Multimaterial Heat Flow Verification

Robert L Singleton Jr, Christopher M Malone, and Cora L Brown

Los Alamos National Laboratory
Los Alamos, New Mexico 87545, USA

The University of Minnesota
Minneapolis, Minnesota 55455, USA

(Dated: 625 March 2018)

Abstract
Multimaterial heat diffusion can be a challenging numerical problem when the material boundaries are misaligned with the numerical grid. Even when the boundaries start out aligned, they typically become misaligned through hydrodynamic motion. There are usually a number of methods for handling multimaterial cells in any given hydro code. One of the simplest methods is to replace the multimaterial cell by an average single-material cell whose heat capacity and conductivity are averages over the constituent materials. One can further refine this model by using either the arithmetic or harmonic averages, thereby providing two distinct (albeit naive) multimaterial models for the arithmetic and harmonic averages. More sophisticated models typically involve a surrogate mesh of some kind, as with the thin mesh and static condensation methods. In this paper, we perform rigorous code verification of the multiphysics hydrocode FLAG, including grid resolution studies. We employ a number of newly constructed 2D heat flow solutions that generalize the standard planar sandwich solution, and this paper offers a smorgasbord of exact solutions for heat flow verification. To perform the analyses and to produce the corresponding convergence plots, we employ the code verification tool ExactPack.
## Contents

I. Introduction  

II. The Planar Sandwich Test Problem  

III. Using ExactPack Heat Solvers  

IV. Grid Resolution Studies of the Planar Sandwich  
   A. The Arithmetic Average  
   B. The Harmonic Average  
   C. Thin Mesh and Static Condensation  

V. Generalizations of the Planar Sandwich  
   A. BC2: The Hot and Warm Planar Sandwiches  
   B. The Half Planar Sandwich  

VI. Conclusions and Future Research  
   A. Python script for the Planar Sandwich  
   B. ExactPack Script for Convergence Analysis  
   C. Solution to the Planar Sandwich  
      1. The Static Nonhomogeneous Problem  
      2. The General Homogeneous Problem  
      3. The General Solution  

References
I. INTRODUCTION

This paper is concerned with verifying a number of 2D multimaterial heat-flow algorithms in the multi-physics computational hydrodynamics code FLAG [1]. As a general principle, code verification is the process of comparing and analyzing the differences between numerical results and exact analytic results. Technically, the term exact solution means a solution that can be expressed solely in terms of known analytic functions.\(^1\) These solutions are exceedingly rare, with the Noh problem providing the quintessential example of an exact solution. A more common form of solution is the semi-exact or semi-analytic solution. These solutions can be expressed in terms known analytic functions, supplemented by simple numerical operations, such as 1D quadrature, root finding, numerical ODE solves, or summing infinite series. The planar sandwich solutions are of the latter category. We will use the term exact solution for both cases. The relevance of exact solutions is that their errors can be systematically controlled. Exact solutions usually exploit the symmetry of the problem, such as spherical or planar symmetry, scale invariance, or more general Lie Group symmetries.

We concentrate on an exact 2D solution of the heat flow equation called the planar sandwich [2], performing a series of rigorous convergence analyses. This solution has been analyzed by Dawes, Shashkov, and Malone [3] in the context of multimaterial heat diffusion, although these authors do not present convergence analyses. A number of generalizations of the planar sandwich test problem have been presented in Ref. [5], and we explore these solutions as well. To perform the analyses and to produce the corresponding convergence plots, we employ the code verification tool ExactPack [6]. This paper provides a summary of the planar sandwich solution and its implementation in ExactPack, including the basic Python source code used to produce the various figures and to perform the convergence analyses.

The focus of this paper is the multimaterial heat flow algorithms in FLAG. By a multimaterial cell, we mean a computational cell containing multiple materials, each with their own distinct physical properties, and with a clear interface between the separate materials. Figure illustrates a numerical mesh with multimaterial cells for a square grid based on the planar sandwich geometry. In the context of heat flow, a multimaterial cell contains a number of individual materials labeled by an index \(m\) with heat conductivity \(\kappa_m\). Subgrid models must be employed to resolve such physics in a hydrocode. The simplest subgrid model is obtained by replacing a multimaterial cell by a uniform single-material cell with an average

---

\(^1\) By a known analytic function, we mean a function defined in terms of the conical analytic functions from classical 19th century mathematics. These functions have been exhaustively studied, and have been implemented in hosts of numerical packages.
FIG. 1: An example of a multimaterial grid. The rightmost figure shows a square numerical grid overlaid upon a rectangular physical geometry consisting of three parallel material regions with differing diffusion coefficients. The numerical grid partitions the physical geometry into a number of corresponding numerical cells, none of which need align with the material regions. The outer two regions are composed of the same material with diffusion coefficient $\kappa_1$, while the inner region has diffusion coefficient $\kappa_2$, forming a sandwich-like configuration. The planar sandwich test problem takes the outer two regions (the bread) to be nonconducting, with $\kappa_1 = 0$, while the inner region (the meat of the sandwich) conducts heat with $\kappa_2 \equiv \kappa > 0$. In the Figure, the numerical grid is misaligned relative to the material boundaries of the inner conducting region. This is illustrated in left panel figure.

conductivity $\bar{\kappa}$. An average material is meant to reproduce the collective effects of the individual sub-materials with differing values of $\kappa_m$, and FLAG utilizes both the arithmetic and harmonic averages,

$$\bar{\kappa}_a = \sum_m V_m \kappa_m$$

$$\bar{\kappa}_h^{-1} = \sum_m V_m \kappa_m^{-1},$$

where $V_m$ is the corresponding volume fraction of the cell associated with material $m$ and diffusion coefficient $\kappa_m$. In Fig. 1 the multimaterial index runs over $m = 1, 2$ along the cells containing the material boundary, and for the arithmetic average, the heat diffusion coefficient is $\bar{\kappa}_a = V_1 \kappa_1 + V_2 \kappa_2$; for the harmonic average, the heat diffusion coefficient along the boundary is determined by $\bar{\kappa}_h^{-1} = V_1 \kappa_1^{-1} + V_2 \kappa_2^{-1}$. As we shall see, the arithmetic average overestimates the heat flow along the boundary, while the harmonic average underestimates the heat flow.

2 If appropriate, one can also use the mass fraction $M_m$, rather than the volume fraction $V_m$, to define a mass-weighted average.
Averaging techniques cannot always faithfully represent the physics of multimaterial cells. Consequently, FLAG employs more sophisticated multimaterial diffusion options, namely the thin mesh \cite{7} and static condensation \cite{8} algorithms. The thin mesh method starts with the volume fractions of each material region, and reconstructs the material interfaces using interface reconstruction methods. The mesh is then subdivided along the interfaces, making sure that the final polyhedral mesh conforms with the numerical mesh. This new unstructured mesh is constructed with full connectivity each cycle. The heat diffusion equation is solved on the subdivided mesh containing only single material cells.

The static condensation approach also makes use of the reconstructed material interfaces, but does not require all the details regarding connectivity across material interfaces within a cell. Instead, the global system for the diffusion equation is rewritten in terms of unknown face-centered temperature values. Total flux continuity is enforced at each cell face by ensuring that the sum of the fluxes from all materials on either side of the face are the same. There is an approximation here in that each cell face is assigned a single temperature associated with it; however, the fluxes contain the material-based diffusion coefficient $\kappa_m$, which are allowed to vary. The material-centered temperatures are eliminated from the system (via the Schur Complement), thus condensing the number of degrees of freedom. The global system is then solved for the unknown face temperatures using standard mimetic techniques. The result is that each cell now has a known solution on its boundary (faces), which becomes a local Dirichlet problem that can be solved independently to recover updated material-centered temperatures. For more details on the algorithm, see Ref. \cite{9}; this method is reported to be second order accurate in Ref. \cite{8}.

II. THE PLANAR SANDWICH TEST PROBLEM

This section is devoted to the planar sandwich test problem of Ref. \cite{3}, as illustrated in Fig. 2. It is a 2D-Cartesian heat flow problem on the square domain $D = [0, L] \times [0, L]$, where the $x$-axis runs horizontally and the $y$-axis is vertical. We take $L = 2$ (in arbitrary units) in all figures and examples that follow. The domain $D$ is partitioned into three vertical sandwich-like regions delimited by $x = a_1$ and $x = a_2$, with $0 < a_1 < a_2 < L$. The inner region $a_1 \leq x \leq a_2$ is composed of a heat-conduction material with heat diffusion coefficient $\kappa > 0$, and is called the meat of the sandwich. The meat is surrounded by two non-heat conducting materials, called the bread of the sandwich, for which $\kappa = 0$ in $0 \leq x < a_1$ and $a_2 < x \leq L$. Working in arbitrary temperature units, we wish to solve the 2D heat equation

$$\frac{\partial T}{\partial t} = \nabla \cdot \left[ \kappa \nabla T \right] \text{ for } (x, y) \in (0, L) \times (0, L), \tag{2.1}$$
FIG. 2: The Planar Sandwich is a 2D-Cartesian heat flow problem defined on the square domain $D = [0,L] \times [0,L]$. The temperature at location $(x,y) \in D$ at time $t$ is denoted by $T(x,y,t)$. The inner heat-conducing region $M = \{ (x,y) \in D \mid a_1 \leq x \leq a_2 \}$, for which $\kappa > 0$, is called the meat of the sandwich. The outer non-conducting materials, the bread of the sandwich, are located within $B_1 = \{ 0 \leq x < a_1 \}$ or $B_2 = \{ a_2 < x \leq L \}$, and are insulating, with $\kappa = 0$. In numerical work, we take $\kappa = 10^{-12}$ rather than zero. In setting the boundary conditions, we take the upper and lower boundary temperatures to be constant and uniform along $y = 0$ and $y = L$ with values $T_1$ and $T_2$, respectively, i.e. we impose the Dirichlet boundary conditions $T(x,y = 0,t) = T_1$ and $T(x,y = L,t) = T_2$. On the left and right boundaries, we take the temperature flux in the $x$-direction to vanish, i.e. we use the Neumann boundary conditions $\partial_x T(x = 0,y,t) = 0$ and $\partial_x T(x = L,y,t) = 0$. We must also impose an initial condition: we set the temperature in the interior of the sandwich to zero, i.e. $T(x,y,0) = 0$ for $(x,y) \in (0,L) \times (0,L)$. When $T_1 > T_2$, heat flows in the $y$-direction from the lower to the upper boundary. This allows us to describe the planar sandwich in terms of 1D profiles $T(y,t)$. For numerical work, we take $a_1 = 0.75$ and $a_2 = 1.25$, or a shifted variant with $a_1 = 0.77$ and $a_2 = 1.27$.

where $T = T(x,y,t)$ is the temperature field at position $(x,y)$ and time $t$, with the diffusion coefficient taking the form

$$\kappa(x,y) = \begin{cases} 
0 & ; 0 \leq x < a_1 \\
\kappa & ; a_1 \leq x \leq a_2 \\
0 & ; a_2 < x \leq L.
\end{cases} \quad (2.2)$$

The initial condition (IC) is chosen so that the temperature vanishes in the interior of the domain $D$ at time $t = 0$,

$$T(x,y,0) = 0 \quad \text{for} \quad (x,y) \in (0,L) \times (0,L), \quad (2.3)$$
FIG. 3: The 2D planar sandwich reduces to a 1D heat flow problem along a vertical rod of length $L$. In the top panels, the 2D temperature profile is a function of the $x$ and $y$ coordinates, $T = T(x, y, t)$. Since there is no $x$ dependence in the boundary conditions $T_1$ and $T_2$, the problem reduces to 1D temperature flow along the vertical direction, in which case $T = T(y, t)$ in the central heat-conducting region. To be consistent with the 2D geometrical setup, we will refer to the 1D boundary conditions $T(0, t) = T_1$ and $T(L, t) = T_2$ as the bottom and top boundary condition, respectively.

while the boundary conditions (BCs) are taken to be

$$T(x, 0, t) = T_1 \quad \text{for} \quad x \in [0, L]$$  \hspace{1cm} (2.4)

$$T(x, L, t) = T_2$$  \hspace{1cm} (2.5)

$$\partial_x T(0, y, t) = 0 \quad \text{for} \quad y \in [0, L]$$  \hspace{1cm} (2.6)

$$\partial_x T(L, y, t) = 0.$$  \hspace{1cm} (2.7)

Note that the left- and right-hand sides of the rectangle are insulating, i.e. the temperature flux along the $x$-direction at the far left- and right-ends of $D$ vanishes, $\partial_x T(x = 0, y, t) = 0$ and $\partial_x T(x = L, y, t) = 0$ for $0 \leq y \leq L$. Along the lower and upper boundaries $y = 0, L$, the temperature profiles are uniform in $x$ with $T(x, y = 0, t) = T_1$ and $T(x, y = L, t) = T_2$, where $T_1$ and $T_2$ are constant temperature values over the length of the region $0 \leq x \leq L$. When $T_1 > T_2$, the Second Law of Thermodynamics ensures that heat flows upward from $y = 0$ to $y = L$. We shall generalize the planar sandwich problem in Section V by considering nonhomogeneous boundary condition with nonzero heat flux and non-trivial initial conditions [5].

