THE UNIVERSITY OF SYDNEY

The University of Sydney

School of Aerospace, Mechanical and Mechatronic Engineering

Major Project - 3D Transient Cooling Fin Analysis

Author:	**SID:**
Tom Bullock	530754154
Iestyn Hughes	530705835

An Assignment submitted for the UoS:

AMME3060 - Engineering Methods

November 5, 2023

Contents

1 Introduction and Problem Description

To operate efficiently, a computer's central processing unit (CPU) must be held in a suitable temperature window. When switched on, the CPU temperature of a computer begins to rise. Heat must be shed efficiently to maintain a suitable operating temperature that is safe and efficient, minimising thermal throttling. The motherboard must also be shielded from excessive damaging temperatures. Engineers have developed a breadth of cooling solutions to achieve this. A popular approach deploys cooling 'fins' that exploit their surface area to provide convective cooling. The temperature distribution within such geometries can be difficult to evaluate - an analytic solution is impractical and time-consuming. Numerical methods serve as powerful conduits to overcome this challenge.

A single fin from a repeating heat sink geometry was chosen for investigation. The upper fin surface, and underside of the motherboard, were exposed to convective cooling from ambient air. All other external surfaces were considered to be insulated.

To explore the temperature distribution on start-up, the heat flow in the fin was modelled as a time-variant response initialised at room temperature ($T_\alpha = 20°$). The heat radiated by the CPU was captured by a volumetric heat source term q_v in a constrained region. The compound fin was made up of 3 materials, each with their own thermal conductivity κ, specific heat capacity c_p and density ρ. The governing equation to be solved numerically was the 3D transient heat conduction equation, given by Equation 1.

$$\rho c_p \frac{\partial T}{\partial t} = \kappa \left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} + \frac{\partial^2 T}{\partial z^2} \right) + q_v \tag{1}$$

A numerical model was developed in MATLAB following the spatio-temporal discretisation of Equation 1 as outlined in Section 2. ANSYS' transient thermal solver was used to produce an alternative solution for the same problem. Each model underwent accuracy testing before the techniques were compared for final analysis and program verification.

2 Development of Custom Solver in MATLAB

The heat transfer within the fin was described by the governing equation, Equation 1, such that a Galerkin Finite Element solution could be developed for spatial discretisation. A first-order backwards temporal discretisation was completed such that a linear solver could be implemented. The full derivation was given in Appendix A.

2.1 Spatial Discretisation

The residual equation for the 2D problem was given as Equation 2. From this, the minimised residual equation for a local element was defined as the integral of the interpolation function N_i and residual equation $\mathscr{R}(x, y)$ over the element volume V. The transient term was extracted to produce Equation 3.

$$\rho c_p \frac{\partial T}{\partial t} - \kappa \left(\frac{\partial^2 T}{\partial y^2} + \frac{\partial^2 T}{\partial y^2} + \frac{\partial^2 T}{\partial z^2} \right) - q_v(x, y, z) = \mathscr{R}(x, y, z) \tag{2}$$

$$\iiint_V \left(\kappa \left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} + \frac{\partial^2 T}{\partial z^2} \right) + q_v(x, y, z) \right) N_i \; dV = \rho c_p \iiint_V \frac{\partial T}{\partial t} N_i \; dV$$

$$\text{for } i = 1, 2, 3, 4 \tag{3}$$

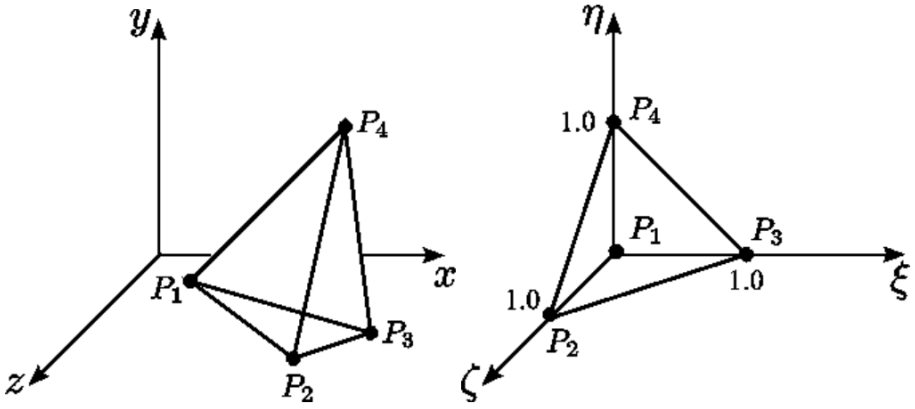


Figure 1: Tetrahedral finite element shown in both the traditional Cartesian coordinate system and a transformed natural coordinate system [2].

The shape functions, N_i, were formed by geometric relationships from the tetrahedral element as interpolations between each node. The transformation to a natural coordinate system enabled the analysis [3]. The linear shape functions were given as Equation 18 (Appendix A, Section 5.1.1)

Integration by parts and Stokes' theorem were used to break up the second-order derivatives and separate Equation 3 into its constituent terms, producing Equation 4. l_x, l_y, and l_z were used to denote the x,y, and z direction cosines of the outward-drawn normals.

$$\iiint_V \left(-\kappa\frac{\partial N_i}{\partial x}\frac{\partial T}{\partial x} - \kappa\frac{\partial N_i}{\partial y}\frac{\partial T}{\partial y} - \kappa\frac{\partial N_i}{\partial z}\frac{\partial T}{\partial z}\right) N_i \ dV + \iiint_V q_v N_i \ dV + \kappa \iint_S N_i \left[\frac{\partial T}{\partial x}l_x + \frac{\partial T}{\partial y}l_y + \frac{\partial T}{\partial z}l_z\right] \ ds = \rho c_p \iiint_V \frac{\partial T}{\partial t} N_i \ dV \quad \text{for } i = 1, 2, 3, 4 \tag{4}$$

Each term was converted to a matrix form for concision; [**B**] was formed from the points as generated by the mesh, whilst [**N**] was formed from the shape functions defined for linear tetrahedral elements (Appendix A, Section 5.1.1). In addition, terms containing T were evaluated as $T = [\mathbf{N}]\{\mathbf{T}\}$ so as to be functions of the element's nodal temperatures. The direction cosine terms were coalesced using Equation 5 to produce Equation 6.

$$\frac{\partial T}{\partial x}l_x + \frac{\partial T}{\partial y}l_y + \frac{\partial T}{\partial z}l_z = \frac{\partial T}{\partial n} \tag{5}$$

$$\kappa \iiint_V [\mathbf{B}]^T [\mathbf{B}] \{\mathbf{T}\} \ dV + \rho c_p \iiint_V [\mathbf{N}]^T [\mathbf{N}] \{\mathbf{T}'\} \ dV = \kappa \iint_S [\mathbf{N}]^T \frac{\partial T}{\partial n} \ ds + q_v \iiint_V [\mathbf{N}]^T \ dV \tag{6}$$

The surface integral term was identified as capturing the boundary conditions of the problem for the faces of a given element. For three nodes bounding a convective face, the boundary condition term was given by Equation 7.

$$\begin{aligned} \kappa \iint_S [\mathbf{N}]^T \frac{\partial T}{\partial n} \ ds &= -h \iint_S [\mathbf{N}]^T [\mathbf{N}] \{\mathbf{T}\} \ ds + T_\alpha h \iint_S [\mathbf{N}]^T \ ds \\ &= -\frac{hA_{jkl}}{12} \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix} \{\mathbf{T}\} + \frac{hA_{jkl}T_\alpha}{3} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \end{aligned} \tag{7}$$

