Implementing the SP_3-Approximation for Radiation Heat Transfer in OpenFOAM

Andrea Correa

Luleå University of Technology,
Luleå, Sweden

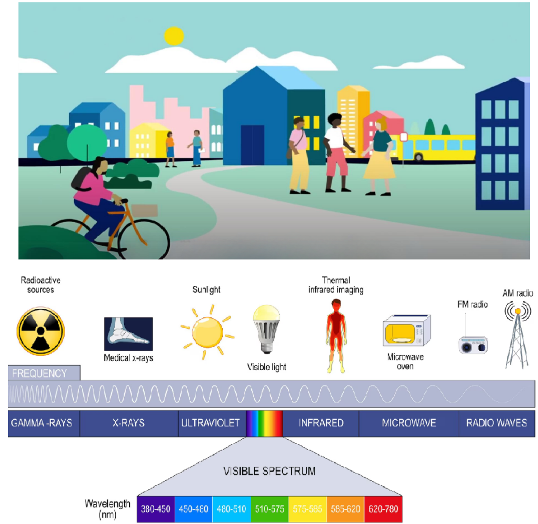
January 21, 2025

LULEÅ
UNIVERSITY
OF TECHNOLOGY

Agenda

Radiation

- Thermal radiation is the emission of electromagnetic waves by matter due to its temperature.
- Energy transfer without a medium.
- Spectral characteristics: Covers infrared, visible, and ultraviolet wavelengths.
- Radiation interacts with surfaces and materials in three ways: absorption, reflection, and transmission.



Radiation - electromagnetic spectrum

Introduction

Applications in fire safety

- Radiation transfers heat from flames to nearby objects.
- Firefighter protective equipment minimizes risks from thermal radiation.
- Radiation accelerates fire spread in enclosed spaces.
- **Hydrogen flames emit less visible radiation, posing unique detection and safety challenges.**

Applications in fire safety

- Is radiation dangerous?
- How much intensity?
- Is it possible to measure?

Introduction

Applications in fire safety

Radiative Transfer Equation (RTE)

- Radiative intensity, denoted as I_η, is the amount of radiant energy traveling in a specific direction.

$$\frac{dI_\eta}{ds} = \hat{s} \cdot \nabla I_\eta = \kappa_\eta I_{\eta b} - \beta_\eta I_\eta + \frac{\sigma_{s\eta}}{4\pi} \int_{4\pi} I_\eta(\hat{s}_i) \Phi_\eta(\hat{s}_i, \hat{s}) d\Omega_i,$$

- It can be determined experimentally using radiometers or spectroradiometers and computationally by solving RTE through methods such as Discrete Ordinates (DOM), Monte Carlo, or Spherical Harmonics.

Introduction

Radiative Transfer Equation (RTE)

$$\frac{dI_\eta}{ds} = \hat{s} \cdot \nabla I_\eta = \kappa_\eta I_{\eta b} - \beta_\eta I_\eta + \frac{\sigma_{s\eta}}{4\pi} \int_{4\pi} I_\eta(\hat{s}_i) \Phi_\eta(\hat{s}_i, \hat{s}) d\Omega_i,$$

Numerical Solutions

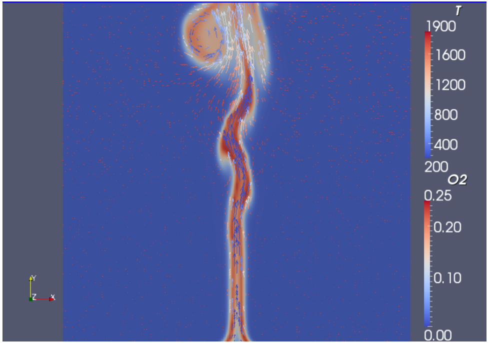
- **Discrete Ordinates Method (DOM):** Discretizes the angular domain into finite directions.
- **Spherical Harmonics Approximations (P_N Models):** Expands radiative intensity into spherical harmonics for approximate solutions.
- **Monte Carlo Method:** Uses stochastic techniques to simulate photon paths and interactions.
- **Finite Volume Method (FVM):** Solves the RTE numerically by dividing the domain into finite volumes.
- **Ray Tracing:** Tracks individual rays to compute radiative intensity along paths.
- **Rosseland Mean Approximation:** Uses average properties weighted by the temperature gradient for optically thick media.

Radiative Transfer Equation (RTE)

$$\frac{dI_\eta}{ds} = \hat{s} \cdot \nabla I_\eta = \kappa_\eta I_{\eta b} - \beta_\eta I_\eta + \frac{\sigma_{s\eta}}{4\pi} \int_{4\pi} I_\eta(\hat{s}_i) \Phi_\eta(\hat{s}_i, \hat{s}) d\Omega_i,$$

Radiation models in OpenFOAM

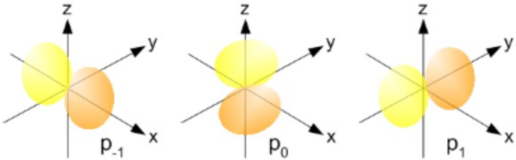
- fvDOM
- noRadiation
- opaqueSolid
- P1
- solarLoad
- viewFactor
- laserDTRM



Introduction

The method of spherical harmonics

- Simplifies radiative transfer equations into solvable partial differential equations.
- Higher-order approximations improve accuracy in anisotropic or optically thin media.