As illustrated in Fig. 3, the planar symmetry of the problem allows us to express the 2D solution in terms of a 1D profile along the $y$-direction, independent of the $x$-position within the central heat-conducting region $a_1 \leq x \leq a_2$. This is because the upper and lower boundary conditions are uniform along the $x$-direction over the whole range $0 \leq x \leq L$, and
FIG. 4: Temperature profiles $T(y, t)$ for the planar sandwich solution at six representative time slices, $t = 0.001, 0.01, 0.05, 0.1, 0.2, 1$, ranging from early to late times. The heat diffusion coefficient is taken to be $\kappa = 1$, the length of the 1D heat-conducting rod is set to $L = 2$ (the abscissa of the graph), and the first 1000 terms of the series have been summed. We have employed the boundary conditions $T_1 = 1, T_2 = 0$ at $y = 0, y = L$, with the initial condition $T_0 = 0$. Note that the solution at the earliest time $t = 0.001$ has negligible temperature for all $y$ outside a small neighborhood about $y = 0$, as required by the initial condition $T_0 = 0$. Also note that the late-time solution at $t = 1$ is very close to the static equilibrium solution $\bar{T}(y) = T_1 + (T_2 - T_1) y/L$. We shall perform all future verification analyses and convergence plots at time $t = 0.1$. This time is early enough to capture the dynamics of the diffusive heat flow across a large range of $y$, and it is late enough to have a contribution from the static equilibrium solution.

Therefore heat flows only along the vertical direction. Thus, the corresponding 1D heat flow equation for $y$-the profile $T = T(y, t)$ is

$$\frac{\partial T}{\partial t} = \kappa \frac{\partial^2 T}{\partial y^2},$$

(2.8)

where the 1D initial condition (IC) is

$$T(y, 0) = T_0 = 0,$$

(2.9)

and the corresponding 1D boundary conditions (BCs) are

$$T(0, t) = T_1 \quad \text{(bottom)}$$

(2.10)

$$T(L, t) = T_2 \quad \text{(top)}.$$

(2.11)

We refer to the BCs as the bottom and top boundary conditions, respectively, as suggested by the 1D rod in Fig. 3. The exact analytic solution for IC (2.9) and the BCs (2.10)–(2.11)
was presented in Ref. [3], and takes the form

\[
T(y,t) = T_1 + \frac{(T_2 - T_1)}{L} y + \sum_{n=1}^{\infty} B_n \sin(k_n y) e^{-\kappa k_n^2 t}
\]

(2.12)

\[
k_n = \frac{n\pi}{L} \quad \text{and} \quad B_n = \frac{2T_2 (-1)^n - 2T_1}{n\pi}.
\]

(2.13)

The solution profiles \(T(y,t)\) are plotted in Fig. 4 for six time slices \(t = 0.001, 0.01, 0.05, 0.1, 0.2, 1\). The boundary conditions are \(T_1 = 1\) and \(T_2 = 0\), and the initial condition is \(T_0 = 0\). We also take the diffusion constant to be \(\kappa = 1\), the length of the domain to be \(L = 2\), and we sum over the first 1000 terms of the series. As illustrated in Fig. 4, the solution at the earliest time \(t = 0.001\) has a negligible temperature for all values of \(y\) outside a small neighborhood about \(y = 0\). Also note that the late-time solution at \(t = 1\) is very close to the static equilibrium solution

\[
\bar{T}(y) = T_1 + \frac{(T_2 - T_1)}{L} y.
\]

(2.14)

We shall preform the verification analyses and convergence plots at time \(t = 0.1\). The choice \(t = 0.1\) is an intermediate time that is sensitive to both the dynamics of the heat flow and to the late-time equilibrium solution. This choice tests the static boundary conditions and the dynamics of the heat flow solver.

Even though we perform the FLAG simulations in a 2D cartesian geometry, we shall express the numerical solution in terms of its 1D profile. However, before proceeding, it is convenient to visualize the solutions in 2D. The \(x\)- and \(y\)-axes are divided into \(N\) segments, with \(N = 5, 10, 20, 40, 80, 160\). This gives the six grid spacings \(h = 0.5, 0.2, 0.1, 0.05, 0.025, 0.0125\). Note that heat flows outside the boundary region, particularly at low resolutions. Figures 5 and 6 illustrate the arithmetic and harmonic average models for the 2D numerical FLAG solutions, for six levels of increasing resolution. To better understand these Figures, we examine the arithmetic technique in more detail. We shall see that the arithmetic average has the effect of extending the inner material beyond the boundary, while the harmonic average decreases the inner material within the boundary. For the planar sandwich geometry illustrated in Fig. 1, the material boundaries are static with \(x = a_1\) and \(x = a_2\). The arithmetic average of the diffusion coefficient on a boundary cell is

\[
\bar{\kappa}_a = V_1 \kappa_1 + V_2 \kappa_2 = V_1 \epsilon + V_2 \kappa \to V_2 \kappa > 0,
\]

(2.15)

and we should expect the arithmetic average to overestimate the effects of multimaterial heat flow along the boundary cells.
FIG. 5: Illustration of the 2D planar sandwich solution employing the arithmetic average multimaterial algorithm for six resolutions. The $x$- and $y$-axes are divided into $N$ segments, with $N = 5, 10, 20, 40, 80, 160$. For $L = 2$ this gives the six resolutions $h = 0.5, 0.2, 0.1, 0.05, 0.025, 0.0125$. Note that heat flows outside the central boundary region $a_1 = 0.77 \leq x \leq 1.27 = a_2$. This is particularly noticeable at low resolutions.

FIG. 6: Illustration of the 2D planar sandwich solution employing the harmonic average multimaterial algorithm for six resolutions. The $x$- and $y$-axes are divided into $N$ segments, with $N = 5, 10, 20, 40, 80, 160$. For $L = 2$ this gives the six resolutions $h = 0.5, 0.2, 0.1, 0.05, 0.025, 0.0125$. Note that the heat is underestimated in the multimaterial regions.
Conversely, the harmonic average gives
\[ \bar{\kappa}_h^{-1} = V_1 \kappa_1^{-1} + V_2 \kappa_2^{-1} = V_1 \epsilon^{-1} + V_2 \kappa^{-1} \rightarrow V_1 \epsilon^{-1} \quad \text{or} \quad \bar{\kappa}_h \rightarrow 0, \] (2.16)
which underestimates the effects of multimaterial heat flow along the boundary cells. The numerical FLAG results of Ref. [3] indeed show that the arithmetic mean emphasizes larger values of the diffusion coefficient, while the harmonic mean emphasizes smaller values.

III. USING EXACTPACK HEAT SOLVERS

In this section we present an example of how to use ExactPack [6] to perform code verification for the exact solution of the planar sandwich.\(^3\) One starts by importing the planar sandwich solution module into Python,

```python
from exactpack.solvers.heat import PlanarSandwich
```

As discussed in the last section, the exact solution for the 2D planar sandwich can be described in terms of a 1D rod of length \( L \) with a uniform heat diffusion coefficient \( \kappa \). The 1D solution is implemented in ExactPack by the solver `PlanarSandwich`. The planar sandwich class comes with a number of default settings for the input parameters, such as the length of the rod, the value of the heat diffusion coefficient, settings for the boundary and initial conditions, and the number of terms to be summed in the series. To instantiate and use the `PlanarSandwich` class with default values, one invokes

```python
solver = PlanarSandwich()
```

The parameter values can be explicitly set by

```python
solver = PlanarSandwich(T1=1, T2=0, L=2, kappa=1, Nsum=1000)
```

This creates an ExactPack object called `solver` with boundary conditions \( T_1 = 1 \) and \( T_2 = 0 \), the length of the 1D rod set to \( L = 2 \), the diffusion coefficient set to \( \kappa = 1 \), and we have specified that we want to retain the first 1000 terms in the summation. The default initial condition is \( T_0 = 0 \); this can be changed by setting the variables `TA` and `TB`. Access to all the properties of the planar sandwich definition can be controlled through the class `PlanarSandwich`.

The planar sandwich object encapsulates the definition of the problem, but it is unaware of the spatial grid necessary for a specific realization of the problem. Therefore, a `solver` object must be used to produce a `solution` object on a spatial array at a given time \( t=0.1 \). The corresponding Python code is

---

\(^3\) ExactPack is an open source project available at [https://github.com/lanl/ExactPack](https://github.com/lanl/ExactPack)
from exactpack.solvers.heat import PlanarSandwich
import numpy as np

solver = PlanarSandwich(T1=1, T2=0, L=2)
L = 2
y = np.linspace(0, L, 1000)
t1 = 1.0
t2 = 0.2
t3 = 0.1
t4 = 0.05
t5 = 0.01
t6 = 0.001

soln1 = solver(y, t1)
soln2 = solver(y, t2)
soln3 = solver(y, t3)
soln4 = solver(y, t4)
soln5 = solver(y, t5)
soln6 = solver(y, t6)

soln1.plot('temperature')
soln2.plot('temperature')
soln3.plot('temperature')
soln4.plot('temperature')
soln5.plot('temperature')
soln6.plot('temperature')

FIG. 7: An illustration of the Python source code used to plot the planar sandwich contour solutions of Fig. 4. The Python script to produce this figure is provided in Appendix A. The planar sandwich solver has been instantiated by `solver = PlanarSandwich(T1=1, T2=0, L=2)`, which sets the boundary conditions to $T(0,t) = 1$, $T(L,t) = 0$, the initial condition to $T(y,0) = 0$, and the length of the rod (along the $y$-direction) to $L = 2$. The values $\kappa = 1$ and $N_{\text{sum}} = 1000$ are the default settings, and are not specified in the above code segment. The solver is used to form a solution object from a spatial array $y$ and a time $t$ by `soln = solver(y, t)`.

```
solver = PlanarSandwich()

L = 2
y = numpy.linspace(0, L, 1000)
t = 0.1

soln = solver(y, t)
soln.plot('temperature')
```

The solver object `solver` takes a spatial array $y$ and a time variable $t$, and produces a solution object called `soln` with the exact solution evaluated on the spatial array at the given time. As illustrated by the last line above, a solution object is equipped with a plotting method, in addition to various analysis methods not shown here. The Python script that produces Fig. 4 is given in Appendix A and is summarized in Fig. 7.
IV. GRID RESOLUTION STUDIES OF THE PLANAR SANDWICH

In the previous sections, we examined the planar sandwich test problem in some detail, in particular, we provided the exact solution in a semi-analytic form in Eq. (2.12). The geometry of the planar sandwich is illustrated in Fig. 1 and the material interfaces and their heat conduction properties are defined in Fig. 2. In this section, we build on these results by performing rigorous convergence analyses for the four primary multimaterial heat flow algorithms in FLAG, namely, (i) the arithmetic average, (ii) the harmonic average, (iii) thin mesh, and (iv) static condensation. As we have already emphasized, the analyses are performed at time $t = 0.1$, and the domain of the planar sandwich is the two dimensional region $D = [0, L] \times [0, L]$. In numerical simulations we take $L = 2$, partitioning the domain $D$ into $N \times N$ square cells with sides of length $h = L/N$. In other words, the 2D computational grid is formed by dividing the $x$- and $y$-grids into $N$ equal segments of length $h$, thereby creating $N^2$ square cells with sides of length $\Delta x = \Delta y = h$. It should be noted that the algorithms in FLAG do not require the cells to be square, and all multimaterial methods work on general polytopal meshes. We only use square cells to provide a unique length scale $h$ with which to plot the norms. In all numerical simulations, we take the number of segments to increase by a factor of two, starting with five segments for the lowest resolution and ending with 640 for the highest resolution,

$$N = 5, 10, 20, 40, 80, 160, 320, 640.$$

For $L = 2$, the square cells have sides of length

$$h \equiv L/N = 0.4, 0.2, 0.1, 0.05, 0.025, 0.0125, 0.00625, 0.003125.$$

It is important to note that in all numerical simulations, we halve the maximum time step for every doubling in $N$. The central heat-conducting material, the meat of the sandwich, is the 2D region $M = \{(x, y) \in D \mid a_1 \leq x \leq a_2 \text{ and } 0 \leq y \leq L\}$, within which the heat diffusion coefficient takes the value $\kappa = 1$. The outer two materials $B = \{(x, y) \in D \mid 0 \leq x < a_1 \text{ or } a_2 < x \leq L \text{ with } 0 \leq y \leq L\}$ are called the bread of the sandwich, and are composed of an insulated material for which $\kappa = 0$. In our numerical simulations, we do not actually take the heat diffusion coefficient to vanish inside $B$, but rather, we set $\kappa = \epsilon \equiv 10^{-12}$. We should therefore think of the condition $\kappa = 10^{-12}$ as a limiting procedure in which $\kappa = \epsilon$ with $\epsilon \to 0^+$. 

13
FIG. 8: The numerical FLAG runs for the planar sandwich at time \( t = 0.1 \) for the arithmetic average technique. The exact solution is plotted in black, although it is difficult to resolve against the dense numerical background. We have performed numerical runs for \( N = 5, 10, 20, 40, 80, 160, 320, 640 \). Choosing length \( L = 2 \), the corresponding resolutions are \( h \equiv N/L = 0.4, 0.2, 0.1, 0.05, 0.025, 0.0125, 0.00625, 0.003125 \). For every resolution \( h \), the numerical solution can be expressed by the triplets \((x_i, y_j, T_{ij})\) for \( i, j = 1, 2 \cdots N \). We have projected out the \( x \)-coordinates, plotting \((y_j, T_{ij})\) in the Figure. At high resolutions, the solution becomes independent of \( x_i \), and indeed, as the grid resolution increases, the points become more closely spaced, and lie closer to the exact solution. For these numerical runs, we have taken the conducting region (the meat of the sandwich) to be \( a_1 = 0.75 \leq x \leq 1.25 = a_2 \). The numerical results for the shifted region \( a_1 = 0.77 \leq x \leq 1.27 = a_2 \) are qualitatively similar in appearance.