Finally, the resolved boundary condition term was substituted into Equation 6 to form the final discrete equation for the element, Equation 8. The terms were collected into a local stiffness matrix [**k**], mass/capacitance matrix [**c**] and load vector [**f**] as given by Equation 9.

$$\frac{\rho c_p V}{20} \begin{bmatrix} 2 & 1 & 1 & 1 \\ 1 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 2 \end{bmatrix} \{\mathbf{T}'\} + \left(\kappa [\mathbf{B}]^T [\mathbf{B}] V + \frac{hA_{jkl}}{12} \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}\right) \{\mathbf{T}\} = \frac{q_v V}{4} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} + \frac{hA_{jkl}T_\alpha}{3} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \tag{8}$$

$$[\mathbf{c}]\{\mathbf{T}'\} + [\mathbf{k}]\{\mathbf{T}\} = [\mathbf{f}] \tag{9}$$

2.2 Temporal Discretisation

$\{\mathbf{T}'\}$ terms were discretised using a first order backwards finite difference method. $\{\mathbf{T}\}$ terms were discretised at the $n+1$ time level to produce Equation 10.

$$\left.\frac{dT_i}{dt}\right|_{n+1} = \frac{T_i^{n+1} - T_i^n}{\Delta t} + \mathcal{O}(\Delta t) \quad \text{for } i = 1, 2, 3, 4 \tag{10}$$

Equation 9 was discretised at $n+1$ time-step; Equation 10 was then used to discretise the $\{\mathbf{T}'\}$ terms forming Equation 11.

$$[\mathbf{c}]\left\{\mathbf{T^{n+1}}\right\} - [\mathbf{c}]\left\{\mathbf{T^n}\right\} + \Delta t\,[\mathbf{k}]\left\{\mathbf{T^{n+1}}\right\} = \Delta t\,[\mathbf{f}] \tag{11}$$

The terms were collected and the final equation was given as Equation 14.

$$([\mathbf{c}] + \Delta t\,[\mathbf{k}])\left\{\mathbf{T^{n+1}}\right\} = \Delta t\,[\mathbf{f}] + [\mathbf{c}]\left\{\mathbf{T^n}\right\} \tag{12}$$

$$[\mathbf{M}]\left\{\mathbf{T^{n+1}}\right\} = [\mathbf{R}] \tag{13}$$

$$\left\{\mathbf{T^{n+1}}\right\} = [\mathbf{M}]^{-1}[\mathbf{R}] \tag{14}$$

The solution domain was moved forward in time by calculating Equation 14 for a given time step until the total simulation time was reached.

2.3 Solver Selection and Justification

The direct linear solver built into MATLAB was used, mldivide(), for the development of the custom solver mostly due to its ease of use and time constraints on the project. The development of an iterative solver would be beneficial, as it would reach a solution faster. The reason for this was two-fold; the simple geometry would allow for a high-quality well-conditioned mesh, and the final matrix of equations were diagonally dominant. The factors allow an iterative solver to perform more efficiently than a direct linear solver.

The transient discretisation scheme used was first order backwards (FOB), this removed a stability limitation found in an equivalent first order forwards scheme. This allowed for very high time steps to be used without causing stability issues.

2.4 Validation

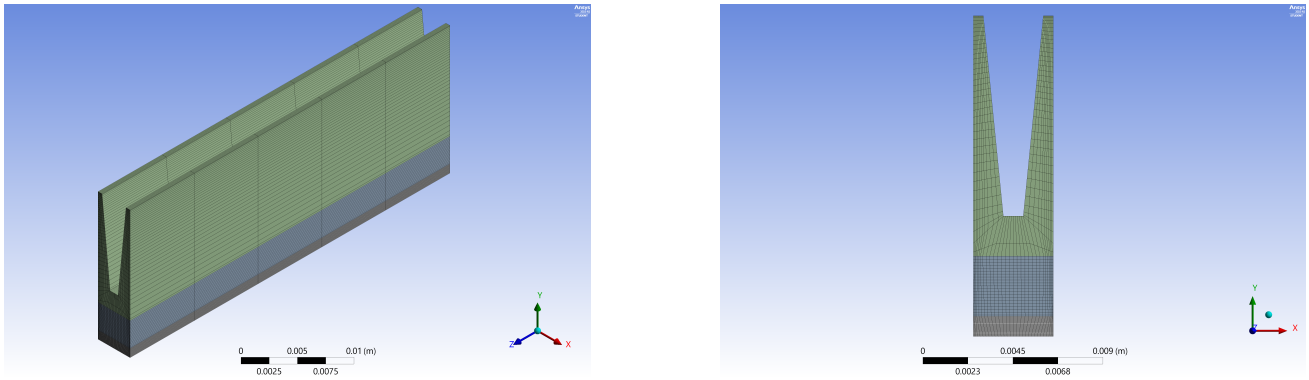
The accuracy of the MATLAB solution was verified using three techniques. Firstly, a grid convergence test was completed for the steady-state results. Then, a time-step convergence test was completed to assess the accuracy of the transient scheme used. The accuracy of the code was further verified through direct comparison with an ANSYS Transient Thermal solution for the same model with the same initial conditions. Taken together, the validation strategy was selected to provide a strong foundation on which engineering conclusions were built.

3 Development of Model within Commercial Package

3.1 Meshing Strategy

Commerical Package: ANSYS 2023 Student

The shape of the cooling fin allowed for various meshing strategies to be implemented. As the chip and motherboard of the fin were cuboids, and the temperature gradient varied primarily in the Y axis, it was appropriate to implement a coarse quad swept mesh in the Z direction and a fine mesh in the X and Y directions. This was done with an 'edge sizing' method with 20 divisions. There was a large temperature variation across the motherboard-chip interface, hence a denser mesh was used to better capture the variation in this part of the geometry. The quad mesh ensured the residual error was weighted equally between all nodes, a desirable property with temperature gradients. The mesh is shown in Figure 2.



(a) (b)

Figure 2: Plots of (a) the fin in an isometric view and (b) the fin viewed from the side.

