



**הטכניון – מכון טכנולוגי לישראל**

**NUMERICAL METHODS IN AEROSPACE  
ENGINEERING**

**HOMEWORK ASSIGNMENT x**

**סמסטר אביב תשפ"ה**

**SPRING SEMESTER 2025**

<b>GRADE</b>	<b>OUT OF</b>	<b>CHAPTER</b>
	2	ABSTRACT
	2	CONTENTS, STYLE &C.
	4	PHYSICAL PROBLEM
	4	MATHEMATICAL MODEL
	26	NUMERICAL METHODS
	20	INFLUENCE OF NUMERICAL METHODS
	20	RESULTS
	2	SUMMARY & CONCLUSIONS
	20	COMPUTER PROGRAM
	<b>100</b>	<b>TOTAL</b>

Almog Dobrescu

ID 214254252

June 2, 2025

## Abstract

In this assignment, a comparison between the finite difference method and the shooting method has been conducted for the evaporation front of fuel spray. A discussion about the optimal parameters was held, and the chosen parameters were used in the results. The results show that the temperature distribution obtained by both methods is the same, however, the runtime of the shooting method is significantly shorter.

## Contents

<b>1</b>	<b>The Physical Problem</b>	<b>1</b>
<b>2</b>	<b>The Mathematical Model</b>	<b>1</b>
<b>3</b>	<b>The Numerical Methods</b>	<b>1</b>
3.1	Finite Difference Method . . . . .	1
3.2	Shooting Method . . . . .	2
<b>4</b>	<b>Influence of The Numerical Methods</b>	<b>3</b>
4.1	Finite Difference Method . . . . .	3
4.1.1	Influence of number of elements $N$ . . . . .	3
4.1.2	Influence of convergence criteria $\varepsilon$ . . . . .	3
4.2	Shooting Method . . . . .	4
4.2.1	Influence of number of elements $N$ . . . . .	4
4.2.2	Influence of convergence criteria $\varepsilon$ . . . . .	4
4.2.3	Influence of initial guess $s_0$ . . . . .	5
<b>5</b>	<b>Results and Discussion</b>	<b>6</b>
<b>6</b>	<b>Summary and Conclusion</b>	<b>7</b>
<b>A</b>	<b>Listing of The Computer Program</b>	<b>8</b>
A.1	Parameters . . . . .	8
A.2	Main Code . . . . .	8

## List of Figures

1	FD - Influence of the number of elements $N$ . . . . .	3
2	FD - Influence of the convergence criteria $\varepsilon$ . . . . .	3
3	Shooting - Influence of the number of elements $N$ . . . . .	4
4	Shooting - Influence of the convergence criteria $\varepsilon$ . . . . .	4
5	Number of iterations as a function of initial guess . . . . .	5
6	Temperature as a function of $\zeta$ . . . . .	6
7	Mass fraction as a function of $\zeta$ . . . . .	6

## Listings

1	Parameters file . . . . .	8
2	The main file . . . . .	8

## Nomenclature

$\alpha$	mass ratio between the liquid fuel and environment oxygen
$\beta$	ratio between the latent heat of the liquid fuel and between the chemical reaction heat of liquid vapor with oxygen
$\Lambda$	empirical constant of droplet vaporation
$\varepsilon$	convergence limit
$h$	size of each cell in the domain
$i$	cell index
$m_d$	mass fraction of liquid fuel in droplets
$N$	number of elements
$s$	temporeary variable
$T$	temperature
$T_u$	environment temperature of the liquid fuel upstream
$T_v$	vaporization temperature of the liquid fuel
$x$	spatial coordinate
$\square^{(n)}$	value at time step n



## 1 The Physical Problem

Liquid rockets are an important part of the future of rocket engine propulsion. To maximize the efficiency of the engine, it is crucial to know the location of the evaporation front. The physical problem at hand is the location of the evaporation front of a fuel spray after the atomization injector.

## 2 The Mathematical Model

The evaporation front of fuel spray can be described by solving the following equations:

$$\begin{aligned} \frac{d^2 T}{d\zeta^2} &= \Lambda e^T \left( T_v - T_u + \alpha\beta - \frac{dT}{d\zeta} \right) \\ m_d &= (\alpha\beta\Lambda e^T)^{-1} \frac{d^2 T}{d\zeta^2} \end{aligned} \quad (1)$$

The boundary condition of the problem:

$$\begin{aligned} \zeta \rightarrow -\infty : \quad & m_d \rightarrow 1 & T &\rightarrow \zeta \cdot (T_v - T_u) \\ \zeta \rightarrow +\infty : \quad & T \rightarrow \zeta \cdot (T_v - T_u + \alpha\beta) & m_d &\rightarrow 0 \end{aligned} \quad (2)$$

- According to the definition of  $T$ :  $T|_{\zeta=0} = 0$

## 3 The Numerical Methods

Eq.1 can be rewrite as:

$$\frac{d^2 T}{d\zeta^2} + \Lambda e^T \frac{dT}{d\zeta} - \Lambda e^T (T_v - T_u + \alpha\beta) = 0$$

In our case:

- $\infty$  is at around 30
- $\Lambda = 0.1$
- $T_v = 0.203$
- $T_u = 0.152$
- $\alpha\beta = 0.0234$

To make sure that  $T|_{\zeta=0} = 0$ , we will solve the ODE in two steps, one from  $\zeta \rightarrow -\infty$  to  $\zeta = 0$  and the second from  $\zeta = 0$  to  $\zeta \rightarrow \infty$ .

### 3.1 Finite Difference Method

Using central difference we can write the difference equations:

$$\begin{aligned} \frac{T_{i+1} - 2T_i + T_{i-1}}{h^2} + \Lambda e^{T_i} \frac{T_{i+1} - T_{i-1}}{2h} - \Lambda e^{T_i} (T_v - T_u + \alpha\beta) &= 0 \quad O(h^2) \\ i = 1, 2, \dots, N \quad h &= \frac{\zeta|_{i=N+1} - \zeta|_{i=0}}{N+1-0} \end{aligned} \quad (3)$$

We will use the 'explicit point Jacobi' method:



1. Set the field with initial condition (linear interpolation).
2. Calculate the temperature at index  $i$  and time step  $n+1$  from the previous time step:

$$T_i^{n+1} = \frac{1}{2} (T_{i+1}^n + T_{i-1}^n) + \frac{h}{4} \Lambda e^{T_i^n} (T_{i+1}^n - T_{i-1}^n) - \frac{h^2}{2} \Lambda e^{T_i^n} (T_v - T_u + \alpha\beta) \quad (4)$$

3. The solution is considered converged when:

$$|T_i^{n+1} - T_i^n| < \varepsilon \quad \forall i \in [1, N] \quad (5)$$

### 3.2 Shooting Method

Let's rewrite Eq.3 as a system of 2 ODE:

$$\begin{cases} \frac{dT}{d\zeta} = s & T|_{\zeta \rightarrow -\infty} \rightarrow \zeta \cdot (T_v - T_u) \\ \frac{ds}{d\zeta} = -\Lambda e^T s - \Lambda e^T (T_v - T_u + \alpha\beta) & T|_{\zeta \rightarrow +\infty} \rightarrow \zeta \cdot (T_v - T_u + \alpha\beta) \end{cases} \quad (6)$$