A. The Arithmetic Average

In this section we perform the convergence analysis for the arithmetic average multimaterial algorithm in FLAG. We first consider the case in which \( a_1 = 0.75 \) and \( a_2 = 1.25 \) (with \( L = 2 \)). At grid point \((x_i, y_j) \in D\) and time \( t = 0.1\), the numerical algorithm returns a temperature \( T_{ij} \), and therefore, the numerical results can be expressed by the \( N^2 \) triplets \((x_i, y_j, T_{ij})\) for \( i, j = 1, 2 \cdots N \). In Fig. 8, we have projected out the \( x \)-coordinate and plotted the points \((y_j, T_{ij})\) at time \( t \). Note that the numerical solutions becomes more finely spaced in \( y \) with increasing \( x \)-resolution, until the points \((y_j, T_{ij})\) lie on top of the exact 1D profile \( T(y, t) \). In the Fig. 8, the exact solution is the solid black line, although it is difficult to resolve against the dense numerical background.

Let us examine the ExactPack script used to produce Fig. 8. ExactPack contains an object called Study that, among other things, can be used to plot the numerical data alongside the exact solution:
study = Study(datasets=dumpfiles,
                reference=PlanarSandwich(),
                study_parameters=[0.4, 0.2, 0.1, 0.05, 0.0125, 0.00625, 0.003125],
                time=0.1,
                reader=FlagVarDump(),
                abscissa='y_position'
            )

study.plot('temperature')

The statement `study=Study(···)` instantiates the object `Study` by the instance `study`, where the latter inherits its properties and methods from the former. The study object contains a plot method, `study.plot('temperature')`, which instructs the object to plot itself, thereby producing Fig. 8. The object `Study()` takes a number of arguments. The first argument `datasets` has been assigned the value `dumpfiles`, which is a regular expression for the path of the code output. The output consists of separate code runs at the resolutions specified by `study_parameters` and at the time specified by `time`. The argument `reference` is used to select the reference solver in ExactPack, which in this case is `PlanarSandwich`. The argument `reader` provides the interface between the code output and ExactPack. In this case, the code reader is specific to FLAG; however, ExactPack will soon use the VTK format by default.

In general, the numerical output can always be expressed in the form \((x_i, y_j, T_{ij})\) with \(i, j = 1, \ldots, N\), for every resolution \(N\), and the final argument `abscissa='y_position'` specifies that only data only along the \(y\)-direction be used in the analyses. With this setting, the analyses and figures are performed using the 1D profile representation. The ExactPack script and corresponding Figure are summarized in Fig. 9 with the upper left margin specifying the ExactPack script.

The \(L_1\) error norm is defined by

\[
L_1 = \sum_{i,j=1}^{N} \left| T_{ij}^{\text{num}} - T_{ij}^{\text{ex}} \right|,
\]

(4.3)

where \(T_{ij}^{\text{num}} = T_{ij}\) is the numerical solution at position \((x_i, y_j)\) and time \(t\), and \(T_{ij}^{\text{ex}} = T(x_i, y_j, t)\) is the corresponding exact solution. We remind the reader that we take \(t = 0.1\) in all numerical analyses, simulations, and figures. We can define a restricted metric \(L_1(R)\) over a subset \(R \subset D\), such that

\[
L_1(R) = \sum_{ij,(x_i,y_j) \in R} \left| T_{ij}^{\text{num}} - T_{ij}^{\text{ex}} \right|.
\]

(4.4)
FIG. 9: A summary of the Python source code used to produce Fig. [8]. The exact solution is plotted in black, although it is hard to resolve against the dense numerical background. The study object contains a plotting method that plots the exact solution alongside the numerical results.

ExactPack currently only supports the data format of the 1D profile along $y$; therefore, the norms are only calculated over 1D regions $R \subset (0, L)$ along the $y$-axis,

$$L_1(R) = \sum_{ij, y_j \in R} \left| T_{ij}^{\text{num}} - T_{ij}^{\text{ex}} \right|. \tag{4.5}$$

Since the temperature vanishes inside region $B$, the only sizable contributions to $L_1$ occur for $(x_i, y_j) \in M$.

Figure [10] illustrates the convergence analysis for the planar sandwich test problem in FLAG, where we have plotted the $L_1$ norm against the resolution $h$ on a log-log scale. For the heat-conducting region delimited by $a_1 = 0.75 \leq x \leq 1.25 = a_2$ with $L = 2$ and $N = 5$, the 4th mesh refinement has resolution $h = 0.05$, at which point the numerical grid aligns with the material boundaries at $x = a_1$ and $x = a_2$. At the alignment, the accuracy of the code increases by over an order of magnitude, thereby giving rise to a discontinuity in the convergence plot. Note, however, that the upper and lower branches converge at approximately the same rate $p = 1.2$. As shown in Fig. [11], we can eliminate the discontinuity by slightly shifting the conduction region to $a_1 = 0.77 \leq x \leq 1.27 = a_2$, in which case the grid points of the shifted region never align with a material interfaces. As we see in the lower panel of the Figure, the convergence rate is $p = 1.0$. In everything that follows, we use the
FIG. 10: Convergence analysis at time $t = 0.1$ for the planar sandwich using the arithmetic average treatment of multimaterial cells. The length of the domain is $L = 2$ with eight uniform regions $N = 5, 10, 20, 40, 80, 160, 320, 640$, corresponding to resolutions $h = 0.4, 0.2, 0.1, 0.05, 0.025, 0.0125, 0.00625, 0.003125$. Since $a_1 = 0.75 \leq x \leq 1.25 = a_2$ with $L = 2$, the numerical grid and the material interface align on the $4$th iteration at $h = 0.05$. When the mesh aligns with the material boundary, the accuracy increases by an order of magnitude, producing a discontinuity in the convergence graph. However, both the upper and lower branches converge at approximately the same rate.

shifted region by default. Before continuing, we provide a brief summary of the ExactPack code used to create the convergence study of Fig. 10:

```python
domain = (0, L)
fiducials = {'temperature': 1}
fit = RegressionConvergenceRate(study, domain=domain, fiducials=fiducials)
fit[0:3].plot_fit('temperature', '-', c='b')
fit[3:8].plot_fit('temperature', '-', c='r')
fit.norms.plot('temperature')
fit.plot_fiducial('temperature')
fit.plot('temperature')
```

The above script starts with the assignment statement `domain = (0, L)`, which specifies the region along $y$ over which to calculate the norm, where, as usual, we take $L = 2$. The command `fiducials = {'temperature': 1}` defines a Python dictionary that sets the slope of the fiducial triangle to unity. The next line, `fit = RegressionConvergenceRate(···)`, instantiates the convergence rate object `RegressionConvergenceRate` by the instance `fit`. The object `RegressionConvergenceRate` performs a convergence analysis based on an error Ansatz $L_1(h) = Ah^p$, and it takes the following arguments: the object `study` discussed above, the domain over which the norm is calculated, and the fiducial dictionary. The command `fit.plot_fit` exercises the method `plot_fit`, which uses the matplotlib.pyplot.plot routine to plot the error norms and the best fit convergence rate for the error Ansatz. Other error Ansatze can be used if desired. Since the convergence rate is discontinuous at the $4$th
The convergence analysis in FLAG at time $t = 0.1$ for the planar sandwich using the arithmetic average treatment of multimaterial cells gives $p = 1.0$. The length of the domain is $L = 2$ with eight uniform regions $N = 5, 10, 20, 40, 80, 160, 320, 640$, corresponding to resolutions $h = 0.4, 0.2, 0.1, 0.05, 0.025, 0.0125, 0.00625, 0.003125$. The central region has been shifted to $a_1 = 0.77 \leq x \leq 1.27 = a_2$, so that the mesh refinements are never commensurate with this region. The convergence plot is then rendered continuous.

Iteration in resolution, we use the nomenclature `fit[0,3]` and `fit[3,8]` to perform the fits on the first three and last five data sets independently, thereby giving convergence rates for the upper and lower branches. It is necessary to do an explicit `matplotlib.pyplot.show` to display the plot, `fit.plot('temperature')`. Figure 12 summarizes the results of this section.

### B. The Harmonic Average

We now perform a convergence analysis for the harmonic average multimaterial model in FLAG. As discussed in the previous section, it is convenient to use the shifted heat-conducting region delimited by $a_1 = 0.77 \leq x \leq 1.27 = a_2$, where the length of the domain is take to be $L = 2$. In this way, the numerical grid never aligns with the material interfaces at $x = a_1$ and $x = a_2$, thereby rendering the convergence plot continuous. Figure 13 illustrates the convergence analysis for the harmonic average. The upper panel of the Figure plots the numerical solutions $(y_j, T_{ij})$ alongside the exact 1D profile, and the lower panel gives the corresponding convergence analysis. Note that the first two points, $N = 5, 10$, converges at a rate $p = 0.4$, while $N = 20, 40, \ldots, 640$ converges approximately linearly with $p = 0.95$, indicating that the first two points lie outside the asymptotic range of convergence. Comparing the upper panels of Figs. 12 and 13, we see that the harmonic average solutions do not have the same degree of scattering as the arithmetic average, and they lie much closer to the exact analytic profile. Note, however, that the harmonic average solutions contain a
FIG. 12: The arithmetic average for multimaterial cells. Numerical profiles at time $t = 0.1$ are shown in the top panel, and the convergence analysis in the bottom panel, corresponding to Figs. [10] and [11]. The length of the domain is $L = 2$ for eight uniform segments with $N = 5, 10, 20, 40, 80, 160, 320, 640$, corresponding to resolutions $h = 0.4, 0.2, 0.1, 0.05, 0.025, 0.0125, 0.00625, 0.003125$. The central heat-conducting region has been shifted to $a_1 = 0.77 \leq x \leq 1.27 = a_2$, so that the mesh refinements never align with a material boundary. The convergence rate for the arithmetic average is approximately $p = 1$.

spurious branch along the horizontal axis. This arises because of a bug in FLAG, which fails to propagate the heat flow on a small number of discrete grid points.

C. Thin Mesh and Static Condensation

We now turn to the more advanced multimaterial algorithms of FLAG, namely, thin mesh [7] and static condensation [8]. As discussed in the Introduction, the thin mesh algorithm uses the volume fractions of each material to reconstruct the material interfaces by employing interface reconstruction methods. The mesh is then subdivided along the interfaces, making sure that the final polyhedral mesh conforms with the numerical mesh. This method is quite accurate but very time consuming. Figure [14] illustrates the analysis, with the upper panel showing the numerical solutions plotted alongside the exact solution, and the lower panel giving the convergence plot. Note that thin mesh starts out with 2nd order convergence, $p = 2$, and becomes lower order order as the mesh resolution is refined, eventually giving $p = 1$ at the smallest resolutions. There are a number of possible reasons why the
FIG. 13: The harmonic average for multimaterial cells. The parameter setting are the same as in Fig. 12.

FIG. 14: The thin mesh option for multimaterial cells. The parameter setting are the same as in Fig. 12.
convergence rate levels off as the grid is refined, and we are currently investigating this. One possibility is that the order of accuracy is limited by the interface reconstruction algorithm critical to both of these methods.

The static condensation approach also makes use of the reconstructed material interfaces, but does not require the connectivity information across material interfaces within a cell. The global system for the diffusion equation is rewritten in terms of unknown face-centered temperature values, and flux continuity is enforced at each cell face by ensuring that the sum of the fluxes from all materials on either side of the face are the same. The global system is then solved for the unknown face temperatures. The result is that each cell now has a known solution on its boundary (faces), which becomes a local Dirichlet problem that can be solved independently to recover updated material-centered temperatures. For more details on the algorithm, see Ref. [9]; this method is reported to be second order accurate in Ref. [8]. Figure 15 illustrates the analysis for static condensation, which exhibits quantitatively similar scaling to thin mesh. In fact, for the planar sandwich test problem, static condensation agrees with thin mesh almost exactly.
V. GENERALIZATIONS OF THE PLANAR SANDWICH

As we have seen, the 2D planar sandwich test problem can be expressed in terms of heat flow along a 1D rod oriented in the horizontal direction, as illustrated in Fig. 3. The planar sandwich is a special case of the more general heat flow problem \[11\],

\[
\frac{\partial T(y,t)}{\partial t} = \kappa \frac{\partial^2 T(y,t)}{\partial y^2} \quad (5.1)
\]

\[
\alpha_1 T(0,t) + \beta_1 \partial_y T(0,t) = \gamma_1 \quad (5.2)
\]

\[
\alpha_2 T(L,t) + \beta_2 \partial_y T(L,t) = \gamma_2 \quad (5.3)
\]

\[
T(y,0) = T_0(y) \quad . \quad (5.4)
\]

This problem is implemented by the ExactPack class Rod1D. We use an arbitrary but consistent set of temperature units. Equation (5.1) is a diffusion equation describing the temperature response to heat flow in a material with constant diffusivity. The next two equations, Eqs. (5.2) and (5.3) are the boundary conditions (BCs), which we take to be nonhomogeneous linear combinations of Dirichlet and Neumann conditions. The initial condition (IC) is given by Eq. (5.4), and specifies the \( t = 0 \) temperature profile along the rod. When the right-hand sides of the BCs vanish, \( \gamma_1 = \gamma_2 = 0 \), the problem is called homogeneous, otherwise the problem is called nonhomogeneous. The distinction between homogeneous and nonhomogeneous solutions has far-reaching implications for how one goes about solving the heat problem. The special property of homogeneous solutions is that the sum of any two homogeneous solutions is another homogeneous solution. However, such linearity is not true for nonhomogeneous solutions, as adding two nonzero values of \( \gamma_i \) violates the BCs.