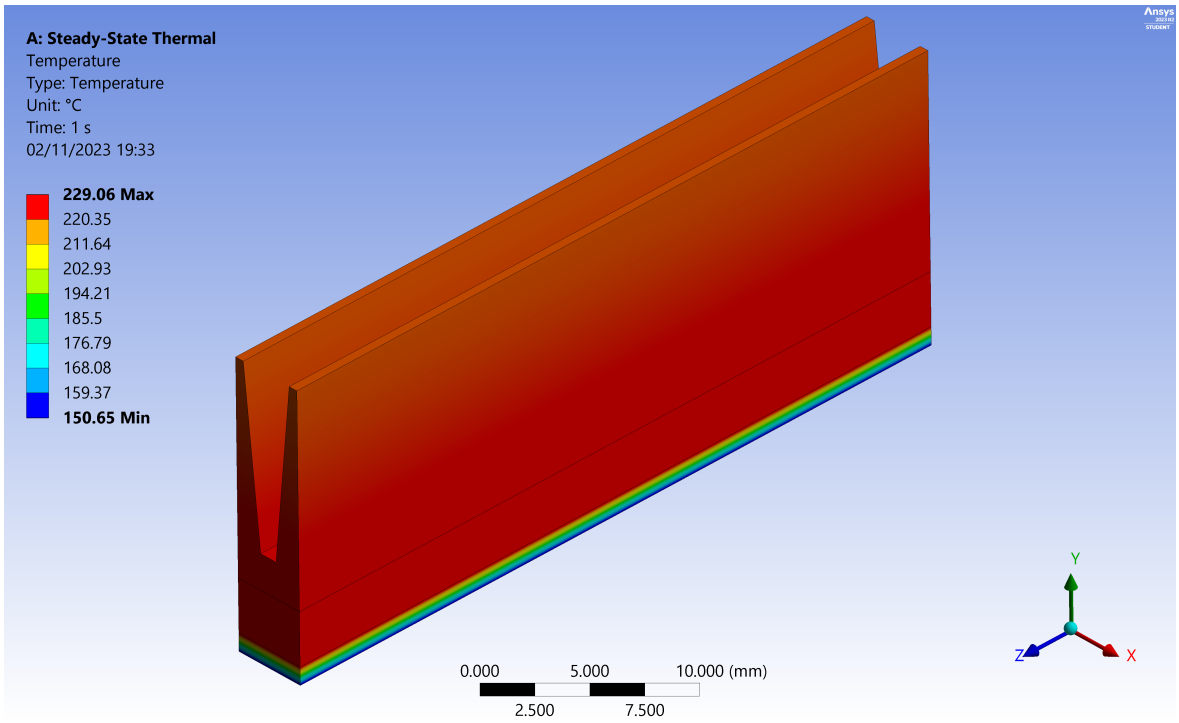


Figure 3: Temperature contour plot of the cooling fin once in steady state

3.1.1 Mesh Settings

The non-default mesh settings were shown in Table 1.

Table 1: Non-Default Meshing Parameters

Parameter	Value
Mesh	Swept method along Z and edge sizing on face
Element Order	Quadratic
Element Size	5 swept divisions along the Z and an edge sizing of 20 divisions on the face
Use Adaptive Sizing	Yes
Mesh Type	All Quad

3.1.2 Solver Settings

For the reasons discussed in Section 2.3 the ANSYS JCG iterative solver was used. However, for comparison to the MATLAB solution this was changed to the ANSYS direct solver to ensure a fair comparison.

3.2 Mesh Demonstration

The geometry of the Fin used, and hence mesh required, was basic. Another mesh strategy on a full CPU heat sink was shown in Figures 4 and 5 as a starting point from which to extend the project. NB: this was for demonstration purposes only and was not used elsewhere.

3.2.1 Mesh Strategy

A sweeping method was not possible, as the source and target faces could not be defined for mesh projection. The main features that made this geometry more challenging to mesh were the holes and the fins with rounded tops. For those reasons, as well as the model being homogeneous, a patch-conforming tetrahedral mesh with inflation layers around the holes was implemented. Additionally, edge sizing was placed on the round edges to maintain an acceptable mesh quality and accurately capture the temperature gradient across the tips.

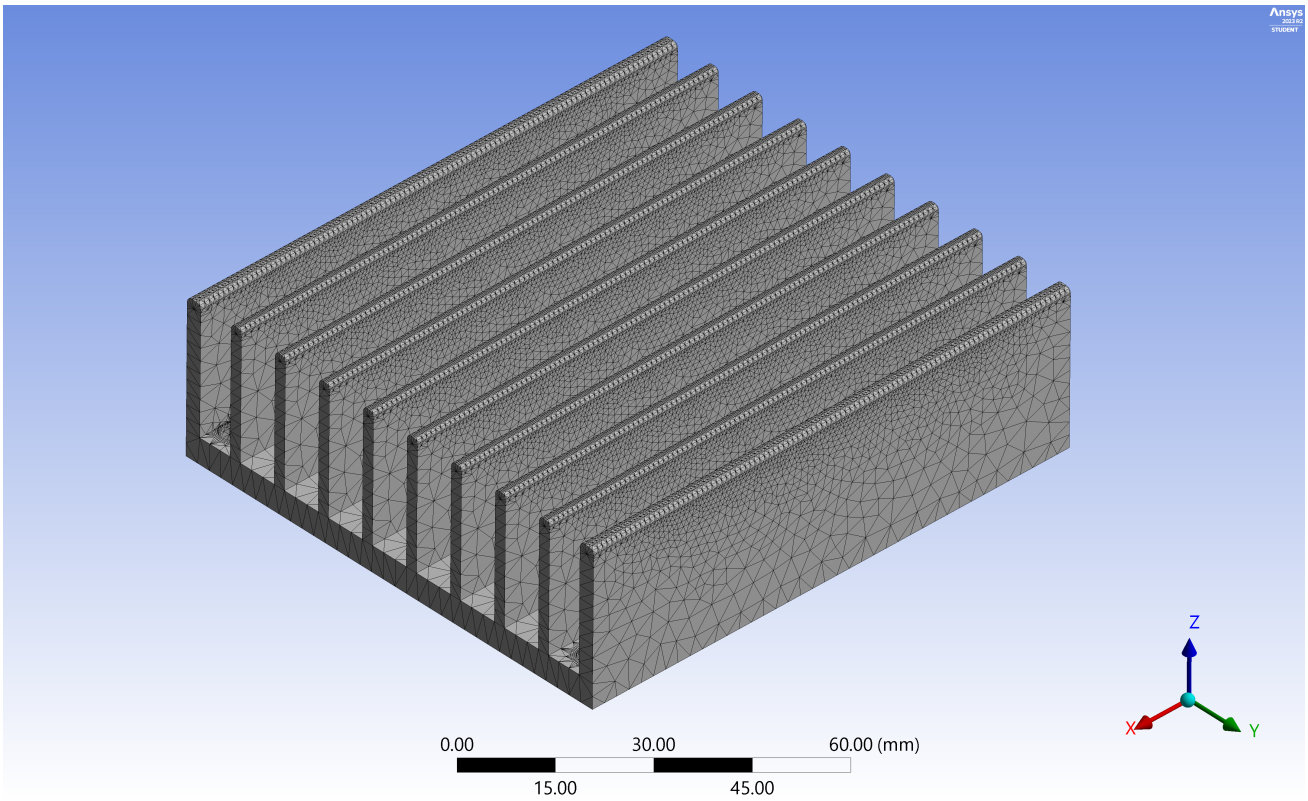


Figure 4: Isometric view of the meshed heat sink highlighting the edge sizing on the tips

