Design and Analysis of Thermal and Structural Systems Through Implementation of Open-Source Finite Element Software

Max Kessler

May 17, 2019

2.S976 Finite Element Methods for Mechanical Engineers
Professor Anthony Patera
Department of Mechanical Engineering
Massachusetts Institute of Technology

Table of Contents

Overview

This is the project report for 2.S976, Finite Element Methods for Mechanical Engineers, a course taught at the Massachusetts Institute of Technology (MIT) in spring 2019. The project used open-source finite element software developed in MATLAB to analyze various user-defined models. While some coding was performed, this report focuses on the formulation, implementation, and verification of the software. The report is motivated by the following questions [1]:

- What mathematical equations can be used to describe the behavior of a system?
- How can partial differential equations be strategically formulated?
- Does a design optimization satisfy problem constraints?
- How can the accuracy of the finite element solution be assessed?

In answering these questions, the finite element method (FEM) is shown to be a powerful tool for the design and analysis of mechanical systems common to engineering. This report is organized in five chapters. Chapter 1 explores the Rayleigh-Ritz method, which lays the foundation for FEM. Chapter 2 introduces the finite element method for one-dimensional (1D), second-order boundary value problems (BVPs). Chapter 3 adds time-dependency. Chapter 4 applies FEM to 1D, fourth-order BVPs. Chapter 5 was a design competition at the end of the course, and it is presented as a series of slides in the addendum.

While there exist very robust third-party finite element software packages capable of solving complex 3D problems, there is much to gain from using more simplistic software. Third-party software is often a black box with inflexible inputs and outputs that limit the types of problems able to be solved. Furthermore, the level of understanding required to implement FEM from the ground up gives an engineer an appreciation for the capabilities and limitations of the finite element method.

This project uses FEM to model the behavior of various thermal and structural systems. Problems include: analyzing heat transfer through frustums, fins, and walls; optimizing parameters for cooking a burger and tuning a xylophone bar; and designing the tallest column to withstand its own weight.

Chapter 1: The Rayleigh-Ritz Method

ABSTRACT

The Rayleigh-Ritz method is a powerful computational tool for modeling complex boundary value problems. This chapter implements the algorithm through two illustrative problems: the temperature profile of a conical frustum and the heat flux into a right-cylinder fin. The results provide confidence for the correct implementation of the method. The influence of key problem parameters on the accuracy of the approximation is also investigated.

INTRODUCTION

The first problem, Model I, describes quasi-1D conduction through a conical frustum with adiabatic lateral surfaces and heat fluxes through its end surfaces (**Figure 1.1**).

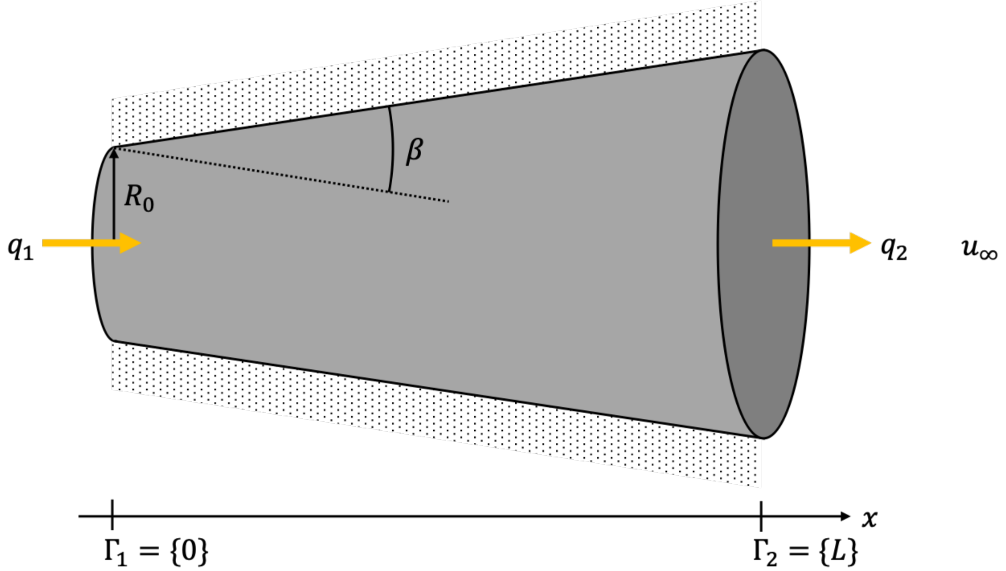


Figure 1.1: Frustum geometry of Model I.

The differential equation and boundary conditions for Model I are given by [2]:

$$-k\frac{d}{dx}\left(\pi R_0^2\left(1+\beta\frac{x}{L}\right)^2\frac{du}{dx}\right) = 0 \ \text{ in } \mathit{\Omega} \tag{1}$$

$$k\frac{du}{dx} = -q_1 \ \text{ on } \mathit{\Gamma}_1 \tag{2}$$

$$-k\frac{du}{dx} = \eta_2(u-u_\infty) \ \text{ on } \mathit{\Gamma}_2 \tag{3}$$

where:

$\mathit{\Omega} = (0, L)$ is the domain in x
$\mathit{\Gamma}_1$ and $\mathit{\Gamma}_2$ are the left and right axial surfaces, respectively
L is the length of the body [m]
R_0 is the radius at $x = 0$ [m]
β is central angle [radians]
k is the thermal conductivity of the body [W/m °C]

η_2 is the heat transfer coefficient between the right surface and the air [W/m^2 °C]
q_1 is the heat flux through the left surface the body [W/m^2]
u_∞ is the ambient temperature [°C]
$u(x)$ is the temperature profile of the body [°C]

The exact solution to the frustum problem is:

$$u = u_\infty + \frac{q_1 L}{k}\left(\frac{1+\beta+\left(\frac{k}{\eta_2 L}\right)}{(1+\beta)^2} - \frac{\left(\frac{x}{L}\right)}{1+\beta\left(\frac{x}{L}\right)}\right) \tag{4}$$

Note that for $\beta = 0$, u depends linearly on x:

$$u = u_\infty + \frac{q_1 L}{k}\left(1+\left(\frac{k}{\eta_2 L}\right) - \frac{x}{L}\right) \tag{5}$$

The temperature at the left end of the frustum is the output of interest:

$$\begin{aligned} s &\equiv u(x=0) \\ s^{RR} &\equiv u^{RR}(x=0) \end{aligned} \tag{6}$$

where the superscript denotates functions derived from the Rayleigh-Ritz approximation.

The second problem, Model II, describes the heat transfer of a right-cylinder fin with constant temperature on its left surface and zero heat flux through its right surface (**Figure 1.2**).

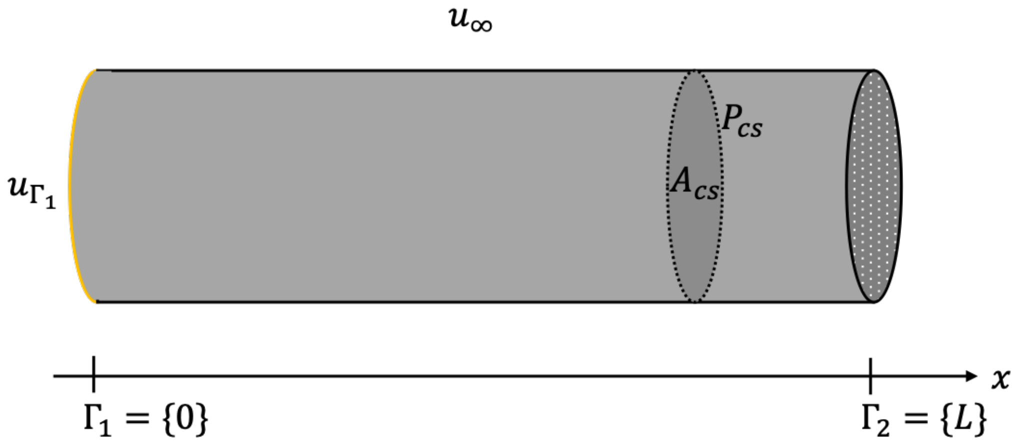


Figure 1.2: Fin geometry of Model II.

The differential equation and boundary conditions for Model II are given by [2]:

$$-k\mathrm{A}_{cs}\frac{d^2u}{dx^2} + \eta_3 P_{cs}(u-u_\infty) = 0 \ \text{ in } \Omega \tag{7}$$

$$u = u_{\Gamma_1} \ \text{ on } \Gamma_1 \tag{8}$$

$$-k\frac{du}{dx} = 0 \ \text{ on } \Gamma_2 \tag{9}$$

where the variables have the same assignments as Model I, except for:

A_{cs} is the cross sectional area of the body [m^2]
P_{cs} is the cross sectional perimeter of the body [m]
η_3 is the heat transfer coefficient between the right surface and the air [W/m^2 °C]
u_{Γ_1} is the temperature of the left surface of the body [°C]

A quantity of interest in Model II is the non-dimensionalized fin parameter, which describes the right surface (tip) condition. It is defined as:

$$\mu_0 = \frac{\eta_3 P_{cs} L^2}{k A_{cs}} \tag{10}$$

The exact solution to the fin problem is given by:

$$u = u_\infty + \left(u_{\Gamma_1} - u_\infty\right) \frac{\cosh\left(\sqrt{\mu_0}\left(1 - \frac{x}{L}\right)\right)}{\cosh\left(\sqrt{\mu_0}\right)} \tag{11}$$

The heat flux into the left surface of the fin is the output of interest:

$$\begin{gathered} s \equiv -k \frac{du}{dx}(x=0) \\ s^{RR} \equiv -k \frac{du^{RR}}{dx}(x=0) \end{gathered} \tag{12}$$

BACKGROUND

The Rayleigh-Ritz method is a powerful, numerical approximation tool that can be applied to the frustum and fin problems. In these cases, knowing the exact solutions provides a means to verify the results of the Rayleigh-Ritz method. In other cases, when an analytical solution is unknown or unsolvable, the Rayleigh-Ritz method may be the best approach for finding a solution.

The method requires a candidate function, w, as a best guess for the unknown solution, u. The problem's boundary conditions and w are used to generate an energy functional, $\Pi(w)$, the total energy of the system. The Minimization Proposition says that the actual solution satisfies [3]:

$$\Pi(u) < \Pi(w) \ \ \forall w \in X, w \neq u \tag{13}$$

This implies that given two approximations of u, w_1 and w_2, if $\Pi(w_1) < \Pi(w_2)$, then w_1 is a better approximation of u. This means that a candidate function that minimizes the energy functional can be sought after without knowing the actual solution. The Rayleigh-Ritz method finds the best candidate function through the linear combination of a selection of functions with real coefficients. In other words, given n^{RR} basis functions $\{\psi_1, \psi_2, \dots, \psi_{n^{RR}}\}$, the objective is to find the coefficients $\{\alpha_1^{RR}, \alpha_2^{RR}, \dots, \alpha_{n^{RR}}^{RR}\}$ that minimize the energy functional:

$$\min \Pi\left(\sum_{i=1}^{n^{RR}} \alpha_i^{RR}\, \psi_i\right) \rightarrow \underline{A}\, \underline{\alpha}^{RR} = \underline{F} \tag{14}$$

The protocol for evaluating the Rayleigh-Ritz coefficients depends on the problem's boundary conditions. The formulation is examined in Chapter 2, where several types of boundary conditions are defined. For now, the results are given. For Model I, the Rayleigh-Ritz approximation is:

$$u^{RR}(x) = \sum_{i=1}^{n^{RR}} \alpha_i^{RR}\,\psi_i(x) \tag{15}$$

The energy functional is:

$$\begin{aligned}\Pi(w) = \frac{1}{2}\int_0^L k\pi R_0^2\left(1+\beta\frac{x}{L}\right)^2\left(\frac{dw}{dx}\right)^2 dx + \frac{1}{2}\left(\eta_2\pi R_0^2(1+\beta)^2 w^2(L)\right)\\ - q_1\pi R_0^2 w(0) - \eta_2\pi R_0^2(1+\beta)^2 u_\infty w(L)\end{aligned} \tag{16}$$

And the elemental matrices of Eq. (14) are:

$$A_{ij} = \int_0^L k\pi R_0^2\left(1+\beta\frac{x}{L}\right)^2\frac{\partial\psi_i}{\partial x}\frac{\partial\psi_j}{\partial x}dx + \eta_2\pi R_0^2(1+\beta)^2\psi_i(L)\psi_j(L)\quad 1\le i,j\le n^{RR} \tag{17}$$

$$F_i = q_1\pi R_0^2\psi_i(0) + \eta_2\pi R_0^2(1+\beta)^2 u_\infty\psi_i(L)\quad 1\le i\le n^{RR} \tag{18}$$

For Model II, the Rayleigh-Ritz approximation is:

$$u^{RR}(x) = u_{\Gamma_1}\psi_0(x) + \sum_{i=1}^{n^{RR}} \alpha_i^{RR}\,\psi_i(x) \tag{19}$$

The energy functional is:

$$\Pi(w) = \frac{1}{2}\int_0^L \left[kA_{cs}\left(\frac{dw}{dx}\right)^2 + \eta_3 P_{cs} w^2\right]dx - \int_0^L \eta_3 P_{cs} u_\infty w dx \tag{20}$$

And the elemental matrices are:

$$A_{ij} = \int_0^L \frac{\partial\psi_i}{\partial x}\frac{\partial\psi_j}{\partial x} + \frac{\eta_3 P_{cs}}{kA_{cs}}\psi_i(x)\psi_j(x)\,dx\quad 1\le i,j\le n^{RR} \tag{21}$$

$$F_i = \int_0^L \frac{\eta_3 P_{cs} u_\infty}{kA_{cs}}\psi_i(x)dx\quad 1\le i\le n^{RR} \tag{22}$$

RESULTS

Implementation of exactinclude

Two sets of basis functions are considered in Models I and II. The first one, `exactinclude`, is a set of basis functions that includes the exact solution. In Model I, `exactinclude` contains: $\psi_1(x) = u(x)$ of Eq. (4) and $\psi_2(x) = x$. The candidate function is $w = \alpha_1^{RR}u(x) + \alpha_2^{RR}x$. The Minimization Proposition states that the energy functional is absolutely minimized when $u(x)$ is used as the candidate function. Therefore, the Rayleigh-Ritz method, seeking to minimize $\Pi(w)$, returns $\alpha_1^{RR} = 1$ and $\alpha_2^{RR} = 0$, which can be seen in the left plot of **Figure 3** by the zero value of $\alpha_2^{RR}\psi_2$ and the nonzero value of $\alpha_1^{RR}\psi_1$. The right plot shows that u^{RR} perfectly matches u with zero error in the relative output temperature, which is defined as

$|s - s^{RR}|/|s|$ with s from Eq. (6). (Note: the relative output error is displayed in **Figure 1.3** as 9.5x10^{-16} instead of 0. This is simply an artifact of MATLAB's floating-point accuracy.)

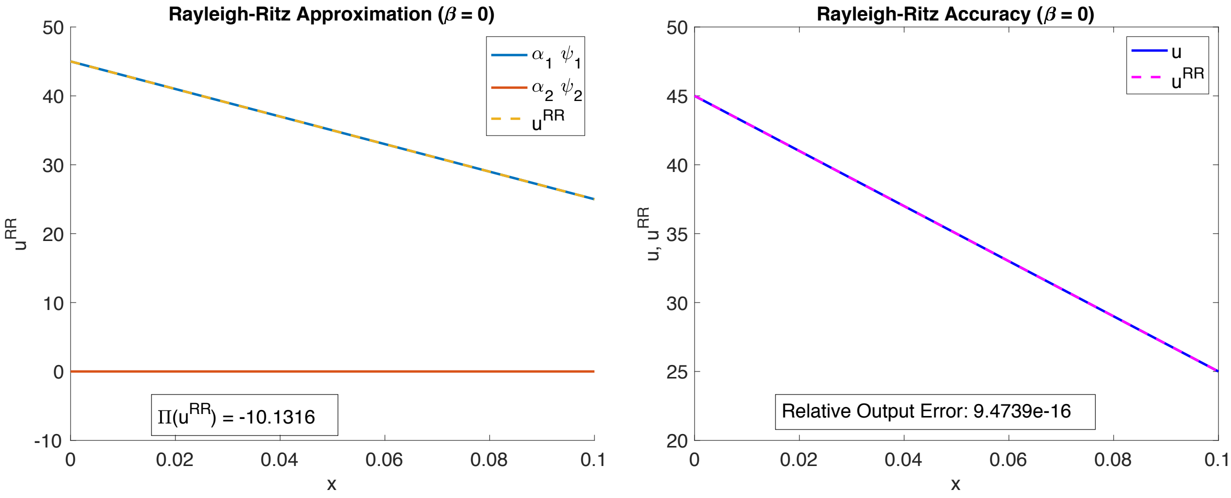


Figure 1.3: Results of `exactinclude` for Model I with $\beta = 0$. The left plot shows the contributions of $\alpha_1^{RR}\psi_1$ and $\alpha_2^{RR}\psi_2$ to construct the Rayleigh-Ritz approximation u^{RR}. The right plot compares u^{RR} to the exact solution u.

In Model II, `exactinclude` contains: $\psi_0(x) = u(x)/u_{\Gamma_1}$for $u(x)$ of Eq. (11) and $\psi_1(x) = x$. As with Model I, the results of Model II show that the Rayleigh-Ritz method chooses coefficients for the combination of ψ_0 and ψ_1 that best approximates u (left plot of **Figure 1.4**). Since u is a basis function, there is a perfect match between u^{RR} and u (right plot) with zero error in the relative output heat flux, which is defined as $|s - s^{RR}|/|s|$ with s from Eq. (12).

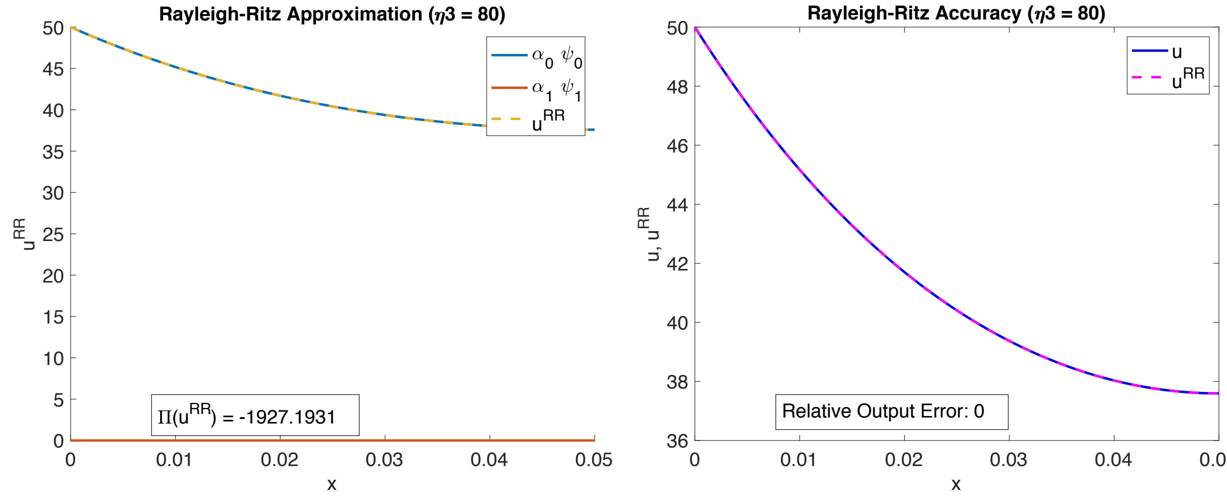


Figure 1.4: Results of `exactinclude` for Model II with $\eta_3 = 80$. The left plot shows the contributions of $\alpha_0^{RR}\psi_0$ and $\alpha_1^{RR}\psi_1$ to construct the Rayleigh-Ritz approximation u^{RR}. The right plot compares u^{RR} to the exact solution u.

The results of `exactinclude` from Models I and II provide evidence that the code implements the Rayleigh-Ritz method correctly for the `exactinclude` basis functions.

Implementation of constlinquad

The second set of basis functions considered by Models I and II is `constlinquad`, which contains monomials of the indeterminate x. Now, the exact solution is not included in the set, and the goodness of the Rayleigh-Ritz approximation depends on the type and number of basis functions. Results for $\beta = 0$ in Model I are shown in **Figure 1.5** on the next page. For $n^{RR} = 1$ (top row of plots), the relative error output is 44.4%, which is not a great approximation. For $n^{RR} = 2$ (middle row of plots), the error is 0%, which indicates that $\alpha_1^{RR}\psi_1 + \alpha_2^{RR}\psi_2$ captures the exact solution. This finding agrees with Eq. (5), which states that u depends linearly on x for $\beta = 0$, and provides confidence that the model is working. Furthermore, for $n^{RR} = 3$ (bottom row of plots), it is evident that ψ_3 is not necessary for the model because $\alpha_3^{RR} = 0$ and the error is still zero.

Results for $\eta_3 = 80$ in Model II are shown in **Figure 1.6** on the next page. For $n^{RR} = 0$ (top row of plots), relative output error is 100%, which is a very bad approximation. Remarkably, the approximation improves with $n^{RR} = 1$ to 51.6% error and becomes quite good with $n^{RR} = 2$ at 7.1% error. The observation that increasing the number of basis functions improves the approximation is another piece of evidence that Rayleigh-Ritz method is implemented correctly.

DISCUSSION

Overall Assessment of Code

Considering the implementation of `exactinclude` and `constlinquad` together provides a unified means of assessing the accuracy of the code. According to the Minimization Proposition, Eq. (13), the energy functional is absolutely minimized when its argument, the candidate function, is the actual solution. In `exactinclude`, where the candidate function contains the exact solution, the absolute minimum of the energy functional is found: $\Pi(u)$. On the other hand, in `constlinquad`, where the candidate function does not contain the exact solution, the local minimum of the energy functional is found: $\Pi(u^{RR})$. A comparison of $\Pi(u^{RR})$ and $\Pi(u)$ provides a metric by which to gauge the goodness of the Rayleigh-Ritz approximation:

- In Model I, $\Pi(u)$ = -10.13, while $\Pi(u^{RR})$ = -9.82, -10.31, and -10.31 for n^{RR} = 1, 2, and 3, respectively. It is apparent that for $n^{RR} \geq 2$, the Rayleigh-Ritz approximation achieves the actual solution because $\Pi(u^{RR}) = \Pi(u)$. Thus, two shape functions are sufficient to exactly represent Model I.
- In Model II, $\Pi(u)$ = -1927.19, while $\Pi(u^{RR})$ = 1600, -1221.57, and -1921.75 for n^{RR} = 0, 1, and 2, respectively. As n^{RR} increases, the Rayleigh-Ritz method better approximates the actual solution because $\Pi(u^{RR})$ approaches $\Pi(u)$. This agrees with the theory that as more shape functions are used, the energy functional decreases and the approximation improves.

This assessment presents evidence of the successful implementation of the Rayleigh-Ritz method to model heat transfer through a conical frustum and right cylinder fin. These examples are among a large set of problems in engineering that have particular geometries and boundary conditions that make deriving analytical solutions difficult. The Rayleigh-Ritz method is the foundation of a toolset of computational methods.

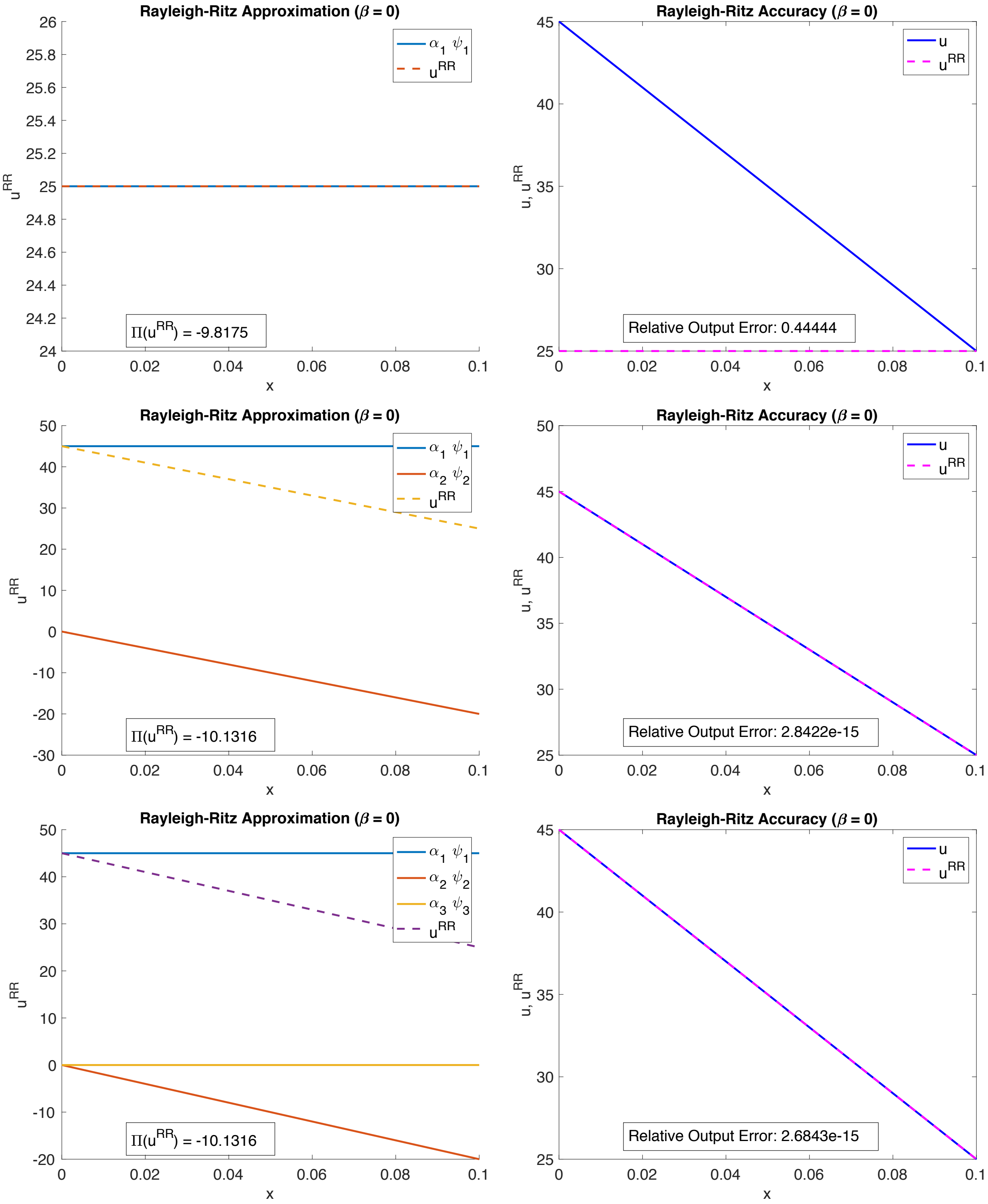


Figure 1.5: Results of `constlinquad` for Model I with $\beta = 0$.
In the top row, $n^{RR} = 1$ and $\psi_1(x) = 1$.
In the middle row, $n^{RR} = 2$, $\psi_1(x) = 1$ and $\psi_2(x) = x$.
In the bottom row, $n^{RR} = 3$, $\psi_1(x) = 1$, $\psi_2(x) = x$, and $\psi_3(x) = x^2$.

