Mathematical Model for Determining Flashover in a Compartment Fire

University of Waterloo

ME203 – Ordinary Differential Equations

Grace Dice

November 21, 2025

Contents

List of Tables

List of Figures

1 Introduction

This report analyzes a compartment fire to determine whether flashover will occur under steady burning. The model predicts the smoke-layer temperature over time and compares it to the McCaffrey-Quintiere-Harkleroad (MQH) flashover criterion. The goal is to determine whether the current room geometry and materials provide acceptable fire safety.

Flashover marks the point where the upper smoke layer heats all combustibles to ignition temperature. It transforms a small fire into a full-room involvement, threatening structure and life within seconds. Predicting this condition early allows the design to prevent it through ventilation, insulation, or suppression.

2 Compartment Description and Fire Scenario

The analysis models a 4 m × 4 m × 3.3 m compartment with one doorway (2 m × 1.4 m).

Table 1: Compartment Input Parameters

Parameter	Symbol	Value	Unit
Wall heat transfer coefficient	h_k	12	$\mathrm{W\,m^{-2}\,K^{-1}}$
Discharge coefficient	C_d	0.55	—
Ambient temperature	T_∞	27	°C
Air density	ρ_∞	1.2	$\mathrm{kg\,m^{-3}}$
Specific heat	c_p	1000	$\mathrm{J\,kg^{-1}\,K^{-1}}$

The fuel package, a 1.3 m × 1.4 m plastic surface, releases 494.5 $\mathrm{kW\,m^{-2}}$.

$$\dot{Q} = 1.3 \times 1.4 \times 494.5 = 900 \text{ kW}$$

This constant HRR defines a severe but credible design fire.

3 Physical Model and Governing Equations

3.1 Smoke Layer Dynamics

Hot gases rise in a buoyant plume, bringing in the air with it. When the plume hits the ceiling, it spreads, forming a smoke layer that thickens and descends as heat continues to enter. Mass leaves the layer through the doorway, and the balance between inflow and outflow governs the layer height H_s and density ρ_s.

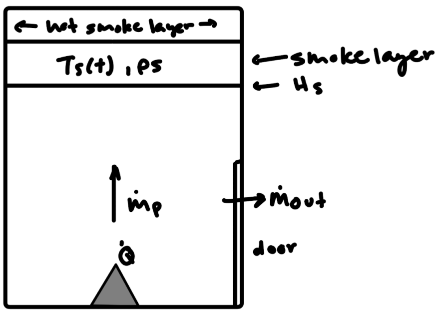

Figure 1: Compartment Cross-Sectional Drawing

3.2 Mass Balance

$$\frac{dm_s}{dt} = \dot{m}_p - \dot{m}_{out} \tag{1}$$

Plume entrainment:

$$\dot{m}_p = 0.071\,\dot{Q}^{1/3} H_s^{5/3} \tag{2}$$

Doorway outflow:

$$\dot{m}_{out} = \frac{2}{3} C_d W_o \sqrt{2g(\rho_\infty - \rho_s)}\,\rho_s^{1/2} (H_o - H_s)^{3/2} \tag{3}$$

Ideal gas relation:

$$\rho_s = \rho_\infty \frac{T_\infty}{T_s} \tag{4}$$

At steady state, inflow equals outflow. H_s depends on both T_s and the geometry of the doorway.

In simpler terms, the plume entrainment term $\dot{m}_p$ is the mass of air pulled into the plume and transported to the hot layer under the ceiling. It increases with the fire size ($\dot{Q}^{1/3}$) and with the plume path length ($H_s^{5/3}$): a taller distance to the layer entrains more air.

The outflow term $\dot{m}_{out}$ is the buoyancy-driven flow of hot, low-density gas out through the doorway. It increases with the doorway width W_o, with the buoyancy (difference between ambient and hot layer), and with the available opening height above the smoke layer ($H_o - H_s$).

The ideal gas link ties the hot layer density ρ_s to its temperature T_s. Another layer means a smaller ρ_s, greater buoyancy, and then a larger outflow for the same opening geometry.

At steady state, entrainment and outflow are equal. The key unknown is the smoke layer interface height (H_s) or, equivalently, ($H_T - H_s$). Since outflow depends on the layer temperature, H_s and T_s are coupled. The mass balance alone cannot determine both, and they may be obtained together by solving both the mass and energy balances.