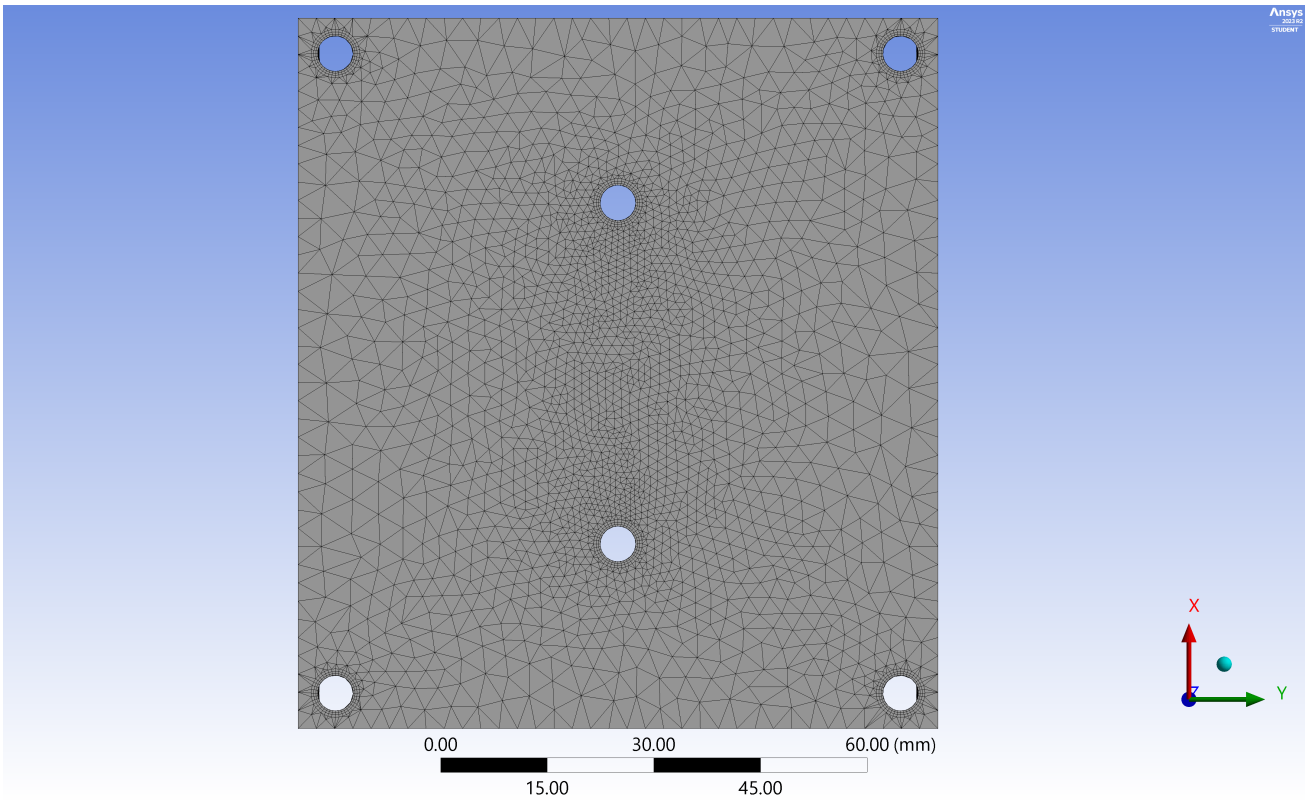


Figure 5: Underside of the heat sink highlighting the inflation layers around the holes

3.2.2 Mesh Settings

The non-default mesh settings used were outlined in Table 2.

Table 2: Non-Default Meshing Parameters

Parameter	**Value**
Mesh	Patch Conforming Tetrahedrons
Edge Sizing	Edges of the holes - 40 Divisions Tops of cooling fins - 300 Divisions Behaviour - Hard
Face Meshing	Placed on the inside face of the holes
Element Order	Quadratic
Global Element Size	5mm
Use Adaptive Sizing	Yes
Mesh Type	Tetrahedrons
Inflation Layers	Inflation Option - Total Thickness Number of layers - 3 Growth rate - 1.2 Max. thickness - 1mm

4 Results, Accuracy, Discussion and Physical Interpretation

4.1 ANSYS Accuracy Testing

4.1.1 Element Size Convergence

A grid size convergence was conducted in steady state and the results were shown in Table 3 and plotted in Figure 6.

Table 3: ANSYS Outputs with Varying Element Size in Steady State

Element Size [mm]	Element Qual.	Min Temp/°C	Max Temp/°C	Average Temp/°C
7.5	0.34	150.63	229.03	214.95
5	0.36	150.64	229.03	214.95
2.5	0.70	150.64	229.04	220.37
1	0.83	150.64	229.04	220.37
0.75	0.89	150.65	229.06	219.19
0.5	0.87	150.65	229.06	220.95

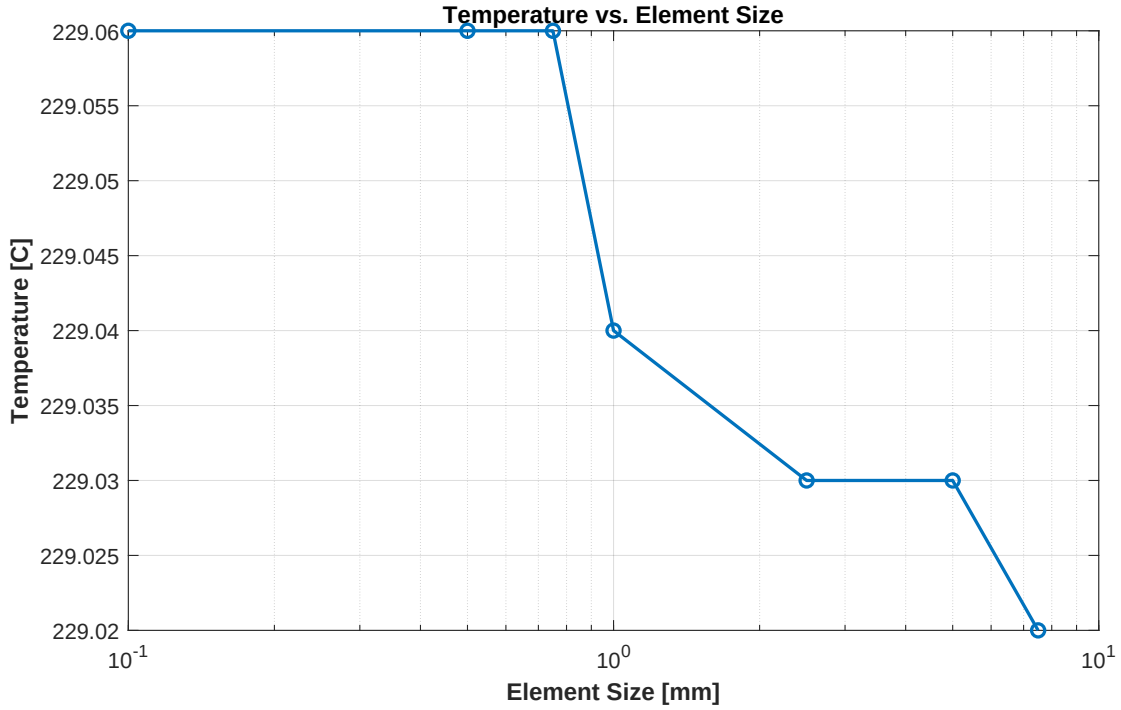


Figure 6: Graph showing how maximum temperature converges with element size in a steady state test

The range of temperatures in Figure 6 was very small despite a large difference in element size. This was largely due to the simplicity of the geometry and physics in the problem.

4.1.2 Time Step Convergence

A time-step divided difference test was conducted in ANSYS and recorded in Table 4.

Table 4: Divided difference of time step

Time Step	Max. Temperature at 150s	2^n	n
128	192.8536072		
64	210.6373138		
32	221.5089417	1.635790608	0.709988086
16	224.3210754	3.865971047	1.950830833
8	225.9339752	1.743526674	0.802008436
4	226.5773315	2.507008515	1.325966896
2	226.8727417	2.177840909	1.122898569
1	227.0184021	2.028074586	1.020110711
0.5	227.0896454	2.044549154	1.031782748
0.25	227.1248627	2.022963605	1.016470365

Table 4 shows that the order accuracy tends to 1, confirming first-order accuracy - the same as the time discretisation scheme. Therefore, the solution was converging, in the asymptotic region and accurate.