To solve the system of equations using the shooting method, we will guess  $s_{(i=0)}^{(n)}$  and solve the system of equations using forward and backward differences (semi-implicit Euler). Namely:

$$\begin{cases} s_{i+1}^{(n)} = \left( -\Lambda e^{T_i^{(n)}} s_i^{(n)} + \Lambda e^{T_i^{(n)}} (T_v - T_u + \alpha\beta) \right) \cdot h + s_i^{(n)} & O(h) & s_{i=0}^n = s_0^n \\ T_{i+1}^{(n)} = s_i^{(n+1)} \cdot h + T_i^{(n)} & O(h) & T_{i=0}^n = \zeta \cdot (T_v - T_u + \alpha\beta) \end{cases} \quad i = 0, 1, \dots, N \quad (7)$$

To correct the guess of  $s_{(i=0)}^{(n)}$ , let's define:

$$F_{(s_{(i=0)})} = T_{(i=N+1)}^{(n)} - T|_{\zeta \rightarrow +\infty} \quad (8)$$

- When  $F = 0$ , the guess of  $s_{(i=0)}^{(n)}$  is correct

The next guess of  $s_{(i=0)}^{(n+1)}$  will be calculated numerically by using a method to find the root of an equation. Namely:

$$s_{(i=0)}^{(n+1)} = s_{(i=0)}^{(n)} - F_{(s_{(i=0)})}^{(n)} \cdot \frac{s_{(i=0)}^{(n)} - s_{(i=0)}^{(n-1)}}{F_{(s_{(i=0)})}^{(n)} - F_{(s_{(i=0)})}^{(n-1)}} \quad O(h) \quad (9)$$



## 4 Influence of The Numerical Methods

### 4.1 Finite Difference Method

#### 4.1.1 Influence of number of elements $N$

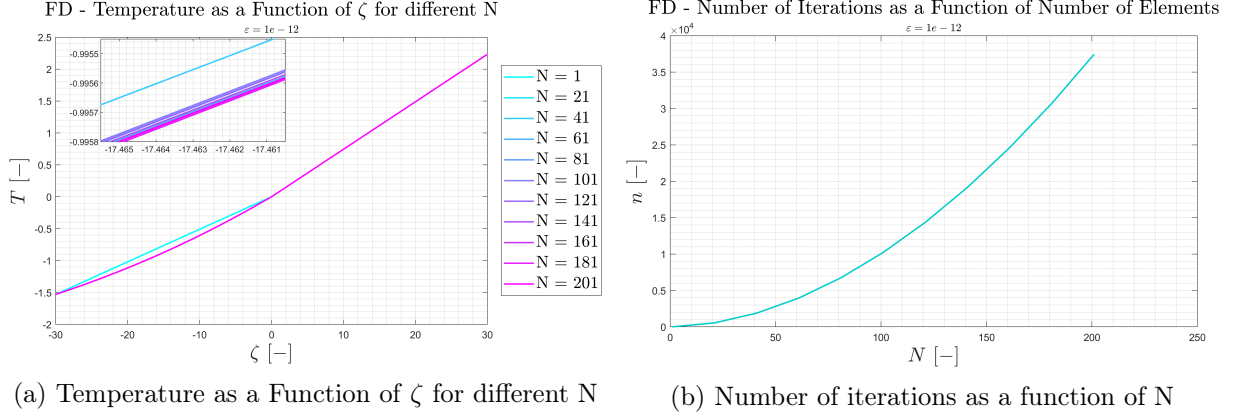


Figure 1: FD - Influence of the number of elements  $N$

In Fig.1a we can see that for  $N$  bigger than 100, the solution does not really change. From Fig.1b we can see that as the number of elements increases, the number of iterations increases as well. We can conclude that  $N = 101$  is a sufficient number of elements.

#### 4.1.2 Influence of convergence criteria $\varepsilon$

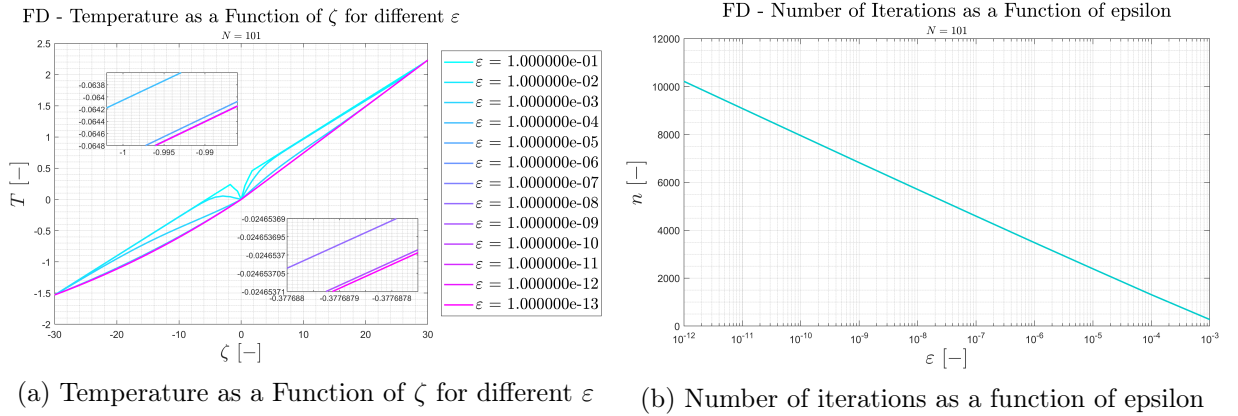


Figure 2: FD - Influence of the convergence criteria  $\varepsilon$

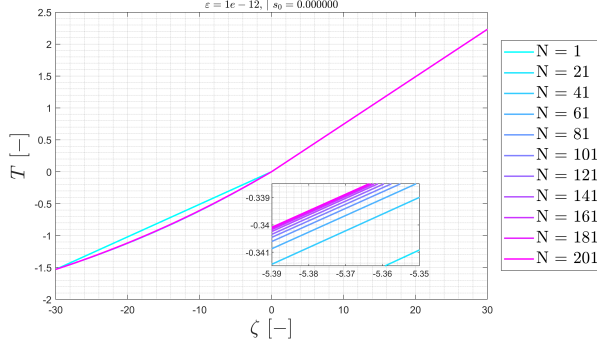
From Fig.2a we can conclude that for a convergence criteria smaller than  $1e^{-8}$ , the solution stays the same. From Fig.2b we can determine that the number of iterations grows exponentially with the decrease of  $\varepsilon$ . With this two insights at hand, we can determine that  $\varepsilon = 1e^{-12}$  is a good choice (although it is not economical with the number of iterations).