3.3 Energy Balance

The energy balance for the upper smoke layer is derived from the First Law of Thermodynamics applied to an open control volume. The upper layer stores energy, gains heat from the fire, and loses energy through hot gases leaving the compartment and through heat transfer to the walls, ceiling, and floor.

Rate of energy in − Rate of energy out = Rate of energy storage

For the control volume defined by the upper smoke layer:

$$m_s c_p \frac{dT_s}{dt} = \dot{Q} \cdot 1000 - \dot{m}_{out} c_p (T_s - T_\infty) - h_k A_T (T_s - T_\infty)$$

where:

- $\dot{Q}$ is the heat release rate of the fire (kW),
- m_s is the mass of the smoke layer (kg),
- c_p is the specific heat of the gas (kJ/kg·K),
- h_k is the overall heat transfer coefficient to the compartment surfaces (W/m^2·K),
- A_T is the total internal surface area (82 m^2),
- T_s is the temperature of the smoke layer (K),
- and T_∞ is the ambient temperature of the incoming air (K).

Assumptions:

(a) The lower layer and incoming air remain at the ambient temperature, T_∞.

(b) The properties of air (c_p, ρ_s) are constant.

(c) The heat release rate, $\dot{Q}$, is constant during the scenario.

(d) The system is well-mixed, such that the upper layer has a uniform temperature T_s.

The first term on the right-hand side, $\dot{Q} \cdot 1000$, represents the rate of heat addition from the fire. The second term, $\dot{m}_{out} c_p (T_s - T_\infty)$, accounts for the energy loss as hot gases exit the doorway and cooler air enters at T_∞. The final term, $h_k A_T (T_s - T_\infty)$, represents the combined losses to the surrounding surfaces.
Defining $\theta = T_s - T_\infty$ and combining all loss terms into a single coefficient:

$$K = \dot{m}_{out} c_p + h_k A_T$$

gives the simplified form:

$$m_s c_p \frac{d\theta}{dt} = \dot{Q} \cdot 1000 - K\theta$$

At steady state ($\frac{d\theta}{dt} = 0$), the heat input equals total losses:

$$\dot{Q} \cdot 1000 = K\theta_\infty \quad \Rightarrow \quad \theta_\infty = \frac{\dot{Q} \cdot 1000}{K}$$

The transient form of this first-order linear ODE can be expressed as:

$$\frac{d\theta}{dt} + \frac{\theta}{\tau} = \frac{\theta_\infty}{\tau} \quad \text{where} \quad \tau = \frac{m_s c_p}{K}$$

The solution is:

$$T_s(t) = T_\infty + \theta_\infty \left(1 - e^{-t/\tau}\right)$$

Here, τ is the time constant representing the thermal response of the smoke layer. Physically, a large τ indicates that the system has a high thermal inertia (large mass or heat capacity) and heats up more slowly, while a small τ corresponds to a faster temperature rise. The assumption that the incoming air temperature equals T_∞ allows the inflow term to depend only on the temperature difference $(T_s - T_\infty)$, simplifying the co

4 Numerical Implementation and Results

The coupled mass and energy balances require iterative solution. Starting with $H_s = H_o/2$ and $T_s = T_\infty + 300$ K, five iterations were performed. Convergence was rapid, as shown below.

Table 2: Convergence of Key Parameters Over 5 Iterations

Iteration	H_s (m)	$T_{s,\infty}$ (°C)	ρ_s (kg/m^3)	m_s (kg)	$\dot{m}_{out}$ (kg/s)
1	1.215	492.70	0.600	20.02	0.949
2	1.208	495.00	0.470	15.74	0.939
3	1.208	495.05	0.469	15.70	0.939
4	1.208	495.05	0.469	15.70	0.939
5	1.208	495.05	0.469	15.70	0.939

Table 3: Final Converged Values (Iteration 5)

Parameter	**Value**
H_s	1.208 m
$T_{s,\infty}$	495.05 °C (768.20 K)
ρ_s	0.469 kg/m^3
m_s	15.70 kg
$\dot{m}_{out}$	0.939 kg/s
K	1922.87 W/K
τ	8.2 s
θ_∞	468.05 K

The system converges within 3 iterations. The final smoke layer interface is at $H_s = 1.208$ m, and the steady-state temperature is $T_{s,\infty} = 495.1$°C.