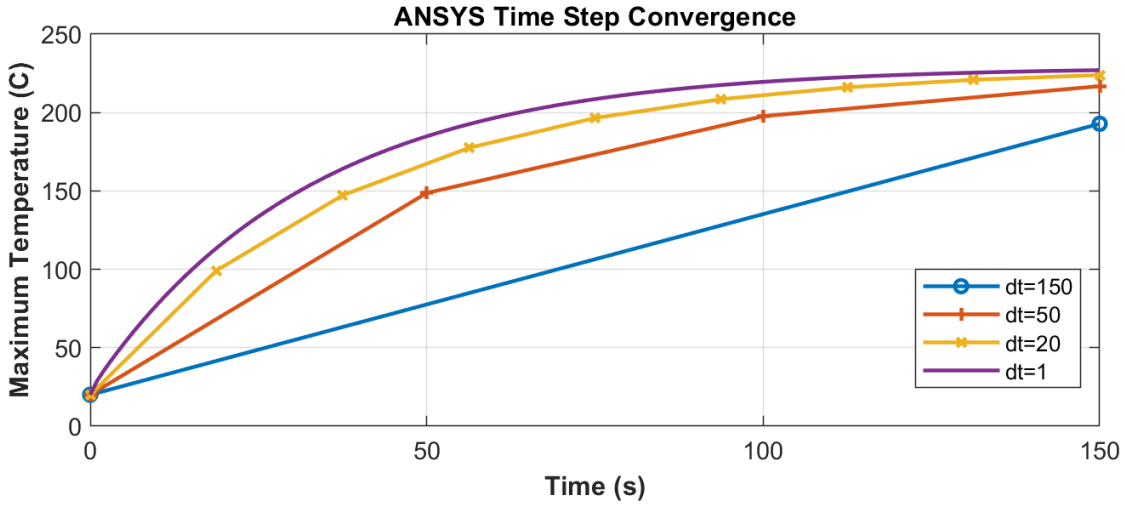


Figure 7: Graph showing time step convergence over the whole time domain with varying time steps

4.1.3 Residual Independence

The transient solution was further verified by examining how maximum temperature changed with the residual criterion. The results were shown in Figure 8.

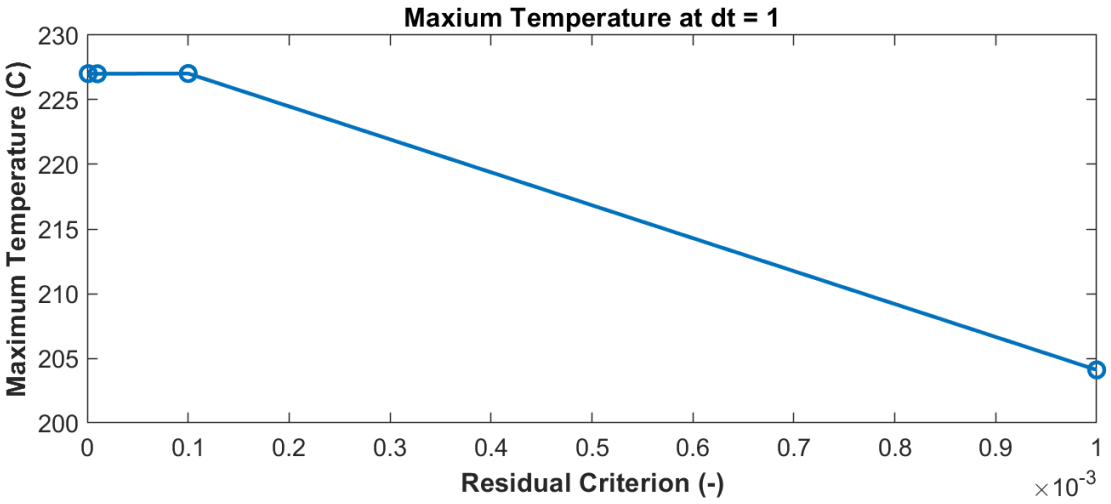


Figure 8: Graph showing that the solution becomes independent of residual

The results showed that for a time step of 1 second, the transient solution becomes independent of residual at a residual criterion of 0.1×10^{-3}. This value was used in further testing to ensure maximum computational efficiency.

4.2 MATLAB Accuracy Testing

4.2.1 General solution results

To visualise the transient heat flow in the fin on CPU startup, the temperature constrained along a vertical (Z-axis) edge was determined for each time-step. This was plotted as Figure 9.

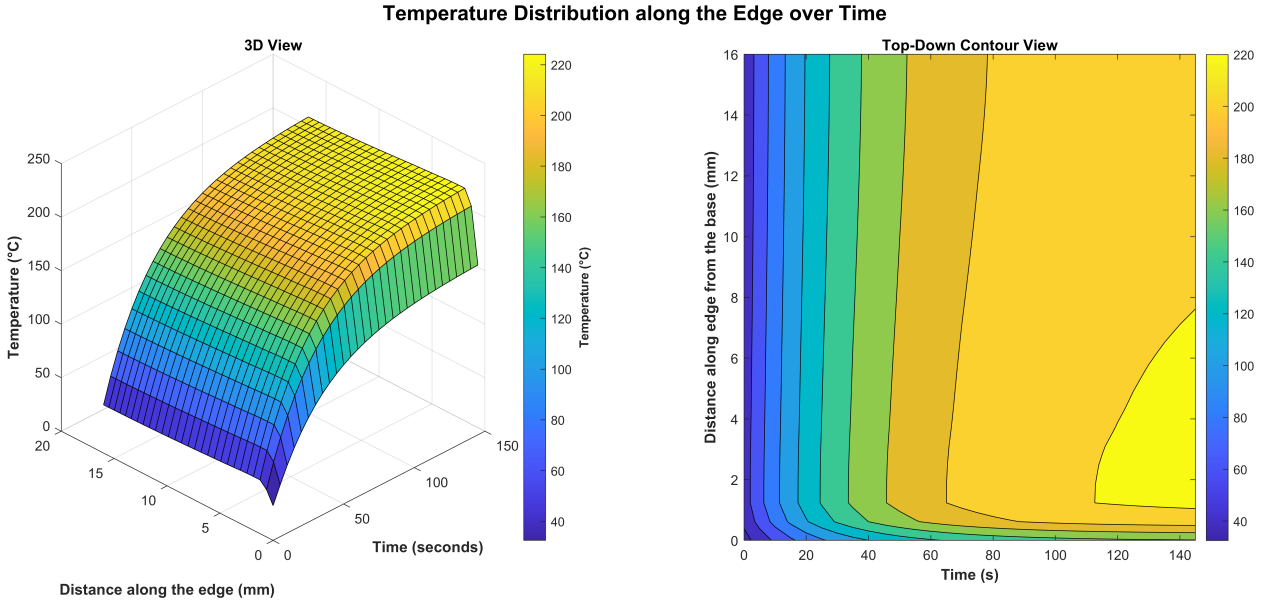
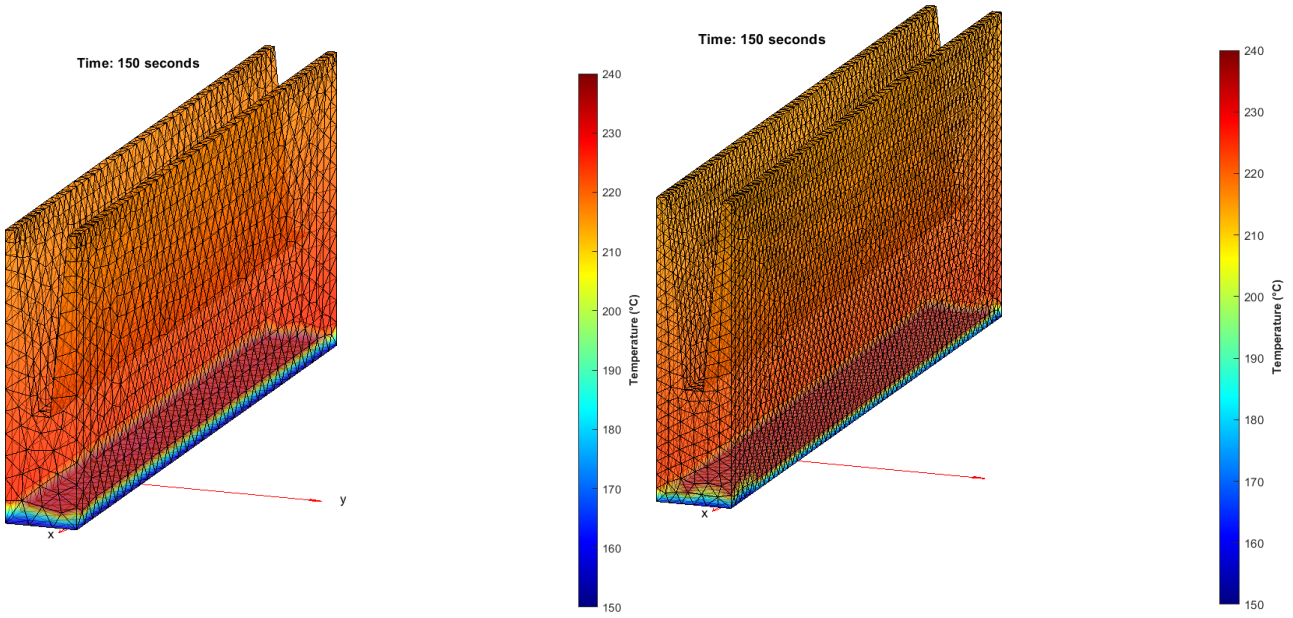