Finding a solution to the general problem (5.1)–(5.4) involves solving both the homogeneous and nonhomogeneous problems. The general homogeneous solution, for which \( \gamma_1 = \gamma_2 = 0 \), will be denoted by \( \tilde{T}(y,t) \). To construct the exact solution, we must also find a specific static solution to the nonhomogeneous problem, which we denote by \( \bar{T}(y) \).\(^4\) The general solution therefore takes the form

\[
T(y,t) = \tilde{T}(y,t) + \bar{T}(y) \quad . \quad (5.5)
\]

The homogeneous solution \( \tilde{T}(y,t) \) will be expressed as a Fourier series, and its coefficients will be chosen so that the initial condition (5.4) is satisfied by \( T(y,t) \). This means that we choose the Fourier coefficients of \( \tilde{T} \) such that

\[
\tilde{T}(y,0) = T_0(y) - \bar{T}(y) \quad . \quad (5.6)
\]

\(^4\) We choose the strategy of finding a static \( \bar{T}(y) \) because this is usually easier than finding a time-dependent specific solution.
The static nonhomogeneous solution $\bar{T}(y)$ is itself linear in $y$, and takes the form

$$\bar{T}(y; T_1, T_2) = T_1 + \frac{T_2 - T_1}{L} y,$$

where the temperatures $T_1$ and $T_2$ are defined in terms of the parameters $\alpha_i$, $\beta_i$, and $\gamma_i$ for $i = 1, 2$. Although there exists a solution to the heat flow equations for any continuous initial condition $T_0(y)$, it is convenient for our purposes to consider only the linear initial condition

$$T_0(y; T_A, T_B) = T_A + \frac{T_B - T_A}{L} y,$$

for independent temperature parameters $T_A$ and $T_B$. This form can also be used for the flux boundary condition $\partial_y T(L) = F_B$ by setting $T_B = F_B L$. Therefore, (5.6) can be written as

$$\tilde{T}(y, 0) = T_0(y) - \bar{T}(y)$$

$$= T_a + \frac{T_b - T_a}{L} y,$$

where

$$T_a = T_A - T_1$$

$$T_b = T_B - T_2.$$  

By assuming a linear IC, we find that the boundary conditions and initial conditions can be interchanged according to (5.11) and (5.12). It would be an interesting verification exercise to observe the extent to which this holds true in a code, since code algorithms handle boundary and initial conditions quite differently.

Let us briefly discuss some basic issues involving *uniform convergence* [12]. This is related to order-of-limits questions in classical mathematics, which have direct practical implications for many mathematical systems of interest. The parameters $T_\lambda$ and $T_B$ specify the initial temperatures at the bottom and top of the rod, or the bottom and top of the 1D slice in Fig. 3. By *bottom* and *top*, we really mean the limits $y = 0^+$ and $y = L^-$, respectively. In other words, the linear initial condition $T_0(y)$ is defined only on the open interval $(0, L)$, with $T_\lambda$ being the value of $T_0(y)$ as $y \to 0^+$ and $T_B$ being the value of $T_0(y)$ as $y \to L^-$. The temperatures $T_\lambda$ and $T_B$ need not be equal to the boundary conditions $T_1$ and $T_2$. If the BCs and the IC do not agree, then the solution is nonuniformly convergent for $t \to 0$.

As an example, consider BC1 with $T_1 \neq T_\lambda$ and $T_2 \neq T_B$. The profile $T(y, t)$ converges point-wise to the initial profile $T_0(y)$ as $t$ goes to zero over the domain $(0, L)$, that is to say, the temperature $T(y, t) \to T_0(y)$ as $t \to 0$ for all $y \in (0, L)$. However, this point-wise convergence in $y$ is *nonuniform* on the closed interval $[0, L]$, in that $T(0, t) = T_1 \neq T_\lambda$. 

23
although \( \lim_{y \to 0^+} T(y,t) = T_\lambda \neq T_1 = T(0,t) \). Similarly, \( \lim_{y \to L^-} T(y,t) \neq T(L,t) \). These conditions place a limit on how close to \( t = 0 \) one can set the time \( t \) in convergence plots. See Ref. [12] for an introductory but solid treatment of real analysis and nonuniform convergence.

The boundary conditions (5.2) and (5.3) are specified by the coefficients \( \alpha_i, \beta_i, \) and \( \gamma_i \) for \( i = 1, 2 \). These parameters are not all independent, and various combinations will produce the same temperatures \( T_i \) and fluxes \( F_i = \partial_y T_i \); therefore, it is often more convenient to specify the boundary conditions directly in terms of \( T_i \) and \( F_i \). For example, if \( \beta_1 = 0 \) in (5.2), then the boundary condition becomes \( \alpha_1 T(0,t) = \gamma_1 \), which we can rewrite in the form \( T(0,t) = T_1 = \gamma_1 / \alpha_1 \). There are four special boundary conditions that provide particularly simple solutions. The first class is specified by \( \beta_1 = \beta_2 = 0 \) with \( \alpha_i \neq 0 \), and gives the Dirichlet boundary conditions

\[
\text{BC1} \\
T(0,t) = T_1 : \quad \alpha_1 \neq 0 \quad \beta_1 = 0 \quad T_1 = \frac{\gamma_1}{\alpha_1} \tag{5.13} \\
T(L,t) = T_2 : \quad \alpha_2 \neq 0 \quad \beta_2 = 0 \quad T_2 = \frac{\gamma_2}{\alpha_2} . \tag{5.14}
\]

The planar sandwich of the previous sections is a subclass of BC1. The next class of boundary conditions is obtained by by setting \( \alpha_1 = \alpha_2 = 0 \) with \( \beta_i \neq 0 \), and this gives the Neumann boundary conditions

\[
\text{BC2} \\
\partial_y T(0,t) = F_1 : \quad \alpha_1 = 0 \quad \beta_1 \neq 0 \quad F_1 = \frac{\gamma_1}{\beta_1} \tag{5.15} \\
\partial_y T(L,t) = F_2 : \quad \alpha_2 = 0 \quad \beta_2 \neq 0 \quad F_2 = \frac{\gamma_2}{\beta_2} . \tag{5.16}
\]

Since the differential equation does not contain sources or sinks of heat, energy conservation requires that we must further constrain the heat fluxes to be equal, \( F_1 = F_2 \equiv F \). In other words, the heat flowing into the system must equal the heat flowing out of the system. We will refer to such solutions as the \textit{hot and warm planar sandwiches}, depending on whether \( F = 0 \) or \( F \neq 0 \). The final two classes of BCs are mixed Dirichlet and Neumann conditions,

\[
\text{BC3} \\
T(0,t) = T_1 : \quad \alpha_1 \neq 0 \quad \beta_1 = 0 \quad T_1 = \frac{\gamma_1}{\alpha_1} \tag{5.17} \\
\partial_y T(L,t) = F_2 : \quad \alpha_2 = 0 \quad \beta_2 \neq 0 \quad F_2 = \frac{\gamma_2}{\alpha_2} , \tag{5.18}
\]
and

\[
\begin{align*}
BC4 \\
\partial_y T(0,t) = F_1 : \alpha_1 = 0 \quad \beta_1 \neq 0 \quad F_1 = \frac{\gamma_1}{\beta_1} \quad (5.19) \\
T(L,t) = T_2 : \alpha_2 \neq 0 \quad \beta_2 = 0 \quad T_2 = \frac{\gamma_2}{\alpha_2}. 
\end{align*}
\]

The boundary conditions BC3 and BC4 define the half planar sandwich. Note that BC3 and BC4 are physically equivalent, and represent a rod that has been flipped about its midpoint. Boundary conditions BC1–BC4 can be instantiated by

\[
\begin{align*}
\text{solver1} &= \text{Rod1D}(alpha1=1, \text{beta1}=0, \gamma1=T1, \\
& \quad \alpha2=1, \text{beta2}=0, \gamma2=T2, \ TA=Ta, \ TB=Tb) \\
\text{solver2} &= \text{Rod1D}(alpha1=0, \text{beta1}=1, \gamma1=F, \\
& \quad \alpha2=0, \text{beta2}=1, \gamma2=F, \ TA=Ta, \ TB=Tb) \\
\text{solver3} &= \text{Rod1D}(alpha1=1, \text{beta1}=0, \gamma1=T1, \\
& \quad \alpha2=0, \text{beta2}=1, \gamma2=F2, \ TA=Ta, \ TB=Tb) \\
\text{solver4} &= \text{Rod1D}(alpha1=0, \text{beta1}=1, \gamma1=F1, \\
& \quad \alpha2=1, \text{beta2}=0, \gamma2=T2, \ TA=Ta, \ TB=Tb) 
\end{align*}
\]

We have specified the boundary conditions of Rod1D by setting \(\gamma_i\) to the appropriate temperature or flux, and by taking the corresponding coefficients \(\alpha_i\) and \(\beta_i\) to unity or zero, as in \(\text{alpha1}=0, \text{beta1}=0\) and \(\gamma1=T1\) for BC1. The linear initial condition \(T_0(y)\) is specified by \(TA=Ta\) and \(TB=Tb\), which sets the temperature values \(T_A\) and \(T_B\) at the bottom and top of the 1D profile.

As noted above, the planar sandwich is of type BC1. As another example of BC1, we impose the homogeneous BCs

\[
\begin{align*}
T_1 &= 0 \quad (5.21) \\
T_2 &= 0 \quad (5.22)
\end{align*}
\]

and we choose the linear initial condition \([5.8]\) specified by the values \(T_A\) and \(T_B\) at the endpoints. The specific nonhomogenous solution \(\bar{T}(y)\) vanishes, since the boundary conditions are zero, and the solution takes the form \([5]\),

\[
\begin{align*}
T(y,t) &= \sum_{n=1}^{\infty} B_n \sin(k_n y) e^{-\kappa k_n^2 t} \\
k_n &= \frac{n\pi}{L} \text{ with } B_n = \frac{2T_A - 2T_B(-1)^n}{n\pi}. 
\end{align*}
\]

25
We are tempted to call this solution the *homogenous planar sandwich*, and it is illustrated in Fig. 16 for $T_A = 3$ and $T_B = 4$. We have plotted the temperature profiles at times $t = 1, 0.2, 0.1, 0.05, 0.01, 0.001$ with $\kappa = 1$ and $L = 2$. We could have taken $T_A = T_B$, but we chose to plot the case for which the initial profile $T_0(y)$ has a nonzero slope. This case is a bit subtle to implement in a code. For example, if the temperature lives on the midpoint of cells, we must initialize the problem with the values of $T_0(y)$ at these points. For nonuniform initial conditions, one will always encounter the problem of sampling the profile $T_0(y)$ at specific points. While we have derived the solutions for a general linear profile specified by independent values of $T_A$ and $T_B$, the numerical work has been performed only for constant initial conditions, $T_A = T_B = T_0$. We intend to use the solutions with $T_A \neq T_B$ for verification problems involving nonuniform initial setups.

![Temperature profiles for the homogenous planar sandwich at times $t = 1, 0.2, 0.1, 0.05, 0.01, 0.001$. The ExactPack class is PlanarSandwich($T_1=0$, $T_2=0$, $T_A=3$, $T_B=4$, $L=2$, $Nsum=1000$), where we have set $\kappa = 1$, $L = 2$, $T_A = 3$, $T_B = 4$, with $T_1 = T_2 = 0$. The boundary conditions $T_1 = 0$ and $T_2 = 0$ render the solution homogenous, while the linear initial condition $T_0(y)$ is specified by $T_A$ and $T_B$ via $T_0(y) = T_A + (T_B - T_A)y/L$.](image)

**FIG. 16**: Temperature profiles for the *homogeneous planar sandwich* at times $t = 1, 0.2, 0.1, 0.01$, and $0.001$. The ExactPack class is PlanarSandwich($T_1=0$, $T_2=0$, $T_A=3$, $T_B=4$, $L=2$, $Nsum=1000$), where we have set $\kappa = 1$, $L = 2$, $T_A = 3$, $T_B = 4$, with $T_1 = T_2 = 0$. The boundary conditions $T_1 = 0$ and $T_2 = 0$ render the solution homogenous, while the linear initial condition $T_0(y)$ is specified by $T_A$ and $T_B$ via $T_0(y) = T_A + (T_B - T_A)y/L$.

We close this discussion with a few comments on the general BCs for the class Rod1D. As an example let us consider the case

```
solver = Rod1D(alpha1=3, beta1=-1, gamma1=1,
alpha2=1, beta2=2, gamma2=1, L=2, TL=3, TR=3),
```

This corresponds to the BCs

$$3T(0, t) - \partial_y T(0, t) = 1 \quad (5.25)$$

$$T(L, t) + 2 \partial_y T(L, t) = 1,$$

and the solution is plotted in Fig. 17. Unlike the previous case, the IC is uniform, with $T_A = T_B = 3$. The boundary conditions are mixed, and require numerically solving
the equation $\mu \tan \mu = 1$. In the next section, we perform rigorous convergence analyses for the half, hot, and warm planar sandwich variants. In fact, Rod1D is the parent class of PlanarSandwich and all other specialized planar sandwich classes, such as PlanarSandwichHalf and PlanarSandwichHot of the next two sections.

FIG. 17: General boundary conditions: Rod1D(alpha1=3, beta1=-1, gamma1=1, alpha2=1, beta2=2, gamma2=1, TL=3, TR=3).