Source: Magula, A., Models of Nuclear Orbitals.

Radiation model in OpenFOAM

P1-Model: is the simplest form of the spherical harmonics method, where the radiative intensity is expanded up to the first term in the spherical harmonics series.

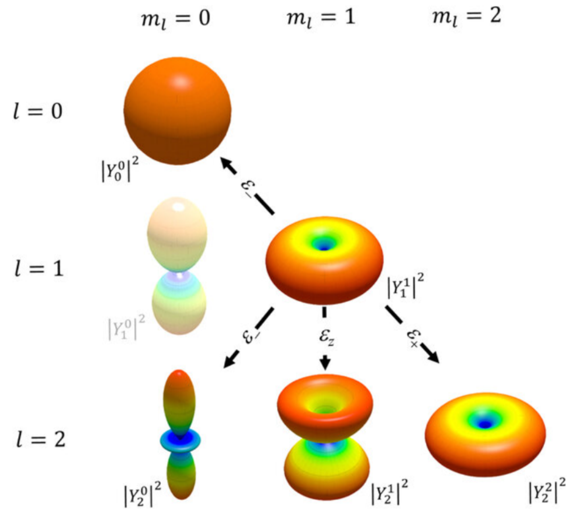
Implementation

SP3-Model: extends the P1 method by including higher-order terms, specifically the third-order term, which improves accuracy in capturing angular variations.

Spherical Harmonic Approximation - P_N Models

The radiative intensity $I(\mathbf{r}, \hat{s})$ at position $\mathbf{r}$ in the direction $\hat{s}$ can be expanded in terms of spherical harmonics $Y_l^m(\hat{s})$ as follows

$$I(\mathbf{r}, \hat{s}) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} I_l^m(\mathbf{r}) Y_l^m(\hat{s}),$$



Source: Thini,"Photo-ionization of polarized lithium atoms out of an all-optical atom trap: A complete experiment."

Spherical Harmonic Approximation - P_N Models

The radiative intensity $I(\mathbf{r}, \hat{s})$ at position $\mathbf{r}$ in the direction $\hat{s}$ can be expanded in terms of spherical harmonics $Y_l^m(\hat{s})$ as follows

$$I(\mathbf{r}, \hat{s}) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} I_l^m(\mathbf{r}) Y_l^m(\hat{s}),$$

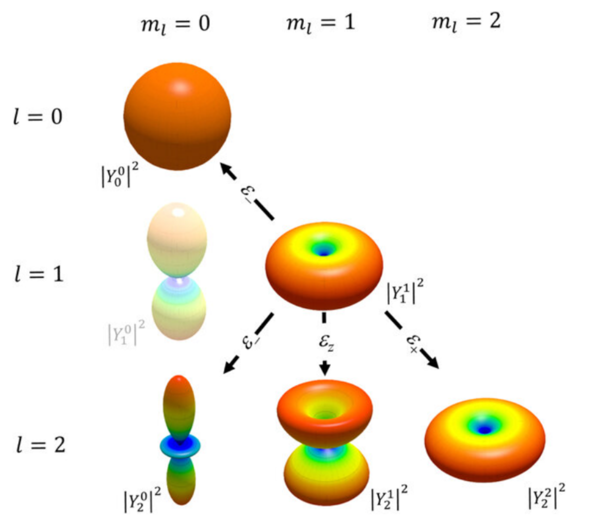
where

$$Y_l^m(\hat{s}) = \begin{cases} \cos(m\psi) P_l^m(\cos\theta), & \text{for } m \geq 0, \\ \sin(m\psi) P_l^m(\cos\theta), & \text{for } m < 0, \end{cases}$$

and P_l^m are the associated Legendre polynomials

$$P_l^m(\mu) = (-1)^l \frac{(1-\mu^2)^{|l|/2}}{2^l l!} \frac{d^{l+|l|}}{d\mu^{l+|l|}} \left(\mu^2 - 1\right)^l$$

l degree of angular momentum, $m$describes the azimuthal orientation, $-l < m < l$.



Source: Thini,"Photo-ionization of polarized lithium atoms out of an all-optical atom trap: A complete experiment."