Figure 9: Transient surface and contour plots showing the temperature distribution along a Z-axis edge of the model as the computer is switched on from room temperature. The noticeable lip reflects the lower temperature of the motherboard material due to its lower thermal conductivity κ.

4.2.2 Model plots and grid convergence testing

For an easily computable coarse mesh, the temperature distribution was plotted as Figure 10a. The mesh was then refined and used to generate Figure 10b. The difference in the steady state maximum temperature T_{max} in the fin for the coarse and fine meshes was $9.35 \times 10^{-1}\%$ indicating a high degree of mesh convergence and solution accuracy. These temperature distributions were plotted as Figure 10.



(a) (b)

Figure 10: The calculated temperature distribution in (a) a coarse mesh and (b) a more refined mesh. These distributions were closely aligned.

Due to computational limits, it was not possible to analyse extremely fine meshes. Furthermore, model intricacies ensured that larger grid sizes were unable to capture the different materials and regions appropriately. As such, the grid convergence study was limited in scope for the MATLAB solution.

4.2.3 Time-step convergence testing

To complete a time-step convergence study, T_{max} in the fin was plotted throughout the transient response. This was repeated for different time-step (dt) sizes to produce Figure 11.

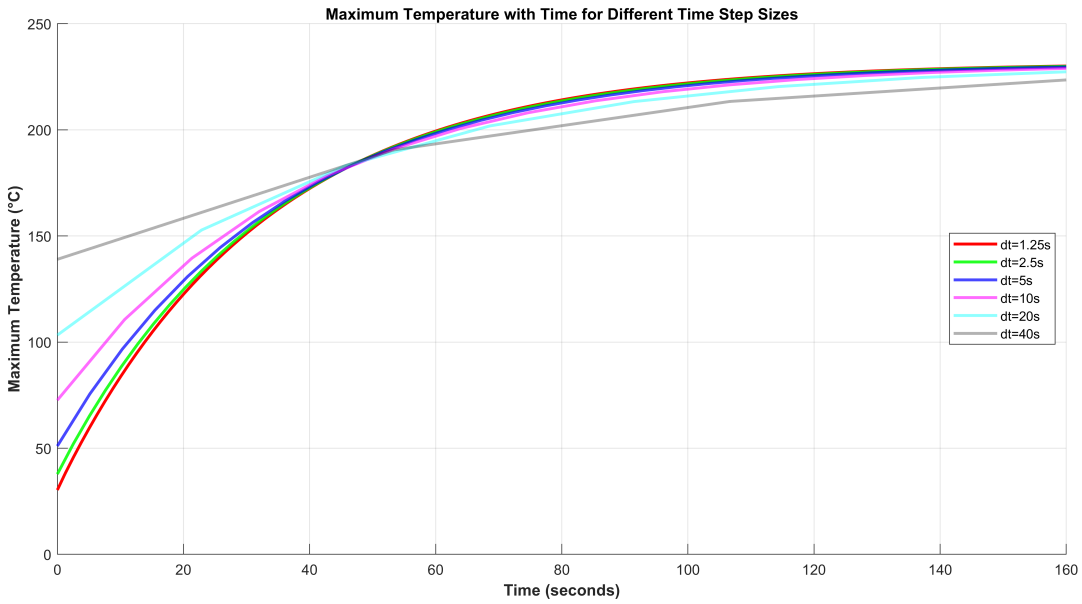


Figure 11: Plot showing the transient values of T_{max} for different time-step sizes. A clear convergence was demonstrated, trending towards an infinitesimal dt.

4.3 MATLAB vs ANSYS Comparison Testing

To confirm that the MATLAB solution was accurate, it was compared with the results derived from the ANSYS model detailed in Section 3. T_{max} and T_{min} were selected as analysis parameters as their values exhibited minimal dependence on the meshing strategy selected for each model. T_{max} and T_{min} were plotted for the the transient response from each method, with a time-step of 1s as shown in Figure 12.

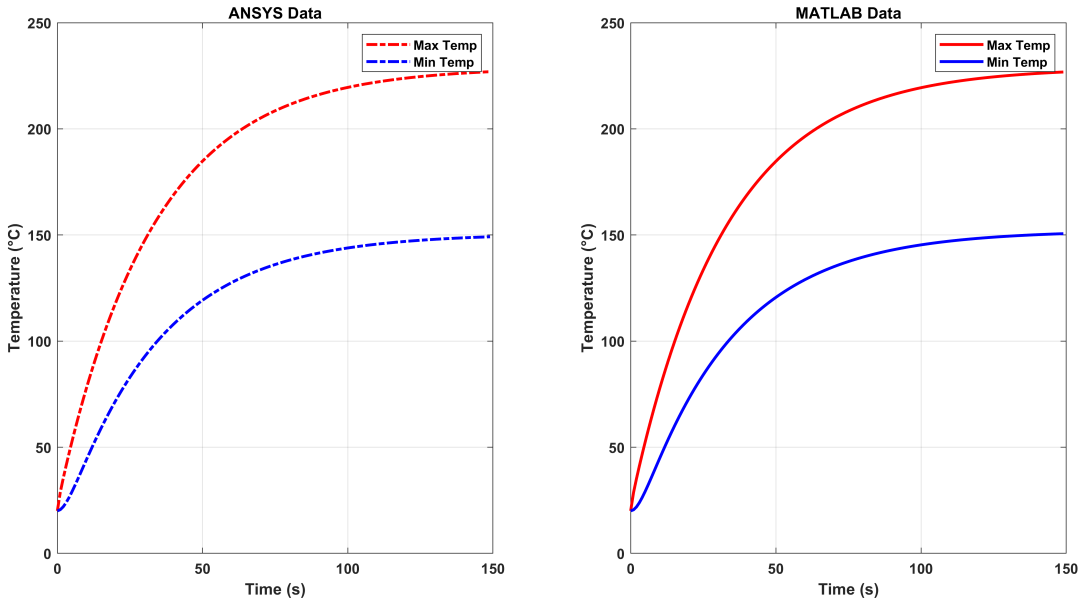


Figure 12: Plots showing T_{max} and T_{min} for the ANSYS and MATLAB solutions. Both models demonstrated clear asymptotic behaviour, trending towards the expected steady state solution. The plots were visually congruent.