A. BC2: The Hot and Warm Planar Sandwiches

We now perform convergence analyses for several solutions of the boundary condition class BC2. These solutions are specified by the flux $F$ at the boundary points $y = 0, L$, and the linear IC specified by $T_A$ and $T_B$. As shown in Ref. [5], the corresponding exact solutions are of the form

$$T(y,t) = Fy + A_0 + \sum_{n=1}^{\infty} A_n \cos(k_n y) e^{-\kappa k_n^2 t}$$

$$k_n = \frac{n\pi}{L}, \quad A_0 = \frac{T_A + T_B}{2}, \quad A_n = 2\left(T_A - T_B\right) \frac{1 - (-1)^n}{n^2 \pi^2}, \quad n \geq 1.$$  \hspace{1cm} (5.27) \hspace{1cm} (5.28)

The hot planar sandwich of Ref. [3] is a special case in which the BC is $F = 0$ and the IC is $T(y,0) = T_0$, and on physical grounds, the exact solution is given by the trivial solution

$$T(y,t) = T_0.$$  \hspace{1cm} (5.29)

This solution is constant in time and uniform in space with value $T_0$. This also follow from (5.27) and (5.28) by setting $F_A = F_B = T_0$ and $F = 0$. The hot planar sandwich is therefore
given by $T_A = T_B = T_0$. In the analysis that follows, we take $T_0 = 3$, and consequently, $A_0 = T_0$ and $A_n = 0$ for $n \geq 1$, which reduces to the constant solution $T(y,t) = T_0$. This is illustrated in Fig. 18. This new variant of the planar sandwich can be instantiated by

$$\text{solver} = \text{PlanarSandwichHot}(F=0, \ TA=3, \ TB=3, \ L=2, \ Nsum=1000) .$$

The heat flux $F$ on the boundaries has been set to zero, and a constant initial condition $T_0 = 3$, which has been specified $T_0 = T_A = T_B = 3$ in the ExactPack solution interface. On physical grounds, heat cannot escape from the material, and the temperature must remain constant, $T(y,t) = T_0$, as illustrated by the exact solution plotted in Fig. 18. The numerical results are give in Fig. 19 at time $t = 0.1$, which indeed shows that the temperature remains constant. The lower panel of this Figure gives the convergence analysis. Note that the error is of order machine precision, and the points are scattered somewhat randomly, with a systematic linear increase at the higher precisions.

FIG. 18: The hot planar sandwich in ExactPack: $\text{PlanarSandwichHot}(F=0, \ TA=3, \ TB=3, \ L=2, \ Nsum=1000)$. Since the heat flux on the boundaries vanishes, heat cannot escape from the material, and the temperature must remain constant in time. The temperature profile has been plotted for the times $t = 1, 0.2, 0.1, 0.01, \text{ and } 0.001$, and is indeed constant.
FIG. 19: The hot planar sandwich. The parameter setting are the same as in Fig. 12 except the arithmetic average is used for multimaterial cells.
The next test problem will be called the *warm planar sandwich*. This problem allows heat to escape along the boundaries with flux $F = 1$. The temperature profiles are illustrated in Fig. 20, and the numerical results in Figs. 21–24.

**FIG. 20:** The *warm planar sandwich* in ExactPack: `PlanarSandwichHot(F=1, TA=3, TB=3, L=2, Nsum=1000)`. The profiles are plotted for times $t = 1, 0.2, 0.1, 0.01$, and 0.001. The heat flux at the boundaries is $F = 1$, and we see that the temperature profile changes as heat flows out of the rod. In contrast to Fig. 19, when the heat flux is nonzero, heat is free to flow from the sandwich to the environment, and the temperature need not remain constant.

**FIG. 21:** The parameter setting are the same as in Fig. 12, except the arithmetic average is used for multimaterial cells.

![Graphs showing temperature profiles and error norms for the warm planar sandwich problem.](image-url)
FIG. 22: The parameter setting are the same as in Fig. 12 except the harmonic average is used for multimaterial cells.

FIG. 23: The parameter setting are the same as in Fig. 12 except the thin mesh option is used for multimaterial cells.
FIG. 24: The parameter settings are the same as in Fig. 12 except the static condensation option is used for multimaterial cells.

B. The Half Planar Sandwich

The final variant we shall consider is given by choosing a vanishing heat flux on the upper boundary, \( F_2 = \partial_y T(L) = 0 \), and zero temperature on the lower boundary, \( T_1 = T(0) = 0 \). This is an example of boundary condition BC3, and we call the solution the half planar sandwich. For initial condition \( (5.8) \), Ref. [5] shows that the solution takes the form

\[
T(y, t) = \sum_{n=0}^{\infty} B_n \sin(k_n y) e^{-\kappa k_n^2 t} \tag{5.30}
\]

\[
k_n = \frac{(2n + 1)\pi}{2L} \quad \text{with} \quad B_n = \frac{4T_B}{(2n + 1)\pi} - \frac{8(T_B - T_\lambda)}{(2n + 1)^2\pi^2} \tag{5.31}
\]

Taking the initial condition \( T_0 = 3 \) \( (T_\lambda = T_b = 3) \) gives Fig. 25, which is instantiated by

\[
solver = \text{PlanarSandwichHalf(T1=0, F2=0, TA=3, TB=3, L=2, Nsum=1000)}
\]

If we had chosen \( \partial_y T(0) = 0 \) and \( T(L) = 0 \), as in BC4, then the plot would have been reflected about the midpoint of the rod, but is otherwise physically identical, as illustrated in Fig. 26. Figures 27 and 28 provide the solution plots and the convergence analyses for the
FIG. 25: The half planar sandwich in ExactPack: PlanarSandwichHalf(T1=0, F2=0, TA=3, TB=3, L=2, Nsum=1000). The profiles are plotted for times $t = 1, 0.2, 0.1, 0.01,$ and $0.001$. Note that the temperature profile vanishes on the bottom and the derivative of the temperature vanishes on the top.

arithmetic and harmonic averages of the half planar sandwich. As in the previous section, the top panel for the harmonic average in Fig. 28 exhibits near-zeros along the horizontal axis. Similarly, Figs. 29 and 30 illustrate the convergence analyses for the thin mesh and static condensation options for the half planar sandwich. The convergence plots for the half planar sandwich are qualitatively similar to those for the planar sandwich. In both cases, the arithmetic and harmonic averages are approximately 1\textsuperscript{st} order, while thin mesh and static condensation start out at 2\textsuperscript{nd} order and approach 1\textsuperscript{st} order as the mesh is refined.
FIG. 26: Note that BC3 and BC4 are physically equivalent, and are related by a parity inversion across the midpoint. The Figure uses $\kappa = 1$, $L = 2$, $T_A = T_B = 3$. By symmetry principles, the two profiles are mirror images of one another. BC3 can be instantiated by Rod1D($\alpha_1=1$, $\beta_1=0$, $\gamma_1=1$, $\alpha_2=0$, $\beta_2=1$, $\gamma_2=0$, $TL=0$, $TR=0$), and BC4 by Rod1D($\alpha_1=0$, $\beta_1=1$, $\gamma_1=0$, $\alpha_2=0$, $\beta_2=1$, $\gamma_2=F1$, $TL=0$, $TR=0$). Note that $T_A$ and $T_B$ are interchanged between BC3 and BC4. The class Rod1D is the parent class of PlanarSandwich.

FIG. 27: The parameter setting are the same as in Fig. 12.
FIG. 28: The parameter setting are the same as in Fig. 12 except the harmonic average is used for multimaterial cells.

FIG. 29: The parameter setting are the same as in Fig. 12 except the thin mesh option is used for multimaterial cells.
VI. CONCLUSIONS AND FUTURE RESEARCH

Two dimensional (2D) multimaterial heat diffusion can be a challenging numerical problem when the material boundaries are misaligned with the numerical grid. Even when the boundaries start out aligned, they typically become misaligned through hydrodynamic motion; therefore, it is important to perform rigorous verification analyses of the heat transport algorithms in any heat-conduction hydrocode. In this paper we perform convergence analyses for the four multimaterial heat flow algorithms in the multi-physics hydrodynamics code FLAG: (i) the arithmetic average, (ii) the harmonic average, (iii) thin mesh, and (iv) static condensation. To perform the analyses and to produce the corresponding convergence plots, we employ the code verification tool ExactPack. We concentrate on the 2D planar sandwich test problem, along with three generalizations called the half planar sandwich, the hot planar sandwich, and the warm planar sandwich, all of which possess simple exact solutions. These test problems were designed to exhibit multimaterial cells along a fixed boundary, thereby exercising the multimaterial algorithms by the simplest means possible.

The geometry of the planar sandwich is illustrated in Fig. 1 which shows a square numerical grid overlaid on a rectangular physical geometry consisting of three parallel sandwich-like regions. The numerical grid partitions the physical geometry into a number of corresponding numerical cells, which need not align with the material regions. The outer two regions,
called the *bread* of the sandwich, are composed of an insulating material with zero heat diffusion constant $\kappa_1 = 0$, while the inner region, called the *meat* of the sandwich, has diffusion coefficient $\kappa_2 = \kappa > 0$.

One of our primary results is that the arithmetic and harmonic averages converge at 1st order for all three variants of the planar sandwich. We also find that both the thin mesh and static condensation algorithms start out converging at 2nd order, but as the mesh is refined, the convergence rate levels off to 1st order. We conjecture that this is because the error in interface reconstruction algorithms becomes less precise at finer resolutions.

With more work, one can also construct an exact solution for the case in which $\kappa_1 \neq 0$ in the bread of the sandwich [10], and these solutions might be an interesting avenue for future verification work. We are currently adding these solutions to ExactPack. We are also exploring Voronoi mesh simulations for the planar sandwich. Unlike the square mesh, Voronoi verification must be performed in 2D. This is because the Voronoi mesh does not align uniformly along $y =$constant, and one cannot use the 1D profiles for the exact solutions. Voronoi mesh verification is further complicated by the fact that the cells are not necessarily of equal area. Ref. [13] explores the various choices of norm in such cases. We are adding a VTK reader to ExactPack, and this will greatly facilitate verification work with nonuniform meshes.

We also plan to study problems with more complicated geometry, such as the *cylindrical sandwich*. This test problem was proposed by Alan Dawes in Ref [4], and has been analyzed by Dawes, Shashkov, and Malone [3]. These authors did not perform convergence analyses, and they used a highly resolved “reference solution” rather than an exact solution, although the exact solution was presented in Appendix B of Ref. [3]. This test problem is much like the planar sandwich, except that the heat-conducting material is annular and lies in the first quadrant. The left most region is set to $T = 1$ and the bottom region to $T = 0$, and therefore heat moves clockwise along the annulus. See the left panel of Fig. [31]. As illustrated by the right panel of Fig. [31] the square mesh will never align with the material boundary. Generalizations of the cylindrical sandwich exist much like those of Ref. [5], and we plan to investigate these solutions as well.
FIG. 31: The cylindrical sandwich test problem. On the square grid, the multimaterial cells never align with the material boundary.
Appendix A: Python script for the Planar Sandwich

```python
import numpy as np
import matplotlib.pylab as plt

from exactpack.solvers.heat import PlanarSandwich

L = 2.0
x = np.linspace(0.0, L, 1000)
t1 = 1.0
t2 = 0.2
t3 = 0.1
t4 = 0.05
t5 = 0.01
t6 = 0.001

solver = PlanarSandwich(TB=1, TT=0, L=L, Nsum=1000)
soln1 = solver(x, t1)
soln2 = solver(x, t2)
soln3 = solver(x, t3)
soln4 = solver(x, t4)
soln5 = solver(x, t5)
soln6 = solver(x, t6)
soln1.plot('temperature', label=r'$t=1.000$')
soln2.plot('temperature', label=r'$t=0.200$')
soln3.plot('temperature', label=r'$t=0.100$')
soln4.plot('temperature', label=r'$t=0.050$')
soln5.plot('temperature', label=r'$t=0.010$')
soln6.plot('temperature', label=r'$t=0.001$')

plt.title('PlanarSandwich')
plt.ylabel(r'temperature', fontsize='18')
plt.ylim(0, 1)
plt.xlim(0, L)
plt.legend(loc=0)
plt.grid(True)
plt.savefig('planar_sandwich.png')
plt.show()
```
Appendix B: ExactPack Script for Convergence Analysis

```python
import numpy as np
import os.path
import glob
import matplotlib.pyplot as plt
from exactpack.solvers.heat import PlanarSandwich
from exactpack.analysis import *
from lanl_readers.flag import FlagVarDump

# problem parameters
L = 2.0
t = 0.1

# solver name
solver_name = 'PlanarSandwich'

# multimaterial algorithm
multimat = 'c1'  # arithmetic average
multimat_name = 'Arithmetic Average'

# study resolutions
res_study = [5, 10, 20, 40, 80, 160, 320, 640]  # 8

# output tag
problem_out = 'planar_sandwich_{}_run16'.format(multimat)

# variable selection
variables = 'temperature'

# plot parameters
plot_params = {'temperature': {'ymin': 0, 'ymax': 1, 'xmin': L, 'xmax': L, 'error_min': 1.e-4, 'error_max': 1.e-0, 'num': [8], 'loc': [1, 2, 1, 2]}}

# run dir and dump files
dumpfiles = '/Users/bobs/mygit_repos/heat/runs/planar_sandwich/{}/res*/vardump/planar_sandwich_{}_VarDump.00000.1.000000E-01.zx'.format(multimat, multimat)

# create study_parameters
study_parameters = [L/float(res) for res in res_study]

# create solver
solver = PlanarSandwich(TB=1, TT=0, L=L, Nsum=1000)

# study object
print "*** plot solution and code data ..."
```

40
study = Study(sorted(glob.glob(dumpfiles)),
    reference=solver,
    study_parameters=study_parameters,
    time=t,
    reader=FlagVarDump(),
    abscissa='y_position'
)

# plot solution profiles
plt.clf()
study.plot('temperature')
plt.xlim(xmin, xmax)
plt.ylim(ymin, ymax)
plt.ylabel('$T$')
plt.xlabel('$y$')
plt.title(r'{} : {}: t={}'.format(solver_name, multimat_name, t))
plt.legend(loc=loc)
plt.grid(True)
plt.savefig(problem_out+'_'+'temperature_soln.png')
plt.show()