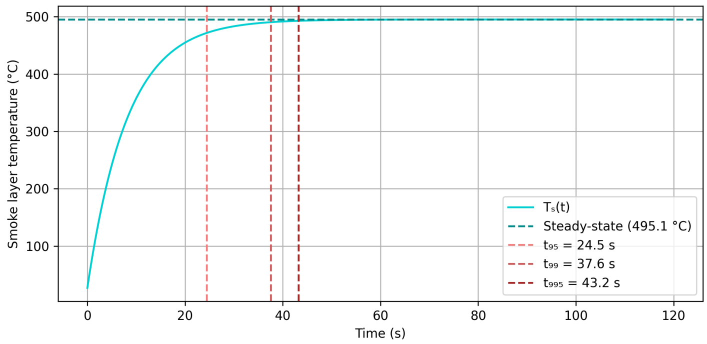

Figure 2: Smoke Layer Temperature Evolution and Steady-State Times

Time to reach:

- 95% of steady state: $t_{95} = 24.5$ s
- 99%: $t_{99} = 37.6$ s
- 99.5%: $t_{99.5} = 43.2$ s

Using the MQH correlation:

$$\Delta T_{FO,MQH} = 6.85 \left(\frac{\dot{Q}^2}{A_o H_o^{1/2} h_k A_T} \right)^{1/3} = 405.8 \text{ K} \tag{5}$$

$$t_{FO} = -\tau \ln \left(1 - \frac{\Delta T_{FO,MQH}}{\theta_\infty} \right) = 16.5 \text{ s} \tag{6}$$

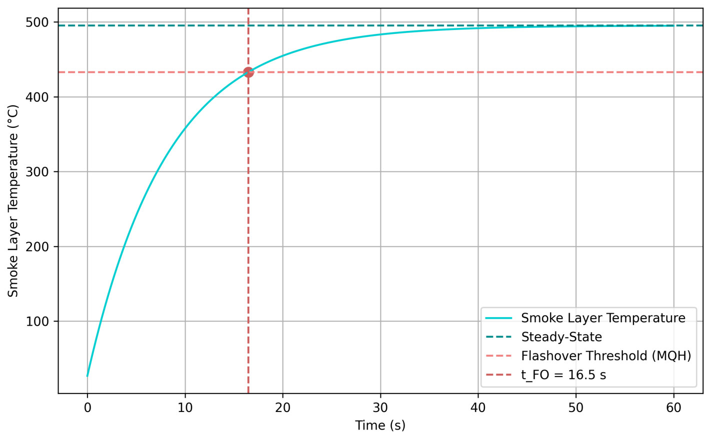

Figure 3: Flashover Criterion Using MQH Model

The model predicts flashover at $t = 16.5$ s when the smoke layer reaches $T_{FO} = 432.8°\text{C}$. This is extremely rapid and indicates high risk.

5 Analysis

5.1 Temperature Evolution

The smoke layer heats quickly because of the large HRR and limited venting. Most temperature rise occurs in the first few seconds; then the curve flattens as well, and outflow losses balance the input. The exponential form captures this physical slowdown.

5.2 Influence of Parameters

Increasing h_k increases heat losses through the walls and ceiling. This lowers the steady-state temperature rise θ_∞ and shortens the time constant τ, since the layer cools more efficiently. This explains why concrete or gypsum delays flashover compared to insulated or wood-lined surfaces that promote it.

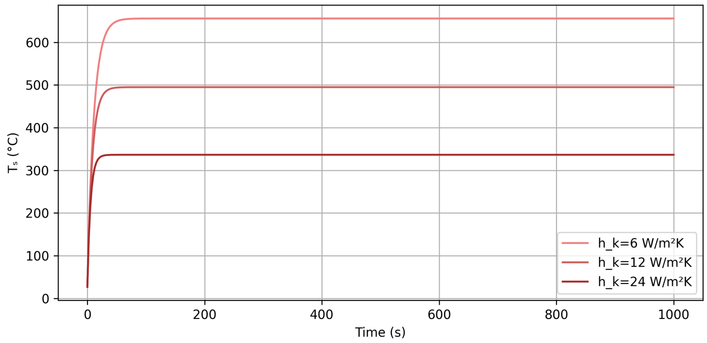

Figure 4: Effect of Wall Material (h_k) on Smoke Temperature

A larger doorway opening increases the buoyancy-driven outflow, and more gas escapes in consequence. More hot gas leaving lowers the temperature of the smoke layer and slows its rise. Narrower openings act the opposite and restrict flow and cause heat to build up in the compartment.