To directly compare the two models, T_{max} and T_{min} were used to calculate a percentage difference as plotted in Figure 13.

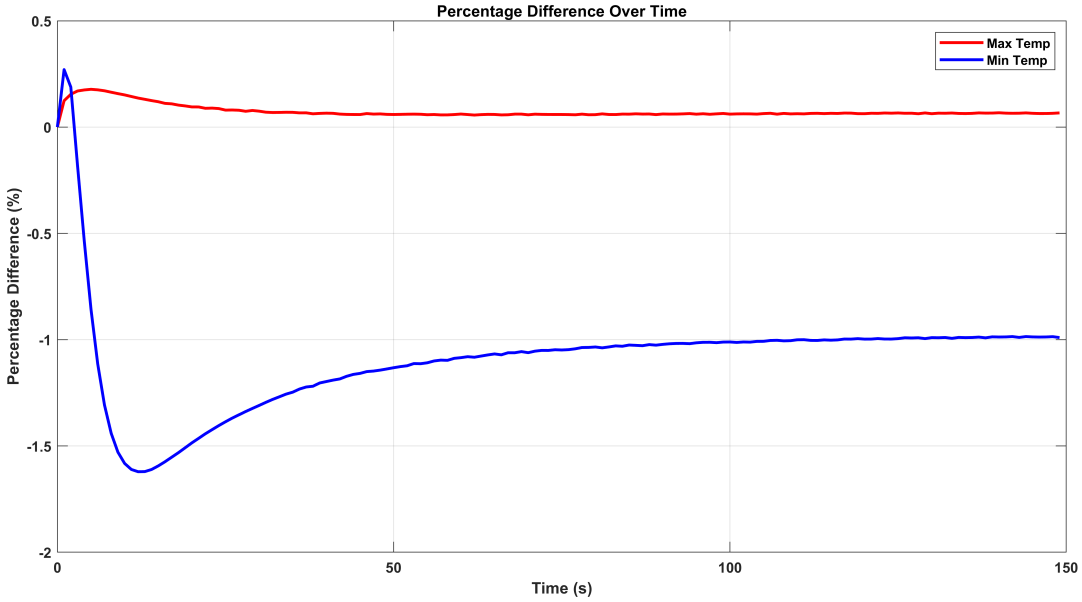


Figure 13: Plot showing the percentage difference between the ANSYS and MATLAB solutions throughout the transient response. The solution converged rapidly towards zero, indicating a high degree of accuracy.

In Figure 13 there were two key error features observed. Firstly, there was a significant discrepancy between the solutions in the time-steps following initialisation, particularly in T_{min}. The magnitude of this error was reduced with grid-convergence and time-step convergence. Much of this error was related to the inability to significantly refine the mesh. However, some of the error was likely solver specific, relating to the intricacies of the MATLAB and ANSYS solvers and how each handle numerical artifacts. MATLAB's mldivide (backlash) solver algorithmically selects between LU, Cholesky, Hessenberg, triangular, diagonal and LDL direct solvers and as such, was vulnerable to rounding errors.

Secondly, there was a small steady state residual error between the two solutions of approximately 0.1% in T_{max} and -1% in T_{min}. The discrepancy remained with time-step convergence and decreased gradually with grid-convergence. This error was attributed to the mesh structure and functionality within MATLAB's PDEToolbox for 3D meshing. To model the compound materials of the fin, the average z-coordinate of each element was used to determine in which region of the domain each element was located. This was imprecise - producing distinctive aberrations in the solution for coarser meshes as shown in Figure 14.

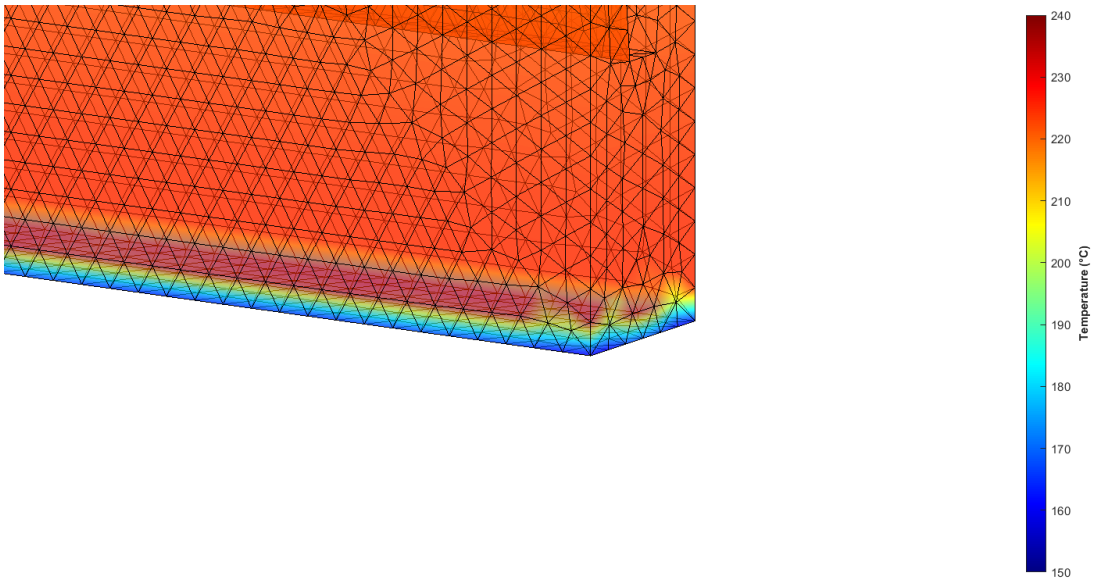


Figure 14: Plot showing steady state solution for a coarse mesh. The solution was affected by ambiguity in the CPU region; elements were often not oriented with a face aligned along the material boundary.

It was not possible to further reduce this error due to computational expense, despite using efficient memory allocations where possible. To alleviate this problem, a more comprehensive and flexible meshing solution would be needed. Overall, the error magnitude was deemed sufficiently small such that a high degree of confidence in the MATLAB accuracy and suitability was asserted.

The MATLAB solution accurately captured the expected behaviour, making it suitable for defining design parameters for a CPU heat sink. Whilst industry predominantly uses professional tools like ANSYS for understanding complex behaviors and crafting advanced products, MATLAB proved its efficacy in this context. However, commercial engineering software offers greater flexibility, user-friendliness and efficiency. To harness such tools effectively, engineers must grasp the underlying mechanics to build reliable real-world solutions.

References

[1] S. S. Rao, *The Finite Element Method in Engineering (Fourth Edition)*, ch. 13 - Formulation and Solution Procedure, 16 - Three-Dimensional Problems. 2005.

