This potential depends on y, v_2, v_3, and angles Γ_2, Γ_3, $(\psi+\Gamma_2)$ and $(\psi+\Gamma_3)$.

2.4. Excess Interaction Energy E

The excess energy E is energy per unit substrate area, and the interaction potential Φ is energy per unit volume at point M. Thus, we take

$$E = \int_D^h \Phi dy. \quad (2.4.1)$$

The integration starts from the cut-off distance D because the cut-off sphere around M contains liquid, and Φ becomes unbounded if M touches the solid. As derived in §3, $\Phi = \Phi(y, v_2, v_3, \Gamma_2, \Gamma_2, \psi+\Gamma_2, \psi+\Gamma_3)$.

From geometry,

$$\sin(\psi+\Gamma_2) = -\frac{x_2}{\sqrt{x_2^2+y^2}}, \quad (2.4.2)$$

$$\cos\Gamma_2 = \frac{y}{\sqrt{x_2^2+y^2}}\cos\psi - \frac{x_2}{\sqrt{x_2^2+y^2}}\sin\psi, \quad (2.4.3)$$

$$v_2 = x_2\cos\psi + y\sin\psi, \quad (2.4.4)$$

where x_2 is the horizontal position of point M from O_2:

$$x_2 = S\sin\psi - \left[S^2 - (S\cos\psi + h)^2\right]^{1/2}. \quad (2.4.5)$$

If point M is moved close to O_2, i.e., if $h << S \sin^2\psi$, then

$$x_2 \approx \frac{h}{\tan\psi}\left(1+\frac{h}{2S\sin^2\psi\cos\psi}\right). \tag{2.4.6}$$

Similarly,

$$\cos\Gamma_3 = -\frac{x_3}{\sqrt{x_3^2+y^2}}, \tag{2.4.7}$$

$$\sin(\psi+\Gamma_3) = -\frac{x_3}{\sqrt{x_3^2+y^2}}\sin\psi + \frac{y}{\sqrt{x_3^2+y^2}}\cos\psi, \tag{2.4.8}$$

$$v_3 = x_3\cos\psi + y\sin\psi, \tag{2.4.9}$$

where x_3 is the horizontal position of point M from O_3:

$$x_3 = S\sin\psi + \left[S^2-(S\cos\psi+h)^2\right]^{1/2}. \tag{2.4.10}$$

In the limit $h << S \sin^2\psi$,

$$x_3 = S\sin\psi\left(2-\frac{h}{S\sin^2\psi}\right). \tag{2.4.11}$$

Thus, the integration in (2.4.1) yields E = E(S, ψ, h, D), as shown in Appendix B. In the limit $D / h \to 0$, $h / x_2 \to 0$, and $h / x_3 \to 0$, we get

$$E = -\frac{\pi n_f^2\beta_{ff}(1-\rho_2)}{64h^2}\left[\frac{16(1-\rho_1)}{3(1-\rho_2)}+\frac{\tan^4\psi\left(4+\cos^6\psi\right)}{\cos^6\psi}-\frac{h\sin^2\psi\left(2+\cos^2\psi\right)}{\cos^7\psi}\right]$$
$$+\frac{7\pi n_f^2 L_{ff}(1-\lambda_2)}{5120h^8}\left[\frac{128(1-\lambda_1)}{63(1-\lambda_2)}+\frac{\tan^{10}\psi\left(10+8\cos^8\psi\right)}{\cos^8\psi}-\frac{5h\sin^8\psi\left(8+\cos^8\psi\right)}{\cos^{19}\psi}\right]. \tag{2.4.12}$$

As point M is moved close to O_2,

$$\tan\psi = h_x, \tag{2.4.13}$$

and since $\psi << 1$,

$$\sin\psi = h_x, \quad \cos\psi = 1, \quad S = -\frac{1}{h_{xx}}. \qquad \text{(2.4.14a, b, c)}$$

Thus,

$$\begin{aligned} E = {} & \frac{77\pi n_f^2 L_{ff}(1-\lambda_2)}{5120h^8}\left[\frac{128(1-\lambda_1)}{693(1-\lambda_2)} + h_x^{10} + \frac{45}{11}hh_x^8 h_{xx}\right] \\ & - \frac{5\pi n_f^2 \beta_{ff}(1-\rho_2)}{64h^2}\left[\frac{16(1-\rho_1)}{15(1-\rho_2)} + h_x^4 + \frac{6}{5}hh_x^2 h_{xx}\right]. \end{aligned} \qquad (2.4.15)$$

The first term in (2.4.15) is from the repulsive component and the second is from the attractive component. Both of them depend on curvature.

2.5. Disjoining Pressure

Following (2.2.5), the disjoining pressure Π is then determined as:

$$\Pi = \frac{J}{h^9}\left(\eta^{10} - h_x^{10} + \frac{5}{4}hh_x^8 h_{xx}\right) - \frac{B}{h^3}\left(\alpha^4 - h_x^4 + 2hh_x^2 h_{xx}\right), \qquad \text{(2.5.1a)}$$

where

$$J = \frac{1449 n_f^2 L_{ff}\pi(1-\lambda_2)}{320}, \; B = \frac{21 n_f^2 \beta_{ff}\pi(1-\rho_2)}{32},$$

$$\eta = \left[\frac{64(1-\lambda_1)}{13041(1-\lambda_2)}\right]^{\frac{1}{10}}, \; \alpha = \left[\frac{16(1-\rho_1)}{63(1-\rho_2)}\right]^{\frac{1}{4}}. \qquad \text{(2.5.1b)}$$

The functional form in (2.5.1a) is the same as that derived using only the slope-dependent E [2]. However, the macroscopic contact angle α and the microscopic contact angle η, and the strengths J and B of the repulsive and attractive components have different numerical constants. Although J, B, η, and α are different, the products $J\eta^{10}$ and $B\alpha^4$ remain the same.

Since the functional form (2.5.1a) is the same, this new Π is also bounded as h→0 if

$$\left|h_x\right| \to \eta, \tag{2.5.2a}$$

$$\frac{d^n h}{dx^n} \to 0, n = 2-6,8, \tag{2.5.2b}$$

$$\frac{d^7 h}{dx^7} \to \frac{288B\left(\eta^4 - \alpha^4\right)}{J\eta^3}. \tag{2.5.2c}$$

Then, as h→0,

$$\Pi \to \frac{J}{290304}\frac{d^{10}h}{dx^{10}} \ , \tag{2.5.3}$$

which is bounded if $d^{10}h/dx^{10}$ is finite. Note that d^9h/dx^9 needs to be finite but not necessarily zero as h→0. The above conditions are found by use of L'Hospital's rule.

2.6. Boundary Conditions

The boundary condition at x = 0 (2.2.8) leads to

$$\frac{\sigma h_x}{(1+h_x{}^2)^{1/2}} + \frac{13\pi n_f^2 h_x{}^3}{32h^8}\left[\frac{595}{416}L_{ff}(1-\lambda_2)h_x^6 - \beta_{ff}(1-\rho_2)h^6\right] = 0, \tag{2.6.1}$$

which simplifies to the symmetry condition $h_x = 0$.

Form Eq. (2.4.12),

$$E - h_x\frac{\partial E}{\partial h_x} + h_x\frac{d}{dx}\left(\frac{\partial E}{\partial h_{xx}}\right) - h_{xx}\frac{\partial E}{\partial h_{xx}} = \frac{J}{8}\left(\frac{\eta^{10} - h_x{}^{10}}{h^8}\right) - \frac{B}{2}\left(\frac{\alpha^4 - h_x{}^4}{h^2}\right). \tag{2.6.2}$$

Near the contact line as h→0, if the conditions in Eq. (2.5.2c) are met,

$$E - h_x\frac{\partial E}{\partial h_x} + h_x\frac{d}{dx}\left(\frac{\partial E}{\partial h_{xx}}\right) - h_{xx}\frac{\partial E}{\partial h_{xx}} \to -\frac{J\eta}{32256}\frac{d^9h}{dx^9} \ . \tag{2.6.3}$$

Since d^9h/dx^9 needs to be finite, but need not be zero for Π to be bounded as $h\to 0$. Young's equation may not hold at the microscopic contact line.

Young's equation can be satisfied at the macroscopic contact line. In the macroscopic region, the short-range forces are negligible compared with the long-range forces. Thus, the macroscopic region can be studied by setting J = 0. Then, Eq. (2.6.2) gives that as $h\to 0$,

$$E - h_x \frac{\partial E}{\partial h_x} + h_x \frac{d}{dx}\left(\frac{\partial E}{\partial h_{xx}}\right) - h_{xx}\frac{\partial E}{\partial h_{xx}} \to B\alpha \frac{d^3 h}{dx^3}. \qquad (2.6.4)$$

The above result requires $|h_x| \to \alpha$ and $h_{xx} \to 0$, which are the necessary conditions for Π (with J = 0) to be bounded as $h\to 0$ [1]. We show in §2.10 that the term in Eq. (2.6.4) is small compared with the other terms in Eq. (2.2.7) for a macroscopic drop. Thus, Young's equation is recovered at the macroscopic contact line.

2.7. Augumented Young-Laplace Equation

From Eqt. (2.2.4) and (2.5.1), the augmented Young-Laplace equation reads

$$p_c = \sigma h_{xx} + \frac{J}{h^9}\left(\eta^{10} - h_x^{10} + \frac{5}{4}hh_x^8 h_{xx}\right) - \frac{B}{h^3}\left(\alpha^4 - h_x^4 + 2hh_x^2 h_{xx}\right), \qquad (2.7.1)$$

where h_x^2 in the curvature has been dropped since $h_x << 1$. The vapor-liquid pressure difference p_c at the liquid-vapor interface is balanced by the capillary and the disjoining pressures.

By defining dimensionless variables:

$$\varepsilon = \frac{B\alpha^2}{\sigma {h_0}^2}, R = \frac{J\alpha^8}{\varepsilon\sigma {h_0}^8}, C = \frac{p_c h_0}{\varepsilon\sigma\alpha^2}, \xi = \frac{\eta}{\alpha}, H = \frac{h}{h_0}, X = \frac{\varepsilon^{1/2}\alpha x}{h_0}, \qquad (2.7.2)$$

where h_0 is an unspecified film thickness, ε is the dimensionless attractive component in Π, R is the ratio between the dimensionless repulsive component to attractive component, C is the ratio between the dimensionless vapor-liquid pressure difference to the attractive component, and ξ is the ratio between the macroscopic contact angle to the microscopic one. The augmented Young-Laplace equation is nondimensinalized as

$$C = H_{XX} + \frac{R}{H^9}\left(\xi^{10} - \varepsilon^5 H_X^{10} + \frac{5}{4}\varepsilon^5 H H_X^8 H_{XX}\right) - \frac{1}{H^3}\left(1 - \varepsilon^2 H_X^4 + 2\varepsilon^2 H H_X^2 H_{XX}\right). \quad (2.7.3)$$

Integrating (2.7.3) with respect to H gives

$$\frac{1}{2}{H_X}^2 + \frac{1-\varepsilon^2 {H_X}^4}{2H^2} - R\left(\frac{\xi^{10} - \varepsilon^5 {H_X}^{10}}{8H^8}\right) = CH + K, \quad (2.7.4)$$

where K is an integration constant. Together with appropriate boundary conditions, this equation leads to the precursor film thickness.

2.8. Equilibrium Film Profiles

For a uniform film, we have

$$H = 1, H_X = 0, H_{XX} = 0. \quad (2.8.1)$$

Substituting these conditions into the augmented Young-Laplace equation (2.7.3) gives

$$C = C_1 = R\xi^{10} - 1. \quad (2.8.2)$$

Thus, when $C = C_1$, the equilibrium film profile is a uniform film with a unit height.

To find the other equilibrium film solutions we set $H_X = H_{XX} = 0$, which makes (2.7.3) reduce to

$$C = \frac{R\xi^{10}}{H^9} - \frac{1}{H^3}. \quad (2.8.3)$$

As it is proposed by Yi & Wong [2], one of the real solutions to (2.8.3) corresponds to the precursor film thickness. We want to study a drop with a precursor film. Therefore, we set H = 1 and $H_X = 0$ at the center of the drop, which gives

$$K = \frac{1}{2} - \frac{R\xi^{10}}{8} - C. \tag{2.8.4}$$

This together with (2.8.4) leads to

$$C = -\frac{1+H}{2H^2} + \frac{R\xi^{10}(1+H)(1+H^2)(1+H^4)}{8H^8}. \tag{2.8.5}$$

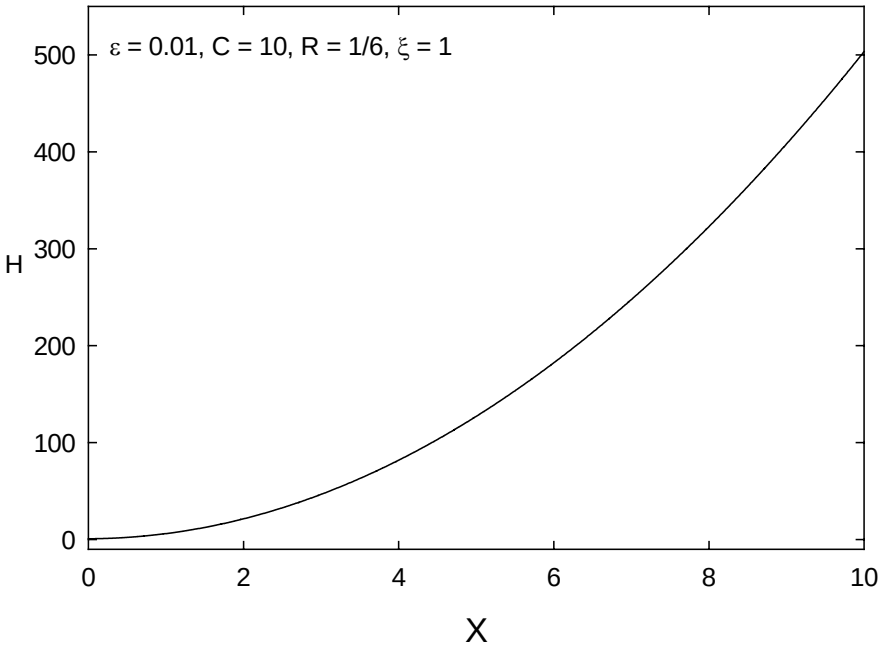
Since (2.8.3) is valid for any equilibrium film profile, we can equate (2.8.3) with (2.8.5) to get the precursor film thickness, $H_p = H_p(R\,\xi^{10})$, which is one of the positive real solutions to the following equation,

$$(1-H)\left\{4H^6(H+2) - R\xi^{10}\left(\sum_{n=0}^{7}(8-n)H^n\right)\right\} = 0. \tag{2.8.6}$$

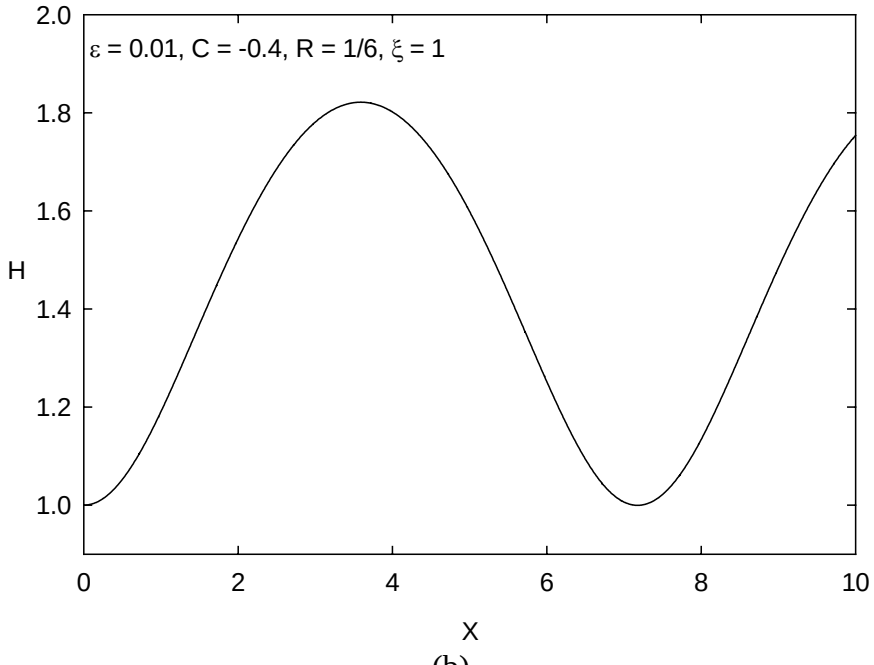
Thus, for a drop with unity height at center and attached with a precursor film, the vapor-liquid pressure difference C is

$$C = C_2 = \frac{R\xi^{10}}{H_p^9} - \frac{1}{H_p^3}. \tag{2.8.7}$$

The pressure difference C is a controlling parameter during experiments. Therefore, we study different equilibrium film profiles by varying C. By setting H = 1 and $H_X = 0$ at X = 0 and using fourth-order Runge-Kutta method to integrate (2.8.3), we find a family of solutions to the equilibrium film profiles (Appendix C.1). There exists five categories: parabolic films, uniform films, periodic drops without uniform film, periodic drops with uniform film, drops attached to an infinite long precursor film, drops with finite precursor film, and pancakes or drops. These film profiles are shown in Fig. 2.3.

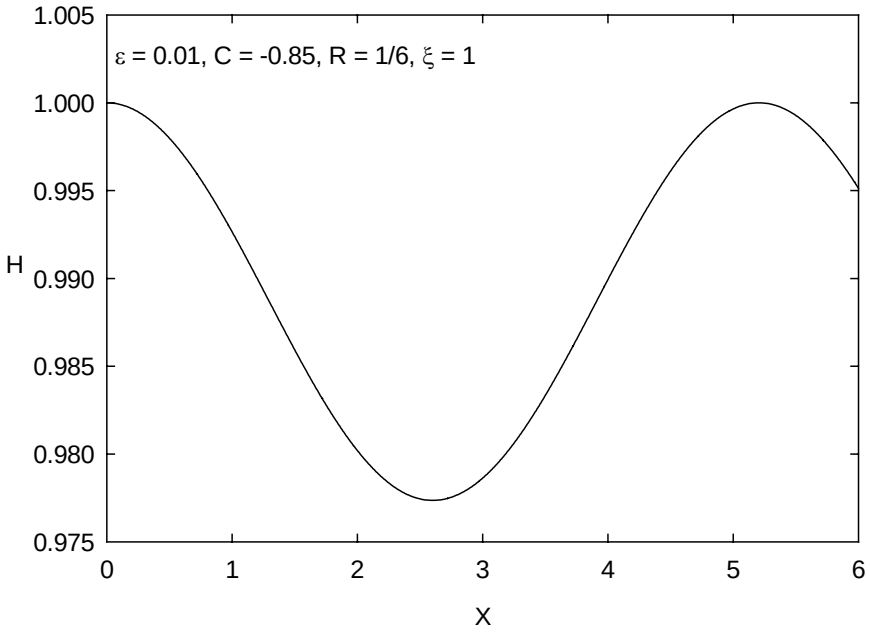


(a)

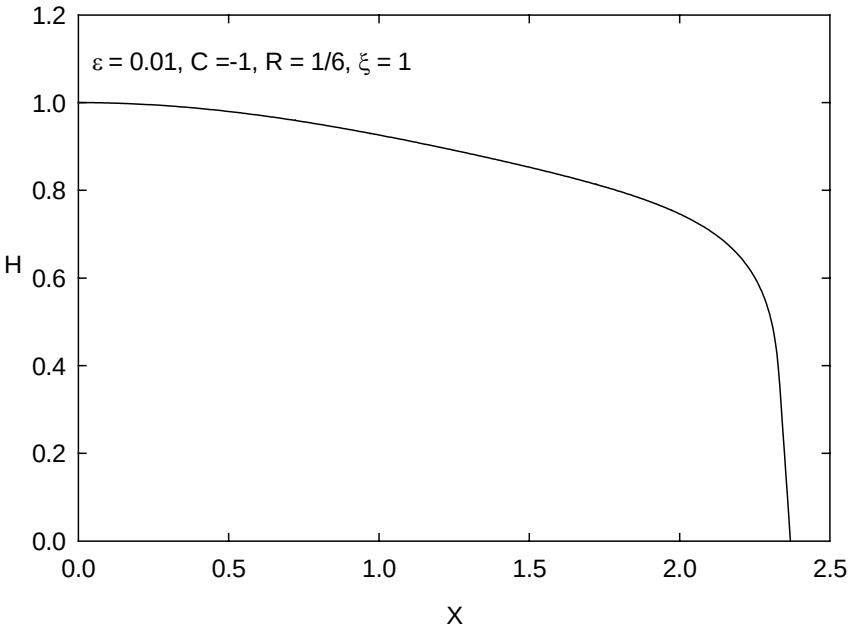


(b)

Figure 2.3 Equilibrium film profiles. 3(a). A parabolic film profile. 3(b). Periodic drops with height greater than 1. 3(c). Periodic drops with height less than 1. 3(d) A pancake-shape film profile. (Figure continued)


ε = 0.01, C = -0.85, R = 1/6, ξ = 1
1.005
1.000
0.995
0.990
0.985
0.980
0.975
H
0
1
2
3
4
5
6
X

(c)


ε = 0.01, C =-1, R = 1/6, ξ = 1
1.2
1.0
0.8
0.6
0.4
0.2
0.0
H
0.0
0.5
1.0
1.5
2.0
2.5
X

(d)

The family of the equilibrium film profiles can be categorized by the characteristic diagram (Fig. 2.4). We find that this diagram is unique for all ε. The characteristic diagram shows that the C and $R\xi^{10}$ values determine the equilibrium film profiles. For a given (C, $R\xi^{10}$), the equilibrium film profile unity height and zero slope at center is unique. And when $C = C_1$, for a uniform film with small perturbation $H = 1 + \delta$ will either grow to a meniscus ($\delta > 0$) or becomes a pancake ($\delta < 0$).

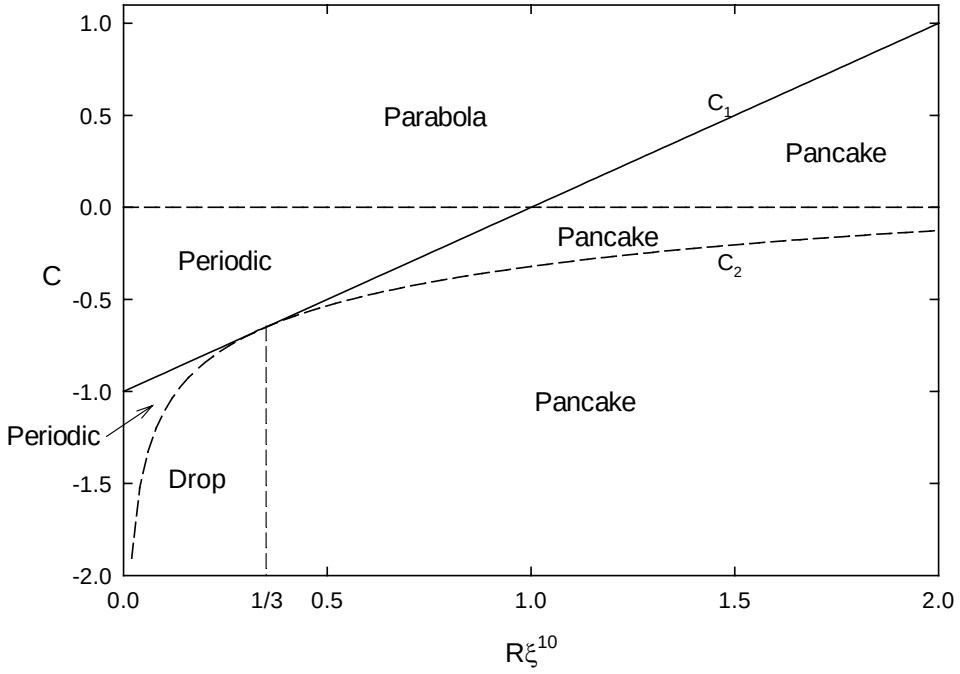


Figure 2.4 Characteristic diagram.

2.9. Linear Stability of Equilibrium Drops

The evolution of a thin film with small slopes obeys [20]

$$\frac{\partial h}{\partial t} - \frac{1}{3\mu}\frac{\partial}{\partial x}\left[h^3 \frac{\partial p_f}{\partial x}\right] = 0, \tag{2.9.1}$$

where t is time and μ is viscosity. The liquid pressure p_f is found by the augmented Young-Laplace equation (2.7.3) as

$$p_f = p_g - \sigma h_{xx} - \Pi, \tag{2.9.2}$$

Because the slope of the film is assumed small, the curvature in the capillary pressure term reduces to h_{xx}.

