Survey and Applications of the 1D Finite Element Method

Fidel Cano Rentería*
Massachusetts Institute of Technology
Department of Mechanical Engineering

May 16, 2019

*I'd like to thank Anthony Patera for his guidance and instruction in 2.S976: Finite Element Methods for Mechanical Engineers

Abstract

We present a survey of the one dimensional Finite Element Method, and a survey of 3D problems that can be simplified to Quasi-1D models that can be solved by the one dimensional Finite Element Method. The ability to solve 3D problems with high fidelity with a 1D Finite Element Method can greatly speed up computations and optimize memory usage. In particular, we test the ability of our 1D Finite Element Method to corroborate parameters specified in popular burger recipes, as well as those specified in the manufacturing of xylophone bars.

1 Introduction

The heat equation, the Euler-Bernoulli beam bending equations, and the Navier-Stokes fluid equations are all examples of Partial Differential Equations (PDE's) that are heavily studied across engineering disciplines. Despite their ubiquity, these are often impossible to solve without simplifying assumptions that may alter some problems significantly; as such, numerical methods are essential for the study of these equations. High-dimensional PDE's, however, are notoriously difficult to solve in a reasonable amount of time, and it is often difficult to find computers that can even store the solutions [Weinan et al., 2017]. In this work, we survey some inherently 3D problems that can be converted into 1D problems, so that they can be solved by a simple 1D Finite Element Method, instead of the computationally intensive 3D methods.

To begin exploring the 1D Finite Element Method, we introduce the Rayleigh-Ritz method. The Rayleigh-Ritz algorithm is a simple finite element method that takes in a basis for a vector space of functions and a metric, and proceeds to find the function in the specified vector space that minimizes the distance between it and the solution to a partial differential equation of interest under the specified metric [Patera, 2019n]. We seek to employ the Rayleigh-Ritz method to develop an understanding of the evolution of temperature distributions on pots and pans while cooking. In industry, an adequate model can improve cooking time estimates and optimize kitchen scheduling, propelling the emerging field of robotic kitchens forward. At home, the same model can help foster safer cooking practices seeking to prevent burns and other kitchen related injuries.

We will also advance the theory developed from the Rayleigh-Ritz Method to create software that can assist in the manufacturing of musical instruments - xylophones, in particular.

2 The Rayleigh-Ritz Method for One Dimensional Boundary Value Problems

A proper description of a pan's temperature distribution time evolution requires solving for a temperature function U that depends on 3 spatial coordinates, and one time coordinate. This function $U : R^4 \to R$ can be seen as a solution to the heat equation (Eq. 1) subject to an appropriate set of boundary conditions [Patera, 2019m].

$$\rho C_p \frac{\partial U}{\partial t} - \nabla \cdot (k \nabla U) = f \tag{1}$$

Here, ρ, C_p and k refer to the possibly space dependent density, specific heat capacity, and thermal conductivity of the material, respectively. Furthermore, t is the time coordinate, ∇ is the del operator on R^3, and f is an arbitrary forcing function.

Under some special conditions, there exist solutions to the heat equation that can be written in terms of elementary functions. Unfortunately, the geometries bestowed upon kitchen pans do not exhibit exploitable symmetries, making numerical methods necessary for an adequate description of the function U. As such, the Rayleigh-Ritz method has been selected in this work to yield an accurate approximation of the temperature distribution function.

To develop the necessary mathematical tools for a successful implementation of the Rayleigh-Ritz method, we explore the method on two simpler, one dimensional versions of the heat equation. In both cases, we analyze a steady state scenario ($\frac{\partial U}{\partial t} = 0$) and focus on a skillet handle of length L that can be approximated to have a uniform temperature distribution in each cross-section so that only one spatial coordinate is necessary for describing U.

2.1 Insulated Conical Frustum

The first of the two simplified heat equations models the skillet handle as a conical fustrum insulated throughout its lateral area, with conductive heat transfer occurring at the boundary with the pan ($x = 0$) and convective heat transfer with the air around it at the tip ($x = L$). In this case, the heat equation collapses to Eq. 2 [Patera, 2019n].

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

Subject to:

- $k \frac{dU}{dx}|_{x=0} = -q_1$
- $-k \frac{dU}{dx}|_{x=L} = \eta_2 \left(U(L) - U_\infty \right)$

Here, k is the thermal conductivity of the skillet's material, R_0 is a nominal radius, η_2 is the effective convective heat transfer coefficient at the tip of the skillet handle, U_∞ is the ambient temperature, and q_1 is the heat flux coming into the skillet handle from the pan.

The Rayleigh-Ritz method begins with the specification of a vector space of functions over which we seek to find an approximation to U [Patera, 2019o]. The vector space is specified by a set of basis functions $\{\psi_1, \psi_2, ..., \psi_n\}$. We now consider the functional $\pi(w)$ defined below.

$$\boxed{\pi(w)=\frac{1}{2}\int_0^L k\left(1+\beta\frac{x}{L}\right)^2\left(\frac{dw}{dx}\right)^2dx+\frac{1}{2}\left(1+\beta\right)^2\eta_2 w(L)^2-q_1w(0)-\left(1+\beta\right)^2\eta_2U_\infty w(L)}$$

We now show that the function that minimizes $\pi(w)$ is the solution to our problem [Patera, 2019o].

Proof. Let U be a solution to the posed problem. Now consider an arbitrary function v. We now compute $\pi(U+v)$.

$$\pi(U+v)=\frac{1}{2}\int_0^L k\left(1+\beta\frac{x}{L}\right)^2\left(\frac{dU}{dx}+\frac{dv}{dx}\right)^2dx+\frac{1}{2}\left(1+\beta\right)^2\eta_2\left(U(L)+v(L)\right)^2$$
$$-q_1\left(U(0)+v(0)\right)-\left(1+\beta\right)^2\eta_2U_\infty\left(U(L)+v(L)\right)$$

Expanding the quadratic terms, we get:

$$\frac{1}{2}\int_0^L k\left(1+\beta\frac{x}{L}\right)^2\left(\frac{dU}{dx}\right)^2dx+\int_0^L k\left(1+\beta\frac{x}{L}\right)^2\left(\frac{dU}{dx}\frac{dv}{dx}\right)dx+\frac{1}{2}\int_0^L k\left(1+\beta\frac{x}{L}\right)^2\left(\frac{dv}{dx}\right)^2dx$$
$$+\frac{1}{2}\left(1+\beta\right)^2\eta_2U(L)^2+\left(1+\beta\right)^2\eta_2U(L)v(L)+\frac{1}{2}\left(1+\beta\right)^2\eta_2v(L)^2$$
$$-q_1U(0)-q_1v(0)-\left(1+\beta\right)^2\eta_2U_\infty U(L)-\left(1+\beta\right)^2\eta_2U_\infty v(L)$$

Now note that when we group the first integral term with the $\frac{1}{2}\left(1+\beta\right)^2\eta_2U(L)^2$ term, the $-q_1U(0)$ term, and the $-\left(1+\beta\right)^2U_\infty U(L)$ term, we get $\pi(U)$ by definition of $\pi(U)$. We will simplify the second integral term using integration by parts.

$$\int_0^L k\left(1+\beta\frac{x}{L}\right)^2\left(\frac{dU}{dx}\frac{dv}{dx}\right)dx=\left[k\left(1+\beta\frac{x}{L}\right)^2\frac{dU}{dx}v\right]_0^L-\int_0^L\frac{d}{dx}\left(k\left(1+\beta\frac{x}{L}\right)^2\frac{dU}{dx}\right)\frac{dv}{dx}dx$$

Recall the original differential equation $\left(-k\frac{d}{dx}\left(\pi R_0^2\left(1+\beta\frac{x}{L}\right)^2\frac{dU}{dx}\right)=0\right)$. Since π, R_0 and k are constants, we can move them in and out of the derivative operator. Thus, our problem requires that $\frac{d}{dx}\left(k\left(1+\beta\frac{x}{L}\right)^2\frac{dU}{dx}\right)=0$, which makes the second integral resulting from integration by parts zero. We thus have:

$$\pi(U+v)=\pi(U)+k\left(1+\beta\right)^2v(L)\frac{dU}{dx}|_{x=L}-kv(0)\frac{dU}{dx}|_{x=0}+\frac{1}{2}\int_0^L k\left(1+\beta\frac{x}{L}\right)^2\left(\frac{dv}{dx}\right)^2dx$$

$$+\left(1+\beta\right)^2\eta_2 U(L)v(L)+\frac{1}{2}\left(1+\beta\right)^2\eta_2 v(L)^2-q_1v(0)-\left(1+\beta\right)^2\eta_2 U_\infty v(L)$$

Note that our first boundary condition ($k\frac{dU}{dx}|_{x=0}=-q_1$) converts the $-kv(0)\frac{dU}{dx}|_{x=0}$ term to $q_1v(0)$, which cancels out with $-q_1v(0)$. Likewise, the second boundary condition ($-k\frac{dU}{dx}|_{x=L}=\eta_2\left(U(L)-U_\infty\right)$) converts the $k\left(1+\beta\right)^2v(L)\frac{dU}{dx}|_{x=L}$ term to $\left(1+\beta\right)^2\eta_2v(L)U_\infty-\left(1+\beta\right)^2\eta_2v(L)U(L)$, which also cancels out with the $-\left(1+\beta\right)^2\eta_2U_\infty v(L)$ term and the $\left(1+\beta\right)^2\eta_2U(L)v(L)$ term. With these observations, we have:

$$\pi(U+v)=\pi(U)+\frac{1}{2}\int_0^L k\left(1+\beta\right)^2\left(\frac{dv}{dx}\right)^2dx+\frac{1}{2}\left(1+\beta\right)^2\eta_2v(L)^2$$

Now we make note of the fact that since $k>0$, $(1+\beta)^2>0$, $\left(\frac{dv}{dx}\right)^2>0$ for an uncountable number of values between 0 and L (as long as $v\neq 0$), and $v(L)\geq 0$, the remaining terms in the expression can only increase the value of $\pi(U+v)$. We thus minimize π whenever $v=0$, which implies that the function U minimizes $\pi(w)$.

$\square$

We're thus trying to find the function in the vector space spanned by $\{\psi_1,\psi_2,...,\psi_n\}$ that minimizes $\pi(w)$. Let U_{RR} be the function that minimizes π. Since U_{RR} is in the vector space spanned by $\{\psi_1,\psi_2,...,\psi_n\}$, we can express it as:

$$U_{RR}=\sum_{i=1}^n\alpha_i\psi_i$$

Plugging this representation of U_{RR} into the definition of $\pi(w)$, we get:

$$\pi(U)=\frac{1}{2}\int_0^L k\left(1+\beta\frac{x}{L}\right)^2\sum_{i=1}^n\sum_{j=1}^n\alpha_i\alpha_j\frac{d\psi_i}{dx}\frac{d\psi_j}{dx}dx+\frac{1}{2}\left(1+\beta\right)^2\eta_2\sum_{i=1}^n\sum_{j=1}^n\alpha_i\alpha_j\psi_i(L)\psi_j(L)$$

$$-q_1\sum_{i=1}^n\alpha_i\psi_i(0)-\left(1+\beta\right)^2\eta_2U_\infty\sum_{i=1}^n\alpha_i\psi_i(L)$$

Using linearity of integrals and finite sums, we can rearrange the expression to read as:

$$\pi(U)=\sum_{i=1}^n\sum_{j=1}^n\alpha_i\alpha_j\left[\frac{1}{2}\int_0^L k\left(1+\beta\frac{x}{L}\right)^2\frac{d\psi_i}{dx}\frac{d\psi_j}{dx}dx+\frac{1}{2}\left(1+\beta\right)^2\eta_2\psi_i(L)\psi_j(L)\right]$$

$$+\sum_{i=1}^n\alpha_i\left[-q_1\psi_i(0)-\left(1+\beta\right)^2\eta_2U_\infty\psi_i(L)\right]$$

Since we have chosen the basis, the only degree of freedom in the optimization that we have is the α_i's. We are thus looking for the set of α_i's that minimizes π. This can be achieved if the gradient of π with respect to $\{\alpha_1, \alpha_2, \alpha_3, ..., \alpha_n\}$ is the zero vector in R^n. We thus require that $\frac{\partial \pi}{\partial \alpha_m} = 0 \; \forall \; m \in \{1, 2, 3, ..., n\}$.

$$\frac{\partial \pi(U)}{\partial \alpha_m} = \frac{\partial}{\partial \alpha_m}\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j\left[\frac{1}{2}\int_0^L k\left(1+\beta\frac{x}{L}\right)^2\frac{d\psi_i}{dx}\frac{d\psi_j}{dx}dx + \frac{1}{2}\left(1+\beta\right)^2\eta_2\psi_i(L)\psi_j(L)\right]\right]$$

$$+\frac{\partial}{\partial \alpha_m}\left[\sum_{i=1}^{n}\alpha_i\left[-q_1\psi_i(0) - \left(1+\beta\right)^2\eta_2 U_\infty\psi_i(L)\right]\right] = 0$$

Now note that when differentiating with respect to α_m, we only care about the terms that contain α_m. In the double sum, there are the n that appear when $i = m$, and one more for each $j \neq n$. It's easy to see that the $\alpha_m\alpha_k$ terms are linear in α_m. Since there are two of each, these become $2\alpha_k$. Finally, there is one α_m^2 term which becomes $2\alpha_m$ upon differentiation. We thus get:

$$\sum_{i=1}^{n}\alpha_i\left[\int_0^L k\left(1+\beta\frac{x}{L}\right)^2\frac{d\psi_i}{dx}\frac{d\psi_m}{dx}dx + \left(1+\beta\right)^2\eta_2\psi_i(L)\psi_m(L)\right]$$

$$+\left(-q_1\psi_m(0) - \left(1+\beta\right)^2\eta_2 U_\infty\psi_m(L)\right) = 0$$

This is now a system of n equations, which we can represent as $B\vec{\alpha} = G$. Here, $\vec{\alpha} = [\alpha_1, \alpha_2, ...\alpha_n]^T$, B is the $n \times n$ matrix such that:

$$\boxed{B_{i,j} = \int_0^L k\left(1+\beta\frac{x}{L}\right)^2\frac{d\psi_i}{dx}\frac{d\psi_m}{dx}dx + \left(1+\beta\right)^2\eta_2\psi_i(L)\psi_m(L)}$$

and G is the vector in R^n defined such that:

$$\boxed{G_i = q_1\psi_m(0) + \left(1+\beta\right)^2\eta_2 U_\infty\psi_m(L)}$$

We proceed to test our program with parameters $k = 0.5\frac{W}{mK}$, $L = 1m$, $\beta = 1$, $\eta_2 = 100\frac{W}{m^2K}$, $q_1 = 100\frac{W}{m^2}$ and $U_\infty = 24^oC$.

1. Since the proposed problem is simple enough, we have an analytic function of the form:

$$U(x) = U_\infty + \frac{q_1 L}{k}\left(\frac{1+\beta+\frac{k}{\eta_2 L}}{(1+\beta)^2} - \frac{\frac{x}{L}}{1+\beta\frac{x}{L}}\right)$$

   To test the program we supply the exact analytic solution as $\psi_1(x)$, and $\psi_2(x) = x$ as an extraneous basis function. We thus expect the Rayleigh-Ritz method to output $\alpha_1 = 1$ and $\alpha = 0$. We plot the results below.

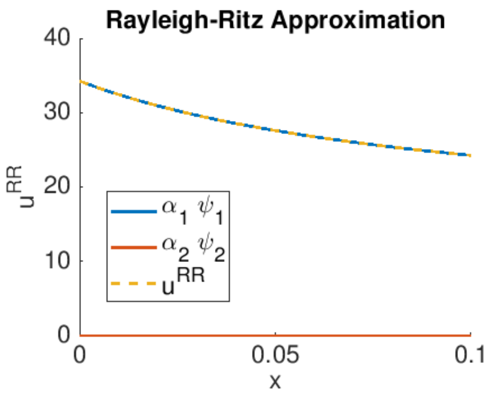


Figure 1: $\alpha_1\psi_1$, $\alpha_2\psi_2$ and their sum plotted, where ψ_1 is the analytic solution to the problem and $\psi_2(x) = x$. We get the expected result $\alpha_1 = 1$ and $\alpha = 0$.

2. We now test the method with $\psi_0(x) = 1$, $\psi_1(x) = x$ and $\psi_2(x) = x^2$. The next three figures illustrate what happens if we only use ψ_1, only use ψ_2 and ψ_3, and use all 3 functions respectively. We see that once we use three basis functions, our approximation looks remarkably similar to the analytic solution.

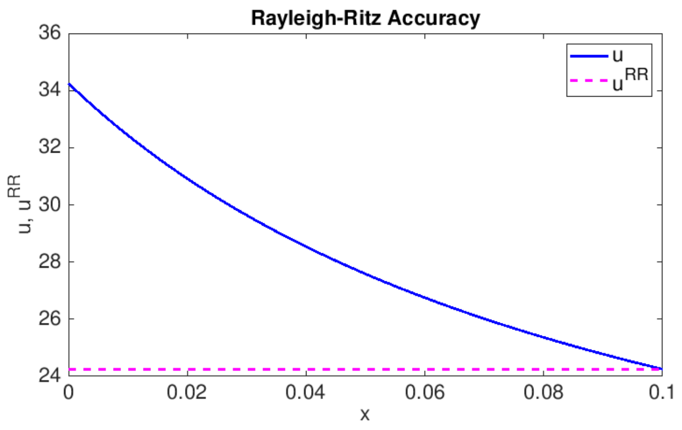


Figure 2: The approximation U^{RR} compared to the analytic solution U, when the basis function is simply $\psi_1(x) = 1$.

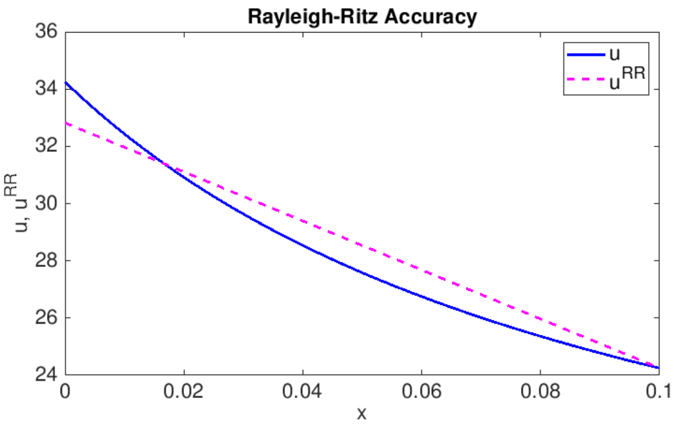


Figure 3: The approximation U^{RR} compared to the analytic solution U, when the basis functions are $\psi_1(x) = 1$ and $\psi_2(x) = x$.

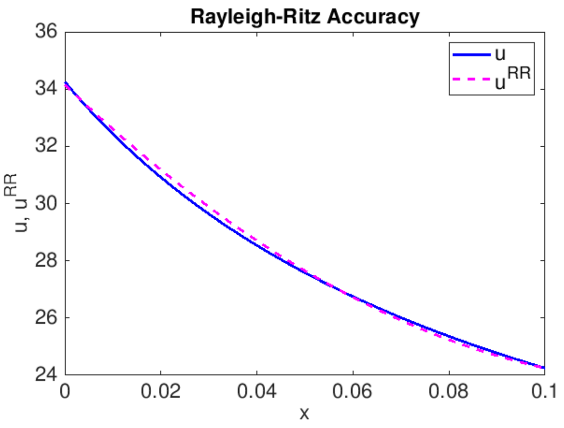


Figure 4: The approximation U^{RR} compared to the analytic solution U, when the basis functions are $\psi_1(x) = 1$, $\psi_2(x) = x$ and $\psi_3(x) = x^2$.

3. We've mathematically shown that if U solves our boundary value problem, then it must minimize the functional π. Furthermore, when implementing the minimization in code, we were able to retrieve the exact solution by forcing the analytic solution to be in the vector space of functions over which we are seeking approximations.

   Though we weren't able to extract the exact solution by minimizing over a vector space spanned by polynomials, this should not be surprising since the analytic solution is not a polynomial as long as $\beta \neq 0$. Therefore, there is no reason to expect to be able to find a non-polynomial function the exact solution in a vector space of polynomials. Nevertheless, the polynomial curves we retrieved looked strikingly similar to the analytic solution.

   With these observations in mind, we have an decent amount of evidence suggesting that our code is working adequately.

4. We now test the Rayleigh-Ritz method on the polynomial vector space of functions outlined above, for various values of β. We test $\beta \in \{0, 1, 5, 10, 100, 1000\}$.

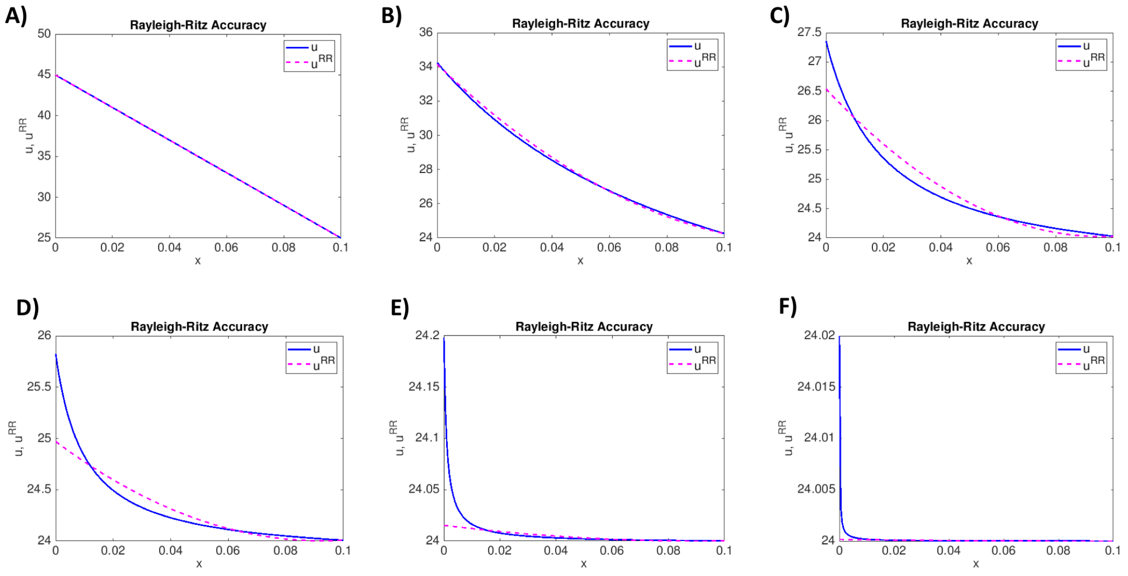


Figure 5: The Rayleigh-Ritz solution approximated by $\psi_1(x) = 1$, $\psi_2(x) = x$ and $\psi_3(x) = x^2$, plotted against the analytic solutions for different values of β. **Panel A** corresponds to $\beta = 0$ - the solution is exact because when $\beta = 0$, the analytic solution is linear, which is in the space of polynomial functions spanned by ψ_1, ψ_2 and ψ_3. **Panel B** corresponds to $\beta = 1$ (the same as Figure 2). **Panel C** corresponds to $\beta = 5$. **Panel D** corresponds to $\beta = 10$. **Panel E** corresponds to $\beta = 100$. **Panel F** corresponds to $\beta = 1000$.