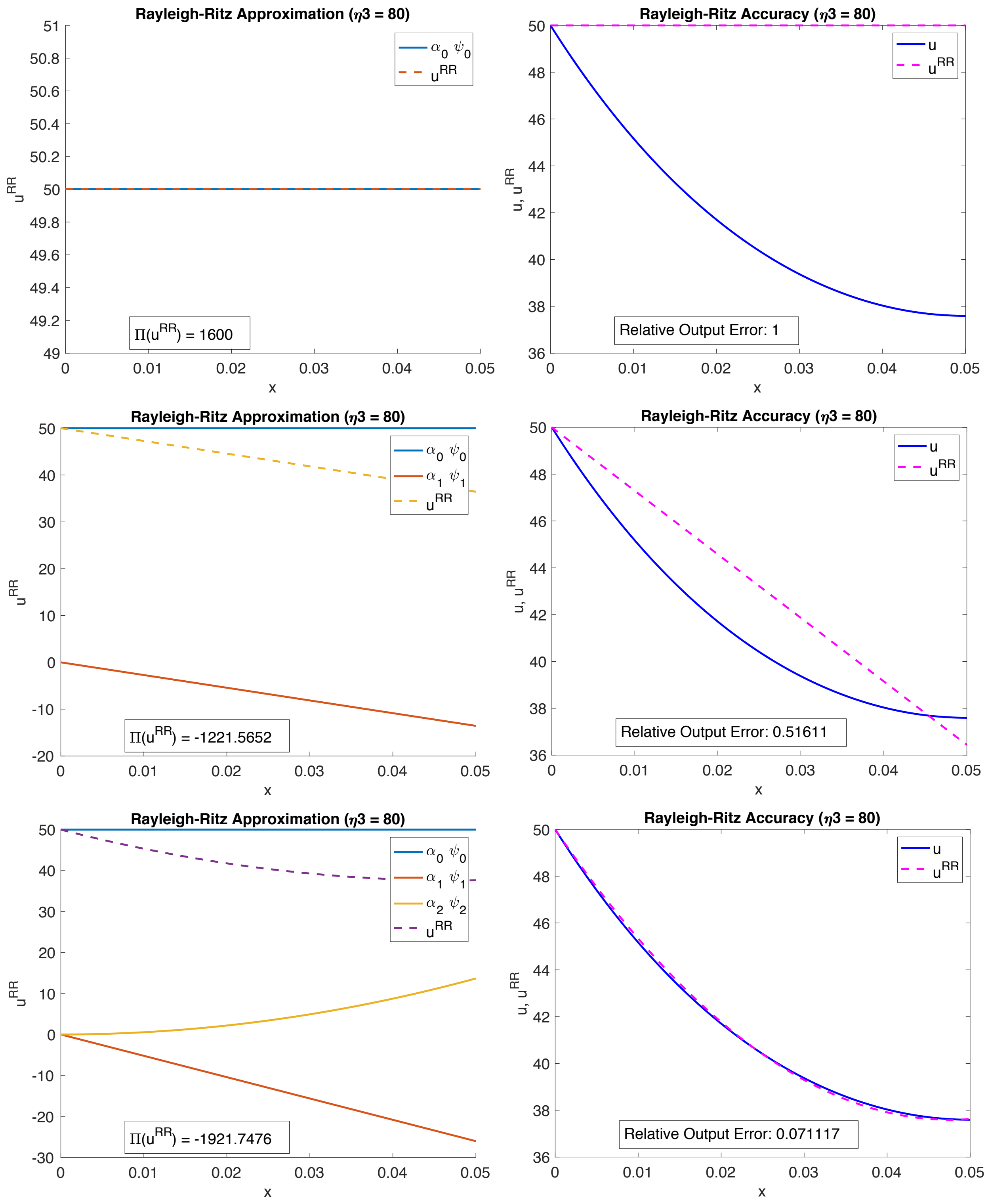


Figure 1.6: Results of `constlinquad` for Model II with $\eta_3 = 80$.
In the top row, $n^{RR} = 0$ and $\psi_0(x) = 1$.
In the middle row, $n^{RR} = 1$, $\psi_0(x) = 1$ and $\psi_1(x) = x$.
In the bottom row, $n^{RR} = 2$, $\psi_0(x) = 1$, $\psi_1(x) = x$, and $\psi_2(x) = x^2$.

Model I β Parameter

The parameter β is the central angle (measured in radians) of the frustum in Model I. When $\beta = 0$ the frustum is simply a right cylinder. The temperature profile is linear for $\beta = 0$ (Eq. (5)), and `constlinquad` has zero relative output error when $n^{RR} \geq 2$. When $\beta \neq 0$, the exact solution (Eq. (4)) takes a shape that cannot be captured with $\psi(x) = 1$, x, and x^2 (**Figure 1.7**). Therefore, the error of the Rayleigh-Ritz approximation increases as β increases and the temperature profile deviates from being linear (**Table 1.1**). Model I uses the Neumann/Robin boundary condition for *one-dimensional* problems. When β is small, the approximation does well because heat transfer occurs primarily in the x-direction (quasi-1D); however, as β increases, heat transfer in the other directions increases and the 1D model becomes less accurate.

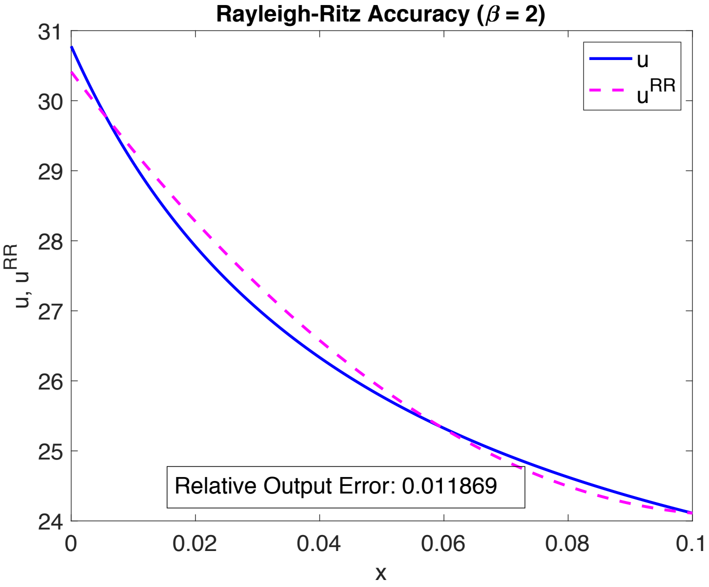


Figure 1.7: Model I `constlinquad` for $\beta = 2$. The relative output error is larger at larger angles.

Table 1.1: β effect on output error

β [rad]	Relative output error [%]
0	0
0.5	0.045896
1	0.301
2	1.1869

Model II μ_0 Parameter

As defined by Eq. (10), μ_0 is the non-dimensionalized fin parameter of Model II. μ_0 describes a length factor over which the temperature changes in the fin, which is an important consideration when designing fins. For example, when μ_0 is on the order of 1 or smaller, the temperature changes across the entire length of the fin, as shown in the left plot of **Figure 1.8**. Albeit in case shown, μ_0 is *too small* for the fin to be an effective transferrer of heat, since $\Delta u <$ 0.3 °C. When μ_0 is much larger than 1, the temperature change occurs across only a portion of the fin, as shown in the right plot of **Figure 1.8**. In this case, the entire length of the fin is not being utilizing, which may be a poor design choice.

The effect of μ_0 on the accuracy of the Rayleigh-Ritz approximation can be investigated by holding the geometry and thermal conductivity of the fin constant and varying the heat transfer coefficient: thus μ_0 is directly proportional to η_3. As shown in **Table 1.2**, the relative output error increases as μ_0 increases. When μ_0 is small, the exact solution has a parabola-like shape (note: it is actually a catenary) that the basis functions of `constlinquad` can approximate. The error in the approximation is small. As μ_0 increases, however, the catenary shape of the exact solution becomes more distinct, and the basis functions give a poor approximation. The error is large.

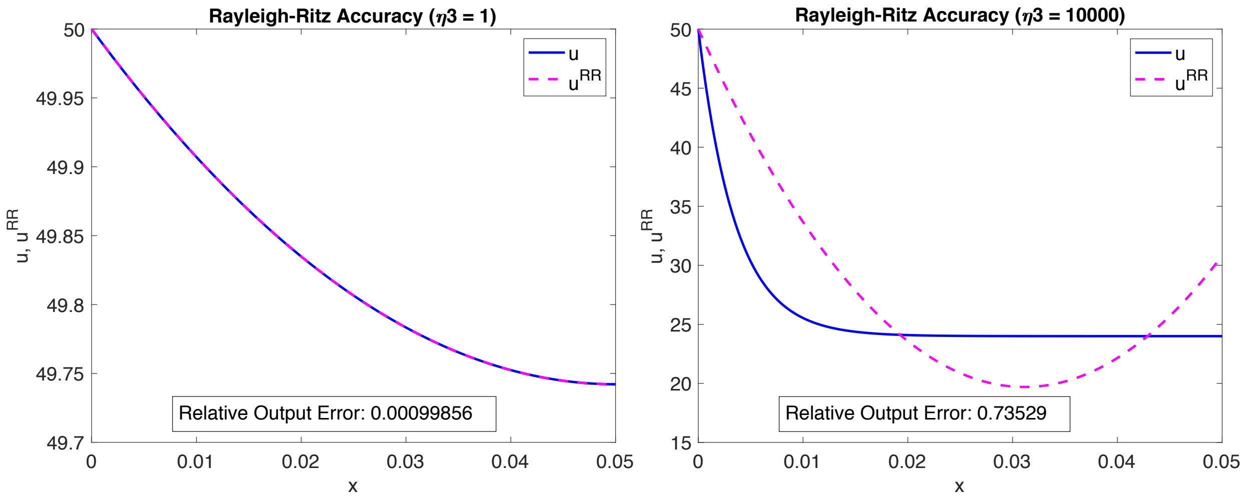


Figure 1.8: Model II `constlinquad`. In the left plot, $\mu_0 = 0.02$ and the Rayleigh-Ritz approximation has a small relative output error. In the right plot, $\mu_0 = 200$ and the approximation is inaccurate.

Table 1.2: η_3 effect on output error

η_3 [W/m^2 °C]	μ_0 [/]	Relative output error [%]
1	0.02	0.099856
80	1.6	7.1117
10000	200	73.529

Chapter 2: The Finite Element Method for 1D 2nd-Order BVPs

ABSTRACT

In Chapter 2, finite element analysis is performed on Models I and II from Chapter 1 as well as a new Model Mine. The method is compared to the Rayleigh-Ritz method, with attention to the structure and advantage a new set of basis functions. Convergence of the approximate solution to the actual solution is tested and error estimators are considered. Finally, two theoretical models are discussed to further verify the finite element method.

INTRODUCTION

Chapter 2 builds upon Models I and II from the last chapter (see *Chapter 1* for model descriptions). A new model, Model Mine, is also used, which describes 1D condition through a wall exposed to air temperatures and heat fluxes through the left and right surfaces (**Figure 2.1**).

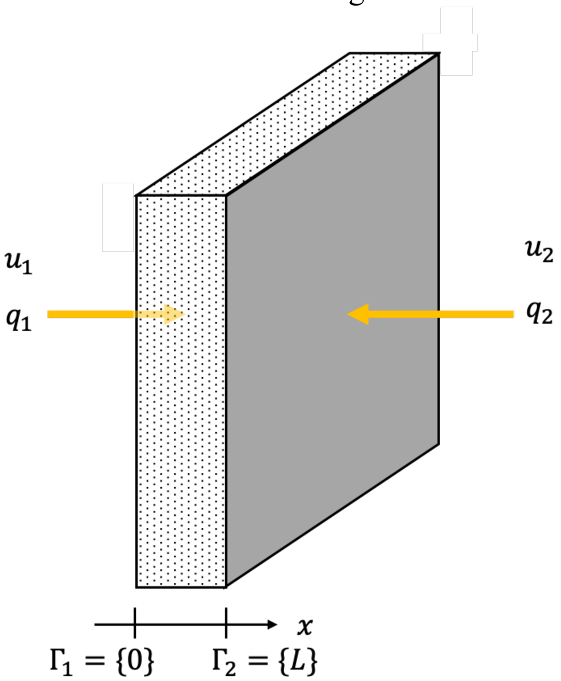


Figure 2.1: Wall of Model Mine

The differential equation and boundary conditions for Model Mine are given by [4]:

$$\frac{d}{dx}\left(k\frac{du}{dx}\right) = 0 \ \text{ in } \Omega \tag{23}$$

$$-k\frac{du}{dx} = \eta_1(u - u_1) - q_1 \ \text{ on } \Gamma_1 \tag{24}$$

$$-k\frac{du}{dx} = \eta_2(u - u_2) - q_2 \ \text{ on } \Gamma_2 \tag{25}$$

where the variables have the same assignments as Models I and II, with the exception of:

η_1 is the heat transfer coefficient between the left surface and the air [W/m^2 °C]

η_2 is the heat transfer coefficient between the right surface and the air [W/m^2 °C]

q_1 is the heat flux through the left surface of the wall [W/m^2]
q_2 is the heat flux through the right surface of the wall [W/m^2]
u_1 is the ambient temperature on the left side of the wall [°C]
u_2 is the ambient temperature on the left side of the wall [°C]

The exact solution to the wall problem is given by:

$$u = \left[\frac{\eta_1 q_2 - \eta_2 q_1 - \eta_1\eta_2 u_1 + \eta_1\eta_2 u_2}{\eta_1 k + \eta_2 k + L\eta_1\eta_2}\right]x + \left[\frac{q_1 + q_2 + \eta_2(u_2 + L(q_1 + \eta_1 u_1)/k) + \eta_1 u_1}{\eta_1 + \eta_2(1 + L\eta_1/k)}\right] \tag{26}$$

Thus, the temperature within the wall follows a linear profile. The output of interest is

$$s \equiv u(x = 0), \tag{27}$$

the temperature of the left surface of the wall.

BACKGROUND

Finite Element Method Formulation

The finite element method (FEM) is a numerical method used to analyze physical systems that can be modeled by partial differential equations subject to boundary conditions. Particularly in problems involving irregular geometry or composite materials, an analytical solution, u, may not exist. FEM produces an approximate solution, u_h, by dividing the domain into a mesh of elements at nodal positions, x_i. The problem's governing equations can be applied to these elements. Union of the elements reproduces the domain and yields a system of equations, which FEM solves through an optimal combination of basis functions, φ_i, and coefficients, u_{h_i} [5]:

$$u_h(x) = \sum_{i=1}^{n} u_{h_i}\,\varphi_i(x) \tag{28}$$

FEM is a special case of the Rayleigh-Ritz method in which the basis functions are piecewise linear ($p = 1$) or piecewise quadratic ($p = 2$) polynomials. In either case the basis function $\varphi_i(x)$ for $1 \le i \le n_{node}$ is defined as:

$$\varphi_i(x^j) = \begin{cases} 1, & j = 1 \\ 0, & j \neq 1 \end{cases} \tag{29}$$

FEM maps a problem's governing equations to general formulations. For a 1D 2nd order boundary value problem, the general differential equation is:

$$-\frac{d}{dx}\left(\kappa(x)\frac{du}{dx}\right) + \mu(x)u = f_\Omega(\mathrm{x}) \text{ in } \Omega \tag{30}$$

The generalized boundary conditions are summarized in **Table 2.1**. The Neumann condition imposes a heat flux, the Robin condition describes convection by a heat transfer coefficient, and the Dirichlet condition specifies a fixed end temperature.

Table 2.1: Generalized boundary conditions for 1D 2nd order BVP

End of domain	Neumann-Robin boundary condition	Dirichlet boundary condition
On $\mathit{\Gamma}_1$	$\kappa(x)\dfrac{du}{dx} = \gamma_1 u - f_{\mathit{\Gamma}_1}$	$u = u_{\mathit{\Gamma}_1}$
On $\mathit{\Gamma}_2$	$-\kappa(x)\dfrac{du}{dx} = \gamma_2 u - f_{\mathit{\Gamma}_2}$	$u = u_{\mathit{\Gamma}_2}$

An energy functional can be constructed from the general formulation:

$$\Pi(u_h) = \frac{1}{2}\int_0^L \left[\kappa(x)\left(\frac{du_h}{dx}\right)^2 + \mu(x){u_h}^2\right]dx + \frac{1}{2}\left(\gamma_1 {u_h}^2(0) + \gamma_2 {u_h}^2(L)\right) - \int_0^L f_\Omega(x) u_h dx - u_h(0) f_{\Gamma_1} - u_h(L) f_{\Gamma_2} \tag{31}$$

The coefficients of the approximate solution are found by minimizing the energy functional:

$$\min \mathit{\Pi}(u_h) \; \rightarrow \; \underline{A}\, \underline{u_h} = \underline{F} \tag{32}$$

where the elements of $\underline{A}$ and $\underline{F}$ are defined by:

$$A_{ij} = \int_0^L \kappa(x)\frac{\partial\varphi_i}{\partial x}\frac{\partial\varphi_j}{\partial x} + \mu(x)\varphi_i\varphi_j dx + \gamma_1\varphi_i(0)\varphi_j(0) + \gamma_2\varphi_i(L)\varphi_j(L) \quad 1 \le i,j \le n \tag{33}$$

$$F_i = \int_0^L f_\Omega(x)\varphi_i dx + f_{\Gamma_1}\varphi_i(0) + f_{\Gamma_2}\varphi_i(L) \quad 1 \le i \le n \tag{34}$$

Implementation of Models

Solving Models I, II, and Mine with FEM follows a systematic workflow. **Table 2.2** summarizes the types of boundary conditions imposed in the models. Each model's governing equations are mapped to Eq. (30) and the generalized boundary conditions (see **Table 2.3**) [6].

Table 2.2: Types of boundary conditions in the models

End of domain	Model I	Model II	Model Mine
On $\mathit{\Gamma}_1$	*Neumann*	*Dirichlet*	*Neumann-Robin*
On $\mathit{\Gamma}_2$	*Robin*	*Neumann*	*Neumann-Robin*

Table 2.3: Mapping models to generalized parameters

Parameter	Heat transfer meaning	Model I	Model II	Model Mine
$\kappa(x)$	Conductivity times area	$k\pi R_0^2\left(1+\beta\frac{x}{L}\right)^2$	kA_{cs}	kA
$\mu(x)$	Heat transfer coefficient times perimeter	0	$\eta_3 P_{cs}$	0
$f_\Omega(\mathrm{x})$	Generalized heat source	0	$\eta_3 P_{cs} u_\infty$	0
γ_1	Generalized heat transfer coefficient	0	$--$	$\eta_1 A$
γ_2	Generalized heat transfer coefficient	$\eta_2\pi R_0^2(1+\beta)^2$	0	$\eta_2 A$
f_{Γ_1}	Generalized heat flux	$q_1\pi R_0^2$	$--$	$(\eta_1\mu_1+q_1)A$
f_{Γ_2}	Generalized heat flux	$\eta_2\pi R_0^2(1+\beta)^2 u_\infty$	0	$(\eta_2\mu_2+q_2)A$
u_{Γ_1}	Body temperature on Γ_1	$--$	u_{Γ_1}	$--$
u_{Γ_2}	Body temperature on Γ_2	$--$	$--$	$--$

The elemental matrices $\underline{A}^N$ and $\underline{F}^N$ are formed from the integral terms of Eq.s (33) and (34), where the superscript indicates that *no* boundary condition has been imposed. Each element is constructed through a mapping to a reference element that features the basis functions. Note: in implementation, a basis function is called a shape function and denoted $\hat{s}$. The reference elements for $p = 1$ and $p = 2$ are shown in **Figure 2.2**.

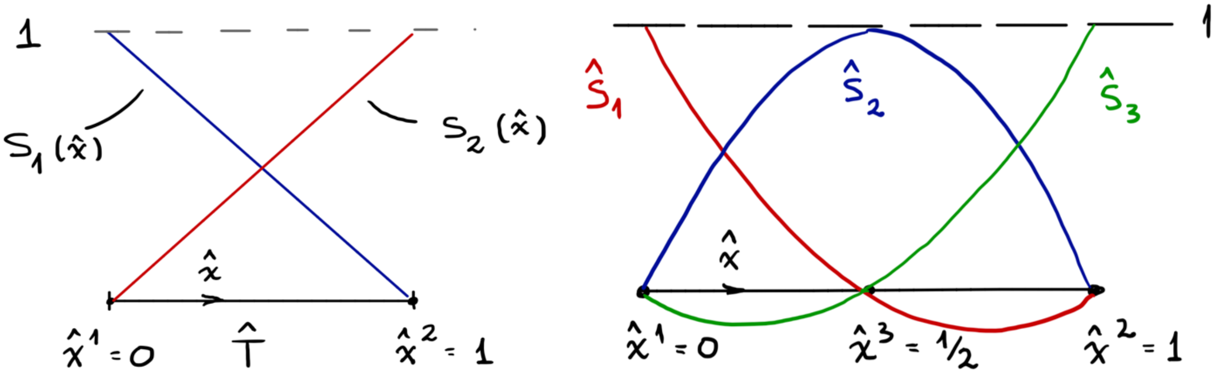


Figure 2.2: Reference element for $p = 1$ (left) [6] and $p = 2$ (right) [7].

Numerical quadrature is used to compute the integrals in A_{ij} and F_i. To impose the boundary conditions, the remaining terms in Eq.s (33) and (34) are applied to $\underline{A}^N$ and $\underline{F}^N$:

$$\begin{aligned} &\underline{\tilde{A}} = \underline{A}^N\text{, but…} \qquad \tilde{A}_{11} = \tilde{A}_{11} + \gamma_1 \qquad \tilde{A}_{n_{el}+1\, n_{el}+1} = \tilde{A}_{n_{el}+1\, n_{el}+1} + \gamma_2 \\ &\underline{\tilde{F}} = \underline{F}^N\text{, but…} \qquad \tilde{F}_1 = \tilde{F}_1 + f_{\Gamma_1} \qquad \tilde{F}_{n_{el}+1} = \tilde{F}_{n_{el}+1} + f_{\Gamma_2} \end{aligned} \tag{35}$$

Note: a feature of the piecewise basis functions is that the boundary conditions are simply imposed on the end elements of the matrices. In Models I and Mine, which impose Neumann/Robin conditions, $\underline{A} = \underline{\tilde{A}}$ and $\underline{F} = \underline{\tilde{F}}$, and Eq. (32) is solved for $\underline{u_h}$, the solution's coefficients. However, in Model II, which imposes a left end Dirichlet condition, $\underline{A}$, $\underline{F}$, and a matrix $\underline{b}$ are extracted from $\underline{\tilde{A}}$ and $\underline{\tilde{F}}$ as shown in **Figure 2.3**. Then,

$$\underline{A}\underline{u_h^0} = \underline{F} - u_{\Gamma_1}\underline{b} \tag{36}$$

is solved for $\underline{u_h^0}$. Finally, $\underline{u_h}$ is formed from $\underline{u_h^0}$ and u_{Γ_1}.

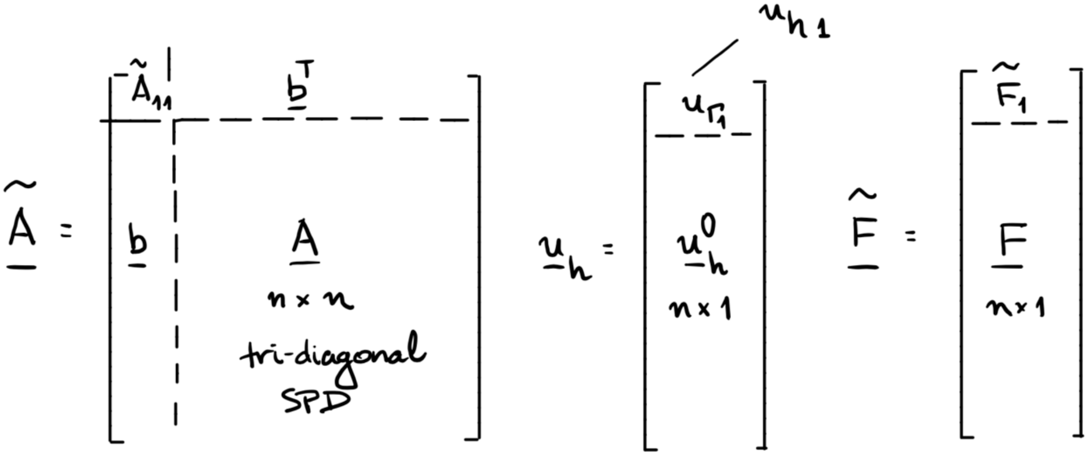


Figure 2.3: Extracting and forming matrices in a Dirichlet left end boundary condition [7].