# plot on same axis

error_max = plot_params[variable][‘error_max’]
error_min = plot_params[variable][‘error_min’]
loc = plot_params[variable][‘loc’][1]
n0 = plot_params[variable][‘num’][0]
domain = (0, L)
 fiducials = {'temperature': 1}
fit = RegressionConvergenceRate(study, norm=PointNorm(), domain=domain, fiducials=fiducials)
fit[:n0].plot_fit('temperature', '-', c='r')
fit.norms.plot('temperature', label=None, markersize=10)
fit.plot_fiducial('temperature')
plt.ylim(error_min, error_max)
plt.title(r'{} : {}: t={}'.format(solver_name, multimat_name, t))
plt.xlabel(r'$h$')
plt.ylabel(r'$L_1$ error norm')
plt.legend(loc=loc)
plt.savefig(problem_out+'_'+'temperature_conv.png')
plt.show()
xmin = plot_params[variable][‘xmin’]
xmax = plot_params[variable][‘xmax’]
ymin = plot_params[variable][‘ymin’]
ymax = plot_params[variable][‘ymax’]
error_max = plot_params[variable][‘error_max’]
error_min = plot_params[variable][‘error_min’]
n0 = plot_params[variable][‘num’][0]
loc1 = plot_params[variable][‘loc’][2]
loc2 = plot_params[variable][‘loc’][3]

plt.figure(figsize=(11, 11))
ax = plt.subplot(211)
study.plot(‘temperature’)
plt.xlim(xmin, xmax)
plt.ylim(ymin, ymax)
plt.ylabel(‘$T$’)
plt.xlabel(‘$y$’)
plt.title(r’{} : {}: t={}'.format(solver_name, multimat_name, t))
plt.legend(loc=loc1)
plt.grid(True)

ax = plt.subplot(212)
domain = (0, L)
fiducials = {‘temperature’: 1}
fit = RegressionConvergenceRate(study, norm=PointNorm(),
    domain=domain, fiducials=fiducials)
fit[:n0].plot_fit(‘temperature’, “-”, c=’r’)
fit.norms.plot(‘temperature’, label=None, markersize=10)
fit.plot_fiducial(‘temperature’)
plt.ylim(error_min, error_max)
plt.xlabel(r’$h$’)
plt.ylabel(r’$L_1$ error norm’)
plt.legend(loc=loc2)
plt.savefig(problem_out+’ ‘+temperature_soln_conv.png)
plt.show()
Appendix C: Solution to the Planar Sandwich

In this section we find the general solution to the heat equation (5.1)–(5.4). The differential equation (DE), the boundary conditions (BCs), and the initial condition (IC) are reproduced here for convenience,

**DE:** \[ \frac{\partial T(y,t)}{\partial t} = \kappa \frac{\partial^2 T(y,t)}{\partial y^2} \] \[ 0 < y < L \text{ and } t > 0 \] (C1)

**BCs:** \[ \alpha_1 T(0,t) + \beta_1 \frac{\partial y}{\partial y} T(0,t) = \gamma_1 \] \[ t > 0, \ y = 0, L \] (C2)

\[ \alpha_2 T(L,t) + \beta_2 \frac{\partial y}{\partial y} T(L,t) = \gamma_2 \] (C3)

**IC:** \[ T(y,0) = T_0(y) \] \[ 0 < y < L, \ t = 0 \] (C4)

The solution is obtained by solving two independent problems: (i) finding a specific *static nonhomogeneous* solution \( \bar{T}(y) \) and (ii) finding the general *homogeneous* solution \( \tilde{T}(y,t) \) satisfying the initial condition

\[ \tilde{T}(y,0) = T_0(y) - \bar{T}(y) . \] (C5)

The homogeneous solution \( \tilde{T}(y, t) \) can be represented as a Fourier series. Note that \( \bar{T} \) depends upon the BCs, while \( \tilde{T} \) depends upon the IC and the BCs. Once \( \bar{T} \) and \( \tilde{T} \) have been found, the general solution is given by

\[ T(y,t) = \bar{T}(y) + \tilde{T}(y,t) . \] (C6)

Note that \( T(y,t) \) satisfies the initial condition \( T(y,0) = T_0(y) \), and is the unique solution because of the maximum principle.

1. **The Static Nonhomogeneous Problem**

Because of its simplicity, we first turn to finding the static nonhomogeneous solution \( \bar{T}(y) \). In the static limit, the differential equation (C1) and boundary conditions (C3) reduce to

\[ \frac{\partial^2 \bar{T}(y)}{\partial x^2} = 0 \] (C7)

\[ \alpha_1 \bar{T}(0) + \beta_1 \bar{T}'(0) = \gamma_1 \] (C8)

\[ \alpha_2 \bar{T}(L) + \beta_2 \bar{T}'(L) = \gamma_2 . \] (C9)

The initial condition (C4) of the full time dependent problem can be ignored since we are only interested in static solutions. The general solution to (C7) is trivial, and takes the form

\[ \bar{T}(y) = a + by . \] (C10)
For Dirichlet boundary conditions, we must specify the temperature values \( T_1 \) and \( T_2 \) at the endpoints, thereby giving the nonhomogenous static solution

\[
T(y) = T_1 + \frac{T_2 - T_1}{L} y .
\] (C11)

The coefficients \( a \) and \( b \), or equivalently \( T_1 \) and \( T_2 \), are determined by \( \bar{T}(0) = a = T_1 \) and \( \bar{T}(L) = a + bL = T_2 \). For a Neumann BC specified by flux \( F_i \), we can always rewrite the corresponding homogeneous solution in the form (C11) by the defining temperature \( T_i = F_i L \), where we can take \( i = 1, 2 \). The solution to

The BCs (C8) and (C9) can be written as a linear equation in terms of \( a \) and \( b \),

\[
\begin{pmatrix}
\alpha_1 & \beta_1 \\
\alpha_2 & \beta_2 + \alpha_2 L \\
\end{pmatrix}
\begin{pmatrix}
a \\
b \\
\end{pmatrix}
= \begin{pmatrix}
\gamma_1 \\
\gamma_2 \\
\end{pmatrix} .
\] (C12)

Upon solving the system of equations we find

\[
a = \frac{\beta_2 \gamma_1 - \beta_1 \gamma_2 + L \alpha_2 \gamma_1}{\alpha_1 \beta_2 - \alpha_2 \beta_1 + L \alpha_1 \alpha_2} \] (C13)

\[
b = \frac{\alpha_1 \gamma_2 - \alpha_2 \gamma_1}{\alpha_1 \beta_2 - \alpha_2 \beta_1 + L \alpha_1 \alpha_2} ,
\] (C14)

or in terms of temperature parameters \( T_1 = a \) and \( T_2 = a + bL \),

\[
T_1 = \frac{\beta_2 \gamma_1 - \beta_1 \gamma_2 + L \alpha_2 \gamma_1}{\alpha_1 \beta_2 - \alpha_2 \beta_1 + L \alpha_1 \alpha_2} \] (C15)

\[
T_2 = \frac{\beta_2 \gamma_1 - \beta_1 \gamma_2 + L \alpha_1 \gamma_2}{\alpha_1 \beta_2 - \alpha_2 \beta_1 + L \alpha_1 \alpha_2} .
\] (C16)

Note that the determinant of the linear equations vanishes for BC2, and we must handle this case separately. We can also express the BCs in terms of the fluxes

\[
F_1 = T_1 / L 
\] (C17)

\[
F_2 = T_2 / L 
\] (C18)

by writing

\[
\bar{T}(y) = T_1 + (F_1 - F_2)y .
\] (C19)

We can also express \( \bar{T}(y) \) by combinations of temperature and flux.
Special Cases of the Static Problem

a. BC1

Let us consider the simple Dirichlet boundary conditions (5.13) and (5.14),
\[
\bar{T}(0) = T_1 \quad \text{(C20)}
\]
\[
\bar{T}(L) = T_2 \quad \text{(C21)}
\]
which gives the solution
\[
\bar{T}(y) = T_1 + \frac{T_2 - T_1}{L} y \quad \text{(C22)}
\]
The temperature coefficients are given by
\[
T_1 = \frac{\gamma_1}{\alpha_1} \quad \text{(C23)}
\]
\[
T_2 = \frac{\gamma_2}{\alpha_2} \quad \text{(C24)}
\]
which follows from Eqs. (C8) and (C9), or equivalently from Eqs. (C15) and (C16) with \( \beta_1 = \beta_2 = 0 \). Similarly, the coefficients in (C10) are \( a = T_1 \) and \( b = (T_2 - T_1)/L \).

b. BC2

Let us now find the nonhomogeneous equilibrium solution for the Neumann boundary conditions (5.15) and (5.16),
\[
\partial_y \bar{T}(0) = F_1 \quad \text{(C25)}
\]
\[
\partial_y \bar{T}(L) = F_2 \quad \text{(C26)}
\]
where \( F_1 \) and \( F_2 \) are the heat fluxes at \( y = 0 \) and \( y = L \), respectively. The fluxes and are related to the boundary condition parameters in (C8) and (C9) by \( F_1 = \gamma_1/\beta_1 \) and \( F_2 = \gamma_2/\beta_2 \) with \( \alpha_1 = \alpha_2 = 0 \). As before, the general solution is \( \bar{T}(y) = a + by \), and we see that \( \bar{T}'(y) = b \) is independent of \( y \). In other words, the heat flux at either end of the rod must be identical, \( F_1 = b = F_2 \). In fact, this result follows from energy conservation, since, in static equilibrium, the heat flowing into the rod must be equal the heat flowing out of the rod. More correctly, we should therefore start with the boundary conditions
\[
\partial_y \bar{T}(0) = F \quad \text{(C27)}
\]
\[
\partial_y \bar{T}(L) = F \quad \text{(C28)}
\]
where
\[
F = \frac{\gamma_1}{\beta_1} = \frac{\gamma_2}{\beta_2} \quad \text{(C29)}
\]
The value of the constant term $a$ is not uniquely determined in this case; however, we are free to set it to zero, or to combine it with the constant $A_0$ term of $\tilde{T}(y,t)$, thereby giving

$$\tilde{T}(y) = F y .$$  \hfill (C30)

There is nothing wrong with setting $a = 0$, since we only need to find one nonhomogeneous solution, and (C30) fits the bill. We can write this solution in the form (C11), with

$$T_1 = 0 \quad \text{(C31)}$$
$$T_2 = FL . \quad \text{(C32)}$$

c. $BC3$

We now consider the mixed Dirichlet and Neumann boundary conditions (5.17) and (5.18),

$$\tilde{T}(0) = T_1 \quad \text{(C33)}$$
$$\partial_y \tilde{T}(L) = F_2 . \quad \text{(C34)}$$

We can express the solution (C11) in terms of the temperature $T_1$, and the effective temperature

$$T_2 = T_1 + F_2 L = \frac{\gamma_1}{\alpha_1} + \frac{\gamma_2 L}{\beta_2} , \quad \text{(C35)}$$

and the solution takes the form

$$\tilde{T}(y) = T_1 + F_2 y . \quad \text{(C36)}$$

d. $BC4$

The boundary conditions are (5.19) and (5.20),

$$\partial_y \tilde{T}(0) = F_1 \quad \text{(C37)}$$
$$\tilde{T}(L) = T_2 , \quad \text{(C38)}$$

and the solution (C11) can be written The solution takes the form

$$\tilde{T}(y) = (T_2 - F_1 L) + F_1 y . \quad \text{(C39)}$$

We can define an effective temperature

$$T_1 = T_2 - F_1 L = \frac{\gamma_2}{\alpha_2} - \frac{\gamma_1 L}{\beta_1} , \quad \text{(C40)}$$
2. The General Homogeneous Problem

Now that we have constructed the static nonhomogenous solution $\overline{T}(y)$ appropriate to the choice of boundary conditions, we turn to the slightly more involved task of finding the general homogeneous solution $\tilde{T}(y,t)$. This is equivalent to solving a discrete eigenvalue problem, albeit in an infinite number of dimensions. We then construct the solution $\tilde{T}(y,t)$ as a weighted sum over the normal modes, where the weights are determined by the choice of BCs and the IC. The special cases BC1, BC2, BC3, and BC4 are particularly simple. The homogeneous equations of motion, for which $\gamma_1 = 0$ and $\gamma_2 = 0$, take the form

$$\text{DE} : \quad \frac{\partial \tilde{T}(y,t)}{\partial t} = \kappa \frac{\partial^2 \tilde{T}(y,t)}{\partial y^2}, \quad 0 < y < L \text{ and } t > 0 \quad (C41)$$

$$\text{BC} : \quad \alpha_1 \tilde{T}(0,t) + \beta_1 \partial_y \tilde{T}(0,t) = 0, \quad t > 0 \quad (C42)$$

$$\alpha_2 \tilde{T}(L,t) + \beta_2 \partial_y \tilde{T}(L,t) = 0$$

$$\text{IC} : \quad \tilde{T}(y,0) = T_0(y), \quad 0 < y < L. \quad (C43)$$

As we have discussed, we shall focus on the linear initial condition

$$T_0(y) = T_A + \frac{T_B - T_A}{L} y, \quad 0 < y < L, \quad (C44)$$

although, more generally, any continuous function $T_0(y)$ will produce a solution.

