Washington University in St. Louis

JAMES MCKELVEY SCHOOL OF ENGINEERING

Computation of Heat Transfer for Optimal Cooking of a Hot Dog

MEMS 3420: Heat Transfer, 2021 Spring Semester

David Allen
Benjamin Horine
Jacob Ridderhoff
Elizabeth Saliba

We hereby certify that this report herein is our original academic work, completed in accordance with the McKelvey School of Engineering and Undergraduate Student academic integrity policies, and submitted to fulfill the requirements of this assignment:

Introduction

Heat transfer is prevalent in everyday life, and computational analysis allows for a better understanding of conduction, convection, and radiation on an object. This report documents an analysis of heat transfer when determining the optimal cooking of a hot dog, specifically focusing on the convective and radiative heat transfer that the hot dog experiences. A figure depicting the basic problem setup of this report, is shown in Figure 1 below. The goal of this report was to determine the ideal conditions to barbeque a perfect hot dog in the minimum amount of time. To approach this goal, two solutions were found: an analytical solution using calculated convective and radiative heat transfer coefficients, and a numerical finite differencing solution analyzing the temperature distribution at nodal points along the hot dog's cross section.

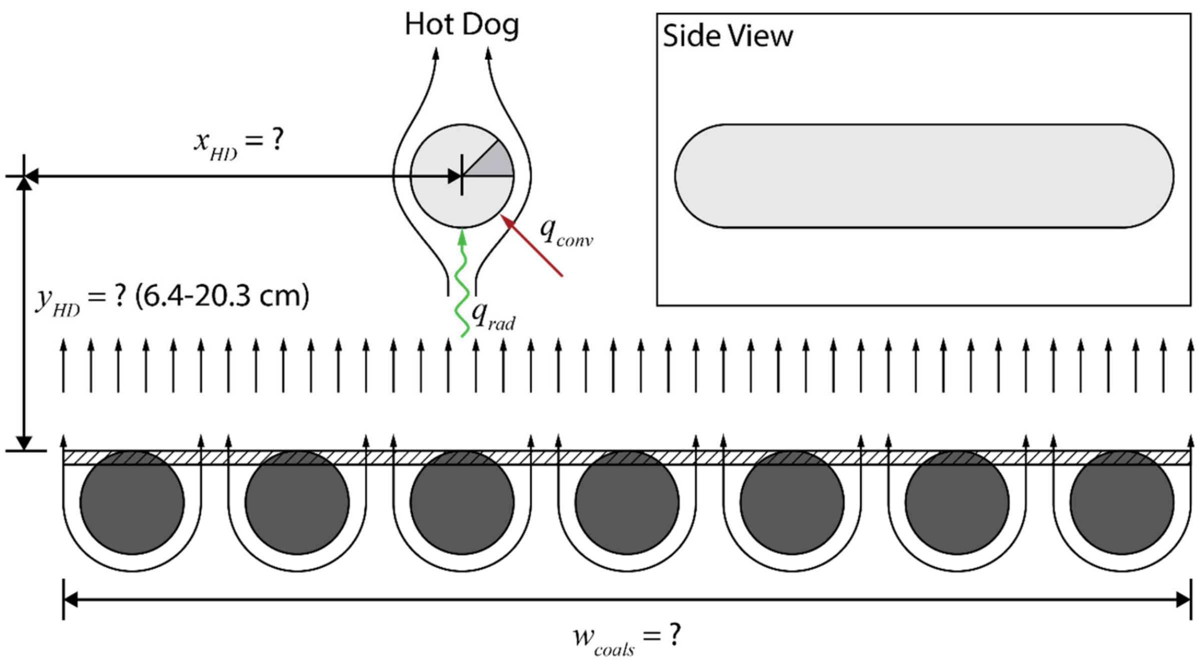

Figure 1 Schematic representation of problem.

Assumptions and Boundary Conditions. The following assumptions were made to simplify the analysis on the hot dog.

(1) Conductive heat transfer between the grill grates and hot dog was ignored. For the purposes of this analysis, it was assumed that the hot dog was floating over the coals.
(2) It was assumed the width of the coal bed was 23 inches, which was based on the rectangular Grainger adjustable steel pedestal grill [1]. The hot dog was placed directly at the center of this width.
(3) The surrounding air temperature was assumed to be 23°C.
(4) The emissivity of the coals was assumed to be 0.8. The coals were treated as a rectangular plate, and it was assumed they had a constant temperature of 450°C.
(5) The fluid flow from the coals to the hot dog was assumed to have ideal gas properties and was treated as air.

(6) The hot dog was modelled as an infinitely long cylinder since the ratio between the length of a typical hot dog and its radius is greater than 10. This assumption allowed for the use of equations 5.52a, 5.52b, and 5.52c when modeling the temperature distribution analytically [2].

(7) To assume uniform heating of the surface by convection and radiation, it was assumed that the hot dog was rotated at the halfway point of the cooking process.

(8) The hot dog properties were assumed to remain constant over time, and they are documented in Table 1.

Table 1 Properties of a hot dog.

Density, ρ	880 $\frac{kg}{m^3}$
Thermal conductivity, k	0.52 $\frac{W}{m \cdot K}$
Specific heat capacity, c	3350 $\frac{J}{kg \cdot K}$
Diameter, D	0.0254 m
Emissivity, ϵ	0.45

The following boundary conditions were defined to perform the analysis on the hot dog.

(1) The initial temperature of the hot dog was 10°C.

(2) The surface temperature of the hot dog was not allowed to exceed 100°C at any time.

(3) The final temperature of the centerline of the hot dog was 68°C.

Calculations of Heat Transfer Coefficients

Convective Heat Transfer Coefficient. To calculate the convective heat transfer coefficient, concepts and equations from Chapters 7 and 9 were applied. To begin, the free convection equations were used to determine the velocity of the fluid flow from the coals.

$$u_{01}^2 = g\beta\left(T_s - T_{\text{inf}}\right)L \tag{1}$$

Where g is the acceleration due to gravity, β is the inverse of the film temperature, T_c is the surface temperature of the coals, T_{inf} is the atmospheric air temperature, and L is the diameter of a coal. The film temperature was calculated using the following equation:

$$T_f = \frac{T_c + T_{\text{inf}}}{2} \tag{2}$$

The values used in these calculations are tabulated in Table 2.

Table 2 Values for calculating free convection velocity from coals.