The piecewise basis functions that characterize FEM have special features. Firstly, a coefficient at node i is physically meaningful: it is exactly the estimate of u at that node, since $u_h(x^i) = u_{h_i}$. In Models I and Mine, FEM immediately outputs nodal temperatures in computing the coefficients. Secondly, the $\underline{A}$ matrix is sparse. It is tridiagonal for $p = 1$ and pentadiagonal for $p = 2$, which makes FEM is computationally efficient.

RESULTS

Model II Results

The output of Model II is shown in **Figure 2.4**. Notice that in the top row, the left plot shows that the finite element (FE) solution is composed of piecewise continuous linear polynomials ($p = 1$) and its derivative is disjointed flat lines. After six uniform refinements, the FE solution is not visibly distinguishable from the actual solution, suggesting convergence. Similar observations can be made for $p = 2$ in the bottom row of plots; note the piecewise quadratic polynomials in the FE solution and the linear polynomials in the FE derivative. Another verification of successful FEM implementation is that the error converges at the expected rates (to be discussed in *Model II Error Analysis*). The results of Model II provide confidence that `form_elem_mat_sver`[8], the code that constructs the elemental matrices, is working properly since Model II has nonzero boundary parameters $\mu(x)$ and f_Ω(x). Furthermore, matrix $\underline{A}$ has the expected sparsity pattern (**Figure 2.5**).

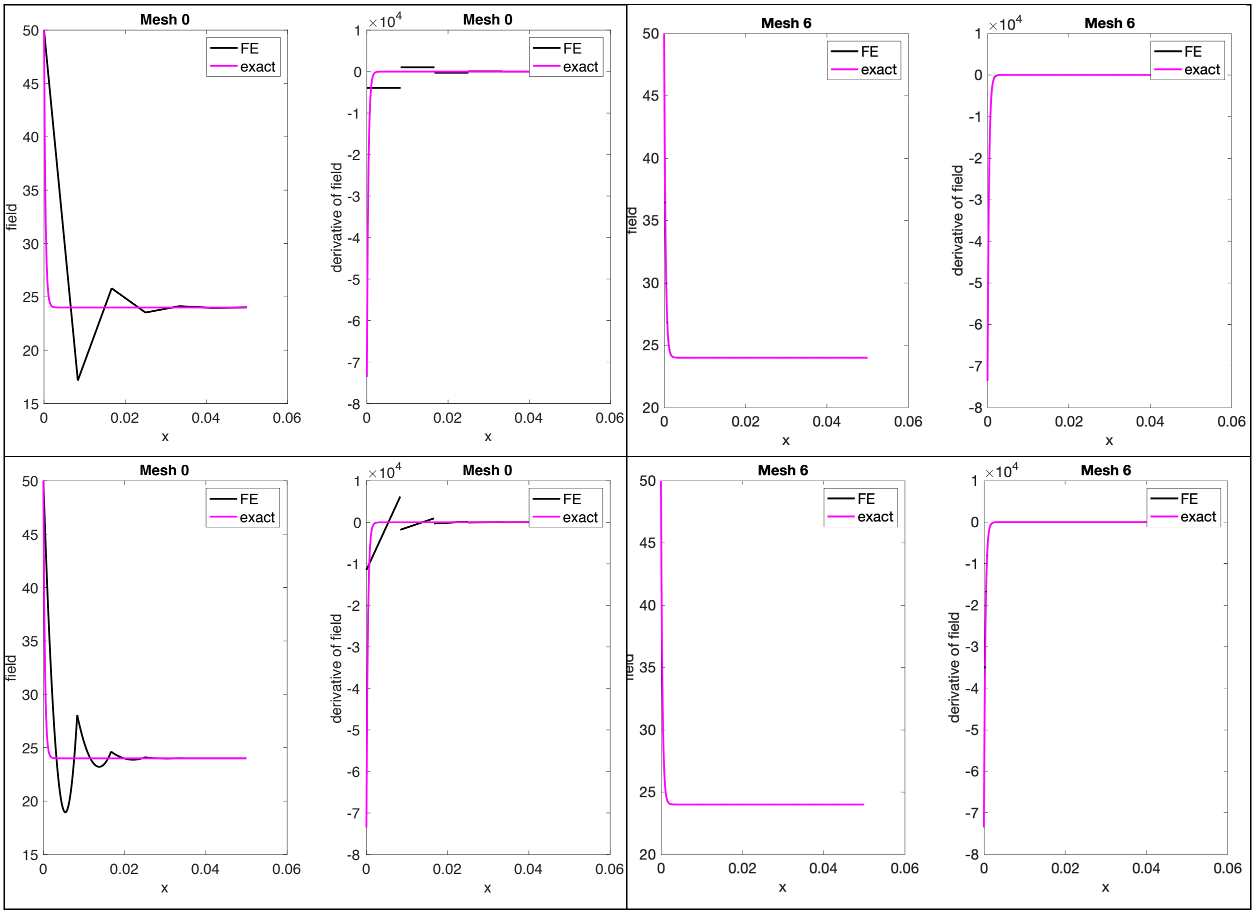


Figure 2.4: Model II Results ($p = 1$ in top row and $p = 2$ in bottom row).

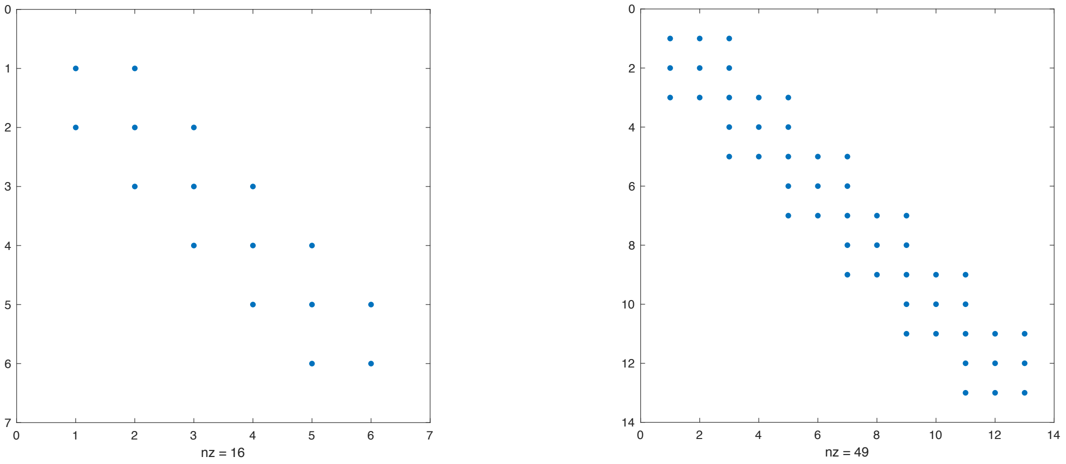


Figure 2.5: Cropped sections of matrix $\underline{A}$ of Model II to visualize sparsity pattern. Note that for $p = 1$ $\underline{A}$ is tridiagonal and for $p = 2$ $\underline{A}$ is pentadiagonal, as expected.

Model I Results

The output of Model I is shown in **Figure 2.6**. The convergence of u_h to u is evident by the improvement of the FE solution from Mesh 0 to Mesh 6. For Model I, $p = 2$ is particularly effective at capturing the derivative solution. The results of Model I provide confidence that `impose_boundary_cond_sver`[9], the code that implements Eq. (35), is working properly since Model I has nonzero boundary parameters γ_2, f_{Γ_1}, and f_{Γ_2}.

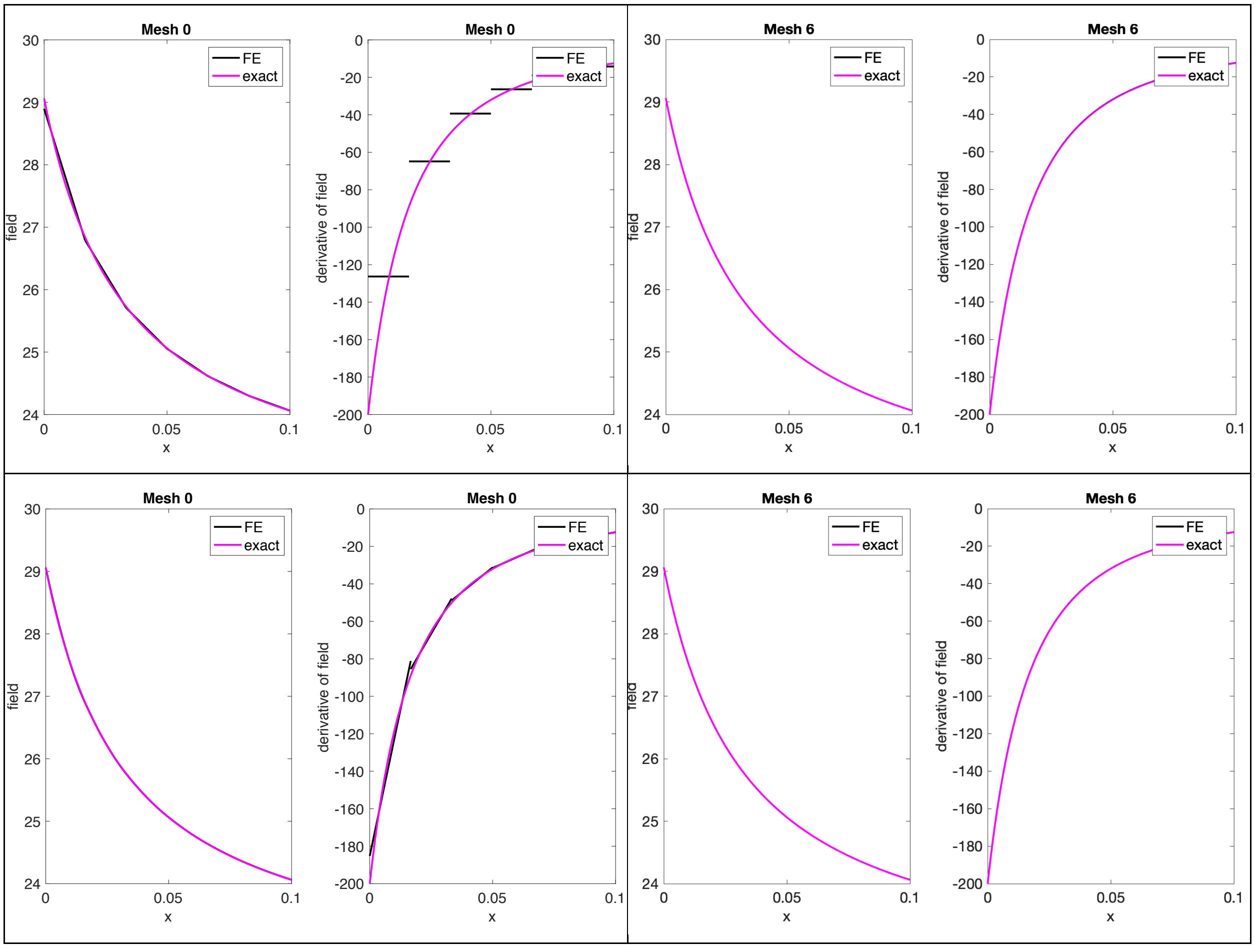


Figure 2.6: Model I results ($p = 1$ in top row and $p = 2$ in bottom row).

Model Mine Results

The output of Model Mine is shown in **Figure 2.7**. Since the exact solution of Model Mine is linear, the FE solution with $p = 1$ captures the exact solution on Mesh 0. An interesting phenomenon occurs with further refinement. Because the finite precision of computer computation can only represent numbers to a finite precision (14 digits), the error in the approximation appears to increase with further refinement because the finite precision is compounded (**Figure 2.8**). This artifact is also evident from the strange-looking plots of the FE derivative in **Figure 2.7**; note the scale of the vertical axis.

Model Mine provides greater confidence of correct FEM implementation than Model I because Model Mine has complete boundary conditions (heat flux and convection) at both ends of the domain; i.e. γ_1, γ_2, f_{Γ_1}, and f_{Γ_2} are nonzero. In contrast, $\gamma_1 = 0$ in Model I, and an implementation error in `impose_boundary_cond_sver` involving γ_1 could go unnoticed when analyzing Model I. Such a bug would be detected when running Model Mine.

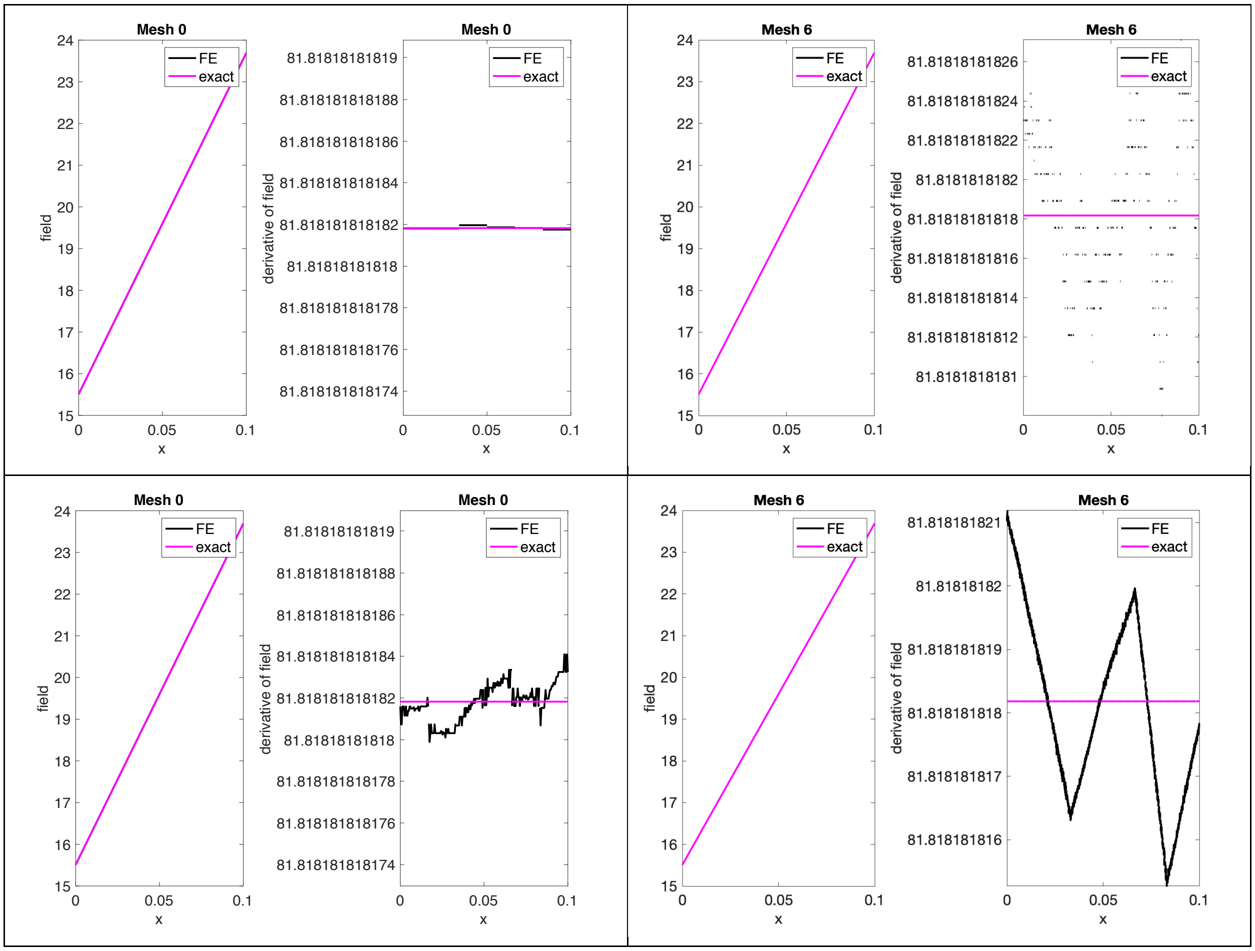


Figure 2.7: Model Mine Results ($p = 1$ in top row and $p = 2$ in bottom row).

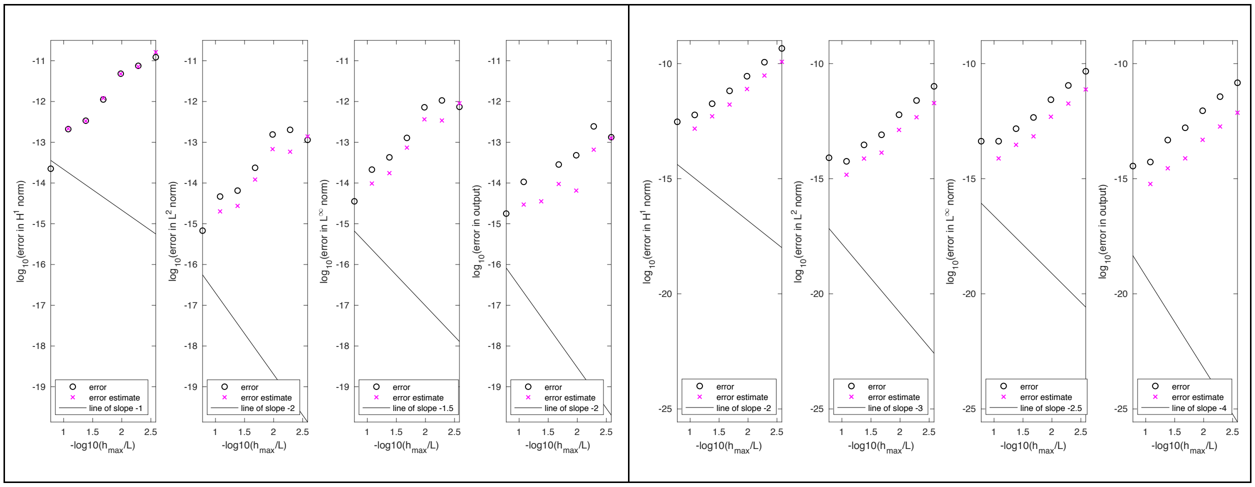


Figure 2.8: Model Mine error estimators ($p = 1$ in left plot; $p = 2$ in right plot).

Model II Error Analysis

The more accurate the FE approximation, the more time the algorithm takes to run. This tradeoff is balanced when FEM refines just until numerical specifications are met. Error in the approximation can be found by comparing the FE solution to the actual solution, which is known as *a priori* error estimation. However, how can the error be found when the exact solution is not known? *A posteriori* error estimators derive estimates of the error using just the FE solution by comparing u_h to $u_{h/2}$, the estimate after a refinement. Common error estimators are summarized in **Table 2.4**. The exponent of the bound, the convergence rate, is the absolute value of the slope on a log(error) vs. log(*L*/*h*) plot. Checking that the error converges at the expected rate is a good verification technique.

Table 2.4: FE error estimators [10]

Description	Definition	A priori error estimator	Bound	A posteriori error estimator
A norm reflecting temperature and its gradient	$\|v\|_{H^1(\Omega)} = \int_0^L \left(\frac{dv}{dx}\right)^2 + \frac{v^2}{L} dx$	$\|u - u_{h/2}\|_{H^1(\Omega)}$	$\sim C_u \left(\frac{h}{2}\right)^p$	$\leq \frac{\|u_{h/2} - u_h\|_{H^1(\Omega)}}{2^p - 1}$
A norm reflecting temperature	$\|v\|_{L^2(\Omega)} = \int_0^L v^2 dx$	$\|u - u_{h/2}\|_{L^2(\Omega)}$	$\sim C_u \left(\frac{h}{2}\right)^{p+1}$	$\leq \frac{\|u_{h/2} - u_h\|_{L^2(\Omega)}}{2^{(p+1)} - 1}$
A norm reflecting maximum temperature over domain	$\|v\|_{L^\infty(\Omega)} = \max\|v\|$	$\|u - u_{h/2}\|_{L^\infty(\Omega)}$	$\sim C_u \left(\frac{h}{2}\right)^{p+1/2}$	$\leq \frac{\|u_{h/2} - u_h\|_{L^\infty(\Omega)}}{2^{(p+1/2)} - 1}$
The problem's output	s	$\|s - s_{h/2}\|$	$\sim C_u \left(\frac{h}{2}\right)^{2p}$	$\leq \frac{\|s_{h/2} - s_h\|}{2^{2p} - 1}$

To realistically assess the accuracy of the FE solution, Model II was studied on a sequence of 9 meshes (8 refinements) for $p = 1$ without reference to the exact solution. Error plots are shown in **Figure 2.9**. What do these plots convey about the minimum necessary refinements to achieve a numerical specification? Two criteria must be met to achieve a desired accuracy most efficiently. The first criterion says that convergence is uncertain until error in the $H^1(\Omega)$ norm *decreases* with further refinement. The second criterion says that the error in the output or a chosen norm must be less than a numerical specification.

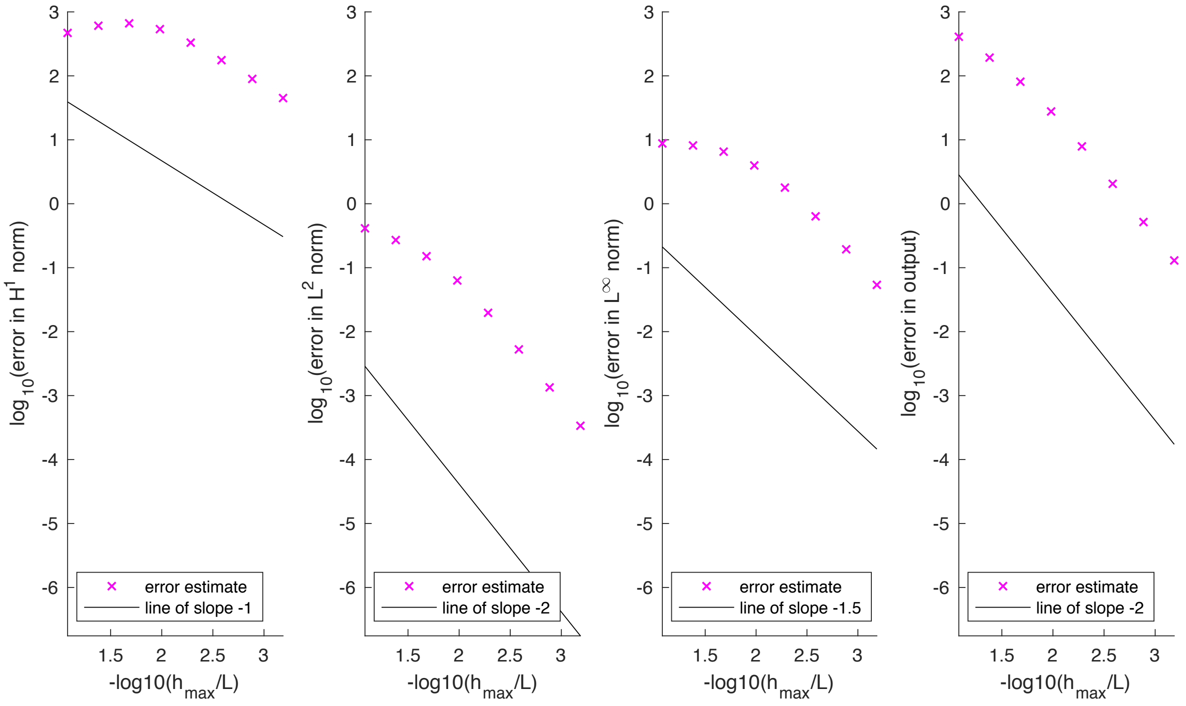


Figure 2.9: Estimated error in Model II ($p = 1$) without reference to the exact error.

For example, say that $||u - u_h||_{L^\infty(\Omega)}$ must be less than 1.00. On a log scale, the error is then less than 0. From the $H^1(\Omega)$ plot in **Figure 2.9**, it is evident that first criterion is not satisfied until Mesh 5 (the fourth point). Then, from the $L^\infty(\Omega)$ plot, the coarsest mesh that satisfies $\log||u - u_h||_{L^\infty(\Omega)} < 0$ can be identified as Mesh 7 (the sixth point). Mesh 7 is finer than Mesh 5, so both criteria are met. On Mesh 7, $\log||u - u_h||_{L^\infty(\Omega)} = -0.1967$. Then the error in the $L^\infty(\Omega)$ norm is $10^{-0.1967} = 0.6358$. In practice the higher order contributions to the error bound should be accounted for by including a safety factor. If SF=2, the upper bound for error in the $L^\infty(\Omega)$ norm is 1.2715, which does not meet specification. Thus, one further refinement is necessary for the prediction to be sufficiently accurate. On Mesh 8, $\log||u - u_h||_{L^\infty(\Omega)} = -0.7126$. Then the upper bound for the error in the $L^\infty(\Omega)$ norm is $2 * 10^{-0.7126} = 0.3876$, which is within specification.

As another example, consider the upper bound for the error in the output on Mesh 5. $\log|s - s_h| = 1.443$ on Mesh 5, so $|s - s_h| = 10^{1.443} = 27.73$. If SF=2, then the error in the output on Mesh 5 has an upper bound of 55.47.

Since the exact solution to Model II is known, the actual errors can be found. As shown in **Figure 2.10,** the *a posteriori* error estimators are quite good, and a safety factor of 2 is not necessary. This is also evident in **Table 2.5**. However, when the actual error is unknown, a finer mesh that may be required by including a safety factor is worth the higher confidence in accuracy.