## 4.2 Shooting Method

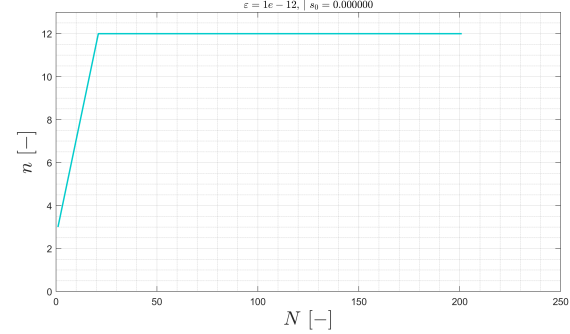
### 4.2.1 Influence of number of elements N

Shooting - Temperature as a Function of  $\zeta$  for different N



(a) Temperature as a Function of  $\zeta$  for different N

Shooting - Number of Iterations as a Function of Number of Elements



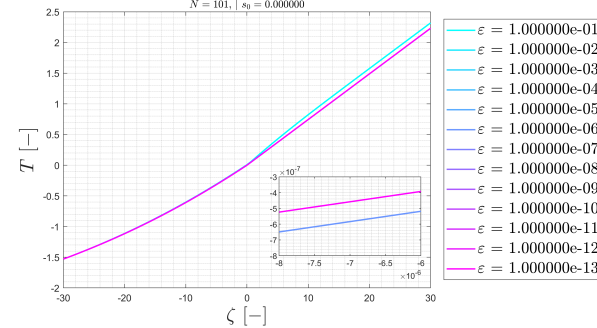
(b) Number of iterations as a function of N

Figure 3: Shooting - Influence of the number of elements N

In Fig.3a we can see that for N bigger than 100, the solution does not really change. From Fig.3b we can see that for more than 25 elements, the number of iterations stays constant for a certain convergence criteria and initial condition. We can conclude that  $N = 101$  is a sufficient number of elements.

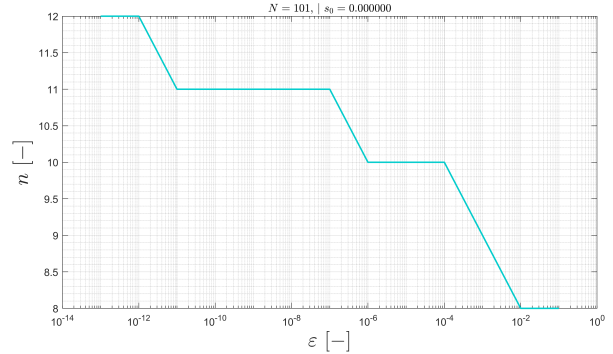
### 4.2.2 Influence of convergence criteria $\varepsilon$

Shooting - Temperature as a Function of  $\zeta$  for different  $\varepsilon$



(a) Temperature as a Function of  $\zeta$  for different  $\varepsilon$

Shooting - Number of Iterations as a Function of Epsilon



(b) Number of iterations as a function of epsilon

Figure 4: Shooting - Influence of the convergence criteria  $\varepsilon$

Figure 4a shows that for a convergence criteria smaller than  $1e^{-6}$ , the differences between the solutions are because of rounding errors. From Fig.4b we can learn that the number of iterations does not change much for Different convergence criteria. We can determine that  $\varepsilon = 1e^{-12}$  is a good choice.



### 4.2.3 Influence of initial guess $s_0$

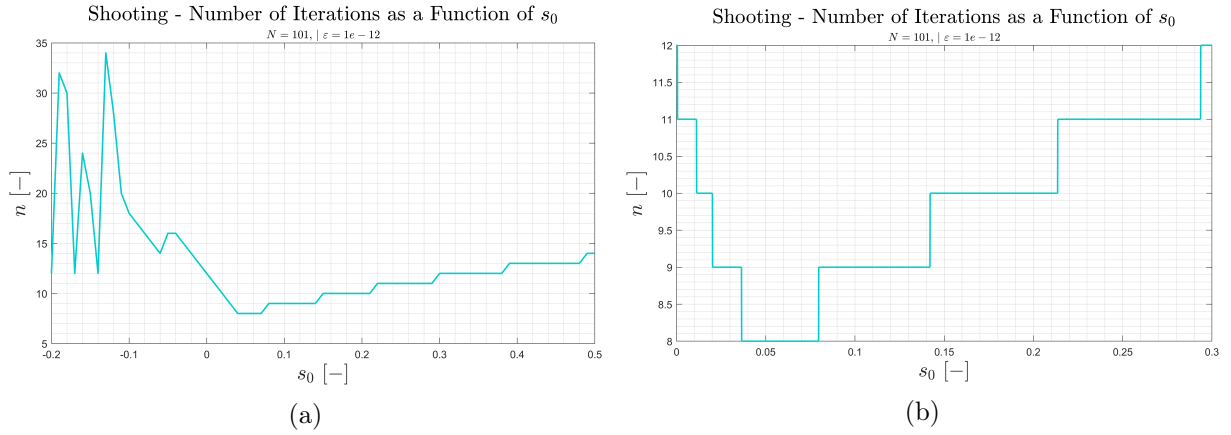


Figure 5: Number of iterations as a function of initial guess

Figure 5 shows that for different negative initial conditions, the number of iterations is not stable. Moreover, we can learn that the real initial slope is around 0.05, as this is the condition for which it took the least amount of steps to converge.



## 5 Results and Discussion

In the following section, the numerical solution for the ODE will be presented using the parameters chosen in the previous section (Sec.4).