We can see that the solution is fairly accurate for small β, since as discussed, the $\beta = 0$ analytic solution is actually in the space of solutions over which we are minimizing π. Thus, the lower the β, the closer we get to a solution that is in the space of viable approximations. As β increases, the approximation gets worse around the $u = 0$ point gets worse, but for larger values of x it doesn't change much. To make sense of this, it's worth looking at the analytic solution:

$$U(x) = U_\infty + \frac{q_1 L}{k}\left(\frac{1+\beta+\frac{k}{\eta_2 L}}{(1+\beta)^2} - \frac{\frac{x}{L}}{1+\beta\frac{x}{L}}\right)$$

More notably, its derivative tells us that

$$U'(x) = -\frac{q_1 L}{k}\frac{\frac{1}{L}(1+\beta\frac{x}{L}) - \frac{x}{L}\frac{\beta}{L}}{(1+\beta\frac{x}{L})^2} = -\frac{q_1}{k(1+\beta\frac{x}{L})^2}$$

At $x = 0$, the derivative will simply be $-\frac{q_1}{k}$ as required by the boundary condition. However, note that as $\beta \to \infty$, varying x from 0 to L causes the derivative to drop dramatically to zero. Since Parabolas are symmetric, the dramatic drop to zero can't be adequately approximated without having it dramatically rise again as soon as the lowest point in the parabola is hit; therefore, it's very difficult to capture the dramatic drop to zero slope with functions spanned by ψ_1, ψ_2 and ψ_3.

However, at larger values of x, the function seems to behave like a constant since the derivative is approximately zero, and constant functions are in the space of functions over which we seek to minimize π. As a result, the function doesn't diverge much from the approximation away from the $x = 0$ point.

Nevertheless, it is worthy to take note of our method's sensitivity to the parameter β. We now proceed to test the Rayleigh-Ritz method on another model.

2.2 Right-Cylinder Thermal Fin

Our next model approximates a skillet's handle as a right-cylinder with a cross sectional area of A_{cs} and lateral perimeter P_{cs} with a prescribed temperature at the boundary of the handle and the pan ($x = 0$) and no heat transfer occurring at its tip ($x = L$). In this case, the heat equation collapses to Eq. 3 [Patera, 2019o].

$$-kA_{cs}\frac{d^2U}{dx^2} + \eta_3 P_{cs} U = \eta_3 P_{cs} U_\infty \tag{3}$$

Subject to:

- $U(0) = U_{\Gamma_1}$
- $-k\frac{dU}{dx}|_{x=L} = 0$

Here, k is the thermal conductivity of the skillet's material, η_3 is the effective convective heat transfer between the pan and the air, U_∞ is the ambient temperature, and U_{Γ_1} is the temperature at the boundary of the skillet handle and the pan.

We will now propose the following functional π and prove (in a less detailed fashion than before) that the solution to the problem above minimizes this functional [Patera, 2019o].

$$\boxed{\pi(w) = \frac{1}{2}\int_0^L \left[kA_{cs}\left(\frac{dw}{dx}\right)^2 + \eta_3 P_{cs} w^2(x)\right]dx - \int_0^L \eta_3 P_{cs} U_\infty w(x) dx}$$

Consider a function of the form $U+v$, where U is the solution to our problem.

$$\pi(U+v) = \frac{1}{2}\int_0^L \left[kA_{cs}\left(\frac{dU}{dx}+\frac{dv}{dx}\right)^2 + \eta_3 P_{cs}(U(x)+v(x))^2\right]dx - \int_0^L \eta_3 P_{cs} U_\infty U(x) dx - \int_0^L \eta_3 P_{cs} U_\infty v(x) dx$$

$$= \frac{1}{2}\int_0^L \left[kA_{cs}\left(\frac{dU}{dx}\right)^2 + \eta_3 P_{cs} U^2(x)\right]dx + \int_0^L kA_{cs}\frac{dU}{dx}\frac{dv}{dx}dx + \frac{1}{2}\int_0^L kA_{cs}\left(\frac{dv}{dx}\right)^2 dx$$

$$+ \int_0^L \eta_3 P_{cs} U(x) v(x) dx - \int_0^L \eta_3 P_{cs} U_\infty U(x) dx + \frac{1}{2}\int_0^L \eta_3 P_{cs} v^2(x) dx - \int_0^L \eta_3 P_{cs} U_\infty v(x) dx$$

$$= \pi(U) + \left[kA_{cs}\frac{dU}{dx}v\right]_0^L - \int_0^L kA_{cs}\frac{d^2U}{dx^2}v(x)dx + \frac{1}{2}\int_0^L kA_{cs}\left(\frac{dv}{dx}\right)^2 dx$$

$$+ \int_0^L \eta_3 P_{cs} U(x) v(x) dx + \frac{1}{2}\int_0^L \eta_3 P_{cs} v^2(x) dx - \int_0^L \eta_3 P_{cs} U_\infty v(x) dx$$

$$= \pi(U) + \left[kA_{cs}\frac{dU}{dx}v\right]_0^L + \int_0^L \left(-kA_{cs}\frac{d^2U}{dx^2} + \eta_3 P_{cs}(U(x) - U_\infty)\right) v(x) dx$$

$$+ \frac{1}{2}\int_0^L kA_{cs}\left(\frac{dv}{dx}\right)^2 dx + \frac{1}{2}\int_0^L \eta_3 P_{cs} v^2(x) dx$$

$$= \pi(U) + kA_{cs}v(L)\frac{dU}{dx}|_{x=L} - kA_{cs}v(0)\frac{dU}{dx}|_{x=0} + \frac{1}{2}\int_0^L kA_{cs}\left(\frac{dv}{dx}\right)^2 dx + \frac{1}{2}\int_0^L \eta_3 P_{cs} v^2(x) dx$$

$$= \pi(U) - kA_{cs}v(0)\frac{dU}{dx}|_{x=0} + \frac{1}{2}\int_0^L kA_{cs}\left(\frac{dv}{dx}\right)^2 dx + \frac{1}{2}\int_0^L \eta_3 P_{cs} v^2(x) dx$$

We finally note that if we limit the vector space to functions in it that equal u_{Γ_1} at $x = 0$, then $U(0)+v(0) = u_{\Gamma_1}$, which via our first initial condition implies that $v(0) = 0$. Hence:

$$\pi(U+v) = \pi(U) + \frac{1}{2}\int_0^L kA_{cs}\left(\frac{dv}{dx}\right)^2 dx + \frac{1}{2}\int_0^L \eta_3 P_{cs} v^2(x)dx$$

Clearly, the last two terms are strictly positive if $v \neq 0$. Thus, U minimizes π. We now show (again, with less rigor than before) how to set up a linear system to minimize this new functional. We write U in terms of our basis functions $U(x) = \sum_{i=0}^{n} \alpha_i\psi_i(x)$ and compute the functional.

$$\pi(U) = \frac{1}{2}\int_0^L \left[kA_{cs}\sum_{i=0}^{n}\sum_{j=0}^{n}\alpha_i\alpha_j\frac{d\psi_j}{dx}\frac{d\psi_i}{dx} + \eta_3 P_{cs}\sum_{i=0}^{n}\sum_{j=0}^{n}\alpha_i\alpha_j\psi_i(x)\psi_j(x)\right]dx$$

$$-\int_0^L \eta_3 P_{cs} U_\infty \sum_{i=0}^{n}\alpha_i\psi_i(x)dx$$

We know we must set the gradient of π to zero, but we need to handle the Dirichlet condition with a bit of care. To ensure that we only consider functions in the vector space with $f(0) = U_{\Gamma_1}$, we select ψ_i such that $\psi_i(0) = 0$ $\forall\ i \in \{1, 2, 3, ..., n\}$, and $\psi_0(0) = 1$. Hence, we know that $\alpha_0 = U_{\Gamma_1}$ for any of our candidate functions. Keeping in mind that in each term in the sum each α_0 pair appears twice except for the α_0^2 term which only appears once, we can get a modified expression for $\pi(U)$ as follows:

$$\pi(U) = \frac{1}{2}\int_0^L \left[kA_{cs}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j\frac{d\psi_j}{dx}\frac{d\psi_i}{dx} + \eta_3 P_{cs}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j\psi_i(x)\psi_j(x)\right]dx$$

$$+\int_0^L \left[kA_{cs}\alpha_0\sum_{i=1}^{n}\alpha_i\frac{d\psi_0}{dx}\frac{d\psi_i}{dx} + \eta_3 P_{cs}\alpha_0\sum_{i=1}^{n}\alpha_i\psi_i(x)\psi_0(x)\right]dx + \frac{1}{2}\int_0^L kA_{cs}\alpha_0^2\left(\frac{d\psi_0}{dx}\right)^2 dx$$

$$+\frac{1}{2}\int_0^L \eta_3 P_{cs}\left(\alpha_0^2\psi_0^2(x) - U_\infty\alpha_0\psi_0\right)dx - \int_0^L \eta_3 P_{cs}U_\infty\sum_{i=1}^{n}\alpha_i\psi_i(x)dx$$

Noting that α_0 is a constant, we can now differentiate with respect to α_i $\forall$ $i \in \{1, 2, 3, ..., n\}$. We thus get:

$$\frac{\partial\pi}{\partial\alpha_k} = \sum_{i=1}^{n}\alpha_i\left[\int_0^L\left(kA_{cs}\frac{d\psi_i}{dx}\frac{d\psi_k}{dx} + \eta_3 P_{cs}\psi_i(x)\psi_k(x)\right)dx\right] - \int_0^L \eta_3 P_{cs}U_\infty\psi_k(x)dx$$

$$+\int_0^L\left[kA_{cs}\alpha_0\frac{d\psi_0}{dx}\frac{d\psi_k}{dx} + \eta_3 P_{cs}\alpha_0\psi_k(x)\psi_0(x)\right]dx$$

To get a minimum we want $\frac{\partial \pi}{\partial \alpha_k} = 0 \; \forall \; k \in \{1, 2, 3, ..., n\}$. We thus get the matrix equation $B\vec{\alpha} = G$, where $\vec{\alpha} = [\alpha_1, \alpha_2, ..., \alpha_n]^T$, B is the $n \times n$ matrix such that:

$$\boxed{B_{i,j} = \int_0^L \left(kA_{cs} \frac{d\psi_i}{dx} \frac{d\psi_j}{dx} + \eta_3 P_{cs} \psi_i(x)\psi_j(x) \right) dx}$$

and G is the vector in R^n defined such that:

$$\boxed{G_i = \int_0^L \left[\eta_3 P_{cs} \left(U_\infty \psi_i(x) - U_{\Gamma_1} \psi_i(x)\psi_0(x) \right) - kA_{cs} U_{\Gamma_1} \frac{d\psi_0}{dx} \frac{d\psi_i}{dx} \right] dx}$$

We now test the algorithm with values of $k = 50\frac{W}{mK}$, $L = 0.05m$, $U_\infty = 24^oC$, $U_{\Gamma_1} = 50^oC$, $A_{cs} = 0.0001m^2$, $P_{cs} = 0.04m$, and $\eta_3 = 100\frac{W}{m^2K}$.

1. Like our previous model, the boundary value problem posed is simple enough that it can be solved analytically, giving us:

$$U(x) = U_\infty + (U_{\Gamma_1} - U_\infty) \frac{\cosh\left(\sqrt{\mu}\left(1 - \frac{x}{L}\right)\right)}{\cosh(\sqrt{\mu})}$$

   where $\mu = \eta_3 P_{cs} L^2/(kA_{cs})$. We thus proceed to provide as basis functions ψ_0 which is the analytic solution normalized by U_{Γ_1} (which is necessary since we're forcing α_0 to be U_{Γ_1}), and $\psi_1(x) = x$. We thus expect the Rayleigh-Ritz method to output $\alpha_1 = U_{\Gamma_1}$ and $\alpha_2 = 0$. We plot the results below.

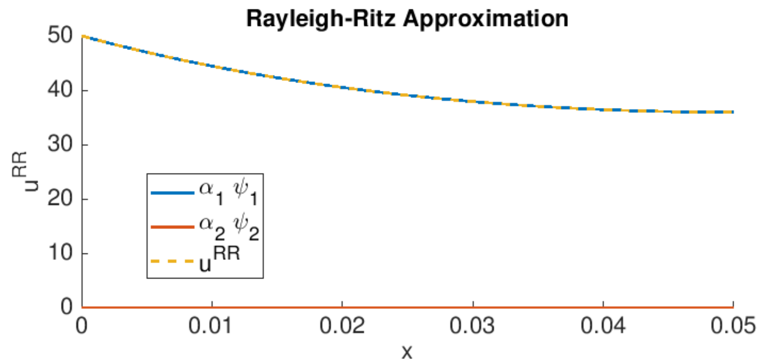


Figure 6: $\alpha_0\psi_0$, $\alpha_1\psi_1$ and their sum plotted, where ψ_1 is the analytic solution to the problem normalized by U_{Γ_1} and $\psi_2(x) = x$. We get the expected result $\alpha_1 = 50 = U_{\Gamma_1}$ and $\alpha = 0$.

2. We now test the method with $\psi_0(x) = 1$, $\psi_1(x) = x$ and $\psi_2(x) = x^2$. The next three figures illustrate what happens if we only use ψ_0, only use ψ_1 and ψ_0, and use all 3 functions respectively. We see that once we use three basis functions, our approximation looks remarkably similar to the analytic solution.

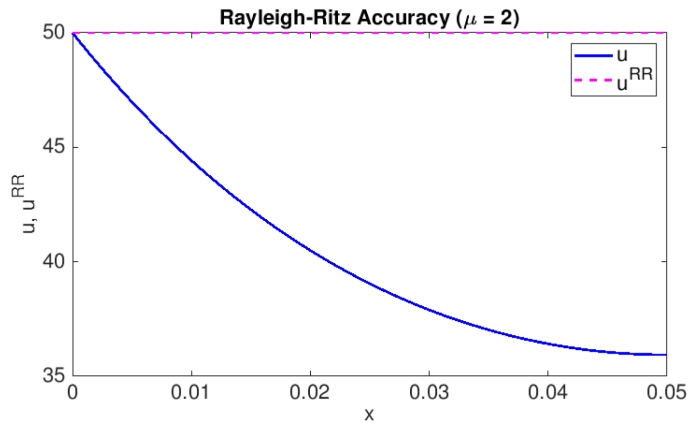


Figure 7: The approximation U^{RR} compared to the analytic solution U, when the basis functions are $\psi_0(x) = 1$ and $\psi_1(x) = x$.

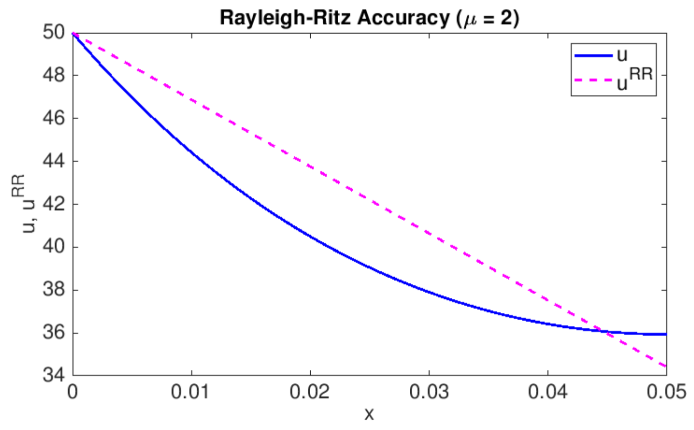


Figure 8: The approximation U^{RR} compared to the analytic solution U, when the basis functions are $\psi_0(x) = 1$ and $\psi_1(x) = x$.

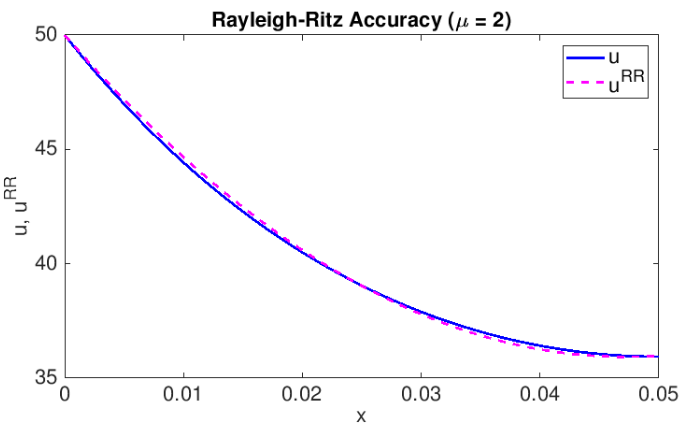


Figure 9: The approximation U^{RR} compared to the analytic solution U, when the basis functions are $\psi_0(x) = 1$, $\psi_1(x) = x$ and $\psi_2(x) = x^2$.

3. Once again, we've mathematically shown that if U solves our boundary value problem, then it must minimize the functional π. Furthermore, when implementing the minimization in code, we were able to retrieve the exact solution by forcing the analytic solution to be in the vector space of functions over which we are seeking approximations.

   Though we weren't able to extract the exact solution by minimizing over a vector space spanned by polynomials, this should not be surprising since the analytic solution is not a polynomial as long as $\mu \neq 0$ (in which case it'll be a constant function). Therefore, there is no reason to expect to be able to find a non-polynomial function the exact solution in a vector space of polynomials. Nevertheless, the polynomial curves we retrieved looked strikingly similar to the analytic solution.

   With these observations in mind, we have an decent amount of evidence suggesting that our code is working adequately.

4. We now test our algorithm's sensitivity to the parameter μ. The only parameter that appears in μ and was not specified is η_3. As a result, we will select $\eta_3 \in \{1, 80, 1000\}\frac{W}{m^2K}$. These values cover the different regimes of natural convection and correspond to values of μ of 0.02, 1.6 and 20, respectively. We see the results in a Figure below.

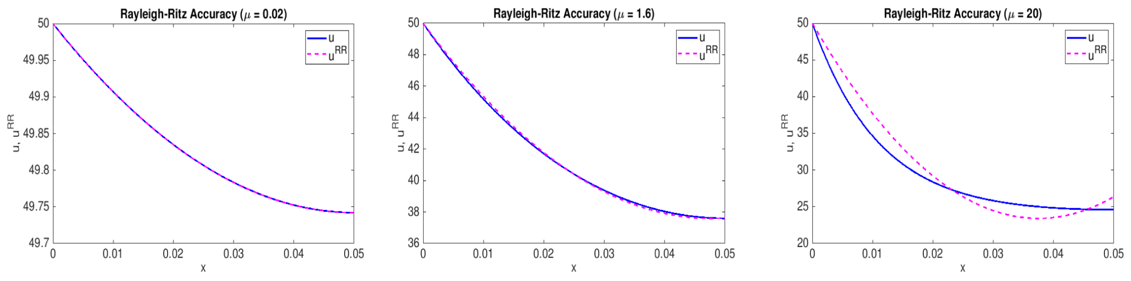


Figure 10: The Rayleigh-Ritz solution approximated by $\psi_1(x) = 1$, $\psi_2(x) = x$ and $\psi_3(x) = x^2$, plotted against the analytic solutions for different values of μ. The **Left Panel** corresponds to $\mu = 0.02$. The **Middle Panel** corresponds to $\mu = 1.6$. The **Right Panel** corresponds to $\mu = 20$.

We observe that as μ increases, the approximation becomes less accurate. This is because the derivative at $x = 0$ scales like $-\sqrt{\mu}$. As a result, larger values of μ generate a steeper descent from U_{Γ_1} to U_∞, which becomes increasingly difficult for a parabola to approximate, since as discussed before, a steep descent to a zone of ≈ 0 slope would signify a rapid symmetric ascent.

3 The One Dimensional Finite Element Method

Previously, we explored the general Rayleigh-Ritz method. Now, instead of considering optimization over an arbitrary vector space of functions, like we have been doing, we will consider two particular vector spaces; this, will give us the Finite Element Method in 1D.

We're trying to approximate a function $u : [0, L] \to R$, so if we're given a mesh, or a collection of closed intervals called elements that only intersect at the endpoints, but whose union is exactly $[0, L]$, we can consider the two vector spaces of piece-wise functions that are (1) linear and (2) quadratic on each element of the given mesh [Patera, 2019f].