The evolution equation can be made dimensionless by defining

$$H = \frac{h}{h_0}, X = \frac{\alpha x}{h_0}, \tau = \frac{t\sigma\alpha^4}{\mu h_0}. \tag{2.9.3}$$

Eq. (2.9.1) is transformed into

$$\frac{\partial H}{\partial \tau} + \frac{1}{3}\frac{\partial}{\partial X}\left[H^3 \frac{\partial}{\partial X}\left(\begin{array}{l} H_{XX} - \frac{\varepsilon}{H^3}(1 - H_X^4 + 2HH_X^2 H_{XX}) \\ + \frac{R}{H^9}(\xi^{10} - H_X^{10} + \frac{5}{4}HH_X^8 H_{XX}) \end{array} \right) \right] = 0. \tag{2.9.4}$$

We study the linear stability of equilibrium drops with or without uniform film. We consider a unit cell of the periodic equilibrium profile between the center of a drop $(X = 0)$ and the center of the adjacent precursor film $(X = X_m)$. If we represent the equilibrium film height by $H = H_0(X)$, then the perturbed film height is

$$H(\tau, X) = H_0(X) + \delta(\tau, X). \tag{2.9.5}$$

When this is substituted into the evolution equation (2.9.4) and expanded assuming $|\delta| << H_0$, we arrive at a linear equation in δ, which admits a normal-mode solution as

$$\delta(\tau, X) = e^{\omega\tau} f(X). \tag{2.9.6}$$

This leads to an eigenvalue problem for f. The eigenfunction f obeys periodic conditions: at $X = 0$ and $X = X_m$,

$$\frac{df}{dX} = \frac{d^3 f}{dX^3} = 0 \quad . \tag{2.9.7}$$

However, instead of solving for f, we find it more convenient to solve for

$$F = \int_0^X f dX. \tag{2.9.8}$$

Thus, the eigenvalue problem is transformed into

$$g_4 \frac{d^4F}{dX^4} + g_3 \frac{d^3F}{dX^3} + g_2 \frac{d^2F}{dX^2} + g_1 \frac{dF}{dX} + \omega F = 0, \tag{2.9.9a}$$

Where g_1, g_2, g_3, and g_4 are functions of $H_0(X)$:

$$\begin{aligned} g_4 &= \frac{5RH_{0X}{}^8 - 8\varepsilon H_{0X}{}^2 H_0{}^6 + 4H_0{}^8}{12H_0{}^5}, \\ g_3 &= \frac{1}{3H_0{}^6}\left[H_{0XX}(20RH_0H_{0X}^7 - 8\varepsilon H_{0X}H_0^7) - 20RH_{0X}{}^9 + 8\varepsilon H_{0X}{}^3 H_0{}^6\right] \end{aligned} \tag{2.9.9b}$$

$$g_2 = \frac{1}{3H_0{}^7}\begin{bmatrix} H_{0XXX}(10RH_{0X}^7H_0^2 - 4\varepsilon H_{0X}H_0^8) - 9R\xi^{10} \\ + 3\varepsilon H_0^6 + 70RH_{0X}^6H_{0XX}^2H_0^2 - 15\varepsilon H_{0X}{}^4H_0^6 \\ + 99RH_{0X}{}^{10} - 180RH_{0X}{}^8H_{0XX}H_0 \\ + 24\varepsilon H_{0X}{}^2H_{0XX}H_0{}^7 - 4\varepsilon H_{0XX}{}^2H_0{}^8 \end{bmatrix}, \tag{2.9.9c}$$

$$g_1 = \frac{1}{12H_0{}^8}\begin{bmatrix} H_{0XXX}(12H_0^{10} - 8\varepsilon H_{0X}^2H_0^8 - 25RH_{0X}^8H_0^2) \\ - 12\varepsilon H_0^6H_{0X} + 12\varepsilon H_0^6H_{0X}{}^5 + 252R\xi^{10}H_{0X} \\ - 200RH_{0X}^7H_{0XX}{}^2H_0^2 + 480RH_{0X}^9H_{0XX}H_0 \\ - 16\varepsilon H_{0X}H_{0XX}{}^2H_0{}^8 - 252RH_{0X}{}^{11} \end{bmatrix}. \tag{2.9.9d}$$

This equation is subject to the following boundary conditions: at the drop center $X = 0$,

$$F = 0, \tag{2.9.10a}$$

$$\frac{d^2F}{dX^2} = 0. \tag{2.9.10b}$$

At $X = X_m$,

$$F = 0, \tag{2.9.11a}$$

$$\frac{d^2F}{dX^2} = 0. \qquad (2.9.11b)$$

Eq. (2.9.10a) comes from the definition of F in Eq. (2.9.8). Eq. (2.9.11a) imposes mass conservation. The second derivatives represent the symmetry conditions. Eq. (2.9.9a) is discretized by second-order finite differences. The coefficients g_1, g_2, g_3, and g_4 are evaluated at the discretized points using the numerical solution of the equilibrium film profile $H_0(X)$. The resulting algebraic eigenvalue problem is solved by the DEVCRG subroutine in the IMSL package (Appendix C.2, Appendix C.3).

We try different ε, R, and ξ, and find that the growth rate ω is always real and negative for periodic drops connected with precursor film, whereas, the first mode's ω is always real and positive for periodic drops connected without precursor film. This implies that the periodic drops connected with precursor film are linear stable and those without are linear unstable. For the case $C = C_2 + 10^{-8}$ (two drops connected with a uniform film), the domain size X_m = 2.305 and the first three eigenvalues are $\omega_1 = -0.3052$, $\omega_2 = -3.248$, and $\omega_3 = -6.533$. The corresponding eigenfunctions are plotted in Fig. 2.5. For the case C = -40 (two drops connected without a uniform film), the domain size Xm = 2.230 and the first three eigenvalues are $\omega_1 = 2.048$, $\omega_2 = -1.857$, and $\omega_3 = -12.38$. The corresponding eigenfunctions are plotted in Fig. 2.6.

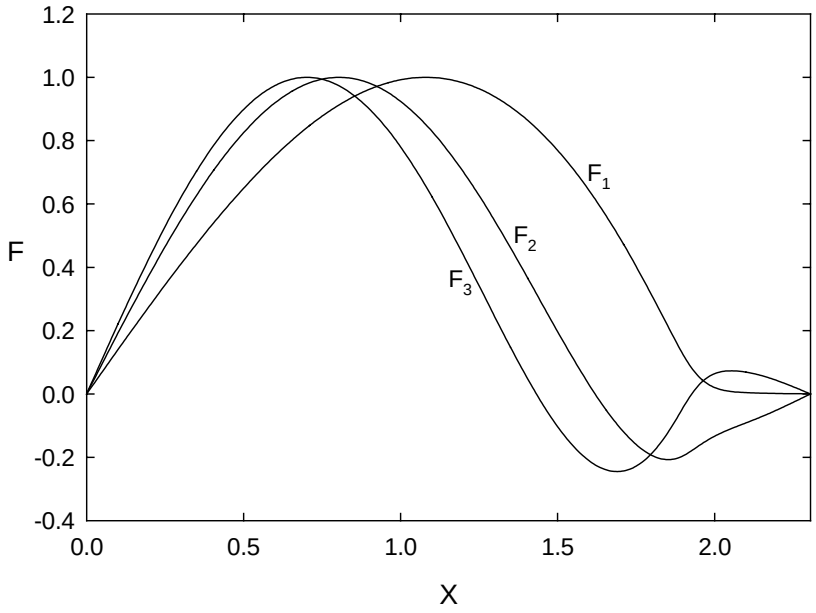


Figure 2.5. The first three stable modes for the perturbation of a drop connected with a precursor film.

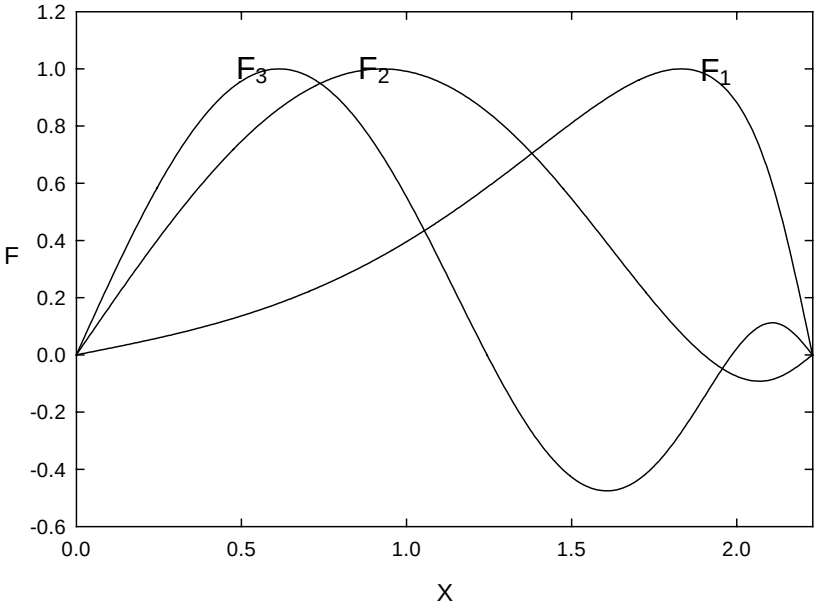


Figure 2.6. The first unstable mode and the first two stable modes for the perturbation of a drop connected without a precursor film.

2.10. Discussion

Since the precursor film thickness H_p depends only on $R\xi^{10}$, it is important to estimate the value of $R\xi^{10}$. Consider another form of the Lennard-Jones potential: [6]

$$\phi(r) = 4w\left[\left(\frac{\rho}{r}\right)^{12} - \left(\frac{\rho}{r}\right)^{6}\right], \tag{2.10.1}$$

where w is the well depth and r is the collision diameter. By comparing with Eq. (2.3.1), we find $L = 4w\rho^{12}$ and $\beta = 4w\rho^{6}$. Following the definition of ε, R, and ξ in Eq. (2.7.2), the ratio is given by

$$R\xi^{10} = \frac{4347(n_f L_{ff} - n_s L_{fs})}{3260(n_f \beta_{ff} - n_s \beta_{fs})h_0^6} = \frac{4347(n_f w_{ff}\rho_{ff}^{12} - n_s w_{sf}\rho_{fs}^{12})}{3260(n_f w_{ff}\rho_{ff}^{6} - n_s w_{sf}\rho_{fs}^{6})h_0^6}. \tag{2.10.2}$$

Thus, H_p depends on both short and long-range forces, and on the macroscopic drop height. If we take $\rho_{ff} = \rho_{fs} = \rho$, then

$$R\xi^{10} = \frac{4347}{3260}\left(\frac{\rho}{h_0}\right)^{6}. \tag{2.10.3}$$

If $h_0 >> \rho$, then $R\xi^{10} << 1$. In the limit $R\xi^{10} \to 0$, the asymptotic expansion of H_p [2] gives

$$h_p \to \left(\frac{4347}{3260}\right)^{1/6}\rho + \frac{1}{24}\left(\frac{4347}{3260}\right)^{1/3}\left(\frac{\rho}{h_0}\right)\rho + \ldots, \tag{2.10.4}$$

Where $h_p = H_p h_0$ is the dimensional precursor film thickness. In this case, h_p depends only on the collision diameter to leading order, but the higher-order terms contain ρ / h_0, i.e., they contain information of the macroscopic drop.

By including the gradient dependence in Π, we are able to regularize Π for non-uniform films near the contact line as $h \to 0$. However, for uniform films, $\Pi = \Pi(h)$ is still singular as $h \to 0$. This question of divergence can also be addressed using our

theory. The uniform film solutions in §8 include a precursor film of thickness h_p. Eq. (2.10.4) shows that to leading order $h_p \sim \rho$, which is the intermolecular spacing or the size of one molecule, i.e., to leading order h_p is independent of the macroscopic length scales. Thus, the uniform film thickness can reach at most the molecular size and no thinner, no matter what you do at the macroscopic level. Thus, a uniform film can never reach zero thickness, and therefore Π will never become unbounded.

Young's equation is recovered at the macroscopic contact line. In the macroscopic region, the film thickness $h >> h_p$. Since $h_p \sim h_0 (R\xi^{10})^{1/6}$, we can study the macroscopic region by setting R = 0 or J = 0. If J = 0, then the boundary conditions at the macroscopic contact line in Eqs. (2.2.7) and (2.6.2) become

$$|h_x| = \alpha. \tag{2.10.5}$$

$$h_{xx} = 0. \tag{2.10.6}$$

$$\frac{\sigma}{(1+h_x^{\,2})^{1/2}} + \sigma_{fs} - \sigma_{sg} + B\alpha h_{xxx} = 0. \tag{2.10.7}$$

Given a drop of height h_0 and slope $|h_x| \sim \alpha$,

$$\frac{B\alpha h_{xxx}}{\sigma} \sim \frac{B\alpha^4}{\sigma h_0^2} = \varepsilon\alpha^2, \tag{2.10.8}$$

where ε defined in (2.7.2) is the ratio of long-range disjoining pressure to capillary pressure. If the drop height h_0 is large such that $\varepsilon << 1$, then Young's equation is recovered at the macroscopic contact line according to (2.10.7). If h_0 decreases to the range where $\varepsilon \sim 1$, then Young's equation still holds at the macroscopic contact line because $\alpha << 1$ in this work. Thus, as long as $h_0 >> h_p$, a macroscopic contact line can be defined and Young's equation is valid at the contact line.

Our curvature-dependent disjoining pressure includes all the curvature effects. It yields different equilibrium film solutions. Based on different $R\xi^{10}$ and C, as shown in Fig. 2.4, the family of equilibrium film solutions are: parabolic meniscus, uniform films (complete wetting), periodic drops connected without uniform film, drops with infinite long precursor film, periodic drops connected without uniform film, drops with finite precursor film (pseudo partial wetting), drops or pancakes (partial wetting). For a small perturbation on a uniform film, when $R\xi^{10} > 1$ and $C = C_1$, if $H = 1 + \delta$, where $\delta << 1$, then the uniform film will grow to a parabolic menisci; however, if $H = 1 - \delta$, then the uniform film will become a pancake.

We apply the mean-field approximation to the Lennard-Jones potential, which consists of a long-range attractive component and a short-range repulsive core. Usually, the mean-field approximation is only applied to the long-range potential with a cut-off distance of the order of molecular spacing. The cut-off distance serves two purposes. It prevents the infinite self-interaction and represents the effect of the repulsive core. In this work, we extend the mean-field approximation to the short-range repulsive potential with a cut-off distance D that is much smaller than the molecular spacing. Mathematically, the procedure is correct because the repulsive component in the Lennard-Jones potential is still a potential. Physically, D can no longer represent the effect of short-range forces. Rather, D has the sole purpose of avoiding the infinite self-interaction. This loss of physical meaning is caused by the extension of a continuum model (disjoining pressure) to capture events at the molecular level (precursor film). (The precursor film is of the order of molecular spacing, as listed in Eq. (2.10.4).) Although we don't have a physical justification for extending the mean-field approximation to the repulsive core, the end results seem to suggest that the procedure is fine, as shown by our modeled drop profiles.

The Helmholtz free energy of a drop on a solid substrate is minimized to yield the curvature-dependent Π. However, once the functional form is derived, the new Π can be applied to other situations. For example, it holds for a liquid film on a solid substrate surrounded not by vapor but by another immiscible liquid. The substrate could be liquid instead of solid. For a freely-suspended liquid film in a fluid, the substrate is replaced by the surrounding fluid. In all the situations, the functional form in (2.5.1a) should be valid.

The disjoining pressure in (2.5.1a) works for two dimensions. It is possible to extend the expression to three dimensions by replacing h_x^2 by $\nabla h \bullet \nabla h$ and h_{xx} by $\nabla^2 h$.

2.11. Conclusions

A new theory of disjoining pressure is formulated and applied to Lennard-Jones liquid films. The expression contains a short-range repulsive term and a long-range attractive term. This disjoining pressure is incorporated into the augmented Young-Laplace equation, which governs equilibrium drop shapes. The governing equation is made dimensionless by the maximum height of the drop and four dimensionless numbers emerge: C, ε, R, and ξ. The parameter C reflects the pressure difference between vapor and liquid, and ε, R, and ξ are related to the intermolecular potential. Hence, C can be varied in an experiment, whereas ε, R, and ξ are fixed for a particular material system. We find a critical value of $C = C_1(R, \xi)$ corresponding to a uniform film, and a $C = C_2(R, \xi)$ corresponding to a drop with a precursor film that extends to infinity. We find that if $C > 0$ and $R\xi^{10} < 1$ or $C > C_1$ and $R\xi^{10} > 1$, then the film height $H \sim X^2$ as $X \to \infty$, where X is a coordinate along the substrate. If $0 > C > C_2$ and $R\xi^{10} < 1/3$ or $0 > C > C_2$ and $1/3 < R\xi^{10} < 1$, then the drop height is a periodic function of X, implying an array of

drops. If $C < C_2$ and $R\xi^{10} < 1/3$, then the precursor film terminates at the substrate. If $C < C_1$ and $R\xi^{10} > 1/3$, then the equilibrium profiles are pancakes. We show that the new Π captures all three regimes of drop behavior: complete wetting, partial wetting, and pseudo partial wetting.

We also study the linear stability of equilibrium film profiles and find that the periodic drops connected without precursor film are unstable, whereas the precursor film, the periodic drops connected by precursor films, and drops or pancakes are unconditionally stable.

CHAPTER 3. SATELLITE DROPLET FORMATION DURING EVAPORATION OF A STEP LIQUID FILM

3.1. Review

The dynamics of evaporating thin liquid films play an important role in many scientific and engineering applications such as tear film drainage [3], solvent coating [5], and thin-film heat transfer. Evolution of an evaporating liquid film is crucial to solvent coating because coating quality highly depends on the evaporation rate of the solvent. Liquid film evaporation is also important to understand thin film heat transfer, which is a fundamental phenomenon in many heat transfer devices.

There has been a lot of interest in evaporating/condensing liquid films. Burelabach, Bankoff, and Davis [41] proposed a one-sided model of heat and mass transfer in analysis of the nonlinear stability of evaporating/condensing liquid films. This model decouples the dynamics of the liquid from that of vapor. Panzarella, Davis, and Bankoff [42] investigated the nonlinear dynamics in horizontal film boiling. They considered buoyancy, capillarity, and evaporative effects on the stability of the film. Braun and Fitt [4] modeled the drainage of the precorneal tear film in humans. The film is on a vertical plate, and is changing under the influence of evaporation, capillarity, and gravity. Spreading of volatile liquid drops also has been studied intensively. Anderson and Davis modeled steady and unsteady spreading and evaporation of liquid drops [43]. They described the effects of capillarity, thermocapillarity, vapor recoil, viscous spreading, contact-angle hysteresis, and mass loss on the behavior of the droplet. Ajaev [44] developed a model for the spreading of a thin volatile liquid droplet on a uniformly heated surface. His model accounts for the effects of surface tension, evaporation, thermocapillarity, gravity, and disjoining pressure [1, 2].

Even though evolution equations including diverse effects have been developed, evolution of evaporating liquid films has not yet been thoroughly studied. The objective of this Chapter is to model satellite droplet formation as a liquid film evaporates. We consider a two-dimensional evaporating thin film on a heated horizontal substrate. The film initially forms a small contact angle with the substrate at one end and extends uniformly to infinity at the other end. In §3.2 we employ the lubrication theory to describe the evolution of film profile. The governing equations are non-dimensionalized in §3.3. In §3.4 we present the numerical method for solving the equaitons and results of the film evolution profiles and satellite droplets profiles. We also show effects of capillarity and evaporation on the evolution of the film and resulting drops. Finally, in §3.5 we summarize this chapter.

3.2. Formulation

We model the evolution of a two-dimensional evaporating thin film on a uniformly heated horizontal substrate. The film initially forms a small contact angle α with the substrate at one end and extends uniformly to infinity at the other end with a film height H. A Cartesian coordinate system (x^*, y^*) is defined as shown in Fig. 3.1. Because the film is thin and the contact angle is small, lubrication theory is employed to describe the evolution of film height h^*.

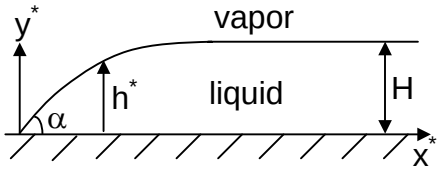


Figure 3.1. Schematic diagram of an evaporating liquid film on a solid horizontal substrate. The liquid-vapor interface is located at $y^* = h^*$. The film contacts the substrate at $x^* = 0$ with a small contact angle α. The film profile begins with a straight line followed by a portion of a circular arc and then extends uniformly with a height H to infinity.

The thermal conductivity, viscosity, and density of the vapor phase are much less than those of the liquid phase. Thus, the dynamics of the vapor can be neglected in comparison to that of the liquid [41]. The liquid film obeys the following set of equations:

$$\nabla^* \cdot \mathbf{u}^* = 0, \tag{3.2.1a}$$

$$\rho\left(\frac{\partial \mathbf{u}^*}{\partial t^*} + \mathbf{u}^* \cdot \nabla^* \mathbf{u}^*\right) = -\nabla^* p^* + \mu \nabla^{*2} \mathbf{u}^*, \tag{3.2.1b}$$

$$\frac{\partial T^*}{\partial t^*} + \mathbf{u}^* \cdot \nabla^* T^* = \beta \nabla^{*2} T^*. \tag{3.2.1c}$$

Here, $\mathbf{u}^* = u^* \mathbf{i} + v^* \mathbf{j}$ is liquid velocity, p^* is pressure, ρ is density, μ is viscosity, T^* is temperature, t^* is time, $\beta = k / \rho C_p$ is the thermal diffusivity (k is the thermal conductivity and C_p is the specific heat), and $\nabla^* = \partial / \partial x^* \mathbf{i} + \partial / \partial y^* \mathbf{j}$ is the two-dimensional spatial gradient operator.

On the substrate where $y^* = 0$, we apply the no-slip boundary condition:

$$u^* = 0, \; v^* = 0, \tag{3.2.2a, b}$$

and impose a prescribed constant temperature

$$T^* = T_\infty. \tag{3.2.2c}$$

Initially, the liquid film is maintained at constant temperature T_v, and the vapor is at the corresponding equilibrium pressure p_v. The substrate temperature is subsequently increased by ΔT to T_∞ and maintained at that level. (In this work, we assume $\Delta T = (T_\infty - T_v) << T_v$.) This induces evaporation at the liquid-vapor interface. Thus, the liquid velocity at the interface y = h:

$$J^* = \rho\left(\mathbf{u}^* - \mathbf{u}_i^*\right) \cdot \mathbf{n}, \tag{3.2.3}$$

where J^* is the evaporative mass flux, $\mathbf{u}_i^* = \mathbf{j}\,\partial h^*/\partial t^*$ is the interfacial velocity, and $\mathbf{n} = \mathbf{i}\,\partial h^*/\partial x^* - \mathbf{j}$ is the unit normal vector, which has been simplified by applying $\partial h^*/\partial x^* << 1$. Energy leaving the interface due to evaporation is balanced with the energy entering the interface due to conduction:

$$J^* h_{fg} + k\nabla^* T^* = 0, \tag{3.2.4}$$

where h_{fg} is the latent heat. In this Chapter, we are interested in the effects of surface tension and evaporation on the film evolution. Therefore, the normal stress balance at the interface gives

$$p_v^* - p^* = \sigma \frac{\partial^2 h^*}{\partial x^{*2}}, \tag{3.2.5}$$

where p_v^* is vapor pressure and σ is surface tension. We neglect surface-tension gradients since the Marangoni number is small. Therefore, the shear stress balance at the interface is

$$\mathbf{t} \cdot T \cdot \mathbf{n} = 0, \tag{3.2.6}$$

where $\mathbf{t} = \mathbf{i} + \mathbf{j}\,\partial h^*/\partial x^*$ is the unit tangent vector, and $\mathbf{T}$ is the stress tensor. We also need the mass constitutive equation to evaluate the evaporative mass flux at the interface. The evaporative mass flux is assumed to be proportional to the deviation of vapor pressure from the equilibrium value [45, 46]:

$$J^* = \lambda\left(p_e^* - p_v^*\right), \tag{3.2.7}$$

where $\lambda = \alpha_0(2\pi R T_\infty)^{-1/2}$ is the evaporative mass transfer coefficient as determined by the kinetic theory, (α_0 is the accommodation coefficient, and R is the specific gas constant), and p_e^* is the equilibrium vapor pressure at the interfacial temperature T_i^*. Because we

consider a bulk liquid interface, p_e^* also depends on the interfacial curvature. It can be obtained from the extended Kelvin equation [47]

$$p_e^* = p_0 \exp\left(-V_L \sigma \kappa^* / R_u T^*\right), \tag{3.2.8}$$

where p_0 is the equilibrium vapor pressure at a flat bulk liquid surface at temperature T_i^*, V_L is the molar volume of the liquid, κ^* is the curvature, and R_u is the universal gas constant. For a 0.1 mm high water film at 20 °C with 10° contact angle, σ = 0.0728 N / m, V_L = 0.018 m^3 / mol, and the curvature $\kappa^* \approx 300$ m^{-1}. These give $V_L\sigma\kappa^*/RT^* \approx 10^{-4}$. Therefore, we can neglect the curvature effect and take

$$p_e^* \approx p_0. \tag{3.2.9}$$

Since p_0 is only a function of temperature T_i^*,which is close to T_v, we can use Taylor's expansion,

$$p_0 = p_v + \frac{dp_0}{dT}(T_i^* - T_v), \tag{3.2.10a}$$

$$\frac{dp_0}{dT} = \frac{\rho_v h_{fg}}{T_v}, \tag{3.2.10b}$$

where ρ_v is the equilibrium vapor density at temperature T_v, and the gradient dp_0 / dT is calculated from the Clapeyron relation, assuming $|T_i^* - T_v| << T_v$ [48]. Thus, to the leading order, the evaporative mass flux is

$$J^* = \lambda\left[\frac{dp_0}{dT}\left(T_i^* - T_v\right)\right], \tag{3.2.11}$$

In (3.2.11), the term represents the contribution from the deviation of the interfacial temperature from T_v.