Table 2.5: Error comparison

	$\|\|u - u_h\|\|_{L^\infty(\Omega)}$ on Mesh 8	$\|s - s_h\|$ on Mesh 5
Error estimate	0.3876	55.47
Actual error	0.0979	31.84

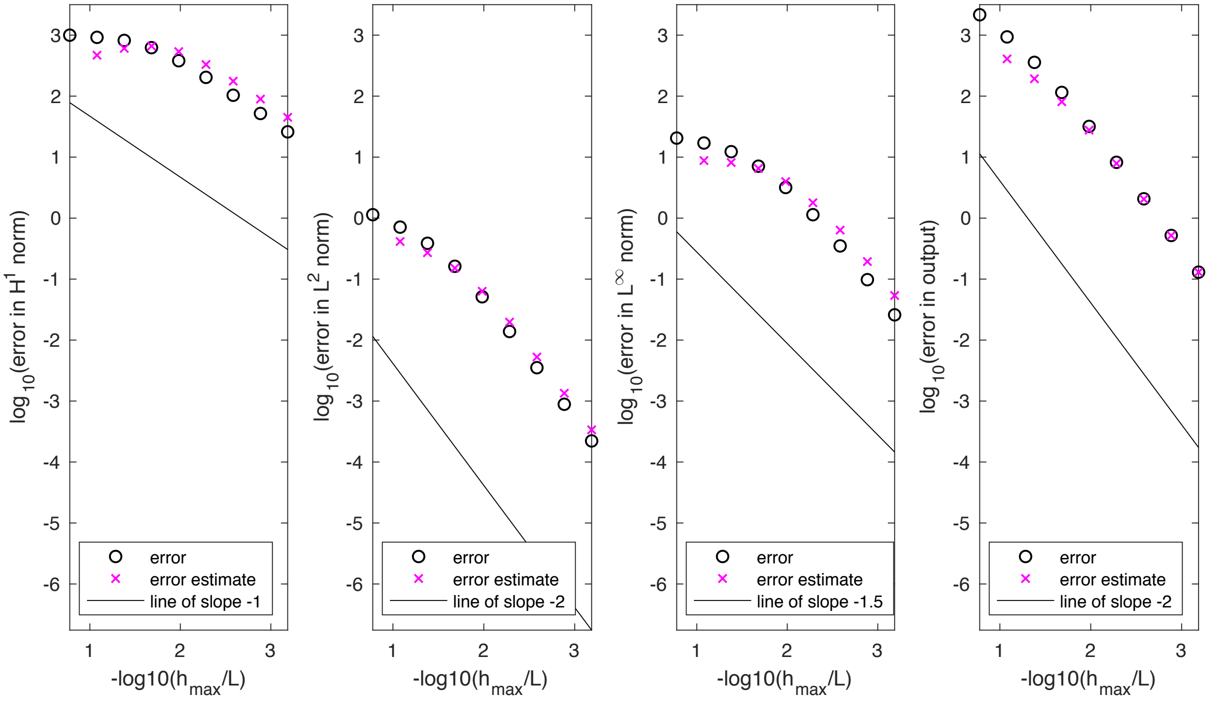


Figure 2.10: Estimated and exact errors in Model II ($p = 1$).

DISCUSSION

Method of Manufactured Solutions

Convergence of u_h to u at the correct rate does not necessarily prove that the code `form_elem_mat_sver` is bug-free for all possible instantiations of $\mu(x)$. For example, $\mu(x)$ may not be a constant, such as when the perimeter of a fin depends on x. Let's define a new model that describes a conical frustum-shaped fin with constant temperature on its left surface and zero heat flux through its right surface, such that $\mu(x) = 2\pi R_0\left(1+\frac{x}{L}\right)\eta_2$, $\kappa(x) = k\pi R_0^2\left(1+\frac{x}{L}\right)^2$, and $\gamma_2 = 0$. Using the method of manufactured solutions [10], say $u = \alpha x^2$. From Eq. (32), then:

$$\begin{aligned} f_\Omega(x) &= -\frac{d}{dx}\left(\kappa\frac{du}{dx}\right) + \mu(u - u_\infty) \\ &= -2\alpha k\pi R_0^2\left(1 + \frac{4x}{L} + \frac{3x^2}{L^2}\right) + 2\pi R_0\left(1 + \frac{x}{L}\right)\eta_2(\alpha x^2 - u_\infty) \end{aligned} \tag{37}$$

Then the model's differential equation and boundary conditions are:

$$\begin{aligned} -\frac{d}{dx}&\left(k\pi R_0^2\left(1+\frac{x}{L}\right)^2\frac{du}{dx}\right) + 2\pi R_0\left(1+\frac{x}{L}\right)\eta_2 u \\ &= -2\alpha k\pi R_0^2\left(1 + \frac{4x}{L} + \frac{3x^2}{L^2}\right) + 2\pi R_0\left(1 + \frac{x}{L}\right)\eta_2(\alpha x^2 - u_\infty) \ \text{ in } \varOmega \end{aligned} \tag{38}$$

$$u = u_{\Gamma_1} \ \text{ on } \varGamma_1 \tag{39}$$

$$-4k\pi R_0^2\frac{du}{dx} = f_{\Gamma_2} \text{ on } \varGamma_2 \tag{40}$$

Since u is known, the FE solution to this model can be validated, providing additional confidence that `form_elem_mat_sver` is correctly implemented.

Model X

In a theoretical Model X the exact solution is not known. Running the FE code, it is observed that for sufficiently small h the extrapolation error estimators converge at the anticipated rates in all norms. Can it be concluded that u_h converges to the exact solution u of Model X? No, it cannot, due to the possibility of implementation error.

There are three potential sources of error when running FEM. One type of error comes from the model itself. Is the mathematical model sufficiently detailed to capture the physics of the system? Another type of error comes from numerical specification. Are the FEM parameters, such as mesh size and basis polynomial order, tuned to meet the required specification? Numerical specification was explored in *Model II Error Analysis*. Lastly, there may be implementation error. *Model Mine Results* demonstrated that there is inherent implementation error due to the finite precision of computer computation. However, implementation error can be more consequential when the code is not implemented correctly. There could be typos in imputing the model's, such as omitting a negative sign or mistaking a decimal place, or the code could run the wrong model. For example, `run_uniform_refinement`[11] features four models. It could be easy to confuse the results of one model for another without attention what section the code is running or an intuition of the model's solution. Model X could be prone to these human implementation errors.

Chapter 3: Time Dependent Finite Element Method for 1D 2nd-Order BVPs

ABSTRACT

In Chapter 3, the finite difference tool is added the finite element method to study time-dependent problems. A simple model of a semi-infinite fin is first analyzed to provide confidence that the method is implemented correctly. Then a more complicated model describing cooking a burger is analyzed. A recipe for "Tender, Juicy Grilled Burgers" from *Cook's Illustrated* is implemented to assess the predictability of the model.

INTRODUCTION

Model Semiinf_plus describes the transfer of heat through a right-cylinder fin at an initial uniform temperature and experiencing convection on its left surface and zero heat transfer through its right surface (**Figure 3.1**).

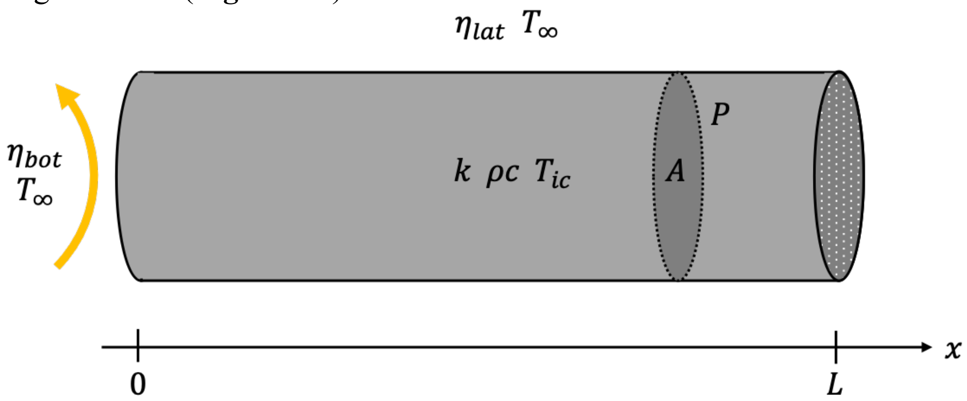


Figure 3.1: Fin geometry of Model Semiinf_plus.

The differential equation, boundary conditions, and initial condition are given by [12]:

$$\rho c A \frac{\partial T}{\partial t} = kA \frac{\partial^2 T}{\partial x^2} + \eta_{lat} P (T - T_\infty) \qquad 0 \le x \le L, 0 < t \le t_f \qquad (41)$$

$$kA \frac{\partial T}{\partial x} = \eta_{bot} \mathrm{A} (T - T_\infty) \qquad x = 0, 0 < t \le t_f \qquad (42)$$

$$-kA \frac{\partial T}{\partial x} = 0 \qquad x = L, 0 < t \le t_f \qquad (43)$$

$$T = T_{ic} \qquad 0 \le x \le L, t = 0 \qquad (44)$$

where:

L is the length of the body [m]
A is the cross sectional area of the body [m^2]
P is the cross sectional perimeter of the body [m]
ρc is the volumetric specific heat [J/m^3 °C]
k is the thermal conductivity of the body [W/m °C]
η_{lat} is the heat transfer coefficient between the lateral surface and the air [W/m^2 °C]
η_{bot} is the heat transfer coefficient between the left surface and the air [W/m^2 °C]

T_∞ is the ambient temperature [°C]
$T(x, t)$ is the spatial and temporal temperature profile of the body [°C]

Model Burger describes cooking a burger on a skillet (**Figure 3.2**). Cooking the burger involves three stages: Stage I (pre-flip) from $0 \leq t \leq t^I$, during which side α cooks; Stage II (post-flip) from $t^I \leq t \leq t^{II}$, during which side β cooks; and Stage III (repose) from $t^{II} \leq t \leq t^{III}$, during which the burger cools on a rack.

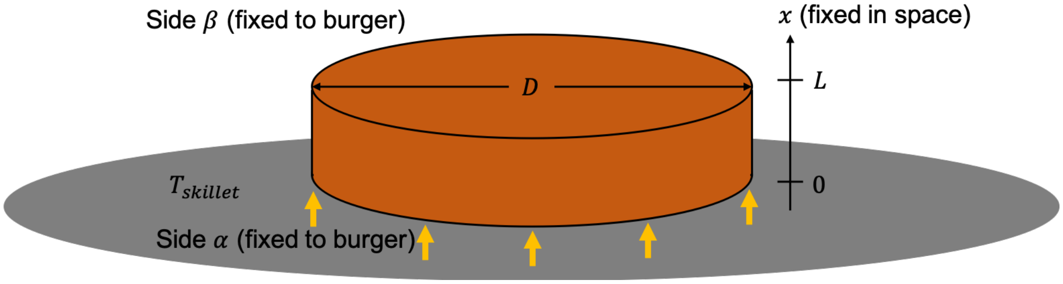

Side β (fixed to burger)
x (fixed in space)
D
L
0
$T_{skillet}$
Side α (fixed to burger)

Figure 3.2: Geometry of Model Burger

The differential equation, boundary conditions, and initial condition of the burger problem for the three stages are summarized in **Table 3.1**.

Table 3.1: Model Burger governing equations

Stage I ($0 \leq t \leq t^I$)	Stage II ($t^I \leq t \leq t^{II}$)	Stage III ($t^{II} \leq t \leq t^{III}$)	On domain
$\rho c A \frac{\partial T^I}{\partial t} = kA \frac{\partial^2 T^I}{\partial x^2} + \eta_{lat}^I P(T^I - T_\infty)$	$\rho c A \frac{\partial T^{II}}{\partial t} = kA \frac{\partial^2 T^{II}}{\partial x^2} + \eta_{lat}^{II} P(T^{II} - T_\infty)$	$\rho c A \frac{\partial T^{III}}{\partial t} = kA \frac{\partial^2 T^{III}}{\partial x^2} + \eta_{lat}^{III} P(T^{III} - T_\infty)$	$0 < x < L$
$kA \frac{\partial T^I}{\partial x} = \eta_{bot}^I A(T^I - T_{skillet})$	$kA \frac{\partial T^{II}}{\partial x} = \eta_{bot}^{II} A(T^{II} - T_{skillet})$	$kA \frac{\partial T^{III}}{\partial x} = \eta_{bot}^{III} A(T^{III} - T_\infty)$	$x = 0$
$-kA \frac{\partial T^I}{\partial x} = \eta_{top}^I A(T^I - T_\infty)$	$-kA \frac{\partial T^{II}}{\partial x} = \eta_{top}^{II} A(T^{II} - T_\infty)$	$-kA \frac{\partial T^{III}}{\partial x} = \eta_{top}^{III} A(T^{III} - T_\infty)$	$x = L$
$T^I(x, 0) = T_{ic}$	$T^{II}(x, t^I) = T^I(L - x, t^I)$	$T^{III}(x, t^{II}) = T^{II}(x, t^{II})$	$0 < x < L$

where:

L is the burger thickness [m]
A is the burger cross sectional area [m^2]
P is the burger cross sectional perimeter [m]
ρc is the burger volumetric specific heat [J/m^3 °C]
k is the burger thermal conductivity [W/m °C]
η_{lat} is the heat transfer coefficient on the lateral surface [W/m^2 °C]
η_{top} is the heat transfer coefficient on the top surface [W/m^2 °C]
η_{bot} is the heat transfer coefficient on the bottom surface [W/m^2 °C]
T_∞ is the ambient temperature [°C]
$T_{skillet}$ is the skillet temperature [°C]
$T(x, t)$ is the spatial and temporal temperature profile of the burger [°C]

BACKGROUND

Formulation of the Heat Equation

The solution to the heat equation is a temperature profile, u, that depends on time and space. The spatial component is approximated by the finite element (FE) method, which divides the domain into a mesh of n_{el} elements of length h. The temporal component is approximated by the finite difference (FD) method, which discretizes time into n_{tsteps} steps of duration Δt. Just as the FE method can implement different basis functions, most notably piecewise linear ($p = 1$) and piecewise quadratic ($p = 2$), the FD method invokes different schemes (θ) to approximate time-dependent derivatives. Three common schemes are summarized in **Table 3.2**. This analysis utilizes the Euler backward and Crank-Nicolson schemes because they are convergent and stable. They are also less numerically costly than Euler forward, which requires much smaller time steps to reach a desired accuracy.

Table 3.2: Finite difference schemes

θ	Name	Rule implemented	Characteristics in time
0	Euler forward	Rectangle right	1st order, explicit, unstable
1/2	Crank-Nicolson	Trapezoidal	2nd order, implicit, stable
1	Euler backward	Rectangle left	1st order, implicit, stable

The FD-FE method approximates the solution to the heat equation in both time and space. The heat equation and N/R-N/R boundary conditions, applicable to Models Semiinf_Plus and Burger, take the form [13]:

$$-\frac{\partial}{\partial x}\left(\kappa(x)\frac{\partial u}{\partial x}\right)+\mu(x)u = f_{\Omega}-\rho(x)\dot{u} \text{ in } \mathit{\Omega}, 0 < t \le t_f \tag{45}$$

$$\kappa\frac{\partial u}{\partial x} = \gamma_1 u - f_{\Gamma_1} \text{ on } \mathit{\Gamma}_1, 0 < t \le t_f \tag{46}$$

$$-\kappa\frac{\partial u}{\partial x} = \gamma_2 u - f_{\Gamma_2} \text{ on } \mathit{\Gamma}_2, 0 < t \le t_f \tag{47}$$

$$u = u_{ic}(x) \text{ in } \mathit{\Omega}, t = 0 \tag{48}$$

The FD-FE solution to the heat equation is:

$$u_{h,\Delta t}^{k}(x), 1 \le k \le n_{tsteps} \tag{49}$$

$u_{h,\Delta t}^{k}(x)$, a numerical approximation to the true solution $u(x,t)$, is found by solving:

$$\begin{gathered}\underline{M}^{inertia}\frac{\underline{u}_{h,\Delta t}^{k}-\underline{u}_{h,\Delta t}^{k-1}}{\Delta t}+\underline{A}\left(\theta\underline{u}_{h,\Delta t}^{k}-(1-\theta)\underline{u}_{h,\Delta t}^{k-1}\right)=\underline{F}\;, 2 \le k \le n_{tsteps}\\ \underline{u}_{h,\Delta t}^{k} = \underline{I_h u_{ic}}\;, k = 1\end{gathered} \tag{50}$$

where the elemental matrices are defined by the parameters of the boundary value problem:

$$M_{ij}^{inertia} = \int_0^L \rho(x)\varphi_i\varphi_j dx \quad 1 \le i,j \le n \tag{51}$$

$$A_{ij} = \int_0^L \kappa(x)\frac{\partial\varphi_i}{\partial x}\frac{\partial\varphi_j}{\partial x}+\mu(x)\varphi_i\varphi_j dx+\gamma_1\varphi_i(0)\varphi_j(0)+\gamma_2\varphi_i(L)\varphi_j(L) \quad 1 \le i,j \le n \tag{52}$$

$$F_i = \int_0^L f_\Omega(x)\varphi_i dx + f_{\Gamma_1}\varphi_i(0) + f_{\Gamma_2}\varphi_i(L) \quad 1 \le i \le n \tag{53}$$

Implementation of Models

To solve the Semiinf_plus and Burger problems, the models' boundary value equations are mapped to Eq.s (45)-(47), which is summarized in **Table 3.3**. The code `run_uniform_refinement` solves Eq. (50) for $u_{h,\Delta t}^k(x)$.

Table 3.3: Mapping models to generalized parameters

Parameter	Heat transfer meaning	Value in Model Semiinf_plus	Value in Model Burger pre-flip (*I*), post-flip (*II*), repose (*III*)
$\rho(x)$	Density times specific heat times area	1	$\rho c A$
$\kappa(x)$	Conductivity times area	1	kA
$\mu(x)$	Heat transfer coefficient times perimeter	1	$\eta_{lat}^I P$ $\eta_{lat}^{II} P$ $\eta_{lat}^{III} P$
$f_\Omega(\mathrm{x})$	Generalized heat source	0	$\eta_{lat}^I P T_\infty$ $\eta_{lat}^{II} P T_\infty$ $\eta_{lat}^{III} P T_\infty$
γ_1	Generalized heat transfer coefficient	1	$\eta_{bot}^I A$ $\eta_{bot}^{II} A$ $\eta_{bot}^{III} A$
γ_2	Generalized heat transfer coefficient	0	$\eta_{top}^I A$ $\eta_{top}^{II} A$ $\eta_{top}^{III} A$
f_{Γ_1}	Generalized heat flux	0	$\eta_{bot}^I A T_{skillet}$ $\eta_{bot}^{II} A T_{skillet}$ $\eta_{bot}^{III} A T_\infty$
f_{Γ_2}	Generalized heat flux	0	$\eta_{top}^I A T_\infty$ $\eta_{top}^{II} A T_\infty$ $\eta_{top}^{III} A T_\infty$

RESULTS

Verification of Model Semiinf_plus Implementation

The output of Model Semiinf_plus is shown in **Figure 3.3**. The convergence of the FE solution, $u_{h,\Delta t}^{k}(x)$, to the exact solution, $u(x,t^{k})$, is evident by the improvement of fit from Mesh 0 to Mesh 3. Observe that the $p = 2$ scheme does a better job capturing the solution at fewer refinements than the $p = 1$ scheme, and it is much more effective at capturing the derivative solution.

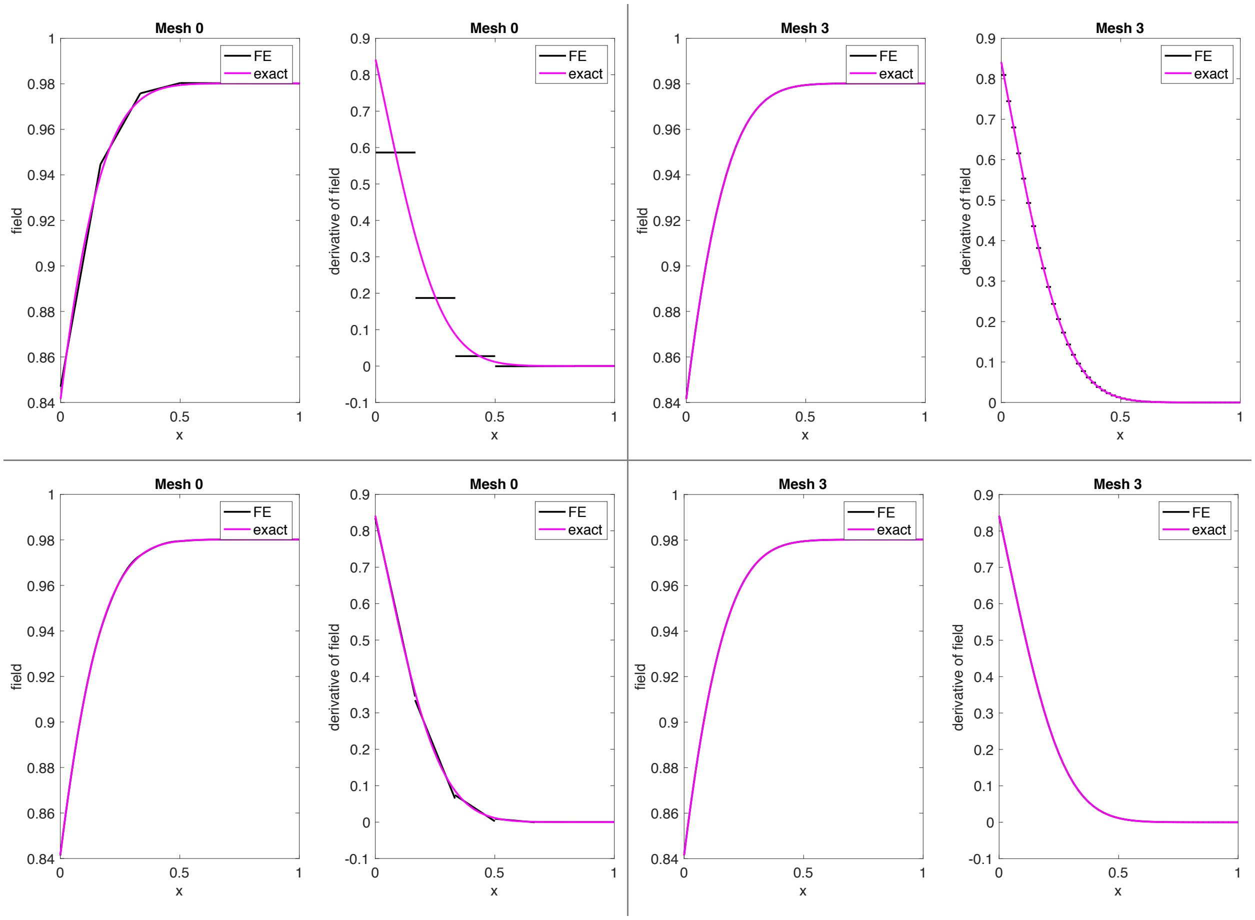


Figure 3.3: Model Semiinf_plus results (Top row: $p = 1$ and $\theta = 1$; Bottom row: $p = 2$ and $\theta = 1/2$).

With each refinement, the FD-FE method divides h by a factor of 2 and Δt by a factor of σ, which is chosen such that FD and FE approximations converge at the same rate [14]. The convergence rate for $L^2(\Omega)$ is $\sim C_{u,Q}2^{-rl}$, where $C_{u,Q}$ is the sum of the spatial and temporal convergence constants, l is the refinement level, and r is the convergence rate. For the $L^2(\Omega)$ norm, $r = p + 1$. As seen by the negative slope in the error estimators plotted in **Figure 3.4**, the solution converges at the expected rates, giving confidence that the code `solve_fld_output_t_sver`[15] is implemented correctly.

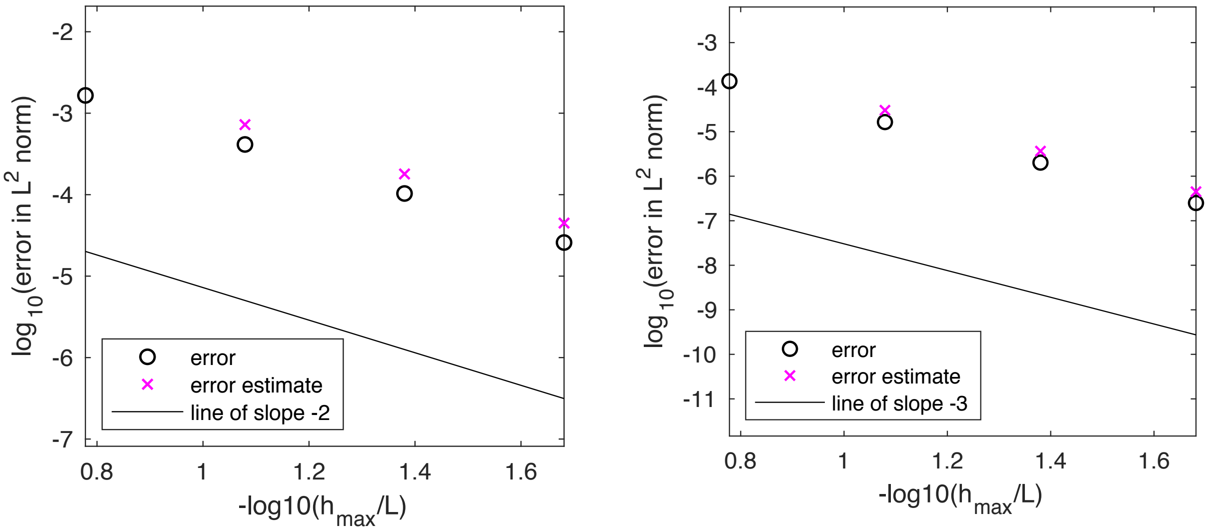


Figure 3.4: Model Semiinf_plus $L^2(\Omega)$ norm error estimators (Left: $p = 1$ and $\theta = 1$; Right $p = 2$ and $\theta = 1/2$).