We now revisit the functional π we introduced in the previous chapter, and show why this particular vector space of functions is computationally favorable. In the previous chapter, we considered a specific functional π for each boundary value problem. We now show a general functional form for any heat equation problem subject to Neumann-Robin boundary conditions. That is:

$$-\frac{d}{dx}\left(k(x)\frac{dU}{dx}\right) + \mu(x)U = f_\Omega(x) \tag{4}$$

Subject to:

- $k\frac{dU}{dx}|_{x=0} = \gamma_1 U(0) - f_{\Gamma_1}$

- $-k\frac{dU}{dx}|_{x=L} = \gamma_2 U(L) - f_{\Gamma_2}$

We propose the functional:

$$\boxed{\pi(w) = \frac{1}{2}\int_0^L \left[k(x)\left(\frac{dw}{dx}\right)^2 + \mu(x)w^2\right]dx + \frac{1}{2}\left(\gamma_1 w^2(0) + \gamma_2 w^2(L)\right) - \int_0^L f_\Omega(x)w dx - w(0)f_{\Gamma_1} - w(L)f_{\Gamma_2}}$$

We proceed to show that the solution to the posed problem indeed minimizes the functional proposed above [Patera, 2019h].

Proof. Like before, we consider $w = U + v$, where U is the solution to the problem.

$$\pi(U{+}w) = \frac{1}{2}\int_0^L \left[k(x)\left\{\left(\frac{dU}{dx}\right)^2 + 2\left(\frac{dU}{dx}\right)\left(\frac{dv}{dx}\right) + \left(\frac{dv}{dx}\right)^2\right\} + \mu(x)\left\{U^2 + 2Uv + v^2\right\}\right]dx$$

$$+\frac{1}{2}\left(\gamma_1\left\{U^2(0) + 2U(0)v(0) + v^2(0)\right\} + \gamma_2\left\{U^2(L) + 2U(L)v(L) + v^2(L)\right\}\right)$$

$$-\int_0^L f_\Omega(x)\left(U+v\right)dx - U(0)f_{\Gamma_1} - v(0)f_{\Gamma_1} - U(L)f_{\Gamma_2} - v(L)f_{\Gamma_2}$$

Collecting the $\frac{dU}{dx}^2$, U^2, $U^2(0)$, $U^2(L)$, $f_\Gamma U$, $U(0)$ and $U(L)$ terms we get $\pi(U)$.

$$\pi(U{+}w) = \pi(U){+}\frac{1}{2}\int_0^L \left[k(x)\left\{2\left(\frac{dU}{dx}\right)\left(\frac{dv}{dx}\right) + \left(\frac{dv}{dx}\right)^2\right\} + \mu(x)\left\{Uv + v^2\right\}\right]dx$$

$$+\frac{1}{2}\left(\gamma_1\left\{2U(0)v(0) + v^2(0)\right\} + \gamma_2\left\{2U(L)v(L) + v^2(L)\right\}\right)$$

$$-\int_0^L f_\Omega(x)v dx - v(0)f_{\Gamma_1} - v(L)f_{\Gamma_2}$$

We now integrate the $\frac{dU}{dx}\frac{dv}{dx}$ term of the integral with integration by parts.

$$\int_0^L k(x)\frac{dU}{dx}\frac{dv}{dx}dx = \left[k(x)\frac{dU}{dx}v\right]_0^L - \int_0^L \frac{d}{dx}\left(k(x)\frac{dU}{dx}\right)v dx$$

Invoking boundary conditions of our problem, we see that $\left(\kappa(x)\frac{dU}{dx}\right)_{x=L}$ is just $-\gamma_2 U(L) + f_{\Gamma_2}$ and $\left(\kappa(x)\frac{dU}{dx}\right)_{x=0}$ is $\gamma_1 U(L) - f_{\Gamma_1}$. Hence:

$$\int_0^L k(x)\frac{dU}{dx}\frac{dv}{dx}dx = -\gamma_2 U(L)v(L)+f_{\Gamma_2}v(L)-\gamma_1 U(0)v(0)+f_{\Gamma_1}v(0)-\int_0^L \frac{d}{dx}\left(k(x)\frac{dU}{dx}\right)vdx$$

When combining it with the rest of the equation, we see that all the boundary condition terms cancel out with other terms in the expression. Hence:

$$\pi(U+w) = \pi(U) + \frac{1}{2}\int_0^L \left[k(x)\left(\frac{dv}{dx}\right)^2 + \mu(x)\left\{v^2\right\}\right]dx$$

$$+\frac{1}{2}\left(\gamma_1\left\{v^2(0)\right\}+\gamma_2\left\{v^2(L)\right\}\right) - \int_0^L f_\Omega(x)vdx$$

$$+\int_0^L v\left\{-\frac{d}{dx}\left(k(x)\frac{dU}{dx}\right)+\mu(x)U\right\}dx$$

Notice that the term in curly braces in the last integral is simply f_Ω since our differential equation dictates so. This makes the two integral terms cancel out. We then note that the remaining terms are all square quantities of real valued functions multiplied by positive values. Hence, adding any non-zero function to U can only increase the value of π. Hence U minimizes π. $\square$

Now that we have a more general functional to minimize, we see some advantages of selecting vector spaces of piece-wise linear and piece-wise quadratic functions over a given mesh.

Let us consider the piece-wise linear case first. Given a mesh with n elements numbered $\{T_1, T_2, ..., T_n\}$, we define $n+1$ basis functions $\{\phi_1, \phi_2, ..., \phi_{n+1}\}$. If $i \neq n+1$, we have:

$$\phi_i = \begin{cases} 1+\frac{1}{h_i}(x_{i,l}-x) & x \in T_i \\ 1+\frac{1}{h_i}(x-x_{i,l}) & x \in T_{i-1} \quad \text{(unless i=1)} \\ 0 & \text{elsewhere} \end{cases}$$

where $x_{i,l}$ is the left endpoint of T_i and h_i is the length of T_i [Patera, 2019h]. Otherwise, we have:

$$\phi_{n+1} = \begin{cases} 1+\frac{1}{h_n}(x-L) & x \in T_n \\ 0 & \text{elsewhere} \end{cases}$$

where L is the right endpoint of the domain [Patera, 2019h]. In practice, these functions look like triangles whose base extends two elements in the mesh - unless it is ϕ_1 and ϕ_{n+1}, which only span one element [Patera, 2019g]. Let us now express our approximation U_{FE} for U in terms of these functions.

$$U_{FE} = \sum_{i=1}^{n+1} \alpha_i \phi_i$$

Plugging the expression into π gives us:

$$\pi(U_{FE}) = \frac{1}{2}\int_0^L \left[k(x)\sum_{i=1}^{n+1}\sum_{j=1}^{n+1}\alpha_i\alpha_j\frac{d\phi_i}{dx}\frac{d\phi_j}{dx} + \mu(x)\sum_{i=1}^{n+1}\sum_{j=1}^{n+1}\alpha_i\alpha_j\phi_i\phi_j\right]dx$$

$$+\frac{1}{2}\gamma_1\sum_{i=1}^{n+1}\sum_{j=1}^{n+1}\alpha_i\alpha_j\phi_i(0)\phi_j(0) + \frac{1}{2}\gamma_2\sum_{i=1}^{n+1}\sum_{j=1}^{n+1}\alpha_i\alpha_j\phi_i(L)\phi_j(L) - \int_0^L f_\Omega\sum_{i=0}^{n+1}\alpha_i\phi_i dx$$

$$-f_{\Gamma_1}\sum_{i=0}^{n+1}\alpha_i\phi_i(0) - f_{\Gamma_2}\sum_{i=0}^{n+1}\alpha_i\phi_i(L)$$

Like in the previous chapter, we find the gradient of π with respect to the coordinates $\{\alpha_i\}_{i=1}^{n+1}$. Using the same arguments as before we get that:

$$\frac{\partial\pi}{\partial\alpha_k} = \sum_{i=1}^{n+1}\alpha_i\left\{\int_0^L\left[k(x)\frac{d\phi_i}{dx}\frac{d\phi_k}{dx} + \mu(x)\phi_i\phi_k\right]dx + \gamma_1\alpha_i\phi_i(0)\phi_k(0) + \gamma_2\alpha_i\phi_i(L)\phi_k(L)\right\}$$

$$-\int_0^L f_\Omega\phi_k dx - f_{\Gamma_1}\phi_k(0) - f_{\Gamma_2}\phi_K(L)$$

Since this holds for every k and we want all of these to equal zero when optimizing, we have a system of $n+1$ linear equations, which we can represent as $A\vec{\alpha} = F$, where:

$$\boxed{A_{i,k} = \int_0^L\left[k(x)\frac{d\phi_i}{dx}\frac{d\phi_k}{dx} + \mu(x)\phi_i\phi_k\right]dx + \gamma_1\alpha_i\phi_i(0)\phi_k(0) + \gamma_2\alpha_i\phi_i(L)\phi_k(L)}$$

And:

$$\boxed{F_k = \int_0^L f_\Omega\phi_k dx + f_{\Gamma_1}\phi_k(0) + f_{\Gamma_2}\phi_K(L)}$$

Here lies the advantage of using the basis functions defined above. First, we note that by design, $\phi_i(0) = 0$ for all $i \neq 1$ and $\phi_{n+1}(L) = 0$ for all $i \neq n+1$. Thus, we can ignore the $\gamma_1\alpha_i\phi_i(0)\phi_k(0)$ term for all entries except $A_{1,1}$ and we can ignore the $\gamma_2\alpha_i\phi_i(L)\phi_k(L)$ term for all entries except $A_{n+1,n+1}$.

Furthermore, we note that by design, the support of ϕ_i only overlaps with the support of ϕ_{i-1}, ϕ_{i+1} and itself. The same is true of the derivative. As a result, $A_{i,k} = 0$ if $k \notin \{i-1, i, i+1\}$. This implies that A is a tri-diagonal matrix, which means we can solve $A\vec{\alpha_i} = F$ in $O(n)$ steps instead of $O(n^3)$, which would be the case for any arbitrary matrix [Patera, 2019g].

Furthermore, note that if ϕ_i overlaps with ϕ_k and $i \neq k$, then

$$\frac{d\phi_i}{dx}\frac{d\phi_k}{dx} = -\frac{1}{h_i h_k}$$

and if $i = k$, then

$$\frac{d\phi_i}{dx}\frac{d\phi_i}{dx} = \frac{1}{h_i^2}$$

.

Though the ϕ_i products don't simplify as thoroughly, they're still much simpler to handle when performing numerical quadrature [Patera, 2019g].

If we consider quadratic piece-wise functions, then we must now include mesh points in between the element boundaries of the given mesh. We have thus meshed the domain into $2n$ elements, giving us $2n + 1$ basis functions [Patera, 2019i]. Here, it is convenient to refer to ϕ_i as the function that is 1 at node i, where nodes 1 through n are the left endpoints of the original mesh, node $n+1$ is the right endpoint of the entire domain, and nodes $n+2$ through $2n+1$ are the new "midpoint" nodes. We define our basis functions as [Patera, 2019i]:

$$\phi_i = \begin{cases} (x - x_{i-1})(x - x_{i+n}) & x \in T_{i-1} \quad \text{(unless i=1)} \\ (x - x_{i+1})(x - x_{i+n+1}) & x \in T_i \\ 0 & \text{elsewhere} \end{cases}$$

if $i < n+1$. If $i = n+1$, we have:

$$\phi_{n+1} = \begin{cases} (x - x_n)(x - x_{2n+1}) & x \in T_n \\ 0 & \text{elsewhere} \end{cases}$$

And if $i > n+1$, we have:

$$\phi_i = \begin{cases} -(x - x_{i-n-1})(x - x_{i-n}) & x \in T_{i-n+1} \\ 0 & \text{elsewhere} \end{cases}$$

Though a bit more complicated, the A matrix and F vector are defined in the exact same way. All we must handle carefully is the $\phi_i(L)$ term (it still is only non zero when $i = n+1$, but this is now in the middle of the matrix/vector whereas it was the last entry in the piece-wise linear case) [Patera, 2019i]. We also note that there is more overlap between the support of the basis functions, so the matrix will now become penta-diagonal instead of tri-diagonal. Luckily, penta-diagonal systems can still be solved in $O(n)$ steps [Patera, 2019g].

We still haven't really discussed what to do in the case of Dirichlet boundary conditions, but the procedure is quite similar to the one in the previous chapter, so we won't repeat it in detail. We let α_1 or α_{n+1} equal whatever value is specified by at $U(0)$ or $U(L)$ respectively, and incorporate these terms into the F vector [Patera, 2019g]. This reduces the number of variables in the system by the number of Dirichlet boundary conditions. We now test our newly developed method on the two heat transfer models of our previous chapter and a new one that we will introduce now.

3.1 The Wall Model

We introduce a new case study for the heat equation to test with our newly developed Finite Element Method. Since the method hasn't been developed enough to handle multiple dimensions (including time), we will consider the steady-state operations of building wall during the winter exposed to an exterior temperature u_{out} and a thermostat regulated interior temperature of u_{in}.

Let $x = 0$ denote the exterior of the wall and $x = L$ the interior. The steady-state heat equation gives us:

$$-\frac{d}{dx}\left(k\frac{dU}{dx}\right) = f \tag{5}$$

Assuming that the wall has a constant thermal conductivity k and that there are no sources of heat in the wall (i.e. $f = 0$), we get the simple equation:

$$\frac{d^2U}{dx^2} = 0 \tag{6}$$

This ordinary differential equation has the general solution $U(x) = C_1x + C_0$. To specify the values of C_1 and C_0, we must invoke boundary conditions. For this, we consider the convective heat transfer occurring at the interior and exterior of the wall. Invoking convective heat transfer at the interior and exterior of the wall, we get:

- $k\frac{dU}{dx}|_{x=0} = \eta_1(U(0) - U_{\text{out}})$
- $-k\frac{dU}{dx}|_{x=L} = \eta_2(U(L) - U_{\text{in}})$

With this information, we can solve for C_1 and C_0. First we note that $\frac{dU}{dx} = C_1$ everywhere in $[0, L]$. Hence, we get the equations:

- $kC_1 = \eta_1(C_0 - U_{\text{out}})$
- $-kC_1 = \eta_2(C_1L + C_0 - U_{\text{in}})$

Solving this system for C_0 and C_1, gives us the analytic solution:

$$U(x) = \left(\frac{U_{\text{in}} - U_{\text{out}}}{L + \left(\frac{1}{\eta_1} + \frac{1}{\eta_2}\right)k}\right)x + \frac{k}{\eta_2}\left(\frac{U_{\text{in}} - U_{\text{out}}}{L + \left(\frac{1}{\eta_1} + \frac{1}{\eta_2}\right)k}\right) + U_{\text{out}} \tag{7}$$

3.2 Experiments and Convergence Rates

We first test our newly developed method on the Right-Cylinder Thermal Fin model on meshes with different number of elements.

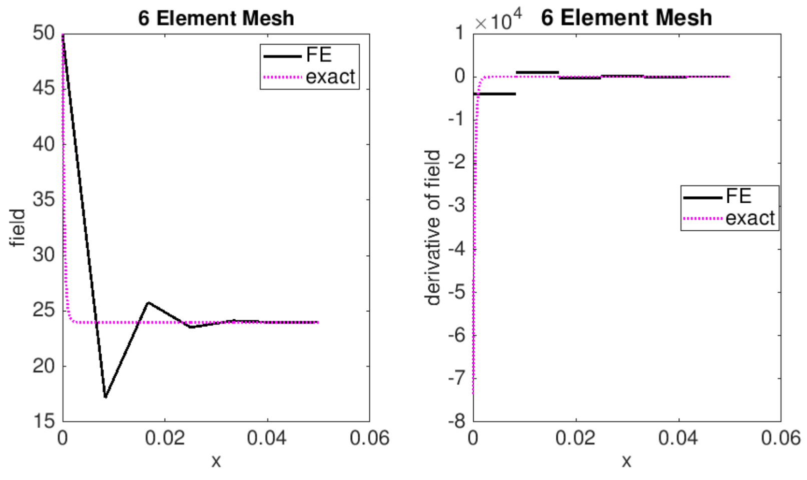


Figure 11: The exact solution and derivative plotted against the finite element solution for a mesh with six elements.

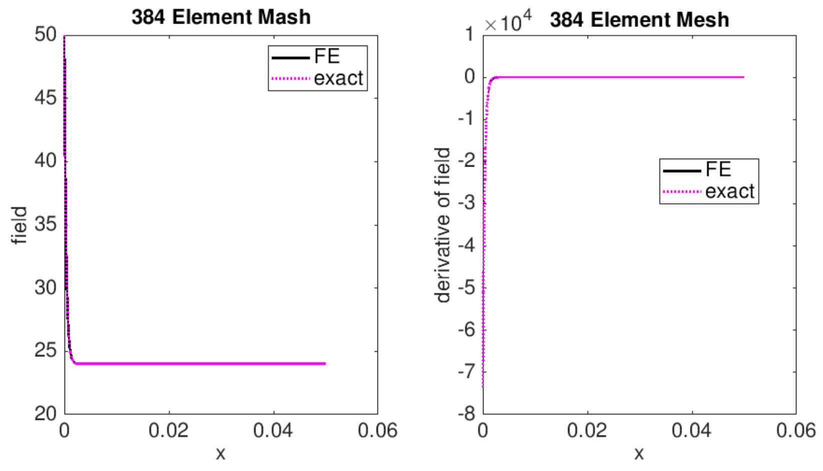


Figure 12: The exact solution and derivative plotted against the finite element solution for a mesh with 384 elements.

As we can see from these two plots, the finite element approximations qualitatively match the shapes of the analytic solutions. Furthermore, we note that for a mesh with more elements, the approximation looks more similar to the analytic solution to the point that at 384 elements, the FEM approximation and the exact solution are basically indistinguishable from each other.

To make these observations more rigorous, we take a look at a few norms and how their error scales with the largest element in the mesh. In particular, we consider:

$$||w||_{H_\Omega} = \left(\int_0^L \left\{ \left(\frac{dw}{dx} \right)^2 + \frac{1}{L^2} w^2 \right\} dx \right)^{\frac{1}{2}}$$

$$||w||_{L^2} = \left(\int_0^L w^2 dx \right)^{\frac{1}{2}}$$

$$||w||_{L^\infty} = \sup_{x \in \Omega} |w|$$

Depending on whether we use piece-wise linear basis or the piece-wise quadratic basis, we get different bounds on the asymptotic behavior of the error in these bounds [Patera, 2019a]. In particular, we have:

$$||u - u_{FE}||_{H_\Omega} = O(h^p)$$
$$||u - u_{FE}||_{L^2} = O(h^{p+1})$$
$$||u - u_{FE}||_{L^\infty} = O(h^{p+\frac{1}{2}})$$

where $p = 1$ in the piece-wise linear case and $p = 2$ in the piece-wise quadratic case [Patera, 2019a]. We test these theoretical asymptotic limits for various meshes with a maximum element length of $h_{\max}$ by plotting $h_{\max}$ against the error in a log-log plot.

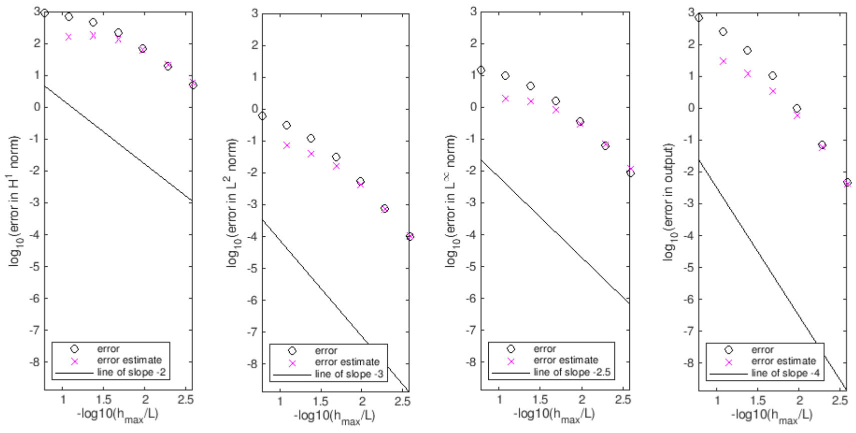


Figure 13: Log-log plots of the errors between the analytic function and the finite element function in different norms as a function of mesh granularity, where the mesh granularity is dictated by $h_{\max}/L$. The solid line indicates the theoretical slope of the error based on asymptotic analysis. We also include the error in the flux at $x = 0$ and its theoretical asymptotic decade in the order of $O(h^4)$.

As seen in Figure 13, we can see that in all four cases, once $-log(h_{\max}/L)$ exceeds 2, the convergence of the error estimates, whose computations we will review later, and the error between the analytic solution and the FEM approximation all approach a line with the specified slope. We now show the 6 element mesh of the piece-wise quadratic approximation and the same error plots.

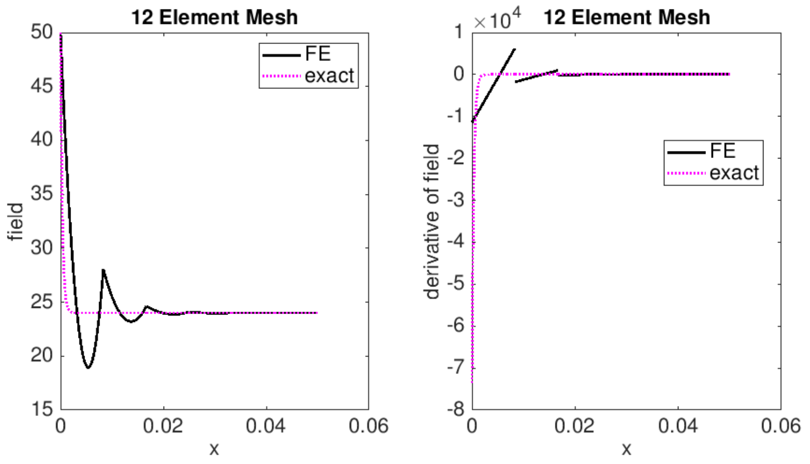


Figure 14: The exact solution and derivative plotted against the finite element solution for a mesh with six elements.