Acceleration due to gravity, g	9.81 m/s^2
Coal Temperature, T_c	450°C (723 K)
Air Temperature, T_{inf}	23°C (296 K)
Film Temperature, T_f	236.5°C (509.5 K)
β	0.00196 K$^-$1
Diameter of Coal, L	0.0508 m
Free Convection Velocity From Coals, u_{01}	0.646 m/s

Using the velocity from the coals, a similarity relationship was applied using the equation below. Since it was assumed that the coals took up half the area of the grill, the velocity at the hot dog was assumed to be twice that of the coals. The following equations show this relationship and were used to calculate the forced convection velocity at the hot dog.

$$u_{01} A_c = u_{02} A_{\mathrm{HD}} \tag{3}$$

$$u_{01} = 2u_{02} \tag{4}$$

Where A_c is the area that fluid flows through from the coals, u_{02} was the forced convection velocity at the hot dog, and A_{HD} is the area that fluid flows through from the hot dog.

Next, Reynold's number of flow at the hot dog was calculated using the following equation.

$$Re = \frac{u_{02} D}{v} \tag{5}$$

Where v is the kinematic viscosity of air at the film temperatures. The values used to calculate Reynold's number are tabulated in Table 3.

Table 3 Values for calculating Reynold's number of flow at hot dog.

Forced Convection Velocity at Hot Dog, u02	0.323 m/s
Diameter of Hot Dog, D	0.0254 m
Kinematic Viscosity of Air,	$40.07 \cdot 10^{-6} m^2/s$
Reynold's Number, Re	204.75

The Reynold's number was used to calculate the Nusselt number using the equation below.

$$Nu_D = C Re_D^m Pr^{\frac{1}{3}} \tag{6}$$

The values for C and m were obtained from Table 7.2 for a circular cylinder in cross flow with a Reynold's number between 40 and 400, and the Prandtl number was obtained from air property tables at the film temperature [3]. The values used to calculate the Nusselt number are tabulated in Table 4.

Table 4 Values for calculating Nusselt number.

C (From Table 7.2)	0.683
M (From Table 7.2)	0.466
Prandtl Number, Pr	0.695
Nusselt Number, Nu_D	7.225

Finally, the following equation was used to calculate the convective heat transfer coefficient.

$$h_{\mathrm{conv}} = \frac{Nu_D k}{D} \tag{7}$$

Where k was the thermal conductivity of the air at 250°C, which was the assumed film temperature at the hot dog. The values used for this calculation are tabulated in Table 5.

Table 5 Values for calculating convective heat transfer coefficient.

Thermal Conductivity of Air, k	0.04104 $\frac{W}{m \cdot K}$
Convective Heat Transfer Coefficient, hconv	11.67 $\frac{W}{m^2 \cdot K}$

Radiative Heat Transfer Coefficient. To calculate the Heat Transfer Coefficient, h_{rad}, concepts from Chapter 12 and 13 were used. To begin an approximation for the view factor using two dimensional geometries needed to be made, this was done using Table 13.1 from the textbook. The hot dog with respect to the coals was simplified to a cylinder and parallel rectangle, and a depiction of this simplification is shown below in Fig. 2. The following equation can be used to find the view factor:

$$F_{ij} = \frac{r}{s_1 - s_2}\left[\tan^{-1}\frac{s_1}{L} - \tan^{-1}\frac{s_2}{L}\right] \quad (8)$$

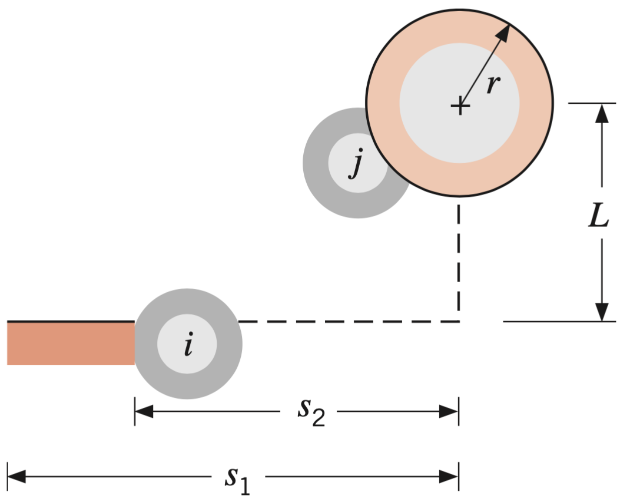

Figure 2 View Factor simplification of hot dog and coals.

The coals, represented by the parallel rectangle, and the hot dog, represented by the cylinder. Allow a calculation for the view factor from the coals with respect to the hot dog. In the equation above, values for r, s_1, s_2 and L are shown in the Table below, where L is an optimization parameter, with a range of 6.4cm to 20.3cm. This variable was optimized in MATLAB, with every iteration of L being used starting at 6.4cm and ending at 20.3cm, each with a difference of 0.1cm. s_1 and s_2 were calculated, by assuming an 18in grill with the hot dog being placed in the middle.

Table 6 Values for calculating view factor.

r	0.0127m
s_1	0.2921m
s_2	-0.2921m
L	Optimization Parameter

Once the view factor was calculated for the coals with respect to the hot dog, the view factor for the hot with respect to the coals needed to be calculated. This was done using reciprocity shown in eq. 9.

$$A_{\text{dog}} \cdot F_{\text{dog,coals}} = A_{\text{coals}} \cdot A_{\text{coals,dog}} \tag{9}$$

Since the area for the hot dog and the coals are both functions of length, which is assumed to be infinite, they are equal on both sides and cancel if multiplied through on both sides. The values for the area of the hot and the coals are shown in the table below.

Table 7 Area of hot dogs and coals.

Area of hot dog	0.0399 m^2
Area of coals	0.5842 m^2

Once the view factor for the hot dog with respect to the coals was found, the absorbed radiation for the hot dog with respect to the coals was calculated using the following equation.

$$Q_{\text{abs}} = \epsilon_{\text{coals}} \cdot \epsilon_{\text{dog}} \cdot A_{\text{dog}} \cdot F_{\text{dog,coals}} \cdot \sigma \cdot T^4_{\text{o, coals}} \tag{10}$$

This equation was written by taking into account the emissivity of the hot dog and coals, and the area of the hot dog, the view factor of the dog with respect to the coals, Boltzman's constant, σ, and the initial temperature of the hot dog at its surface. The table below lists and defines these variables.