Verification of Model Burger Implementation

Key results of Model Burger are shown in **Figure 3.5**. The content of these plots is explored in the Discussion. For now, the interest is verification of correct implementation, which is done by direct comparison. The top row of **Figure 3.5** shows results obtained from the code `run_uniform_refinement_burger_sver`[16], where the function `make_probdef_burger` was modified with the burger parameters shown in **Table 3.3**. In comparison, the bottom row of **Figure 3.5** shows results reported by AT Patera [17]. Using MATLAB's interactive data cursor, precise comparisons can be made on selected points. This test is a basic demonstration of the common technique of verifying the implementation of a code, such as that used for this project, by comparing its results to the results of rigorously developed commercial software.

Verification of Model Burger Numerical Specifications

Error estimators of the output—the burger temperature T^I at the skillet side at time t^I just before the flip—are shown in **Figure 3.6**. For the output error, $r = 2p$. As an exercise, say the error is specified to be less than 0.001 °C (less than -3 on a logarithmic scale). A safety factor of 1 is sufficient for a burger. Then, for $p = 1$ and $\theta = 1$, the coarsest mesh that meets the prescribed tolerance is Mesh 6 (fifth refinement) with coordinates (2.283, -3.443). The tolerance is thus $10^{-3.443} = 0.00036$ °C. Mesh 6 is circled in red on the left plot of **Figure 3.6**. For $p = 2$ and $\theta = 1/2$, the coarsest mesh that meets the prescribed tolerance appears to be Mesh 2 (first refinement) with coordinates (1.079, -3.273). However, there is a special caveat for this particular scheme. Note, the slope of the data in the right plot is -3 instead of -4. The *output error of a FD-FE second order scheme* convergences at a rate of $\sim C_{u,Q} 2^{-(r-1)l}$. Thus, it is good practice to create an upper bound on the error by scaling by a factor of 2, which shifts the data points up by $\log(2) = 0.301$. Then, $-3.273 + 0.301 = -2.972 > -3$, and Mesh 2 does not achieve specification. Conversely, Mesh 3 (second refinement) with coordinates (1.38, -4.175) does meet specification because $-4.175 + 0.301 = -3.874 < -3$. The tolerance is $10^{-3.874} = 0.00013$ °C. Mesh 3 is circled in red on the right plot of **Figure 3.6**.

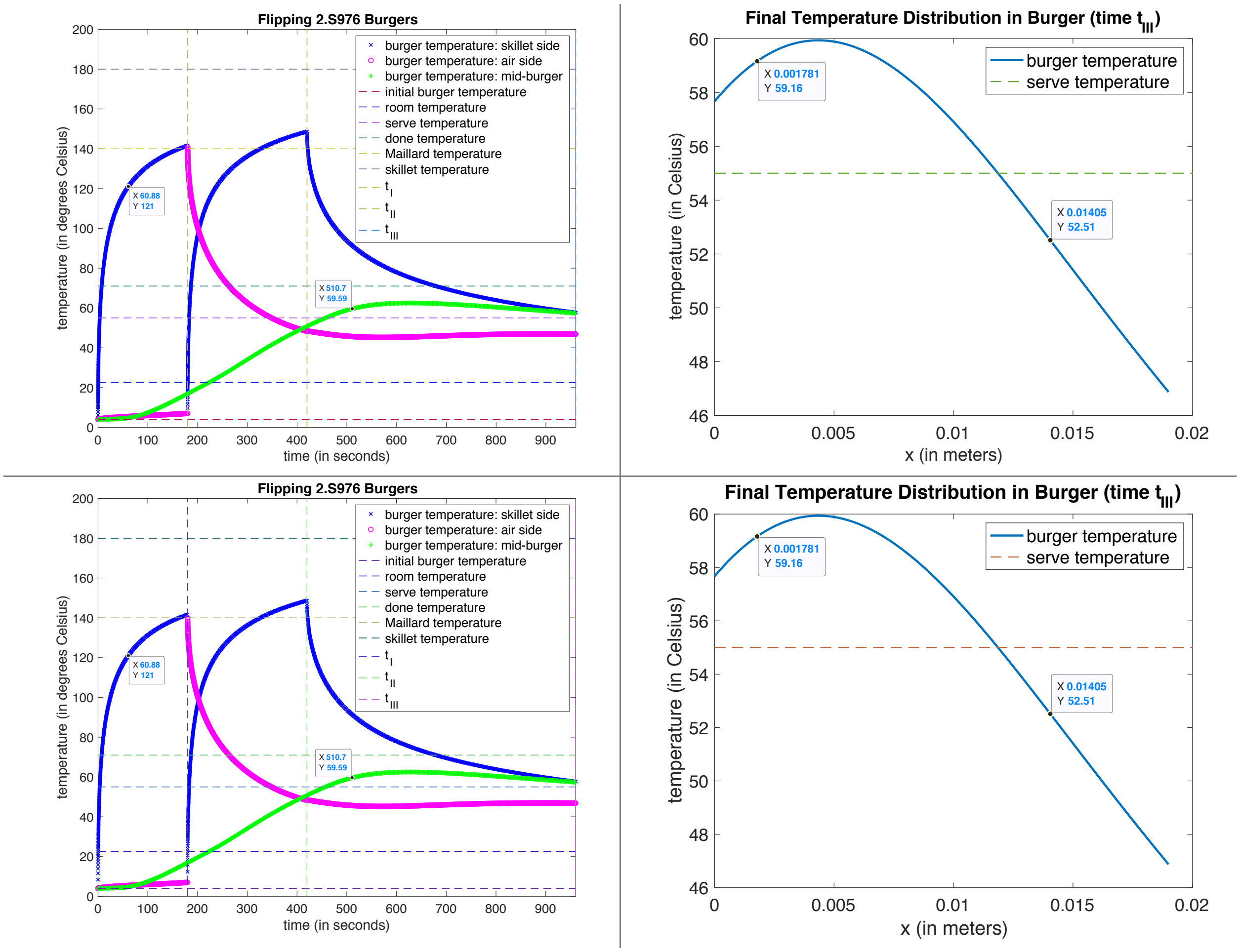


Figure 3.5: Verification of Model Burger implementation using the data cursor. Top row: Results obtained from FEM. Bottom row: Results reported by AT Patera.

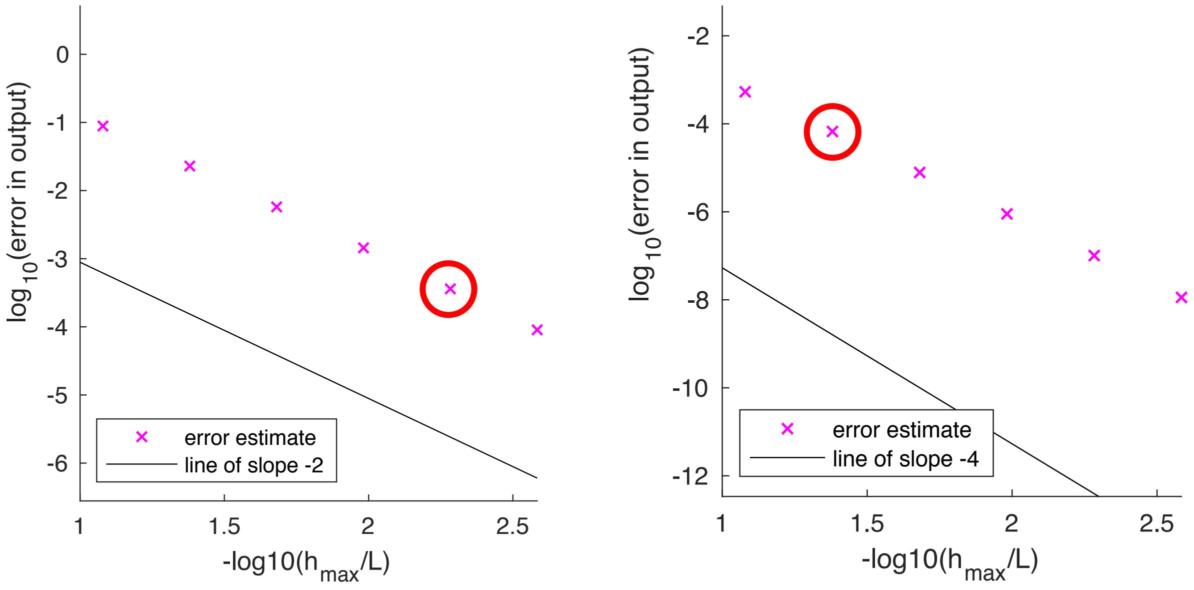


Figure 3.6: Model Burger output error estimators (Left: $p = 1$ and $\theta = 1$; Right $p = 2$ and $\theta = 1/2$).

In truth, the Burger mathematical model are input parameters are not remotely accurate to the prescribed tolerance, nor does cooking a burger require such high precision. However, the (unreasonable) tolerance of 0.001 °C illustrates the advantage of a higher order methods; five refinements are required for $p = 1$, while only two are required for $p = 2$. The computational time to implement the FD-FE method at refinement level l is:

$$T_{comp} \propto n_{el}^{(l)} * n_{tsteps}^{(l)} \tag{54}$$

$$n_{el}^{(l)} = n_{el}^{(0)} * 2^l \tag{55}$$

$$n_{tsteps}^{(l)} = n_{tsteps}^{(0)} * \sigma^l \tag{56}$$

where $n_{el}^{(0)}$ and $n_{tsteps}^{(0)}$ are the initial number of elements and timesteps before refinement, respectively. Eq. (55) applies to uniform refinement that divides the elements in half. The factor σ in Eq. (56) is chosen based on p and θ (see **Table 3.4**).

Table 3.4: Choice of σ

	$p = 1$	$p = 2$
$\theta = 1$	$\sigma = 4$	$\sigma = 8$
$\theta = 1/2$	$\sigma = 2$	$\sigma = 2\sqrt{2}$

In Model Burger, $n_{el}^{(0)} = 6$ and $n_{tsteps}^{(0)} = 20$. For $p = 1$ and $\theta = 1$, specification is reached in five refinements, so $T_{comp} \propto 6 * 2^5 * 20 * 4^5 = 3{,}932{,}160$. For $p = 2$ and $\theta = 1/2$, specification is reached in two refinements, so $T_{comp} \propto 6 * 2^2 * 20 * \left(2\sqrt{2}\right)^2 * 2 = 7{,}680$, where the final factor of 2 accounts for the fact that the operation count to solve a penta-diagonal system ($p = 2$) is twice that of a tri-diagonal system ($p = 1$). The ratio of computational times is $\frac{3{,}932{,}160}{7{,}680} = 512$. Thus, the second order scheme is 512 times more efficient than the first order scheme under the specification that the output error is less than 0.001 °C.

DISCUSSION

To assess whether Model Burger is sufficiently accurate to provide design guidance, a recipe from *Cook's Illustrated*, a magazine noted for its detailed instructions and extensively-tested recipes, was implemented. "Tender, Juicy Grilled Burgers" [18] calls for freezing (T_{ic} = –18°C) a 1.9 cm thick, 11.42 cm diameter, patty. Cook the patty on a gas grill on high (T_{grill} = 205 °C) for 4 to 7 minutes (240 sec ≤ $t_{pre-flip}$ ≤ 420 sec). Flip and grill for another 4 to 7 minutes (240 sec ≤ $t_{post-flip}$ ≤ 420 sec) until the meat reads 130 °F (T_{done} = 54 °C). Let rest on a plate for 5 minutes (t_{repose} = 300 sec) before serving. The FDA recommends serving hot foods at 140 °F (T_{serve} = 60 °C). The meat's flavor is greatly enhanced when the Maillard reaction temperature ($T_{Maillard}$ = 140 °C) is reached on the surfaces of the patty. Room temperature is taken to be T_∞ = 23°C. Thermal properties are [5]: $\eta_{oil\ layer}$ = 220 W/m °C, k_{beef} = 0.38 W/m °C, ρ_{beef} = 1030 kg/m^3, and cp = 3684 J/kg °C. These parameters are entered in the FD-FE code `run_uniform_refinement_burger_recipe`. Results are shown in **Figure 3.7** and **Figure 3.8** for the minimum and maximum cooking times recommended by recipe, respectively.

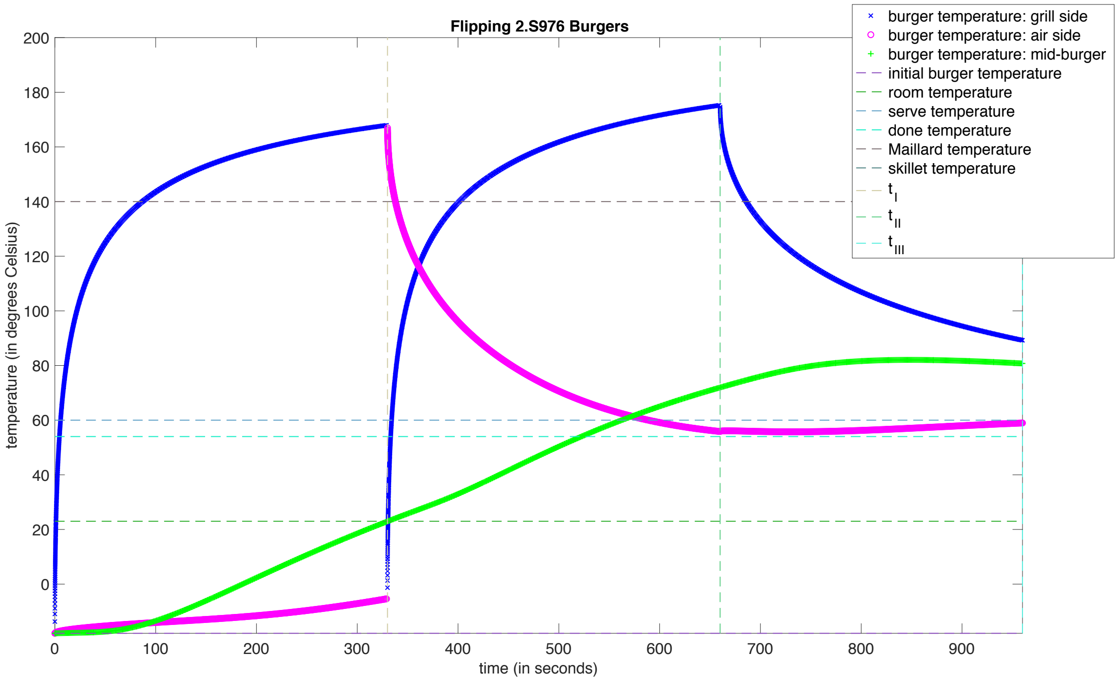


Figure 3.7: Burger temperatures for $t_{pre-flip}$ = 330 sec and $t_{post-flip}$ = 330 sec (minimum cooking time recommended by recipe). $p = 1$ and $\theta = 1$.

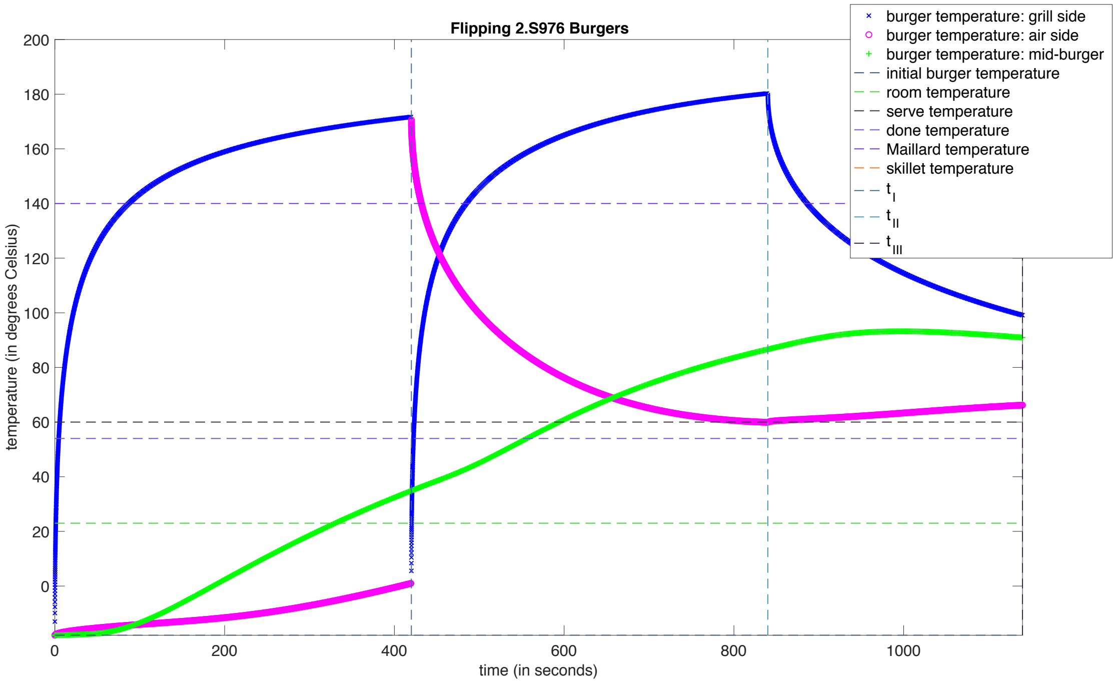


Figure 3.8: Burger temperatures for $t_{pre-flip}$ = 420 sec and $t_{post-flip}$ = 420 sec (maximum cooking time recommended by recipe). $p = 1$ and $\theta = 1$.

Using the data cursor, no meaningful difference in the temperature profile is found between the results of first order and second order schemes. In both the maximum cooking and minimum cooking versions of the recipe, the FD-FE method predicts that the sides of the burger surpass the Maillard onset temperature, which is good for taste, and the final mid-burger temperature is above the serving temperature, which is good for health. However, the cooked burger is *too hot*. The final mid-burger temperature is 81 °C with minimum recommended cooking, while the final mid-burger temperature is 91 °C with maximum recommended cooking. In comparison, hot beverages are generally served between 70 °C to 85 °C, but need time to cool to avoid burning the consumer. According to the model, there are several ways the recipe could be *improved* to reach T_{serve} = 60 °C. Keeping the pre-flip and post-flip cooking times at the minimum recommended by the recipe, the repose time could be increased until the mid-burger temperature cools to 60 °C. This is shown in **Figure 3.9**. The repose time is about 1,280 seconds! No one wants to wait 20 minutes for a burger to cool.

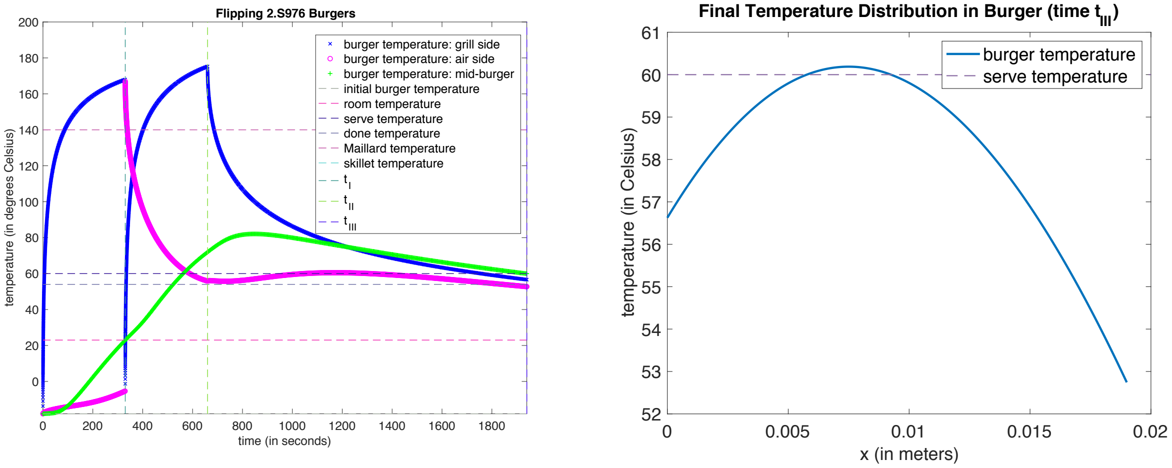


Figure 3.9: Cooling the burger to serving temperature by increasing the repose time.

Another way to make the burger more consumption-friendly is to lower the grill temperature (**Figure 3.10**) or decrease the cooking time (**Figure 3.11**).

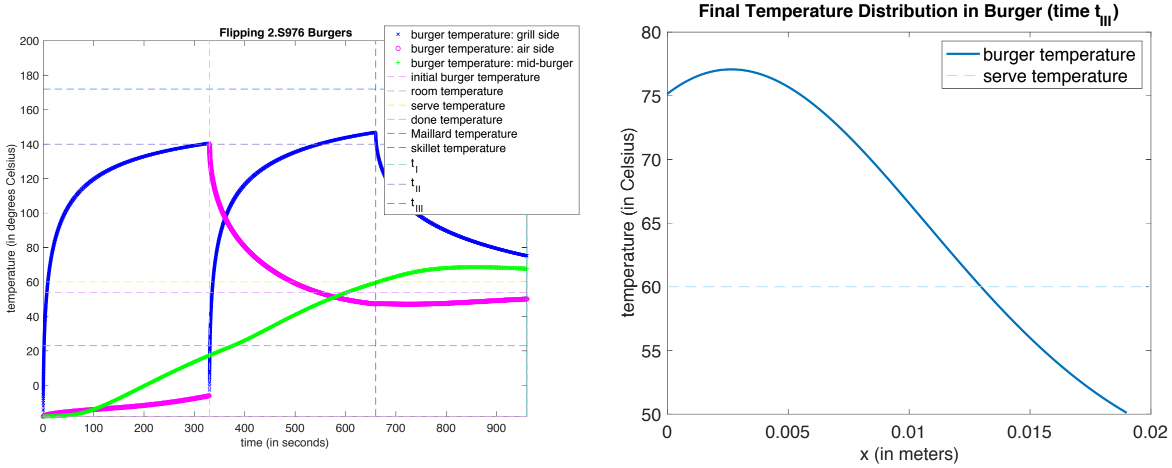


Figure 3.10: A better modeled burger by cooking at a lower temperature (T_{grill} = 172 °C).

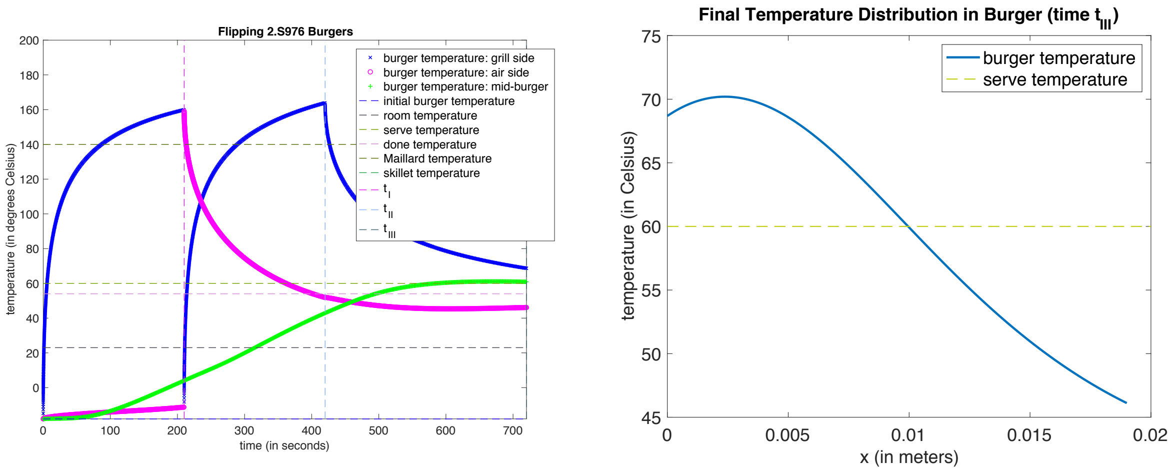


Figure 3.11: A better modeled burger by decreasing the cooking times ($t_{pre-flip}$ = 250 sec, $t_{post-flip}$ = 250 sec, t_{repose} = 300 sec).

Of course, the original recipe should not be discarded without following it first. While the FD-FE numerical specification many introduce small errors, most of the inaccuracies in the burger problem come from the mathematical model. Some of the limiting assumptions are [2]: (i) the oil layer specification is not well justified; (ii) the effects of moisture are neglected, (iii) the temperature of the grill is assumed to be constant; (iv) the thermal properties from literature are not well characterized; (v) natural convection and radiation are linearized; and (vi) the lateral Biot number may not be small enough to justify quasi one-dimensional heat transfer.

Chapter 4: The Finite Element Method for 1D 4th-Order BVPs

ABSTRACT

This chapter implements the finite element method for fourth order beam bending. A new set of basis functions is introduced to model such systems. To assess the accuracy of the FEM code, data from a laboratory experiment studying beam bending are implemented, and FEM results are compared to the study's findings. As another illustrative example, the eigenproblem of a vibrating xylophone bar is solved to optimize the bar's geometry when its first two natural frequencies are specified. Finally, a more realistic physical model of the xylophone bar is discussed.

INTRODUCTION

A xylophone consists of wooden bars suspended from strings (**Figure 4.1**). Each bar is a different length to produce a unique pitch; the shorter the bar, the higher the pitch. Material is removed from the underside of the bars for fine tuning. When a bar is struck by a mallet, it vibrates according to its natural modes. The bar is tuned to the mode with the lowest frequency, called the fundamental frequency. All other natural frequencies are the called the harmonic frequencies. While there are an infinite number of harmonic modes, their amplitudes become negligibly small as their frequencies increase due to damping. The strings are loosely tensioned to isolate each bar's vibration. To hold the bar freely, the strings are placed where the fundamental mode causes zero displacement of the bar.