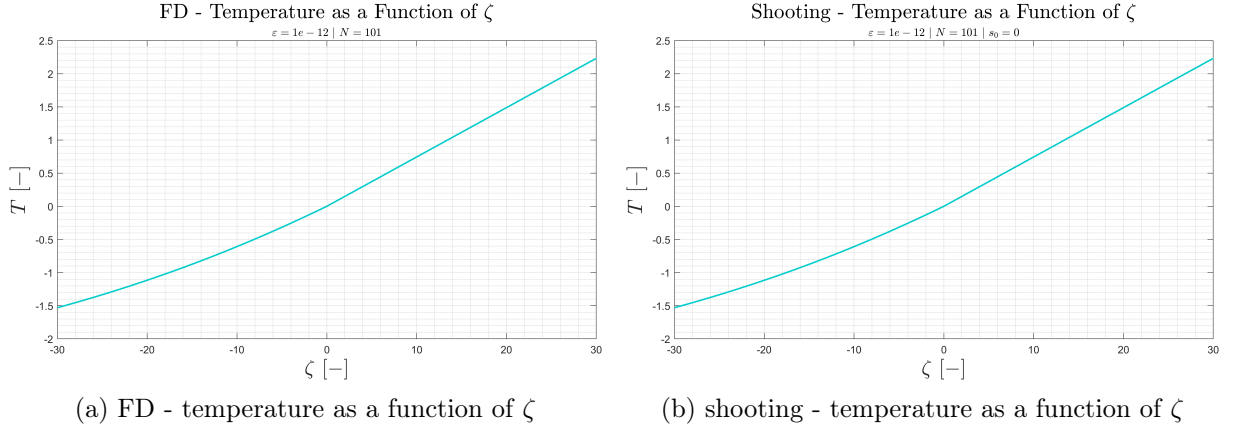
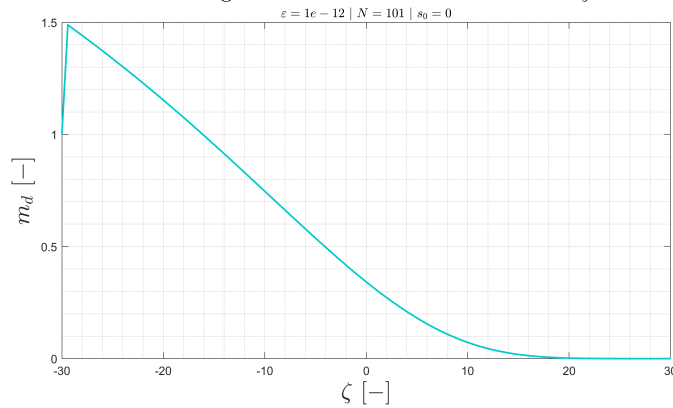
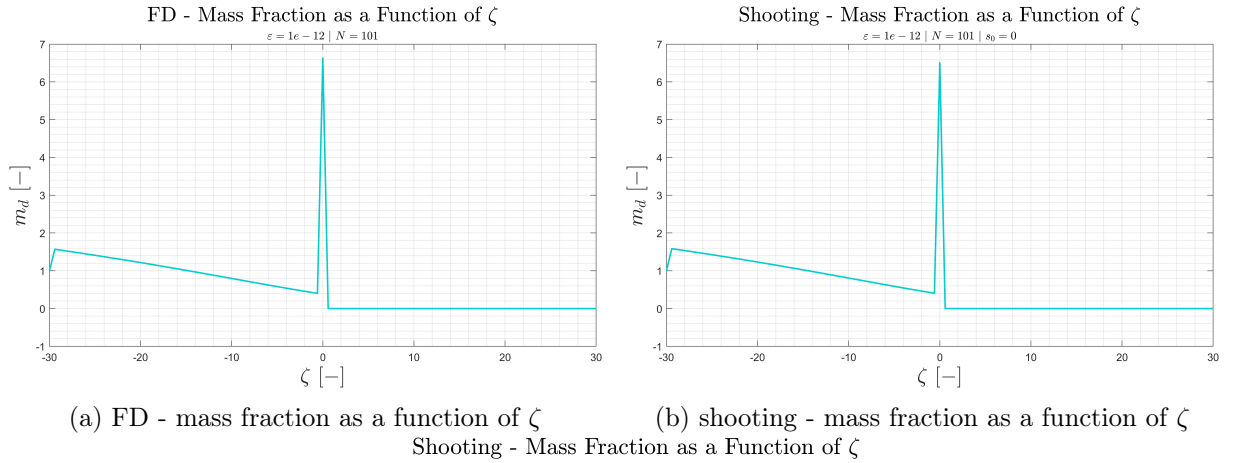


Figure 6: Temperature as a function of  $\zeta$

We can see in Fig.6 that there are no differences in the final result between the two methods. With the finite differences method, it took about 10,000 iterations to converge, while with the shooting method, it took only around 10 steps. On the left side of  $\zeta = 0$ , the solution increases monotonically and faster than linearly. On the right side of  $\zeta = 0$ , the solution increases linearly with  $\zeta$ .



(c) shooting - mass fraction as a function of  $\zeta$  - no zero

Figure 7: Mass fraction as a function of  $\zeta$



In Fig.7 we can see that indeed for  $\zeta > 0$ , the temperature is linear as the mass fraction is constant and depends on the second derivative of the temperature. Moreover, the forced  $T|_{\zeta=0} = 0$  creates an artificial peak in the mass fraction, which causes it to have a single discontinuity. Additionally, the left boundary condition is not met as the analytically calculated boundary conditions for the temperature are only correct when  $\zeta \rightarrow \pm\infty$ .

## 6 Summary and Conclusion

In this assignment, a comparison between the finite difference method and the shooting method has been conducted. A discussion about the optimal parameters was held, and the chosen parameters were used in the result section (Sec.5). The following conclusions came to light:

- The shooting method is more efficient in terms of run time but has a smaller order of local error.
- The number of iterations of the finite difference method increases exponentially with the decrease of the convergence criteria.
- For more than 25 elements, the number of iterations of the shooting methods stays constant.
- The size of the convergence criteria has a minor effect on the number of iterations and the temperature distribution.
- Negative initial guesses of the slope for the shooting method result in drastically changing number of iterations.
- The boundary conditions of the temperature are correct only when  $\zeta \rightarrow \pm\infty$ .
- Forcing  $T|_{\zeta=0} = 0$  results in discontinuity in the mass fraction and the second order derivative of the temperature.

## A Listing of The Computer Program

```

1      N          = 101;
2
3      epsilon = 1e-12;
4
5      lambda   = 0.1;
6      T_v      = 0.203;
7      T_u      = 0.152;
8      alpha_beta = 0.0234;
9      infinity  = 30;
10     zeta_max   = infinity;
11     zeta_min   = -infinity;
12     h          = (zeta_max - zeta_min) / (N+1);
13     T_start    = zeta_min * (T_v - T_u);
14     T_end      = zeta_max * (T_v - T_u + alpha_beta);
15     md_start   = 1;
16     md_end     = 0;

```

## A.2 Main Code



```

30 box on
31 legend(lg, 'FontSize', size-3, 'Location', 'eastoutside', 'Interpreter', 'latex')
32
33 zoom = axes('position', [0.175 0.6 0.275 0.275]);
34 box on % put box around new pair of axes
35 hold all
36 for i = 1:length(Ns)
37     N = Ns(i);
38     colors = cool(length(Ns));
39     h = (zeta_max - zeta_min) / (N+1);
40     plot(linspace(zeta_min, zeta_max, N+2), result{i}{end,:}, '-', 'LineWidth', 1.5, '
        Color', colors(i,:))
41 end
42 zoom.XLim = [-17.4655, -17.4605];
43 zoom.YLim = [-0.9958, -0.99545];
44 grid on
45 grid minor
46 % exportgraphics(fig1, 'images/FD - T vs zeta for diff N.png', 'Resolution', 400);
47
48 %%
49
50 parameters
51
52 fig2 = figure('Position', [150 50 900 500]);
53 hold all
54
55 Ns = 1:20:201;
56 ns = [];
57 for i = 1:length(Ns)
58     N = Ns(i);
59     colors = cool(length(Ns));
60     h = (zeta_max - zeta_min) / (N+1);
61     T_history_FD = finite_difference_method(T_start, T_end, lambda, alpha_beta, T_v, T_u,
        epsilon, h, N);
62     ns(i) = length(T_history_FD(:, 1));
63 end
64 size = 20;
65 plot(Ns, ns, '-', 'LineWidth', 1.5, 'Color', cool(1)*0.8)
66 title('FD - Number of Iterations as a Function of Number of Elements', 'FontSize', size, '
    Interpreter', 'latex');
67 subtitle(sprintf('\varepsilon=%g$', epsilon), 'Interpreter', 'latex')
68 ylabel('$n$ $[-]$', 'FontSize', size, 'Interpreter', 'latex')
69 xlabel('$N$ $[-]$', 'FontSize', size, 'Interpreter', 'latex')
70 grid on
71 grid minor
72 box on
73
74 % exportgraphics(fig2, 'images/FD - n vs N.png', 'Resolution', 400);
75
76 %%
77
78 parameters
79
80 fig3 = figure('Position', [300 50 900 500]);
81
82 epss = logspace(-3, -12, 10);
83 ns = [];
84 for i = 1:length(epss)