3.3. Non-dimensionalization of the Governing Equations

We choose the initial uniform film thickness H as the length scale for the film height, as well as the vertical length scale. Thus, the horizontal length scale is H / α, and the temperature scale is the constant substrate temperature difference $\Delta T = T_\infty - T_v$. A set of dimensionless variables is defined:

$$x = \frac{\alpha x^*}{H}, y = \frac{y^*}{H}, h = \frac{h^*}{H}, T = \frac{T^* - T_v}{T_\infty - T_v}, t = \frac{t^*}{\delta t}, u = \frac{u^*}{U}, v = \frac{v^*}{\alpha U}, p - p_v = \frac{\left(p^* - p_v^*\right)H}{\alpha^2 \sigma}, J = \frac{J^*}{J_0}. \tag{3.3.1a}$$

Here, δt is the time scale, U is the characteristic velocity derived from the momentum equation, and J_0 is the scale of evaporative mass flux from the energy balance at the interface:

$$\delta t = \frac{H}{\alpha U}, \quad U = \frac{\alpha^3 \sigma}{\mu}, \quad J_0 = \frac{k\Delta T}{H h_{fg}}. \tag{3.3.2b,c,d}$$

Because the film is thin and the contact angle $\alpha << 1$, to the leading order, the dimensionless governing equations become

$$\nabla \cdot \mathbf{u} = 0, \tag{3.3.3a}$$

$$-\frac{\partial p}{\partial x} + \frac{\partial^2 u}{\partial y^2} = 0, \tag{3.3.3b}$$

$$\frac{\partial p}{\partial y} = 0, \tag{3.3.3c}$$

$$\frac{\partial^2 T}{\partial y^2} = 0. \tag{3.3.3d}$$

At the solid substrate y = 0,

$$u = 0, \tag{3.3.4a}$$

$$v = 0, \tag{3.3.4b}$$

$$T = 1. \tag{3.3.4c}$$

At the interface y = h, mass balance reads

$$\frac{\partial h}{\partial t} + u\frac{\partial h}{\partial x} - v + EJ = 0, \tag{3.3.5a}$$

where E is the evaporation parameter, which is the ratio between the viscous time scale to the evaporation time scale:

$$E = \frac{k\Delta T}{\rho_v h_{fg} H \alpha U}. \tag{3.3.5b}$$

It reflects the effect of evaporation on the evolution of the film. When E = 0, the evaporation has no effect, and when E increases, that effect becomes strong. The energy balance becomes

$$J = -\frac{\partial T}{\partial y}. \tag{3.3.6}$$

The normal stress and shear stress balance are reduced to

$$p_v - p = \frac{\partial^2 h}{\partial x^2}, \tag{3.3.7}$$

$$\frac{\partial u}{\partial y} = 0. \tag{3.3.8}$$

And the mass constitutive equation relates the temperature to the evaporative mass flux as

$$KJ = T, \tag{3.3.9a}$$

where

$$K = \left(\frac{\lambda}{J_0}\frac{dp_0}{dT}\Delta T\right)^{-1}. \tag{3.3.9b}$$

For a water film with height 1 mm at 20 °C, taking $\alpha_0 = 0.03$, $h_{fg} \approx 2454.1$ kJ /kg, k = 0.6 W / mk, we calculate $\lambda \approx 2 \times 10^{-3}$ s / m. Thus, assume $\Delta T = 10$ °C, $T_v = 300$ k, $\rho_v = 0.0173$ kg / m^3, we get $K \approx 10^{-3}$.

The velocity field is resolved by solving equations (3.3.3a), (3.3.3b) and (3.3.3c) by applying the boundary conditions (3.3.4a), (3.3.4b), and (3.3.8). At the interface

$$u = \frac{1}{2}h^2 \frac{\partial^3 h}{\partial x^3}, \quad v = -\frac{1}{3}h^3 \frac{\partial^4 h}{\partial x^4} - \frac{1}{2}h^2 \frac{\partial h}{\partial x}\frac{\partial^3 h}{\partial x^3}. \tag{3.3.10a,b}$$

And the temperature field is determined from equation (3.3.3d) and boundary condition (3.3.4c) and the mass constitutive equation (3.3.9a):

$$T = 1 - \frac{1}{K + h}y. \tag{3.3.10c}$$

Thus, from energy balance, the evaporative mass flux at y = h is

$$J = \frac{1}{K + h}. \tag{3.3.10d}$$

Therefore, apply y = h, from equation (3.3.5a) we can get the evolution equation for the evaporating liquid film:

$$\frac{\partial h}{\partial t} + \frac{1}{3}\frac{\partial}{\partial x}\left(h^3 \frac{\partial^3 h}{\partial x^3}\right) + \frac{E}{K + h} = 0. \tag{3.3.11}$$

This indicates that the evolution of the interface is due to both the capillary pressure represented by the second term in the above equation and the evaporation described by the third term.

3.4. Numerical Method and Results

The initial film profile is a straight line with slope unity at x = 0, followed by a circular arc, and then extends horizontally to x = L, where L is the size of the

computational domain and is chosen to be large enough to ensure that the film is uniformly distributed at x = L. Therefore, the boundary conditions at $x = x_0$, where x_0 is a function of time and $x_0(0) = 0$, are

$$h = 0, \quad \frac{\partial^2 h}{\partial x^2} = 0. \tag{3.4.1a,b}$$

At x = L the symmetric boundary conditions are imposed:

$$\frac{\partial h}{\partial x} = 0, \quad \frac{\partial^3 h}{\partial x^3} = 0. \tag{3.4.2a,b}$$

We solve (3.3.11), (3.4.1), and (3.4.2) using Crank-Nicolson Scheme and Newton-Raphson iteration (Appendix C.5). We vary space step from 0.01 to 0.0005 and time step from 10^{-4} to 10^{-6} and find the result is accurate to four significant digits.

Fig. 3.2(a) shows snapshots of the film profiles at different times with the evaporation parameter E = 1, and K = 1. The film height keeps decreasing and the film retracts due to the evaporation and a satellite drop is formed at t = 1.4916. The remaining step film continues to evolve. Fig. 3.2(b) shows the evolution of that remaining film. Similarly, it forms one drop, and the remaining step film keeps evolving. Fig. 2(c) shows the film evolution of the step film resulting from (b). It forms a even smaller drop.

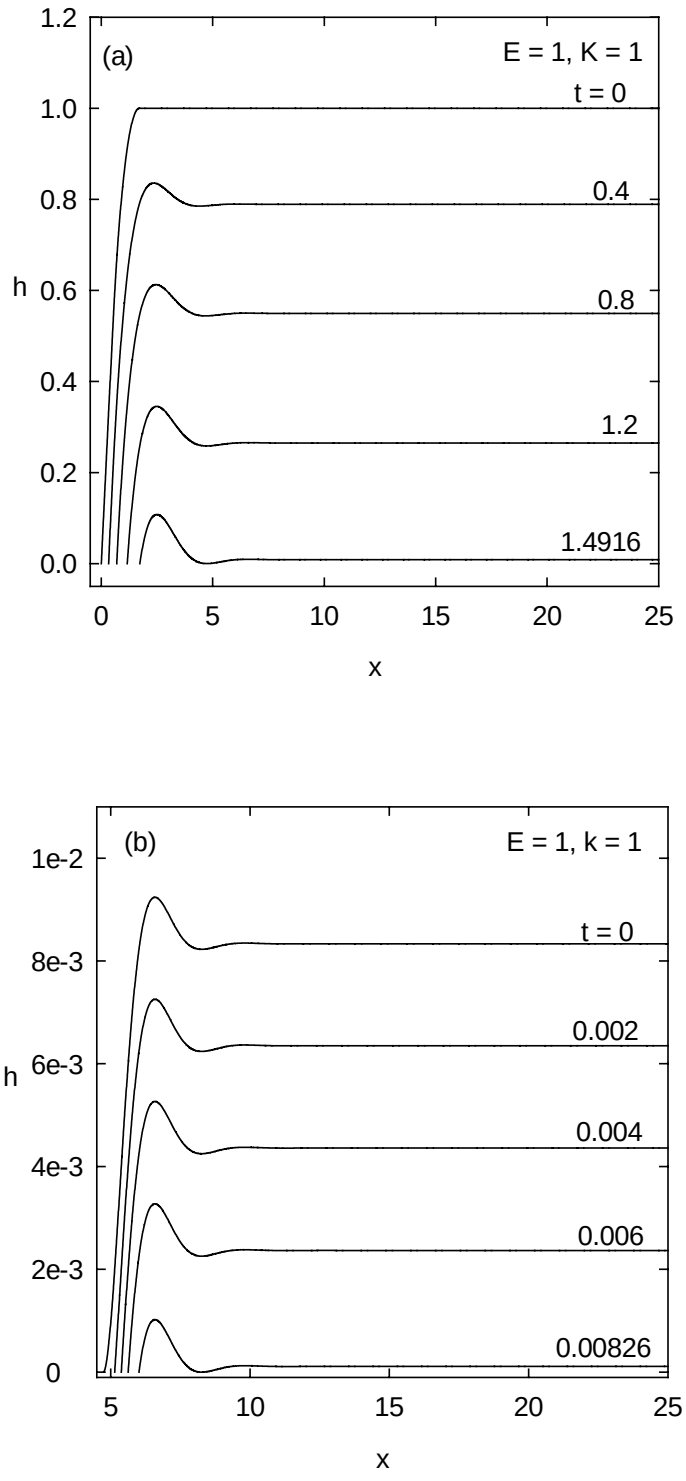


Figure 3.2. Snapshots of film profiles. (a) the first step. (b) the second step. (c) the third step. (Figure continued)

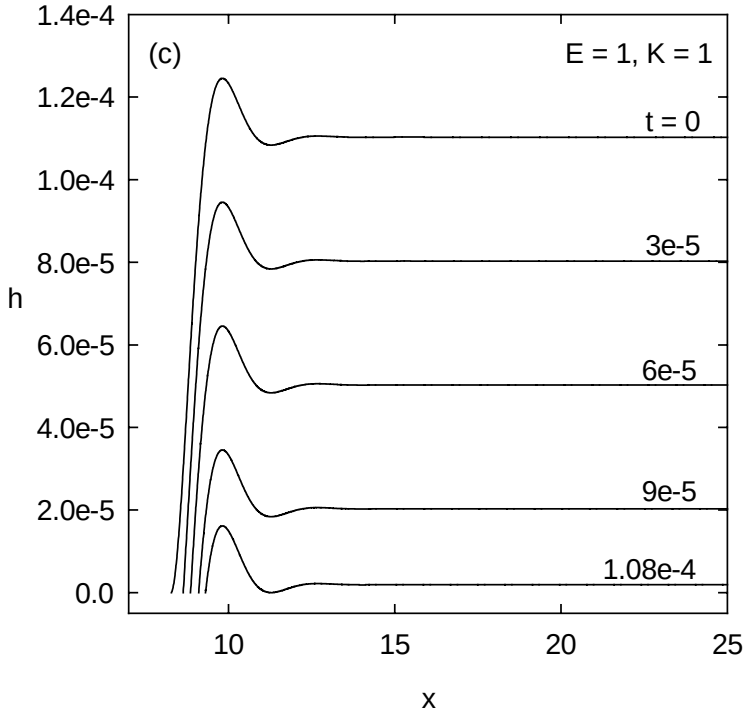
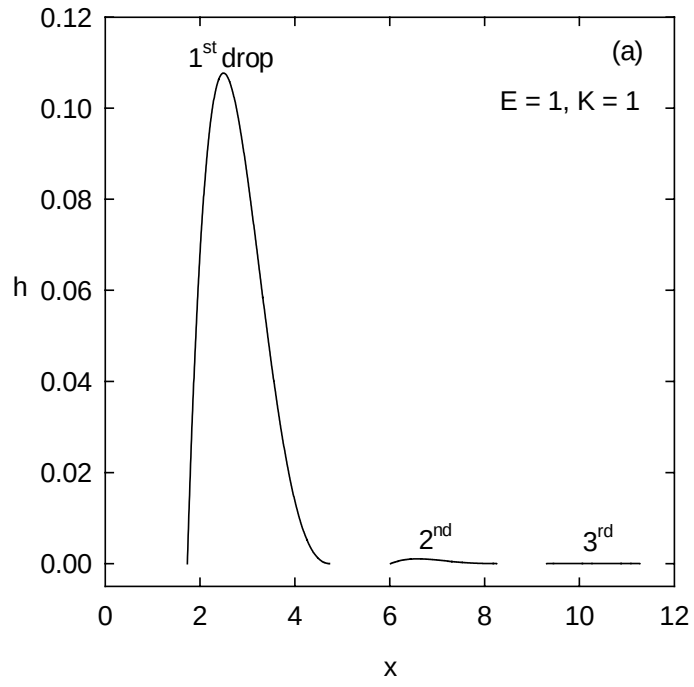


Fig. 3.3(a) shows the three satellite droplets formed from the retracting evaporating liquid film. We can see that the droplet's size is decreasing. The relative size of the second and the third drops are shown in Fig. 3.3(b). And the magnitude of the height of the third drop can be seen clearly in Fig. 3.3(c). The interesting phenomenon is that if we resale the width and height of the second and the third drops according to the first one, then all three drops have exactly same profiles. This is shown in Fig. 3.3(d). Fig. 3.3(e) shows the same self-similar drop profiles for E = 10 and K = 10.



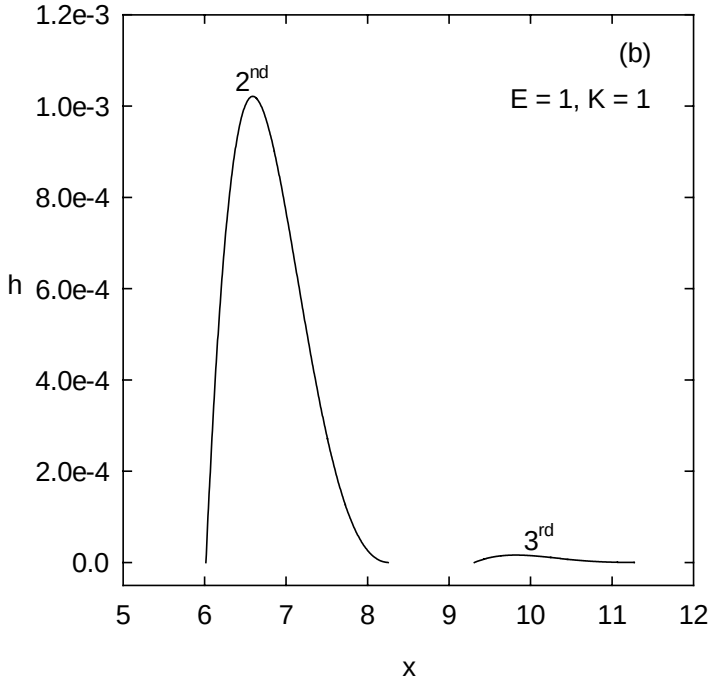
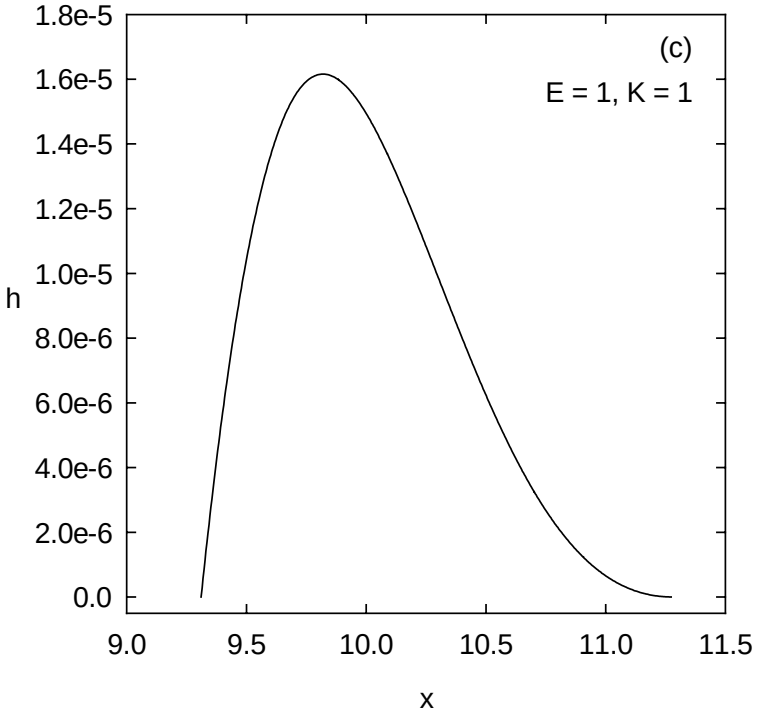
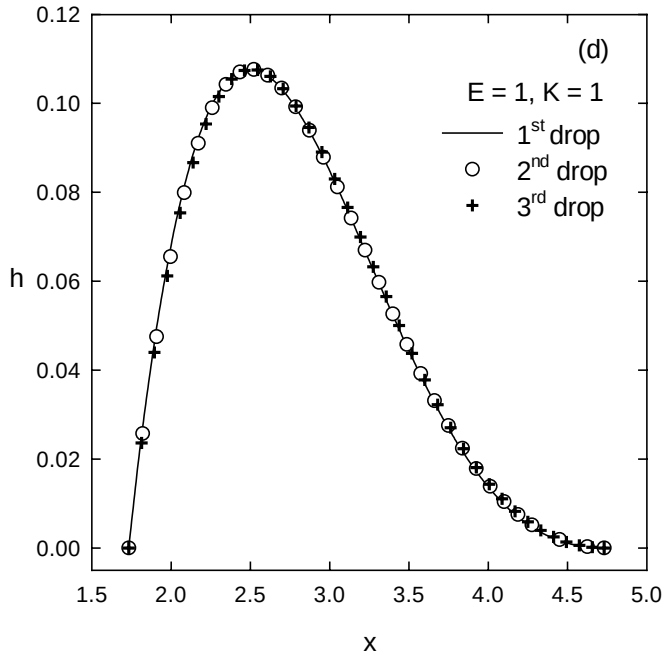
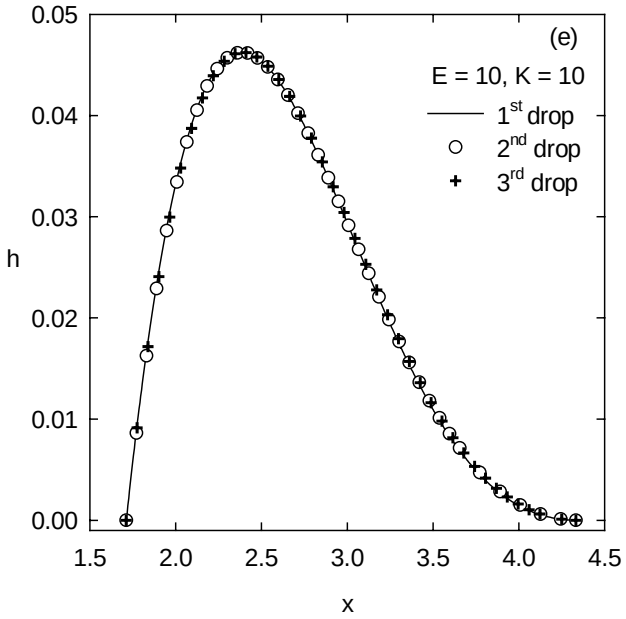


Figure 3.3. Satellite droplets from the retracting evaporating liquid film. (a) Three satellite droplets, E = 1, K =1. (b) The second and the third drops, E = 1, K = 1. (c) The third drop, E = 1, K =1. (d) Rescaled three satellite droplets, E = 1, K = 1. (e) Rescaled three satellite droplets, E = 10, K = 10. (Figure continued)





(Figure continued)



We also did some parametric studies to investigate the effect of the evaporation parameters. When we increase E, the evaporation becomes stronger. Thus, the remaining drop is much smaller than that when E is small. This is demonstrated in Fig. 3.4, in which the evaporation parameter K = 0.01 and the drop is formed from the initial film profile.

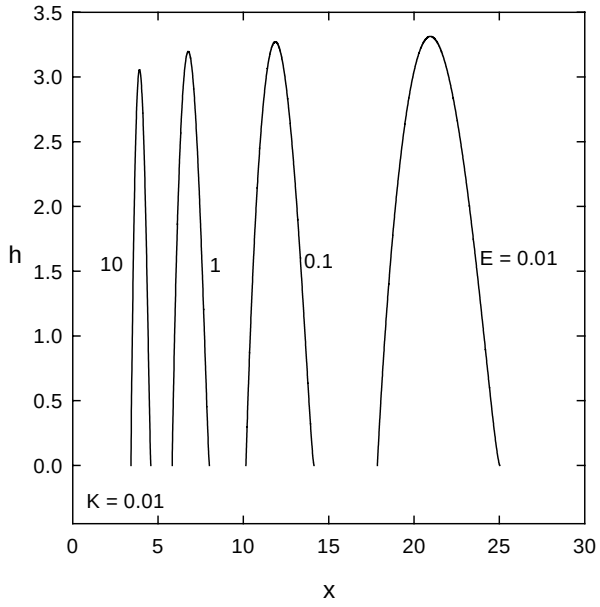


Figure 3.4. Profiles for the first drop with different E.

3.5. Discussion

The evolution of the film is subject to both the capillary and the evaporation effects. While the capillary force is important to the shape of the film profile, the evaporation has both effects on the shape of the film profile and the thickness of the film. The snapshots of film profiles at different time suggest that within a short period of time, the liquid-vapor interface will rearrange to form a certain shape. Then the shape of the film profile is frozen whereas its thickness decreases uniformly.

The similar satellite droplets profiles indicate the evolution equation has a self-similar solution. Since the evaporation term is only a function of film thickness h but not x, it cannot be a key role in the self-similar behavior. Therefore, only the capillary effect is important to that self-similar behavior.

3.6. Conclusions

We have simulated the formation of satellite droplets resulting from a retracting evaporating liquid film. The liquid film contacts a heated solid substrate with a small contact angle α at one end and extends uniformly at the other end. We study the effects of capillary force and evaporation on the evolution of the evaporating film. Applying lubrication theory and one-sided evaporation model, we derive the evolution equation governing the behavior of an evaporating liquid thin film. We solve the evolution equation by a finite-difference method using the Crank-Nicolson scheme, and treat the nonlinearity by the Newton-Raphson method. It is found that the film shape is determined by the capillary force. The satellite drop volume depends on the evaporation parameters. The series of satellite drops have similar profiles. This suggests the evolution equation has a self-similar solution.

CHAPTER 4. SELF-SIMILAR RETRACTION OF A STEP LIQUID FILM PINNED AT THE CONTACT LINE

4.1. Review

Liquid films are important in biological and industrial processes. The evolution of liquid films is usually analyzed using the lubrication approximation because the film slope is small. This yields

$$\frac{\partial h}{\partial t} + \frac{\sigma}{3\mu}\frac{\partial}{\partial x}\left(h^3 \frac{\partial^3 h}{\partial x^3}\right) = 0, \tag{4.1.1}$$

where h is film thickness, t is time, x is a coordinate along the solid substrate, σ is surface tension, and μ is viscosity [20]. The capillary pressure is the driving force and is balanced by the viscous shear stress in the liquid film to arrive at (4.1.1). Also, the film surface is assumed to experience zero shear stress. Other effects can be added to (4.1.1), including gravity, surface tension gradients due to temperature variation and surfactant distribution, intermolecular forces (disjoining pressure), evaporation and condensation, etc. [20].

Self-similar solutions to (4.1.1) have been found in cases where there is no imposed time or length scale. The film thickness is made dimensionless by time, and the non-dimensionalized thickness depends only on a self-similar variable, which is the x coordinate normalized by time. This reduces the partial differential equation (4.1.1) into an ordinary differential equation.

We study the evolution of a uniform film that terminates at a contact line. The other end of the film extends to infinite. We assume that the contact line is pinned. This pinning can result from a change in the surface wetting property of the solid substrate [49]. Given an initial profile, the film will evolve to reduce the surface energy or area. Since the uniform film thickness imposes a length scale, conventional wisdom suggests

that no self-similar solution can exist. However, we find a self-similar solution in which the x-coordinate is scaled with t, as described in §4.2. We also simulate numerically the evolution of a pinned step film and find that the film profile always approaches the self-similar solution as $t \to \infty$ (§4.3). In §4.4, we study the linear stability of the self-similar solution and find that the self-similar solution is stable. We discuss the implications and applications of this solution in §4.5 and conclude in §4.6.

4.2. Self-Similar Solution

We consider a step film ending at a solid substrate at one end and extending uniformly with height h_0 to infinity at the other end (Fig. 4.1). The contact line is pinned and a horizontal coordinate x originates from the contact line. The film thickness h is made dimensionless by h_0, and we define a self-similar variable η:

$$H_s(\eta) = \frac{h(x,t)}{h_0},\ \eta = \frac{x}{h_0^{3/4}}\left(\frac{\mu}{\sigma t}\right)^{1/4}. \qquad (4.2.1a,b)$$

Then (4.1.1) is reduced to

$$\frac{\partial}{\partial \eta}\left(H_s^3 \frac{\partial^3 H_s}{\partial \eta^3}\right) - \frac{3}{4}\eta \frac{\partial H_s}{\partial \eta} = 0. \qquad (4.2.2)$$

This equation is subject to the following boundary conditions. At the contact point where $\eta = 0$,

$$H_s = 0. \qquad (4.2.3a)$$

Far from the contact line as $\eta \to \infty$,

$$H_s \to 1. \qquad (4.2.3b)$$

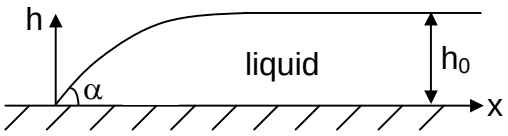


Figure 4.1. A step liquid film on a solid substrate with contact angle α, and extends uniformly to infinity at the other end. The film height h is normalized by the uniform film height h_0. A Cartesian coordinate system is defined with the origin at the contact point.