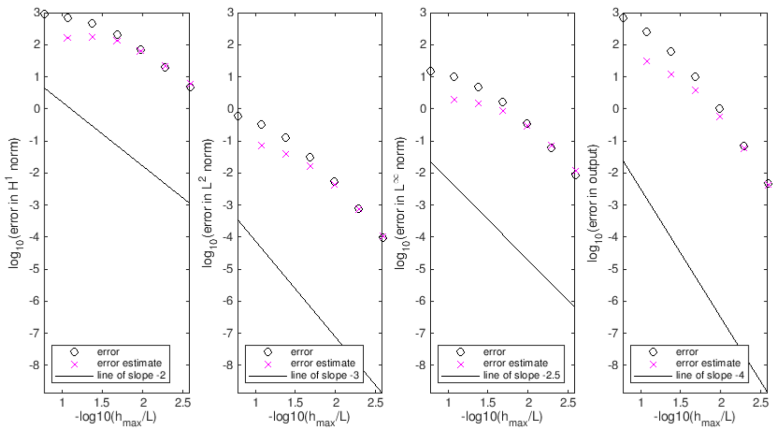


Figure 15: Log-log plots of the errors between the analytic function and the finite element function in different norms as a function of mesh granularity, where the mesh granularity is dictated by $h_{\max}/L$. The solid line indicates the theoretical slope of the error based on asymptotic analysis. We also include the error in the flux at $x = 0$ and its theoretical asymptotic decade in the order of $O(h^4)$.

Note that in this case, we manage to get a more controlled oscillation with a 6 element mesh[1] that still qualitatively matches the shape of the analytic

[1]Since we have a quadratic mesh, we actually have 12 elements.

solution. Fruthermore, we note that the convergence rates of the norms still follow the theoretical decay power functions for values of $-\log(h_{\max}/L)$ greater than 2.

Given that, for this model, the FEM approximations qualitatively match the analytic solutions, and the theoretical power rules that dictate the descent of the errors in various norms matches our experiments, we have some confidence that our code is working appropriately. However, we will provide some additional models that can be used to test functionality. In particular, let us consider systems with non-trivial $k(x)$ and $\mu(x)$ functions.

Let us consider a problem where $k(x) = k_0 \ln(\frac{x}{L} + e)$, μ is constant and our solution is $U_0 \sin(\pi \frac{x}{L})$. Using the method of manufactured solutions, we get that f_Ω must be the following:

$$f_\Omega = \left[k_0 \ln\left(\frac{x}{L} + e\right)\frac{\pi^2}{L^2} + \mu\right] U_0 \sin\left(\pi\frac{x}{L}\right) - \frac{k_0 \pi U_0}{Lx + eL^2}\cos\left(\pi\frac{x}{L}\right)$$

We thus have the problem:

$$-\frac{d}{dx}\left(k_0 \ln\left(\frac{x}{L} + e\right)\frac{dU}{dx}\right) + \mu U = \left[k_0 \ln\left(\frac{x}{L} + e\right)\frac{\pi^2}{L^2} + \mu\right]\sin\left(\pi\frac{x}{L}\right) - \frac{k_0 \pi}{Lx + eL^2}\cos\left(\pi\frac{x}{L}\right)$$

Subject to:

- $U'(0) = \frac{1}{L}U(0) + \frac{\pi}{L}U_0$
- $U'(L) = \frac{1}{L}U(L) - \frac{\pi}{L}U_0$

Likewise, we can set $\mu(x) = \mu_0 \ln(\frac{x}{L} + e)$ and $k(x) = k$, hoping to get the same answer $U(x) = U_0 \sin(\pi\frac{x}{L})$. We get:

$$-k\frac{d^2U}{dx^2} + \mu_0 \ln\left(\frac{x}{L} + e\right) U = \left[k\frac{\pi^2}{L^2} + \mu_0 \ln\left(\frac{x}{L} + e\right)\right] U_0 \sin\left(\pi\frac{x}{L}\right)$$

- $U'(0) = \frac{1}{L}U(0) + \frac{\pi}{L}U_0$
- $U'(L) = \frac{1}{L}U(L) - \frac{\pi}{L}U_0$

We propose these problems to be used for verification of the code's correctness. We now show the same five plots for the insulated conical frustum model and discuss its results.

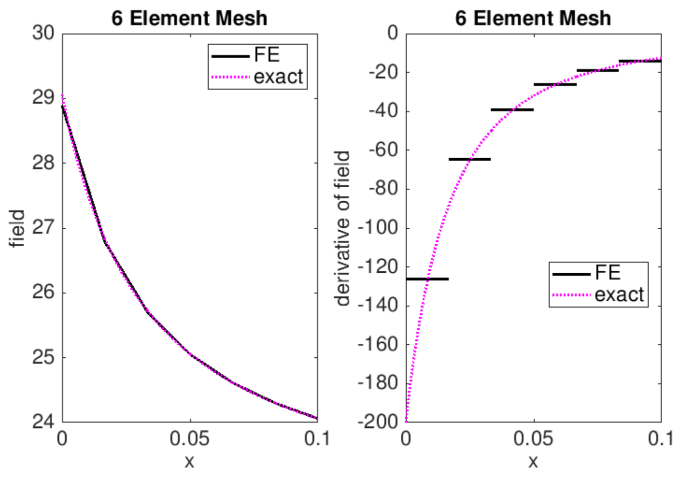


Figure 16: The exact solution and derivative plotted against the finite element solution for a mesh with 6 elements.

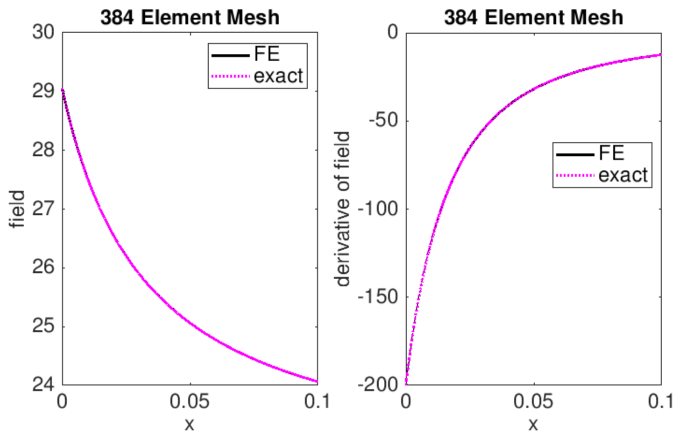


Figure 17: The exact solution and derivative plotted against the finite element solution for a mesh with 384 elements.

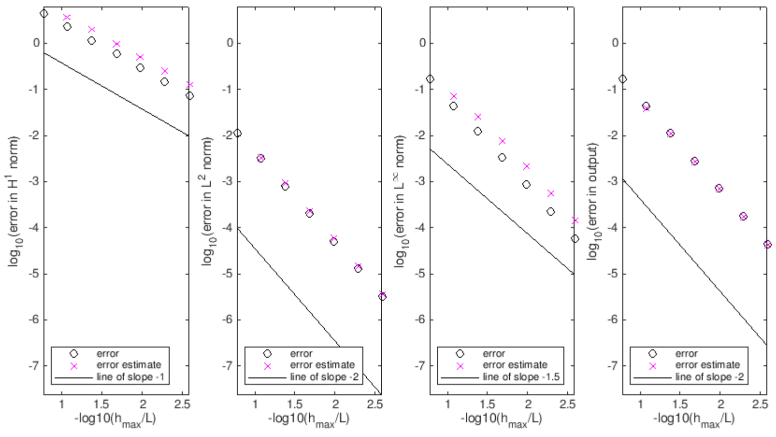


Figure 18: Log-log plots of the errors between the analytic function and the finite element function in different norms as a function of mesh granularity, where the mesh granularity is dictated by $h_{\max}/L$. The solid line indicates the theoretical slope of the error based on asymptotic analysis. We also include the error in the temperature at $x = 0$ and its theoretical asymptotic decade in the order of $O(h^2)$.

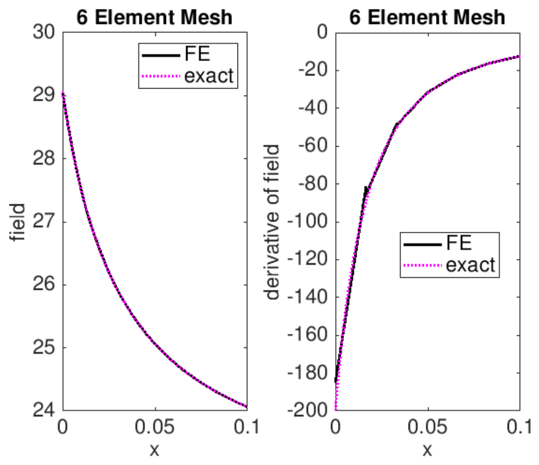


Figure 19: The exact solution and derivative plotted against the finite element solution for a mesh with 6 elements.

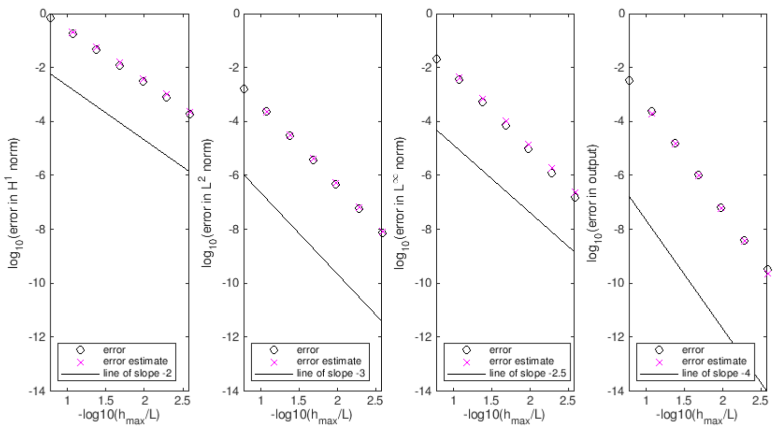


Figure 20: Log-log plots of the errors between the analytic function and the finite element function in different norms as a function of mesh granularity, where the mesh granularity is dictated by $h_{\max}/L$. The solid line indicates the theoretical slope of the error based on asymptotic analysis. We also include the error in the temperature at $x = 0$ and its theoretical asymptotic decade in the order of $O(h^4)$.

Given that, for this model, the FEM approximations qualitatively match the analytic solutions, and the theoretical power rules that dictate the descent of the errors in various norms matches our experiments, we have further evidence that our code is working appropriately. We finally take into account how our FEM code performs on the wall model that we presented before.

Note that the solution to this model is a linear function, which is in the space of piece-wise linear and quadratic functions over any mesh. Hence, we should be able to retrieve the exact solution by using either the piece-wise linear or piece-wise quadratic FEM. Let us consider the 384 element quadratic mesh, which theoretically should be the method whose error should be smallest. We get:

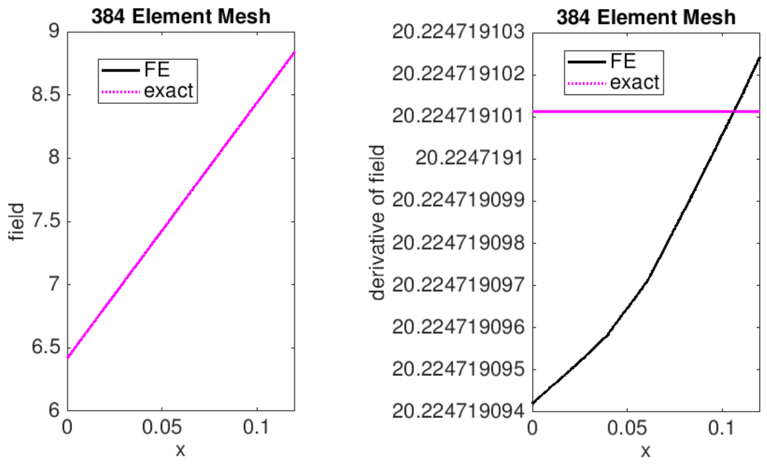


Figure 21: The exact solution and derivative plotted against the finite element solution for a mesh with 384 elements.

We note that the function we retrieve, while indistinguishable from the analytic solution, is not exact as illustrated by the derivative. However, note that the maximum difference between values of the derivative is in the order of 10^{-9}; therefore, the solution the FEM yields does generate an approximately constant derivative despite not being exact. Nevertheless, the solution is in the space of solutions, so by our minimization lemma, we should have gotten the exact solution back. We now explore this phenomenon with our log-log plots for the piece-wise linear vector space and the piece-wise quadratic vector space.

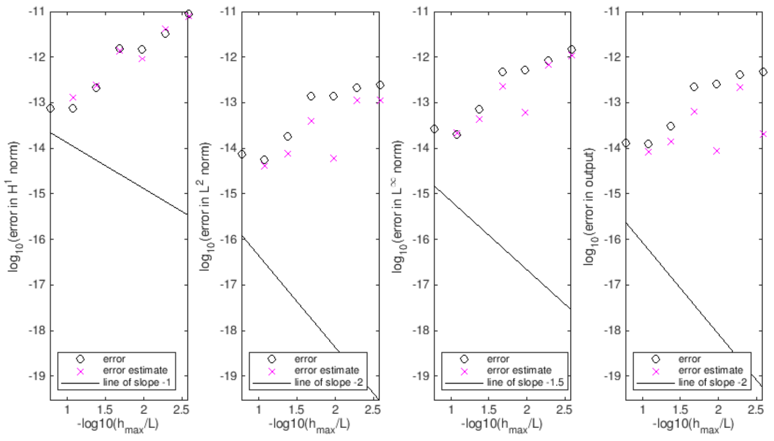


Figure 22: Log-log plots of the errors between the analytic function and the finite element function in different norms as a function of mesh granularity, where the mesh granularity is dictated by $h_{\max}/L$. The solid line indicates the theoretical slope of the error based on asymptotic analysis. We also include the error in the flux at $x = 0$ and its theoretical asymptotic decade in the order of $O(h^2)$.

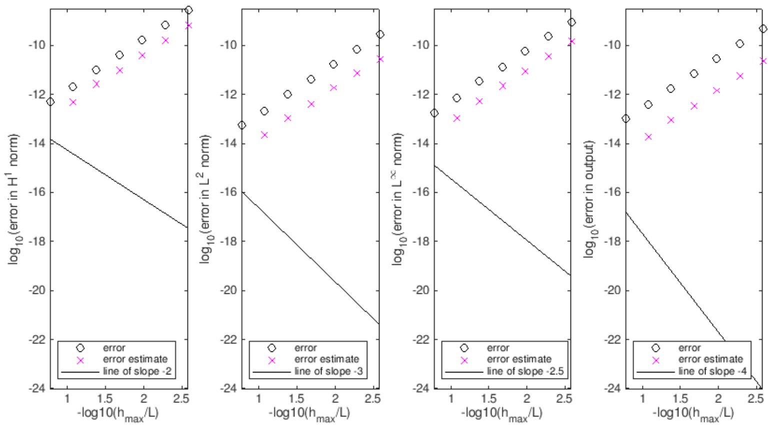


Figure 23: Log-log plots of the errors between the analytic function and the finite element function in different norms as a function of mesh granularity, where the mesh granularity is dictated by $h_{\max}/L$. The solid line indicates the theoretical slope of the error based on asymptotic analysis. We also include the error in the flux at $x = 0$ and its theoretical asymptotic decade in the order of $O(h^4)$.

Interestingly, the error increases as the mesh becomes finer and obviously, the error doesn't follow the theoretical power laws. However, note that the error in the norms are all less than 10^{-10}, which is about 4 orders of magnitude smaller than any of the errors in the previous models.

This must be the result of an effect that is not taken into account when doing the asymptotic analysis of the errors. In fact, the finer the mesh gets, the larger the system matrices become, which decreases the numerical stability of the floating point operations. Therefore, the error being plotted is not dominated by the error induced by the method, like we assumed, but the error induced by floating point operations which we don't account for when doing the big-O analysis. This makes sense because the errors being plotted are in the order of machine precision. Though we get an unexpected consequence from increasing the mesh granularity, the fact that floating point errors are the dominant source of error indicates that we've effectively eliminated, in this case, errors induced by the method, giving us greater confidence in the implementation of the software.

Finally, we explore how our error estimates are computed, and what can be done if the analytic solution is not available.

3.3 Error Estimates

In order to get an estimate for the errors in the different norms, we use the asymptotic convergence rates listed in the previous section. By definition of the big-O notation, we know that for sufficiently fine meshes, we can bound the errors by a constant times $g(h)$ if $||u - u_{FE}|| = O(g(h))$. We can approximate this constant via [Patera, 2019a]:

$$C_u = \max_{m \in \{1,2,\ldots,n\}} \max_{x \in T_m} |u''(x)|$$

Thus, getting an approximation for C_u can give us an upper bound on the error on any of the norms. Furthermore, we can do a sequence of mesh refinements and compare the distance between subsequent approximations to get an upper bound on this constant C_u [Patera, 2019a]. We call these estimates the a-posteriori estimates. In the cases where we don't have an explicit solution, we can use these a-posteriori approximation to verify that after some nominal value of h/L, the error drops off in a power law fashion that matches the theoretical asymptotic behavior. If our code is implemented correctly, we can be sure that this will occur.

The converse, however, may not necessarily true. Imagine, for example, misplacing a decimal place such that one of the parameters is a few orders of magnitude off from what it's supposed to be. In this case, if everything else is implemented correctly, the error estimate would be the error between the solution to the heat equation with the erroneous parameter and the finite element approximation, which would still drop in the appropriate power law fashion. So even though the error in a particular norm may be dropping in the appropriate power law fashion, the finite element solution is converging to the solution with the erroneous parameter, which in general, will be different form the correct solution.

We now repeat the convergence experiments we performed in the previous section for the right-cylinder thermal fin model in the space of piece-wise linear functions without supplying the exact solution and go through interpreting useful information from the estimates. Plotting the error estimates, we get the following figure.

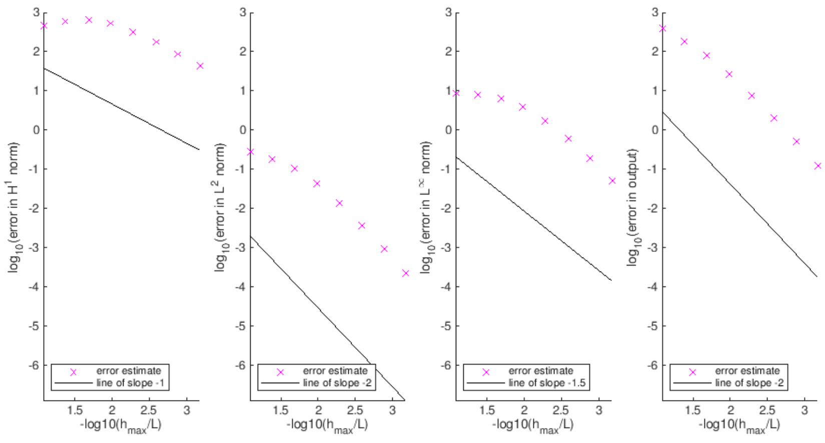


Figure 24: Log-log plots of the approximated errors between the analytic function and the finite element function in different norms as a function of mesh granularity, where the mesh granularity is dictated by $h_{\max}/L$. The solid line indicates the theoretical slope of the error based on asymptotic analysis. We also include the error in the flux at $x = 0$ and its theoretical asymptotic decade in the order of $O(h^4)$.

If we want to ensure that at no point in the domain the percent error is no more than say 0.02, we can simply require that the L^∞ norm be less than approximately 1. In Figure 24, each pink "x" mark denotes one of the meshes used to generate u_{FE}. We can see that the first mesh to have a log error less than zero (which corresponds to an error of less than 1) is the 7^{th} mesh refinement in the uniform refinement scheme. Using the data tool-tip, we can see that the log of the error estimate is -0.1967. Hence:

$$||u - u_{FE}||_{L^\infty} \leq 0.6344$$

If instead, we were curious in the output (i.e. the flux of the function at x=0) of the system at a particular mesh (say the fifth refinement), we could look at the right-most plot. At the point where $-log\,(h/L) = 1.982$. We can thus bound the output using the data tool-tip once again.

$$|U'(0) - U'_{FE}(0)| \leq 27.714$$

We now repeat the experiment and get the equivalent metrics for the frustum model.

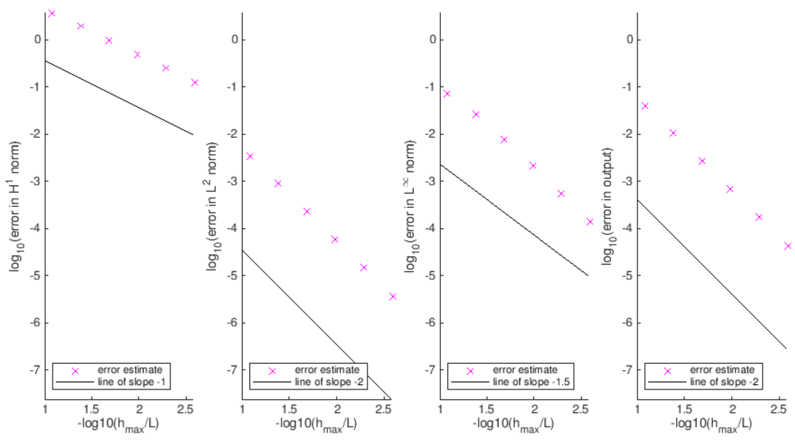


Figure 25: Log-log plots of the approximated errors between the analytic function and the finite element function in different norms as a function of mesh granularity, where the mesh granularity is dictated by $h_{\max}/L$. The solid line indicates the theoretical slope of the error based on asymptotic analysis. We also include the error in the flux at $x = 0$ and its theoretical asymptotic decade in the order of $O(h^4)$.

In this model, the first mesh in the refinement (i.e. the second mesh) is actually already good enough to give us an L^∞ norm that is less than 1. In fact:

$$||u - u_{FE}||_{L^\infty} \leq 0.0747$$

Here, we consider the temperature at $x = 0$ as the output. And at the fifth mesh, we get:

$$|U(0) - U_{FE}(0)| \leq 7.107 \times 10^{-3}$$

4 The Time Dependent Finite Element Method

We now wish to use the machinery developed in the previous two chapters and add the possibility of having the spatial distribution of temperature depend on time. Note that in the previous chapter, we converted the equation

$$-\frac{d}{dx}\left(k(x)\frac{dU}{dx}\right) + \mu(x)U = f_\Omega(x) \tag{8}$$

into a system of equations $A\vec{\alpha} = \vec{F}$, where $\vec{\alpha}$ is the vector of coefficients assigned to each of the basis functions, and A, as we discussed, was either a tri-diagonal or penta-diagonal matrix depending on whether the basis functions were piecewise linear or piecewise quadratic.