```



```

85     epsilon = epss(i);
86     colors = cool(length(epss));
87     T_history_FD = finite_difference_method(T_start, T_end, lambda, alpha_beta, T_v, T_u,
88         epsilon, h, N);
89     ns(i) = length(T_history_FD(:,1));
90 end
91 size = 20;
92 semilogx(epss, ns, '-', 'LineWidth', 1.5, 'Color', cool(1)*0.8)
93 title('FD — Number of Iterations as a Function of epsilon', 'FontSize', size, 'Interpreter',
94     'latex');
95 subtitle(sprintf('$N=%d$', N), 'Interpreter', 'latex')
96 ylabel('$n$ $[-]$', 'FontSize', size, 'Interpreter', 'latex')
97 xlabel('$\varepsilon$ $[-]$', 'FontSize', size, 'Interpreter', 'latex')
98 grid on
99 grid minor
100 box on
101
102 % exportgraphics(fig3, 'images/FD — n vs epsilon.png', 'Resolution', 400);
103
104 %%
105 parameters
106
107 fig4 = figure ('Position', [450 50 900 500]);
108 hold all
109
110 epss = logspace(-1, -13, 13);
111 result = {};
112 lg = {};
113 for i = 1:length(epss)
114     epsilon = epss(i);
115     colors = cool(length(epss));
116     result{end+1} = finite_difference_method(T_start, T_end, lambda, alpha_beta, T_v, T_u,
117         epsilon, h, N);
118     plot(linspace(zeta_min, zeta_max, N+2), result{end}{end,:}, '-', 'LineWidth', 1.5, '
119         Color', colors(i,:))
120     lg{end+1} = sprintf('$\varepsilon$ = %d', epsilon);
121 end
122
123 size = 20;
124 title('FD — Temperature as a Function of $\zeta$ for different $\varepsilon$', 'FontSize',
125     size, 'Interpreter', 'latex');
126 subtitle(sprintf('$N=%d$', N), 'Interpreter', 'latex')
127 ylabel('$T$ $[-]$', 'FontSize', size, 'Interpreter', 'latex')
128 xlabel('$\zeta$ $[-]$', 'FontSize', size, 'Interpreter', 'latex')
129 grid on
130 grid minor
131 box on
132 legend(lg, 'FontSize', size-3, 'Location', 'eastoutside', 'Interpreter', 'latex')
133
134 zoom = axes('position', [0.45 0.2 0.2 0.2]);
135 box on % put box around new pair of axes
136
137 for i = 1:length(epss)
138     epsilon = epss(i);
139     colors = cool(length(epss));
140     plot(linspace(zeta_min, zeta_max, N+2), result{i}{end,:}, '-', 'LineWidth', 1.5, '
141         Color', colors(i,:))
142     hold all
143 end

```



```

137 zoom.XLim = [-0.377688, -0.37768775];
138 zoom.YLim = [-0.02465371, -0.02465369];
139 grid on
140 grid minor
141
142 zoom = axes('position',[0.175 0.6 0.2 0.2]);
143 box on % put box around new pair of axes
144 for i = 1:length(epss)
145     epsilon = epss(i);
146     colors = cool(length(epss));
147     plot(linspace(zeta_min, zeta_max, N+2), result{i}{end,:}, '-', 'LineWidth', 1.5, '
        Color', colors(i,:))
148     hold all
149 end
150 zoom.XLim = [-1.002, -0.986];
151 zoom.YLim = [-0.0648, -0.0636];
152 grid on
153 grid minor
154 % exportgraphics(fig4, 'images/FD — T vs zeta for diff epsilon.png','Resolution',400);
155
156 %%
157 parameters
158
159 fig5 = figure ('Position',[0 200 900 500]);
160 hold all
161
162 Ns = 1:20:201;
163 s0 = 0;
164 result = {};
165 lg = {};
166 for i = 1:length(Ns)
167     N = Ns(i);
168     colors = cool(length(Ns));
169     h = (zeta_max - zeta_min) / (N+1);
170     result{end+1} = shooting_method_with_zero(T_start, T_end, s0, lambda, alpha_beta, T_v
        , T_u, epsilon, h, N);
171     plot(linspace(zeta_min, zeta_max, N+2), result{end}{end,:}, '-', 'LineWidth', 1.5, '
        Color', colors(i,:))
172     lg{end+1} = sprintf('N = %d', N);
173 end
174
175 size = 20;
176 title('Shooting — Temperature as a Function of  $\zeta$  for different N','FontSize',size,
        'Interpreter','latex');
177 subtitle(sprintf('\varepsilon=%g$,  $s_0$ =%f', epsilon, s0), 'Interpreter','latex')
178 ylabel('$T$ $[-]$', 'FontSize',size, 'Interpreter','latex')
179 xlabel('$\zeta$ $[-]$', 'FontSize',size, 'Interpreter','latex')
180 grid on
181 grid minor
182 box on
183 legend(lg,'FontSize',size-3, 'Location','eastoutside', 'Interpreter','latex')
184
185 zoom = axes('position',[0.43 0.2 0.22 0.22]);
186 box on % put box around new pair of axes
187 hold all
188 for i = 1:length(Ns)
189     N = Ns(i);
190     colors = cool(length(Ns));

```



```

191     h = (zeta_max - zeta_min) / (N+1);
192     plot(linspace(zeta_min, zeta_max, N+2), result{i}{end,:}, '-', 'LineWidth', 1.5, '
        Color', colors(i,:))
193 end
194 zoom.XLim = [-5.39, -5.35];
195 zoom.YLim = [-0.3415, -0.3385];
196 grid on
197 grid minor
198 % exportgraphics(fig5, 'images/shooting - T vs zeta for diff N.png','Resolution',400);
199
200 %%
201
202 parameters
203
204 fig6 = figure ('Position',[150 200 900 500]);
205 hold all
206
207 Ns = 1:20:201;
208 s0 = 0;
209 ns = [];
210 for i = 1:length(Ns)
211     N = Ns(i);
212     colors = cool(length(Ns));
213     h = (zeta_max - zeta_min) / (N+1);
214     T_history_FD = shooting_method_with_zero(T_start, T_end, s0, lambda, alpha_beta, T_v,
        T_u, epsilon, h, N);
215     ns(i) = length(T_history_FD(:,1));
216 end
217 size = 20;
218 plot(Ns, ns, '-', 'LineWidth', 1.5, 'Color', cool(1)*0.8)
219 title('Shooting - Number of Iterations as a Function of Number of Elements','FontSize',
        size, 'Interpreter','latex');
220 subtitle(sprintf('\varepsilon=%g$, $|s_0|=%f$', epsilon, s0), 'Interpreter','latex')
221 ylabel('$n$ $[-]$', 'FontSize',size, 'Interpreter','latex')
222 xlabel('$N$ $[-]$', 'FontSize',size, 'Interpreter','latex')
223 grid on
224 grid minor
225 box on
226 ylim([0,13])
227
228 % exportgraphics(fig6, 'images/shooting - n vs N.png','Resolution',400);
229
230 %%
231
232 parameters
233
234 fig7 = figure ('Position',[300 200 900 500]);
235
236 epss = logspace(-1,-13,13);
237 s0 = 0;
238 ns = [];
239 for i = 1:length(epss)
240     epsilon = epss(i);
241     colors = cool(length(epss));
242     T_history_shooting = shooting_method_with_zero(T_start, T_end, s0, lambda, alpha_beta
        , T_v, T_u, epsilon, h, N);
243     ns(i) = length(T_history_shooting(:,1));
244 end