The solution technique is by separation of variables, for which we assume the solution to be a product of independent functions of $y$ and $t$,

$$\tilde{T}(y,t) = Y(y) U(t). \quad (C45)$$

Substituting this Ansatz into the heat equation gives

$$\frac{dU(t)}{dt} Y(y) = \kappa U(t) \frac{d^2Y(y)}{dy^2}, \quad (C46)$$

or

$$\frac{1}{\kappa} \frac{U'(t)}{U(t)} = \frac{Y''(y)}{Y(y)} = \text{const} \equiv -k^2, \quad (C47)$$

where we have chosen the constant negative value $-k^2$, and we have expressed the derivatives of $U(t)$ and $Y(y)$ by primes. As usual in the separation of variables technique, when two functions of different variables are equated, they must be equal to a constant, independent of $y$ and $t$. The variable $U(t)$ satisfies

$$U'(t) = -\kappa k^2 U(t), \quad (C48)$$
which has the solution

\[ U_k(t) = U_0 e^{-\kappa k^2 t} . \] (C49)

We have introduced a \( k \)-subscript in \( U_k \) to indicate that the solution depends upon the value of \( k \). Without loss of generality we set \( U_0 = 1 \). We now find that the equation for \( Y \) reduces to

\[ Y''(y) + k^2 Y'(y) = 0 \] (C50)

\[ \alpha_1 Y(0) + \beta_1 Y'(0) = 0 \] (C51)

\[ \alpha_2 Y(L) + \beta_2 Y'(L) = 0 . \]

The general solution to (C50) is

\[ Y_k(y) = A_k \cos ky + B_k \sin ky . \] (C52)

When the BCs are applied, the modes \( Y_k \) will be orthogonal,

\[ \int_0^L dx Y_k(y) Y_{k'}(y) = N_k \delta_{kk'} . \] (C53)

It is instructive to prove the orthogonality relation (C53) directly from the differential equation. Given two solutions \( Y_k \) and \( Y_{k'} \) to (C50), we can write the two alternative forms,

\[ Y_{k'} \left[ Y''_k + k^2 Y_k \right] = 0 \] (C54)

\[ Y_k \left[ Y''_{k'} + k'^2 Y_{k'} \right] = 0 . \] (C55)

These forms differ only in the interchange of \( k \) and \( k' \). Upon subtracting these equations, and then integrating over space, we find

\[ (k^2 - k'^2) \int_0^L dy Y_k Y_{k'} = \int_0^L dy \left[ Y_k Y''_{k'} - Y_{k'} Y''_k \right] \]

\[ = \int_0^L dy \left[ \frac{d}{dy} \left( Y_k Y'_{k'} \right) - Y_k' Y_{k'}' - \frac{d}{dy} \left( Y_{k'} Y'_k \right) + Y'_{k'} Y'_k \right] \]

\[ = \int_0^L dy \frac{d}{dy} \left( Y_k Y'_{k'} - Y_{k'} Y'_k \right) \]

\[ = \left( Y_k Y'_{k'} - Y_{k'} Y'_k \right) \bigg|_0^L = 0 , \] (C57)

where each contribution from \( y = 0 \) and \( y = L \) vanishes separately because of their respective boundary conditions. We therefore arrive at

\[ (k^2 - k'^2) \int_0^L dy Y_k Y_{k'} = 0 . \] (C59)
Provided \( k \neq k' \), we can divide (C59) by \( k^2 - k'^2 \) to obtain
\[
\int_0^L dy Y_k(y) Y_{k'}(y) = 0 \quad \text{when} \quad k \neq k'. \tag{C60}
\]

When \( k = k' \), (C59) gives no constraint on the normalization integral. However, since the differential equation is linear, and since the BCs are homogeneous and linear, we are free to normalize \( Y_k \) over \([0, L]\) such that \( \int dy Y_k^2 = N_k \) for any convenient choice of \( N_k \).

We now express the general time dependent solution as a sum over all modes,
\[
\tilde{T}(y, t) = \sum_k D_k Y_k(y) e^{-\kappa k^2 t}. \tag{C61}
\]
The coefficients \( D_k \) themselves are chosen so that the initial condition is satisfied,
\[
\tilde{T}(y, 0) = \sum_k D_k Y_k(y) = T_0(y) - \bar{T}(y) \tag{C62}
\]
\[
\Rightarrow \quad D_k = \frac{1}{N_k} \int_0^L dy \left[ T_0(y) - \bar{T}(y) \right] Y_k(y). \tag{C63}
\]
Substituting (C44) and (C11) into (C62) gives
\[
\tilde{T}(y, 0) = T_0(y) - \bar{T}(y) \tag{C64}
\]
\[
= T_a + \frac{T_b - T_a}{L} y, \tag{C65}
\]
with
\[
T_a = T_\alpha - T_1 \tag{C66}
\]
\[
T_b = T_\beta - T_2. \tag{C67}
\]
Therefore, the coefficients \( D_k = D_k(T_a, T_b) \) are functions of \( T_a \) and \( T_b \).

**Special Cases of the Homogeneous Problem**

**a. BC1**

In the first case we hold the temperature fixed to zero at both ends of the rod,
\[
\tilde{T}(0, t) = 0 \tag{C68}
\]
\[
\tilde{T}(L, t) = 0. \tag{C69}
\]
The general solution \( Y_k(y) = A_k \cos ky + B_k \sin ky \) reduces to \( Y_k(y) = B_k \sin ky \) under (C68), while (C69) restricts the wave numbers to satisfy \( \sin kL = 0 \), \textit{i.e.} \( k = k_n = n\pi/L \) for \( n = 1, 2, 3, \ldots \). Note that \( n = 0 \) does not contribute, since this gives the trivial vanishing
solution. It is convenient to label the modes by the mode number \( n \) rather than the wave number \( k_n \), so that the homogeneous solution takes the form

\[
\tilde{T}(y, t) = \sum_{n=1}^{\infty} B_n Y_n(y) e^{-\kappa k_n^2 t} \tag{C70}
\]

\[
Y_n(y) = \sin k_n y \tag{C71}
\]

\[
k_n = \frac{n\pi}{L} \quad n = 1, 2, 3, \cdots . \tag{C72}
\]

The tilde over the temperature is meant to explicitly remind us that this is the general homogeneous solution. The orthogonality condition on the modes \( Y_n \) can be checked by a simple integration,

\[
\int_0^L dy Y_n(y) Y_m(y) = \frac{L}{2} \delta_{nm} \tag{C73}
\]

For an initial condition \( \tilde{T}(y, 0) = T_0(y) \), we can calculate the corresponding coefficients in the Fourier sum,

\[
B_n = \frac{2}{L} \int_0^L dy T_0(y) \sin k_n y . \tag{C74}
\]

For the linear initial condition \((C44)\), a simple calculation gives

\[
B_n = 2T_\lambda \frac{1 - (-1)^n}{n\pi} + 2(T_\lambda - T_u) \frac{(-1)^n}{n\pi} \tag{C75}
\]

\[
= \frac{2T_\lambda - 2T_u(-1)^n}{n\pi} . \tag{C76}
\]

b. BC2

The second special boundary condition that we consider sets the heat flux at both ends of the rod to zero,

\[
\partial_y \tilde{T}(0, t) = 0 \tag{C77}
\]

\[
\partial_y \tilde{T}(L, t) = 0 . \tag{C78}
\]

This is the hot planar sandwich. The general solution \( Y_k(y) = A_k \cos k y + B_k \sin k y \) reduces to \( Y_k(y) = A_k \cos k y \) under \((C77)\), while \((C78)\) restricts the wave numbers to \( k \sin k L = 0 \), so that \( k = k_n = n\pi/L \) for \( n = 0, 1, 2 \cdots . \) In this case, the \( n = 0 \) mode is permitted (and indeed essential), and the solution can be written

\[
\tilde{T}(y, t) = \frac{A_0}{2} + \sum_{n=1}^{\infty} A_n Y_n(y) e^{-\kappa k_n^2 t} \tag{C79}
\]

\[
Y_n(y) = \cos k_n y \tag{C80}
\]

\[
k_n = \frac{n\pi}{L} \quad n = 0, 1, 2, \cdots . \tag{C81}
\]
where a conventional factor of $1/2$ has been inserted in the $A_0$ term. This is because of the difference in normalization between $n = 0$ and $n \neq 0$,

$$
\int_0^L dy Y_0^2(y) = L \quad \text{(C82)}
$$

$$
\int_0^L dy Y_n^2(y) = \frac{L}{2} \quad n \neq 0 \quad \text{(C83)}
$$

Given the initial condition $\tilde{T}(y, 0) = T_0(y)$, the Fourier modes become

$$
A_n = \frac{2}{L} \int_0^L dy T_0(y) \cos k_n y \quad \text{(C84)}
$$

for $n = 0, 1, 2 \cdots$. This holds for all values of $n$, including $n = 0$. This is why we used the factor of $1/2$ in the $A_0$ term of (C79). For simplicity, we will take the linear initial condition (C44) for $T_0(y)$, in which case, (C84) gives the coefficients

$$
\frac{A_0}{2} = \frac{1}{2} \left( T_A + T_B \right) \quad \text{(C85)}
$$

$$
A_n = 2 \left( T_A - T_B \right) \frac{1 - (-1)^n}{n^2 \pi^2} \quad \text{(C86)}
$$

For pedagogical purposes, let us work through the algebra for the $A_n$ coefficients, doing the $n = 0$ case first:

$$
\frac{A_0}{2} = \frac{1}{L} \int_0^L T_0(y) = \frac{1}{L} \int_0^L \left[ T_A + \frac{T_B - T_A}{L} y \right] \quad \text{(C87)}
$$

$$
= T_L + \left[ \frac{T_B - T_A}{2} \right] = \frac{1}{2} [T_B + T_A] \quad \text{(C88)}
$$

Next, taking $n \neq 0$, we find:

$$
A_n = \frac{2}{L} \int_0^L dy T_0(y) \cos k_n y \quad \text{(C89)}
$$

$$
= \frac{2}{L} \int_0^L dy \left[ T_A + \frac{T_B - T_A}{L} y \right] \cos k_n y \quad \text{(C90)}
$$

$$
= T_A \frac{2}{L} \int_0^L dy \cos k_n y + \left( T_B - T_A \right) \frac{2}{L^2} \int_0^L dy y \cos k_n y \quad \text{(C91)}
$$

The first term integrates to zero since

$$
\frac{2}{L} \int_0^L dy \cos k_n y = \frac{2}{L} \sin k_n y \bigg|_{y=0}^{y=L} = 0 \quad \text{(C92)}
$$
and the second term gives
\[
\frac{2}{L^2} \int_0^L dy \cos k_n y = \frac{2}{L^2} \left[ \frac{\cos k_n y}{k_n^2} + \frac{y \sin k_n y}{k_n} \right]_{y=0}^{y=L} = \frac{2}{L^2} \frac{L^2}{n^2 \pi^2} \left[ \cos k_n L - 1 \right] = 2 \frac{(-1)^n - 1}{n^2 \pi^2}, \tag{C94}
\]
which leads to (C86).

c. BC3

The next specialized boundary condition is
\[
\tilde{T}(0, t) = 0 \quad \tag{C95}
\]
\[
\partial_y \tilde{T}(L, t) = 0. \tag{C96}
\]
This is the half planar sandwich. The general solution \( Y_k(y) = A_k \cos ky + B_k \sin ky \) under (C95) reduces to \( Y_k(y) = B_k \sin ky \), while (C96) restricts the wave numbers to \( k \cos kL = 0 \), so that \( k = k_n = \frac{(2n + 1)\pi}{2L} \) for \( n = 0, 1, 2 \cdots \). The general homogeneous solution is therefore
\[
\tilde{T}(y, t) = \sum_{n=0}^{\infty} B_n Y_n(y) e^{-\kappa k_n^2 t} \tag{C97}
\]
\[
Y_n(y) = \sin k_n y \tag{C98}
\]
\[
k_n = \frac{(2n + 1)\pi}{2L} \quad n = 0, 1, 2, \cdots \tag{C99}
\]
The initial condition \( \tilde{T}(y, 0) = T_0(y) \) gives the Fourier modes
\[
B_n = \frac{2}{L} \int_0^L dy T_0(y) \sin k_n y, \tag{C100}
\]
and, upon taking the linear initial condition (5.8), we find
\[
B_n = \frac{4T_A}{(2n + 1)\pi} + 4(T_B - T_A) \left[ \frac{1}{(2n + 1)\pi} - \frac{2}{(2n + 1)^2 \pi^2} \right] \tag{C101}
\]
\[
= \frac{4T_B}{(2n + 1)\pi} - \frac{8(T_B - T_A)}{(2n + 1)^2 \pi^2}. \tag{C102}
\]
d. BC4

The last special case is the boundary condition

\[ \frac{\partial}{\partial y} \tilde{T}(0, t) = 0 \quad \text{(C103)} \]
\[ \tilde{T}(L, t) = 0 . \quad \text{(C104)} \]

The general solution \( Y_k(y) = A_k \cos ky + B_k \sin ky \) reduces to \( Y_k(y) = A_k \cos ky \) under (C103), while (C104) restricts the wave numbers to \( \cos kL = 0 \), i.e. \( k = k_n = (2n + 1)\pi/2L \) for \( n = 0, 1, 2 \ldots \), which gives rise to the homogeneous solution

\[ \tilde{T}(y, t) = \sum_{n=0}^{\infty} A_n Y_n(y) e^{-\kappa k_n^2 t} \quad \text{(C105)} \]
\[ Y_n(y) = \cos k_n y \quad \text{(C106)} \]
\[ k_n = \frac{(2n + 1)\pi}{2L} \quad n = 0, 1, 2, \ldots . \quad \text{(C107)} \]

Similar to (C100), the mode coefficient is

\[ A_n = \frac{2}{L} \int_0^L dy T_0(y) \cos k_n y , \quad \text{(C108)} \]

and, upon taking the linear initial condition (C44), we find

\[ A_n = 4T_A \frac{(-1)^n}{(2n + 1)\pi} - 8 \left( T_B - T_A \right) \frac{1 - (-1)^n}{(2n + 1)^2 \pi^2} . \quad \text{(C109)} \]

Note that BC3 and BC4 are in fact equivalent, and represent a rod that has been flipped from left to right about its center, as illustrated in Fig. 26.