Table 8 Values for calculating absorbed radiation heat transfer of the hot dog.

ϵ_{coals}	0.80
ϵ_{dog}	0.45
$T_{o,\text{coals}}$	723K

Next, the emitted radiation from the hot dog needed to be calculated, which used the following equation. This equation differs from the absorbed radiation because it takes the temperature at the surface of the hot dog, with respect to the view factor and emissivity of the hot dog.

$$Q_{\text{emit}} = \epsilon_{\text{coals}} \cdot \epsilon_{\text{dog}} \cdot A_{\text{dog}} \cdot F_{\text{dog,coals}} \cdot \sigma \cdot T^4_{2,\text{s}} \tag{11}$$

The equation for $T_{2,s}$ is written below in eq. 12. The temperature at the surface of the hot dog initially is 283K, which is a given variable.

$$T_{2,s} = \frac{T_{o,dog}}{\left(\epsilon_{\text{coals}} \cdot F_{\text{dogs,coals}}\right)^{0.25}} \tag{12}$$

Once the emitted heat transfer was calculated, the total radiation heat transfer was found using the following equation:

$$Q_{\text{total}} = Q_{\text{abs}} - Q_{\text{emit}} \tag{13}$$

This gave a value for the total radiation heat transfer, and from there the radiation heat transfer could be calculated using the following equation.

$$h_{\text{rad}} = \frac{Q_{\text{total}}}{\left(A_{\text{dog}}\left(T_{\text{o,coals}} - T_{2,s}\right)\right)} \tag{14}$$

In Table 6, the value for L was an optimization parameter, and when coding the previous 6 equations into MATLAB, at each value for L, starting at 6.4cm, and increasing by 0.1cm each time. This was done using a while loop, and after each iteration, L increased by 0.1cm. Once the code completed the loop, there were 139 different values for the radiation heat transfer, h_{rad}, each according to the specified height of the hot dog above the coals, L. Since the value for the heat transfer coefficient was hinted at in the problem statement to be around 6 $\frac{W}{m^2 \cdot K}$, all I had to do was find the height (L) that gave a value for h closest to this value. The final result was a height of 9.2cm above the coals, which correlated to a value of 6.01 $\frac{W}{m^2 \cdot K}$ for the radiation heat transfer coefficient. The code used to find this value, and all the equations described above is given in Appendix A.

Total Heat Transfer Coefficient. The total heat transfer coefficient was calculated with the following equation.

$$h_{\text{tot}} = h_{\text{conv}} + h_{\text{rad}} \tag{15}$$

$$= 11.67 + 6.01 \tag{16}$$

$$= 17.68 \frac{W}{m^2 \cdot k} \tag{17}$$

Analytical Solution

The infinite cylinder can be approximated by a single term approximation if n=1 for $F_o > 0.2$. The assumption that the hot dog is an infinitely long cylinder can be verified if $\frac{L}{R_o} \geq 10$ is valid. The equation to find the centerline temperature distribution over time is included below.

$$\theta_o^2 = C_1 exp\left(-\zeta_1 F_o\right) \tag{18}$$

The equation to find the temperature distribution at points radially from the center of the cylinder can be found with the following equation:

$$\theta^* = C_1 exp\left(-\zeta_1 F_o\right) J_o exp\left(\zeta_1 r^*\right) \tag{19}$$

Where $F_o = \frac{\alpha t}{k}$, and $\alpha = \frac{k}{\rho C_p}$, the coefficients, ζ_1 and C_1 are determined from the Biot number found in Table 5.1 in the Bergman [2]. θ^* is defined as:

$$\theta^* = \frac{T - T_\infty}{T_i - T_\infty} \tag{20}$$

The sum of the Biot numbers for both convection and radiation can be used to find the coefficients used in eq. 18 and eq. 19. The Biot number is defined as:

$$Bi = \frac{h_{\text{tot}} r_o}{k} \tag{21}$$

Where h_{tot} is the heat transfer coefficient, r_o is the outer radius of the cylinder, and k is the thermal conductivity.

The Biot number for the total heat transfer coefficient was found to be 0.432. Using interpolation for the values found on Table 5.1, ζ_1 was 0.88014 rad and C_1 was 1.09995. The Fourier number remains as a function of time with a value of 0.00108t.

Since the Fourier number is a function of time, it can be plugged into eq. 19. All other values are known, therefore time can be found as t_f=444.52s. To check to make sure this time is valid it can be plugged back into $F_o = \frac{\alpha t}{k}$. This returns a Fourier number of 0.48, which is greater than 0.2. Therefore the one-term approximation is valid.

MATLAB code was written to visualize the hot dog's centerline and surface temperature over time with the equations above. This code is included in Appendix A. Fig 3 includes the plot of the hot dog's centerline temperature over time.

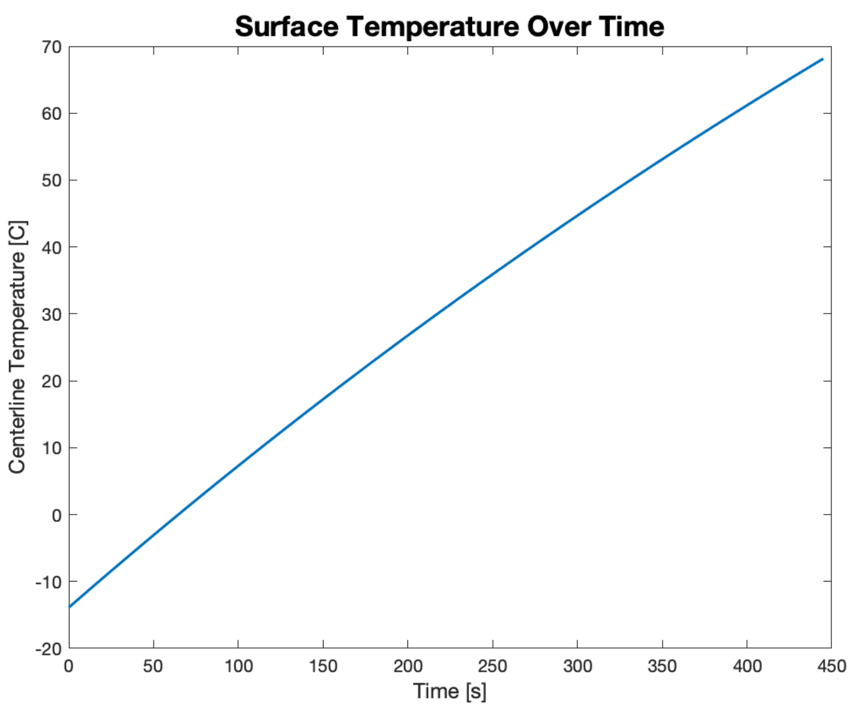

Figure 3 Ceneterline temperature over time.