```



```

245 size = 20;
246 semilogx(epss, ns, '-', 'LineWidth', 1.5, 'Color', cool(1)*0.8)
247 title('Shooting — Number of Iterations as a Function of Epsilon','FontSize',size, '
    Interpreter','latex');
248 subtitle(sprintf('$N=%d$, $|s_0|=%f$', N, s0), 'Interpreter','latex')
249 ylabel('$n$ $[-]$', 'FontSize',size, 'Interpreter','latex')
250 xlabel('$\varepsilon$ $[-]$', 'FontSize',size, 'Interpreter','latex')
251 grid on
252 grid minor
253 box on
254
255 % exportgraphics(fig7, 'images/shooting — n vs epsilon.png','Resolution',400);
256
257 %%
258 parameters
259
260 fig8 = figure ('Position',[450 200 900 500]);
261 hold all
262
263 epss = logspace(-1,-13,13);
264 s0 = 0;
265 result = {};
266 lg = {};
267 for i = 1:length(epss)
268     epsilon = epss(i);
269     colors = cool(length(epss));
270     result{end+1} = shooting_method_with_zero(T_start, T_end, s0, lambda, alpha_beta, T_v
        , T_u, epsilon, h, N);
271     plot(linspace(zeta_min, zeta_max, N+2), result{end}{end,:}, '-', 'LineWidth', 1.5, '
        Color', colors(i,:))
272     lg{end+1} = sprintf('$\varepsilon$ = %d', epsilon);
273 end
274
275 size = 20;
276 title('Shooting — Temperature as a Function of $\zeta$ for different $\varepsilon$', '
    FontSize',size, 'Interpreter','latex');
277 subtitle(sprintf('$N=%d$, $|s_0|=%f$', N, s0), 'Interpreter','latex')
278 ylabel('$T$ $[-]$', 'FontSize',size, 'Interpreter','latex')
279 xlabel('$\zeta$ $[-]$', 'FontSize',size, 'Interpreter','latex')
280 grid on
281 grid minor
282 box on
283 legend(lg, 'FontSize',size-3, 'Location','eastoutside', 'Interpreter','latex')
284
285 zoom = axes('position',[0.43 0.2 0.22 0.22]);
286 box on % put box around new pair of axes
287 hold all
288 for i = 1:length(epss)
289     epsilon = epss(i);
290     colors = cool(length(epss));
291     plot(linspace(zeta_min, zeta_max, N+2), result{i}{end,:}, '-', 'LineWidth', 1.5, '
        Color', colors(i,:))
292 end
293 zoom.XLim = [-8e-6, -6e-6];
294 zoom.YLim = [-8e-7, -3e-7];
295 grid on
296 grid minor

```





```

297 % exportgraphics(fig8, 'images/shooting — T vs zeta for diff epsilon.png','Resolution
    ',400);
298
299 %%
300 parameters
301
302 fig9 = figure ('Position',[600 200 900 500]);
303 hold all
304
305 s0s = -0.5:0.1:0.5;
306 result = {};
307 lg = {};
308 for i = 1:length(s0s)
309     s0 = s0s(i);
310     colors = cool(length(s0s));
311     result{end+1} = shooting_method_with_zero(T_start, T_end, s0, lambda, alpha_beta, T_v
        , T_u, epsilon, h, N);
312     plot(linspace(zeta_min, zeta_max, N+2), result{end}{end,:}, '-', 'LineWidth', 1.5, '
        Color', colors(i,:))
313     lg{end+1} = sprintf('$s_0$ = %7.4f', s0);
314 end
315
316 size = 20;
317 title('Shooting — Temperature as a Function of $\zeta$ for different $s_0$', 'FontSize',
    size, 'Interpreter','latex');
318 subtitle(sprintf('$N=%d$, $\epsilon$ \varepsilon=%g$', N, epsilon), 'Interpreter','latex')
319 ylabel('$T$ $[-]$', 'FontSize',size, 'Interpreter','latex')
320 xlabel('$\zeta$ $[-]$', 'FontSize',size, 'Interpreter','latex')
321 grid on
322 grid minor
323 box on
324 legend(lg, 'FontSize',size-3, 'Location','eastoutside', 'Interpreter','latex')
325
326 zoom = axes('position',[0.43 0.2 0.22 0.22]);
327 box on % put box around new pair of axes
328 hold all
329 for i = 1:length(s0s)
330     s0 = s0s(i);
331     colors = cool(length(s0s));
332     plot(linspace(zeta_min, zeta_max, N+2), result{i}{end,:}, '-', 'LineWidth', 1.5, '
        Color', colors(i,:))
333 end
334 zoom.XLim = [-6.905842e-6, -6.905838e-6];
335 zoom.YLim = [-4.5194355e-7, -4.519434e-7];
336 grid on
337 grid minor
338 % exportgraphics(fig9, 'images/shooting — T vs zeta for diff s0.png','Resolution',400);
339 % exportgraphics(fig8, 'images/shooting — T vs zeta for diff epsilon.png','Resolution
    ',400); exportgraphics(fig7, 'images/shooting — n vs epsilon.png','Resolution',400);
    exportgraphics(fig6, 'images/shooting — n vs N.png','Resolution',400); exportgraphics
    (fig5, 'images/shooting — T vs zeta for diff N.png','Resolution',400); exportgraphics
    (fig4, 'images/FD — T vs zeta for diff epsilon.png','Resolution',400); exportgraphics
    (fig3, 'images/FD — n vs epsilon.png','Resolution',400); exportgraphics(fig2, 'images
    /FD — n vs N.png','Resolution',400); exportgraphics(fig1, 'images/FD — T vs zeta for
    diff N.png','Resolution',400);
340
341 %%
342 parameters