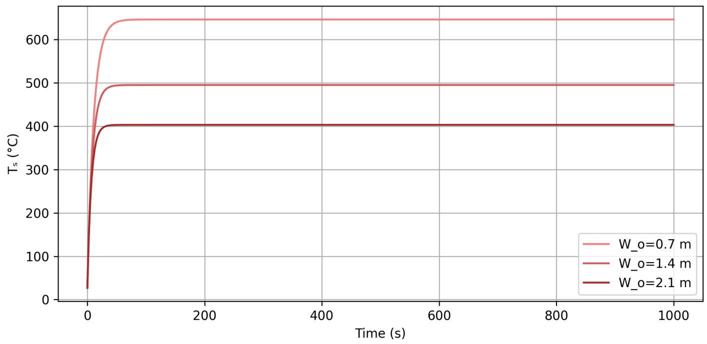

Figure 5: Effect of Doorway Width on Smoke Temperature

Increasing the height of the compartment increases the total air mass above the fire. The same heat release is distributed over a larger volume, so the temperature rises more slowly. Taller ceilings also delay the smoke layer from reaching any occupants and give a longer plume path.

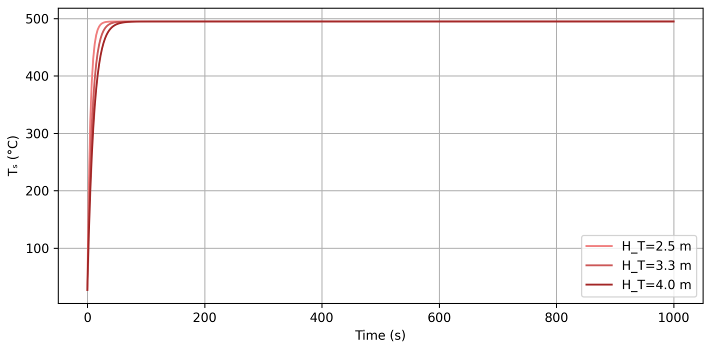

Figure 6: Effect of Compartment Height on Smoke Temperature

5.3 Physical Meaning of Parameters

θ_∞ (steady-state temperature rise):

The asymptotic temperature rise of the smoke layer above ambient if the fire and the loss mechanisms remain constant long enough. It is the balance of input power $\dot{Q}$ and the total loss conductance K. If θ_∞ is large enough to exceed the flashover temperature rise, then flashover will occur eventually.

τ (time constant):

The characteristic time for the layer to approach steady state. After time $t = \tau$, the layer will have reached about 63% of its eventual temp rise θ_∞. Numerically, a few τ (3–5τ) is the ratio of thermal inertia (mass times heat capacity of the layer) to the loss rate K.

$$\theta(\tau) = \theta_\infty(1 - e^{-\tau/\tau}) = \theta_\infty(1 - e^{-1})$$

$$e^{-1} \approx 0.3679 \quad \Rightarrow \quad 1 - 0.3679 = 0.6321$$

$$\boxed{\theta(\tau) \approx 0.632 \cdot \theta_\infty \quad \Rightarrow \quad 63\% \text{ of final rise}}$$

Table 4: Percentage of final temperature rise θ_∞ reached after $n\tau$.

Time	**% of θ_∞**	**Formula**
$t = \tau$	63%	$1 - e^{-1}$
$t = 2\tau$	86%	$1 - e^{-2}$
$t = 3\tau$	95%	$1 - e^{-3}$
$t = 4\tau$	98%	$1 - e^{-4}$
$t = 5\tau$	99.3%	$1 - e^{-5}$

6 Limitations

A key limitation is the constant heat release rate (HRR) assumption. In real fires, HRR varies over time as it grows, peaks, and decays with changing fuel availability and ventilation. The plume entrainment and smoke layer formation are therefore not constant and depend on transient combustion dynamics, oxygen availability, and fuel surface regression. This assumption can overpredict or underpredict the smoke temperature [1]:

- Overpredict if the real HRR drops early.
- Underpredict if the fire grows rapidly before the steady state assumption is reached.

The model also does not capture localized effects, such as flame impingement on the ceiling, radiation feedback to surfaces, or any non-uniform gas layers.