Figure 4 includes a plot of the hot dog's surface temperature over time.

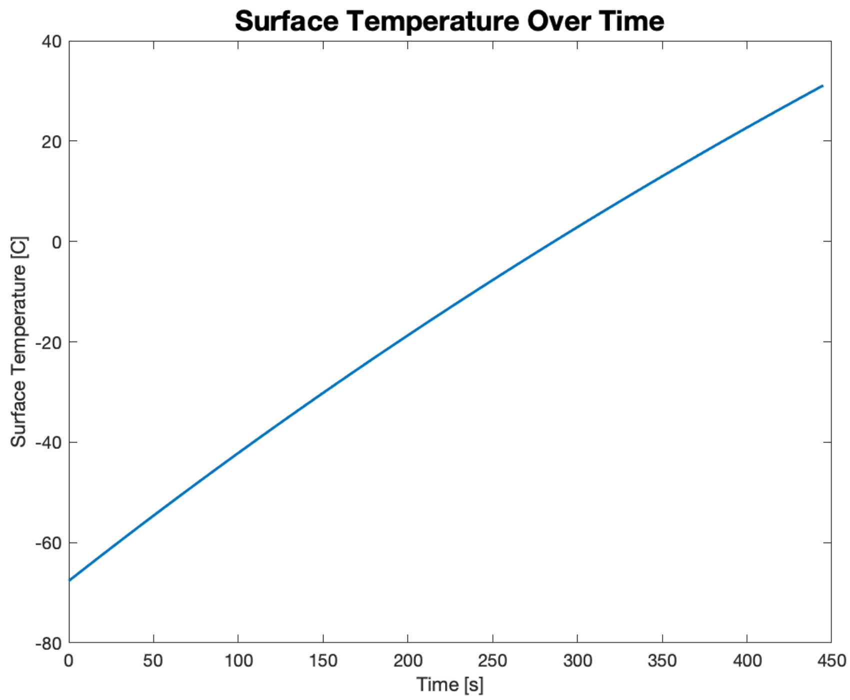

Figure 4 Surface temperature over time.

MATLAB code was also written to visualize the temperature distribution at different points along the radius of the hot dog at various time points. Figure 5 includes this plot for temperature distributions at 30 seconds, 2 minutes, and 7.41 minutes.

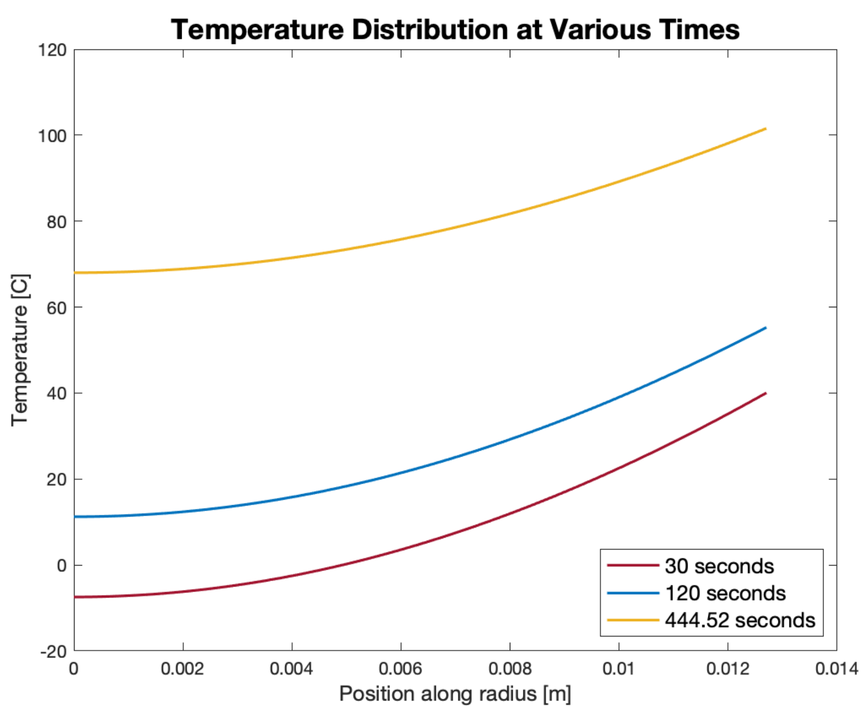

Figure 5 Temperature distributions at various points in time.

At low times, the plots in Figures 3, 4, and 5 display inaccurate values. This is because the one-term approximation is only valid for Fourier numbers greater than 0.2, and small increments of time yield a Fourier number less than 0.2. To be precise, the one-term approximation does not become valid until 185.19 seconds. However, since we are more interested in the total cooking time and less interested in the early points in time in the cooking process, this simplification is plausible.

Figures 3 and 4 follow an expected trend of increasing centerline and surface temperature over time. Logically, this makes sense since heat transfer is occurring from hot coals to an initially 10°C hot dog. Figure 5 follows the expected trend that increased cooking time corresponds to greater temperatures across the cross section of the hot dog as well as surface temperatures greater than the centerline temperature. At the final cooking time, the centerline temperature was at 68°C and the surface temperature was at 100°C. While the temperature distributions at 30 seconds and two minutes follow the expected trend, the numerical values corresponding with these temperature distributions are inaccurate due to the invalidity of the one-term approximation.