We now consider the general, time dependent, one dimensional heat equation

$$\rho(x)\frac{\partial U}{\partial t} - \frac{\partial}{\partial x}\left(k(x)\frac{\partial U}{\partial x}\right) + \mu(x)U = f_\Omega(x) \tag{9}$$

We now express U as a linear combination of our basis functions and allow the coefficients α to depend on time. The idea is that at every given point in time, the function in the vector space of functions that best approximates the solution may be changing; however, since we're still searching in the same space of functions, we should still be able to express it in the basis we've chosen for the space. Hence, only the coefficients have to change. We thus note that:

$$\frac{\partial U_{RR}}{\partial t} = \sum_{i=1}^{n} \frac{\partial \alpha_i}{\partial t}\phi_i$$

Furthermore, let's assume we know the value of $\frac{\partial \alpha_i}{\partial t}$ at the point in time where we're interested in finding the spatial distribution of U. In that case, we can simply lump the $\rho(x)\frac{\partial U}{\partial t}$ term in the $f_\Omega(x)$ function on the right hand side of the equation and call this new function $F_\Omega(x)$ [Patera, 2019l]. We now apply the method from the previous chapter to get a matrix vector equation of the form $A\vec{\alpha} = \vec{G}$. Recall from chapter two the integral expression used to calculate $\vec{G}$. We can thus express $\vec{G}$ as [Patera, 2019k]:

$$G_k = \int_0^L \left(f_\Omega \phi_k - \rho(x)\phi_k \sum_{i=1}^{n} \frac{\partial \alpha_i}{\partial t}\phi_i\right) dx + f_{\Gamma_1}\phi_k(0) + f_{\Gamma_2}\phi_K(L)$$

$$= F_k - \int_0^L \rho(x)\phi_k \sum_{i=1}^{n} \frac{\partial \alpha_i}{\partial t}\phi_i dx = F_k - \sum_{i=1}^{n} \frac{\partial \alpha_i}{\partial t}\int_0^L \rho(x)\phi_k\phi_i dx$$

Note that the right most expression corresponds to the dot product between $\dot{\vec{\alpha}}$ and $\vec{V_i}$, the vector whose kth entry is $\int_0^L \rho(x)\phi_i\phi_k dx$. Hence:

$$G_k = F_k - \vec{V_k} \cdot \dot{\vec{\alpha}}$$

To construct the entire vector $\vec{G}$, we have to take n dot products between n different vectors $\vec{V_k}$ and $\dot{\vec{\alpha}}$. This is equivalent to doing $M\dot{\vec{\alpha}}$, where M is the matrix whose kth row is $\vec{V_k}^T$. We thus get:

$$\vec{G} = \vec{F} - M\dot{\vec{\alpha}}$$

Putting everything together, we get the following system of liner differential equations:

$$\boxed{M\dot{\vec{\alpha}} + A\vec{\alpha} = \vec{F}}$$

Where A and $\vec{F}$ hold the same definitions from chapter 2 and

$$\boxed{M_{i,k} = \int_0^L \rho(x)\phi_k\phi_i dx}$$

To solve the system, we can chose any of numerous ODE time stepping schemes. For the purpose of this work we've chosen to focus on two implicit methods, both of which are unconditionally stable.

4.1 Survey of Time Stepping Methods

- **Backward Euler**

  This method is a simple as implicit methods get. It makes use of the fact that if one Taylor expands around a point t_k, and seeks to evaluate the function at $t_k - \Delta t$, where Δt is our chosen time step, one gets:

  $$U(t_k - \Delta t) = U(t_k) - U'(t_k)\Delta t + O(\Delta t^2)$$

  Hence, one can approximate $U'(t_k)$ as

  $$U'(t_k) = \frac{U(t_k) - U(t_k - \Delta t)}{\Delta t} + O(\Delta t)$$

  Defining $t_k = (k-1)\Delta t$ and $U_k = U(t_k)$, we re-express it in a format that more closely resembles a programmable algorithmic iteration:

  $$\dot{U_k} = \frac{U_k - U_{k-1}}{\Delta t} + O(\Delta t)$$

  We can see that the error in the approximation is asymptotically linear in Δt, so it certainly is consistent, but not strikingly fast at converging. Applying this approximation to the given system of differential equations we previously derived, we get the vector iterative equation:

  $$M\left(\frac{\vec{\alpha}_k - \vec{\alpha}_{k-1}}{\Delta t}\right) + A\vec{\alpha}_k = \vec{F}$$

  Rearranging, we get:

  $$\left(\frac{1}{\Delta t}M + A\right)\vec{\alpha}_k = \vec{F} + \frac{1}{\Delta t}M\vec{\alpha}_{k-1}$$

  Hence, given an initial condition $\vec{\alpha}_1$, we can do a linear solve of the above system to get $\vec{\alpha}_2$. We can then use our result to do another linear solve and get $\vec{\alpha}_3$ and so until we reach the total number of steps n_{steps} we chose to solve for.

- **Crank-Nicolson**

  We now present a slightly more elaborate implicit method. Here, we consider two Taylor expansions about the point $t_k - \frac{1}{2}\Delta t$.

$$U(t_k-1) = U(t_k-\frac{1}{2}\Delta t)-U'(t_k-\frac{1}{2}\Delta t)\frac{\Delta t}{2}+\frac{1}{2}U''(t_k-\frac{1}{2}\Delta t)\left(\frac{\Delta t}{2}\right)^2+O(\Delta t^3)$$

$$U(t_k) = U(t_k-\frac{1}{2}\Delta t)+U'(t_k-\frac{1}{2}\Delta t)\frac{\Delta t}{2}+\frac{1}{2}U''(t_k-\frac{1}{2}\Delta t)\left(\frac{\Delta t}{2}\right)^2+O(\Delta t^3)$$

  Subtracting the two equations from each other gives us:

$$U(t_k-1)-U(t_k) = U'(t_k-\frac{1}{2}\Delta t)\Delta t+O(\Delta t^3)$$

  Using the same convention as before, we get the following approximation for the derivative:

$$\dot{U}_{k-\frac{1}{2}} = \frac{U_{k-1}-U_k}{\Delta t}+O(\Delta t^2)$$

  We can see that this approximation of the derivative error is now asymptotically quadratic in Δt, implying that our error with this scheme will decrease faster with Δt than with backwards Euler. Applying this to our differential equation, we get:

$$M\left(\frac{\vec{\alpha}_k-\vec{\alpha}_{k-1}}{\Delta t}\right)+A\vec{\alpha}_{k-0.5} = \vec{F}$$

  Here, we run into the issue that our algorithm should only solve the problem at integer values of k, so the alpha vector at $k-0.5$ is nonsensical. For this, we note that if we add the two Taylor expansions presented above, we get:

$$U_{k-1}+U_k = 2U_{k-0.5}+O(\Delta t^2)$$

  Hence, if we approximate $\alpha_{k-0.5}$ as the average of U_{k-1} and U_k, we get an approximation whose error is still quadratic in Δt. Hence, the error induced by discretizating the system is still better than the Euler method. This gives us the following method:

$$M\left(\frac{\vec{\alpha}_k-\vec{\alpha}_{k-1}}{\Delta t}\right)+\frac{1}{2}A\left(\vec{\alpha}_{k-1}+\vec{\alpha}_k\right) = \vec{F}$$

Re-arranging as we did before gives us our final iteration, which we can solve just as we did in the backwards Euler case.

$$\left(\frac{1}{\Delta t}+\frac{1}{2}A\right)\vec{\alpha}_k=\vec{F}+\left(\frac{1}{\Delta t}M-\frac{1}{2}A\right)\alpha_{k-1}$$

4.2 Verifying Convergence of the Time-Stepping Routines

To test the validity of our time stepping routines, we perform two tests on our scheme. First, we look at how well our scheme performs when solving the semi-infinite solid model, one of the few heat transfer models that has a transient analytic solution [Patera, 2019l]. Then, we look at how our method performs when solving a burger flipping problem that a third-party FEM code has already solved succesfully.

For a given p FEM scheme (that is, piecewise linear [$p = 1$] or piecewise quadratic [$p = 2$]) and time stepping scheme, one can select a value $1/\sigma < 1$ by which to scale Δt in each mesh refinement so that the asymptotic error between the approximated solution and the real solution at any point in time decreases at the same rate as it would in the time independent case [Patera, 2019j]. As such, we examine the convergence plots in the L^2 norm of our approximation to the semi-infinite model at the last time step.

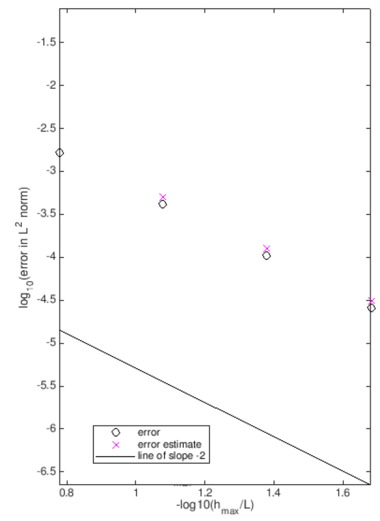


Figure 26: Log-log plot of the L^2 error between our approximation to the semi infinite solid heat transfer problem at $t_{n_{\text{steps}}}$ and $U(t_{n_{\text{steps}}}, x)$ as a function of maximum mesh size. We applied a piece-wise linear FEM scheme and leveraged the Euler backward time stepping routine.

Figure 26 illustrates the convergence plot in the L^2 norm of the aforementioned approximation for the $p = 1$, Euler backward routine. The decrease in actual error and error estimates seems to follow the quadratic power law predicted for the L^2 error of a $p = 1$ scheme in the previous chapter, and the actual errors are consistently below our estimated upper bounds.

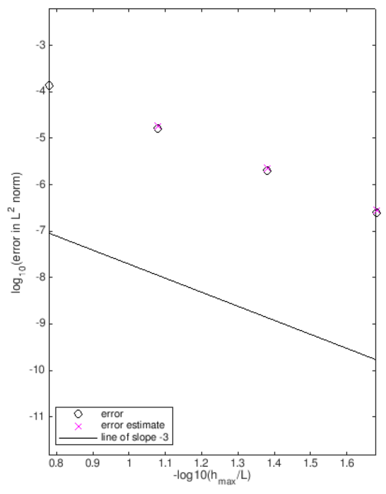


Figure 27: Log-log plot of the L^2 error between our approximation to the semi infinite solid heat transfer problem at $t_{n_{\text{steps}}}$ and $U(t_{n_{\text{steps}}}, x)$ as a function of maximum mesh size. We applied a piece-wise quadratic FEM scheme and leveraged the Crank-Nicolson time stepping routine.

Figure 27 illustrates the convergence plot in the L^2 norm of the aforementioned approximation for the $p = 2$, Crank-Nicolson routine. The decrease in actual error and error estimates seems to follow the cubic power law predicted for the L^2 error of a $p = 2$ scheme in chapter 2, and the actual errors are consistently below our estimated upper bounds.

These results give us confidence that the method is behaving the way it should. To gather more evidence, we shall now regenerate a plot created by a third party FEM when solving the following problem [Patera, 2019l]:

- A burger is taken out of a freezer; its temperature is uniformly $4^{o}C$.
- The burger is placed on a skillet whose temperature is $180^{o}C$.
- The burger cooks until the side that is touching the pan reaches a temperature of $140^{o}C$. This temperature is the temperature at which the Maillard reaction occurs; the reaction that empirically ensures that the burger will taste well.
- At this point, the burger is flipped. It is left cooking until the other side of the burger reaches the Maillard temperature.

- Finally, the burger is taken out of the skillet and it is left to repose, exposed to air in both sides.

Below is the time evolution of the temperature of the burger at $x = 0$ (the skillet touching side), at $x = L$ (the side exposed to air), and at $x = L/2$ (half-way through the burger) as generated by the third party code.

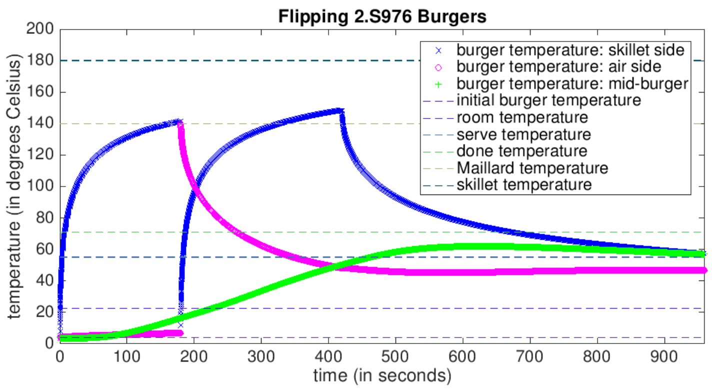


Figure 28: Time evolution plots of the burger flipping problem at $x = 0$, $x = L/2$ and $x = L$ as generated by a Third Party FEM code.

We now reveal our results when applying a $p = 1$ backward Euler scheme.

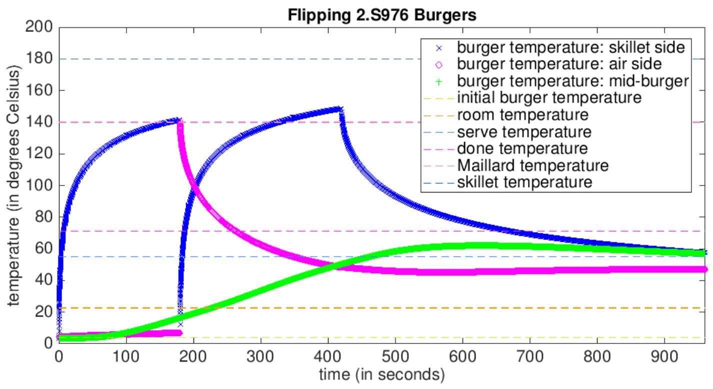


Figure 29: Time evolution plots of the burger flipping problem at $x = 0$, $x = L/2$ and $x = L$ as generated by a Third Party FEM code. The difference between our plots and those generated by the third part code is indiscernible by the naked eye.

It is clear that the two plots look qualitatively equivalent and evaluate to the same values at every point in time, at least to a level of precision admissible by the naked human eye. Upon further inspection, leveraging MATLAB's data tooltips reveals that up to the displayed four decimal places, there is no variation on the function values at 30 randomly selected samples in the plots.

These plots already, with high degree of confidence, suggest that the code has been correctly implemented, but since the third party code provided a second plot corresponding to the spatial distribution of temperature at the final time step, we generated the same. Once again, the graphs are indiscernible from each other by the naked eye and MATLAB's data tooltip.

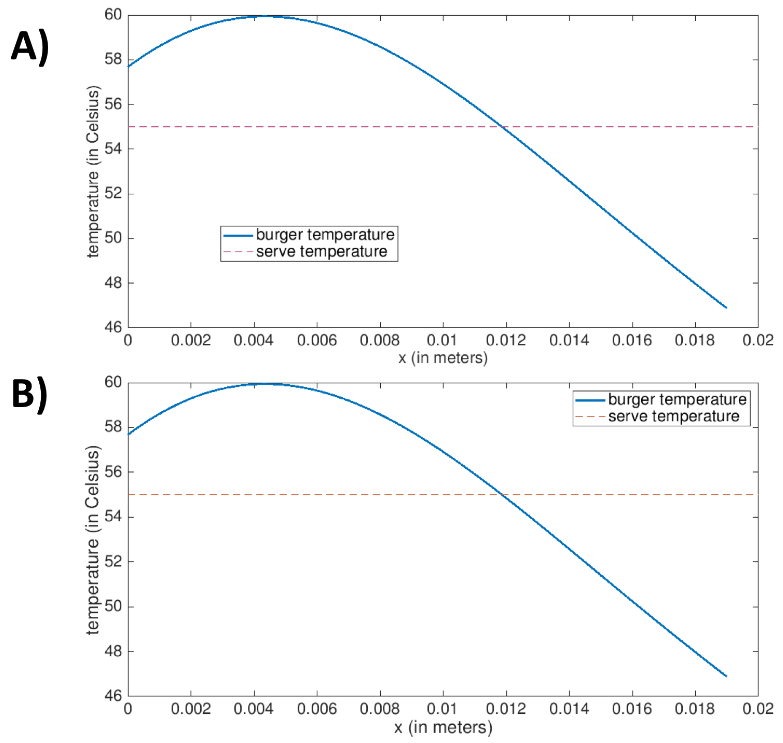


Figure 30: A side by side comparison of the spatial distribution of temperature at the final time generated by **A)** a third party code, and **B)** our code leveraging a $p = 1$ backward Euler scheme.

4.3 Mesh Selection Based on Specifications

Let us consider the burger problem presented in the previous section. There are a few quantities of interest when concerned with cooking a burger to perfection. Let us consider the temperature of the burger on the skillet side just before it gets flipped. We want this value to be close to the Maillard temperature, say 0.001^oC off. Our goal will now be to find a mesh that ensures this will happen.

First, we'll consider a $p = 1$ backward Euler scheme. Below, we can see the convergence plot for the desired quantity under this scheme.

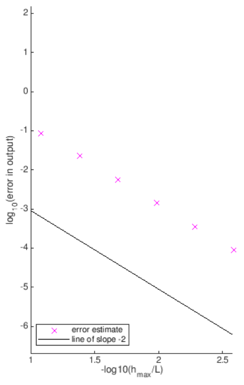


Figure 31: Log-log plot of error upper bound estimates for the temperature of the burger on the skillet side right before it gets flipped as a function of mesh size. This plot is appropriate for a $p = 1$ backward Euler scheme.

Looking at our upper bound estimates, the coarsest mesh in our sequence of mesh refinements that generates an error of at most 10^{-3} is the fifth mesh in the refinement, which will have 192 elements.

Let us now consider a $p = 2$ Crank-Nicolson scheme. Below, we show the convergence plot for the desired quantity under this scheme.

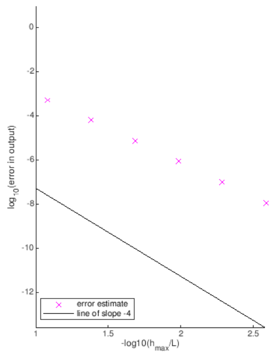


Figure 32: Log-log plot of error upper bound estimates for the temperature of the burger on the skillet side right before it gets flipped as a function of mesh size. This plot is appropriate for a $p = 2$ Crank-Nicolson scheme.

With this method, we achieve our desired threshold with a single mesh refinement, giving us a mesh with 12 elements. Given our mesh size estimates, we now consider the computational complexity of each of our approaches.

Assuming that solving a penta-diagonal system takes about twice as many operations as solving a tri-diagonal one [Patera, 2019k], and nothing that both operations are $O(n)$ for an $n \times n$ matrix, we get the following relationships:

$$C_{p=1,\mathrm{BE}} \approx n_{\mathrm{steps}} \cdot n_{\mathrm{elem}}$$

$$C_{p=2,\mathrm{CN}} \approx 2n_{\mathrm{steps}} \cdot n_{\mathrm{elem}}$$

where C stands for cost, or the approximate number of operations. We further note that given the number of elements and the number of steps are related by the values of σ appropriate for each scheme [Patera, 2019j]. In both instances, we had 20 time steps initially and a mesh with 6 elements. Hence, after k refinements, the number of elements and steps, respectively can be calculated as follows:

$$n_{\mathrm{elem}} = 6(2)^k$$
$$n_{\mathrm{steps}} = 20\sigma^k$$

Now, given the meshes we selected, we know that $k = 5$ for the $p = 1$ backward Euler scheme and $k = 1$ for the $p = 2$ Crank-Nicolson scheme. Furthermore, $\sigma = 4$ for the $p = 1$ backward Euler scheme and $\sigma = 2\sqrt{2}$ for the $p = 2$ Crank-Nicolson scheme [Patera, 2019j]. With this information we get that for the $p = 1$ backward Euler scheme:

$$n_{\mathrm{elem}} = 192$$
$$n_{\mathrm{steps}} = 20480$$

and for the $p = 2$ Crank-Nicolson scheme:

$$n_{\mathrm{elem}} = 12$$
$$n_{\mathrm{steps}} \approx 57$$

We can thus have enough information to get an estimate for the number of steps required to solve the problem in each of the methods. Plugging the values in gives us

$$C_{p=1,\mathrm{BE}} = 3932160$$
$$C_{p=2,\mathrm{CN}} = 1368$$

Hence, despite having to solve a less sparse system, the $p = 2$ Crank-Nicolson FEM scheme is way more computationally effective given the high tolerance of 10^{-3}. As such, we will proceed with this scheme for the remainder of the chapter.

4.4 Applying The Method to a Burger Recipe

After much time developing the machinery necessary to develop a robust FEM code, we are finally ready to apply our code to a real kitchen scenario. We will be following Bobby Flay's burger recipe from the Food Network, and trying to assess if our FEM code can generate a procedure similar to the one illustrated by Flay [Flay, 2009].

Flay instructs in his recipe to "Heat the oil in the pan or griddle over high heat until the oil begins to shimmer. Cook the burgers until golden brown and slightly charred on the first side, about 3 minutes for beef and 5 minutes for turkey. Flip over the burgers. Cook beef burgers until golden brown and slightly charred on the second side, 4 minutes for medium rare (3 minutes if topping with cheese) or until cooked to desired degree of doneness. Cook turkey burgers until cooked throughout, about 5 minutes on the second side" [Flay, 2009].