```

[illegible]



```

396 parameters
397
398 fig12 = figure ('Position',[0 100 900 500]);
399
400 T_history = finite_difference_method(T_start, T_end, lambda, alpha_beta, T_v, T_u,
    epsilon, h, N);
401 plot(linspace(zeta_min, zeta_max, N+2), T_history{end,:}, '-', 'LineWidth', 1.5, 'Color',
    cool(1)*0.8)
402
403 size = 20;
404 title('FD — Temperature as a Function of  $\zeta$ ','FontSize',size, 'Interpreter','latex')
    ;
405 subtitle(sprintf('\varepsilon=%g$  $\zeta$  $N=%d$', epsilon, N), 'Interpreter','latex')
406 ylabel('$T$ $[-]$', 'FontSize',size, 'Interpreter','latex')
407 xlabel('$\zeta$ $[-]$', 'FontSize',size, 'Interpreter','latex')
408 grid on
409 grid minor
410 box on
411 % exportgraphics(fig12, 'images/FD — T vs zeta.png','Resolution',400);
412
413 %%
414 parameters
415
416 s0 = 0;
417
418 fig13 = figure ('Position',[150 100 900 500]);
419
420 T_history = shooting_method_with_zero(T_start, T_end, s0, lambda, alpha_beta, T_v, T_u,
    epsilon, h, N);
421 plot(linspace(zeta_min, zeta_max, N+2), T_history{end,:}, '-', 'LineWidth', 1.5, 'Color',
    cool(1)*0.8)
422
423 size = 20;
424 title('Shooting — Temperature as a Function of  $\zeta$ ','FontSize',size, 'Interpreter','
    latex');
425 subtitle(sprintf('\varepsilon=%g$  $\zeta$  $N=%d$  $s_0$ =%g$', epsilon, N, s0), '
    Interpreter','latex')
426 ylabel('$T$ $[-]$', 'FontSize',size, 'Interpreter','latex')
427 xlabel('$\zeta$ $[-]$', 'FontSize',size, 'Interpreter','latex')
428 grid on
429 grid minor
430 box on
431 % exportgraphics(fig13, 'images/shooting — T vs zeta.png','Resolution',400);
432
433 %%
434 parameters
435
436 fig14 = figure ('Position',[300 100 900 500]);
437
438 T_history = finite_difference_method(T_start, T_end, lambda, alpha_beta, T_v, T_u,
    epsilon, h, N);
439 md = calc_md(T_history{end,:}, md_start, md_end, alpha_beta, lambda, h, N);
440 plot(linspace(zeta_min, zeta_max, N+2), md, '-', 'LineWidth', 1.5, 'Color', cool(1)*0.8)
441
442 size = 20;
443 title('FD — Mass Fraction as a Function of  $\zeta$ ','FontSize',size, 'Interpreter','latex
    ');
444 subtitle(sprintf('\varepsilon=%g$  $\zeta$  $N=%d$', epsilon, N), 'Interpreter','latex')

```



```

445 ylabel('$m_d$ $[-]$', 'FontSize', size, 'Interpreter', 'latex')
446 xlabel('$\zeta$ $[-]$', 'FontSize', size, 'Interpreter', 'latex')
447 grid on
448 grid minor
449 box on
450 % exportgraphics(fig14, 'images/FD — md vs zeta.png', 'Resolution', 400);
451
452 %%
453 parameters
454
455 s0 = 0;
456
457 fig15 = figure ('Position', [450 100 900 500]);
458
459 T_history = shooting_method_with_zero(T_start, T_end, s0, lambda, alpha_beta, T_v, T_u,
    epsilon, h, N);
460 md = calc_md(T_history{end,:}, md_start, md_end, alpha_beta, lambda, h, N);
461 plot(linspace(zeta_min, zeta_max, N+2), md, '-', 'LineWidth', 1.5, 'Color', cool(1)*0.8)
462
463 size = 20;
464 title('Shooting — Mass Fraction as a Function of $\zeta$', 'FontSize', size, 'Interpreter',
    'latex');
465 subtitle(sprintf('\varepsilon=%g$ $|$ $N=%d$ $|$ $s_0=%g$', epsilon, N, s0), '
    Interpreter', 'latex')
466 ylabel('$m_d$ $[-]$', 'FontSize', size, 'Interpreter', 'latex')
467 xlabel('$\zeta$ $[-]$', 'FontSize', size, 'Interpreter', 'latex')
468 grid on
469 grid minor
470 box on
471 % exportgraphics(fig15, 'images/shooting — md vs zeta.png', 'Resolution', 400);
472
473 %%
474 parameters
475
476 s0 = 0;
477
478 fig16 = figure ('Position', [600 100 900 500]);
479
480 T_history = shooting_method(T_start, T_end, s0, lambda, alpha_beta, T_v, T_u, epsilon, h,
    N);
481 md = calc_md(T_history{end,:}, md_start, md_end, alpha_beta, lambda, h, N);
482 plot(linspace(zeta_min, zeta_max, N+2), md, '-', 'LineWidth', 1.5, 'Color', cool(1)*0.8)
483
484 size = 20;
485 title('Shooting — Mass Fraction as a Function of $\zeta$', 'FontSize', size, 'Interpreter',
    'latex');
486 subtitle(sprintf('\varepsilon=%g$ $|$ $N=%d$ $|$ $s_0=%g$', epsilon, N, s0), '
    Interpreter', 'latex')
487 ylabel('$m_d$ $[-]$', 'FontSize', size, 'Interpreter', 'latex')
488 xlabel('$\zeta$ $[-]$', 'FontSize', size, 'Interpreter', 'latex')
489 grid on
490 grid minor
491 box on
492 % exportgraphics(fig16, 'images/shooting — md vs zeta — no zero.png', 'Resolution', 400);
493 % exportgraphics(fig16, 'images/shooting — md vs zeta — no zero.png', 'Resolution', 400);
    exportgraphics(fig15, 'images/shooting — md vs zeta.png', 'Resolution', 400);
    exportgraphics(fig14, 'images/FD — md vs zeta.png', 'Resolution', 400); exportgraphics(
    fig13, 'images/shooting — T vs zeta.png', 'Resolution', 400); exportgraphics(fig12, '