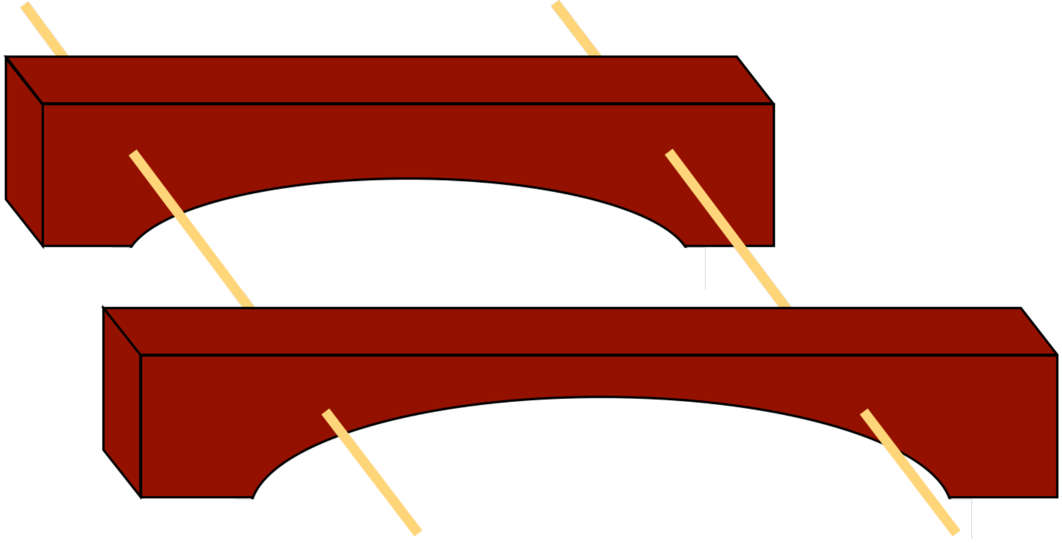


Figure 4.1: Xylophone bars and strings.

This chapter optimizes the geometry of a xylophone bar subject to a desired acoustic performance. As shown in **Figure 4.2**, the modeled bar is rotated 180°, and its height as a function of the coordinate x_d is described by a fourth order polynomial [19]:

$$H_d(x_d) = \begin{cases} H_{max\,d}\left[(1-p_2)\left(\dfrac{L_D/2 - x_d}{L_D/2 - x_d^*}\right)^4 + p_2\right] & \text{for } x_d^* \leq x_d \leq L_d - x_d^* \\ H_{max\,d} & \text{for } 0 \leq x_d < x_d^*,\ \ L_d - x_d^* < x_d \leq L_d \end{cases} \tag{57}$$

where the subscript *d* indicates the dimensional form of quantities, and

x_d^* is the location of the start of the modified section of the bar [m]

L_d is the length of the bar [m]

$H_{max\,d}$ is the unmodified height of the bar [m]

p_2 is a design variable that describes the degree of modification of the bar [/]

Note: when $p_2 = 1$, no material is removed and the height of the bar is constant spanning its length; if $p_2 = 0$, all material would be removed at $L_D/2$. It is recommended that $p_2 \in [0.05, 1.00]$.

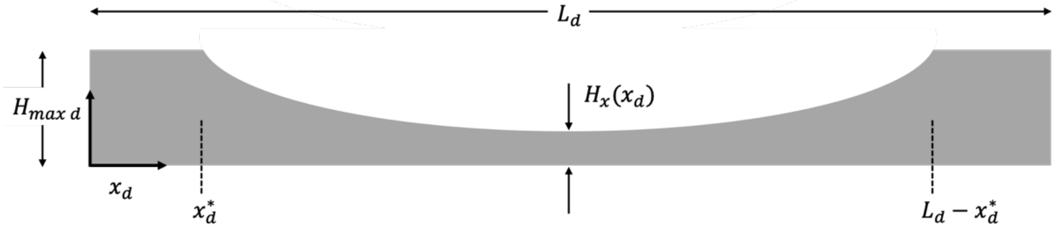


Figure 4.2: Xylophone bar model.

Modeling the vibrations of the xylophone bar is an eigenproblem. The governing differential equation and boundary conditions are:

$$-\frac{d^2}{dx_d^2}\left(\frac{E_d W_d H_d^3(x_d)}{12}\frac{d^2 u_d^{(k)}}{dx_d^2}\right) = \lambda_d^{(k)} \rho_d W_d H_d(x_d) u_d^{(k)} \quad 0 < x_d \leq L_d \tag{58}$$

$$u_{d\,xx}^{(k)}(0) = u_{d\,xxx}^{(k)}(0) = 0 = u_{d\,xx}^{(k)}(L_d) = u_{d\,xxx}^{(k)}(L_d) \tag{59}$$

$$M_d(0) = V_d(0) = 0 = M_d(L_d) = V_d(L_d) \tag{60}$$

where the superscript *k* = 1, 2,…, *n* indicates the *k*th-frequency modeled, and:

W_d is the width of the bar [m]

E_d is the modulus of elasticity [Pa]

ρ_d is the density of the bar [kg/m^3]

$\lambda_d^{(k)}$ is the *k*th eigenvalue [rad^2/s^2]

$u_d^{(k)}(x_d)$ is the deflection of the bar at x_d [m]

$M_d(x_d)$ is the moment about the bar at x_d [Nm]

$V_d(x_d)$ is the shear force on the bar at x_d [N]

Note: an *x*-subscript indicates the *x*th-derivative. The eigenvalues are related to the natural frequencies [Hz] of the bar by:

$$f_d^{(k)} = \sqrt{\lambda_d^{(k)}}/2\pi \tag{61}$$

In solving the eigenproblem, it is convenient to nondimensionalize Eq.s (57)-(61) using the following definitions:

$$x \equiv x_d/L_d \qquad u^{(k)} \equiv u_d^{(k)}/H_{max\,d} \qquad \lambda^{(k)} \equiv \lambda_d^{(k)}\frac{\rho_d L_d^4}{E_d H_{max\,d}^2}$$
$$x^* \equiv x_d^*/L_d \qquad H \equiv H_d/H_{max\,d}$$

Eq.s (57)-(61) become:

$$H(x) = \begin{cases} \left[(1-p_2)\left(\frac{1/2-x}{1/2-x^*}\right)^4 + p_2\right] & \text{for } x^* \le x \le 1-x^* \\ 1 & \text{for } 0 \le x < x^*,\ \ 1-x^* < x \le 1 \end{cases} \tag{62}$$

$$-\frac{d^2}{dx^2}\left(\frac{H^3(x)}{12}\frac{d^2u^{(k)}}{dx^2}\right) = \lambda^{(k)}H(x)u^{(k)} \quad 0 < x \le 1 \tag{63}$$

$$u_{xx}^{(k)}(0) = u_{xxx}^{(k)}(0) = 0 = u_{xx}^{(k)}(1) = u_{xxx}^{(k)}(1) \tag{64}$$

$$M(0) = V(0) = 0 = M(1) = V(1) \tag{65}$$

$$f^{(k)} = \sqrt{\lambda^{(k)}}/2\pi \tag{66}$$

The physical meanings of the natural frequencies are:

$$\begin{array}{ll} f_d^{(1)} = 0 & u_d^{(1)} \propto \text{const (translation)} \\ f_d^{(2)} = 0 & u_d^{(2)} \propto \text{const} * x \ \text{(rotation)} \\ f_d^{(3)} > 0 & \text{Fundamental: playing note (pitch)} \\ \begin{array}{c} f_d^{(k)} > 0 \\ \text{for } k \ge 4 \end{array} & \left.\begin{array}{l} k = 4\text{: first harmonic} \\ k = 5\text{: second harmonic} \\ k = 6\text{: third harmonic} \\ \quad\vdots \end{array}\right\} \text{timbre} \end{array}$$

BACKGROUND

Formulation of the Eigenproblem

The modal deflection of a vibrating bar is represented by a sum of sinusoids [20]:

$$u(x,t) = \sum_{k=1}^{\infty}\left(c_1^{(k)}\cos\omega_n^{(k)}t + c_2^{(k)}\sin\omega_n^{(k)}t\right)u^{(k)}(x) \tag{67}$$

where $c_1^{(k)}$ and $c_2^{(k)}$ are determined from the initial conditions, $\omega_n^{(k)}$ is the natural frequency [rad/s], and $u^{(k)}$ is the mode shape. Eq. (67) is the solution to the eigenproblem:

$$\frac{d^2}{dx^2}\begin{pmatrix}\cos\omega t \\ \sin\omega t\end{pmatrix} = -\omega^2\begin{pmatrix}\cos\omega t \\ \sin\omega t\end{pmatrix} \tag{68}$$

where $\omega^2 = \lambda$, the eigenvalue. Eq. (67) can be applied to generalized problem statement of a beam with free boundary conditions. For $k = 1, 2, \ldots$

$$\frac{d^2}{dx^2}\left(\beta(x)\frac{d^2u^{(k)}}{dx^2}\right) - N_0\frac{d^2u^{(k)}}{dx^2} = \lambda^{(k)}\rho u^{(k)} \quad 0 < x < L \tag{69}$$

$$\begin{matrix}(\beta u_{xx})_x(0) \\ -\beta(0)u_{xx}(0)\end{matrix} = \underline{f_{\Gamma_1}} \quad x = 0 \tag{70}$$

$$\begin{matrix}-(\beta u_{xx})_x(L) \\ \beta(L)u_{xx}(L)\end{matrix} = \underline{f_{\Gamma_2}} \quad x = L \tag{71}$$

The finite element (FE) method approximates the solution by dividing the domain $x \in [0, L_D]$ into a mesh with of n_{el} elements, T^m, separated by nodes, x^i. At each node there are two degrees of freedom: the deflection of the beam ($l = 1$) and the slope ($l = 2$). The FE solution is a linear combination of coefficients $u_{h_{i,l}}$ and basis functions $\Phi_{i,l}$:

$$u_h(x) = \sum_{i=1}^{n_{node}} \sum_{l=1}^{2} u_{h_{i,l}} \Phi_{i,l}(x) \tag{72}$$

Solving the fourth order boundary value problem requires cubic basis functions, which are generated from a Hermite spline, as shown in **Figure 4.3**. Observe that:

- Blue curve: $\Phi_{1,1}(\hat{x}^1) = 1$ and $\Phi_{1,1}(\hat{x}^2) = \Phi'_{1,1}(\hat{x}^1) = \Phi'_{1,1}(\hat{x}^2) = 0$
- Red curve: $\Phi'_{1,2}(\hat{x}^1) = 1$ and $\Phi_{1,2}(\hat{x}^1) = \Phi_{1,2}(\hat{x}^2) = \Phi'_{1,2}(\hat{x}^2) = 0$
- Yellow curve: $\Phi_{2,1}(\hat{x}^2) = 1$ and $\Phi_{2,1}(\hat{x}^1) = \Phi'_{2,1}(\hat{x}^1) = \Phi'_{2,1}(\hat{x}^2) = 0$
- Purple curve: $\Phi'_{2,2}(\hat{x}^2) = 1$ and $\Phi_{2,2}(\hat{x}^1) = \Phi'_{2,2}(\hat{x}^1) = \Phi_{2,2}(\hat{x}^2) = 0$

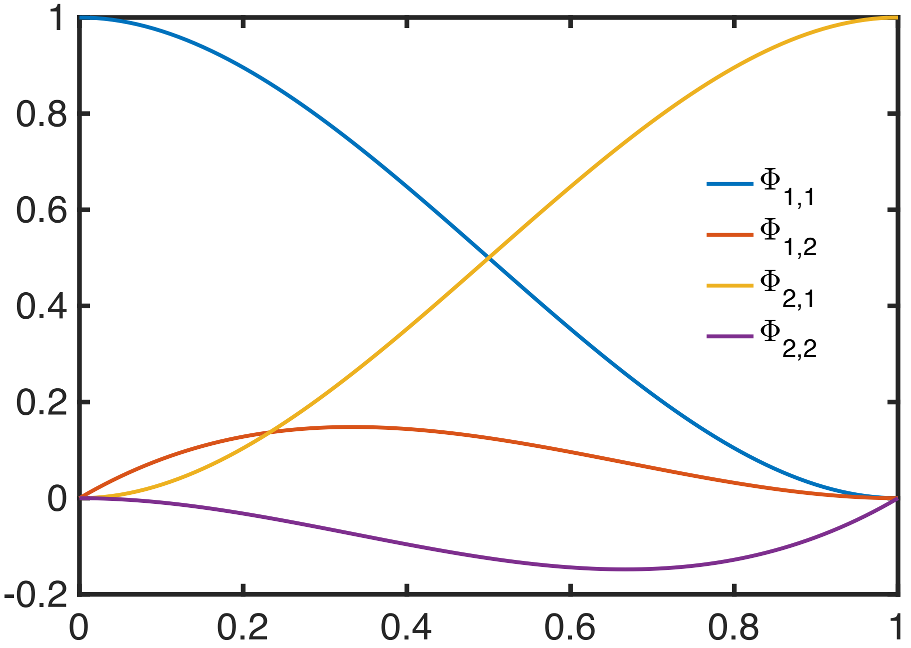


Figure 4.3: Hermitian Basis Functions.

For constructing $u_{h_{i,l}}$ and $\Phi_{i,l}$, it is necessary to map from double indexing to single indexing. The general formulation is [21]:

$$\begin{gathered}\varphi_{[i,l]^{2\to1}} = \Phi_{i,l} \\ u_{h\,[i,l]^{2\to1}} = u_{h\ i,l} \\ [i,l]^{2\to1} = 2(i-1) + l\end{gathered} \tag{73}$$

$[\varphi_1\ \varphi_2\ \varphi_3\ \varphi_4] = [\Phi_{1,1}\ \Phi_{1,2}\ \Phi_{2,1}\ \Phi_{2,2}]$ and $[u_{h\,1}\ u_{h\,2}\ u_{h\,3}\ u_{h\,4}] = [u_{h\,1,1}\ u_{h\,1,2}\ u_{h\,2,1}\ u_{h\,2,2}]$, for instance. Then Eq. (72) is recast as a single sum:

$$u_h(x) = \sum_{j=1}^{2n_{node}} u_{h_j}\varphi_j(x) \tag{74}$$

The coefficients of the FE solution are found by minimizing the energy functional:

$$\min \Pi(u_h) \;\rightarrow\; \underline{A}\,\underline{u}_h^{(k)0} = \underline{F} \tag{75}$$

For the eigenproblem, $\underline{F}$ is recast as the inertia matrix and the eigenvalue:

$$\underline{A}\,\underline{u}_h^{(k)0} = \lambda_h^{(k)}\underline{M}^{inertia}\underline{u}_h^{(k)0} \tag{76}$$

where the elemental matrices are defined by the parameters of the boundary value problem:

$$A_{ij} = \int_0^L \left[\beta(x)\frac{d^2\varphi_i}{dx^2}\frac{d^2\varphi_j}{dx^2} + N_0\frac{d\varphi_i}{dx}\frac{d\varphi_j}{dx}\right]dx \quad 1 \le i,j \le 2n_{node} \tag{77}$$

$$M_{ij}^{inertia} = \int_0^L \rho A_{cs}(x)\varphi_i\varphi_j dx \quad 1 \le i \le 2n_{node} \tag{78}$$

Implementation of Xylophone Model

To implement the vibrating bar problem using the FE method, Eq.s (58)-(60) of the xylophone model are mapped to Eq.s (69)-(71) of the general formulation, as shown in **Table 4.1**.

Table 4.1: Mapping xylophone model to generalized parameters

Parameter	Eigenproblem meaning	Value in Xylophone model (per unit width)
$\beta(x)$	$(EI)_{\text{eff}}(x)$ Effective Young's modulus times area moment of inertia	$\frac{H^3(x)}{12}$
N_0	Axial tension	0
A_{cs}	Cross-sectional area	$H(x)$
$\underline{f_{\Gamma_1}}$	$\begin{bmatrix}-V_{\Gamma_1}\\-M_{\Gamma_1}\end{bmatrix}$ Natural boundary conditions	$\begin{bmatrix}0\\0\end{bmatrix}$
$\underline{f_{\Gamma_2}}$	$\begin{bmatrix}V_{\Gamma_2}\\M_{\Gamma_2}\end{bmatrix}$ Natural boundary conditions	$\begin{bmatrix}0\\0\end{bmatrix}$

There are three objectives in designing the xylophone bar [19]. The first concerns *timbre*, the tone quality. The ratio between the first harmonic frequency and the fundamental frequency is defined as:

$$R \equiv \frac{f_d^{(4)}}{f_d^{(3)}} \tag{79}$$

The natural frequencies depend on x^* and p_2 only. Thus, given x^* and R_{target} (the desired ratio), the objective is to find p_2^{opt}(the optimal material removal parameter). This is done in the code xylo_bar_design3 using a *binary chop* function that refines $R(p_2^{opt})$ until it is within a prescribed tolerance of R_{target}. Common ratio targets that create pleasant timbre are:

$R = 3$ Quint tuning
$R = 4$ Double-octave tuning

The second objective concerns *pitch*, the playing note. Given x^*, p_2^{opt} (from timbre optimization), $H_{max\,d}$, E_d, ρ_d, and $f_{target\,d}^{(3)}$ (the desired fundamental frequency), the objective is to find L_{dh} (the optimal length of the bar) to meet the desired frequency. Using the FEM approximation of the fundamental frequency, $f_h^{(3)} = \text{function}(x^*, p_2^{opt})$, the length is:

$$L_{dh} = \left(\frac{f_h^{(3)}}{f_{target\,d}^{(3)}}\right)^{1/2} \left(\frac{E_d H_{max\,d}^2}{\rho_d}\right)^{1/4} \tag{80}$$

The third objective concerns the *string-hole placement*. Given x^*, p_2^{opt} (from timbre optimization), $H_{max\,d}$, E_d, ρ_d, and L_{dh} (from pitch optimization), the objective is to find $x_d^{hole\,1}$ and $x_d^{hole\,2}$ such that the displacement at these locations is zero:

$$u_d^{(3)}(x_d^{hole\,1}) = u_d^{(3)}(x_d^{hole\,2}) = 0 \tag{81}$$

For each hole, the protocol is:

i. Find the element, m^* that contains zero bar deflection
ii. Find the zero $\hat{x}^{hole} \in [0,1]$ within the m^* element
iii. Scale $\hat{x}^{hole}$ to the dimensional domain

Below is the sample of code from xylo_bar_design3 [22] that locates the holes.

```
%Part (i) find m_stars
m_star = [];                              %create m_star vector
for m = 1:n_el
    if u3(lg2(1,m))*u3(lg2(3,m)) < 0      %the point before a zero times the
                                          point after the zero is negative
        m_star = [m_star,m];              %if true, add element to m_star
    end
end
%Part (ii) find xhatholes
xhathole = zeros(1,length(m_star));       %create xhathole vector
for i = 1:length(m_star)                  %for each hole, find the coordinate
                                          within the m_star element
    func = @(xhat) u3(lg2(:,m_star(i)))'*(hshape_fcn(xhat,h(m_star(i))));
    xhathole(i) = fzero(func,[0,1]);      %solve func for 0 ≤ xhathole ≤ 1
end
%Part (iii) find dimensional xholes
xhole = xpts(lg(1,m_star))+h(m_star).*xhathole;
                                          %locate xhatholes in the entire bar
xhole_d = xhole.*L_d;                     %scale by length of bar
xhole_d = sort(xhole_d);                  %sort in ascending order
```

Correct implementation of this protocol is assessed in the Results section.

RESULTS

Preliminary Test: Caresta Experiment

To verify the FE method, a test case is created based on a laboratory experiment by Caresta that studied the nodes of a vibrating beam [23]. The data of the beam are:

$$L = 1.275 \text{ m} \qquad A = h * b = (0.01 * 0.075)\text{ m}^2 \qquad I = \frac{bh^3}{12}$$

$$\rho = 7800 \text{ kg/m}^3 \qquad E = 2.1 \text{ x } 10^{11} \text{ Pa}$$

Caresta's results are:

$f_n = \frac{w_n}{2\pi}$	Theoretical [Hz]	Experimental [Hz]
$n = 1$	32.80	32.25
$n = 2$	90.44	88.50
$n = 3$	177.30	173.50
$n = 4$	293.08	287.50
$n = 5$	437.82	430.00

Caresta's geometry and properties of the beam, the fundamental frequency, and target frequency ratio are inputs for the FE code `xylo_bar_design3`. An input logical variable `justcalc_L_d` is set to be `true`, in which case the code bypasses the optimization of the material removal parameter and instead calculates the necessary length of the bar L_D to obtain the target natural frequency for the shape of the bar given by the first value of p_2 (which is set to 1 for an unmodified bar). The code then calculates the rest of the outputs with this particular length and shape, as summarized in **Table 4.2**.

Table 4.2: FEM results for Caresta test case

Based on Caresta's theoretical data		Based on Caresta's experimental data	
Inputs	Outputs	Inputs	Outputs
$f^{(3)}_{target\ d} = 32.80$ Hz $R_{target} = 90.44/32.80$ $H_{max\ d} = 0.01$ m $x^*_d = 0.05$ $p_2 = [1.0, 1.0]$ $E_{bar\ d} = 2.1 \text{ x } 10^{11}$ Pa $\rho_{bar\ d} = 7800 \text{ kg/m}^3$	$f^{(3)}_d = 32.800$ Hz $f^{(4)}_d = 90.414$ Hz $\epsilon f^{(3)}_d = 1.9475 \text{ x } 10^{-7}$ Hz $\epsilon f^{(4)}_d = 4.3736 \text{ x } 10^{-6}$ Hz $L_D = 1.2752$ m	$f^{(3)}_{target\ d} = 32.25$ Hz $R_{target} = 88.50/32.25$ $H_{max\ d} = 0.01$ m $x^*_d = 0.05$ $p_2 = [1.0, 1.0]$ $E_{bar\ d} = 2.1 \text{ x } 10^{11}$ Pa $\rho_{bar\ d} = 7800 \text{ kg/m}^3$	$f^{(3)}_d = 32.250$ Hz $f^{(4)}_d = 88.898$ Hz $\epsilon f^{(3)}_d = 1.9148 \text{ x } 10^{-7}$ Hz $\epsilon f^{(4)}_d = 4.3002 \text{ x } 10^{-6}$ Hz $L_D = 1.2860$ m

The FE results are very accurate, providing evident that the code is implemented correctly. With the theoretical data, the beam's predicted length and first harmonic frequency only differ from Caresta's results by 0.016% and -0.0029%, respectively. With the experimental data, the beam's predicated length and harmonic frequency differ by 0.86% and 0.45% respectively. The beam's first two modes are shown in **Figure 4.4**.

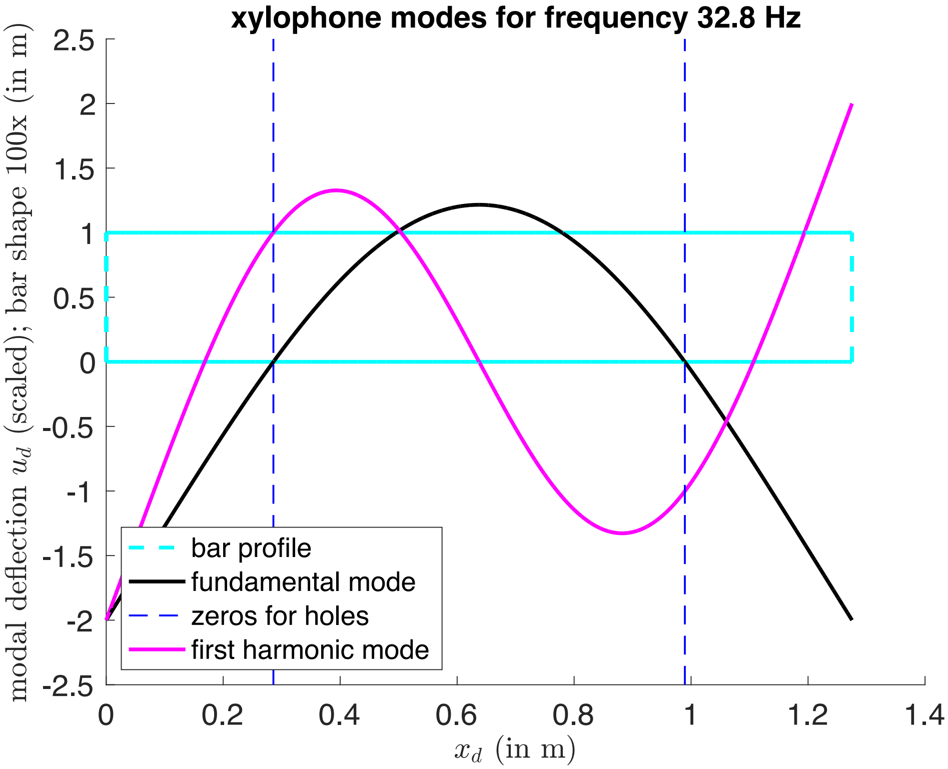


Figure 4.4: Beam modes based on Caresta's theoretical data.