Numerical Solution

The type of numerical solution used in this project is called a finite difference method. This method in general uses small differentials to approximate a derivative. For this problem, the differential equation that is being approximated is shown in Eq. 22 where q is a heat transfer source and $\forall$ is the volume. Specifically, the finite difference method approximates the problem by stating the differentials as $dr \approx \Delta r$ and$dt \approx \Delta t$ for small values of Δr and Δt where t is time and$r$ is the radial location.

$$\sum_i q_i = \dot{e}\Delta\forall \tag{22}$$

In this problem, assuming radial symmetry, there are three types of nodes: internal, exterior, and center-line. This first group, internal, is the most general of the three and is solved using information about the two neighboring nodes. The energy transfer equations for the left node and right node can be seen in Eqs. 23 and 24 respectively where q is the heat transferred, k is the thermal conductivity, m is the node number, H is the length dummy variable and T is a temperature at a nodal location with the relative time specified as the exponent.

$$q_{m-\frac{1}{2}} = k2\pi(m - \frac{1}{2})\Delta r H \frac{(T^i_{m-1} - T^i_m)}{\Delta r} \tag{23}$$

$$q_{m+\frac{1}{2}} = k2\pi(m + \frac{1}{2})\Delta r H \frac{(T^i_{m+1} - T^i_m)}{\Delta r} \tag{24}$$

Once the left hand of Eq. 22 has been defined, the right hand side of the equation must be defined. As there is no significant heat generation within a standard hot dog, the effects of heat generation will be ignored. The right hand side of Eq. 22 can then be defined as in Eq. 25 where ρ is the density and c is the specific heat.

$$\dot{e}\Delta\forall = \rho c \frac{\partial T}{\partial t} 2\pi(m\Delta r)\Delta r H \tag{25}$$

Once converted from differential form to finite difference form, Eq. 25 can be written as shown below in Eq. 26.

$$\dot{e}\Delta\forall = \rho c 2\pi(m\Delta r)\Delta r H \frac{T_m^{i+1} - T_m^i}{\Delta t} \tag{26}$$

Once these equations have been established, they can then be simplified and solved for T_m^{i+1}. In order to do this, it is important that the following simplifications are used. Firstly, when k is divided through, it should be noted that $\frac{\rho c}{k} = \frac{1}{\alpha}$ where α is the thermal diffusivity. Next, it is important to note that $Fo_\Delta = \frac{\alpha \Delta t}{\Delta r^2}$ where Fo_Δ is a finite form of the Fourier Number. The resulting equation can be seen below in Eq. 27.

$$T_m^{i+1} = Fo_\Delta(1 - \frac{1}{2m})T_{m-1}^i + Fo_\Delta(1 + \frac{1}{2m})T_{m+1}^i - (2Fo_\Delta - 1)T_m^i \tag{27}$$

For the second group, the center-line nodes, the process is very similar but the equations that define Eq. 22 are different. For the left hand side of Eq 22, there is only one node that can give or take heat from a center node, therefore, it can be written as in Eq. 28

$$q_{\frac{1}{2}} = k2\pi(\frac{1}{2})\Delta r H \frac{(T_1^i - T_0^i)}{\Delta r} \tag{28}$$

The right side of Eq. 22 for a center-line node then simplifies to Eq. 29.

$$\dot{e}\Delta\forall = \rho c\pi(\frac{\Delta r}{2})^2 H \frac{T_0^{i+1} - T_0^i}{\Delta t} \tag{29}$$

Simplifying in a manner similar to the internal node derivation yields the form seem below in Eq. 30

$$T_0^{i+1} = 4Fo_\Delta T_1^i - (4Fo_\Delta - 1)T_0^i \tag{30}$$

For the final group, the external nodes, beyond the considerations of heat transfer between nodes, the convection and radiation from the grill must be considered. Using the form seen in Eq. 31 these effects can be accounted for where h_{tot} is the effective total heat transfer coefficient, T_∞ is the temperature of the air in the grill, and T_s is the surface temperature of the hot dog.

$$q_{tot} = h_{tot}A_s(T_\infty - T_s) \tag{31}$$

In order to use Eq. 31 in the finite difference process, it is useful to convert as much as possible into variables that are common. This conversion can be seen below in Eq. 32.

$$q_{tot} = h_{tot}2\pi(M\Delta r^2)H(T_\infty - T_M^i) \tag{32}$$

Beyond the effects of convective and radiative heat transfer to the surface, the internal node to the left of the exterior must also be considered. Using a similar process as in the center-line and internal node derivations, first we consider the added internal heat transfer which can be seen in Eq. 33.

$$q_{M-\frac{1}{2}} = k2\pi(M - \frac{1}{2})\frac{\Delta r}{2}H\frac{(T^i_{M+1} - T^i_M)}{\Delta r} \tag{33}$$

The right hand side of Eq. 22 for an external node can then be written as in Eq. 34

$$\dot{e}\Delta V = \rho c 2\pi(M\Delta r)\Delta r H\frac{T^{i+1}_M - T^i_M}{\Delta t} \tag{34}$$

Then simplifying using a similar process as above, the resulting node equation can be seen in Eq. 35.

$$T^{i+1}_M = 2Fo_\Delta(1 - \frac{1}{2M})T^i_{M-1} + \frac{2Fo_\Delta h_{tot}\Delta r}{k}T_\infty - (2Fo_\Delta(1 - \frac{1}{2M}) + \frac{2Fo_\Delta h_{tot}\Delta r}{k} - 1)T^i_M \tag{35}$$

Considering the stability is then the last step in this process. The smallest stability criterion requirement is the one that will be used to solve the equations above. The stability criterion for an interior node gives the requirement seen below in Eq. 36.

$$\Delta t < \frac{\Delta r^2}{2\alpha} \tag{36}$$

The stability criterion for an exterior node gives the requirement seen below in Eq. 37.

$$\Delta t < \frac{\Delta r^2}{2\alpha(1 - \frac{1}{2M} + h_{tot}\Delta r})) \tag{37}$$

The stability criterion for an center-line node gives the requirement seen below in Eq. 38.

$$\Delta t < \frac{\Delta r^2}{4\alpha} \tag{38}$$

The smallest Δt is then required by the center-line stability criterion. It can be easily seen that this requirement is smaller than the internal node requirement. The external requirement, however, has the possibility of being larger, but due to the fact the about 30 nodes are being used for this problem as well as h_{tot} being less than 40 as seen in the analytical solution, this makes the denominator smaller than 2α, therefore, the smallest criterion requirement is the center-line requirement shown in Eq. 38.

The centerline and surface temperature are visualized over time, below in Fig. 6, using MATLAB stemming from the equations above. The code is included in Appendix B.