7 Recommendations and Conclusion

7.1 Design Recommendations

Since flashover is highly probable in this configuration, reducing risk can be done by:

1. Increase passive heat-loss surfaces (raise thermal inertia). Use non combustible and high heat-capacity linings such as concrete, fire-rated gypsum, or plaster. These all will increase h_k, absorb more heat, and reduce the rise in smoke temperature [2], [3].
2. Increase ventilation area to promote more smoke discharge. A wider doorway or a vent increases outflow and lowers the steady-state gas temperature. Solutions include:
   (a) widening the doorway,

(b) adding a high-level vent or transom window,

(c) introducing a mechanical exhaust system if architectural changes are limited.

3. Raise ceiling height to increase upper-layer volume. The same amount of heat would be distributed through a larger mass of air, reducing the temperature growth rate and delaying the smoke layer descension towards any occupants.

7.2 Overall Conclusion

Under a 900 kW fire, the compartment reached flashover in 16.5 s. The current geometry offers inadequate safety margins. Design revisions are required to prevent flashover and maintain structural integrity.

References

[1] W. W. Jones, R. D. Peacock, G. P. Forney, and P. A. Reneke, "Verification and Validation of CFAST, A Model of Fire Growth and Smoke Spread," en,

[2] H. Zhou, A. R. Puttige, G. Nair, and T. Olofsson, "Thermal behaviour of a gypsum board incorporated with phase change materials," *Journal of Building Engineering*, vol. 94, p. 109 928, Oct. 2024, ISSN: 2352-7102. DOI: 10.1016/j.jobe.2024.109928 Accessed: Nov. 15, 2025. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352710224014967

[3] P. Shafigh, I. Asadi, and N. Mahyuddin, "Concrete as a thermal mass material for building applications - A review — Request PDF," en, *ResearchGate*, vol. 19, Apr. 2018. DOI: 10.1016/j.jobe.2018.04.021 Accessed: Nov. 15, 2025. [Online]. Available: https://www.researchgate.net/publication/324763924_Concrete_as_a_thermal_mass_material_for_building_applications_-_A_review