Design Test: Tuning a Xylophone Bar

The FE code was implemented to tune the xylophone bar. Tune 1 specified a C5 pitch target and quint tuning, and Tune 2 specified a F5 pitch target and double-octave tuning. The input parameters and results are summarized in **Table 4.3**. The fundamental and first harmonic frequencies and their associated errors are direct outputs of the function `xylo_bar_design3`. R is then computed from the outputs and compared to the target ratio. The error of R is estimated using two methods. The propagation of uncertainty method dictates that for $R = f_d^{(4)}/f_d^{(3)}$, the error of R is:

$$\epsilon R_{prop} = \sqrt{\left(\frac{\partial R}{\partial f_d^{(3)}}\epsilon f_d^{(3)}\right)^2 + \left(\frac{\partial R}{\partial f_d^{(4)}}\epsilon f_d^{(4)}\right)^2} = \sqrt{\left(\frac{-f_d^{(4)}}{{f_d^{(3)}}^2}\epsilon f_d^{(3)}\right)^2 + \left(\frac{1}{f_d^{(3)}}\epsilon f_d^{(4)}\right)^2} \quad (82)$$

Another, simplifier method uses the errors of $f_d^{(3)}$ and $f_d^{(4)}$ to compute the upper and lower bounds of R, takes their differences with R, and then defines the maximum difference as the error of R:

$$R_{upper} = \frac{\max f_d^{(4)}}{\min f_d^{(3)}} = \frac{f_d^{(4)} + \epsilon f_d^{(4)}}{f_d^{(3)} - \epsilon f_d^{(4)}} \quad \& \quad R_{lower} = \frac{\min\,(f_4)}{\max\,(f_3)} = \frac{f_4 - \epsilon f_4}{f_3 + \epsilon f_3} \quad (83)$$
$$\epsilon R_{max} = \max(|R_{upper} - R|, |R_{lower} - R|)$$

As shown in **Table 4.4**, R is quite close to R_{target} and the error estimators of R are very small. For context, the frequency resolution of the human ear is on the order of 10 Hz; thus, all results reported have unreasonably tighter tolerances for designing a xylophone bar (too many refinements). Nonetheless, these small errors help verify the FE method. Observe that for both tunings, $\epsilon R_{prop} < \epsilon R_{max}$. Since the maximum difference method gives an extreme estimation, the error bound is generally tighter. The propagation method is more accurate because it weighs the function's derivatives. However, this method only holds for small errors because it uses a first-order Taylor series expansion. Finally, observe that $\epsilon f_d^{(4)} > \epsilon f_d^{(3)}$; for the same mesh, the FE method does a poorer job capturing higher harmonics because smaller wavelengths are less precisely captured with finite element resolution.

Table 4.3: Tuning a xylophone bar

Tune 1	Tune 2
Inputs	
$f_{target\ d}^{(3)} = 523.35$ Hz	$f_{target\ d}^{(3)} = 698.46$ Hz
$R_{target} = 3$	$R_{target} = 4$
$H_{max\ d} = 0.015$ m	$H_{max\ d} = 0.015$ m
$x_d^* = 0.05$	$x_d^* = 0.05$
$p_2 = [0.05, 1.0]$	$p_2 = [0.05, 1.0]$
$E_{bar\ d} = 1.4 \text{ x } 10^{10}$ Pa	$E_{bar\ d} = 1.4 \text{ x } 10^{10}$ Pa
$\rho_{bar\ d} = 835$ kg/m^3	$\rho_{bar\ d} = 835$ kg/m^3
Outputs	
$f_d^{(3)} = 523.35$ Hz	$f_d^{(3)} = 698.46$ Hz
$f_d^{(4)} = 1565.5$ Hz	$f_d^{(4)} = 2787.6$ Hz
$\epsilon f_d^{(3)} = 8.7772 \text{ x } 10^{-6}$ Hz	$\epsilon f_d^{(3)} = 1.1584 \text{ x } 10^{-4}$ Hz
$\epsilon f_d^{(4)} = 1.4858 \text{ x } 10^{-4}$ Hz	$\epsilon f_d^{(4)} = 1.6375 \text{ x } 10^{-3}$ Hz
$p_2^{opt} = 0.64380$	$p_2^{opt} = 0.13906$
$L_D = 0.26666$ m	$L_D = 0.095486$ m

Table 4.4: Target ratio results and error estimators

Tune 1	Tune 2
$R = 2.9913$	$R = 3.9911$
$\epsilon R_{prop} = 2.8830 \text{ x } 10^{-7}$	$\epsilon R_{prop} = 2.4362 \text{ x } 10^{-6}$
$\epsilon R_{max} = 3.3407 \text{ x } 10^{-7}$	$\epsilon R_{max} = 3.0064 \text{ x } 10^{-6}$

Error estimators for Tune 1 and Tune 2 are shown in **Figure 4.5** and **Figure 4.6** respectively. Focusing on the error in the output, observe that the error bound converges at the expected rate—a slope of -4 on a log-log plot. From the outputs of **Table 4.3**, it can be seen that the FE prediction is more accurate for Tune 1 than Tune 2.

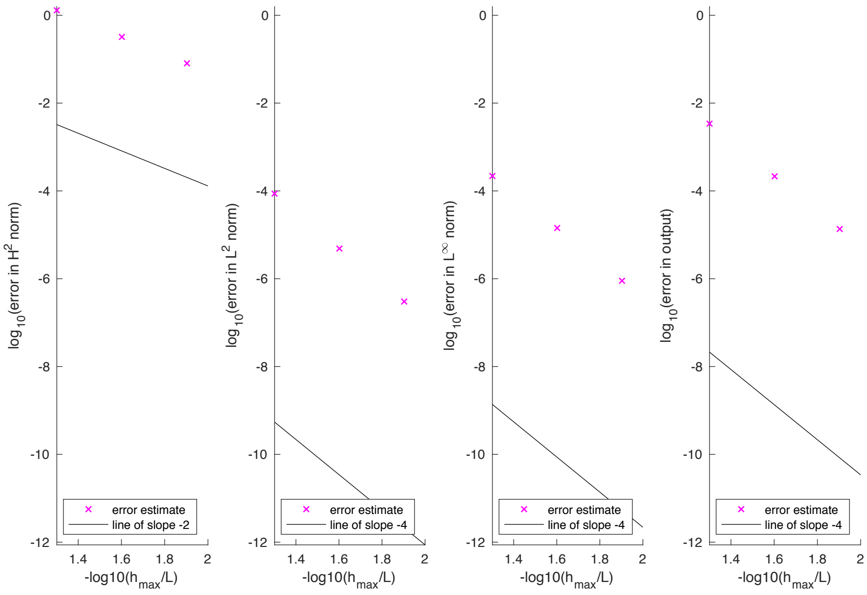


Figure 4.5: Error estimator for Tune 1.

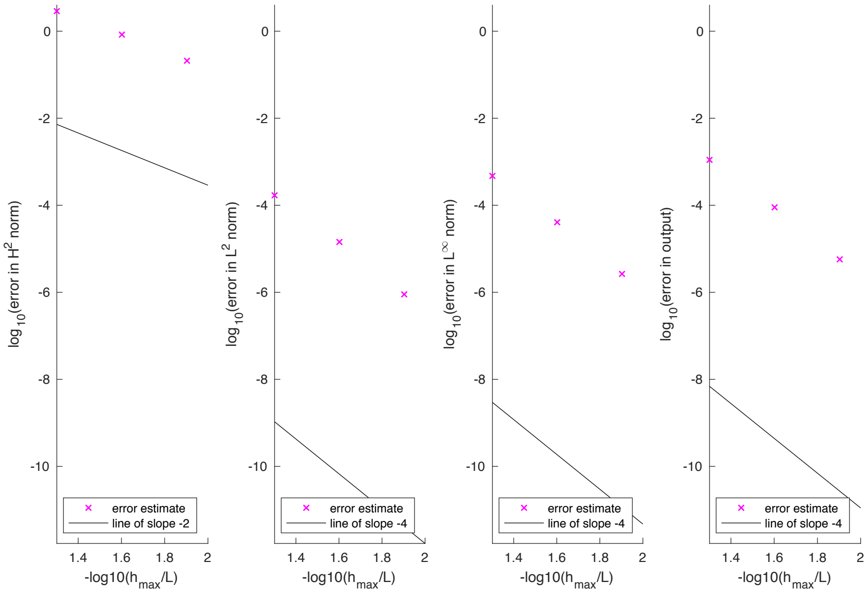


Figure 4.6: Error estimator for Tune 2.

Figure 4.7 and **Figure 4.8** show the first and second modes of the xylophone bar for Tune 1 and Tune 2, respectively. Note that in both cases the hole placements are located where the deflection of the fundamental mode is zero. This is the best evidence that hole finder protocol discussed in the Background has been implemented correctly.

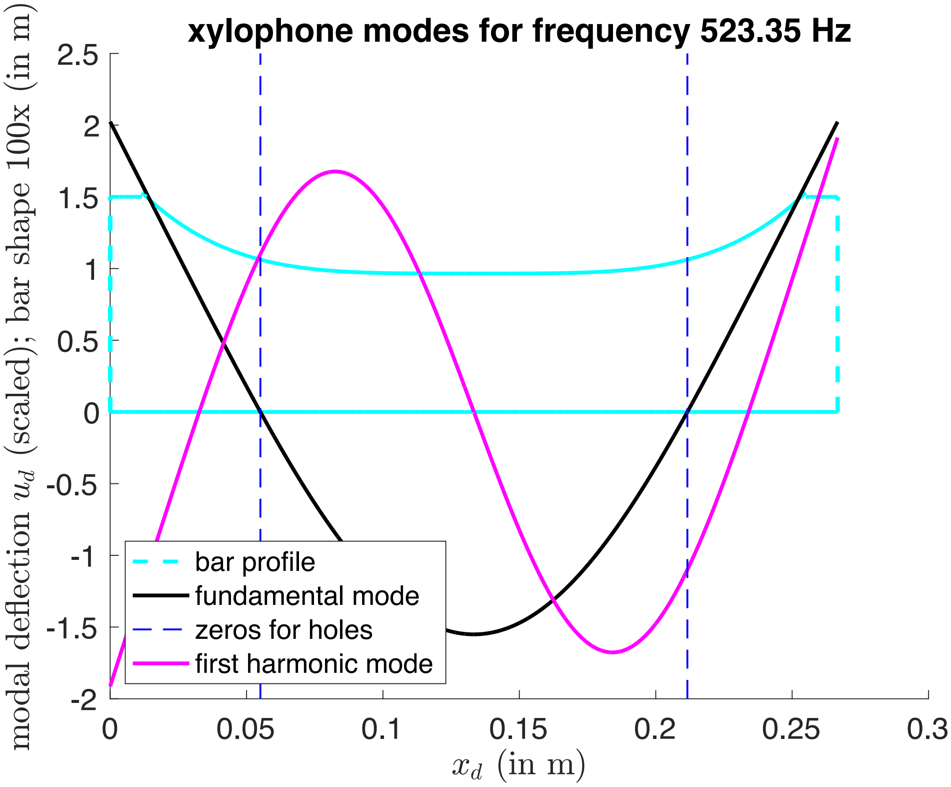


Figure 4.7: Xylophone bar modes for Tune 1.

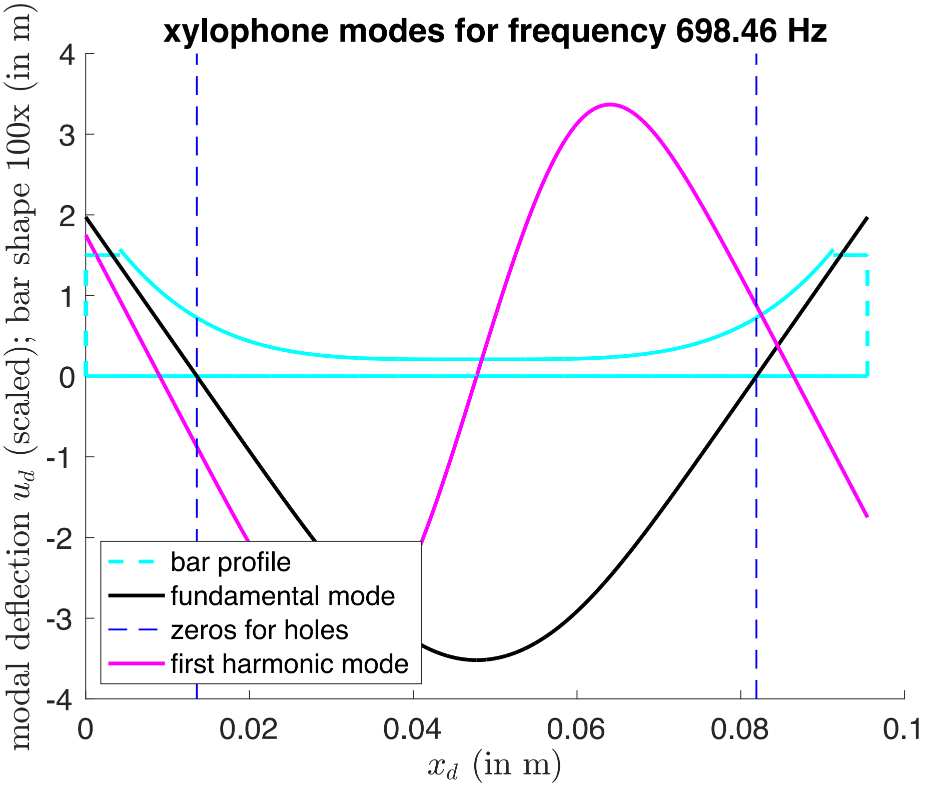


Figure 4.8: Xylophone bar modes for Tune 1.

DISCUSSION

Since the xylophone bar is modeled as a Euler-Bernoulli beam, which applies to slender beams, the model's accuracy depends on the tuned frequency, independent of the FE treatment. For a bar with fixed height $H_{max\,d}$, the higher the tuned frequency, the shorter and less slender the bar, and the less accurate the Euler-Bernoulli model. Thus, larger modeling errors are expected for bars tuned to higher frequencies.

As a final note, to make the xylophone problem more physically accurate, the effect of the support strings that are threaded through the two holes could be modeled. Consider a beam of length L with a lumped (massless) Hookean spring attached to the right end. At the left end, $x = 0$, the beam is free: $u_{xx} = u_{xxx} = 0$. At the right end, $x = L$, the beam has zero moment and shear force equivalent to the spring force: $u_{xx} = 0$ and $-(EIu_{xx})_x = -k_s u$, where k_s is the positive spring constant. The spring introduces a "Robin" boundary condition. The energy functional $\Pi(\omega)$ only requires slight modification and changes the stiffness matrix to [2]:

$$A_{ij} = \int_0^L EI \frac{d^2\varphi_i}{dx^2}\frac{d^2\varphi_j}{dx^2}dx + k_s\varphi_i(L)\varphi_j(L), \quad 1 \le i,j \le 2n_{node} \qquad (84)$$

The load vector $\underline{F}$ is unchanged.

The actual implementation of the red term in Eq. (84) is straightforward. Since $\varphi_i(L)$ and $\varphi_j(L)$ are only non-zero at the rightmost node, where they have values of 1, the red term need only be added to $A_{2n_{node}-1,2n_{node}-1}$, the displacement degree of freedom at the rightmost node. After the stiffness matrix is constructed in the code `impose_boundary_cond` [24], at say line 58, the following lines of code could be added:

```
if exist ks
      RSD = ttomap_fcn(n_el0+1,1);    %maps to the Rightmost Displacement DOF
      A(RSD,RSD) = A(RSD,RSD) + ks;   %adds ks to the stiffness matrix
end
```

Acknowledgements

I would like to thank Professor Anthony Patera for offering this excellent course, his dedication to enhancing my learning, and his thoughtful feedback on my work. I would also like to thank James Penn for his experiments that contributed to the class project and helped validate our results.

References

[1] Patera, Anthony, "2.S976 Syllabus," Version: December 7, 2019.

[2] Patera, Anthony, "Project: Description, Deliverables, and Schedule," Introduction to Finite Element Methods for Mechanical Engineers, Version: May 9, 2019.

[3] Patera, Anthony, "The Rayleigh-Ritz Method for 1D 2nd-Order SPD BVPs," Introduction to Finite Element Methods for Mechanical Engineers, Version: February 14, 2019.

[4] http://web.mit.edu/16.unified/www/FALL/thermodynamics/notes/node117.html

[5] Patera, Anthony, "FEM for 1D SPD BVP: Formulation," Introduction to Finite Element Methods for Mechanical Engineers, Version: February 25, 2019.

[6] Patera, Anthony, "FEM for 1D SPD BVP: Implementation," Introduction to Finite Element Methods for Mechanical Engineers, Version: February 25, 2019.

[7] Patera, Anthony, "FEM for 1D SPD BVP: P2 Elements," Introduction to Finite Element Methods for Mechanical Engineers, Version: March 7, 2019.

[8] Adapted from: Patera, Anthony, "form_elem_mat_sver.m," Version: March 6, 2019.

[9] Adapted from: Patera, Anthony, "impose_boundary_cond_sver.m," Version: March 6, 2019.

[10] Patera, Anthony, "FEM for 1D SPD BVP: Verification Techniques," Introduction to Finite Element Methods for Mechanical Engineers, Version: March 12, 2019.

[11] Adapted from: Patera, Anthony, "run_uniform_refinement.m," Version: March 6, 2019.

[12] Patera, Anthony, "The Heat Equation: Study Cases," Introduction to Finite Element Methods for Mechanical Engineers, Version: April 2, 2019.

[13] Patera, Anthony, "The Heat Equation: Formulation and Implementation," Introduction to Finite Element Methods for Mechanical Engineers, Version: March 5, 2019.

[14] Patera, Anthony, "The Heat Equation: Error Estimation for Verification," Introduction to Finite Element Methods for Mechanical Engineers, Version: February 25, 2019.

[15] Adapted from: Patera, Anthony, "solve_fld_output_t_sver.m," Version: March 6, 2019.

[16] Adapted from: Patera, Anthony, "run_uniform_refinement_burger_sver.m," Version: April 2, 2019.

[17] Adapted from: Patera, Anthony, Figures in Burger Folder, Version: April 2, 2019.

[18] Cook's Illustrated, "Tender, Juicy Grilled Burgers," July 2014, https://www.cooksillustrated.com/recipes/7969-tender-juicy-grilled-burgers

[19] Patera, Anthony, "Bending Eigenproblem Study Case: Xylophone Bar," Introduction to Finite Element Methods for Mechanical Engineers, Version: April 23, 2019.

[20] Patera, Anthony, "Bending: Natural Frequencies Eigenproblem," Introduction to Finite Element Methods for Mechanical Engineers, Version: April 23, 2019.

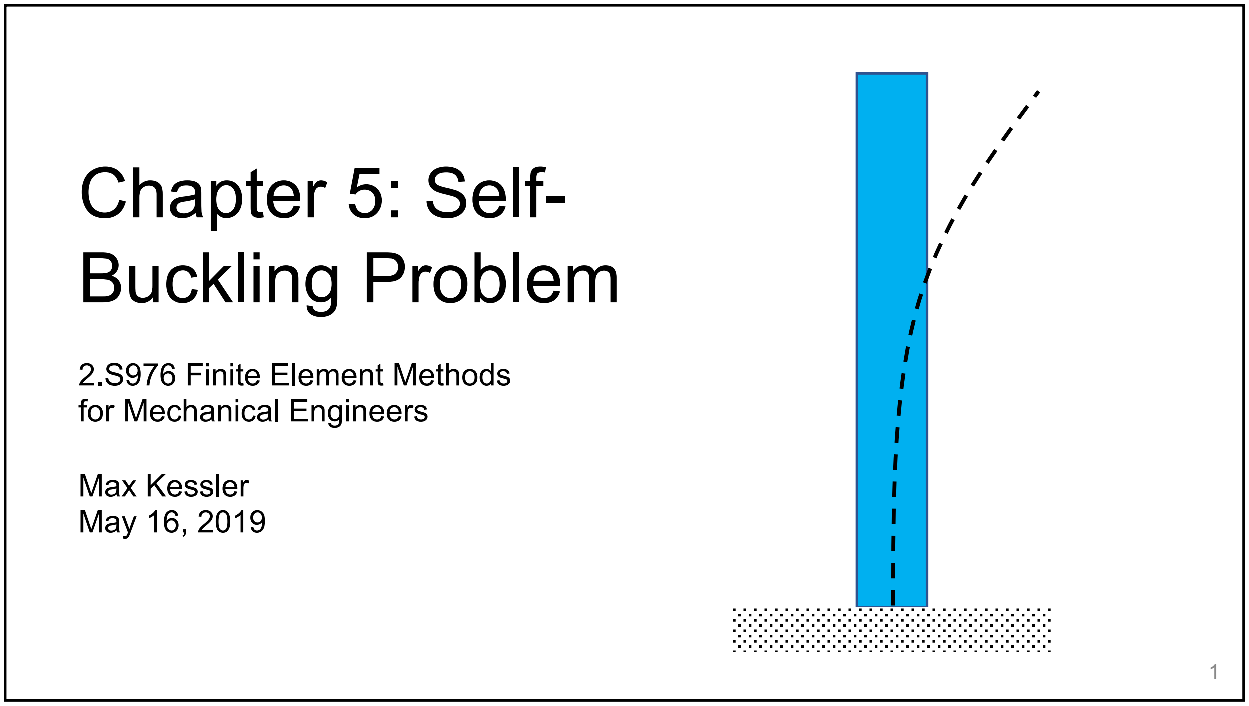
[21] Patera, Anthony, "Bending: Finite Element Method," Introduction to Finite Element Methods for Mechanical Engineers, Version: April 11, 2019.

[22] Adapted from: Patera, Anthony, "xylo_bar_design3.m," Version: April 25, 2019.

[23] Caresta, Mauro, "Vibrations of a Free-Free Beam."

[24] Adapted from: Patera, Anthony, "impose_boundary_cond.m," Version: April 25, 2019.

Addendum – Chapter 5: Self-Buckling

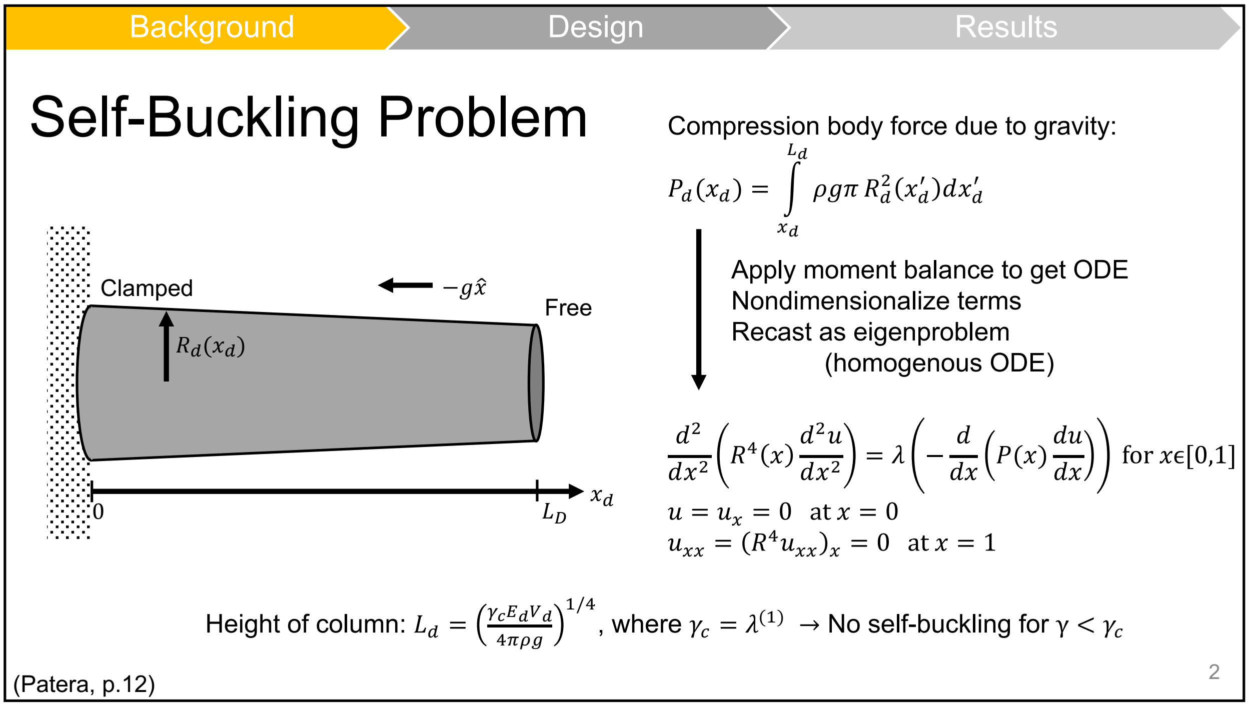


Chapter 5: Self-Buckling Problem

2.S976 Finite Element Methods
for Mechanical Engineers

Max Kessler
May 16, 2019

1



Background | Design | Results

Self-Buckling Problem

Compression body force due to gravity:

$$P_d(x_d) = \int_{x_d}^{L_d} \rho g \pi \, R_d^2(x_d')dx_d'$$

Apply moment balance to get ODE
Nondimensionalize terms
Recast as eigenproblem
(homogenous ODE)

$$\frac{d^2}{dx^2}\left(R^4(x)\frac{d^2u}{dx^2}\right) = \lambda\left(-\frac{d}{dx}\left(P(x)\frac{du}{dx}\right)\right) \text{ for } x\epsilon[0,1]$$