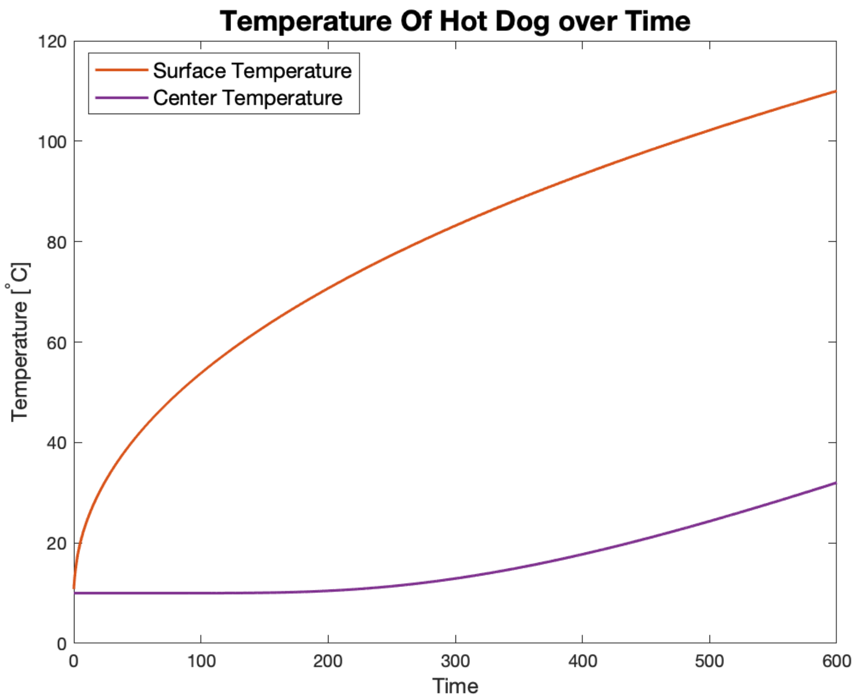

Figure 6 Temperature of surface and centerline of hot dog over time.

Figure 7 displays a graph of the temperature distribution at various times and points along the radius of the hot dog. These various times include 30 seconds, 120 seconds, and the final time which is 7 minutes and 52 seconds.

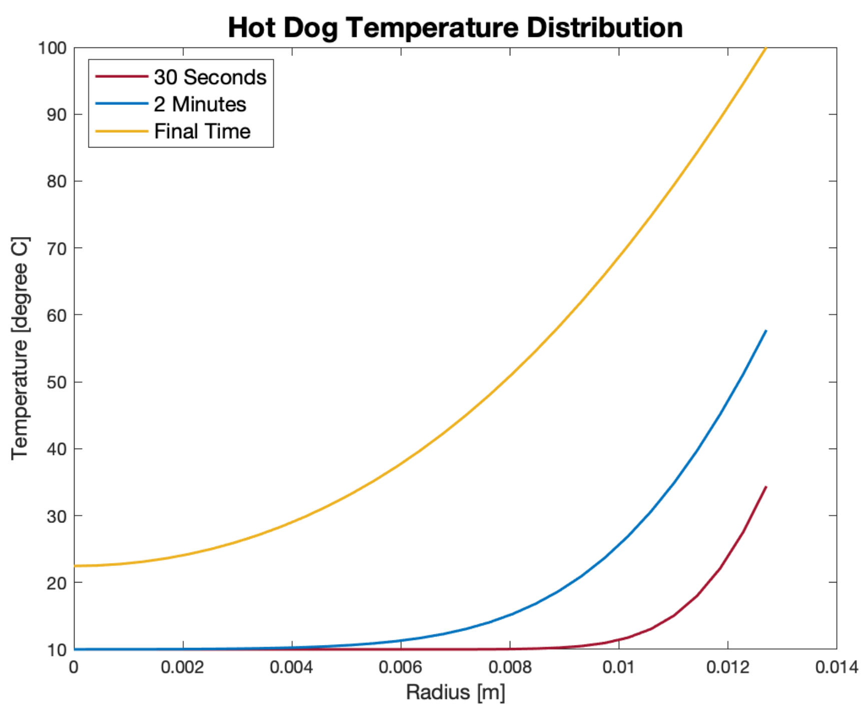

Figure 7 Temperature distributions at various points in time using finite differencing.

Figure 6, at the initial start time, shows that both surface and centerline temperatures have the same initial temperature. As expected the surface heats faster than the center of the hot dog. However, the center of the hot dog never reaches a temperature of 68°. Unfortunately, this is not an ideal temperature to eat a hot dog. However, it is worth pointing out that at no matter what temperature the hot dog is cooked, it can still be consumed safely. We believe that there is an error in the MATLAB code for the graph. The MATLAB code recalculates the h_{total} value after each iteration occurs.

Each curve in Fig. 6 has a shape that is expected, with the exception of the final temperature. Even though the final temperature is not at the desired temperature, it still follows the same trend and is larger than the other two times. Even due to the error in the code, the 30 second curve and the 120 second show an accurate temperature distribution based off the location in the hot dog. As expected the temperatures are warmer after 120 seconds than at 30 seconds. Each start at 10°C and get large closer to the surface of the hot dog.

Comparison of Solutions

Each method generates similar solutions to how long it takes to grill the perfect hot dog on a grill. The analytical solution suggests 7 minutes and 24 seconds to achieve the desired state, while the finite differencing method returns a cooking time of 7 minutes and 52 seconds. The time from each solution fits perfectly with recommended cook times with a range of 7 to 10 minutes [4].

Even though these two solutions were similar in each method, the finite differencing solution is a better method for determining the best time to grill a perfect hot dog. This solution is able to consider more variables that would create an ideal environment for perfection. Note how in Fig. 5 the 30 second and 120 second lines are separated before grilling begins. As mentioned earlier, this is due to one-term approximation only being valid for a Fourier number greater than 0.2. As seen in Fig. 7, the finite differencing solution does not have the same issue. Both the surface temperature and the centerline temperature start at the same value.

The finite differencing method uses nodes to calculate temperature throughout the hot dog. There are three types of nodes used: interior, center, and exterior. The interior and center nodes only consider conduction from other nodes. The exterior nodes take conduction, radiation, and convection into consideration.

Experimentation proved that this might be your grandmother's perfect hot dog, mine prefers a crispy dog left on the grill for 45 minutes. After a short 4 minutes and 35 seconds the hot dog reached the ideal temperature of 68°C. Conduction was ignored in the analytical solution which is why the cook time was three minutes faster than the solution suggests. Additionally, the conduction due to grill grates is not taken into consideration for either solution. One more likely error is due to the inability to control uniform flow from the coals.

References

[1] Grainger, 2021, "Adjustible Powder Coated Steel Pedestal Grill," Grainger, Inc., accessed May 6, 2021, https://www.grainger.com

[2] Bergman, T., Lavine, A., Incropera, F., and Dewitt, D., 2011, *Fundamentals of Heat and Mass Transfer 7th Edition*, John Wiley & Sons, Hoboken, NJ.