Furthermore, he suggests making beef patties a quarter inch thick, or 0.01905 meters [Flay, 2009]. Each of these patties is 6 ounces (0.17 kg), which, using a value of mass density value of $1030kg/m^3$ gives an approximate cross sectional area of $0.00866m^2$ [Flay, 2009]. Moreover, these patties are made form ground beef, so in the time the ground beef was taken out of the fridge and the patties were made, the patties probably reach a temperature closer to room temperature than 0^oC.

We will now run our $p = 2$ Crank-Nicolson simulation with these parameters, prescribed by Flay's recipe, and proceed to flip the burger at the suggested time marks (3 and 7 minutes). The evolution diagram for the burger's temperature at $x = 0$, $x = L/2$ and $x = L$ is illustrated below.

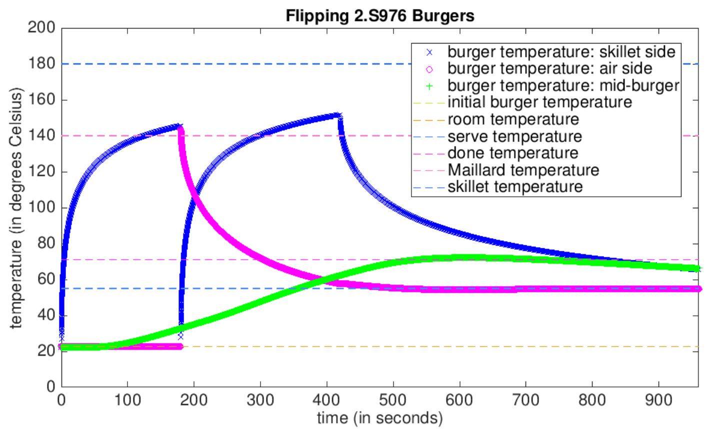


Figure 33: Simulation of Bobby Flay's burger recipe from the Food Netowrk using a $p = 2$ Crank-Nicolson time dependent FEM scheme.

It's clear that with the recommended cooking times and burger dimensions from Flay's recipe, both sides of the burger reach literature values of the Maillard temperature at some point in the process of cooking, so at first glance, it seems like our simulations corroborate Flay's empirical observations.

Nevertheless, it's important to realize that though our numerical approach may be accurate to 10^{-3} degrees, the mathematical model is much more inaccurate, and a more robust model may reveal some discrepancies between proposed recipes and simulations. For example, several recipes instruct chefs to make an indentation with their thumbs instead of making perfectly cylindrical patties. Our current model doesn't capture this more complicated geometry, and as such, there may be discrepancies there that our FEM code has not yet been able to capture. Another source of inaccuracy is our assumption that the heat transfer coefficient is constant within a stage. It's not uncommon, while cooking, to have bursts of smoke around the cooking food suddenly occur, specially right after flipping a burger. These smoke currents increase convective heat transfer, which would in turn change the value of the heat transfer coefficient within the stage [Karnik, 2018]. Given our current model, however, our numerical simulations are performing as expected, and they provide evidence that corroborate cooking recommendations from online recipes.

5 Xylophone Bar Manufacturing

We will now elaborate upon the theory developed on our previous chapters to analyze a new problem - musical instrument manufacturing. It is well known that sound is generated by variations in the pressure of an immersing fluid [Karnik, 2018]; a good way to induce these pressure changes is by making a solid immersed in the fluid oscillate at a certain frequency. This frequency is then perceived as a pitch. As a result, most instruments are designed to have solid parts that are easily set into oscillatory motion upon excitation. Here, we explore xylophone bars and how varying their shape and dimension changes the perceived frequency.

We seek to model a xylophone bar as a long, slender beam. Euler-Bernoulli beam theory provides simple yet extremely ubiquitous models for the study of long and slender beams [Patera, 2019e]. As such, if we model the xylophone bar using Euler-Bernoulli beam theory, we will be able to describe the displacement of the xylophone bar u from its neutral axis via the following equation of motion [Patera, 2019e].

$$\frac{\partial^2}{\partial x^2}\left(EI(x)\frac{\partial^2 u}{\partial x^2}\right) - N_0\frac{\partial^2 u}{\partial x^2} = q(x,t) - \rho(x)\frac{\partial^2 u}{\partial t^2} \tag{10}$$

Here, EI is the possibly space-dependent effective stiffness, which locally equals the product of the Young's Modulus E and area moment of inertia I, N_0 denotes a constant axial force, $\rho(x)$ is the possibly space-dependent linear mass density of the material, and $q(x,t)$ is a possibly space and time dependent distributed force density.

Now, consider we fix a value of x and we're interested in knowing how u evolves at that point in space over time. If we assume that $u \in L^2(\Omega)$, where

Ω is some interval of time (not very important), which is a valid assumption since the system has finite energy, then by Carleson's Theorem, we know that there exists a Fourier series that converges to $u(t)$ almost everywhere in time at every point x [Lacey, 2004]. Hence, we can express $u(x,t)$ in terms of a Fourier series of time at every point in space. We now assume that every point in the bar oscillates in time with the same shape and the only thing that changes from point to point is the amplitude [Patera, 2019d]. We thus define $\hat{u}_k(x)$ to be the amplitude of oscillation, which gives us the following representation for $u(x,t)$.

$$u(x,t) = \sum_{k=0}^{\infty} \hat{u}_k(x) \left(a_k \cos(\omega_{n,k} t) + b_k \sin(\omega_{n,k} t)\right) \tag{11}$$

Let us now assume that there are neither tension nor distributed forces in the xylophone bar, since xylophone bars are usually free to oscillate without any loading. To be more compact, let $\tilde{u}_k(t)$ be $a_k \cos(\omega_{n,k} t) + b_k \sin(\omega_{n,k} t)$. Plugging our ansatz in Equation 11 into Equation 10, we get:

$$\frac{\partial^2}{\partial x^2}\left(EI(x)\frac{\partial^2}{\partial x^2}\left(\sum_{k=0}^{\infty}\hat{u}_k(x)\tilde{u}_k(t)\right)\right) = -\rho(x)\frac{\partial^2}{\partial t^2}\left(\sum_{k=0}^{\infty}\hat{u}_k(x)\tilde{u}_k(t)\right)$$

$$\sum_{k=0}^{\infty}\frac{\partial^2}{\partial x^2}\left(EI(x)\frac{\partial^2}{\partial x^2}\left(\hat{u}_k(x)\tilde{u}_k(t)\right)\right) = -\sum_{k=0}^{\infty}\rho(x)\frac{\partial^2}{\partial t^2}\left(\hat{u}_k(x)\tilde{u}_k(t)\right)$$

We now note that the second derivative in time of $\tilde{u}_k(t)$ is $\omega_{n,k}^2\tilde{u}_k(t)$ for all $k \in \{0,1,2,...\}$. We thus get:

$$\sum_{k=0}^{\infty}\frac{\partial^2}{\partial x^2}\left(EI(x)\frac{\partial^2}{\partial x^2}\left(\hat{u}_k(x)\tilde{u}_k(t)\right)\right) = \sum_{k=0}^{\infty}\rho(x)\omega_{n,k}^2\hat{u}_k(x)\tilde{u}_k(t)$$

To make these two series equal each other at every point (x,t), we must set each term in the sequence equal to each other [Patera, 2019d]. This gives us, for every k, one of the following equations:

$$\frac{\partial^2}{\partial x^2}\left(EI(x)\frac{\partial^2}{\partial x^2}\left(\hat{u}_k(x)\right)\right)\tilde{u}_k(t) = \rho(x)\omega_{n,k}^2\hat{u}_k(x)\tilde{u}_k(t)$$

This equation is true whenever $\tilde{u}_k(t) = 0$, which is not an interesting case, or whenever the following set of equations are satisfied.

$$\frac{\partial^2}{\partial x^2}\left(EI(x)\frac{\partial^2\hat{u}_k(x)}{\partial x^2}\right) = \rho(x)\omega_{n,k}^2\hat{u}_k(x) \;\; \forall k \in \{0,1,2,...\} \tag{12}$$

Note that there is no time dependence in these equations. We have thus converted our Partial Differential Equation into a system of Ordinary Differential Equations. However, we have introduced a set of unknowns $\omega_{n,k}$. Note that the $\omega_{n,k}$ values are frequencies for a set of sinusoids. We thus call these values the frequencies or overtones of our solution, since these oscillations are what humans will perceive as pitch [Patera, 2019d].

The smallest non-zero frequency $\omega_{n,k}$ is what we will perceive as the pitch of the xylophone bar [Patera, 2019d]. We call this the fundamental frequency. The second smallest non-zero frequency $\omega_{n,k}$ is the first overtone, which in xylophones, is often tuned to be exactly 3 or 4 times the fundamental frequency [Caresta, 2014]. Furthermore, xylophone bars are often held together by two strings in tension that go through the bars at certain points [Patera, 2019e]. If there is a lot of oscillation at these points, the string will vibrate, which will excite other bars and affect their pitch. As a result, xylophone manufacturers must know the points in the bar that oscillate the least. These points correspond to the points where the mode shape corresponding to the fundamental frequency equals zero (i.e. there is no displacement) [Patera, 2019e].

It's thus clear that this representation of the Euler-Bernoulli bending equation is valuable for the design of xylophones. As such, we will develop some finite element schemes to be able to retrieve the values of $\omega_{n,k}$ and the mode shapes $\hat{u}_k(x)$. To begin solving for these, however, we must apply some boundary conditions.

Since xylophone bars are free to oscillate, the ends of the bar will not support neither a moment nor a force. This corresponds to the second derivative of u and the first derivative of $EI\frac{\partial^2 u}{\partial x^2}$ being zero at each end of the bar ($x = 0$ and $x = L$) [Patera, 2019e]. At this point, there isn't much we can say about the values of $\omega_{n,k}$ or $u(x,t)$, unless we have some information about $EI(x)$ and $\rho(x)$.

A xylophone bar will often feature a small indentation in the bottom to give it its distinctive tuning [Patera, 2019e]. To model this indentation, we track the height of the bar H at every position x. Furthermore, we assume a variation in the height that can be modeled as a quartic polynomial of the form of Equation 13.

$$H(x) = H_{\max} \min \left\{ 1, \left[(1-p) \left(\frac{L_{d/2} - x}{L_{d/2} - x^*} \right)^4 + p \right] \right\} \tag{13}$$

Here, p denotes a design variable that takes values between 0.05 and 1 that denotes how much material gets cut from the bar (with 1 being no material - i.e. the bar is just a rectangular prism), $H_{\max}$ is the maximum height of the bar, $L_{d/2}$ denotes the half-way point in the bar, and x^* denotes the position at which the dent begins.

Furthermore, we assume that the bar is made from a material with uniform density ρ, and has a uniform width W. Under these assumptions, we get the following governing equation [Patera, 2019e].

$$\frac{d^2}{dx^2} \left(\frac{EH^3(x)}{12} \frac{d^2 \hat{u}_k}{dx^2} \right) = \omega_{n,k}^2 \rho H(x) \hat{u}_k \tag{14}$$

Now, we have an equation that can be solved with a slightly modified finite element method. In the next section, we will present this method and discuss the results we obtain for $\omega_{n,k}$, and the mode shapes $\hat{u}_k$.

6 Finite Element Method for Bending

Much like in the previous report, we must develop an energy functional $\pi(w)$ that is minimized when w is the solution to our problem. For the problem posed above, subject to the aforementioned boundary conditions, we propose the following functional [Patera, 2019b].

$$\pi(w) = \frac{1}{2}\int_0^L \frac{EH^3(x)}{12}\left(\frac{\partial^2 w}{\partial x^2}\right)^2 dx - \int_0^L \omega_{n,k}^2 \rho H(x)u(x)w(x)dx \tag{15}$$

This functional looks a bit odd since it depends on the solution $u(x)$ of the problem, which means that to evaluate π, we need to know what the solution to the problem is to begin with. However, we will see that we can still minimize π without knowing u [Patera, 2019c].

Another peculiar property of this functional is that we can minimize it not only by plugging in u, but also by plugging in $u+v$ where v is an affine function of x (i.e. of the form $v = c_1+c_2x$) [Patera, 2019b]. Although it sounds problematic that minimizing π may get us an answer that is an affine shift away from the real answer, it should not be surprising. After all, an affine shift of u corresponds to being able to shift the xylophone bar up or down, and being able to rotate it [Patera, 2019b]. Sure, solving the problem may give us an arbitrary orientation for the xylophone bar, but the orientation will not change the frequency content of u, which is all that we care about. We thus don't have to worry about this affine shift.

We now prove that this functional is minimized by plugging in $u + v$ where v is an affine function of x [Patera, 2019b].

Proof. Let $w = u + v$, where u is the solution to our problem and v is an arbitrary perturbation. We now evaluate $\pi(u + v)$. To be compact, we will denote partial derivatives with respect to x with subscripts.

$$\pi(u+v) = \frac{1}{2}\int_0^L \frac{EH^3}{12}\left(u_{xx}^2 + 2u_{xx}v_{xx} + v_{xx}^2\right)dx - \int_0^L \omega_{n,k}\rho H u\left(u+v\right)dx$$

Collecting the integrals with u_{xx}^2 and u^2 gives us $\pi(u)$. Hence:

$$\pi(u+v) = \pi(u) + \frac{1}{2}\int_0^L \frac{EH^3}{12}\left(2u_{xx}v_{xx} + v_{xx}^2\right)dx - \int_0^L \omega_{n,k}\rho H uv dx$$

We now focus on the integral with the $u_{xx}v_{xx}$ term. Doing integration by parts once gives us the following.

$$\int_0^L \frac{EH^3}{12}u_{xx}v_{xx}dx = \left[\frac{EH^3}{12}u_{xx}v_x\right]_0^L - \int_0^L \left(\frac{EH^3}{12}u_{xx}\right)_x v_x dx$$

We now note that under the boundary conditions we assumed, $u_{xx} = 0$ at $x = 0$ and $x = L$. Thus the boundary terms vanish giving us the following.

$$\int_0^L \frac{EH^3}{12} u_{xx} v_{xx} dx = -\int_0^L \left(\frac{EH^3}{12} u_{xx}\right)_x v_x dx$$

We now perform integration by parts once again to get the following.

$$\int_0^L \frac{EH^3}{12} u_{xx} v_{xx} dx = -\left\{\left[\left(\frac{EH^3}{12} u_{xx}\right)_x v\right]_0^L - \int_0^L \left(\frac{EH^3}{12} u_{xx}\right)_{xx} v dx\right\}$$

We now note that under the boundary conditions we assumed $\left(\frac{EH^3}{12} u_{xx}\right)_x$ vanishes at $x = 0$ and $x = L$. Thus the boundary terms vanish giving us the following.

$$\int_0^L \frac{EH^3}{12} u_{xx} v_{xx} dx = \int_0^L \left(\frac{EH^3}{12} u_{xx}\right)_{xx} v dx$$

Plugging this result into the functional gives us the following.

$$\pi(u+v) = \pi(u) + \int_0^L \left(\frac{EH^3}{12} u_{xx}\right)_{xx} v dx + \frac{1}{2}\int_0^L \frac{EH^3}{12} v_{xx}^2 dx - \int_0^L \omega_{n,k} \rho H u v dx$$

Combining the first integral term with the last, we get the following.

$$\pi(u+v) = \pi(u) + \int_0^L \left\{\left(\frac{EH^3}{12} u_{xx}\right)_{xx} - \omega_{n,k} \rho H u\right\} v dx + \frac{1}{2}\int_0^L \frac{EH^3}{12} v_{xx}^2 dx$$

Since u solves Equation 14, the term inside the curly braces must go to zero. This gives us the following.

$$\pi(u+v) = \pi(u) + \frac{1}{2}\int_0^L \frac{EH^3}{12} v_{xx}^2 dx$$

We now note that the integral term can only add to the value of $\pi(u)$ unless $v_{xx} = 0$. Therefore this functional is minimized whenever w is u plus a function whose second derivative is zero, also known as an affine function.

□

Like before, we now select a set of basis functions $\{\phi_1, \phi_2, ..., \phi_N\}$ that span the vector space of functions over which we seek to minimize π. We thus write our approximation u_{FE} as follows:

$$u_{\text{FE}}(x) = \sum_{i=1}^{N} \alpha_i \phi_i \tag{16}$$

Plugging this representation of U_{FE} into π, gives us the following [Patera, 2019c].

$$\pi(u_{\mathrm{FE}}) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\int_0^L \left\{ \frac{EH^3}{12}\alpha_i\alpha_j \frac{\partial^2\phi_i}{\partial x^2}\frac{\partial^2\phi_j}{\partial x^2} - \omega_{n,k}^2\rho H\alpha_i\alpha_j\phi_i\phi_j \right\} dx$$

If we fix the basis functions, then π is now a function that takes in N α's and returns a number. To minimize this, we simply have to set the gradient with respect to $\{\alpha_1, \alpha_2, ..., \alpha_N\}$ to zero. Without the details explaining how to take this gradient that we presented in our previous report, we arrive at the following set of equations.

$$\sum_{i=1}^{N}\int_0^L \frac{EH^3}{12}\alpha_i \frac{\partial^2\phi_i}{\partial x^2}\frac{\partial^2\phi_j}{\partial x^2} dx = \sum_{i=1}^{N}\int_0^L \omega_{n,k}^2\rho H\alpha_i\phi_i\phi_j \quad \forall j \in \{1, 2, ..., N\} \tag{17}$$

We can write these equations more compactly with matrices to get the following.

$$A\vec{\alpha} = \omega_{n,k}^2 M\vec{\alpha} \tag{18}$$

Here, the matrix A is the matrix whose entries are defined as follows:

$$A_{i,j} = \int_0^L \frac{EH^3}{12}\frac{\partial^2\phi_i}{\partial x^2}\frac{\partial^2\phi_j}{\partial x^2} dx$$

the matrix M is the matrix whose entries are defined as follows:

$$M_{i,j} = \int_0^L \rho H\phi_i\phi_j dx$$

and $\vec{\alpha} = [\alpha_1, \alpha_2, ..., \alpha_N]^T$. Now recall that we're interested in finding the frequency content of the solution to Equation 10. We thus want to find the values of $\omega_{n,k}$ that makes Equation 18 have non-zero solutions $\vec{\alpha}$. This is equivalent to finding the values of $\omega_{n,k}$ that make $\det(A - \omega_{n,k}M) = 0$.

Once we have these values, then we know that $\vec{\alpha}$ must be in the nullspace of $A - \omega_{n,k}M$. Clearly, if $\vec{\alpha}$ is a solution, then so are all scalar multiples of $\vec{\alpha}$. In order to get a unique solution, we'd need to have some information about how the xylophone bar was struck; more specifically, we'd need two initial conditions in time. Since we only care about the frequency content of the solution and not the amplitudes, we just need to chose one of the solutions [Patera, 2019b]. Let us chose the solution that satisfies the normalization constraint $\vec{\alpha}^T M\vec{\alpha} = 1$.

We now make note two other considerations that we must make before putting this method to use. First, we can expect $\omega_{n,k} = 0$ to be a root of multiplicity 2 for the polynomial $P(\omega_{n,k}) = \det(A - \omega_{n,k}M)$ [Patera, 2019c]. These correspond to the arbitrary affine shift that we alluded to when proving the minimization proposition. We don't care about the xylophone bar's spatial orientation, so we will simply ignore the solutions $\vec{\alpha}$ when $\omega_{n,k} = 0$.

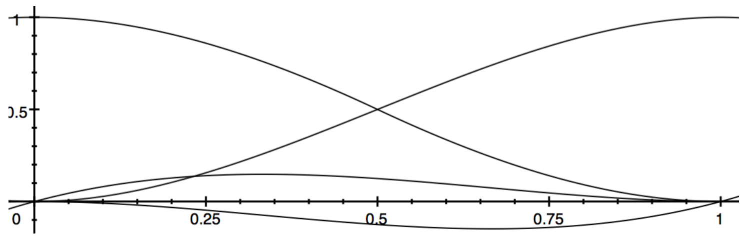


Figure 34: Plot of the four non-zero functions defined over a reference element.

Furthermore, in our previous report, we consider ϕ functions that were piecewise linear or quadratic. However, these are problematic in the calculation of our matrix A. The piecewise linear functions have a second derivative that is zero almost everywhere, so the matrix A would be zero under this scheme. Furthermore, it is important for our approximation to have a continuous first derivative since we're integrating the second derivative and the accumulation function $F(x) = \int_0^x f(x')dx'$ is continuous for any Riemann Integrable function f. However, the piecewise quadratic elements had peaks upon which the first derivative had jump discontinuities. As a result, we can no longer use piecewise quadratic elements, which forces us to use piecewise cubic functions.

A reference element would have 4 non-zero functions defined over it as illustrated in Figure 34 [Patera, 2019c]. Each function has the property that either it or its derivative evaluates to 1 at either the left or right endpoint, and to zero at the other points [Patera, 2019c]. This ensures that we can evaluate U_{FE} and its derivative at the nodes by looking at the coefficients of one of the functions. Finally, this set up ensures that each function only overlaps with at most six other functions (including itself), which makes the matrices sparse and the computations less extensive [Patera, 2019c].

We now proceed to use the developed method to design a xylophone bar tuned to an F4 with double-octave tuning (i.e. the ratio between the first overtone and the fundamental frequency is 4).

7 Xylophone Bar Design

To tune a xylophone bar to F4 with double-octave tuning, we want to fix an $x*$ and $H_{\max}$ and adjust p and L. For each point (p, L) we select, we will solve Equation 18 and compare the fundamental frequency to that of F4 (2194.2768 radians per second or 349.23 Hz), and the ratio between the frequency of the first overtone and the fundamental frequency to 4 [Patera, 2019e].

Once the two parameters have been found, we take the shape of the fundamental frequency, and we find its zeros. Luckily, we have access to the value of the function at the nodes of the finite elements. We can thus find the elements that may have a zero by going through all of the elements and multiplying the function evaluated at the right and left endpoints. If the product is negative or zero, then by the intermediate value theorem, we know there must be a zero in this element.

After flagging all the elements that contain a zero, we pick the first and last one. The zeros found in these elements will be the ones through which we will pass the strings. Once we've selected the elements, we can simply run a root finding algorithm (like a bisection search or Newton's method). These algorithms usually need an initial guess; we simply supplied the method with the halfway point of the element in question and let it run. We show our results in Figure 35.