We solve the self-similar equation (4.2.2) numerically by a shooting method. A local solution is needed to start the integration because the equation is singular at $\eta = 0$. We assume that as $\eta \to 0$,

$$H_s = C_1\eta + f(\eta), \tag{4.2.4}$$

where C_1 is a constant and $f(\eta) << C_1\eta$ as $\eta \to 0$. Substitution of (4.2.4) into (4.2.2) and integrating repeatedly finds $f(\eta)$. Thus, as $\eta \to 0$,

$$H_s \to C_1\eta + \frac{3}{16C_1^2}\eta^2 \ln\eta + C_2\eta^2 + \ldots, \tag{4.2.5}$$

where C_2 is an integration constant. This is used to start the numerical integration at $\eta = \Delta\eta$ by a fourth-order Runge-Kutta scheme, where $\Delta\eta$ is the step size of the integration. We vary C_1 and C_2 until the film height $H_s \to 1$ as $\eta \to \infty$. This asymptotic condition is imposed at a finite distance $\eta = \eta_\infty$ (Appendix C.5). We find $C_1 = 1.4808972$ and $C_2 = -0.53943206$, and the film profile $H_s(\eta)$ in Fig. 4.2. The integrated film profile is accurate to four significant digits as verified by changing $\Delta\eta$ from 0.001 to 0.01 and η_∞ from 15 to 50.

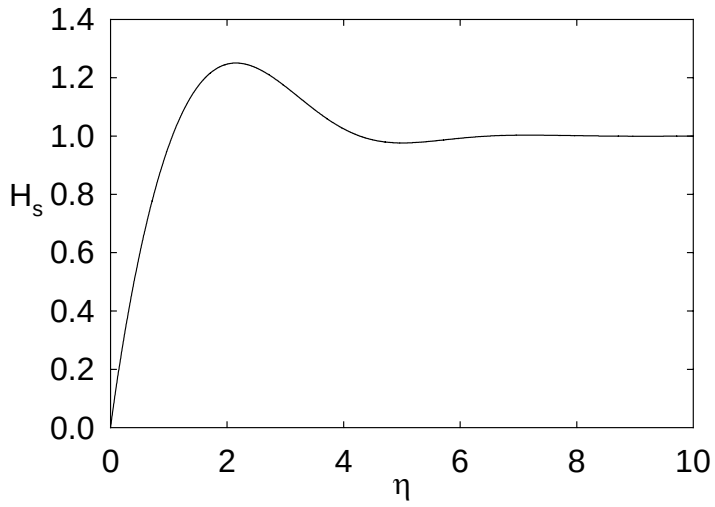


Figure 4.2. Self-similar film profile

Fig. 4.2 shows that the self-similar film profile oscillates with decreasing amplitude as $\eta \to \infty$. A far-field asymptotic solution is found by expanding $H_s \to 1 + \varepsilon(\eta)$ as $\eta \to \infty$, where $\varepsilon << 1$. To the leading order, (4.2.2) gives

$$\frac{\partial^4 \varepsilon}{\partial \eta^4} - \frac{3}{4}\eta \frac{\partial \varepsilon}{\partial \eta} = 0. \tag{4.2.6}$$

The asymptotic solution in the limit $\eta \to \infty$ is obtained following the standard procedure [50]:

$$\varepsilon \sim \eta^{-2/3} \exp\left[(1)^{1/3} \left(\frac{3\eta}{4} \right)^{4/3} \right]. \tag{4.2.7}$$

This yields two decaying solutions and one growing solution. Eqt (4.2.6) also admits ε = constant. The constant and exponential growing solutions are eliminated, and we are left with

$$\varepsilon \sim \eta^{-2/3} \exp\left[\left(-\frac{1}{2} \right) \left(\frac{3\eta}{4} \right)^{4/3} \right] \sin\left[\frac{\sqrt{3}}{2} \left(\frac{3\eta}{4} \right)^{4/3} \right]. \tag{4.2.8}$$

Thus, as $\eta \to \infty$, the film oscillates about the unit height with decreasing wavelength and amplitude. This asymptotic solution can be used to start the numerical shooting method backward from the uniform end, and we recover the self-similar profile in Fig. 4.2.

4.3. Numerical Solution of Evolving Films

The evolution equation (4.1.1) can be non-dimensionalized by defining

$$H = \frac{h}{h_0}, X = \frac{x}{h_0}, \tau = \frac{\sigma t}{\mu h_0}, \tag{4.3.1}$$

$$\frac{\partial H}{\partial \tau} + \frac{1}{3}\frac{\partial}{\partial X}\left(H^3 \frac{\partial^3 H}{\partial X^3} \right) = 0. \tag{4.3.2}$$

This equation obeys the following boundary conditions. At the contact line where X = 0, the film thickness is zero:

$$H = 0, \tag{4.3.3}$$

As $X \to 0$, the second derivative is

$$\frac{\partial^2 H}{\partial X^2} = -\frac{3}{2\alpha^3}\left(\frac{d\alpha}{d\tau} \right) \ln X, \tag{4.3.4}$$

where $\alpha = \alpha(\tau)$ is the instantaneous contact angle. This boundary condition can be derived from (4.3.2) by recognizing that the film surface is pinned at the contact line. The contact angle α can change with time. Near the contact line, the film surface must be planar because of surface tension. Thus, as a first approximation, $H \approx \alpha X$ and $\partial H/\partial \tau \approx (d\alpha/d\tau)X$ as $X \to 0$. When this is substituted into (4.3.2), we get (4.3.4) after two integrations. An integration constant has been omitted in (4.3.4) because it is much smaller than lnX as $X \to 0$. Eq. (4.3.4) agrees with the opening-wedge solution by

Moffatt [51]. At the uniform end of the film, symmetric boundary conditions are applied at X = L >> 1:

$$\frac{\partial H}{\partial X} = 0, \qquad \frac{\partial^3 H}{\partial X^3} = 0. \tag{4.3.5a, b}$$

Equations (4.3.2) – (4.3.5) are solved numerically using the Crank-Nicolson scheme. The nonlinearity is treated by the Newton-Raphson iteration. The equations are discretized by central differences (Appendix C.6). We use the shooting solution (Fig. 4.2) with extended domain size L = 50 as the initial film profile, and apply the boundary condition (4.3.5b) at X = ΔX. We vary ΔX from 0.001 to 0.01, Δt from 10^{-5} to 10^{-6}, and the result is accurate to 4 significant digits. Fig.4.3(a) plots the snapshots of self-similar film evolution profiles at time τ = 0, 60, and 380 (solid lines from left to right). It also plots the evolution of film with initial height less than the self-similar film height (dashed lines from left to right). Fig. 4.3(b) plots the same evolution of self-similar film as that in Fig. 4.3(a) as well as the film evolution profiles with lower initial height (dashedlines from left to right). It shows that for a lower initial height, the film height will increase, and for a higher initial film height, the film height will decrease. And finally, they will approach the self-similar solution.

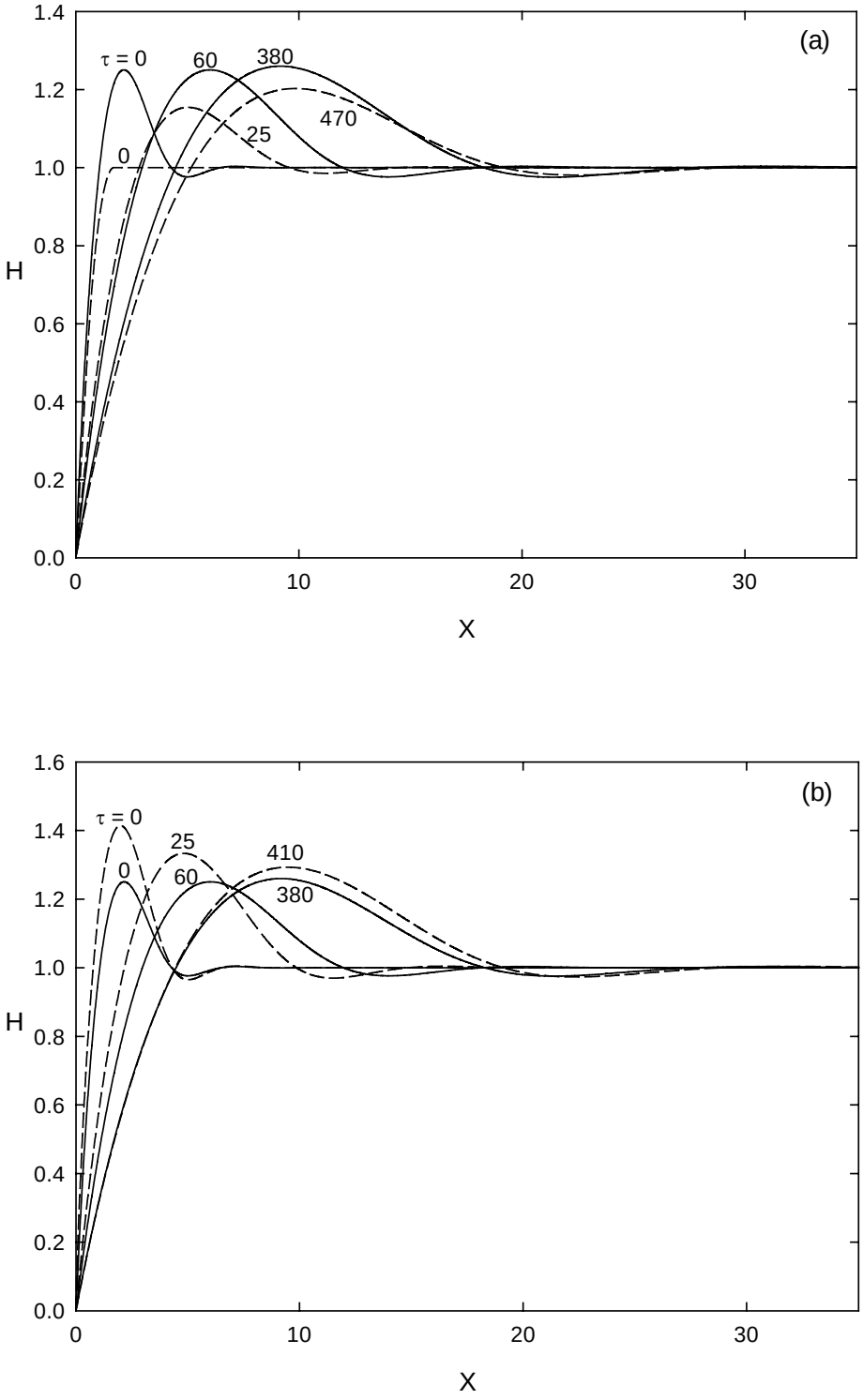


Figure 4.3. Snapshots of film profiles. (a) evolution of self-similar film H_s and film with initial height $H > H_s$. (b) evolution of H_s and film with initial height less than $H < H_s$.

We find X_M, which is the x position of the point with maximum film height, at different time and plot X_M varies t in Fig. 4.4. It shows that $X_M \sim t^{1/4}$. Thus, the time-dependent numerical solution is then rescaled by (4.2.1) & (4.3.1) and compared with the time-independent shooting solution presented in §4.2. We find that they agree well with each other.

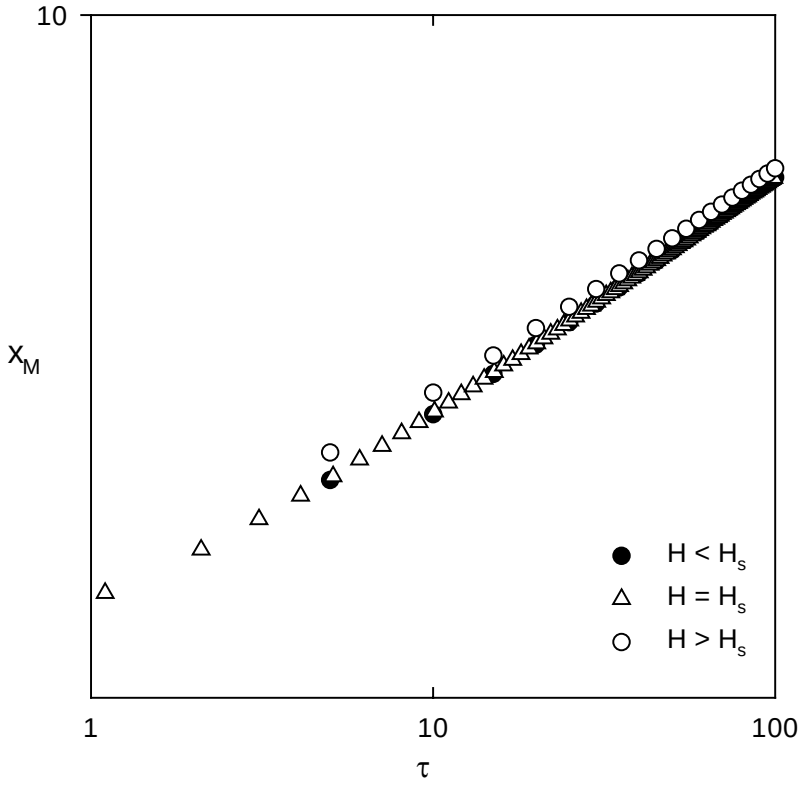


Figure 4.4. The change of the x position of the point with maximum film height with time. The dots are for $H < H_s$ (slope is 0.2451), the triangle is for $H = H_s$ (slope is 0.2503), and the circle is for $H > H_s$ (slope is 0.2589).

4.4. Linear Stability of the Self-Similar Solution

If we assume $H = H(\tau, \eta)$, then (4.1.1) becomes

$$\tau \frac{\partial H}{\partial \tau} + \frac{\partial}{\partial \eta}\left(H^3 \frac{\partial^3 H}{\partial \eta^3} \right) - \frac{3}{4}\eta \frac{\partial H}{\partial \eta} = 0. \qquad (4.4.1)$$

Thus the stability of the self-similar solution for $H_s(\eta)$ can be determined by applying a perturbation:

$$H(\tau,\eta) = H_s(\eta) + \delta(\tau,\eta), \tag{4.4.2}$$

where $|\delta| << H_s$. Substitution into (4.3.1) and keeping only the linear terms in δ yields

$$\tau\frac{\partial\delta}{\partial\tau} + \frac{\partial}{\partial\eta}\left(H_s^3\frac{\partial^3\delta}{\partial\eta^3} + 3H_s^2\frac{d^3H_s}{d\eta^3}\delta\right) - \frac{3}{4}\eta\frac{\partial\delta}{\partial\eta} = 0. \tag{4.4.3}$$

The disturbance is taken to be of the form

$$\delta(\tau,\eta) = \tau^{\omega} f(\eta), \tag{4.4.4}$$

where ω is the growth rate. If ω is negative, then δ decays in time, and the self-similar solution is stable. This form of δ differs from the usual normal-mode analysis, but it eliminates τ from (4.4.1) and creates an eigenvalue problem:

$$\frac{d}{d\eta}\left(H_s^3\frac{d^3f}{d\eta^3} + 3H_s^2\frac{d^3H_s}{d\eta^3}f\right) - \frac{3}{4}\eta\frac{df}{d\eta} + \omega f = 0. \tag{4.4.5}$$

The eigenfunction obeys the following boundary conditions. At $\eta = 0$, the contact line is pinned:

$$f = 0. \tag{4.4.6a}$$

Far from the contact line, the film is not disturbed. As $\eta \to \infty$,

$$f \to 0. \tag{4.4.7b}$$

The coefficient H_s in (4.4.5) is evaluated at the discretized points using the numerical solution of the self-similar film profile $H_s(\eta)$. The resulting algebraic eigenvalue problem is then solved numerically by a finite-difference method and a FORTRAN subroutine DEVCRG in the IMSL package (Appendix C.7). It is found that all the real parts of the eigenvalues are negative. The domain size η_∞ varies from 15.00 to 25.00 and the step size

$\Delta\eta$ varies from 0.01 to 0.02. The result is accurate to four significant digits. The first three eigenvalues are ω_1 = -0.7499, ω_2 = -3.0251, and ω_3 = -4.4176. The corresponding eigen-functions are plotted in Fig. 4.5. They have the usual characteristics: the higher modes have more oscillations.

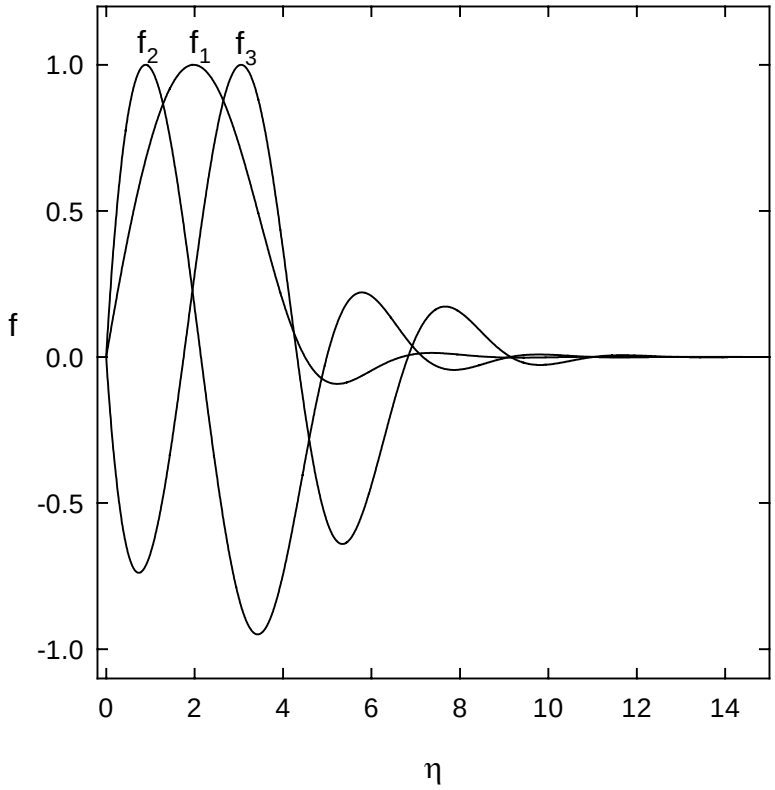


Figure 4.5. The first three stable modes for the perturbation of the self-similar film evolution solution

4.5. Discussion

We find a self-similar solution of film retraction in which the film thickness stays constant, but the film profile spreads with time as $t^{1/4}$. Any initial film profile will eventually approach this self-similar solution. Thus, this solution describes the transfer of information from the contact line towards the uniform film. As such, this self-similar solution behaves like a wave that travels with nonuniform speed. This wave decelerates

with time. Given the universal nature of this spreading solution, it should describe the initial spreading of any initial film profile, not just for a pinned film studied here, but for all films that end at the substrate.

4.6. Conclusions

This work describes the retraction of a step liquid film that is pinned at the contact line and extends uniformly far from the contact line. We find a self-similar solution for the retraction in which the film thickness stays constant, but the film profile spreads with time as $t^{1/4}$. We also simulate numerically the retraction of different film profiles and find that they all approach the self-similar solution as $t \to \infty$. We also study the linear stability of the self-similar solution and find it stable.

CHAPTER 5. CONCLUSIONS

This work presents a theory of curvature-dependent disjoining pressure $\Pi = \Pi(h, h_x, h_{xx})$. The Helmholtz free energy of a drop on a substrate is minimized to yield three equilibrium conditions: one for the drop profile, and two at the drop boundaries. An excess interaction energy $E = E(h, h_x, h_{xx})$ that exists due to thin-film forces. The excess energy E is found by summing up intermolecular potential between a molecule in a liquid drop in the shape of a part circle and other molecules in vapor, liquid, and solid domain. The disjoining pressure is then derived from its relation to E. The new Π thus ensures all the curvature effects have been included. We apply this Π to the augmented Young-Laplace equation. This leads to a characteristic diagram that depicts all the equilibrium film profiles based on material dimensionless parameters $R\xi$[10] and dimensionless pressure difference C. The family of film solutions includes pancakes, periodic drops connected with or without precursor films, a uniform film, and meniscus. By applying linear stability analysis on the equilibrium film profiles, we find that drops connected with precursor films are stable, whereas drops connected without precursor films are unstable.

We also study the satellite droplets formation during the evaporation of a step liquid film. Capillary and evaporation effects are examined. We find that the film profiles are determined by the capillarity, and the evaporation only decrease the height of the film. We find the resulting drops have similar film profiles. The droplet's volume depends on the evaporation parameters.

We investigate the reasons for the similar satellite droplets formed during the evaporation of a step liquid film. A self-similar solution to the evolution equation is found. We also find that the self-similar solution is stable.

REFERENCES

1. Wu, Q. and H. Wong, *A slope-dependent disjoining pressure for non-zero contact angles.* J. Fluid Mech., 2004. **506**: p. 157-185.

2. Yi, T.a.H.W., *Theory of Slope-Dependent Disjoining Pressure with Application to Lennard-Jones Liquid Films.* Physical Review E, 2006.

3. Brown, S.I. and D.G. Dervichian, *Hydrodynamics of blinking.* Arch Ophthal, 1969. **82**: p. 541-547.

4. Braun, R.J., Fitt, A. D., *Modeling drainage of the precorneal tear film after a blink* Mathematical Medicine and Biology 2003. **20**: p. 1.

5. Brown, R.G.W., *Thin-Film Coatings for Optoelectronic Devices*, in *Thin Films for Optical Systems*, F.R. Flory, Editor. 1995, Marcel Dekker: New York. p. 551-574.

6. Israelachvili, J., *Intermolecular and Surface Forces*. Second Edition ed. 2002, San Diego: Academic Press.

7. de Gennes, P.G., F. Brochard-Wyart, and D. Quere, *Capillarity and Wetting Phenomena: Drops, Bubbles, Pearls, Waves*. 2003, New York: Springer.

8. Butt, H.J., K. Graf, and M. Kappl, *Physics and Chemistry of Interfaces*. 2003, Berlin: Wiley-VCH.

9. Derjaguin, B., et al., *Examination of thin layers on various solid substrates*, in *Wetting, Spreading and Adhesion*, J.F. Padday, Editor. 1978, Academic Press: London. p. 201.

10. Ivanov, I.B., *Thin Liquid Films: Fundamentals and Applications*. 1988, New York: Marcel Dekker.

11. Taylor, S.D., J. Czarnecki, and J. Masliyah, *Disjoining pressure isotherms of water-in-bitumen emusion films.* J. Colloid Interf. Sci., 2002. **252**: p. 149-160.

12. Lee, S. and W.M. Sigmund, *AFM study of repulsive van der Waals forces between Teflon AF thin film and silica or alumina.* Colloids Surf. A, 2002. **204**: p. 43-50.

13. Wong, H., I. Fatt, and C.J. Radke, *Deposition and thinning of the human tear film.* J Colloid Interf Sci, 1996. **184**: p. 44-51.

14. Jensen, O.E. and J.B. Grotberg, *Insoluble surfactant spreading on a thin viscous film: shock evolution and film rupture.* J Fluid Mech, 1992. **240**: p. 259-288.

15. Reiter, G., et al., *Thin film instability induced by long-range forces.* Langmuir, 1999. **15**: p. 2551-2558.

16. Kovscek, A.R., H. Wong, and C.J. Radke, *A pore-level scenario for the development of mixed wettability in oil reservoirs.* AIChE J, 1993. **39**: p. 1072-1085.

17. Deryagin, B.V., V.M. Starov, and N.V. Churaev, *Profile of the transition zone between a wetting film and the meniscus of the bulk liquid.* Colloid J, 1976. **38**: p. 875-879.

18. Renk, F., P.C. Wayner, and G.M. Homsy, *On the transition between a wetting film and a capillary meniscus.* J. Colloid Interface Sci., 1978. **67**(3): p. 408-414.

19. Wong, H., S. Morris, and C.J. Radke, *Two-dimensional menisci in non-axisymmetric capillaries.* J. Colloid Interface Sci., 1992. **148**: p. 284-287.

20. Oron, A., S.H. Davis, and S.G. Bankoff, *Long-scale evolution of thin liquid films.* Reviews of Modern Phys., 1997. **69**: p. 931-980.

21. Zhang, W.W. and J.R. Lister, *Similarity solutions for van der Waals rupture of a thin film on a solid substrate.* Phys Fluids, 1999. **11**: p. 2454-2462.

22. Pismen, L.M., B.Y. Rubinstein, and I. Bazhlekov, *Spreading of a wetting film under the action of van der Waals forces.* Phys Fluids, 2000. **12**: p. 480-483.

23. Vaynblat, D., J.R. Lister, and T.P. Witelski, *Rupture of thin viscous films by van der Waals forces: evolution and self-similarity.* Phys Fluids, 2001. **13**: p. 1130-1140.

24. Davis, J.M. and S.M. Troian, *Influence of attractive van der Waals interactions on the optimal excitations in thermocapillary-driven spreading.* Phys Rev E, 2003. **67**: p. 016308.

25. Brochard-Wyart, F., et al., *Spreading of nonvolatile liquids in a continuum picture.* Langmuir, 1991. **7**: p. 335-338.

26. Kim, H.I., et al., *How disjoining pressure drives the dewetting of a polymer film on a silicon surface.* Phys. Rev. Lett., 1999. **17**: p. 3496-3499.

27. Oron, A., *Three-dimensional nonlinear dynamics of thin liquid films.* Phys. Rev. Lett., 2000. **85**: p. 2108-2111.

28. Higgins, A.M. and R.A.L. Jones, *Anisotropic spinodal dewetting as a route to self-assembly of patterned surfaces.* Nature, 2000. **404**: p. 476-478.

29. Seemann, R., S. Herminghaus, and K. Jacobs, *Dewetting patterns and molecular forces: a reconcillation.* Phys. Rev. Lett., 2001. **24**: p. 5534-5537.

30. Becker, J., et al., *Complex dewetting scenarios captured by thin-film models.* Nature Materials, 2003. **2**: p. 59-63.

31. Sharma, A. and R. Verma, *Pattern formation and dewetting in thin films of liquids showing complete macroscale wetting: from "pancakes" to "swiss cheese".* Langmuir, 2004. **20**: p. 10337-10345.

32. Moon, J., et al., *Pseudopartial wetting and precursor film growth in immiscible metal systems.* Langmuir, 2004. **20**: p. 402-408.

33. Mullins, W.W., *Solid Surface Morphologies Governed by Capillarity*, in *Metal Surfaces*. 1963, Am. Soc. Metals: Metals Park, Ohio. p. 17-66.