[3] Yunus A., C., 2008, *Thermodynamics : An Engineering Approach*, Boston: McGraw-Hill Higher Education, New York, NY.

[4] Grainger, "How To Grill Hot Dogs Perfectly," Ready Set Eat, accessed May 6, 2021, https://www.readseteat.com

A MATLAB Code for Analytical Solution

```
%% Centerline Temperatue over Time

clear all;
close all;
clc;

% Just having fun with colors:
colorb = [0, 0.4470, 0.7410];

t = 0:1:445;             % [s]
T_i = 10;                % initial hot dog temp [C]
T_inf = 250;             % air temp at hot dog [C]
C1 = 1.0995;              % [unitless]
zeta = 0.88014;            % [rad]

T_0 = T_inf + (T_i-T_inf)*C1*exp(-1*zeta^2*0.00108.*t);

plot(t,T_0,'Color',colorb,'LineWidth',1.5)
xlabel('Time [s]','FontSize',12);
ylabel('Centerline Temperature [C]','FontSize',12);

title('Surface Temperature Over Time','FontSize',16)

return

% Save as image;
saveas(gcf, 'centerline.png')
```

```
%% Surface Temperatue over Time

clear all;
close all;
clc;

% Just having fun with colors:
colorb = [0, 0.4470, 0.7410];


t = 0:1:445;             % [s]
T_i = 10;                % initial hot dog temp [C]
T_inf = 250;             % air temp at hot dog [C]
C1 = 1.09995;              % [unitless]
zeta = 0.88014;            % [rad]
T_s = T_inf + (T_i-T_inf)*C1*exp(-1*zeta^2*0.00108.*t)*besseli(0,zeta);

plot(t,T_s,'Color',colorb,'LineWidth',1.5)
xlabel('Time [s]');
ylabel('Surface Temperature [C]');

title('Surface Temperature Over Time','FontSize',16)
% ask why surface temp starts so negative & only reaches 30 @ max

```

```
return

% Save as image;
saveas(gcf, 'surface.png')
```

```
%% Temperature Distribution at Various Times

clear all;
close all;
clc;

% Just having fun with colors:
colory = [0.9290, 0.6940, 0.1250];
colorr = [0.6350, 0.0780, 0.1840];
colorb = [0, 0.4470, 0.7410];

t1 = 30;                    % [s]
t2 = 120;                   % [s]
t3 = 444.52;                % [s]
r = 0:0.0001:0.0127;        % [m]
r_o = 0.0127;               % [m]


T_i = 10;                   % initial hot dog temp [C]
T_inf = 250;                % air temp at hot dog [C]
C1 = 1.09995;                 % [unitless]
zeta = 0.88014;               % [rad]
Z = zeta.*(r./r_o);         % [rad]
bessel = besselj(0,Z);

T1 = T_inf + (T_i-T_inf)*C1*exp(-1*zeta^2*0.00108*t1).*besselj(0,Z);
T2 = T_inf + (T_i-T_inf)*C1*exp(-1*zeta^2*0.00108*t2).*besselj(0,Z);
T3 = T_inf + (T_i-T_inf)*C1*exp(-1*zeta^2*0.00108*t3).*besselj(0,Z);

% disp(bessel)

plot(r,T1,'Color', colorr,'LineWidth',1.5)
hold on
plot(r,T2,'Color', colorb,'LineWidth',1.5)
hold on
plot(r,T3,'Color', colory,'LineWidth',1.5)

legend('30 seconds', '120 seconds', '444.52 seconds', 'Location',...
    'SouthEast','FontSize',12)

xlabel('Position along radius [m]','FontSize',12);
ylabel('Temperature [C]','FontSize',12);

title('Temperature Distribution at Various Times','FontSize',16)

return

% Save as image;
saveas(gcf, 'tempdist.png')
```

```matlab
%% Hot Dog Hrad

clear
close all
clc

%HotDog Constants
d_hd = 0.0127;                       %[m]
r_hd = d_hd/2;                       %[m]
T2 = 283;                            %[K]
em_hd = 0.45;                        %[Unitless]
rho_hd = 880;                        %[kg/m^3]
Cp_hd = 3350;                        %[J/kgK]
k_hd = 0.52;                         %[W/mK]
alpha_hd = k_hd/(rho_hd*Cp_hd);      %[m^2/s]
A_hd = pi*d_hd;                      %[m^2]

%Coals Constants
w_c = 0.5842;                        %[m]
T1 = 723;                            %[K]
em_c = 0.80;                         %[Unitless]
A_c = w_c;                           %[m^2]

%Constants
sigma = 5.67*10^-8;                  %[W/m^2*K^4]

%View Factor
s1 = w_c/2;                          %[m]
s2 = -s1;                            %[m]
y = zeros(139,1);
y(1) = 0.064;                        %[m]

%Heat Transfer Coefficient
hrad = zeros(139,1);                 %[W/m^2*K]

i = 1;
while i<139
F12 = (r_hd/(s1-s2))*(atan(s1/y(i)) - atan(s2/y(i)));
F21 = (A_c*F12)/A_hd;
qabs = em_c*em_hd*A_hd*F21*sigma*T1^4;
T2s = T2/((em_c*F21)^(1/4));
qemit = em_c*em_hd*A_hd*F21*sigma*T2s^4;
qrad = qabs - qemit;
hrad(i) = qrad/(A_hd*(T1-T2s));
y(i+1) = y(i) + 0.001;
i = i + 1;
end

y = 28*0.001 + 0.064;
disp(y)
disp(hrad)
```