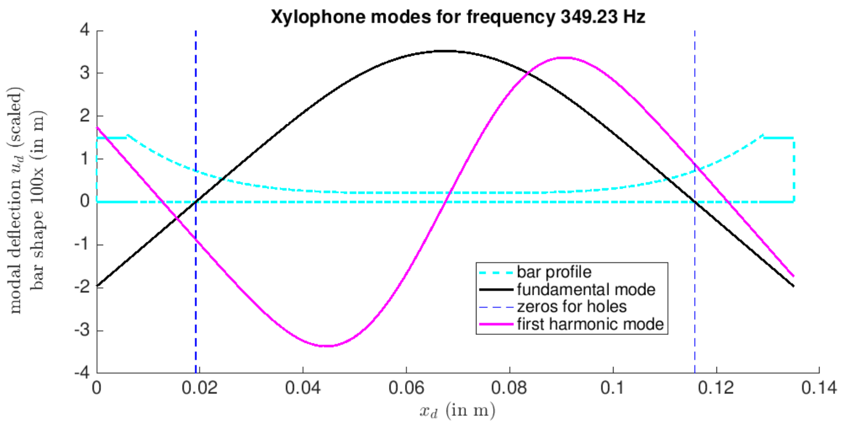


Figure 35: Plot of the fundamental frequency and the first harmonic superposed over a diagram of the bar's optimized shape design. The blue lines indicate the places where our algorithm finds zeros for the fundamental frequency.

With this plot, we get a good sense that the root finding algorithm is working as it is supposed to. The discrepancy between the blue lines and the points where the fundamental frequency actually hits the x axis (i.e. $u = 0$) is not only indiscernible by the naked eye, but also by the Matlab data tool-tip, which is accurate to six decimal places.

While this gives us certainty that once we have a node shape, we can locate the points where to drill the holes with high fidelity, we still have yet to check the accuracy with which our method finds the fundamental frequency and other overtones of our xylophone bar.

We check how well our algorithm performs when designing the F4 xylophone bar in double-octave tuning mentioned above. We provided our algorithm the following parameters:

- $f = 349.23$ Hz (or $\omega_{n,k} = 2194.2768$ radians per second)
- $\frac{f_1}{f} = 4$ (f_1 is the first overtone)
- $H_{\max} = 0.015m$

- $\frac{x^*}{L} = 0.05$
- $p_{\min} = 0.05$
- $p_{\max} = 1$ (the algorithm will thus search for values of p in [0.05,1])
- $E = 14$ GPa (Young's Modulus of Rosewood)
- $\rho = 835 \frac{kg}{m^3}$ (Density of Rosewood)

Our algorithm produced the following values for L, p, f and f_1 with a posteriori error estimates.

- $f = 349.2300 \pm 5.7666 \times 10^{-5}$ Hz
- $f_1 = 1393.8 \pm 1.4448 \times 10^{-5}$ Hz
- $L = 0.135m$
- $p = 0.1319$

As we can see, the error upper bound on the fundamental frequency is well below a tolerance of 10 Hz, which is about what an average human can discern as being two different pitches [Patera, 2019d]. To get an error estimate for the ratio between f_1 and f, we can do a sensitivity analysis. Namely, if we let $R = f_1/f$, we can write its uncertainty ϵ_R as follows [Staff, 2013].

$$\epsilon_R = \sqrt{\left(\frac{\partial R}{\partial f}\right)^2 \epsilon_f^2 + \left(\frac{\partial R}{\partial f_1}\right)^2 \epsilon_{f_1}^2} = \sqrt{\left(-\frac{f_1}{f^2}\right)^2 \epsilon_f^2 + \left(\frac{1}{f}\right)^2 \epsilon_f^2}$$

Plugging in the values our code produced, we arrive at the following error.

$$\epsilon_R = 6.603 \times 10^{-7}$$

We thus report the calculated value and uncertainty of our desired ratio.

$$f_1/f = 3.99106 \pm 6.603 \times 10^{-7}$$

Although 4 does not fall in the 95% confidence interval of our simulated value of R, we are certainly close to it; the value of our first overtone is about 4 Hz away from the actual overtone under perfect double-octave tuning, which as discussed, is a frequency difference that the average human would have trouble discerning.

We continue our discussion of errors by looking at convergence plots (Figures 36 and 37) for the node shapes and frequency values of the fundamental frequency and first overtone of two different xylophone bars. The first is the one we just reversed engineered to be an F4 with double-octave tuning; the second is a bar that is supposed to be a C5 with quint tuning (i.e. $f_1/f = 3$).

We can see that, in both figures, the a posteriori error estimates decrease with the size of the largest mesh element in a power law fashion that is consistent with the asymptotic behavior predicted a priori. This gives us confidence that the error estimators we used in our error analysis for the F4 bar are reliable.

Another indicator of the error estimators' reliability is the difference in error between the fundamental frequency and the first overtone in the same mesh. We expect higher frequencies to be more difficult to approximate with finite elements because the node shapes will changes more drastically at higher frequencies, which becomes increasingly difficult for polynomials to be able to approximate.

Looking at Figures 36 and 37, we can see that for any given mesh size, the error for the node shape in all of the norms is larger for the overtone than the fundamental frequency, and so is the numerical value of the frequency.

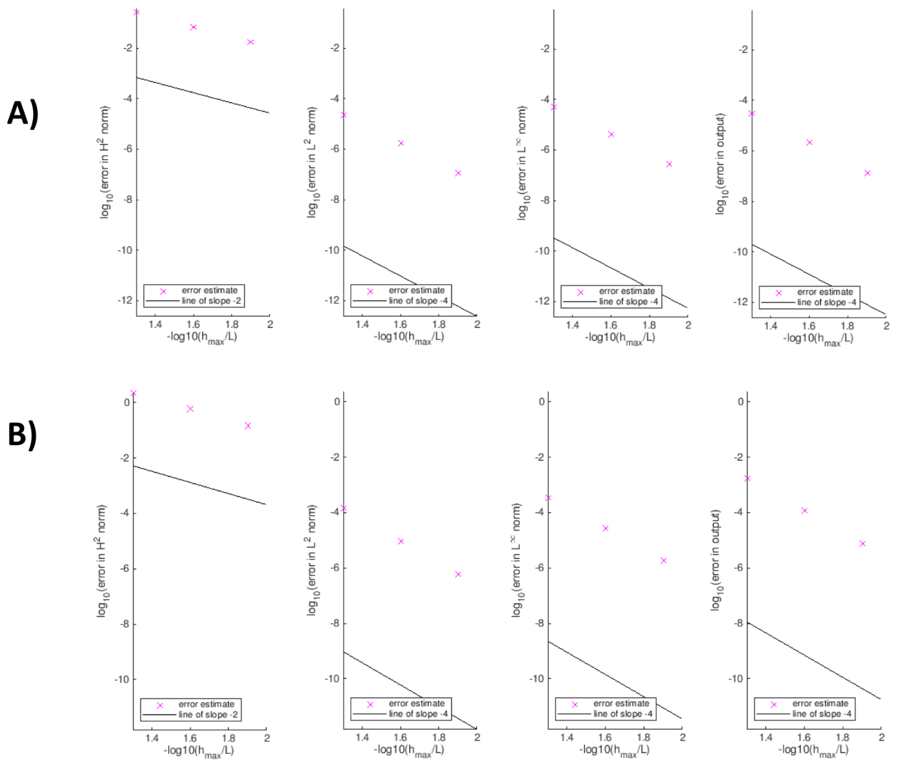


Figure 36: Convergence plot for the node shapes of the fundamental frequency (Panel **A**) and the first overtone (Panel **B**) in an F4 bar with double-octave tuning in 3 norms (H^2, L^2 and L^∞). To the far right in each panel is a convergence plot for the value of the frequency.

From the convergence plots we can also tell that we are refining too much. Sure, we're gaining more accuracy, but recall that the average human cannot discern pitches that are less than 10 Hz apart. Therefore, getting a frequency error in the order of 10^{-7} is not necessary. In fact, it seems like refining once

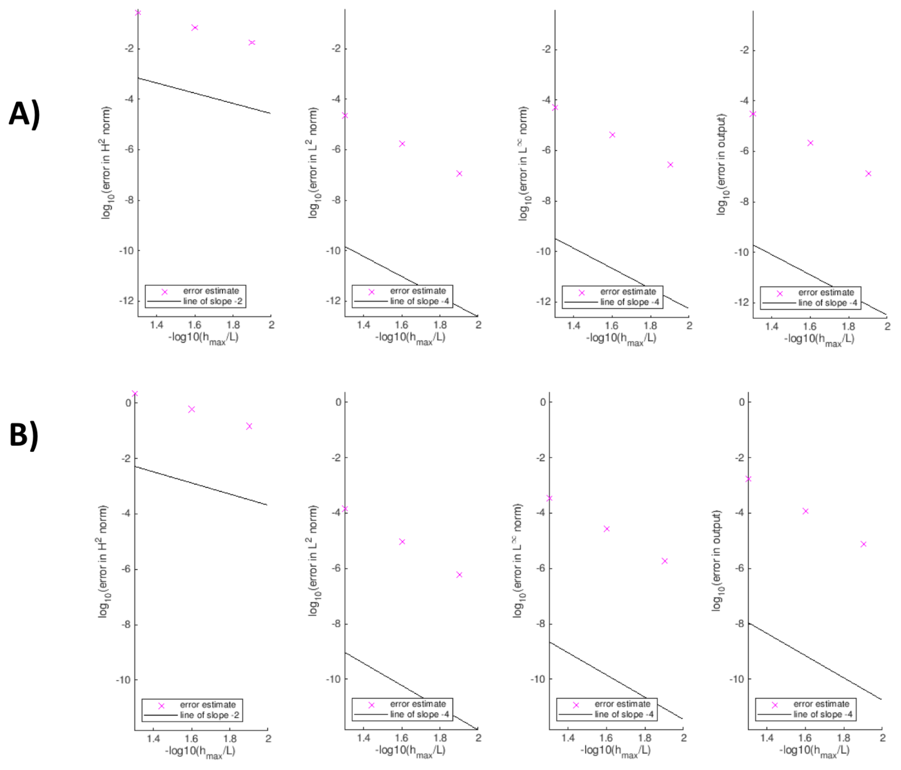


Figure 37: Convergence plots for the node shapes of the fundamental frequency (Panel **A**) and the first overtone (Panel **B**) in a C4 bar with quint tuning in 3 norms (H^2, L^2 and L^∞). To the far right in each panel is a convergence plot for the value of the frequency.

already puts the solution error well below the 10 Hz tolerance. It would seem as though the original mesh of 11 elements would already be fine enough to get us within 10 Hz of the actual frequency; however, we don't have an error estimate for this mesh, so as a measure of precaution, we recommend a single refinement in order to get a reliable answer without having to do unnecessarily many computations.

With these observations, we can be pretty confident that our finite element approach to the xylophone design problem yields high fidelity approximations to solutions of Equation 14. However, there may still be sources of errors surrounding the model. The theory we've developed here is only appropriate if we can model the xylophone bar using principles from Euler-Bernoulli beam theory.

Euler-Bernoulli beam theory assumes that cross sections perpendicular to the neutral axis remain perpendicular to the bending line [Anand, 2011]. This assumption begins to break down when the ratio between the length L of the beam and the thickness $H_{\max}$ becomes small [Anand, 2011]. If we consider bars that all have the same thickness, then we can expect to run into issues with short bars.

Furthermore, using the Buckingham Pi Theorem, we can deduce that the fundamental frequency is inversely proportional to the square of length of the bar [Karnik, 2018] [Patera, 2019d]. As a result, we expect our method to be better suited for low pitched bars, since these bars will inherently be longer.

To check how well our numerical experiments match reality, we will devote the following section to recreating results found in the instrument manufacturing literature.

8 Literature Comparisons

In his preprint "Vibrations of a Free-Free Beam", Dr. Mario Caresta analyzes the frequency content of a rectangular prism with the following parameters [Caresta, 2014].

- $L = 1.275m$
- $H_{\max} = 0.01m$
- $W = 0.075m$
- $\rho = 7800\frac{kg}{m^3}$
- $E = 2.1 \times 10^{11}\frac{N}{m^2}$

He reports the fundamental frequency and the frequencies of the first four overtones in the following table [Caresta, 2014].

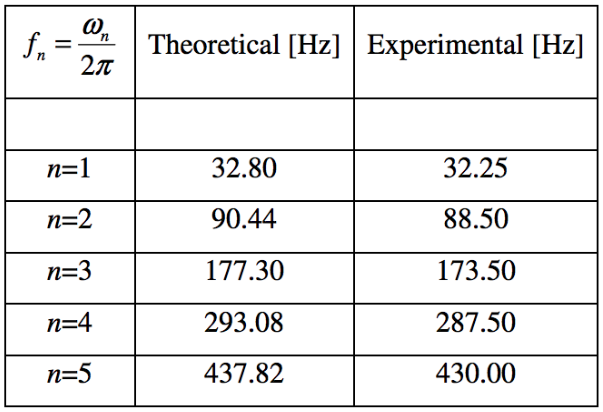
To verify these results, we input the following parameters into our length and p finding algorithm.

- $f = 32.8$ Hz
- L, E, $H_{\max}$, and ρ as specified by Caresta.
- $p_{\min} = p_{\max} = 1$ to force the shape of the bar to be a rectangular prism.
- Random values for the desired ratio between frequencies R and x^* since the shape (and consequently the ratio) is pre-determined.

After running our algorithm, we managed to generate the following figure.

Just by looking at figure 39, we can kind of tell that our algorithm found the length of the bar to be around the same value as the one reported by Caresta. Looking at the output values, we can further verify our agreement with Caresta's experiments. Our algorithm produced the following outputs.

- $L = 1.2752m$



$f_n = \frac{\omega_n}{2\pi}$	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

Figure 38: First five natural frequencies in Caresta's beam as reported in his preprint, "Vibrations of a Free-Free Beam" [Caresta, 2014].

- $f = 32.8$ Hz
- $f_1 = 90.4145$ Hz

Our algorithm was thus able to recover the length of the beam, the fundamental frequency, and the frequency of the first overtone with a percent error of less than 1% when compared to the theoretical values, and less than 3% when compared to the experimental values. These results are valuable for they not only give further evidence that the code is behaving as it should, but it also gives some insight regarding the appropriateness of Euler-Bernoulli beam theory when modeling oscillating bars. We can thus conclude that when looking at long and slender beams, our finite element method can produce frequencies that have less than 3% error when compared to experimental values.

Finally, in the subsequent section, we will propose some slight modifications to the method we've presented in this paper to solve a more general problem.

9 Extensions

So far, we have assumed that inserting strings into the xylophone bar will not affect its vibration patterns, when in reality, the strings will behave as a restoring force, which will introduce some Robin boundary conditions [Patera, 2019e].

Let us now consider a Hookean spring with constant k attached at $x = L$ to model a string being passed through the bar near $x = L$. This will add a $\frac{1}{2}kw^2(L)$ term to the energy functional, which will in turn, slightly modify our definition of the A matrix in Equation 18. In this case, our A matrix will be defined as follows [Patera, 2019c].

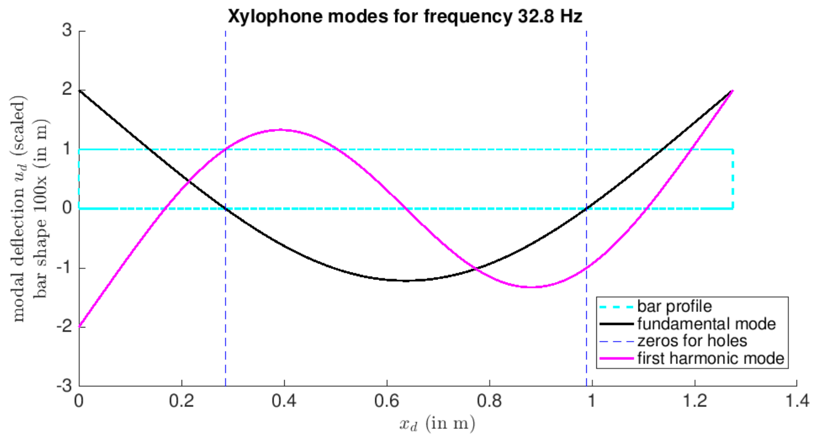


Figure 39: Plot of the fundamental frequency and the first harmonic of our simulation of Caresta's rectangular prism bar.

$$A_{i,j} = \int_0^L \frac{EH^3}{12} \frac{\partial^2 \phi_i}{\partial x^2} \frac{\partial^2 \phi_j}{\partial x^2} dx + k\phi_i(L)\phi_j(L)$$

All we must do now is edit the matrix A and everything else can remain the same. Luckily, we've defined the ϕ functions to have the property that at every node of the mesh, all except one ϕ function evaluate to zero with the exception evaluating to 1. Since the endpoint of the domain $x = L$ is always a node of the mesh, there is only one function ϕ that is non-zero at $x = L$. In fact, if we let ϕ_{2n-1} be the ϕ function that is 1 at node n and ϕ_{2n} be the ϕ function whose derivative equals 1 at node n, then if we have N nodes, there are $2N$ ϕ functions and the second to last one (i.e. ϕ_{2N-1}) will be the one that evaluates to 1 at $x = L$.

Therefore, the only entry of the matrix A that changes by adding the spring at the end of the bar is $A_{2N-1,2N-1}$. To do this, we can simply look at the routine that assembles the system matrix A. Currently, this function looks as follows.

```
function [A_N,M_N,Minertia_N,X_N,Kax_N,F_N] =
    assemble_sys_mat(approx_elem, mesh,
    A_el,M_el,Minertia_el,X_el,Kax_el,F_el)

% unpack

perrow = mesh.perrow;
n_el = mesh.n_el;
n_node = mesh.n_node;
lg = mesh.lg;
n2_node = mesh.n2_node;
```

```
lg2 = mesh.lg2;

%
A_N = spalloc(n2_node,n2_node,perrow*n2_node);
M_N = spalloc(n2_node,n2_node,perrow*n2_node);
Minertia_N = spalloc(n2_node,n2_node,perrow*n2_node);
X_N = spalloc(n2_node,n2_node,perrow*n2_node);
Kax_N = spalloc(n2_node,n2_node,perrow*n2_node);
F_N = zeros(n2_node,1);
for i = 1:n_el
    A_N(lg2(:,i),lg2(:,i)) = A_N(lg2(:,i),lg2(:,i)) + A_el(:,:,i);
    M_N(lg2(:,i),lg2(:,i)) = M_N(lg2(:,i),lg2(:,i)) + M_el(:,:,i);
    Minertia_N(lg2(:,i),lg2(:,i)) = Minertia_N(lg2(:,i),lg2(:,i)) +
        Minertia_el(:,:,i);
    X_N(lg2(:,i),lg2(:,i)) = X_N(lg2(:,i),lg2(:,i)) + X_el(:,:,i);
    Kax_N(lg2(:,i),lg2(:,i)) = Kax_N(lg2(:,i),lg2(:,i)) + Kax_el(:,:,i);

    F_N(lg2(:,i)) = F_N(lg2(:,i)) + F_el(:,i);
end

return
end
```

To implement this change, all we must do is pass a parameter k to the function, which denotes the spring constant, and add the following line immediately after the end of the "for" loop.

```
A_N(2*(n_el0 + 1) - 1,2*(n_el0 + 1) - 1) = A_N(2*(n_el0 + 1) -
    1,2*(n_el0 + 1) - 1) + k;
```

We have thus presented and tested a finite element scheme that can be reliably used to design xylophone bars that are tuned to a certain frequency and have a specific first overtone to fundamental frequency ratio. Additionally, we developed a method for finding the least intrusive points through which to pass support strings, and proposed a method to more adequately capture the effects of making these strings go through the xylophone bar.

10 Appendix

In this appendix, we present a set of slides used in a presentation, displaying another application of our developed software involving self-buckling.