34. Rowlinson, J.S. and B. Widom, *Molecular Theory of Capillarity*. 1989, New York: Oxford University Press.

35. Hocking, L.M., *The influence of intermolecular forces on thin fluid layers.* Phys Fluids, 1993. **5**: p. 793-799.

36. Indeikina, A. and H.C. Chang, *A molecular theory for dynamic contact angles*, in *IUTAM Symposium on Non-Linear Singularities in Deformation and Flow*, D. Durban and J.R.A. Pearson, Editors. 1999, Kluwer Academic: Netherlands. p. 321-337.

37. Yeh, E.K., J. Newman, and C.J. Radke, *Equilibrium configurations of liquid droplets on solid surfaces under the influence of thin-film forces. Part II. Shape calculations.* Colloids Surf. A, 1999. **156**: p. 525-546.

38. Yeh, E.K., J. Newman, and C.J. Radke, *Equilibrium configurations of liquid droplets on solid surfaces under the influence of thin-film forces. Part I. Thermodynamics.* Colloids Surf. A, 1999. **156**: p. 137-144.

39. Morse, P.M. and H. Feshbach, *Methods of Theoretical Physics*. Vol. I. 1953: McGraw-Hill.

40. Courant, R. and D. Hilbert, *Methods of Mathematical Physics*. Vol. Vol. I. 1953: Interscience. II.

41. Burelbach, J.P., S.G. Bankoff, and S.H. Davis, *Nonlinear stability of evaporating/condensing liquid films.* J. Fluid Mechanics, 1988. **195**: p. 463-494.

42. Panzarella, C.H., Davis, S. H., Bankoff S. G., *Nonlinear dynamics in horizontal film boiling.* J. Fluid Mech., 2000: p. 402.

43. Anderson, D.M., Davis, S. H., *The spreading of volatile liquid droplets on heated surfaces* Phys. Fluids, 1995. **7**: p. 248.

44. Ajaev, V.S., *Spreading of thin volatile liquid droplets on uniformly heated surfaces.* J. Fluid Mech., 2005. **528**: p. 279-296.

45. Plesset, M.S. and A. Prosperetti, *Flow of vapour in a liquid enclosure.* J. Fluid Mech., 1976. **78**: p. 433-444.

46. Ajaev, V.S. and G.M. Homsy, *Modeling shapes and dynamics of confined bubbles.* Annu. Rev. Fluid Mech., 2006. **38**: p. 277-307.

47. Wayner, P.C., *Spreading of a liquid film with a finite contact angle by the evaporation/condensation process.* Langmuir, 1993. **9**: p. 294-299.

48. Carey, V.P., *Liquid-vapor Phase-change Phenomena: An Introduction to the Thermophysics of Vaporization and Condensation Processes in Heat Transfer Equipment*. 1992, Hebron: Taylor & Francis.

49. Homsy, G.M., et al., *Multimedia Fluid Mechanics*. 2004, Cambridge University Press: Cambridge.

50. Bender, C.M. and S.A. Orszag, *Advanced Mathematical Methods for Scientists and Engineers*. 1978, New York: McGraw-Hill.

51. Moffatt, H.K., *Viscous and resistive eddies near a sharp corner.* J. Fluid Mech., 1964. **18**: p. 1 - 18.

APPENDIX A. INTERMOLECULAR POTENTIAL Φ_{fg} BETWEEN THE LIQUID MOLECULE M AND A VAPOR MOLECULE N

In sub-domain 1, the potential (2.3.5) is

$$\Phi_{fg1} = n_g \int_{-\Gamma_1}^{2\psi-\Gamma_1} \int_S^{+\infty} \int_{-\infty}^{+\infty} \left[\frac{L_{fg}}{\left(R_1^2 + r_1^2 - 2R_1 r_1 \cos\phi_1 + z^2\right)^6} - \frac{\beta_{fg}}{\left(R_1^2 + r_1^2 - 2R_1 r_1 \cos\phi_1 + z^2\right)^3} \right] dz r_1 dr_1 d\phi_1, \tag{A1}$$

where the coordinates of M and N are defined in Fig. 2.2, and $\phi_1 = \theta_1 - \Gamma_1$. The integration with respect to z is

$$\begin{aligned} &\int_{-\infty}^{+\infty} \left[\frac{L_{fg}}{\left(R_1^2 + r_1^2 - 2R_1 r_1 \cos\phi_1 + z^2\right)^6} - \frac{\beta_{fg}}{\left(R_1^2 + r_1^2 - 2R_1 r_1 \cos\phi_1 + z^2\right)^3} \right] dz \\ &= \frac{3\pi}{8\left(R_1^2 + r_1^2 - 2R_1 r_1 \cos\phi_1\right)^{5/2}} - \frac{63\pi}{256\left(R_1^2 + r_1^2 - 2R_1 r_1 \cos\phi_1\right)^{11/2}}. \end{aligned} \tag{A2}$$

The integration of the first term in (A2) with respect to r_1 is

$$\begin{aligned} \int_S^{+\infty} \frac{r_1}{\left(R_1^2 + r_1^2 - 2R_1 r_1 \cos\phi_1\right)^{5/2}} dr_1 &= \frac{2\cos\phi_1}{3R_1^3 \sin^4\phi_1} + \frac{1}{3\left(S^2 + R_1^2 - 2SR_1\cos\phi_1\right)^{3/2}} \\ &- \frac{\cos\phi_1 (S - R_1\cos\phi_1)}{3R_1\left(S^2 + R_1^2 - 2SR_1\cos\phi_1\right)^{3/2} \sin^2\phi_1} - \frac{2\cos\phi_1 (S - R_1\cos\phi_1)}{3R_1^3\left(S^2 + R_1^2 - 2SR_1\cos\phi_1\right)^{1/2} \sin^4\phi_1}, \end{aligned} \tag{A3}$$

$$= \frac{4S - R_1}{12(S - R_1)^4} - \frac{R_1\left(R_1^2 - 6SR_1 + 15S^2\right)}{24(S - R_1)^6}\phi_1^2 + O\left(\phi_1^4\right), \tag{A4}$$

in the limit $\phi_1 \to 0$. The integration of the second term in (A2) is

$$\int_S^{+\infty} \frac{r_1}{\left(R_1^2 + r_1^2 - 2R_1 r_1 \cos\phi_1\right)^{11/2}} dr_1 = \frac{128\cos\phi_1}{315R_1^9(\sin\phi_1)^{10}} + \frac{1}{9\left(S^2 + R_1^2 - 2SR_1\cos\phi_1\right)^{9/2}}$$
$$- \frac{\cos\phi_1(S - R_1\cos\phi_1)}{9R_1(\sin\phi_1)^2\left(S^2 + R_1^2 - 2SR_1\cos\phi_1\right)^{9/2}} - \frac{8\cos\phi_1(S - R_1\cos\phi_1)}{63R_1^3(\sin\phi_1)^4\left(S^2 + R_1^2 - 2SR_1\cos\phi_1\right)^{7/2}}$$
$$- \frac{16\cos\phi_1(S - R_1\cos\phi_1)}{105R_1^5(\sin\phi_1)^6\left(S^2 + R_1^2 - 2SR_1\cos\phi_1\right)^{5/2}} - \frac{64\cos\phi_1(S - R_1\cos\phi_1)}{315R_1^7(\sin\phi_1)^8\left(S^2 + R_1^2 - 2SR_1\cos\phi_1\right)^{3/2}}$$
$$- \frac{128\cos\phi_1(S - R_1\cos\phi_1)}{315R_1^9(\sin\phi_1)^{10}\left(S^2 + R_1^2 - 2SR_1\cos\phi_1\right)^{1/2}}. \qquad \text{(A5)}$$

$$= \frac{10S - R_1}{90(S - R_1)^{10}} - \frac{R_1\left(R_1^2 - 12SR_1 + 66S^2\right)}{120(S - R_1)^{12}}\phi_1^2 + O\left(\phi_1^4\right) \qquad \text{(A6)}$$

in the limit $\phi_1 \to 0$.

Therefore, integration with respect to ϕ_1 gives

$$\Phi_{fg1} = \frac{63\pi n_g L_{fg}}{256}\left[\frac{(10S - R_1)\psi}{45(S - R_1)^{10}} - \frac{R_1\left(R_1^2 - 12SR_1 + 66S^2\right)\psi\left(4\psi^2 + 3\Gamma_1^2 - 6\psi\Gamma_1\right)}{180(S - \rho)^{12}}\right]$$
$$- \frac{3\pi n_g \beta_{fg}}{8}\left[\frac{(4S - R_1)\psi}{6(S - R_1)^4} - \frac{R_1\left(R_1^2 - 6SR_1 + 15S^2\right)\psi\left(4\psi^2 + 3\Gamma_1^2 - 6\psi\Gamma_1\right)}{36(S - R_1)^6}\right]. \qquad \text{(A7)}$$

This is accurate to $O(\psi^5)$.

In sub-domain 2, the potential (2.3.8) is

$$\Phi_{fg2} = n_g \int_{-\Gamma_2}^{\frac{\pi}{2} - \psi - \Gamma_2} \int_0^{+\infty} \int_{-\infty}^{+\infty} \left[\frac{L_{fg}}{\left(R_2^2 + r_2^2 - 2R_2 r_2\cos\phi_2 + z^2\right)^6} - \frac{\beta_{fg}}{\left(R_2^2 + r_2^2 - 2R_2 r_2\cos\phi_2 + z^2\right)^3}\right] dz r_2 dr_2 d\phi_2, \qquad \text{(A8)}$$

where the coordinates of M and N are defined in Fig. 2.2, and $\phi_2 = \theta_2 - \Gamma_2$. The integration with respect to z is the same as (A2):

$$\int_{-\infty}^{+\infty}\left[\frac{L_{fg}}{\left(R_2^2+r_2^2-2R_2r_2\cos\phi_2+z^2\right)^6}-\frac{\beta_{fg}}{\left(R_2^2+r_2^2-2R_2r_2\cos\phi_2+z^2\right)^3}\right]dz$$
$$=\frac{3\pi}{8\left(R_2^2+r_2^2-2R_2r_2\cos\phi_2\right)^{5/2}}-\frac{63\pi}{256\left(R_2^2+r_2^2-2R_2r_2\cos\phi_2\right)^{11/2}}. \tag{A9}$$

Taking $S \to 0$ in (A3) and (A5) gives the result of the integration with respect to r_2 in (A8):

$$\int_0^{+\infty}\frac{r_2}{\left(R_2^2+r_2^2-2R_2r_2\cos\phi_2\right)^{5/2}}dr_2$$
$$=\frac{1}{3R_2^3}+\frac{2\cos\phi_2}{3R_2^3\sin^4\phi_2}+\frac{\cos^2\phi_2}{3R_2^3\sin^2\phi_2}-\frac{\cos^4\phi_2}{3R_2^3\sin^4\phi_2}=\frac{1}{3R_2^3(1-\cos\phi_2)^2}, \tag{A10}$$

$$\int_0^{+\infty}\frac{r_2}{\left(R_2^2+r_2^2-2R_2r_2\cos\phi_2\right)^{11/2}}dr_2=\frac{1}{9R_2^9}+\frac{128\cos\phi_2}{315R_2^9\sin^{10}\phi_2}+\frac{\cos^2\phi_2}{9R_2^9\sin^2\phi_2}$$
$$+\frac{8\cos^2\phi_2}{63R_2^9\sin^4\phi_2}+\frac{16\cos^2\phi_2}{105R_2^9\sin^6\phi_2}+\frac{64\cos^2\phi_2}{315R_2^9\sin^8\phi_2}+\frac{128\cos^2\phi_2}{315R_2^9\sin^{10}\phi_2}. \tag{A11}$$

The integration with respect to ϕ_2 of (A10) is

$$\int_{\Gamma_2}^{\frac{\pi}{2}-\psi-\Gamma_2}\frac{1}{R_2^3(1-\cos\phi_2)^2}d\phi_2=\frac{4}{3}\left(\frac{a_1}{y^3}+\frac{a_2}{v_2^3}\right), \tag{A12a}$$

where y is the vertical height of M from the solid surface, and

$$a_1=-\frac{1}{2}-\frac{3}{4}\sin(\psi+\Gamma_2)+\frac{1}{4}\sin^3(\psi+\Gamma_2), \tag{A12b}$$

$$a_2=\frac{1}{2}+\frac{3}{4}\cos\Gamma_2-\frac{1}{4}\cos^3\Gamma_2, \tag{A12c}$$

$$v_2=-R_2\sin\Gamma_2. \tag{A12d}$$

The integration with respect to ϕ_2 of (A11) is

$$\int_{-\Gamma_2}^{\frac{\pi}{2}-\psi-\Gamma_2}\left(\begin{array}{l}\dfrac{1}{9R_2^9}+\dfrac{128\cos\phi_2}{315R_2^9\sin^{10}\phi_2}+\dfrac{\cos^2\phi_2}{9R_2^9\sin^2\phi_2}\\ +\dfrac{8\cos^2\phi_2}{63R_2^9\sin^4\phi_2}+\dfrac{16\cos^2\phi_2}{105R_2^9\sin^6\phi_2}+\dfrac{64\cos^2\phi_2}{315R_2^9\sin^8\phi_2}+\dfrac{128\cos^2\phi_2}{315R_2^9\sin^{10}\phi_2}\end{array}\right)d\phi_1$$
$$=\frac{1}{2835}\left(\frac{b_1}{y^9}+\frac{b_2}{v_2^9}\right),$$

(A13a)

where

$$\begin{aligned}b_1 &= -128-315\sin(\psi+\Gamma_2)+420\sin^3(\psi+\Gamma_2)\\ &-378\sin^5(\psi+\Gamma_2)+180\sin^7(\psi+\Gamma_2)-35\sin^9(\psi+\Gamma_2).\end{aligned} \tag{A13b}$$

$$\begin{aligned}b_2 &= 128+315\cos\Gamma_2-420\cos^3\Gamma_2\\ &+378\cos^5\Gamma_2-180\cos^7\Gamma_2+35\cos^9\Gamma_2.\end{aligned} \tag{A13c}$$

Therefore, Φ_{fg} in sub-domain 2 is

$$\Phi_{fg2}=\frac{\pi n_g L_{fg}}{11520}\left(\frac{b_1}{y^9}+\frac{b_2}{v_2^9}\right)-\frac{\pi n_g \beta_{fg}}{6}\left(\frac{a_1}{y^3}+\frac{a_2}{v_2^3}\right), \tag{A14}$$

In sub-domain 3, the potential is the same as that in sub-domain 2, because it is in the same shape as sub-domain 2. However, N is located at (r_3, θ_3, z) and M at (R_3, Γ_3, 0), thus

$$\Phi_{fg3}=\frac{\pi n_g L_{fg}}{11520}\left(\frac{b_3}{y^9}+\frac{b_4}{v_3^9}\right)-\frac{\pi n_g \beta_{fg}}{6}\left(\frac{a_3}{y^3}+\frac{a_4}{v_3^3}\right), \tag{A15a}$$

where

$$a_3=-\frac{1}{2}-\frac{3}{4}\cos\Gamma_3+\frac{1}{4}\sin^3\Gamma_3, \tag{A15b}$$

$$a_4 = \frac{1}{2} + \frac{3}{4}\sin(\psi + \Gamma_3) - \frac{1}{4}\sin^3(\psi + \Gamma_3), \tag{A15c}$$

$$\begin{aligned} b_3 = &-128 - 315\cos\Gamma_3 + 420\cos^3\Gamma_3 \\ &- 378\cos^5\Gamma_3 + 180\cos^7\Gamma_3 - 35\cos^9\Gamma_3. \end{aligned} \tag{A15d}$$

$$\begin{aligned} b_4 = &+128 + 315\sin(\psi + \Gamma_3) - 420\sin^3(\psi + \Gamma_3) \\ &+ 378\sin^5(\psi + \Gamma_3) - 180\sin^7(\psi + \Gamma_3) + 35\sin^9(\psi + \Gamma_3). \end{aligned} \tag{A15e}$$

$$v_3 = -R_3\cos(\psi + \Gamma_3). \tag{A15f}$$

The potential in the solid domain Φ_{fs} only differs from Φ_{fg3} in the range of ϕ_3. For the solid domain, ϕ_3 is from π-Γ_3 to 2π-Γ_3. Therefore,

$$\Phi_{fs} = \frac{\pi n_s L_{fs}}{45y^9} - \frac{\pi n_s \beta_{fs}}{6y^3}. \tag{A16}$$

APPENDIX B. EXCESS INTERACTION ENERGY E

The excess energy E is

$$E = \int_D^h \Phi dy, \tag{B1}$$

where Φ is the total interaction thin-film potential between point M in liquid and other material points:

$$\begin{aligned}\Phi &= n_f\left(\Phi_{fs} + \Phi_{fg} + \Phi_{ff}\right)\\ &= -\frac{\pi n_f^2 L_{ff}}{11520}\left[\frac{256(1-\lambda_1)}{y^9} + \left(\frac{b_1+b_3}{y^9} + \frac{b_2-128}{v_2^9} + \frac{b_4-128}{v_3^9}\right)(1-\lambda_2)\right]\\ &+ \frac{\pi n_f^2 \beta_{ff}}{6}\left[\frac{(1-\rho_1)}{y^3} + \left(\frac{a_1+a_3}{y^3} + \frac{a_2-\frac{1}{2}}{v_2^3} + \frac{a_4-\frac{1}{2}}{v_3^3}\right)(1-\rho_2)\right].\end{aligned} \tag{B2}$$

The integration of the first term in the repulsive component in (B2) is

$$-\frac{\pi n_f^2 L_{ff}(1-\lambda_1)}{45}\int_D^h \frac{1}{y^9}dy = \frac{\pi n_f^2 L_{ff}(1-\lambda_1)}{360}\left(\frac{1}{h^8} - \frac{1}{D^8}\right). \tag{B3}$$

The integration of b_1/y^9 in (B2) is

$$\begin{aligned}&-\frac{\pi n_f^2 L_{ff}(1-\lambda_2)}{11520}\int_D^h \frac{b_1}{y^9}dy = \frac{\pi n_f^2 L_{ff}(1-\lambda_2)}{11520}\times\\ &\left[\frac{-16x_2^6 - 48x_2^4h^2 - 48x_2^2h^4 - 16h^6}{\left(x_2^2+h^2\right)^3 h^8} + \frac{16x_2^7 + 56x_2^5h^2 + 70x_2^3h^4 + 35x_2h^6}{\left(x_2^2+h^2\right)^{7/2} h^8}\right.\\ &\left.+\frac{16x_2^6 + 48x_2^4D^2 + 48x_2^2D^4 + 16D^6}{\left(x_2^2+D^2\right)^3 D^8} - \frac{16x_2^7 + 56x_2^5D^2 + 70x_2^3D^4 + 35x_2D^6}{\left(x_2^2+D^2\right)^{7/2} D^8}\right],\end{aligned} \tag{B4}$$

where $x_2 = x_2(S, \psi, h)$ (Eqt. 2.4.6). Substitution x_2 with x_3 in (B4) gives the integration of b_3/y^9, where x3 = x3(S, ψ, h) (Eq. 2.4.11).

Since $y << x_2$ and $\psi << 1$, v_2 is

$$v_2 = x_2 \cos\psi + y \sin\psi \approx x_2 \cos\psi. \tag{B5}$$

And

$$\cos\Gamma_2 << 1. \tag{B6}$$

Thus, the integration of $(b_2\text{-}128)/v_2^9$ is

$$\begin{aligned} &-\frac{\pi n_f^2 L_{ff}(1-\lambda_2)}{11520}\int_0^h \frac{b2-128}{v_2^9}dy \\ &= -\frac{7\pi n_f^2 L_{ff}(1-\lambda_2)}{512\cos^8\psi}\left(\frac{h^2}{x_2^{10}}\right) + \frac{7\pi n_f^2 L_{ff}(1-\lambda_2)\sin\psi}{256\cos^9\psi}\left(\frac{h}{x_2^9}\right). \end{aligned} \tag{B7}$$

Similarly,

$$v_3 = x_3 \cos\psi + y \sin\psi \approx x_3 \cos\psi. \tag{B8}$$

And

$$\sin(\psi + \Gamma_3) << 1. \tag{B9}$$

Therefore, substitution x_2 with x_3 in (B7) gives the integration of $(b_4\text{-}128)/v_3^9$

The integration of the first term in the attractive component in (B2) is

$$\frac{\pi n_f^2 \beta_{ff}(1-\rho_1)}{6}\int_D^h \frac{1}{y^3}dy = -\frac{\pi n_f^2 \beta_{ff}(1-\rho_1)}{12h^2} + \frac{\pi n_f^2 \beta_{ff}(1-\rho_1)}{12D^2}. \tag{B10}$$

The integration of the second term in the attractive component in (B2) containing a_1/y^3 is

$$\begin{aligned} &\frac{\pi n_f^2 \beta_{ff}(1-\rho_2)}{6}\int_D^h \frac{a_1}{y^3}dy \\ &= \frac{\pi n_f^2 \beta_{ff}(1-\rho_2)}{6}\left(\frac{-x_2}{4h^2\sqrt{x_2^2+h^2}} + \frac{1}{4h^2} + \frac{x_2}{4D^2\sqrt{x_2^2+D^2}} - \frac{1}{4D^2}\right). \end{aligned} \tag{B11}$$

Substitution x_2 with x_3 in (B11) gives the integration of a_3/y^3.

Subject to (B5) and (B6), the integration of $(a_2\text{-}1/2)/v_2^3$ is

$$\begin{aligned}&\frac{\pi n_f^2 \beta_{ff}(1-\rho_2)}{6}\int_0^h \frac{a_2 - 1/2}{v_2^3}dy \\ &= \frac{\pi n_f^2 \beta_{ff}(1-\rho_2)}{16\cos^2\psi}\left(\frac{h}{x_2^2}\right)^2 - \frac{\pi n_f^2 \beta_{ff}(1-\rho_2)\sin\psi}{8\cos^3\psi}\frac{h}{x_2^3}.\end{aligned} \tag{B12}$$

Similarly, substitution x_2 with x_3 in (B12) gives the integration of $(a_4\text{-}1/2)/v_3^3$. Summing up (B3) to (B12) gives $E = E(S, \psi, h, D)$.