B MATLAB Code for Numerical Finite Numerical Differencing Solution

```
%% Finite Differencing Soultion

close all;
clear;
clc;

% Just having fun with colors:
colory = [0.9290, 0.6940, 0.1250];
colorr = [0.6350, 0.0780, 0.1840];
colorb = [0, 0.4470, 0.7410];
coloro = [0.8500, 0.3250, 0.0980];
colorp = [0.4940, 0.1840, 0.5560];

TdogSurf = 10 + 273; % initial temperature of hot dog surface
TdogCent = 10 + 273; % initial temperature of hot dog centerline
Tair = 236.5 + 273; %  temperature of the air during the process
Tcoals = 450 + 273;

% These values can be changed to change per grilling situation etc
L = 0.092;
r = 0.0127;
s1 = 0.2921;
s2 = -s1;
F12 = (r / (s1 - s2)) * (atan(s1/L) - atan(s2 / L));
Ac = 0.5842;
Ad = 0.0399;
F21 = (F12 * Ac) / Ad;



% three types of 30 nodes:
% Node 1 (center node-matlab arrays start at 1) is first lets make two
% arrays one that has current temps and one that has next temps

currentTemps = zeros(31) + 283; % sets all the inital temps to 10 celsius
nextTemps = zeros(31);
dr = 0.0254 / 30;
alpha = 0.52 / (880 * 3350);
Fo = (alpha * 0.2) / (dr^2);
surfaceTemps = zeros(3001);
centerTemps = zeros(3001);
Tdist30 = zeros(31);
Tdist120 = zeros(31);
Tdist474 = zeros(31);
tcount = 1;

for t=0:0.2:600
    % first recalculate htot
    Tfilm = TdogSurf + Tair / 2;
    Re = (0.323 * 0.0254) / v(Tfilm);
    constantResults = constants(Tfilm);
    C = constantResults(1);
    m = constantResults(2);
    NewPr = Pr(Tfilm);
```

```
    newk = k(Tfilm);
    Nu = C * (Re ^ m) * (NewPr ^ (1/3));
    hconv = (newk * Nu) / 0.0254;
    T2star = TdogSurf / ((0.8 * F21)^(1/4));
    hrad = 0.8*0.45*F21*5.67*10^(-8)*(Tcoals + T2star)*(Tcoals^2 + T2star^2);
    htot = hconv + hrad;


    % Now to plug and chug again...
    % start with m=1 for first node and correct the m values used in
    % calculations--this fixes a nonlogical error (thanks matlab for
    % starting arrays at 1 instead of 0!)
    nextTemps(1) = 4*Fo*currentTemps(2) - (4*Fo - 1)*currentTemps(1);
    for m=2:30
        nextTemps(m) = Fo*(1 - 1/(2*(m-1)))*currentTemps(m - 1) + ...
        Fo*(1 + 1/(2*(m-1)))*currentTemps(m + 1) - ...
        (2*Fo - 1)*currentTemps(m);
    end


    % now we do the last node which is the exterior
    nextTemps(31) = ((2*Fo) * (1 - (1/(2 * 30))))*currentTemps(30) + ...
        ((2*Fo*htot*dr*Tair) / 0.52) - (((2*Fo * (1 - (1/(2 * 30)))) + ...
        ((2*Fo*htot*dr) / 0.52)) - 1)*currentTemps(31);


    % update values in order to check temps
    TdogSurf = nextTemps(31);
    TdogCent = nextTemps(1);
    surfaceTemps(tcount) = TdogSurf - 273; % [degrees C]
    centerTemps(tcount) = TdogCent - 273; % [degrees C]
    tcount = tcount + 1;
    if (TdogSurf > 100 + 273)
        disp("Surface temp hit 100 C at " + t + " seconds");
    end
    if (TdogCent > 68 + 273)
        disp("Center temp hit 68 C at " + t + " seconds");
    end
    if (tcount == 151)
        Tdist30 = nextTemps - 273;
    end
    if (tcount == 601)
        Tdist120 = nextTemps - 273;
    end
    if (tcount == 2371)
        Tdist474 = nextTemps - 273;
    end
    currentTemps = nextTemps;
end

x = linspace(0,600,3001);
plot(x, surfaceTemps(:,1),'Color',coloro,'LineWidth',1.5);
hold on;
plot(x, centerTemps(:,1),'Color', colorp,'LineWidth',1.5);

title("Temperature Of Hot Dog over Time",'FontSize',16);
xlabel("Time",'FontSize',12);
```

```
ylabel("Temperature [^{\circ}C]",'FontSize',12);
legend('Surface Temperature', 'Center Temperature','Location',...
    'NorthWest','FontSize',12); %% Kanye Please notice. urs truly, Ben.

% return
% Save as image;
% saveas(gcf, 'fdiftime.png')

figure();
r = linspace(0,0.0127,31);
plot(r, Tdist30(:,1), 'Color', colorr,'LineWidth',1.5);
hold on;
plot(r, Tdist120(:,1),'Color', colorb,'LineWidth',1.5);
hold on;
plot(r, Tdist474(:,1),'Color', colory,'LineWidth',1.5);

title("Hot Dog Temperature Distribution",'FontSize',16);
xlabel("Radius [m]",'FontSize',12);
ylabel("Temperature [degree C]",'FontSize',12);

legend('30 Seconds', '2 Minutes', 'Final Time','Location',...
    'NorthWest','FontSize',12);

% return
% Save as image;
% saveas(gcf, 'fdifdist.png')


% using a quartic curve fit on my calculator I generated these functions to
% calculate the k and Pr values for a given temperature. These functions
% are fitted to the table of material properties for air at various
% temperatures in the textbook. The fits appear very tight and there should
% be very little error for the temperature ranges of this problem

function [k] = k(T)
    k = 3.171763*10^(-14)*(T^4) - 5.90229*10^(-11)*(T^3) + ...
        4.5853086*10^(-9)*(T^2) + 8.8867888*10^(-5)*T + 5.2007224*10^(-4);
end

function [Pr] = Pr(T)
    Pr = -2.08438*10^(-13)*(T^4) - 1.72787*10^(-11)*(T^3) + ...
         8.0062429*10^(-7)*(T^2) - 6.946059*10^(-4)*T + 0.8457812178;
end

function [v] = v(T)
    v = 2.25852*10^(-17)*(T^4) - 8.32093*10^(-14)*(T^3) + ...
        1.726732*10^(-10)*(T^2) + 1.063728*10^(-8)*T - 7.610836*10^(-7);
end

% This function is a little different in that it is just a way to grab
% values form the textbooks table it doesn't approximate the value any
% further.

function [result] = constants(Re)
    C = 0;
    m = 0;
    if (Re < 400000)
```

```
        C = 0.027;
        m = 0.805;
    end
    if (Re < 40000)
        C = 0.193;
        m = 0.618;
    end
    if (Re < 4000)
        C = 0.683;
        m = 0.466;
    end
    if (Re < 40)
        C = 0.911;
        m = 0.330;
    end
    if (Re < 4)
        C = 0.989;
        m = 0.330;
    end
    result = [C, m];
end
```