A Numerical Implementation Code

A.1 Python Code for Compartment Fire Model

```
import numpy as np
from scipy.optimize import minimize_scalar
import matplotlib.pyplot as plt

# Parameters
L = 4.0
d = 4.0
H_T = 3.3
H_o = 2.0
W_o = 1.4
Cd = 0.55
h_k = 12.0
A_T = 84.8 - 2.8  # total surface area including floor + ceiling
Q_dot = 900        # kW
T_inf = 27 + 273.15  # K
rho_inf = 1.2
cp = 1000.0
g = 9.81

# Initial guesses
H_s_guess = H_o / 2
T_s_inf = T_inf + 300  (K)

# Functions
def plume_mdot(Q_kW, Hs):
    """Plume entrainment mass flow (kg/s)"""
    return 0.071 * (Q_kW ** (1/3)) * (Hs ** (5/3))

def door_mdot(Hs, T_s):
    """Smoke outflow through doorway (kg/s)"""
    rho_s = rho_inf * T_inf / T_s
    delta_rho = rho_inf - rho_s
    if delta_rho <= 0:
        return 0.0
    return (2/3) * Cd * W_o * np.sqrt(2*g*delta_rho) * np.sqrt(rho_s) *
        ((H_o - Hs) ** 1.5)

def mass_balance(Hs, T_s):
    """Mass balance: m_plume - m_out"""
    return plume_mdot(Q_dot, Hs) - door_mdot(Hs, T_s)

# Iterative Solution -5 iterations
max_iterations = 5
H_s_values = []
T_s_inf_values = [T_s_inf]
tau_values = []
theta_inf_values = []

print("Starting 5 iterations to converge on H_s and T_s,     ...\n")

for i in range(1, max_iterations + 1):

    result = minimize_scalar(
```

```
            lambda Hs: abs(mass_balance(Hs, T_s_inf_values[-1])),
            bounds=(0.1, H_o - 0.01),
            method='bounded'
        )
        H_s_new = result.x
        H_s_values.append(H_s_new)


        rho_s = rho_inf * T_inf / T_s_inf_values[-1]
        m_s = rho_s * L * d * (H_T - H_s_new)
        m_out = door_mdot(H_s_new, T_s_inf_values[-1])


        K = m_out * cp + h_k * A_T
        tau = (m_s * cp) / K
        theta_inf = (Q_dot * 1000) / K  # Q in Watts


        T_s_inf_new = T_inf + theta_inf
        T_s_inf_values.append(T_s_inf_new)
        tau_values.append(tau)
        theta_inf_values.append(theta_inf)


        print(f"Iteration i = {i} results:")
        print(f"  H_s = {H_s_new:.3f} m")
        print(f"  T_s,  = {T_s_inf_new - 273.15:.2f}  C  ({T_s_inf_new:.2f} 
            K)")
        print(f"  rho_s = {rho_s:.3f} kg/m  ")
        print(f"  m_s = {m_s:.2f} kg")
        print(f"  m_out = {m_out:.3f} kg/s")
        print(f"  K = {K:.2f} W/K")
        print(f"  tau = {tau:.1f} s")
        print(f"   _inf  = {theta_inf:.2f} K\n")


        t = np.linspace(0, 1000, 300)
        T_s = T_inf + theta_inf * (1 - np.exp(-t / tau))
        plt.figure(figsize=(7, 4))
        plt.plot(t, T_s - 273.15, color='darkturquoise')
        plt.axhline(T_s_inf_new - 273.15, color='darkcyan', linestyle='--',
                    label=f"Steady-state  T    = {T_s_inf_new - 273.15:.1f} 
                        C ")
        plt.xlabel("Time (s)")
        plt.ylabel("Smoke layer temperature ( C )")
        plt.title(f"Iteration {i}")
        plt.grid(True)
        plt.legend()
        plt.tight_layout()
        plt.savefig(f"smoke_temp_iteration{i}.png", dpi=300)
        plt.close()

# Use final values from last iteration
H_s_final = H_s_values[-1]
T_s_inf_final = T_s_inf_values[-1]
tau_final = tau_values[-1]
theta_inf_final = theta_inf_values[-1]
rho_s_final = rho_inf * T_inf / T_s_inf_values[-2]
```

```
m_s_final = rho_s_final * L * d * (H_T - H_s_final)
m_out_final = door_mdot(H_s_final, T_s_inf_values[-2])
K_final = m_out_final * cp + h_k * A_T

print(f"\nConvergence after {max_iterations} iterations:")
print(f"Final H_s = {H_s_final:.3f} m")
print(f"Final T_s,  = {T_s_inf_final - 273.15:.2f}  C ")
print(f"Final   = {tau_final:.1f} s")

# Steady-State Analysis
T_inf_C = 27.0
t = np.linspace(0, 120, 600)
T_s_K = T_inf + theta_inf_final * (1 - np.exp(-t / tau_final))
T_s_C = T_s_K - 273.15

t_95 = -tau_final * np.log(1 - 0.95)
t_99 = -tau_final * np.log(1 - 0.99)
t_995 = -tau_final * np.log(1 - 0.995)

print("\n--- Steady-State Analysis ---")
print(f"Steady-state  T  ,  = {T_s_inf_final - 273.15:.1f}  C ")
print(f"Time to 95%:   t   = {t_95:.1f} s")
print(f"Time to 99%:   t   = {t_99:.1f} s")
print(f"Time to 99.5%:   t   = {t_995:.1f} s")