General Boundary Conditions

We now turn to the general form of the boundary conditions, which, expressed in terms of \( Y \), take the form

\[ \alpha_1 Y_k(0) + \beta_1 Y'_k(0) = 0 \quad \text{(C110)} \]
\[ \alpha_2 Y_k(L) + \beta_2 Y'_k(L) = 0 . \quad \text{(C111)} \]

The solution and its derivative are

\[ Y_k(y) = A \cos ky + B \sin ky \quad \text{(C112)} \]
\[ Y'_k(y) = -Ak \sin ky + Bk \cos ky . \quad \text{(C113)} \]
FIG. 32: The roots $\mu_n$ for $\alpha_1 = 1$, $\bar{\beta} = 1/2$, $\alpha_2 = 1$, and $\bar{\beta}_2 = 1$. For $L = 2$ this gives $\beta_1 = 1$ and $\beta_2 = 2$.

Substituting this into (C110) and (C111) gives

$$\alpha_1 A + \beta_1 Bk = 0 \quad \text{(C114)}$$

$$\alpha_2 \left[ A \cos kL + B \sin kL \right] + \beta_2 \left[ -Ak \sin kL + Bk \cos kL \right] = 0 \quad \text{(C115)}$$

There are two cases, $\cos kL \neq 0$ and $\cos kL = 0$. We have already addressed the latter, so let us now consider the former. Upon dividing (C115) by $\cos kL$ we find

$$(\alpha_2 B - \beta_2 Ak) \tan kL + \alpha_2 A + \beta_2 Bk = 0 \quad \text{(C116)}$$

or

$$\tan kL = \frac{\beta_2 Bk + \alpha_2 A}{\beta_2 Ak - \alpha_2 B} \quad \text{(C117)}$$

From (C114) we have $Bk = -\alpha_1 A/\beta_1$ (if $\beta_1 \neq 0$), and substituting this into (C117) gives

$$\tan kL = \frac{-\alpha_1 \beta_2 k + \alpha_2 \beta_1 k}{\beta_1 \beta_2 k^2 + \alpha_2 \alpha_1} \quad \text{(C118)}$$

Similar reasoning provides the same result for the case in which $\beta_1 = 0$. It is convenient to express this equation in the form

$$\tan \mu = \frac{(\alpha_2 \bar{\beta}_1 - \alpha_1 \bar{\beta}_2) \mu}{\alpha_1 \alpha_2 + \bar{\beta}_1 \bar{\beta}_2 \mu^2} \quad \text{(C119)}$$

where $\mu \equiv kL$ and $\bar{\beta}_i \equiv \beta_i/L$. The solution is illustrated in Fig. 32. Equation (C119) will provide the mode numbers $\mu_n$ for $n = 0, 1, 2, \cdots$, which are used to calculate the wave numbers

$$k_n = \frac{\mu_n}{L} \quad \text{(C120)}$$

Note that $\mu_0 = 0$, and therefore $k_0 = 0$. The solution now takes the form

$$Y_n(y) = A_n \cos k_n y + B_n \sin k_n y \quad \text{(C121)}$$

$$A_n = \frac{-\beta_1 k_n}{\alpha_1} B_n \quad \text{(C122)}$$
where \( \alpha_1 \neq 0 \). The case of \( \alpha_1 = 0 \) will be handled separately. Setting \( B_n = 1 \) for convenience, the solution \([C121]\) can be expressed as

\[
Y_n(y) = \sin k_n y - \frac{\beta_1 k_n}{\alpha_1} \cos k_n y ,
\]

while the general solution takes the form

\[
Y(y) = \sum_{n=1}^{\infty} D_n Y_n(y) .
\]

Note that the \( n = 0 \) term does not contribute, and the Fourier coefficients are given by

\[
D_n = \frac{1}{N_n} \int_0^L dy T_0(y) Y_n(y) .
\]

Also note that the modes are orthogonal,

\[
\int_0^L dy Y_n(y) Y_m(y) = 0 \quad \text{for } n \neq m ,
\]

with the normalization factor \( N_k \) determined by

\[
\int_0^L dy Y_n^2(y) = \frac{1}{4k_n\alpha_1^2} \left[ -2\alpha_1 \beta_1 k_n + 2(\beta_1^2 k_n^2 + \alpha_1^2) k_n L + 2\alpha_1 \beta_1 k_n \cos 2k_n L + (\beta_1^2 k_n^2 - \alpha_1^2) \sin 2k_n L \right] .
\]

That is to say,

\[
\int_0^L dy Y_n(y) Y_m(y) = N_n \delta_{nm} ,
\]

\[
N_n = \frac{1}{4k_n\alpha_1^2} \left[ -2\alpha_1 \beta_1 k_n + 2(\beta_1^2 k_n^2 + \alpha_1^2) k_n L + 2\alpha_1 \beta_1 k_n \cos 2k_n L + (\beta_1^2 k_n^2 - \alpha_1^2) \sin 2k_n L \right] .
\]

It is convenient for numerical work to express this in terms of \( A_n \) and \( B_n \) coefficients:

\[
Y(y) = \sum_{n=1}^{\infty} D_n \left[ -\frac{\beta_1 k_n}{\alpha_1} \cos k_n y + \sin k_n y \right] \quad \text{(C130)}
\]

\[
= \sum_{n=1}^{\infty} \left[ A_n \cos k_n y + B_n \sin k_n y \right] \quad \text{with} \quad \text{(C131)}
\]

\[
A_n = -\frac{\beta_1 k_n}{\alpha_1} D_n \quad \text{and} \quad B_n = D_n .
\]
The temperature \( \tilde{T}(y, t) \) is therefore,

\[
\tilde{T}(y, t) = \sum_{n=1}^{\infty} \left[ A_n \cos k_n y + B_n \sin k_n y \right] e^{-\kappa k_n^2 t} \tag{C132}
\]

\[
B_n = \frac{1}{N_n} \int_0^L dy T_0(y) Y_n(y) \tag{C133}
\]

\[
A_n = -\frac{\beta_1 k_n}{\alpha_1} B_n . \tag{C134}
\]

For \( T_0^a(y) = T_1 \) we have

\[
B_n^a = \frac{T_1}{N_n} \left[ \frac{1 - \cos k_n L}{k_n} - \frac{\beta_1 \sin k_n L}{\alpha_1} \right] . \tag{C135}
\]

For \( T_0^b(y) = (T_2 - T_1) y/L \) we have

\[
B_n^b = \frac{T_2 - T_1}{N_n L} \left[ \beta_1 k_n - (\alpha_1 k_n L + \beta_1 k_n) \cos k_n L + (\alpha_1 - \beta_1 k_n^2) \sin k_n L \right] , \tag{C136}
\]

with \( B_n = B_n^a + B_n^b \).

![Graph](image)

**FIG. 33:** The roots \( \mu_n \) for \( \alpha_1 = 0, \alpha_2 = 1, \) and \( \bar{\beta}_2 = 1 \). For \( L = 2 \) we have \( \beta_2 = 2 \).

Let us now consider the case of \( \alpha_1 = 0 \), so that \( \text{(C119)} \) becomes

\[
\tan \mu = \frac{a}{\mu} \quad \text{with} \quad a = \alpha_2 / \bar{\beta}_2 . \tag{C137}
\]

We can find an approximate solution for large values of \( \mu \): since the RHS is very small for \( \mu \gg 1 \), we must solve \( \tan \mu = 0 \), and therefore \( \mu^{(0)}_n = n\pi \). The exact solution can be expressed as \( \mu_n = n\pi + h, \) where \( 0 < h \ll 1 \), and we find \( \text{LHS} = \tan(n\pi + h) = \tan(h) = h + \mathcal{O}(h^2) \). Similarly, \( \text{RHS} = a/(n\pi + h) = (a/n\pi)(1 + h/n\pi)^{-1} = (a/n\pi)(1 - h/n\pi) + \mathcal{O}([h/n]^2) = a/n\pi - ah + \mathcal{O}(h^2/n^2) \), thus

\[
h = \frac{a}{n\pi} - ah \quad \Rightarrow \quad h = \frac{a}{1 + a/n\pi} \tag{C138}
\]

56
and the first order solution becomes

\[ \mu_n^{(1)} = n\pi + \frac{a}{1 + a} \frac{1}{n\pi} + \mathcal{O}(1/n^2). \]  

(C139)

This can be used as an initial guess when using an iteration method to find the \( \mu_n \). The solution is

\[
T(y,t) = \sum_{n=1}^{\infty} A_n Y_n(y) e^{-\kappa k_n^2 t}
\]  

(C140)

\[
Y_n(y) = \cos k_n L
\]  

(C141)

\[
\int_0^L dy Y_n(y) Y_m(y) = N_n \delta_{nm}
\]  

(C142)

\[
N_n = \frac{1}{4k_n} \left[ 2k_n L + \sin 2k_n L \right],
\]  

(C143)

and

\[
A_n = \frac{1}{N_n} \int_0^L dy T(y,0) Y_n(y)
\]  

(C144)

\[
= \frac{T_1}{k_n} \sin k_n L + \frac{T_2 - T_1}{k_n^2 L} \left[ -1 + \cos k_n L + k_n L \sin k_n L \right].
\]  

(C145)

3. The General Solution

a. BC1: Given Temperatures \( T_1 \) and \( T_2 \)

\[
T(y,t) = T_1 + \frac{T_2 - T_1}{L} y + \sum_{n=1}^{\infty} B_n \sin k_n y e^{-\kappa k_n^2 t}
\]  

(C146)

\[
B_n = \frac{2T_a - 2T_b (-1)^n}{n\pi}
\]  

(C147)

\[
k_n = \frac{n\pi}{L}.
\]  

(C148)

b. BC2: Given Flux \( F \)

\[
T(y,t) = F y + \frac{A_0}{2} + \sum_{n=1}^{\infty} A_n \cos k_n y e^{-\kappa k_n^2 t}
\]  

(C149)

\[
A_0 = T_a + T_b
\]  

(C150)

\[
A_n = 2 \left( T_a - T_b \right) \frac{1 - (-1)^n}{n^2 \pi^2}
\]  

(C151)

\[
k_n = \frac{n\pi}{L}.
\]  

(C152)
c. $BC3$: Given $T_1$ and $F_2$

\[ T(y, t) = T_1 + F_2 y + \sum_{n=0}^{\infty} B_n \sin k_n y e^{-\kappa k_n^2 t} \quad (C153) \]

\[ B_n = \frac{4T_B}{(2n+1)\pi} - \frac{8(T_B - T_A)}{(2n+1)^2 \pi^2} \quad (C154) \]

\[ k_n = \frac{(2n+1)\pi}{2L} . \quad (C155) \]

d. $BC4$: Given $T_2$ and $F_1$

\[ T(y, t) = (T_2 - F_1 L) + F_1 y + \sum_{n=0}^{\infty} A_n \cos k_n y e^{-\kappa k_n^2 t} \quad (C156) \]

\[ A_n = 4T_A \frac{(-1)^n}{(2n+1)\pi} - 8(T_B - T_A) \frac{1 - (-1)^n}{(2n+1)^2 \pi^2} \quad (C157) \]

\[ k_n = \frac{(2n+1)\pi}{2L} . \quad (C158) \]
[1] D.E. Burton, *Multidimensional discretization of conservation laws for unstructured polyhedral grids*, Technical Report UCRL-JC-118306, Lawrence Livermore National Laboratory, August 1994, SAMGOP-94: 2nd International Workshop on Analytical Methods and Process Optimization in Fluid and Gas Mechanics, VNIIEF, Holiday Base, Arzamas-16, Russia, September 10-16, 1994; D.E. Burton, *Connectivity structures and differencing techniques for staggered-grid free-Lagrange hydrodynamics*, Lawrence Livermore National Laboratory, Report No. UCRLJCl10555 (1992); D.E. Burton, *Consistent finite-volume discretization of hydrodynamic conservation laws for unstructured grids*, Lawrence Livermore National Laboratory, Report No. UCRLJCl18788 (1994).

[2] M. Shashkov, *Algorithms for improved solution robustness, accuracy and mesh adaptivity*, LA-UR-15-23913, Los Alamos report (2015).

[3] A. Dawes, C. Malone, M. Shashkov, *Some New Verification Test Problems for Multimaterial Diffusion on Meshes that are Non-Aligned with Material Boundaries*, LA-UR-16-24696, Los Alamos report (2016).

[4] A. Dawes, *3D Multimaterial Polyhedral Methods for Diffusion*, MultiMat Conference, Würzburg, Germany, 2015.

[5] Robert L. Singleton Jr, *The Planar Sandwich and Other 2D Planar Heat Flow Test Problems in ExactPack*, LA-UR-17-20460, Los Alamos report (2017), arXiv:1701.07342v1 [physics.comp-ph].

[6] ExactPack source code, LA-CC-14-047, available at https://github.com/lanl/exactpack.

[7] R.V. Garimella and K. Lipnikov, *Solution of the diffusion equation in multimaterial domains by subdivision of elements along reconstructed interface* Int. J. Numer. Meth. Fluids (2011) 1423.

[8] E. Kikinzon, Y. Kuznetsov, and M. Shashkov, *New algorithm for solving multimaterial diffusion problems on mesh non-aligned with material interfaces*, LA-UR-16-21548, Los Alamos Report (2016).

[9] E. Kikinzon, Y. Kuznetsov, and M. Shashkov, *Approximate static condensation algorithm for solving multimaterial diffusion problems on meshes non-aligned with material interfaces*, J. Comp. Phys, 347, 416 (2017).

[10] Personal notes, R. Singleton, Summer 2017.

[11] Paul W Berg and James L McGregor, *Elementary Partial Differential Equations*, Holden Day (1966).

[12] Walter Rudin, *Principles of Mathematical Analysis*, McGraw-Hill, third edition (1976).

[13] James R. Kamm, Jerry S. Brock, Christopher L. Rousculp, William J. Rider, *Verification of an ASCI Shavano Project Hydrodynamics Algorithm*, LA-UR-03-6999 Los Alamos Report (2003).