[2] R. L. de Orio, "Electromigration modeling and simulation." https://www.iue.tuwien.ac.at/phd/orio/node48.html. Accessed: 2023-11-01.

[3] H. Ceric, *Numerical Techniques in Modern TCAD*. Dissertation, Technische Universität Wien, Vienna, 2005.

5 Appendices

5.1 Appendix A: Derivation of the element equation

5.1.1 Linear tetrahedron [1]

$$N_1(\eta,\xi,\zeta) = 1-\eta-\xi-\zeta \tag{15}$$
$$N_2(\eta,\xi,\zeta) = \zeta \tag{16}$$
$$N_3(\eta,\xi,\zeta) = \xi \tag{17}$$
$$N_4(\eta,\xi,\zeta) = \eta \tag{18}$$

$$[\mathbf{P_m}] = \begin{bmatrix} 1 & x_{1i} & x_{2i} & x_{3i} \\ 1 & x_{1j} & x_{2j} & x_{3j} \\ 1 & x_{1k} & x_{2k} & x_{3k} \\ 1 & x_{1l} & x_{2l} & x_{3l} \end{bmatrix} \tag{19}$$

$$V = \frac{1}{6}\,det\,[\mathbf{P_m}] \tag{20}$$

$$[\mathbf{P_m}]^{-1} = \frac{1}{6V}\begin{bmatrix} a_1 & a_2 & a_3 & a_4 \\ b_1 & b_2 & b_3 & b_4 \\ c_1 & c_2 & c_3 & c_4 \\ d_1 & d_2 & d_3 & d_4 \end{bmatrix} \tag{21}$$

$$[\mathbf{B}] = \frac{1}{6V}\begin{bmatrix} b_1 & b_2 & b_3 & b_4 \\ c_1 & c_2 & c_3 & c_4 \\ d_1 & d_2 & d_3 & d_4 \end{bmatrix} \tag{22}$$

$$\int_\Omega N_i^a N_j^b N_k^c N_m^d \, d\Omega = \frac{a!b!c!d!6V}{(a+b+c+3)!} \tag{23}$$

5.1.2 Spatial Discretisation

$$\rho c_p \frac{\partial T}{\partial t} - \kappa\left(\frac{\partial^2 T}{\partial y^2} + \frac{\partial^2 T}{\partial y^2} + \frac{\partial^2 T}{\partial z^2}\right) - q_v\,(x,y,z) = \mathscr{R}\,(x,y,z) \tag{24}$$

$$\iiint_V \left(\rho c_p \frac{\partial T}{\partial t} - \kappa\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} + \frac{\partial^2 T}{\partial z^2}\right) - q_v\,(x,y,z)\right) N_i \;\; dV = 0 \quad \text{for } i = 1,2,3,4 \tag{25}$$

$$\iiint_V \left(\kappa\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} + \frac{\partial^2 T}{\partial z^2}\right) + q_v\,(x,y,z)\right) N_i \;\; dV$$
$$= \rho c_p \iiint_V \frac{\partial T}{\partial t} N_i \;\; dV \quad \text{for } i = 1,2,3,4 \tag{26}$$

$$\iiint_V \left(-\kappa\frac{\partial N_i}{\partial x}\frac{\partial T}{\partial x} - \kappa\frac{\partial N_i}{\partial y}\frac{\partial T}{\partial y} - \kappa\frac{\partial N_i}{\partial z}\frac{\partial T}{\partial z}\right) N_i \ dV + \iiint_V q_v N_i \ dV + \kappa \iint_S N_i \left[\frac{\partial T}{\partial x}l_x + \frac{\partial T}{\partial y}l_y + \frac{\partial T}{\partial z}l_z\right] \ ds = \rho c_p \iiint_V \frac{\partial T}{\partial t} N_i \ dV \quad \text{for } i = 1,2,3,4 \tag{27}$$

$$\iiint_V \left(-\kappa\frac{\partial N_i}{\partial x}\frac{\partial T}{\partial x} - \kappa\frac{\partial N_i}{\partial y}\frac{\partial T}{\partial y} - \kappa\frac{\partial N_i}{\partial z}\frac{\partial T}{\partial z}\right) N_i \ dV + \iiint_V q_v N_i \ dV + \kappa \iint_S N_i \frac{\partial T}{\partial n} \ ds = \rho c_p \iiint_V \frac{\partial T}{\partial t} N_i \ dV \quad \text{for } i = 1,2,3,4 \tag{28}$$

$$\kappa \iiint_V [\mathbf{B}]^T [\mathbf{B}] \{\mathbf{T}\} \ dV + \rho c_p \iiint_V [\mathbf{N}]^T [\mathbf{N}] \{\mathbf{T}'\} \ dV = \kappa \iint_S [\mathbf{N}]^T \frac{\partial T}{\partial n} \ ds + q_v \iiint_V [\mathbf{N}]^T \ dV \tag{29}$$

$$\kappa [\mathbf{B}]^T [\mathbf{B}] V \{\mathbf{T}\} + \frac{\rho c_p V}{20} \begin{bmatrix} 2 & 1 & 1 & 1 \\ 1 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 2 \end{bmatrix} \{\mathbf{T}'\} = \kappa \iint_S [\mathbf{N}]^T \frac{\partial T}{\partial n} \ ds + \frac{q_v V}{4} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \tag{30}$$

$$\kappa \iint_S [\mathbf{N}]^T \frac{\partial T}{\partial n} \ ds = -h \iint_S [\mathbf{N}]^T [\mathbf{N}] \{\mathbf{T}\} \ ds + T_\alpha h \iint_S [\mathbf{N}]^T \ ds = -\frac{h A_{jkl}}{12} \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix} \{\mathbf{T}\} + \frac{h A_{jkl} T_\alpha}{3} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \tag{31}$$

$$\frac{\rho c_p V}{20} \begin{bmatrix} 2 & 1 & 1 & 1 \\ 1 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 2 \end{bmatrix} \{\mathbf{T}'\} + \left(\kappa [\mathbf{B}]^T [\mathbf{B}] V + \frac{h A_{jkl}}{12} \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}\right) \{\mathbf{T}\} = \frac{q_v A_{jkl}}{4} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} + \frac{h A_{jkl} T_\alpha}{3} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \tag{32}$$

$$[\mathbf{c}] \{\mathbf{T}'\} + [\mathbf{k}] \{T\} = [\mathbf{f}] \tag{33}$$

5.1.3 Temporal Discretisation

$$\left.\frac{dT_i}{dt}\right|_{n+1} = \frac{T_i^{n+1} - T_i^n}{\Delta t} + \mathcal{O}(\Delta t) \quad \text{for } i = 1,2,3,4 \tag{34}$$

$$[\mathbf{c}]\left\{\mathbf{T^{n+1}}\right\} - [\mathbf{c}]\left\{\mathbf{T^{n}}\right\} + \Delta t\,[\mathbf{k}]\left\{\mathbf{T^{n+1}}\right\} = \Delta t\,[\mathbf{f}] \tag{35}$$

$$([\mathbf{c}] + \Delta t\,[\mathbf{k}])\left\{\mathbf{T^{n+1}}\right\} = \Delta t\,[\mathbf{f}] + [\mathbf{c}]\left\{\mathbf{T^{n}}\right\} \tag{36}$$

$$[\mathbf{M}]\left\{\mathbf{T^{n+1}}\right\} = [\mathbf{R}] \tag{37}$$

$$\left\{\mathbf{T^{n+1}}\right\} = [\mathbf{M}]^{-1}\,[\mathbf{R}] \tag{38}$$