APPENDIX C. COMPUTER PROGRAMS

C.1. Program for Computing Equilibrium Film Profiles

```
C Calculate equilibrium film profiles
C based on differnt dimensionless pressure difference CDP

      EXTERNAL F
      DIMENSION Y(2), D(2), B(2), C(2), G(2)
      DOUBLE PRECISION Y, D, B, C, G, T, E, DX
C Y(1) IS H0, Y(2) IS HOX
      COMMON /ONE/ CDP, EPS, R, SI10
C CDP: DIMENSIONLESS PRESSURE DIFFERENCE (C)
C EPS: EPSILONE

      OPEN(1, FILE = 'H.TXT', STATUS='UNKNOWN')
      OPEN(2, FILE = 'HX.TXT', STATUS='UNKNOWN')
      EPS = 1D-2
      R = 5.43D-9
      SI10 = 1D0
C     CDP = -.5006164164D0
C FOR DROP WITH PRECURSOR FILM
      CDP = -.50006332993508D0
C FOR UNIFORM FILM PROFILE
C     CDP = R*SI10-EPS
C FOR DROP WITHOUT PRECURSOR FILM
C     CDP = -0.4
      DX = 5.0D-3
      ERROR = 1.0D-7
      T = 0.0D0
C INITIAL CONDITIONS FOR H0 AND H0X
      Y(1) = 1.0D0
      Y(2) = 0.0D0
      M = 2
      WRITE(1, 20) T, Y(1)
      DO 10 I = 1, 1000
C       IF(I .EQ. 508) THEN
C         PRINT*,100
C       END IF
        CALL RKT(T, DX, Y, M, F, ERROR, D, B, C, G)
        T = T + DX
        WRITE(1, 20) T, Y(1)
        WRITE(2, 30) Y(2)
10    CONTINUE
20    FORMAT(2D25.16)
30    FORMAT(D25.16)
      CLOSE(1)
      CLOSE(2)
```

```
      END
C F IS USED TO CALCULATE H0 AND H0X
      SUBROUTINE F(T, Y, M, D)
      DIMENSION Y(M), D(M)
      DOUBLE PRECISION Y, D, T, E, NUMER, DENOM
      COMMONE /ONE/ CDP, EPS, R, SI10
      D(1) = Y(2)
      NUMER = CDP*Y(1)**9D0+EPS*Y(1)**6D0*(1D0-Y(2)**4D0)
     +-R*(SI10-Y(2)**10D0)
      DENOM = Y(1)**9D0+1.25D0*R*Y(1)*Y(2)**8D0
     +-2D0*EPS*Y(1)**7D0*Y(2)**2D0
      D(2) = NUMER/DENOM
      RETURN
      END

C ROUNGE-KUTTA
      SUBROUTINE RKT(T, H, Y, M, F, ERROR, D, B, C, G)
      DIMENSION Y(M), D(M),A(4), B(M), C(M), G(M)
      DOUBLE PRECISION Y, D, B, C, G, A, T, H, X, HH, TT
      HH = H
      N = 1
      P = 1.0D0+ERROR
      X = T
      DO 5 I = 1, M
5     C(I) = Y(I)
10    IF(P .GE. ERROR) THEN
        A(1) = HH/2.0D0
        A(2) = A(1)
        A(3) = HH
        A(4) = HH
        DO 20 I = 1, M
          G(I) = Y(I)
          Y(I) = C(I)
20      CONTINUE
        DT = H / N
        T = X
        DO 100 J = 1, N
          CALL F(T, Y, M, D)
          DO 30 I = 1, M
30        B(I) = Y(I)
          DO 50 K = 1, 3
            DO 40 I = 1, M
              Y(I) = Y(I) + A(K)*D(I)
              B(I) = B(I) + A(K+1)*D(I)/3.0D0
40              CONTINUE
                TT = T + A(K)
            CALL F(TT, Y, M, D)
50            CONTINUE
```

```
          DO 60 I = 1, M
60              Y(I) = B(I) + HH*D(I)/6.0D0
          T = T+DT
100     CONTINUE
        P = 0.0D0
        IF(Y(1) .LE. 0.0D0) THEN
          GOTO 110
        ELSE
          DO 120 I = 1,M
              Q=ABS(Y(I)-G(I))
      IF(Q .GT. P) P = Q
120           CONTINUE
        END IF
        HH = HH/2.0D0
        N = N+N
        GOTO 10
      END IF
110   CONTINUE
      T = X
      RETURN
      END
```

C.2. Program for Checking Stability of Drops with Precursor Film

```
C Solve Eigenvalue Problem of a Half Drop with Uniform Film

      EXTERNAL DEVCRG, DWRCRN
C SUBROUTINE INCLUDED IN IMSL PACKAGE
      DOUBLE PRECISION DX,DX2,DX3,DX4,WAVN1,
     +          WAVN2, WAVN3, GROW1, GROW2, GROW3
C GROW: GROWTH RATE
C WAVN: WAVE NUMBER
C Y(1) IS H0, Y(2) IS HOX
      COMMON /ONE/ CDP, EPS, R, SI10
C CDP: DIMENSIONLESS PRESSURE DIFFERENCE (C)
C EPS: EPSILONE
      PARAMETER (N = 462)
C N IS NUMBER OF POINTS IN THE PROFILE
      DIMENSION X(N),H(N), HX(N), HX2(N), HX3(N),
     +     G4(N), G3(N), G2(N), G1(N), A(N-2,N-2)
      DOUBLE PRECISION X,H, HX,HX2,HX3,G4,G3,G2,G1, A
      DOUBLE PRECISION AWW, AW, AP, AE, AEE
C VARIABLES FOR USING ISML SUBROUTINES
      INTEGER N1, LDA, LDEVEC
      PARAMETER (N1=N-2, LDA=N1, LDEVEC=N1)

      DOUBLE COMPLEX EVAL(N1), EVEC(LDEVEC,N1)

      EPS = 1D-2
      R = 5.43D-9
      SI10 = 1D0

      CDP = -.5006164164D0
      DX = 5.0D-3
      DX2 = DX*DX
      DX3 = DX2*DX
      DX4 = DX3*DX
      ERROR = 1.0D-7

      OPEN(1, FILE='H.TXT', STATUS='UNKNOWN')
      OPEN(2, FILE='HX.TXT', STATUS='UNKNOWN')
      DO 40 I = 1, N
        READ(1, *) X(I), H(I)
        READ(2, *) HX(I)
40    CONTINUE
      CLOSE(1)
      CLOSE(2)
20    FORMAT(2D25.16)
30    FORMAT(1D25.16)
```

```
C CALCULATE HXXX
      DO 50 I = 1, N
       HX2(I) = 4D0 * (EPS * H(I) ** 6D0 - R * SI10 + R * HX(I) ** 10D0
     +   + CDP * H(I) ** 9D0 - EPS * H(I) ** 6D0 * HX(I) ** 4D0)
     +   / H(I) / (4D0 * H(I) ** 8D0 + 5D0 * R * HX(I) ** 8D0
     +   - 8D0 * EPS * H(I) ** 6D0 * HX(I) ** 2D0)
       HX3(I) = -4D0 * HX(I) * (-9D0 * R * SI10 - 20D0 * R * HX(I)**8D0
     +     * H(I) * HX2(I) + 10D0 * R * HX(I) ** 6D0 * H(I) ** 2D0
     +     * HX2(I) ** 2D0 - 4D0 * EPS * H(I) ** 8D0 * HX2(I)**2D0
     +     + 9D0 * R * HX(I) ** 10D0 + 8D0 * EPS * HX(I) ** 2D0
     +     * H(I) ** 7D0 * HX2(I) + 3D0 * EPS * H(I) ** 6D0
     +     - 3D0 * EPS * H(I) ** 6D0 * HX(I) ** 4D0)
     +     / H(I) ** 2D0 / (4D0 * H(I) ** 8D0 + 5D0 * R *
     +       HX(I) ** 8D0 - 8D0 * EPS * H(I) ** 6D0 * HX(I)**2D0)

50    CONTINUE

C DETERMINE G4, G3, G2, G1 (COEFFICIENTS IN THE EIGENVALUE
PROBLEM)
      DO 70 I = 1, N
       G4(I) = 1.D0/H(I)**5D0*(5.D0*R*HX(I)**8D0+4.D0*H(I)**8D0
     +   - 8.D0*H(I)**6D0*EPS*HX(I)**2D0) / 0.12D2

       G3(I) = -0.4D1/0.3D1*HX(I)/H(I)**6D0*(-2.D0*H(I)**6D0
     +   * EPS*HX(I)**2D0+5.D0*R*HX(I)**8D0+2.D0*H(I)**7D0
     +   * EPS*HX2(I)-5.D0*R*HX(I)**6D0*HX2(I)*H(I))

       G2(I) =  -(1.D0/H(I)**7D0*(-99.D0*R*HX(I)**10D0+4.D0*H(I)**8D0
     +   * EPS*HX(I)*HX3(I)-24.D0*EPS*HX(I)**2D0*HX2(I)*H(I)**7D0
     +   + 4.D0*H(I)**8D0*EPS*HX2(I)**2D0-10.D0*R*HX(I)**7D0*HX3(I)
     +   * H(I)**2D0+180.D0*R*HX(I)**8D0*HX2(I)*H(I)-3.D0*EPS*H(I)
     +   **6D0-70.D0*R*HX(I)**6D0*HX2(I)**2D0*H(I)**2D0+9.D0*R*SI10
     +   + 15.D0*EPS*HX(I)**4D0*H(I)**6D0)) / 0.3D1

       G1(I) = (12.D0*H(I)**10D0*HX3(I)-12.D0*EPS*HX(I)*H(I)**6D0+
     +   + 12.D0*EPS*HX(I)**5D0*H(I)**6D0+252.D0*R*HX(I)*SI10
     +   - 8.D0*EPS*HX(I)**2D0*HX3(I)*H(I)**8D0-16.D0*EPS*HX(I)
     +  *HX2(I)**2D0*H(I)**8D0-200.D0*R*HX(I)**7D0*HX2(I)**2D0*H(I)
     +   ** 2D0+480.D0*R*HX(I)**9D0*HX2(I)*H(I)-25.D0*R*HX(I)**8D0
     +   * HX3(I)*H(I)**2D0-252.D0*R*HX(I)**11D0)/H(I)**8D0/0.12D2

70    CONTINUE

C DETERMINE AWW, AW, AP, AE, AEE
      DO 75 I = 1, N-2
       DO 74 J = 1, N-2
        A(I,J) = 0.D0
74     CONTINUE
```

```
75      CONTINUE
        DO 80 I = 2, N-1
          AWW = G4(I)/DX4-G3(I)*0.5D0/DX3
          AW = -4.D0*G4(I)/DX4+G3(I)/DX3+G2(I)/DX2-G1(I)*0.5D0/DX
          AP = 6.D0*G4(I)/DX4-2.D0*G2(I)/DX2
          AE = -4.D0*G4(I)/DX4-G3(I)/DX3+G2(I)/DX2+G1(I)*0.5D0/DX
          AEE = G4(I)/DX4+G3(I)*0.5D0/DX3
          IF(I .EQ. 2) THEN
            A(I-1,I-1) = AP - AWW
            A(I-1,I) = AE
            A(I-1,I+1) = AEE
          ELSEIF(I .EQ. 3) THEN
            A(I-1,I-2) = AW
            A(I-1,I-1) = AP
            A(I-1, I) = AE
            A(I-1, I+1) = AEE
          ELSEIF(I .EQ. N-2) THEN
            A(I-1, I-3) = AWW
            A(I-1, I-2) = AW
            A(I-1, I-1) = AP
            A(I-1, I) = AE
          ELSEIF(I .EQ. N-1) THEN
            A(I-1, I-3) = AWW
            A(I-1, I-2) = AW
            A(I-1, I-1) = AP - AEE
          ELSE
            A(I-1,I-3) = AWW
            A(I-1, I-2) = AW
            A(I-1, I-1) = AP
            A(I-1, I) = AE
            A(I-1, I+1) = AEE
          END IF
80      CONTINUE

C Find eigenvalues and vectors of A

        CALL DEVCRG (N1, A, LDA, EVAL, EVEC, LDEVEC)

C       CALL DWRCRN ('EVAL', 1, N1, EVAL, 1, 0)
        OPEN(1, FILE='EIGVAL.TXT', STATUS='UNKNOWN')
        DO 120 I = 1, N1
          WRITE(1,300) -REAL(EVAL(I))
120     CONTINUE

        OPEN(2, FILE='EIGVECTOR1.TXT', STATUS='UNKNOWN')
        OPEN(3, FILE='EIGVECTOR2.TXT', STATUS='UNKNOWN')
        OPEN(4, FILE='EIGVECTOR3.TXT', STATUS='UNKNOWN')
        OPEN(5, FILE='EIGVECTOR4.TXT', STATUS='UNKNOWN')
```

```
      WRITE(2,200) X(1), 0D0
      WRITE(3,200) X(1), 0D0
      WRITE(4,200) X(1), 0D0
      WRITE(5,200) X(1), 0D0
      DO 90 I = 1,N-2
        WRITE(2, 200) X(I+1), REAL(EVEC(I,N-2))
        WRITE(3, 200) X(I+1), REAL(EVEC(I,N-3))
        WRITE(4, 200) X(I+1), REAL(EVEC(I,N-4))
        WRITE(5, 200) X(I+1), REAL(EVEC(I,N-5))
90    CONTINUE
      WRITE(2,200) X(N), 0D0
      WRITE(3,200) X(N), 0D0
      WRITE(4,200) X(N), 0D0
      WRITE(5,200) X(N), 0D0
      CLOSE(1)
      CLOSE(2)
      CLOSE(3)
      CLOSE(4)
      CLOSE(5)

200   FORMAT(2D25.16)
300   FORMAT(D25.16)
      END
```

C.3. Program for Checking Stability of Drops without Precursor Film

```
C Solve Eigenvalue problem for a drop without uniform film

      EXTERNAL DEVCRG, DWRCRN

      DOUBLE PRECISION DX,DX2,DX3,DX4,WAVN1,
     +          WAVN2, WAVN3, GROW1, GROW2, GROW3
C GROW: GROWTH RATE
C WAVN: WAVE NUMBER
C Y(1) IS H0, Y(2) IS HOX
      COMMON /ONE/ CDP, EPS, R, SI10
C CDP: DIMENSIONLESS PRESSURE DIFFERENCE (C)
C EPS: EPSILONE
      PARAMETER (N = 3096)
      DIMENSION X(N),H(N), HX(N), HX2(N), HX3(N),
     +     G4(N), G3(N), G2(N), G1(N), A(N-2,N-2)
      DOUBLE PRECISION X,H, HX,HX2,HX3,G4,G3,G2,G1, A
      DOUBLE PRECISION AWW, AW, AP, AE, AEE
C VARIABLES FOR USING ISML SUBROUTINES
      INTEGER N1, LDA, LDEVEC
      PARAMETER (N1=N-2, LDA=N1, LDEVEC=N1)

      DOUBLE COMPLEX EVAL(N1), EVEC(LDEVEC,N1)

      EPS = 1D-2
      R = 1D0/6D0*EPS
      SI10 = 1D0
      CDP = -.5D0

      ERROR = 1.0D-7

      OPEN(1, FILE='H.TXT', STATUS='UNKNOWN')
      OPEN(2, FILE='HX.TXT', STATUS='UNKNOWN')
      DO 40 I = 1, N
        READ(1, *) X(I), H(I)
        READ(2, *) HX(I)
40    CONTINUE
      CLOSE(1)
      CLOSE(2)
20    FORMAT(2D25.16)
30    FORMAT(1D25.16)
      DX = X(2) - X(1)
      DX2 = DX*DX
      DX3 = DX2*DX
      DX4 = DX3*DX
```

```
C CALCULATE HXXX
      DO 50 I = 1, N
       HX2(I) = 4D0 * (EPS * H(I) ** 6D0 - R * SI10 + R * HX(I) ** 10D0
     +   + CDP * H(I) ** 9D0 - EPS * H(I) ** 6D0 * HX(I) ** 4D0)
     +   / H(I) / (4D0 * H(I) ** 8D0 + 5D0 * R * HX(I) ** 8D0
     +   - 8D0 * EPS * H(I) ** 6D0 * HX(I) ** 2D0)
       HX3(I) = -4D0 * HX(I) * (-9D0 * R * SI10 - 20D0 * R * HX(I)**8D0
     +     * H(I) * HX2(I) + 10D0 * R * HX(I) ** 6D0 * H(I) ** 2D0
     +     * HX2(I) ** 2D0 - 4D0 * EPS * H(I) ** 8D0 * HX2(I)**2D0
     +     + 9D0 * R * HX(I) ** 10D0 + 8D0 * EPS * HX(I) ** 2D0
     +     * H(I) ** 7D0 * HX2(I) + 3D0 * EPS * H(I) ** 6D0
     +     - 3D0 * EPS * H(I) ** 6D0 * HX(I) ** 4D0)
     +     / H(I) ** 2D0 / (4D0 * H(I) ** 8D0 + 5D0 * R *
     +       HX(I) ** 8D0 - 8D0 * EPS * H(I) ** 6D0 * HX(I)**2D0)

50    CONTINUE

C DETERMINE G4, G3, G2, G1
      DO 70 I = 1, N
       G4(I) = 1.D0/H(I)**5D0*(5.D0*R*HX(I)**8D0+4.D0*H(I)**8D0
     +  - 8.D0*H(I)**6D0*EPS*HX(I)**2D0) / 0.12D2

       G3(I) = -0.4D1/0.3D1*HX(I)/H(I)**6D0*(-2.D0*H(I)**6D0
     +  * EPS*HX(I)**2D0+5.D0*R*HX(I)**8D0+2.D0*H(I)**7D0
     +  * EPS*HX2(I)-5.D0*R*HX(I)**6D0*HX2(I)*H(I))

       G2(I) =  -(1.D0/H(I)**7D0*(-99.D0*R*HX(I)**10D0+4.D0*H(I)**8D0
     +  * EPS*HX(I)*HX3(I)-24.D0*EPS*HX(I)**2D0*HX2(I)*H(I)**7D0
     +  + 4.D0*H(I)**8D0*EPS*HX2(I)**2D0-10.D0*R*HX(I)**7D0*HX3(I)
     +  * H(I)**2D0+180.D0*R*HX(I)**8D0*HX2(I)*H(I)-3.D0*EPS*H(I)
     +  **6D0-70.D0*R*HX(I)**6D0*HX2(I)**2D0*H(I)**2D0+9.D0*R*SI10
     +  + 15.D0*EPS*HX(I)**4D0*H(I)**6D0)) / 0.3D1

       G1(I) = (12.D0*H(I)**10D0*HX3(I)-12.D0*EPS*HX(I)*H(I)**6D0+
     +  + 12.D0*EPS*HX(I)**5D0*H(I)**6D0+252.D0*R*HX(I)*SI10
     +  - 8.D0*EPS*HX(I)**2D0*HX3(I)*H(I)**8D0-16.D0*EPS*HX(I)
     + *HX2(I)**2D0*H(I)**8D0-200.D0*R*HX(I)**7D0*HX2(I)**2D0*H(I)
     +  ** 2D0+480.D0*R*HX(I)**9D0*HX2(I)*H(I)-25.D0*R*HX(I)**8D0
     +  * HX3(I)*H(I)**2D0-252.D0*R*HX(I)**11D0)/H(I)**8D0/0.12D2

70    CONTINUE

C DETERMINE AWW, AW, AP, AE, AEE
      DO 75 I = 1, N-2
       DO 74 J = 1, N-2
        A(I,J) = 0.D0
74     CONTINUE
75    CONTINUE
```

```
      DO 80 I = 2, N-1
        AWW = G4(I)/DX4-G3(I)*0.5D0/DX3
        AW = -4.D0*G4(I)/DX4+G3(I)/DX3+G2(I)/DX2-G1(I)*0.5D0/DX
        AP = 6.D0*G4(I)/DX4-2.D0*G2(I)/DX2
        AE = -4.D0*G4(I)/DX4-G3(I)/DX3+G2(I)/DX2+G1(I)*0.5D0/DX
        AEE = G4(I)/DX4+G3(I)*0.5D0/DX3
        IF(I .EQ. 2) THEN
          A(I-1,I-1) = AP - AWW
          A(I-1,I) = AE
          A(I-1,I+1) = AEE
        ELSEIF(I .EQ. 3) THEN
          A(I-1,I-2) = AW
          A(I-1,I-1) = AP
          A(I-1, I) = AE
          A(I-1, I+1) = AEE
        ELSEIF(I .EQ. N-2) THEN
          A(I-1, I-3) = AWW
          A(I-1, I-2) = AW
          A(I-1, I-1) = AP
          A(I-1, I) = AE
        ELSEIF(I .EQ. N-1) THEN
          A(I-1, I-3) = AWW
          A(I-1, I-2) = AW
          A(I-1, I-1) = AP - AEE
        ELSE
          A(I-1,I-3) = AWW
          A(I-1, I-2) = AW
          A(I-1, I-1) = AP
          A(I-1, I) = AE
          A(I-1, I+1) = AEE
        END IF
80    CONTINUE

C Find eigenvalues and vectors of A

      CALL DEVCRG (N1, A, LDA, EVAL, EVEC, LDEVEC)

C     CALL DWRCRN ('EVAL', 1, N1, EVAL, 1, 0)
      OPEN(1, FILE='EIGVAL.TXT', STATUS='UNKNOWN')
      DO 120 I = 1, N1
        WRITE(1,300) -REAL(EVAL(I))
120   CONTINUE

      OPEN(2, FILE='EIGVECTOR1.TXT', STATUS='UNKNOWN')
      OPEN(3, FILE='EIGVECTOR2.TXT', STATUS='UNKNOWN')
      OPEN(4, FILE='EIGVECTOR3.TXT', STATUS='UNKNOWN')
      OPEN(5, FILE='EIGVECTOR4.TXT', STATUS='UNKNOWN')
      WRITE(2,200) X(1), 0D0
```

```
      WRITE(3,200) X(1), 0D0
      WRITE(4,200) X(1), 0D0
      WRITE(5,200) X(1), 0D0
      DO 90 I = 1,N-2
        WRITE(2, 200) X(I+1), REAL(EVEC(I,N-2))
        WRITE(3, 200) X(I+1), REAL(EVEC(I,N-3))
        WRITE(4, 200) X(I+1), REAL(EVEC(I,N-4))
        WRITE(5, 200) X(I+1), REAL(EVEC(I,N-5))
90    CONTINUE
      WRITE(2,200) X(N), 0D0
      WRITE(3,200) X(N), 0D0
      WRITE(4,200) X(N), 0D0
      WRITE(5,200) X(N), 0D0
      CLOSE(1)
      CLOSE(2)
      CLOSE(3)
      CLOSE(4)
      CLOSE(5)

200   FORMAT(2D25.16)
300   FORMAT(D25.16)
      END
```