```



```

images/FD - T vs zeta.png','Resolution',400); exportgraphics(fig11, 'images/shooting
- n vs s0 - with negative.png','Resolution',400); exportgraphics(fig10, 'images/
shooting - n vs s0 - no negative.png','Resolution',400); exportgraphics(fig8, 'images/
shooting - T vs zeta for diff epsilon.png','Resolution',400); exportgraphics(fig7, '
images/shooting - n vs epsilon.png','Resolution',400); exportgraphics(fig6, 'images/
shooting - n vs N.png','Resolution',400); exportgraphics(fig5, 'images/shooting - T
vs zeta for diff N.png','Resolution',400); exportgraphics(fig4, 'images/FD - T vs
zeta for diff epsilon.png','Resolution',400); exportgraphics(fig3, 'images/FD - n vs
epsilon.png','Resolution',400); exportgraphics(fig2, 'images/FD - n vs N.png','
Resolution',400); exportgraphics(fig1, 'images/FD - T vs zeta for diff N.png','
Resolution',400);

494
495
496
497
498
499
500
501
502 %%
503 % Functions
504 % @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
505
506 function [T_history] = finite_difference_method(T_start, T_end, lambda, alpha_beta, T_v,
    T_u, epsilon, h, N)
507     % check that number of elements is odd
508     if mod(N,2) ~= 1
509         fprintf('N must be odd. It is: %d\n', N);
510         T_history = 0;
511         return
512     end
513     % calc index of zeta = 0
514     index_of_zero = (N + 1) / 2 + 1; % the plus one after the fraction is because of
        matlab
515     % set initial guess of the temperature
516     T_current = T_start + (T_end - T_start) / (N+1) * [0:N+1];
517     % add the current temperature into the temperature history
518     T_history{1,:} = T_current;
519     % the main loop of the solver
520     for i = 1:1e6
521         i
522         % getting the updated temperature vector
523         T_next = finite_difference_method_step(T_current, lambda, alpha_beta, T_v, T_u,
            index_of_zero, h, N);
524         % add the current temperature into the temperature history
525         T_history{end+1,:} = T_next;
526         % check convergence
527         if check_convergence_finite_difference_method(T_next, T_current, epsilon, N)
528             break
529         end
530         % updating the temperature field
531         T_current = T_next;
532     end
533 end
534
535 function converged = check_convergence_finite_difference_method(T_next, T_current,
    epsilon, N)
536     converged = true;

```



```

537     for i = [1:N]+1
538         if abs(T_next(i) - T_current(i)) > epsilon
539             converged = false;
540             return
541         end
542     end
543 end
544
545 function [T_next] = finite_difference_method_step(T_current, lambda, alpha_beta, T_v, T_u
    , index_of_zero, h, N)
546     % setting boundary conditions
547     T_next(1) = T_current(1);
548     T_next(N+1+1) = T_current(N+1+1);
549
550     % setting the temperature at zeta = 0 to be zero
551     T_next(index_of_zero) = 0;
552
553     % performing the step
554     for i = [1:N]+1
555         % keeping the temperature at zeta = 0 to be zero
556         if i == index_of_zero
557             continue;
558         end
559         T_next(i) = 0.5*(T_current(i+1) + T_current(i-1)) + 0.25 * h * lambda * exp(
            T_current(i)) * (T_current(i+1) - T_current(i-1)) - 0.5 * h^2 * lambda * exp(
            T_current(i)) * (T_v - T_u + alpha_beta);
560     end
561 end
562
563 % //////////////////////////////////////
564
565 function [T] = shooting_method_step(T0, s0, lambda, alpha_beta, T_v, T_u, h, N)
566     T(1) = T0;
567     s(1) = s0;
568     for i = [0:N]+1 % semi implicit Euler
569         s(i+1) = lambda * exp(T(i)) * ((T_v - T_u + alpha_beta) - s(i)) * h + s(i);
570         T(i+1) = s(i+1) * h + T(i);
571     end
572 end
573
574 function [s0_next] = guess_next_s(s0_current, s0_past, F_current, F_past)
575     s0_next = s0_current - F_current * (s0_current - s0_past) / (F_current - F_past);
576 end
577
578 function [T_history] = shooting_method(T_start, T_end, s0, lambda, alpha_beta, T_v, T_u,
    epsilon, h, N)
579     F = [];
580     s0_s = [];
581     % setting initial guess
582     s0_s(1) = s0;
583     % the main loop of the solver
584     for i = 1:1e6
585         i
586         % getting temperature vector according to the latest initial guess
587         T = shooting_method_step(T_start, s0_s(end), lambda, alpha_beta, T_v, T_u, h, N);
588         % add the current temperature into the temperature history
589         if i == 1
590             T_history{1,:} = T;

```



```

591     else
592         T_history{end+1,:} = T;
593     end
594     % calculating the temperature difference at the end
595     F(end+1) = T(end) - T_end;
596     % checking if the size of the difference is small enough
597     if abs(F(end)) < epsilon
598         break
599     end
600     % updating the initial guess of s
601     if i == 1
602         s0_s(end+1) = guess_next_s(s0_s(end), s0_s(end)-1, F(end), F(end)-1); %
            gussing initial slop of 1
603     else
604         s0_s(end+1) = guess_next_s(s0_s(end), s0_s(end)-1, F(end), F(end)-1));
605     end
606 end
607 end
608
609 function [T_history] = shooting_method_with_zero(T_start, T_end, s0, lambda, alpha_beta,
    T_v, T_u, epsilon, h, N)
610     % check that number of elements is odd
611     if mod(N,2) ~= 1
612         fprintf('N must be odd. It is: %d\n', N);
613         T_history = 0;
614         return
615     end
616     % calc index of zeta = 0
617     index_of_zero = (N + 1) / 2 + 1; % the plus one after the fraction is because of
        matlab
618     % shooting from the initial temperature to zeta = 0 and setting T = 0
619     % there
620     [T_history_to_zero] = shooting_method(T_start, 0, s0, lambda, alpha_beta, T_v, T_u,
        epsilon, h, (N+1)/2-1);
621     % shooting from zeta = 0 wher T = 0 to the final temperature
622     [T_history_from_zero] = shooting_method(0, T_end, s0, lambda, alpha_beta, T_v, T_u,
        epsilon, h, (N+1)/2-1);
623     % setting the number of steps for each side
624     len_to_zero = length(T_history_to_zero);
625     len_from_zero = length(T_history_from_zero);
626     % zipping the history of the temperature vectors according to the number of steps it
627     % took
628     if len_to_zero > len_from_zero
629         for i = 1:len_from_zero
630             T_history{i,:} = zip_two_arrays(T_history_to_zero{i,:}, T_history_from_zero{i
                ,:}, 1);
631         end
632         for i = len_from_zero+1:len_to_zero
633             T_history{i,:} = zip_two_arrays(T_history_to_zero{i,:}, T_history_from_zero{
                end,:}, 1);
634         end
635     elseif len_from_zero > len_to_zero
636         for i = 1:len_to_zero
637             T_history{i,:} = zip_two_arrays(T_history_to_zero{i,:}, T_history_from_zero{i
                ,:}, 1);
638         end
639         for i = len_to_zero+1:len_from_zero

```



```

640         T_history{i,:} = zip_two_arrays(T_history_to_zero{end,:}, T_history_from_zero
        {i,:}, 1);
641     end
642 else
643     for i = 1:len_to_zero
644         T_history{i,:} = zip_two_arrays(T_history_to_zero{i,:}, T_history_from_zero{i
        ,:}, 1);
645     end
646 end
647 end
648
649 function [array] = zip_two_arrays(a1, a2, num_of_overlap)
650     array = [a1,a2(num_of_overlap+1:end)];
651 end
652
653 function md = calc_md(T, md_start, md_end, alpha_beta, lambda, h, N)
654     md(1) = md_start;
655     md(N+1+1) = md_end;
656     for i = [1:N]+1
657         md(i) = (alpha_beta * lambda * exp(T(i)))^(-1) * (T(i+1) - 2 * T(i) + T(i-1)) / h
        ^2;
658     end
659 end

```

Listing 2: The main file