plt.figure(figsize=(8, 4))
plt.plot(t, T_s_C, label=" T  (t)", color='darkturquoise')
plt.axhline(T_s_inf_final - 273.15, color='darkcyan', linestyle='--',
            label=f"Steady-state ({T_s_inf_final - 273.15:.1f}  C )")
plt.axvline(t_95, color='lightcoral', linestyle='--', label=f"  t   =
    {t_95:.1f} s")
plt.axvline(t_99, color='indianred', linestyle='--', label=f"  t   =
    {t_99:.1f} s")
plt.axvline(t_995, color='brown', linestyle='--', label=f"  t   =
    {t_995:.1f} s")
plt.xlabel("Time (s)")
plt.ylabel("Smoke layer temperature ( C )")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.savefig("steady_state_analysis.png", dpi=300)
plt.show()
plt.close()

#  Sensitivity Analysis
def run_case(h_k_val, W_o_val, H_T_val, T_s_guess):

    result = minimize_scalar(
        lambda Hs: abs(mass_balance(Hs, T_s_guess)),
        bounds=(0.1, H_o - 0.01), method='bounded'
    )
    Hs = result.x
    rho_s = rho_inf * T_inf / T_s_guess
    m_s = rho_s * L * d * (H_T_val - Hs)
    m_out = (2/3) * Cd * W_o_val * np.sqrt(2*g*(rho_inf -
        rho_s))*np.sqrt(rho_s) * ((H_o - Hs)**1.5)
    K = m_out * cp + h_k_val * A_T
    tau_out = (m_s * cp) / K
```

```
    theta_inf_out = (Q_dot * 1000) / K
    return tau_out, theta_inf_out

t_plot = np.linspace(0, 1000, 400)
colors = ['lightcoral', 'indianred', 'brown']

# i. Wall conductivity
plt.figure(figsize=(8, 4))
for idx, hk in enumerate([6, 12, 24]):
    tau_sens, theta_sens = run_case(hk, W_o, H_T, T_s_inf_values[-2])
    T = T_inf + theta_sens * (1 - np.exp(-t_plot / tau_sens))
    plt.plot(t_plot, T - 273.15, color=colors[idx], label=f"h_k = {hk}
       W/m K")
plt.xlabel("Time (s)")
plt.ylabel(" T  ( C )")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.savefig("effect_wall_material.png", dpi=300)
plt.show()
plt.close()

# ii. Doorway width
plt.figure(figsize=(8, 4))
for idx, Wo in enumerate([0.7, 1.4, 2.1]):
    tau_sens, theta_sens = run_case(h_k, Wo, H_T, T_s_inf_values[-2])
    T = T_inf + theta_sens * (1 - np.exp(-t_plot / tau_sens))
    plt.plot(t_plot, T - 273.15, color=colors[idx], label=f"W_o = {Wo} m")
plt.xlabel("Time (s)")
plt.ylabel(" T  ( C )")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.savefig("effect_doorway_width.png", dpi=300)
plt.show()
plt.close()

# iii. Compartment height
plt.figure(figsize=(8, 4))
for idx, HT in enumerate([2.5, 3.3, 4.0]):
    tau_sens, theta_sens = run_case(h_k, W_o, HT, T_s_inf_values[-2])
    T = T_inf + theta_sens * (1 - np.exp(-t_plot / tau_sens))
    plt.plot(t_plot, T - 273.15, color=colors[idx], label=f"H_T = {HT} m")
plt.xlabel("Time (s)")
plt.ylabel(" T  ( C )")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.savefig("effect_compartment_height.png", dpi=300)
plt.show()
plt.close()

# --- Flashover Criterion ---
DeltaT_FO_MQH = 405.78519881836553  # K rise
t_fo = np.linspace(0, 60, 400)
theta_t = theta_inf_final * (1 - np.exp(-t_fo / tau_final))
T_s_fo = T_inf + theta_t
```

```
t_FO = None
if DeltaT_FO_MQH < theta_inf_final:
    t_FO = -tau_final * np.log(1 - DeltaT_FO_MQH / theta_inf_final)
    T_FO = T_inf + DeltaT_FO_MQH

plt.figure(figsize=(8, 5))
plt.plot(t_fo, T_s_fo - 273.15, label="Smoke Layer Temperature",
    color='darkturquoise')
plt.axhline(T_s_inf_final - 273.15, color='darkcyan', linestyle='--',
    label="Steady-State")
plt.axhline(T_inf + DeltaT_FO_MQH - 273.15, color='lightcoral',
    linestyle='--',
            label="Flashover Threshold (MQH)")
if t_FO and t_FO <= 60:
    plt.axvline(t_FO, color='indianred', linestyle='--', label=f"t_FO =
        {t_FO:.1f} s")
    plt.scatter(t_FO, T_FO - 273.15, color='indianred', s=60)
plt.xlabel("Time (s)")
plt.ylabel("Smoke Layer Temperature ( C )")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.savefig("flashover_criterion.png", dpi=300)
plt.show()
plt.close()

print(f"\nT_FO_MQH = {DeltaT_FO_MQH:.1f} K")
print(f"Final  _inf  = {theta_inf_final:.1f} K")
if t_FO and t_FO <= 60:
    print(f"Flashover at t = {t_FO:.1f} s, T = {T_FO - 273.15:.1f}  C ")
else:
    print("No flashover within 60 s (or  _inf  <  T_FO ).")
```