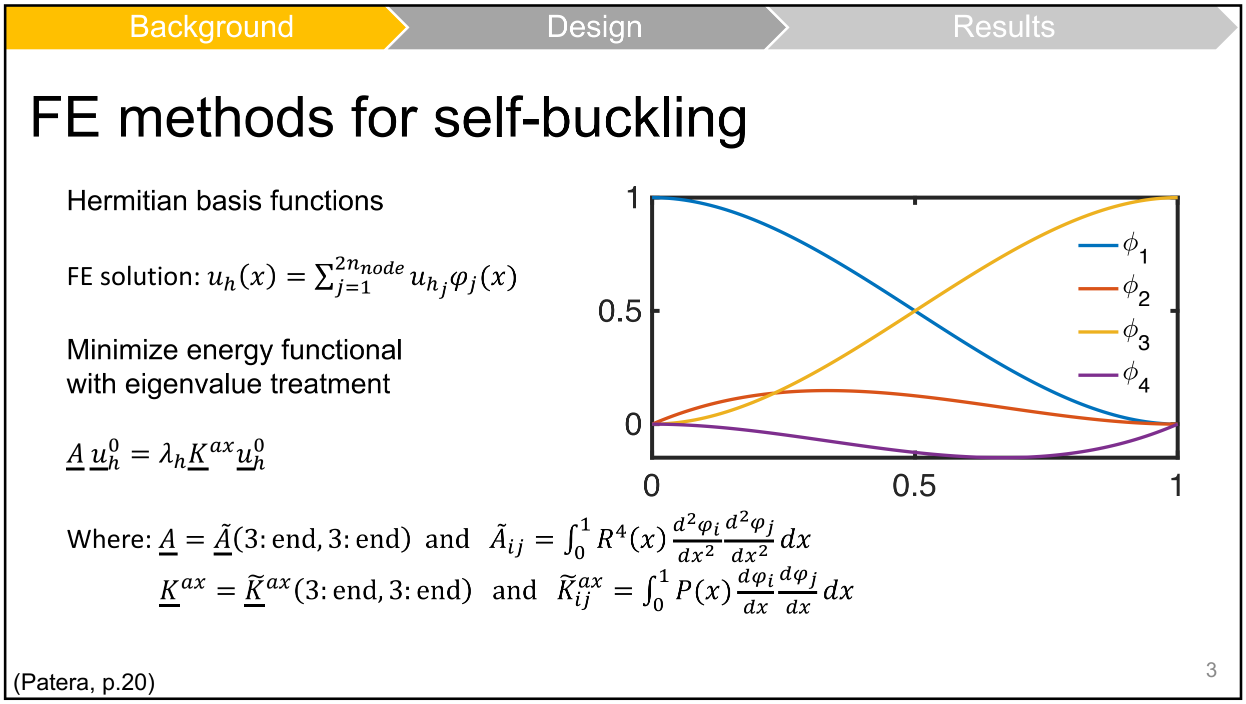
$$u = u_x = 0 \quad \text{at } x = 0$$

$$u_{xx} = (R^4 u_{xx})_x = 0 \quad \text{at } x = 1$$

Height of column: $L_d = \left(\frac{\gamma_c E_d V_d}{4\pi\rho g}\right)^{1/4}$, where $\gamma_c = \lambda^{(1)}$ → No self-buckling for $\gamma < \gamma_c$

(Patera, p.12)

2



Background Design Results

FE methods for self-buckling

Hermitian basis functions

FE solution: $u_h(x) = \sum_{j=1}^{2n_{node}} u_{h_j}\varphi_j(x)$

Minimize energy functional
with eigenvalue treatment

$$\underline{A}\,\underline{u}_h^0 = \lambda_h \underline{K}^{ax}\underline{u}_h^0$$

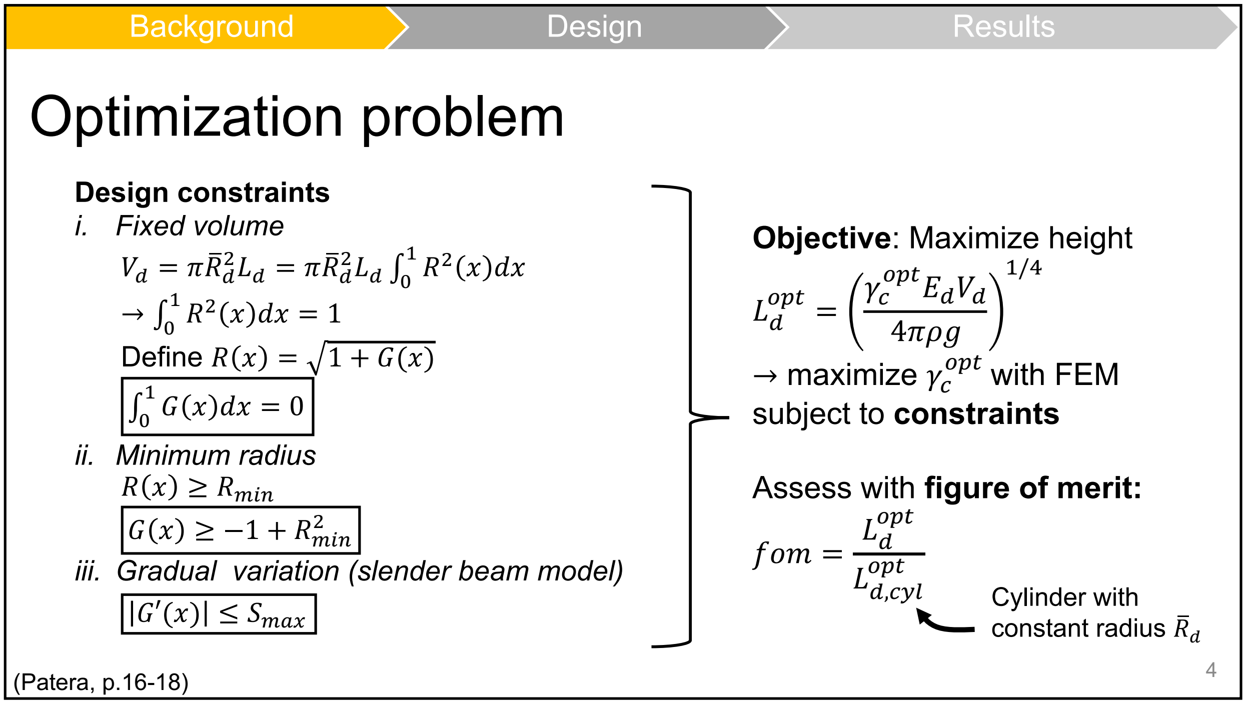
1
0.5
0
0
0.5
1
ϕ_1
ϕ_2
ϕ_3
ϕ_4

Where: $\underline{A} = \underline{\tilde{A}}(3:\text{end},3:\text{end})$ and $\tilde{A}_{ij} = \int_0^1 R^4(x)\frac{d^2\varphi_i}{dx^2}\frac{d^2\varphi_j}{dx^2}dx$

$\underline{K}^{ax} = \underline{\tilde{K}}^{ax}(3:\text{end},3:\text{end})$ and $\tilde{K}^{ax}_{ij} = \int_0^1 P(x)\frac{d\varphi_i}{dx}\frac{d\varphi_j}{dx}dx$

(Patera, p.20)

3



Background Design Results

Optimization problem

Design constraints

i. Fixed volume

$V_d = \pi\bar{R}_d^2 L_d = \pi\bar{R}_d^2 L_d \int_0^1 R^2(x)dx$

$\rightarrow \int_0^1 R^2(x)dx = 1$

Define $R(x) = \sqrt{1+G(x)}$

$\boxed{\int_0^1 G(x)dx = 0}$

ii. Minimum radius

$R(x) \geq R_{min}$

$\boxed{G(x) \geq -1 + R_{min}^2}$

iii. Gradual variation (slender beam model)

$\boxed{|G'(x)| \leq S_{max}}$

Objective: Maximize height

$$L_d^{opt} = \left(\frac{\gamma_c^{opt} E_d V_d}{4\pi\rho g}\right)^{1/4}$$

→ maximize γ_c^{opt} with FEM subject to **constraints**

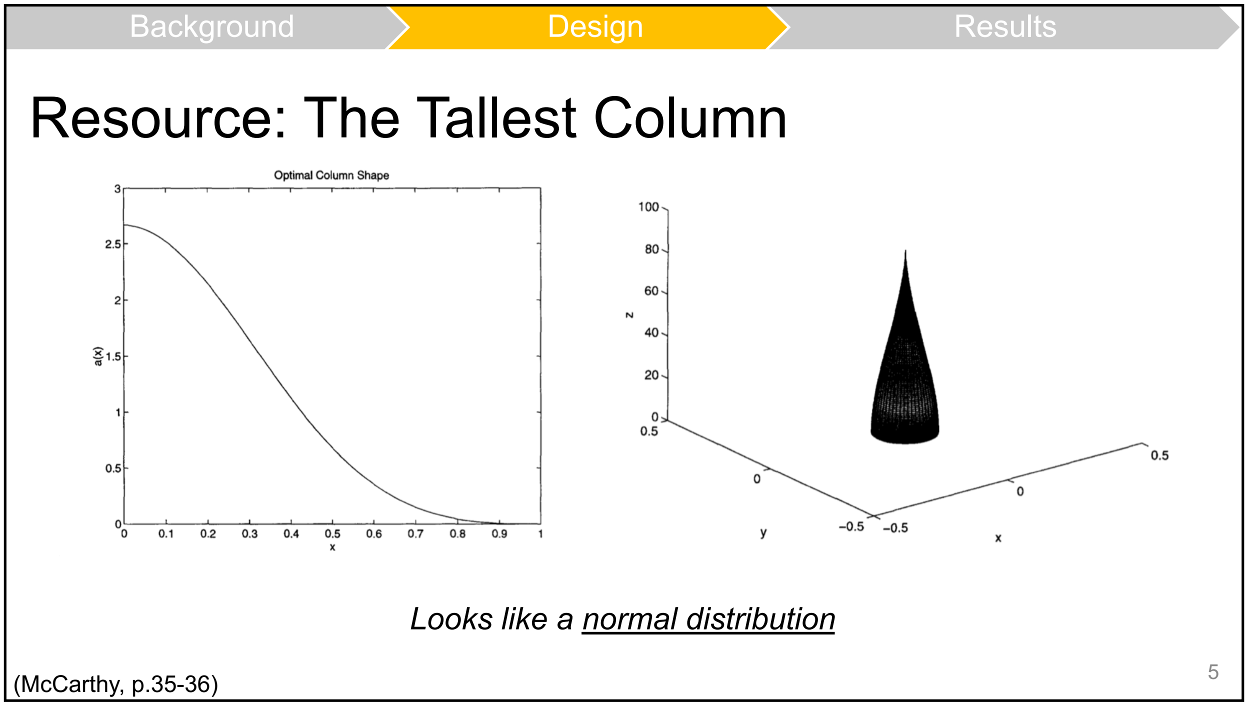
Assess with **figure of merit:**

$$fom = \frac{L_d^{opt}}{L_{d,cyl}^{opt}}$$

Cylinder with constant radius $\bar{R}_d$

(Patera, p.16-18)

4



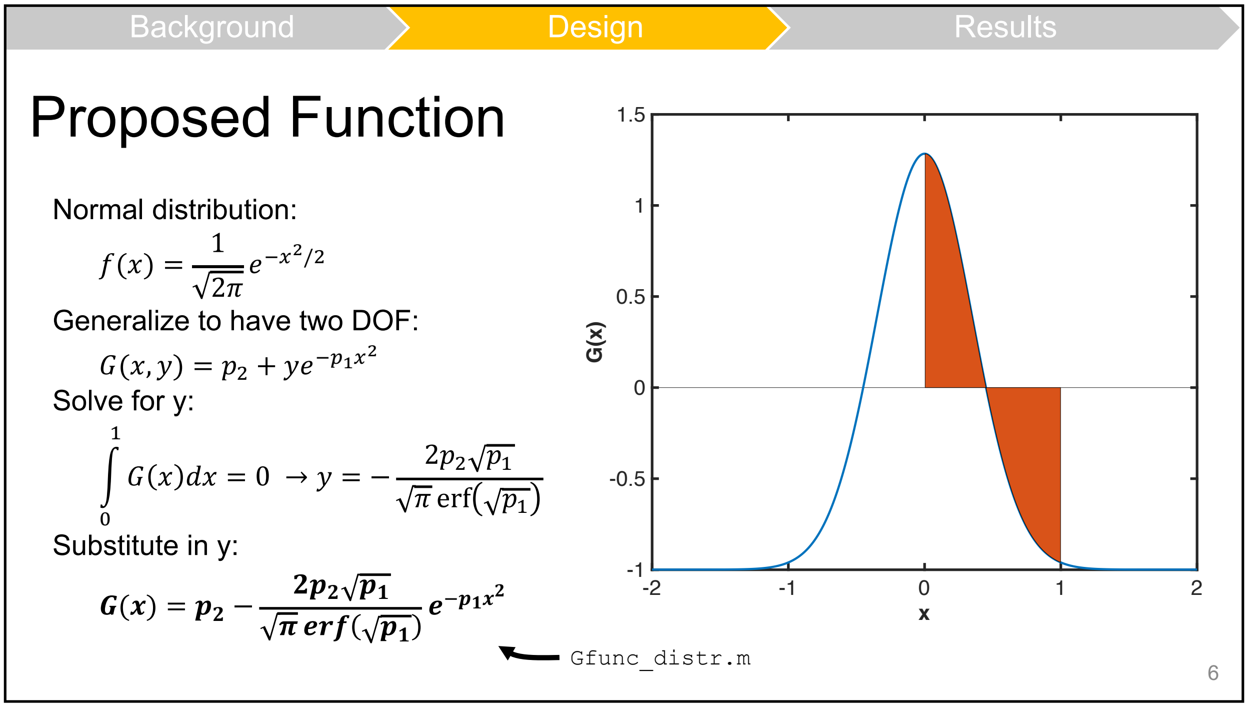
Background | Design | Results

Resource: The Tallest Column

Looks like a <u>normal distribution</u>

(McCarthy, p.35-36)

5



Background | Design | Results

Proposed Function

Normal distribution:

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$$

Generalize to have two DOF:

$$G(x,y) = p_2 + ye^{-p_1 x^2}$$

Solve for y:

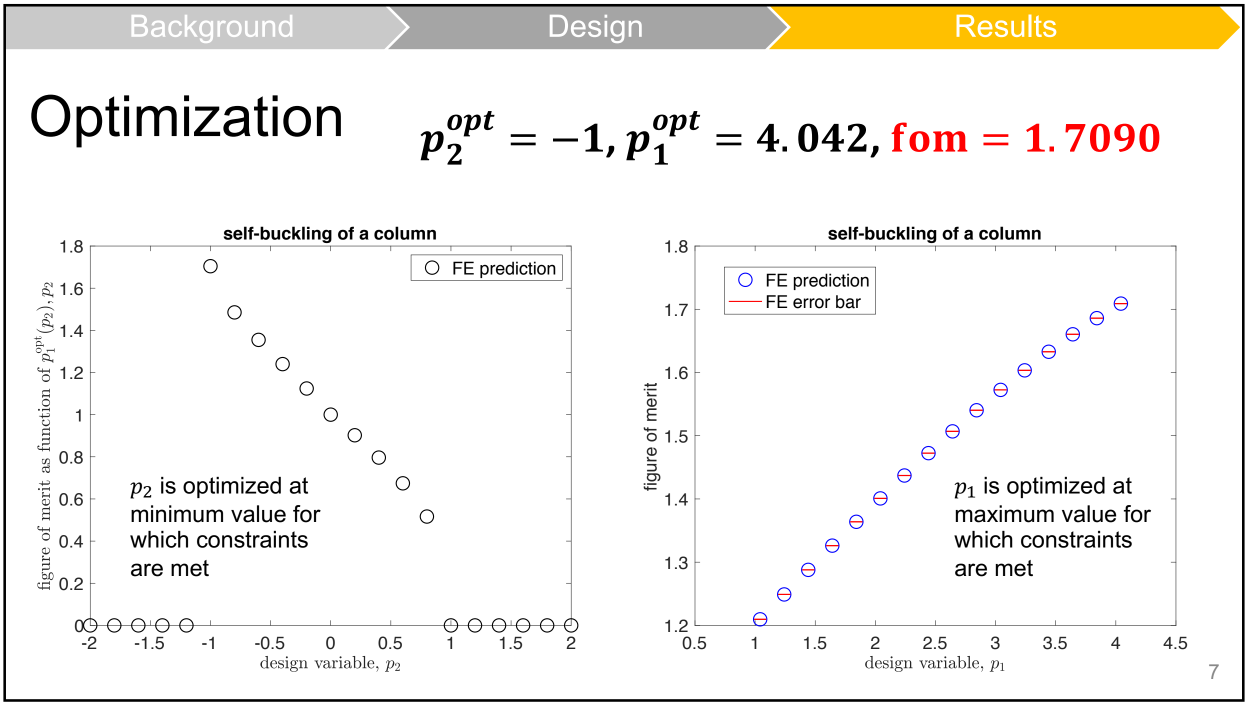
$$\int_0^1 G(x)dx = 0 \;\rightarrow y = -\frac{2p_2\sqrt{p_1}}{\sqrt{\pi}\,\mathrm{erf}\left(\sqrt{p_1}\right)}$$

Substitute in y:

$$\boldsymbol{G(x) = p_2 - \frac{2p_2\sqrt{p_1}}{\sqrt{\pi}\,erf(\sqrt{p_1})} e^{-p_1 x^2}}$$

`Gfunc_distr.m`

6



Background | Design | Results

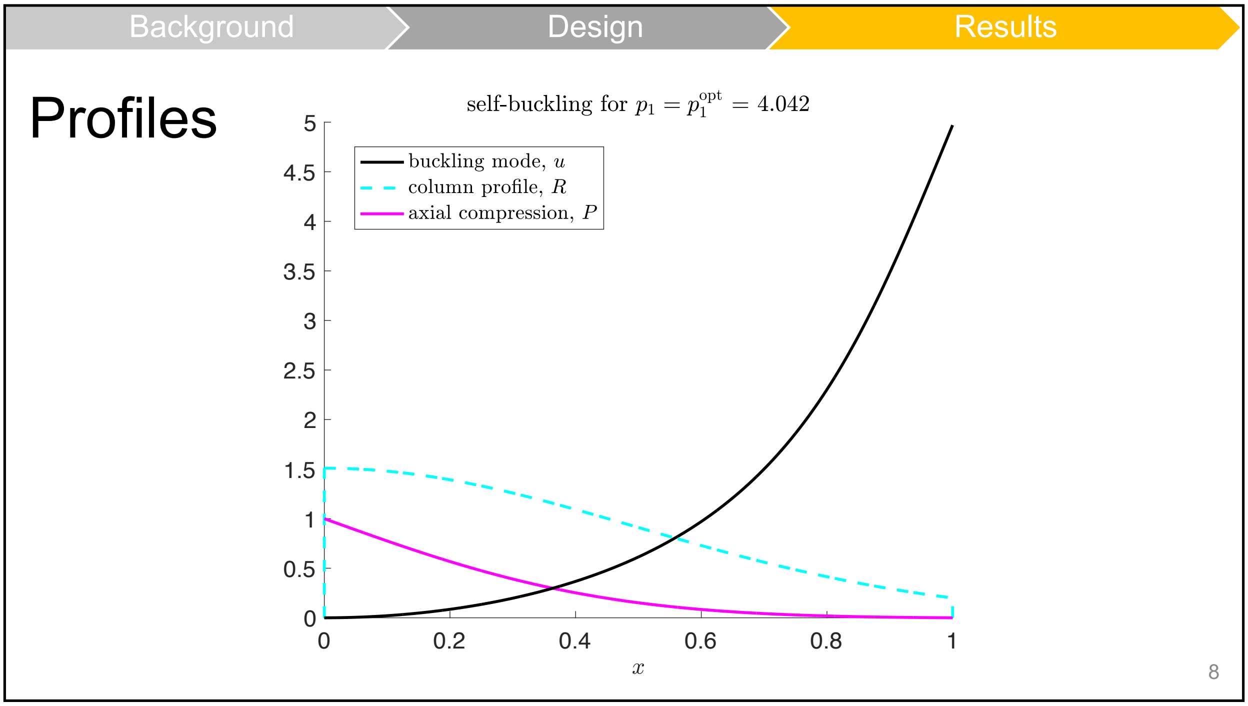
Optimization

$$p_2^{opt} = -1, p_1^{opt} = 4.042, \text{fom} = 1.7090$$

p_2 is optimized at minimum value for which constraints are met

p_1 is optimized at maximum value for which constraints are met

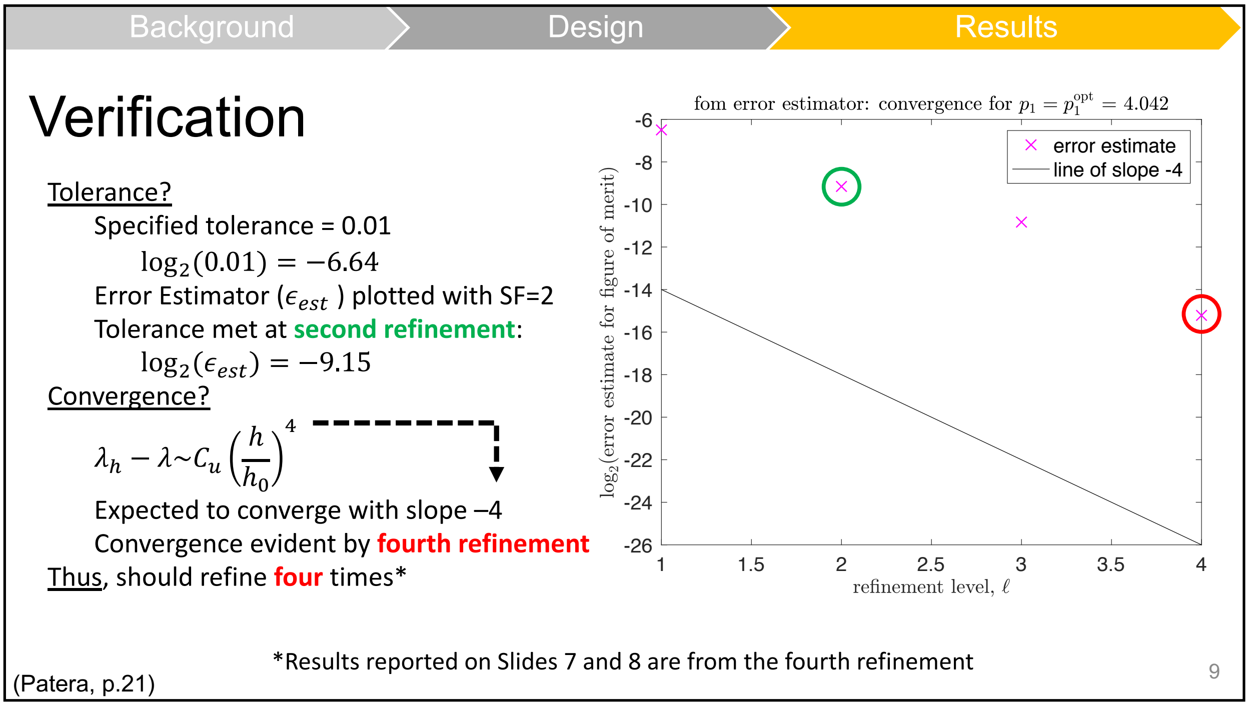
7



Background | Design | Results

Profiles

8



Verification

Tolerance?

Specified tolerance = 0.01

$\log_2(0.01) = -6.64$

Error Estimator (ϵ_{est}) plotted with SF=2

Tolerance met at **second refinement**:

$\log_2(\epsilon_{est}) = -9.15$

Convergence?

$$\lambda_h - \lambda \sim C_u \left(\frac{h}{h_0}\right)^4$$

Expected to converge with slope –4

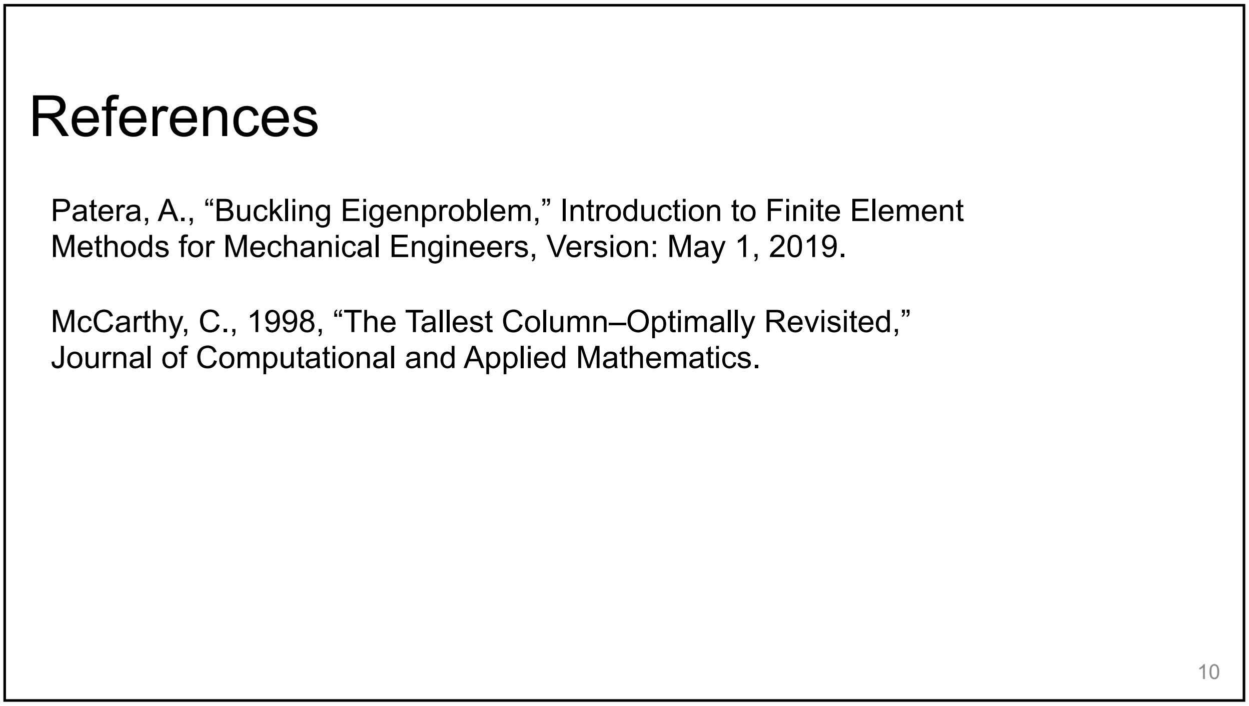
Convergence evident by **fourth refinement**

Thus, should refine **four** times*

*Results reported on Slides 7 and 8 are from the fourth refinement

(Patera, p.21)

9



References

Patera, A., “Buckling Eigenproblem,” Introduction to Finite Element Methods for Mechanical Engineers, Version: May 1, 2019.

McCarthy, C., 1998, “The Tallest Column–Optimally Revisited,” Journal of Computational and Applied Mathematics.

10