C.4. Program for Simulating Evolution of an Evaporating Film

```
/*Simulation of Evolution of an Evaporating Film
  Description: dh / dt + 1 / 3 (h^3hxxx)x + E / (k + h) = 0
        Boundary conditions are: h0(t=j) = 0, h0(t=j+1) = -E/K*dt, hxx(0) = 0,
                        hx(n) = 0, hxxx(n) = 0.
        Initial Shape: a straight line with slope 1 followed by a circular arc
                then extends uniformly to L = 25
        Using Crank Nicholson and Newton-Raphson method
        Note: the index of x, h start at 1
        Redistribution using 4 points, x1 is determined by points 2, 3, 4
*/
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/timeb.h>
#include <time.h>
#include <string.h>
/*N is number of points = # interval + 1*/
#define N 4197
#define M1 2
#define M2 2
#define LENGTH 25.
#define ECONST .1
#define KCONST .1
#define NUM_TIME_STEPS 100000000
#define ITER_BETWEEN_RECORDS 10000
#define PI 2.*acos(0)

/*Read x and h values*/
void ReadXAndH(FILE *pFile, double dX[], double dH[])
{
  int i;
  for(i = 1 ; i <= N ; ++i) fscanf(pFile, "%lf%lf", &dX[i], &dH[i]);
}

/*Write x and h values*/
void WriteXAndH(FILE *pFile, double dX[], double dH[])
{
  int i ;
  for(i = 1 ; i <= N; ++i) fprintf(pFile, "%15.12lf %15.12lf\n", dX[i], dH[i]);
}

/*Set the initial value of x*/
double SetIniX(double dX[])
{
  int i;
```

```
  double dDx = LENGTH / (N - 1) ;

  for(i= 1 ; i <= N; ++i) dX[i] = dDx * (i - 1);
  return dDx;
}

/*Set the initial value of h*/
void SetIniH(double dX[], double dH[] , double dDX)
{
  int i;
  double dX0, dY0;
  dX0 = 1. + sqrt(0.5);
  dY0 = -1. * sqrt(0.5);
  for(i = 1; i <= N; ++i) {
    if(dX[i] <= 0.5) dH[i] = dX[i]; /*a straight line with slope 1*/
          else if(dX[i] > 0.5 && dX[i] < dX0)
      dH[i] = dY0 + sqrt(dX[i]*(2*dX0-dX[i])); /*circular arc*/
          else dH[i] = 1.; /*uniform film*/
  }
}

/*Calculate hx on every point*/
double CalHx(double dH[], int i, double dDx)
{
  double dHx = 0.;
  if(i == N) dHx = 0.;
  else dHx = (dH[i+1] - dH[i-1]) / 2. /dDx;
  return dHx;
}

/*Calculate hxx on every point*/
double CalHxx(double dH[], int i, double dDx)
{
  double dHxx = 0;
  if (i == N) /*hxx(i=n)*/
    dHxx = 2. * (dH[N-1] - dH[N]) / pow(dDx, 2);
  else
    dHxx = (dH[i+1] - 2. * dH[i] + dH[i-1]) / pow(dDx, 2);
  return dHxx;
}

/*Calculate hxxx on every point*/
double CalHxxx(double dH[], int i, double dDx)
{
  double dHxxx = 0.;
  if(i == 2) /*hxxx(i=1)*/
    dHxxx = (dH[4] - 2. * dH[3]  + dH[2]) / 2. / pow(dDx, 3);
  else if(i == N - 1)
```

```
    dHxxx = (dH[N-1] - 2. * dH[N] + 2. * dH[N-2] - dH[N-3]) / 2. / pow(dDx, 3);
  else if(i == N)
    dHxxx = 0.;
  else
    dHxxx = (dH[i+2] - 2. * dH[i+1] + 2. * dH[i-1] - dH[i-2]) / 2. / pow(dDx, 3);
  return dHxxx;
}

/*Calculate hxxxx on every point*/
double CalHxxxx(double *dH, int i, double dDx)
{
  double dHxxxx = 0.;
  if(i == 2)
    dHxxxx = (dH[4] - 4. * dH[3] + 5. * dH[2] - 2. * dH[1]) / pow(dDx, 4);
  else if(i == N - 1)
    dHxxxx = (7. * dH[N-1] - 4. * dH[N] - 4. * dH[N-2] + dH[N-3]) / pow(dDx, 4);
  else if(i == N)
    dHxxxx = 2. * (dH[N-2] - 4. * dH[N-1] + 3. * dH[N]) / pow(dDx, 4);
  else
    dHxxxx = (dH[i+2] - 4. * dH[i+1] + 6. * dH[i] - 4. * dH[i-1] + dH[i-2]) / pow(dDx, 4);
  return dHxxxx;
}

#define SWAP(a,b) {dum = (a); (a) = (b); (b) = dum;}
#define TINY 1.0e-20
/*Given a n*n band diagonal matrix A with m1 subdiagonal rows and m2
//superdiagonal rows, compactly stored in the array a[1..n][1..m1+m2+1],
//this routine constructs an LU decomposition of rowwise permutation of
//A. The upper triangular matrix replaces a, while the lower triangular
//returned in a1[1..n][1..m1]. indx[1..n] is an output vector which
//records the row permutation effected by the partial pivoting;
//d is output as 1/-1 depending on whether the number of row interchanges
//was even or odd, respectively. This routine is used in combination with
//banbks to solve band-diagonal sets of equations*/
void bandec(double a[N][M1+M2+2], unsigned long n, int m1, int m2, double
al[N][M1+1],
 unsigned long indx[], double *d)
{
  unsigned long i,j,k,l;
  int mm;
  double dum;
  mm = m1 + m2 + 1;
  l = m1;
  for(i = 1; i <= (unsigned long)m1; ++i) {
    for(j = (unsigned long)(m1+2-i); j <= (unsigned long)mm; ++j) a[i][j-l]=a[i][j];
    --l;
    for(j = (unsigned long)mm-l; j <= (unsigned long)mm; ++j) a[i][j]=0.0;
  }
```

```
  *d = 1.0;
  l = m1;
  for(k = 1; k <= n; ++k) {
    dum = a[k][1];
    i = k;
    if(l < n) ++l;
    for(j = k+1; j <= l; ++j) {
      if(fabs(a[j][1]) > fabs(dum)) {
        dum = a[j][1];
        i = j;
             }
           }
    indx[k] = i;
    if(dum == 0.0) a[k][1] = TINY;
    if(i != k) {
      *d = -(*d);
      for(j = 1; j <= (unsigned long)mm; ++j) SWAP(a[k][j],a[i][j])
           }
    for(i = k + 1; i <= l; ++i) {
      dum = a[i][1] / a[k][1];
      al[k][i-k] = dum;
      for(j = 2; j <= (unsigned long)mm; ++j) a[i][j-1] = a[i][j] - dum * a[k][j];
      a[i][mm] = 0.0;
    }
  }
}
#undef SWAP
#undef TINY
#define SWAP(a,b) {dum = (a); (a) = (b); (b) = dum;}
/*Given the arrays a, a1, and indx as returned from bandec, and given a right-hand
//side vector b[1..n], solves the band diagonal linear equations ax = b. The
//solution vector x overwrites b[1..n]. The other input arrays are not modified
//and can be left in place for successive calls with different right-hand sides*/
void banbks(double a[N][M1+M2+2], unsigned long n, int m1, int m2, double
al[N][M1+1],
  unsigned long indx[], double b[])
{
  unsigned long i,k,l;
  int mm;
  double dum;
  mm = m1 + m2 + 1;
  l = m1;
  for(k = 1; k <= n; ++k) {
    i = indx[k];
  if(i != k) SWAP(b[k],b[i])
  if (l < n) ++l;
  for(i = k+1; i <= l; ++i) b[i] -= al[k][i-k] * b[k];
  }
```

```
  l = 1;
  for(i = n; i >= 1; --i) {
    dum = b[i];
    for(k = 2; k <= l; ++k) dum -= a[i][k] * b[k+i-1];
    b[i] = dum / a[i][1];
    if(l < (unsigned int)mm) ++l;
  }
}
#undef SWAP

/*Check more iteration is needed or not.
  Return -1, then no more iteration.
  Return 1, still needs iteration
*/
int NeedMoreIter(double dResidual[])
{
  int i;
  double dErr = 0.;
  for(i = 2 ; i <= N ; ++i) dErr += fabs(dResidual[i-1]);
  if(dErr <= 1.e-7 ) return -1; /*O.K. No need to iterate anymore*/
  else return 1;
}

/*March forward for one time step*/
void Iteration(double dX[], double dH[], double dDx, double dDt, double dTime)
{
  double d, dHx, dHxx, dHxxx, dHxxxx, dAww, dAw, dAp, dAe, dAee, dDx2, dDx3,
dDx4, dH2, dH3;
  double Fi[N], dH0[N+1], A[N][M1+M2+2], a1[N][M1+1]; /*A is stored by a N * 5
matrix*/
  int i, nNumIters=0, nNoMoreIter = 1;
  unsigned long indx[N];

  for(i = 1; i <= N; ++i) dH0[i] = dH[i];
  dH[1] = -ECONST/KCONST * dDt;
  dDx2 = dDx * dDx;
  dDx3 = dDx2 * dDx;
  dDx4 = dDx3 * dDx;
  while(nNoMoreIter != -1) {
     nNumIters ++;
     if(nNumIters > 20) {
       printf("Number of iterations > 20\n");
       exit(-1);
      }
    for(i = 2; i <= N ; ++i) {
      dHx = CalHx(dH, i, dDx);
      dHxx = CalHxx(dH, i, dDx);
      dHxxx = CalHxxx(dH, i, dDx);
```

```
dHxxxx = CalHxxxx(dH, i, dDx);
dH2 = dH[i] * dH[i];
dH3 = dH2 * dH[i];
if(i == N)
   dAww = dH3 / 3. / dDx4;
else
  dAww = (-.25 * dHx + dH[i] / 6. / dDx) * dH2 / dDx3;
if(i == N)
  dAw = -4. * dH3 / 3. / dDx4;
else
  dAw = (-.25 * dHxxx + .5 / dDx2 * dHx - 2. * dH[i] / 3. / dDx3 ) * dH2 / dDx;
if(i == 2)
  dAp = 1./dDt+ dH[i] * dHx * dHxxx + .25 / dDx3 * dH2 * dHx+.5 * dH2 * dHxxxx
      + 5. * dH3 / 6. / dDx4 - .5 * ECONST / (KCONST + dH[i]) / (KCONST + dH[i]);
else if(i == N-1)
  dAp = 1./dDt+dH[i]* dHx * dHxxx + .25 / dDx3 * dH2 * dHx + .5 * dH2 * dHxxxx
         + 7.*dH3/6. / dDx4 - .5 * ECONST / (KCONST + dH[i]) / (KCONST + dH[i]);
else if(i == N)
  dAp = 1. / dDt + .5 * dH2 * dHxxxx + dH3 / dDx4
         - .5 * ECONST / (KCONST + dH[i]) / (KCONST + dH[i]);
else
   dAp = 1. / dDt + dH[i] * dHx * dHxxx + .5 * dH2 * dHxxxx + dH3 / dDx4
          - .5 * ECONST / (KCONST + dH[i]) / (KCONST + dH[i]);
   dAe = (.25 * dHxxx - .5 / dDx2 * dHx  - 2. * dH[i] / 3. / dDx3) * dH2 / dDx;
   dAee = (.25 * dHx + dH[i] / 6. / dDx) * dH2 / dDx3;

 if(i == 2) {
  A[1][1] = A[1][2] = 0.;
  A[1][3] = dAp;
  A[1][4] = dAe;
  A[1][5] = dAee;
}
else if(i == 3) {
  A[2][1] = 0.;
  A[2][2] = dAw;
  A[2][3] = dAp;
  A[2][4] = dAe;
  A[2][5] = dAee;
 }
else if(i == N-1) {
  A[N-2][1] = dAww;
  A[N-2][2] = dAw;
  A[N-2][3] = dAp;
  A[N-2][4] = dAe;
  A[N-2][5] = 0.;
}
else if(i == N) {
  A[N-1][1] = dAww;
```

```
        A[N-1][2] = dAw;
        A[N-1][3] = dAp;
        A[N-1][4] = A[N-1][5] = 0.;
      }
      else{
        A[i-1][1] = dAww;
        A[i-1][2] = dAw;
        A[i-1][3] = dAp;
        A[i-1][4] = dAe;
        A[i-1][5] = dAee;
      }
      dH0[1] = 0.;
      Fi[i-1] = -((dH[i] - dH0[i]) / dDt + .5 *(dH2 * dHx * dHxxx + dH3 * dHxxxx / 3.
                  + ECONST / (KCONST + dH[i])));
      dHx = CalHx(dH0, i, dDx);
      dHxx = CalHxx(dH0, i, dDx);
      dHxxx = CalHxxx(dH0, i, dDx);
      dHxxxx = CalHxxxx(dH0, i, dDx);
      dH2 = dH0[i] * dH0[i];
      dH3 = dH2 * dH0[i];
      Fi[i-1] += -.5 * (dH2 * dHx * dHxxx + dH3 * dHxxxx / 3.
                        + ECONST / (KCONST + dH0[i]));
    }
    bandec(A, N-1, M1, M2, a1, indx, &d);
    banbks(A, N-1, M1, M2, a1, indx, Fi);
    for(i = 2; i <= N; ++i) dH[i] += Fi[i-1];
    /*check if more iteration is needed or not*/
    nNoMoreIter = NeedMoreIter(Fi);
  }
}

/*Search for the first point(starting from i = 1) whose h < 0
  Return -1: All h > 0 and pIndex is NULL
  Return 1: at least one point h < 0 and pIndex points to that point
*/
int FindHLessThanZero(double dH[], int *pIndex)
{
  int i, nIsLess = -1;
  pIndex = NULL;
  for(i = 1 ; i <= N; ++i)
    if(dH[i] < 0.) {
      nIsLess = 1;
      *pIndex = i;
      break;
          }
  return nIsLess ;
}
```

```
/*Redistribute all the grid points to get uniform grid at each time step
  Use a 4th order polynomial to fit the curve: h = a + b*(x-x2) + c*(x-x2)^2 + d*(x-x2)^3
*/
void RedistributePoints(double dX[], double dH[], double *dDx)
{
  int i, j;
double dXi = 0., dX0[N+1], dH0[N+1], dA, dB, dC, dD, dX1, dX2, dX3, dX4, dH1,
dH2, dH3, dH4,dt, dt1, dt2, dt3, temp1, temp2, temp3, temp4, temp5, temp6;
  (*dDx) = (dX[N] - dX[1]) / (N-1);
  dX0[1] = dX[1];
  j = 3;
  for(i = 2; i < N; ++i) {
    dXi = dX0[i-1] + (*dDx);
    dX0[i] = dXi;
    while(dX[j] - dXi <= 1.e-6) j++;
   dX1 = dX[j-2];
   dX2 = dX[j-1];
   dX3 = dX[j];
    if(i != N-1) dX4 = dX[j+1];
    else dX4 = dX[j] + (*dDx);
    dH1 = dH[j-2];
    dH2 = dH[j-1];
    dH3 = dH[j];
     if(i != N-1) dH4 = dH[j+1];
     else dH4 = dH[j-1];
     dA = dH2;
          temp1 = pow(dX1-dX2,2);
          temp2 = temp1 * (dX1-dX2);
          temp3 = pow(dX3-dX2,2);
          temp4 = temp3 * (dX3-dX2);
          temp5 = pow(dX4-dX2,2);
          temp6 = temp5 * (dX4-dX2);
          dt = (dX1-dX2)*(temp3*temp6 - temp5*temp4) - (dX3-dX2)*(temp1*temp6 -
          temp5*temp2) + (dX4-dX2)*(temp1*temp4 - temp2*temp3);
          dt1 = (dH1-dH2)*(temp3*temp6 - temp4*temp5) - (dH3-dH2)*(temp1*temp6 -
          temp5*temp2) + (dH4-dH2)*(temp1*temp4 - temp2*temp3);
          dt2 = (dX1-dX2)*((dH3-dH2)*temp6 - (dH4-dH2)*temp4) - (dX3-dX2)*((dH1-
          dH2)*temp6 - (dH4-dH2)*temp2) + (dX4-dX2)*((dH1-dH2)*temp4 - (dH3-
          dH2)*temp2);
          dt3 = (dX1-dX2)*((dH4-dH2)*temp3 - (dH3-dH2)*temp5) - (dX3-dX2)*((dH4-
          dH2)*temp1 - (dH1-dH2)*temp5) + (dX4-dX2)*((dH3-dH2)*temp1 - (dH1-
          dH2)*temp3);
          dB = dt1 / dt;
          dC = dt2 / dt;
          dD = dt3 / dt;
          dH0[i] = dA + dB*(dXi-dX2) + dC*(dXi-dX2)*(dXi-dX2) + dD*(dXi-
          dX2)*(dXi-dX2)*(dXi-dX2);
          if(i == 2 && dH0[i] < 1.e-15)
```

```
            dH0[i] = dH[i]*(*dDx)/(dX[i]-dX[i-1]);
  }
  for(i = 2; i < N; ++i) {
    dX[i] = dX0[i];
     dH[i] = dH0[i];
  }
}

/*Determine the contact point position
  Use a 2nd order polynomial to fit the curve: h = c*(x-x2)^2 + b*(x-x2) + a
*/
void CalX1(double dX[], double dH[], double dDt)
{
  double dA, dB, dC, temp,temp2, x11, x12;
  temp = (dX[1]-dX[2])*pow(dX[3]-dX[2],2)-(dX[3]-dX[2])*pow(dX[1]-dX[2],2);
  dA = dH[2];
  dB = ((-dH[2]-ECONST/KCONST*dDt)*pow(dX[3]-dX[2],2)-(dH[3]-
dH[2])*pow(dX[1]-dX[2],2))/temp;
  dC = ((dX[1]-dX[2])*(dH[3]-dH[2])+(dX[3]-
dX[2])*(dH[2]+ECONST/KCONST*dDt))/temp;
  if(fabs(dC) > 1.e-15) {
    temp2 = sqrt(dB*dB - 4*dA*dC);
    x11 = dX[2] + (-dB+temp2)/2/dC;
    x12 = dX[2] + (-dB-temp2)/2/dC;
    if(x11 < x12)
      if(x12 > dX[2]) dX[1] = x11;
      else dX[1] = x12;
    else
      if(x11 > dX[2]) dX[1] = x12;
      else dX[1] = x11;
  } /*parabola*/
  else /*linear*/
          dX[1] = dX[2] + (-ECONST/KCONST*dDt - dA) / dB;
}

/*- main program-------------------------------------*/
main()
{
  int i, ii, nChoice;
  long li;
  double dDx, dDt=1.e-7, dTime = 0., dX[N+1],dH[N+1], tempX[N+1],  tempH[N+1];
  FILE *pFile = NULL, *pFile1 = NULL;
  char fileName[200][6], fn[200] = "";

  /* Assign the file names*/
  for(i = 1; i <= 200; ++i) sprintf(fileName[i-1], "%d", i*100);

  nChoice = 2; /*nChoice = 1,start from scrach, nChoice = 2, start from previousprofile*/
```

```
if(nChoice == 2) {
  pFile = fopen("t.txt", "r");
  if(pFile == NULL) {
    printf("open t.txt error!\n");
    exit(-1);
  }
  fscanf(pFile, "%lf", &dTime);
  fclose(pFile);
}

if(nChoice == 1) {
  dDx = SetIniX(dX); /*set initial dX*/
  SetIniH(dX, dH, dDx); /*set initial dH*/
  pFile = fopen("h1.txt", "w");
  WriteXAndH(pFile, dX, dH);
  fclose(pFile);
}
else {/*nChoice = 2, read initial profile from a file*/
  pFile = fopen("h.txt", "r");
  ReadXAndH(pFile, dX, dH);
  dDx = dX[2] - dX[1];
  fclose(pFile);
}
RedistributePoints(dX, dH, &dDx);

for(li = 0; li < NUM_TIME_STEPS; ++li) {
  dTime += dDt;
   for(ii = 0; ii <= N; ++ii) {
      tempX[ii] = dX[ii];
      tempH[ii] = dH[ii];
  }
  Iteration(dX, dH, dDx, dDt, dTime);
  dH[1] = 0.;
  CalX1(dX, dH, dDt);

  /*check if dH < 0*/
  for(ii = 2; ii <= N; ++ii)     {
    if(dH[ii] < 0.) {
      pFile = fopen("beforeHLess0.txt", "w");
      WriteXAndH(pFile, tempX, tempH);
      fprintf(pFile, "%lf\n", dTime);
      fclose(pFile);
      pFile = fopen("dropProfile.txt", "w");
      WriteXAndH(pFile, dX, dH);
      fclose(pFile);
      printf("End iter = %d\n", i);
      exit(-1);
     }
```

```
    }
   RedistributePoints(dX, dH, &dDx);

   /*Write intermediate results into specified files*/
   if ((li+1) % ITER_BETWEEN_RECORDS == 0) {
     strcpy(fn, "");
     strcat(fn, fileName[(li+1)/ITER_BETWEEN_RECORDS-1]);
     strcat(fn, ".txt");
     pFile = fopen(fn, "w");
     WriteXAndH(pFile, dX, dH);
     fprintf(pFile, "%lf\n", dTime);
     fclose(pFile);
     pFile = fopen("t.txt", "w");
     fprintf(pFile, "%lf", dTime);
     fclose(pFile);
    }
   if((li+1) % 200 == 0) printf("%d\n", li);
   pFile = fopen("iter.txt", "w");
   fprintf(pFile, "%d\n", li);
   fclose(pFile);
  }
}
```

C.5. Program for Finding Self-similar Film Profile

```
/* Use shooting method to solve the Self-similar Film Profile*/
/* Use Runge-Kutta method */
/* Shoot from X = 0+Dx */

#include "stdio.h"
#include "stdlib.h"
#include "math.h"
#include "string.h"
#define N 30001 /*number of points*/
#define L 15. /*computational domain*/
#define PI 2.*acos(0)
/*for h~x++x^2*ln(x)+x^2
#define C2 -0.4539349490683491
#define C1 1.4808972


/* for h~x+x2lnx+x2+x3lnx*/ /*for dx = 0.01*/
/*#define C1 1.48089
#define C2 -0.4536257435696925
*/
/* for h~x+x2lnx+x2+x3lnx*/ /*for dx = 0.005*/

//dx = 0.001
//#define C1 1.4807974L
//#define C2 -0.45515401246977L

//dx = 0.0005
#define C1 1.4807974L
#define C2 -0.45525347389119377L
/*Initialize the boundary conditions at x = dx*/

void Initialize(long double u1[], long double u2[], long double u3[], long double u4[],
long double dx)
{
/*for h ~ x+x^2lnx+x^2*/
/* long double c12, lndx, dx2;
  c12 = C1 * C1;
  dx2 = dx * dx;
  lndx = log(dx);
  u1[1] = C1*dx+3./16./c12*dx2*lndx+(C2-3./16./c12)*dx2;
  u2[1] = C1+3./16./c12*(2.*dx*lndx+dx)+(2.*C2-3./8./c12)*dx;
  u3[1] = 3./8./c12*lndx+2.*C2+3./16./c12;
  u4[1] = 3./8./c12/dx;
*/
```

```
/* for h ~ x+x2lnx+x2+x3lnx*/
  long double c12, c15, dx2, dx3, lndx;
  c12 = C1*C1;
  c15 = c12*c12*C1;
  dx2 = dx*dx;
  dx3 = dx2*dx;
  lndx = log(dx);
  u1[1] = C1*dx+3./16./c12*dx2*lndx+(C2-3./16./c12)*dx2+1./64./c15*dx3*lndx;
  u2[1] = C1+3./16./c12*(2.*dx*lndx+dx)+(2.*C2-
3./8./c12)*dx+1./64./c15*(3.*dx2*lndx+dx2);
  u3[1] = 3./8./c12*lndx+2.*C2+3./16./c12+1./64./c15*(6.*dx*lndx+7.*dx);
  u4[1] = 3./8./c12/dx+1./64./c15*(6.*lndx+13.);

 /* u1[1] = 0.00321265486031;
  u2[1] = 1.54299193670663;
  u3[1] = -31.31565461735417;
  u4[1] = 27937.58605078799900;
  */
}

/*calculate f for ode u1,u2,u3,u5,u6,u7,u9,u10,u11*/
long double f1(long double ui_1)
{
  return ui_1;
}

/*calcuate f4 for ode u4*/
long double f2(long double x, long double u1, long double u2, long double u4)
{
  long double u12, u13, temp1, temp2;
  u12 = u1*u1;
  u13 = u12*u1;
  temp1 = .75*x*u2;
  temp2 = 3.*u12*u2*u4;
  return (temp1-temp2)/u13;
}

/*Runge-Kutta method*/
void Runge_Kutta(int i, long double x, long double u1[], long double u2[], long double
u3[], long double u4[])
{
  long double h, k11,k12,k13,k14,k21,k22,k23,k24,k31,k32,k33,k34,k41,k42,k43,k44;
  h = L / (N-1); /*step for Runge-Kutta method*/
  k11 = f1(u2[i]);
  k21 = f1(u3[i]);
  k31 = f1(u4[i]);
  k41 = f2(x, u1[i], u2[i], u4[i]);
  k12 = f1(u2[i]+.5*h*k21);
```

```
  k22 = f1(u3[i]+.5*h*k31);
  k32 = f1(u4[i]+.5*h*k41);
  k42 = f2(x+.5*h, u1[i]+.5*h*k11,u2[i]+.5*h*k21,u4[i]+.5*h*k41);
  k13 = f1(u2[i]+.5*h*k22);
  k23 = f1(u3[i]+.5*h*k32);
  k33 = f1(u4[i]+.5*h*k42);
  k43 = f2(x+.5*h, u1[i]+.5*h*k12,u2[i]+.5*h*k22,u4[i]+.5*h*k42);
  k14 = f1(u2[i]+h*k23);
  k24 = f1(u3[i]+h*k33);
  k34 = f1(u4[i]+h*k43);
  k44 = f2(x+h,u1[i]+h*k13,u2[i]+h*k23,u4[i]+h*k43);
  u1[i+1] = u1[i] + h/6.*(k11+2.*(k12+k13)+k14);
  u2[i+1] = u2[i] + h/6.*(k21+2.*(k22+k23)+k24);
  u3[i+1] = u3[i] + h/6.*(k31+2.*(k32+k33)+k34);
  u4[i+1] = u4[i] + h/6.*(k41+2.*(k42+k43)+k44);
}

/*initialize x*/
void SetX(long double x[])
{
  int i;
  long double step = (long double)L / (N-1);
  for(i = 0; i < N; ++i) x[i] = step*i;
}

main()
{
  int i;
  //long double x[N],u1[N], u2[N], u3[N], u4[N], temp;
  long double *x, *u1, *u2, *u3, *u4, temp, dx;
  FILE *fp, *fp1, *fp2, *fp3, *fp4;
  u1 = (long double*)calloc(N, sizeof(long double));
  u2 = (long double*)calloc(N, sizeof(long double));
  u3 = (long double*)calloc(N, sizeof(long double));
  u4 = (long double*)calloc(N, sizeof(long double));
  x = (long double*)calloc(N, sizeof(long double));
  dx = L / (N-1);

  SetX(x);
  Initialize(u1,u2,u3,u4,dx);

  /*use Runge-Kutta to integrate each 1st order ode*/
  for(i = 1; i <= N-2; ++i)
  {
    Runge_Kutta(i,x[i],u1,u2,u3,u4);
  }
  fp = fopen("h.txt", "w");
  fp1 = fopen("hxxx.txt", "w");
```

```
 fp2 = fopen("hx.txt", "w");
 fp3 = fopen("hxx.txt", "w");
 fp4 = fopen("hxxxx.txt", "w");


 for(i = 0; i < N; ++i) {
        fprintf(fp, "%lf %15.14lf\n",x[i], u1[i]);
        fprintf(fp1,"%15.14lf\n", u4[i]);
        fprintf(fp2,"%15.14lf\n", u2[i]);
        fprintf(fp3,"%15.14lf\n", u3[i]);
        temp = (0.75*x[i]*u2[i]-3.*u1[i]*u1[i]*u2[i]*u4[i])/(u1[i]*u1[i]*u1[i]);
        fprintf(fp4, "%15.14lf\n", temp);

 }
  printf("%15.12lf\n", u1[N-1]);
 fclose(fp);
 fclose(fp1);
 fclose(fp2);
 fclose(fp3);
 fclose(fp4);
 free(u1);
 free(u2);
 free(u3);
 free(u4);
 free(x);

}
```