Maximizing Height with Volume Constraints

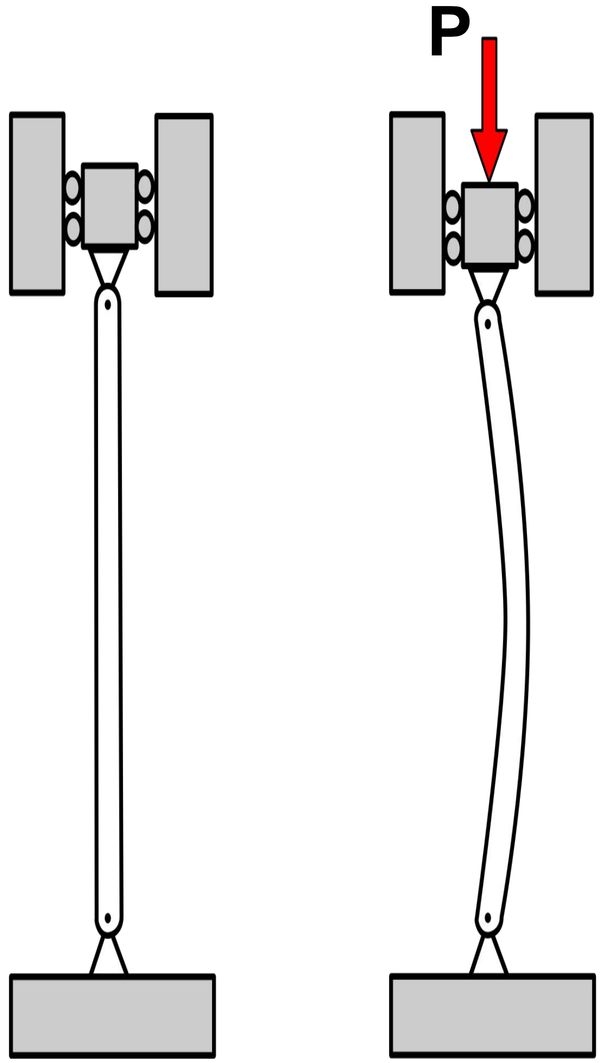
Fidel Cano Rentería

The Problem

Euler Bernoulli Theory:

$$\frac{\partial^2}{\partial x^2}\left(EI(x)\frac{\partial^2 u}{\partial x^2}\right) + \frac{\partial}{\partial x}\left(P(x)\frac{\partial u}{\partial x}\right) = \underbrace{\cancel{q(x,t)}}_{0} - \underbrace{\cancel{\rho(x)\frac{\partial^2 u}{\partial t^2}}}_{0}$$

EI - Stiffness
P - Compressive Axial Force
q - Shear Force Distribution
ρ - Linear Mass Density



The Problem

- Boundary Conditions:

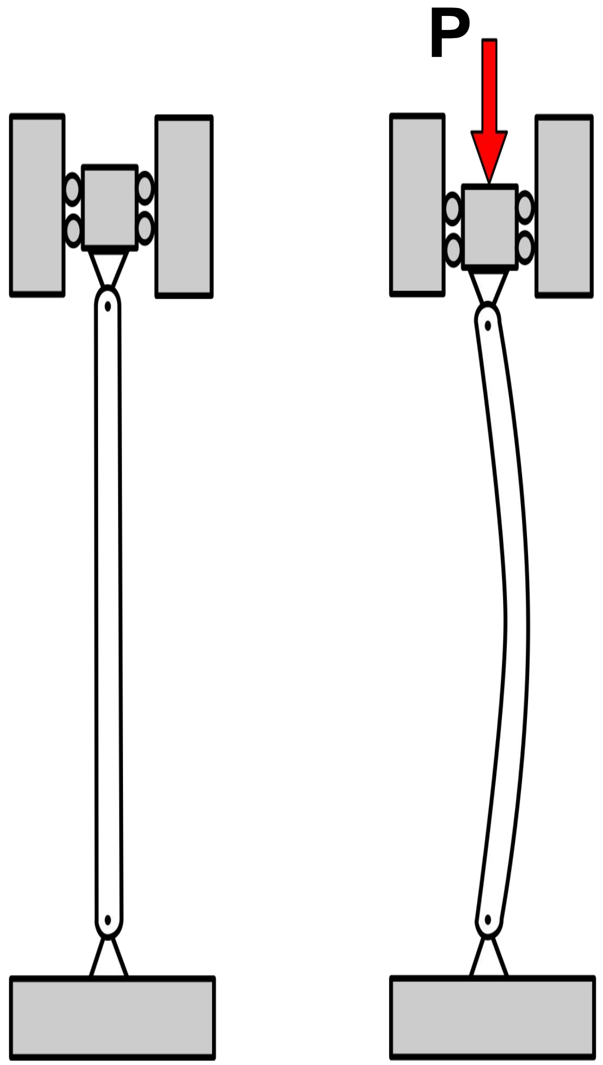
  u(0) = u''(0) = u(L) = u''(L) = 0

- In general, solutions are **not unique.**
- Motivating example: constant circular cross-section

$$\frac{d^4u}{dx^4} + \frac{P}{EI}\frac{d^2u}{dx^2} = 0$$

If $P = \frac{\pi^2 EI}{L^2}$ then adding any multiple of $\sin\left(\sqrt{\frac{P}{EI}}x\right)$

gives **another solution** that satisfies the B.C.'s



The Problem

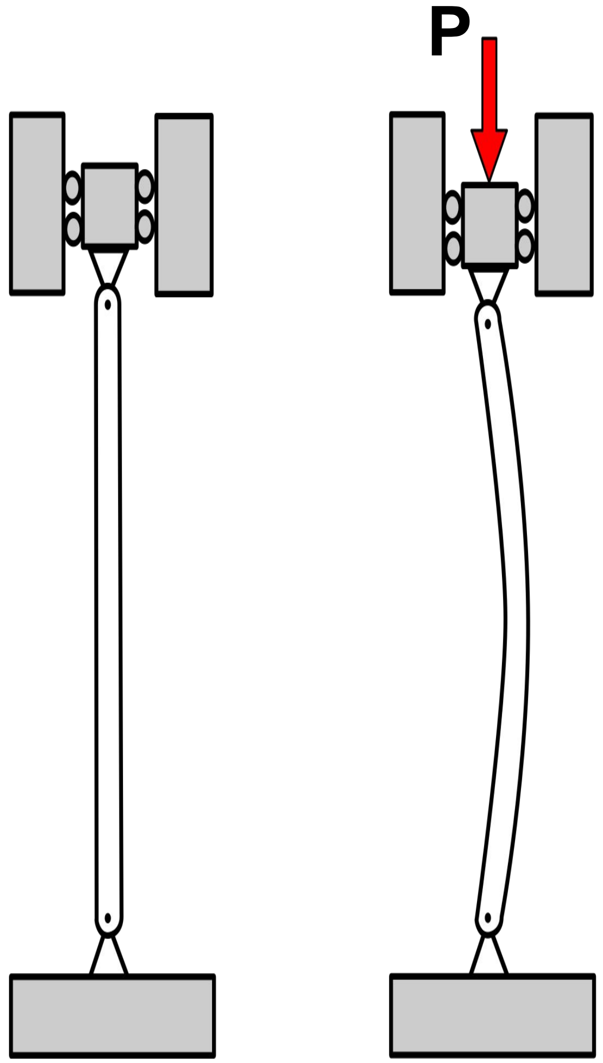
We call $P = \frac{\pi^2 EI}{L^2}$ the critical buckling load.

This expression is specific for the constant circular cross-section case.

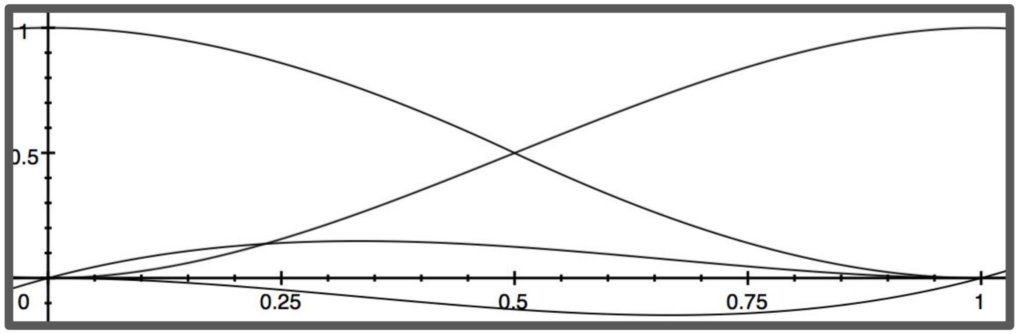
Goal: find the cross-sectional profile that allows for the **largest L** that doesn’t exceeds the critical buckling load **for that shape** loaded under its **own weight**.

$$\frac{\partial^2}{\partial x^2}\left(EI(x)\frac{\partial^2 u}{\partial x^2}\right) + \frac{\partial}{\partial x}\left(P(x)\frac{\partial u}{\partial x}\right) = 0$$

Difficult to solve analytically, so we rely on **FEM.**



Finite Element Representation



$$\mathbf{A}\underline{u} = \lambda \mathbf{K}\underline{u}$$

$$\mathbf{A}_{i,j} = \int_\Omega R^4(x)\,(\varphi_i'')^2(\varphi_j'')^2\,dx$$

$$\mathbf{K}_{i,j} = \int_\Omega P(x)\varphi_i''\varphi_j''\,dx$$

$R(x) = R_d(x)/R_0$

$P(x) = \int_{\Omega(x)} R^2(x')\,dx'$

λ: values of $\mathbf{4\pi\rho g L^4 E^{-1} V^{-1}}$ that make the structure buckle

Optimization Problem

$$\boldsymbol{max}\ \{ \mathbf{L} \} = (\lambda EV(4\pi\rho g)^{-1})^{1/4} \rightarrow \mathbf{max}\ \{ \lambda \}$$

Subject to constant **E, V, ρ,** and **g**

To satisfy the constant volume constraint, we select radii profiles of the form:

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

Where G(x) is a continuous function with the following properties:

- $\int_{\Omega} G(x)\,dx = 0 \rightarrow$ constant volume
- $min_x \{ G \} > -1 \rightarrow$ radius doesn't go to zero anywhere

Proposed G

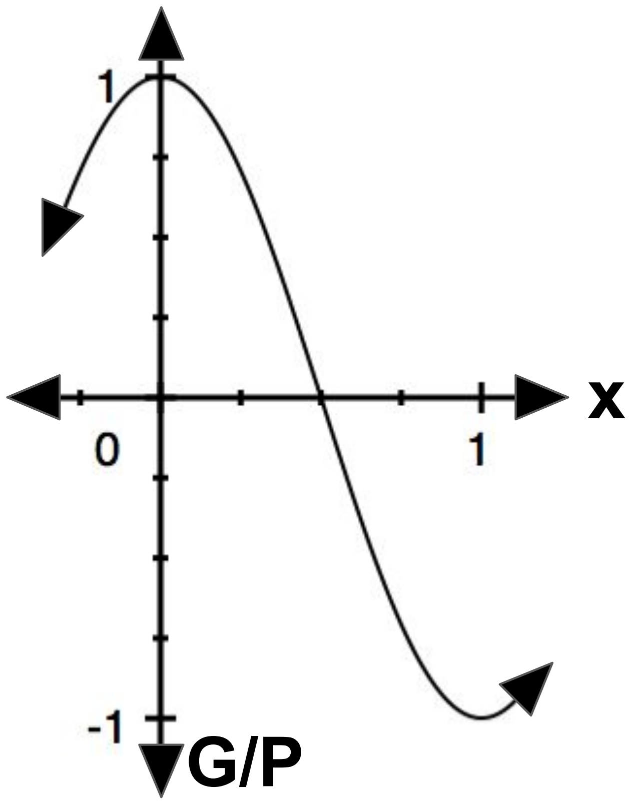
$G(x) = P\cos(\pi x)$

- Continuous
- Symmetric about $x = 0.5$
- Let $P < 1$, and optimize over P

Condition #1 ✓

Condition #2 ✓

Condition #3 ✓

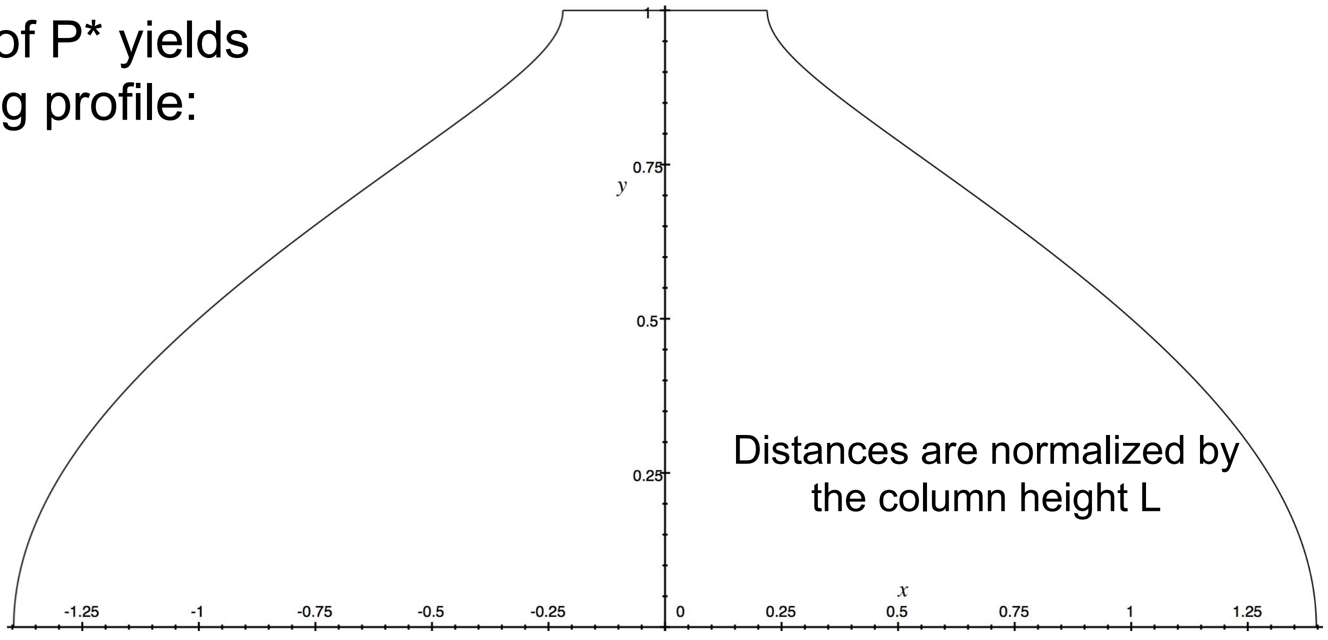


Proposed Profile

Searching over the P range [0,0.96], we find the optimal value of P* to be:

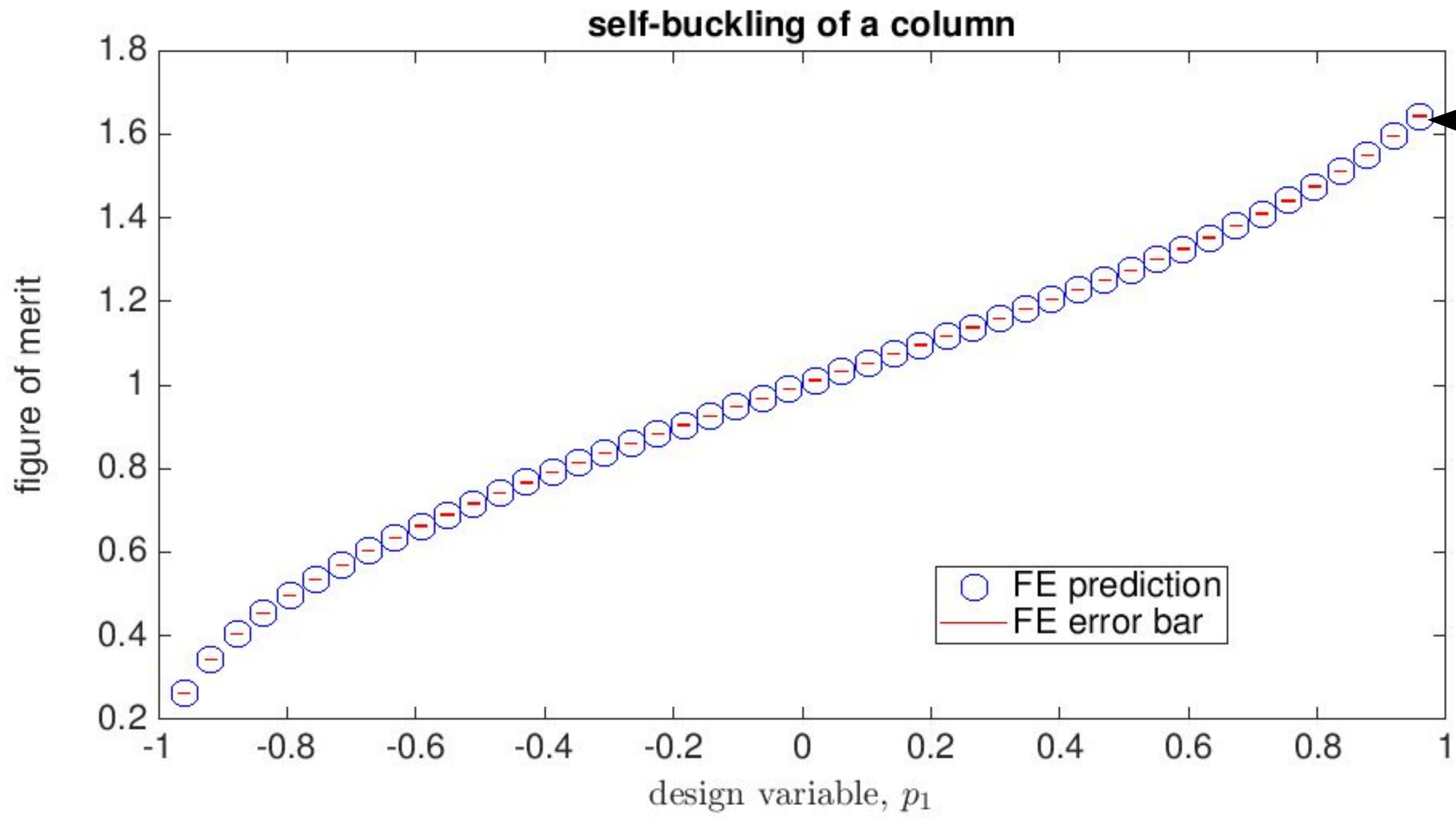
P* = 0.95198

This value of P* yields the following profile:



Distances are normalized by the column height L

Proposed Profile

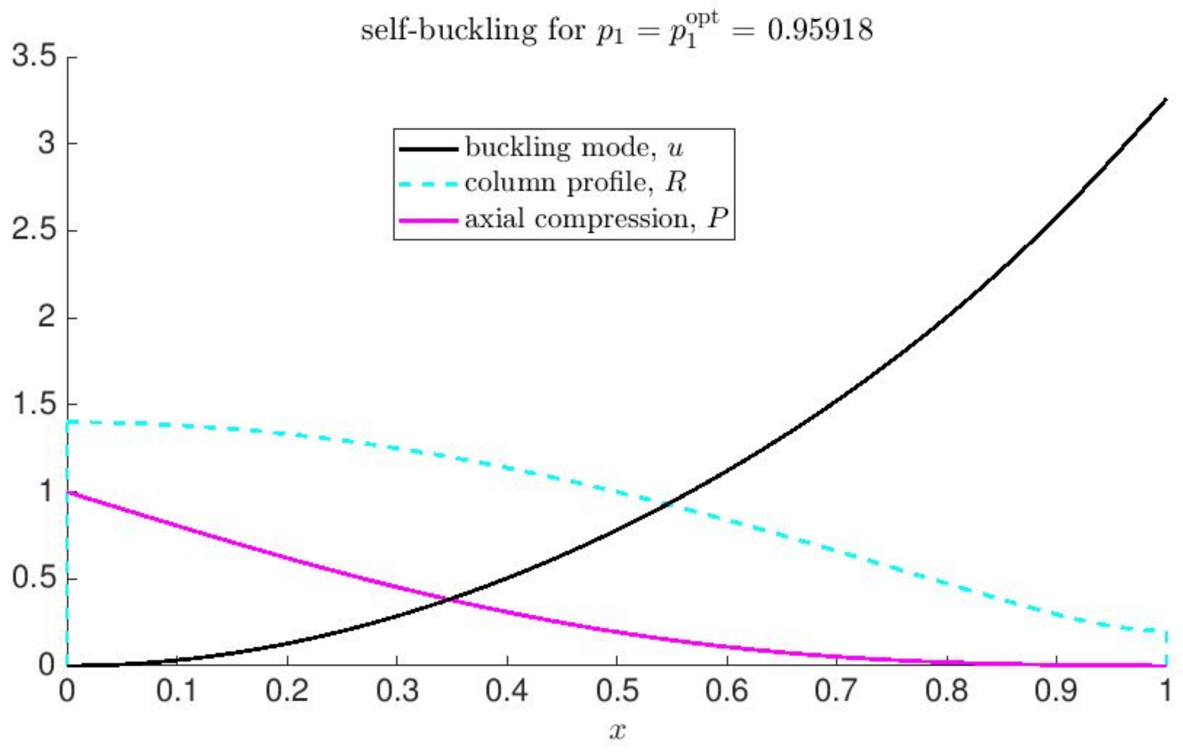


1.6425

That is to say, a structure with our proposed profile can be 1.6425 times higher than a cylinder with the same volume.

Closing Remarks

After the first mesh refinement, the a-posteriori FEM error upper bounds were in the order of 10^{-12}, well below the established tolerance of 0.01



References

[Anand, 2011] Anand, L. (2011). Introduction to the mechanical behavior of materials.

[Caresta, 2014] Caresta, M. (2014). Vibrations of a free-free beam.

[Flay, 2009] Flay, B. (2009). Perfect burger.

[Karnik, 2018] Karnik, R. (2018). *2.006 Fall 2018 Lecture Notes*. Massachusetts Institute of Technology.

[Lacey, 2004] Lacey, M. T. (2004). Carleson's theorem: Proof, complements, variations. *Publicacions Matematiques*, pages 251–307.

[Patera, 2019a] Patera, A. (2019a). Around verification, implementation, and accuracy.

[Patera, 2019b] Patera, A. (2019b). Bending energy formulation.

[Patera, 2019c] Patera, A. (2019c). Bending finite element methods.

[Patera, 2019d] Patera, A. (2019d). Bending natural frequencies.

[Patera, 2019e] Patera, A. (2019e). Bending study case: The xylophone bar.

[Patera, 2019f] Patera, A. (2019f). Finite element 1d symmetric positive-definite boundary value problem formulation.

[Patera, 2019g] Patera, A. (2019g). Finite element 1d symmetric positive-definite boundary value problem implementation.

[Patera, 2019h] Patera, A. (2019h). Finite element 1d symmetric positive-definite boundary value problem theory.

[Patera, 2019i] Patera, A. (2019i). Finite element 1d symmetric positive-definite boundary value problems with p2 elements.

[Patera, 2019j] Patera, A. (2019j). The heat equation: Error estimation.

[Patera, 2019k] Patera, A. (2019k). The heat equation: Formulation and implementation.

[Patera, 2019l] Patera, A. (2019l). The heat equation: Study cases.

[Patera, 2019m] Patera, A. (2019m). Quasi-1d heat transfer.

[Patera, 2019n] Patera, A. (2019n). Rayleigh-ritz method for expositional problem.

[Patera, 2019o] Patera, A. (2019o). Rayleigh-ritz method for general 1d 2nd order symmetric positive-definite boundary value problems.

[Staff, 2013] Staff, H. U. (2013). A summary of error propagation.

[Weinan et al., 2017] Weinan, E., Han, J., and Jentzen, A. (2017). Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations. *Communications in Mathematics and Statistics*, 5(4):349–380.