C.6. Program for Simulating Evolution of a Step Film

```
/* Simulate evolution of a step film */
/* Description: No evaporation
        dh / dt + 1 / 3 (h^3hxxx)x = 0
        Boundary conditions are: h0(t=j) = 0, h0(t=j+1) = 0,
                        hxx(dx) = -(dHx/dt)/2/Hx^3*ln(dx),
                                                hxx(i=2)= (h2-2h1+h0)/dx2
                        hx(n) = 0, hxxx(n) = 0.
        Using Crank Nicholson and Newton-Raphson method
        Note: the index of x, h start at 1
        Redistribution using 4 points
        x1 is determined by points 2, 3, 4
*/

#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/timeb.h>
#include <time.h>
#include <string.h>
/*N is number of points = # interval + 1*/
#define N 10001
#define M1 2
#define M2 2
#define LENGTH 50.
#define DECONST 0.
#define CCONST 0.
#define NUM_TIME_STEPS 100000000
#define ITER_BETWEEN_RECORDS 10000
#define PI 2.*acos(0)

/*Read x and h values*/
void ReadXAndH(FILE *pFile, double dX[], double dH[])
{
  int i;
  for(i = 1 ; i <= N ; ++i) fscanf(pFile, "%lf%lf", &dX[i], &dH[i]);
}

/*Write x and h values*/
void WriteXAndH(FILE *pFile, double dX[], double dH[])
{
  int i ;
  for(i = 1 ; i <= N; ++i) fprintf(pFile, "%15.12lf %15.12lf\n", dX[i], dH[i]);
}

/*Set the initial value of x*/
double SetIniX(double dX[])
```

```
{
  int i;
  double dDx = LENGTH / (N - 1) ;

  for(i= 1 ; i <= N; ++i) dX[i] = dDx * (i - 1);
  return dDx;
}

/*Set the initial value of h*/
void SetIniH(double dX[], double dH[] , double dDX)
{
  int i;
  double dX0, dY0;

  dX0 = 1. + sqrt(0.5);
  dY0 = -1. * sqrt(0.5);
  for(i = 1; i <= N; ++i) {
    if(dX[i] <= 0.5) dH[i] = dX[i]; /*a straight line with slope 1*/
          else if(dX[i] > 0.5 && dX[i] < dX0)
      dH[i] = dY0 + sqrt(dX[i]*(2*dX0-dX[i])); /*circular arc*/
          else dH[i] = 1.; /*uniform film*/
  }
}

/*Calculate hx on every point*/
double CalHx(double dH[], int i, double dDx)
{
  double dHx = 0.;
  if(i == N) dHx = 0.;
  else dHx = (dH[i+1] - dH[i-1]) / 2. /dDx;
  return dHx;
}

/*Calculate hxx on every point*/
double CalHxx(double dH[], int i, double dDx, double dOmega, double dHx)
{
  double dHxx = 0;
  if (i == 2)
          dHxx = -dOmega*.5/pow(dHx,3)*log(dDx);
  if (i == N) /*hxx(i=n)*/
    dHxx = 2. * (dH[N-1] - dH[N]) / pow(dDx, 2);
  else
    dHxx = (dH[i+1] - 2. * dH[i] + dH[i-1]) / pow(dDx, 2);
  return dHxx;
}

/*Calculate hxxx on every point*/
double CalHxxx(double dH[], int i, double dDx, double dHxx)
```

```
{
  double dHxxx = 0., dDx2;
  dDx2 = dDx * dDx;
  if(i == 2) /*hxxx(i=1)*/
    dHxxx = (dH[4] - 2. * dH[3]  + dH[2] - dHxx*dDx2) / 2. / pow(dDx, 3);
  else if(i == N - 1)
    dHxxx = (dH[N-1] - 2. * dH[N] + 2. * dH[N-2] - dH[N-3]) / 2. / pow(dDx, 3);
  else if(i == N)
    dHxxx = 0.;
  else
    dHxxx = (dH[i+2] - 2. * dH[i+1] + 2. * dH[i-1] - dH[i-2]) / 2. / pow(dDx, 3);
  return dHxxx;
}

/*Calculate hxxxx on every point*/
double CalHxxxx(double *dH, int i, double dDx, double dHxx)
{
  double dHxxxx = 0., dDx2;
  dDx2 = dDx * dDx;
  if(i == 2)
    dHxxxx = (dH[4] - 4. * dH[3] + 5. * dH[2] + dHxx*dDx2) / pow(dDx, 4);
  else if(i == N - 1)
    dHxxxx = (7. * dH[N-1] - 4. * dH[N] - 4. * dH[N-2] + dH[N-3]) / pow(dDx, 4);
  else if(i == N)
    dHxxxx = 2. * (dH[N-2] - 4. * dH[N-1] + 3. * dH[N]) / pow(dDx, 4);
  else
    dHxxxx = (dH[i+2] - 4. * dH[i+1] + 6. * dH[i] - 4. * dH[i-1] + dH[i-2]) / pow(dDx, 4);
  return dHxxxx;
}

#define SWAP(a,b) {dum = (a); (a) = (b); (b) = dum;}
#define TINY 1.0e-20
/*Given a n*n band diagonal matrix A with m1 subdiagonal rows and m2
//superdiagonal rows, compactly stored in the array a[1..n][1..m1+m2+1],
//this routine constructs an LU decomposition of rowwise permutation of
//A. The upper triangular matrix replaces a, while the lower triangular
//returned in a1[1..n][1..m1]. indx[1..n] is an output vector which
//records the row permutation effected by the partial pivoting;
//d is output as 1/-1 depending on whether the number of row interchanges
//was even or odd, respectively. This routine is used in combination with
//banbks to solve band-diagonal sets of equations*/
void bandec(double a[N][M1+M2+2], unsigned long n, int m1, int m2, double
al[N][M1+1],
 unsigned long indx[], double *d)
{
  unsigned long i,j,k,l;
  int mm;
  double dum;
```

```
  mm = m1 + m2 + 1;
  l = m1;
  for(i = 1; i <= (unsigned long)m1; ++i) {
    for(j = (unsigned long)(m1+2-i); j <= (unsigned long)mm; ++j) a[i][j-l]=a[i][j];
    --l;
    for(j = (unsigned long)mm-l; j <= (unsigned long)mm; ++j) a[i][j]=0.0;
  }
  *d = 1.0;
  l = m1;
  for(k = 1; k <= n; ++k) {
    dum = a[k][1];
    i = k;
    if(l < n) ++l;
    for(j = k+1; j <= l; ++j) {
      if(fabs(a[j][1]) > fabs(dum)) {
        dum = a[j][1];
        i = j;
            }
          }
    indx[k] = i;
    if(dum == 0.0) a[k][1] = TINY;
    if(i != k) {
      *d = -(*d);
      for(j = 1; j <= (unsigned long)mm; ++j) SWAP(a[k][j],a[i][j])
          }
    for(i = k + 1; i <= l; ++i) {
      dum = a[i][1] / a[k][1];
      al[k][i-k] = dum;
      for(j = 2; j <= (unsigned long)mm; ++j) a[i][j-1] = a[i][j] - dum * a[k][j];
      a[i][mm] = 0.0;
          }
  }
}
#undef SWAP
#undef TINY

#define SWAP(a,b) {dum = (a); (a) = (b); (b) = dum;}
/*Given the arrays a, a1, and indx as returned from bandec, and given a right-hand
//side vector b[1..n], solves the band diagonal linear equations ax = b. The
//solution vector x overwrites b[1..n]. The other input arrays are not modified
//and can be left in place for successive calls with different right-hand sides*/
void banbks(double a[N][M1+M2+2], unsigned long n, int m1, int m2, double
al[N][M1+1],
  unsigned long indx[], double b[])
{
  unsigned long i,k,l;
  int mm;
```

```
  double dum;

  mm = m1 + m2 + 1;
  l = m1;
  for(k = 1; k <= n; ++k) {
    i = indx[k];
  if(i != k) SWAP(b[k],b[i])
  if (l < n) ++l;
  for(i = k+1; i <= l; ++i) b[i] -= al[k][i-k] * b[k];
  }
  l = 1;
  for(i = n; i >= 1; --i) {
    dum = b[i];
    for(k = 2; k <= l; ++k) dum -= a[i][k] * b[k+i-1];
    b[i] = dum / a[i][1];
    if(l < (unsigned int)mm) ++l;
  }
}
#undef SWAP

/*Check more iteration is needed or not.
  Return -1, then no more iteration.
  Return 1, still needs iteration
*/
int NeedMoreIter(double dResidual[])
{
  int i;
  double dErr = 0.;
  for(i = 2 ; i <= N ; ++i) dErr += fabs(dResidual[i-1]);
  if(dErr <= 1.e-7 ) return -1; /*O.K. No need to iterate anymore*/
  else return 1;
}

/*March forward for one time step*/
void Iteration(double dX[], double dH[], double dDx, double dDt, double dTime, double
dOmega, double hx)
{
  double d, dHx, dHxx, dHxxx, dHxxxx, dAww, dAw, dAp, dAe, dAee, dDx2, dDx3,
dDx4, dH2, dH3;
  double Fi[N], dH0[N+1], A[N][M1+M2+2], a1[N][M1+1]; /*A is stored by a N * 5
matrix*/
  int i, nNumIters=0, nNoMoreIter = 1;
  unsigned long indx[N];

  for(i = 1; i <= N; ++i) dH0[i] = dH[i];
  dH[1] = 0.;
  dDx2 = dDx * dDx;
  dDx3 = dDx2 * dDx;
```

```
 dDx4 = dDx3 * dDx;

 while(nNoMoreIter != -1) {
  nNumIters ++;
  if(nNumIters > 20) {
  printf("Number of iterations > 20\n");
   exit(-1);
  }
  for(i = 2; i <= N ; ++i) {
   dHx = CalHx(dH, i, dDx);
   dHxx = CalHxx(dH, i, dDx, dOmega, hx);
   dHxxx = CalHxxx(dH, i, dDx, dHxx);
   dHxxxx = CalHxxxx(dH, i, dDx, dHxx);
   dH2 = dH[i] * dH[i];
  dH3 = dH2 * dH[i];
   if(i == N)
    dAww = dH3 / 3. / dDx4;
   else
    dAww = (-.25 * dHx + dH[i] / 6. / dDx) * dH2 / dDx3;
   if(i == N)
    dAw = -4. * dH3 / 3. / dDx4;
  else
     dAw = (-.25 * dHxxx + .5 / dDx2 * dHx - 2. * dH[i] / 3. / dDx3 ) * dH2 / dDx;
  if(i == 2)
     dAp = 1./dDt + dH[i] * dHx * dHxxx + .25 / dDx3 * dH2 * dHx + .5 * dH2 *
dHxxxx+ 5. * dH3 / 6. / dDx4 - .5 * CCONST * DECONST / (1 + CCONST * dH[i]) / (1
+ CCONST * dH[i]);
 else if(i == N-1)
     dAp = 1. / dDt + dH[i] * dHx * dHxxx + .25 / dDx3 * dH2 * dHx + .5 * dH2 *
dHxxxx + 7. * dH3 / 6. / dDx4 - .5 * CCONST * DECONST / (1 + CCONST * dH[i]) /
(1 + CCONST * dH[i]);
 else if(i == N)
    dAp = 1. / dDt + .5 * dH2 * dHxxxx + dH3 / dDx4 - .5 * CCONST * DECONST / (1
+ CCONST * dH[i]) / (1 + CCONST * dH[i]);
 else
  dAp = 1. / dDt + dH[i] * dHx * dHxxx + .5 * dH2 * dHxxxx + dH3 / dDx4
   - .5 * CCONST * DECONST / (1. + CCONST * dH[i]) / (1. + CCONST * dH[i]);
 dAe = (.25 * dHxxx - .5 / dDx2 * dHx  - 2. * dH[i] / 3. / dDx3) * dH2 / dDx;
 dAee = (.25 * dHx + dH[i] / 6. / dDx) * dH2 / dDx3;

 if(i == 2) {
     A[1][1] = A[1][2] = 0.;
     A[1][3] = dAp;
     A[1][4] = dAe;
     A[1][5] = dAee;
 }
else if(i == 3) {
     A[2][1] = 0.;
```

```
        A[2][2] = dAw;
        A[2][3] = dAp;
        A[2][4] = dAe;
        A[2][5] = dAee;
   }
   else if(i == N-1) {
        A[N-2][1] = dAww;
        A[N-2][2] = dAw;
        A[N-2][3] = dAp;
        A[N-2][4] = dAe;
       A[N-2][5] = 0.;
   }
   else if(i == N) {
        A[N-1][1] = dAww;
        A[N-1][2] = dAw;
        A[N-1][3] = dAp;
        A[N-1][4] = A[N-1][5] = 0.;
   }
   else{
        A[i-1][1] = dAww;
        A[i-1][2] = dAw;
        A[i-1][3] = dAp;
        A[i-1][4] = dAe;
        A[i-1][5] = dAee;
   }
   dH0[1] = 0.;
   Fi[i-1] = -((dH[i] - dH0[i]) / dDt + .5 *(dH2 * dHx * dHxxx + dH3 * dHxxxx / 3.
                    + DECONST / (1 + CCONST * dH[i])));
  dHx = CalHx(dH0, i, dDx);
  dHxx = CalHxx(dH0, i, dDx, dOmega, hx);
  dHxxx = CalHxxx(dH0, i, dDx, dHxx);
  dHxxxx = CalHxxxx(dH0, i, dDx, dHxx);
 dH2 = dH0[i] * dH0[i];
 dH3 = dH2 * dH0[i];
 Fi[i-1] += -.5 * (dH2 * dHx * dHxxx + dH3 * dHxxxx / 3. + DECONST / (1 + CCONST
 * dH0[i]));
 }
    bandec(A, N-1, M1, M2, a1, indx, &d);
    banbks(A, N-1, M1, M2, a1, indx, Fi);
    for(i = 2; i <= N; ++i) dH[i] += Fi[i-1];

    /*check if more iteration is needed or not*/
    nNoMoreIter = NeedMoreIter(Fi);
   }
 }

/*Search for the first point(starting from i = 1) whose h < 0
  Return -1: All h > 0 and pIndex is NULL
```

```
  Return 1: at least one point h < 0 and pIndex points to that point
*/
int FindHLessThanZero(double dH[], int *pIndex)
{
  int i, nIsLess = -1;
  pIndex = NULL;
  for(i = 1 ; i <= N; ++i)
    if(dH[i] < 0.) {
      nIsLess = 1;
      *pIndex = i;
      break;
           }
  return nIsLess ;
}

/*Redistribute all the grid points to get uniform grid at each time step
  Use a 4th order polynomial to fit the curve: h = a + b*(x-x2) + c*(x-x2)^2 + d*(x-x2)^3
*/
void RedistributePoints(double dX[], double dH[], double *dDx)
{
  int i, j;
  double dXi = 0., dX0[N+1], dH0[N+1], dA, dB, dC, dD, dX1, dX2, dX3, dX4, dH1,
dH2, dH3, dH4,
    dt, dt1, dt2, dt3, temp1, temp2, temp3, temp4, temp5, temp6;
  (*dDx) = (dX[N] - dX[1]) / (N-1);
  dX0[1] = dX[1];

  j = 3;
  for(i = 2; i < N; ++i) {
    dXi = dX0[i-1] + (*dDx);
          dX0[i] = dXi;
          while(dX[j] - dXi <= 1.e-6) j++;
          dX1 = dX[j-2];
          dX2 = dX[j-1];
          dX3 = dX[j];
          if(i != N-1) dX4 = dX[j+1];
          else dX4 = dX[j] + (*dDx);
          dH1 = dH[j-2];
          dH2 = dH[j-1];
          dH3 = dH[j];
          if(i != N-1) dH4 = dH[j+1];
          else dH4 = dH[j-1];
          dA = dH2;
          temp1 = pow(dX1-dX2,2);
          temp2 = temp1 * (dX1-dX2);
          temp3 = pow(dX3-dX2,2);
          temp4 = temp3 * (dX3-dX2);
          temp5 = pow(dX4-dX2,2);
```

```
        temp6 = temp5 * (dX4-dX2);

        dt = (dX1-dX2)*(temp3*temp6 - temp5*temp4) - (dX3-dX2)*(temp1*temp6 -
temp5*temp2) + (dX4-dX2)*(temp1*temp4 - temp2*temp3);
        dt1 = (dH1-dH2)*(temp3*temp6 - temp4*temp5) - (dH3-dH2)*(temp1*temp6 -
temp5*temp2) + (dH4-dH2)*(temp1*temp4 - temp2*temp3);
        dt2 = (dX1-dX2)*((dH3-dH2)*temp6 - (dH4-dH2)*temp4) - (dX3-dX2)*((dH1-
dH2)*temp6 - (dH4-dH2)*temp2) + (dX4-dX2)*((dH1-dH2)*temp4 - (dH3-
dH2)*temp2);
        dt3 = (dX1-dX2)*((dH4-dH2)*temp3 - (dH3-dH2)*temp5) - (dX3-dX2)*((dH4-
dH2)*temp1 - (dH1-dH2)*temp5) + (dX4-dX2)*((dH3-dH2)*temp1 - (dH1-
dH2)*temp3);
        dB = dt1 / dt;
        dC = dt2 / dt;
        dD = dt3 / dt;
        dH0[i] = dA + dB*(dXi-dX2) + dC*(dXi-dX2)*(dXi-dX2) + dD*(dXi-
dX2)*(dXi-dX2)*(dXi-dX2);
  }
  for(i = 2; i < N; ++i) {
    dX[i] = dX0[i];
        dH[i] = dH0[i];
  }
}

/*Determine the contact point position
  Use a 2nd order polynomial to fit the curve: h = c*(x-x2)^2 + b*(x-x2) + a
*/
void CalX1(double dX[], double dH[], double dDt)
{
  double dA, dB, dC, temp,temp2, x11, x12;
  temp = (dX[1]-dX[2])*pow(dX[3]-dX[2],2)-(dX[3]-dX[2])*pow(dX[1]-dX[2],2);
  dA = dH[2];
  dB = ((-dH[2]-DECONST*dDt)*pow(dX[3]-dX[2],2)-(dH[3]-dH[2])*pow(dX[1]-
dX[2],2))/temp;
  dC = ((dX[1]-dX[2])*(dH[3]-dH[2])+(dX[3]-dX[2])*(dH[2]+DECONST*dDt))/temp;
  if(fabs(dC) > 1.e-15) {
    temp2 = sqrt(dB*dB - 4*dA*dC);
    x11 = dX[2] + (-dB+temp2)/2/dC;
    x12 = dX[2] + (-dB-temp2)/2/dC;
    if(x11 < x12)
      if(x12 > dX[2]) dX[1] = x11;
            else dX[1] = x12;
    else
      if(x11 > dX[2]) dX[1] = x12;
            else dX[1] = x11;
  } /*parabola*/
  else /*linear*/
        dX[1] = dX[2] + (-DECONST*dDt - dA) / dB;
```

```
}

/*main program*/
main()
{
  int i, ii, nChoice;
  long li;
  double dDx, dOmega, hx, hxOld, dDt=1.e-5, dTime = 0., dX[N+1],dH[N+1],
tempX[N+1], tempH[N+1];
  FILE *pFile = NULL, *pFile1 = NULL;
  char fileName[100][6], fn[100] = "";

  /* Assign the file names*/
  for(i = 1; i <= 100; ++i) sprintf(fileName[i-1], "%d", i*100);
  nChoice = 2; /*nChoice = 1, start from scrach, nChoice = 2, start from
previous profile*/
  if(nChoice == 2) {
    pFile = fopen("t.txt", "r");
    if(pFile == NULL) {
      printf("open t.txt error!\n");
      exit(-1);
          }
    fscanf(pFile, "%lf", &dTime);
    fclose(pFile);
  }
  if(nChoice == 1) {
    dDx = SetIniX(dX); /*set initial dX*/
    SetIniH(dX, dH, dDx); /*set initial dH*/
    pFile = fopen("h1.txt", "w");
    WriteXAndH(pFile, dX, dH);
    fclose(pFile);
  }
  else {/*nChoice = 2, read initial profile from a file*/
    pFile = fopen("h.txt", "r");
    ReadXAndH(pFile, dX, dH);
    dDx = dX[2] - dX[1];
    fclose(pFile);
  }
  hx = 1.;
  dOmega = 0.;
  for(li = 0; li < NUM_TIME_STEPS; ++li) {
    dTime += dDt;
          for(ii = 0; ii <= N; ++ii) {
            tempX[ii] = dX[ii];
            tempH[ii] = dH[ii];
          }
          hxOld = hx;
          Iteration(dX, dH, dDx, dDt, dTime, dOmega, hx);
```

```
            dH[1] = 0.;
            dX[1] = 0.;
    /*update omega and hx*/
            hx = CalHx(dH, 2, dDx);
            dOmega = (hx - hxOld) / dDt;
    /*check if dH < 0*/
    for(ii = 2; ii <= N; ++ii)     {
      if(dH[ii] < 0.) {
        pFile = fopen("beforeHLess0.txt", "w");
        WriteXAndH(pFile, tempX, tempH);
        fprintf(pFile, "%lf\n", dTime);
        fclose(pFile);
        pFile = fopen("dropProfile.txt", "w");
        WriteXAndH(pFile, dX, dH);
        fclose(pFile);
        printf("End iter = %d\n", i);
        exit(-1);
             }
            }
  /*  RedistributePoints(dX, dH, &dDx);*/
    /*Write intermediate results into specified files*/
    if ((li+1) % ITER_BETWEEN_RECORDS == 0) {
      strcpy(fn, "");
      strcat(fn, fileName[(li+1)/ITER_BETWEEN_RECORDS-1]);
      strcat(fn, ".txt");
      pFile = fopen(fn, "w");
      WriteXAndH(pFile, dX, dH);
      fprintf(pFile, "%lf\n", dTime);
      fclose(pFile);
      pFile = fopen("t.txt", "w");
      fprintf(pFile, "%lf", dTime);
      fclose(pFile);
            }
            if((li+1) % 200 == 0) printf("%d\n", li);
    pFile = fopen("iter.txt", "w");
    fprintf(pFile, "%d\n", li);
    fclose(pFile);
  }
}
```

C.7. Program for Checking the Stability of Self-Similar Solution of a Retracting Film

```
C Checking the stability of the self-similar film solution
C Boundary conditions: x = 0, f=0, d2f/dXx=0 (No integration),  at x = infinity, f = 0,
df/dx=0

      EXTERNAL DEVCRG, DWRCRN

      DOUBLE PRECISION DX,DX3,DX4,WAVN1,
     +          WAVN2, WAVN3, GROW1, GROW2, GROW3
C GROW: GROWTH RATE
C WAVN: WAVE NUMBER
C Y(1) IS H0, Y(2) IS HOX

C CDP: DIMENSIONLESS PRESSURE DIFFERENCE (C)
C EPS: EPSILONE
      PARAMETER (N = 1501)
      DIMENSION X(N),H(N), HS(N), A(N-2,N-2)
      DOUBLE PRECISION X,H, HS, A, HS2, HS3, HSX, HSXX, HSXXX,
HSXXXX

      DOUBLE PRECISION AWW, AW, AP, AE, AEE
C VARIABLES FOR USING ISML SUBROUTINES
      INTEGER N1, LDA, LDEVEC
      PARAMETER (N1=N-2, LDA=N1, LDEVEC=N1)
      DOUBLE PRECISION C1
      PARAMETER (C1 = 1.48089D0)

      DOUBLE COMPLEX EVAL(N1), EVEC(LDEVEC,N1)

      OPEN(1, FILE='H.TXT', STATUS='UNKNOWN')
      OPEN(2, FILE='HX.TXT', STATUS='UNKNOWN')
      OPEN(3, FILE='HXX.TXT', STATUS='UNKNOWN')
      OPEN(4, FILE='HXXX.TXT', STATUS='UNKNOWN')
      OPEN(5,FILE='HXXXX.TXT',STATUS='UNKNOWN')

      DO 40 I = 1, N
        READ(1, *) X(I), HS(I)
40    CONTINUE

      CLOSE(1)

20    FORMAT(2D25.16)
30    FORMAT(1D25.16)
      DX = X(2)-X(1)
      DX3 = DX*DX*DX
```

```
      DX4 = DX3*DX
      ERROR = 1.0D-7


C DETERMINE AWW, AW, AP, AE, AEE
      DO 75 I = 1, N1
        DO 74 J = 1, N1
          A(I,J) = 0.D0
74      CONTINUE
75    CONTINUE
      DO 80 I = 2, N-1
        HS2 = HS(I)*HS(I)
        HS3 = HS2*HS(I)
        READ(2, *) HSX
        READ(3, *) HSXX
        READ(4, *) HSXXX
     READ(5, *) HSXXXX

        AWW = HS3/DX4 - 1.5D0*HS2*HSX/DX3
        AW = -4.D0*HS3/DX4+0.375D0*X(I)/DX-1.5D0*HS2*HSXXX/DX
   ++3D0*HS2*HSX/DX3
        AP = 6.D0*HS3/DX4+6.D0*HS(I)*HSX*HSXXX+3.D0*HS2*HSXXXX
        AE = -4.D0*HS3/DX4-0.375D0*X(I)/DX+1.5D0*HS2*HSXXX/DX
   +-3.D0*HS2*HSX/DX3
        AEE = HS3/DX4 + 1.5D0*HS2*HSX/DX3
        IF(I .EQ. 2) THEN
          A(I-1,I-1) = AP - AWW
          A(I-1,I) = AE
          A(I-1,I+1) = AEE
        ELSEIF(I .EQ. 3) THEN
          A(I-1,I-2) = AW
          A(I-1,I-1) = AP
          A(I-1, I) = AE
          A(I-1, I+1) = AEE
        ELSEIF(I .EQ. N-2) THEN
          A(I-1, I-3) = AWW
          A(I-1, I-2) = AW
          A(I-1, I-1) = AP
          A(I-1, I) = AE
        ELSEIF(I .EQ. N-1) THEN
          A(I-1, I-3) = AWW
          A(I-1, I-2) = AW
          A(I-1, I-1) = AP+AEE
        ELSE
          A(I-1,I-3) = AWW
          A(I-1, I-2) = AW
          A(I-1, I-1) = AP
          A(I-1, I) = AE
```

```
          A(I-1, I+1) = AEE
         END IF
80      CONTINUE

        CLOSE(2)
        CLOSE(3)
        CLOSE(4)
        CLOSE(5)

C Find eigenvalues and vectors of A

        CALL DEVCRG (N1, A, LDA, EVAL, EVEC, LDEVEC)

C       CALL DWRCRN ('EVAL', 1, N1, EVAL, 1, 0)
        OPEN(1, FILE='EIGVAL.TXT', STATUS='UNKNOWN')
        OPEN(7, FILE='EIGVALIMG.TXT', STATUS='UNKNOWN')
        DO 120 I = 1, N1
         WRITE(1,300) -REAL(EVAL(I))
         WRITE(7,300) -AIMAG(EVAL(I))
120     CONTINUE

        OPEN(2, FILE='EIGVECTOR1.TXT', STATUS='UNKNOWN')
        OPEN(3, FILE='EIGVECTOR2.TXT', STATUS='UNKNOWN')
        OPEN(4, FILE='EIGVECTOR3.TXT', STATUS='UNKNOWN')
        OPEN(5, FILE='EIGVECTOR4.TXT', STATUS='UNKNOWN')
        OPEN(8, FILE='EIGVECTOR5.TXT', STATUS='UNKNOWN')
        OPEN(9, FILE='EIGVECTOR6.TXT', STATUS='UNKNOWN')
        WRITE(2,200) X(1), 0D0
        WRITE(3,200) X(1), 0D0
        WRITE(4,200) X(1), 0D0
        WRITE(5,200) X(1), 0D0
        WRITE(8,200) X(1), 0D0
        WRITE(9,200) X(1), 0D0


        DO 90 I = 1,N-2
         WRITE(2, 200) X(I+1), REAL(EVEC(I,N-2))
         WRITE(3, 200) X(I+1), REAL(EVEC(I,N-3))
         WRITE(4, 200) X(I+1), REAL(EVEC(I,N-4))
         WRITE(5, 200) X(I+1), REAL(EVEC(I,N-5))
         WRITE(8, 200) X(I+1), REAL(EVEC(I,N-6))
         WRITE(9, 200) X(I+1), REAL(EVEC(I,N-7))

90      CONTINUE
        WRITE(2,200) X(N), 0D0
        WRITE(3,200) X(N), 0D0
        WRITE(4,200) X(N), 0D0
        WRITE(5,200) X(N), 0D0
```

```
      WRITE(8,200) X(N), 0D0
      WRITE(9,200) X(N), 0D0

      CLOSE(1)
      CLOSE(2)
      CLOSE(3)
      CLOSE(4)
      CLOSE(5)
      CLOSE(7)
      CLOSE(8)
      CLOSE(9)

200   FORMAT(2D25.16)
300   FORMAT(D25.16)
      END
```

VITA

The author, Jin Zhang, was born in February 1975, in ZheJiang, People's Republic of China. She graduated from No. 14 Hangzhou High School in July 1993.

In September 1993, the author attended Shanghai Jiaotong University (SJTU) with a major in thermal energy engineering and a minor in accounting. She obtained a Bachelor of Science degree in thermal energy engineering in July 1997. She continued her graduate study in SJTU and obtained her Master of Science degree in the Department of Thermal Energy Engineering in March 2000. After that, she worked as a software engineer in Kinpo Electronic Company, Shanghai, for three months.

In August 2000, the author furthered her graduate study in the Department of Mechanical Engineering, Louisiana State University. And in December 2002, she obtained her degree of Master of Science in Mechanical Engineering. In August 2006, the degree of Doctor of Philosophy will be conferred.