Spherical Harmonic Approximation - P$_N$ Models

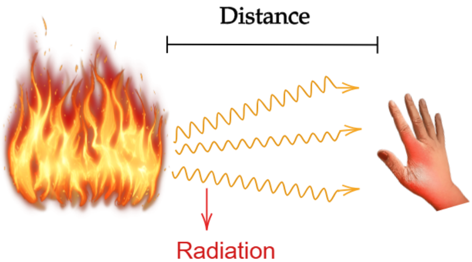
Radiative Transfer Equation

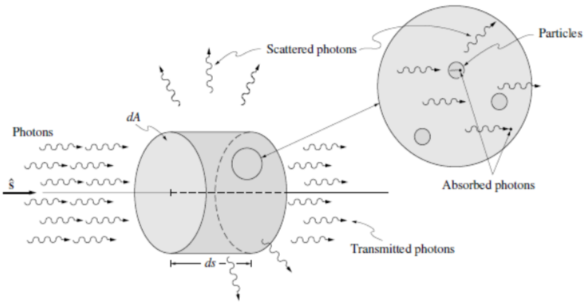
$$\frac{dI_\eta}{ds} = \kappa_\eta I_{\eta b} - \beta_\eta I_\eta + \frac{\sigma_{s\eta}}{4\pi}\int_{4\pi} I_\eta(\hat{s}_i)\Phi_\eta(\hat{s}_i,\hat{s})d\Omega_i,$$

Key Parameters of the RTE

- I_η: Spectral radiative intensity.
- κ_η: Spectral absorption coefficient.
- β_η: Extinction coefficient, defined as $\beta_\eta = \kappa_\eta + \sigma_{s\eta}$.
- $\sigma_{s\eta}$: Spectral scattering coefficient.

.





Source: Modest M., Radiative heat transfer

Spherical Harmonic Approximation - P_N Models

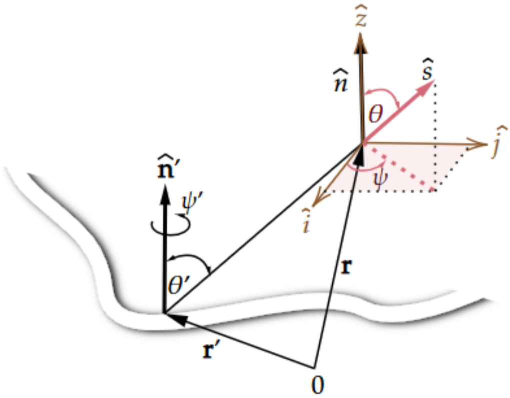
Radiative Transfer Equation

$$\frac{dI_\eta}{ds} = \kappa_\eta I_{\eta b} - \beta_\eta I_\eta + \frac{\sigma_{s\eta}}{4\pi}\int_{4\pi} I_\eta(\hat{s}_i)\Phi_\eta(\hat{s}_i, \hat{s})d\Omega_i,$$

Key Parameters of the RTE

- $d\Omega$: Solid angle
- $\hat{s}$: Unit vector in a given direction.
- $\Phi_\eta(\hat{s}_i, \hat{s})$: Scattering phase function
- θ: Polar angle
- ψ: Azimuthal angle

.



Inspired in the book "Radiative Heat Transfer" by Modest

P_1 and SP_3 - Approximations

P_1 and SP_3 - Approximations

P_1 - Approximations

The $I(\mathbf{r}, \hat{s})$ is truncated beyond $l = 1$ with $m = \pm 1\,, 0$

$$I(\mathbf{r}, \hat{s}) = I_0^0 + I_1^0 \cos\theta - I_1^1 \sin\theta \cos\psi + I_1^{-1} \sin\theta \sin\psi$$

P_1 and SP_3 - Approximations

The $I(\mathbf{r}, \hat{s})$ is truncated beyond $l = 1$ with $m = \pm 1\,, 0$, and the direction vector
$\hat{\mathbf{s}} = \sin\theta\cos\psi\hat{i} + \sin\theta\sin\psi\hat{j} + \cos\theta\hat{k}$

$$I(\mathbf{r}, \hat{s}) = \underbrace{I_0^0}_{\substack{l=0 \\ \downarrow \\ \text{Zeroth moment}}} + \underbrace{I_1^0\cos\theta - I_1^1\sin\theta\cos\psi + I_1^{-1}\sin\theta\sin\psi}_{\substack{l=1 \\ \downarrow \\ \text{First moment}}}.$$

The $I(\mathbf{r}, \hat{s})$ is truncated beyond $l = 1$ with $m = \pm 1, 0$, and the direction vector $\hat{\mathbf{s}} = \sin\theta\cos\psi\hat{i} + \sin\theta\sin\psi\hat{j} + \cos\theta\hat{k}$

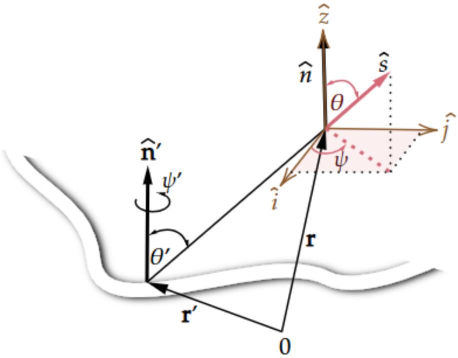
$$I(\mathbf{r}, \hat{s}) = \underbrace{I_0^0}_{\substack{\text{l=0} \\ \downarrow \\ \text{Zeroth moment}}} + \underbrace{I_1^0\cos\theta - I_1^1\sin\theta\cos\psi + I_1^{-1}\sin\theta\sin\psi}_{\substack{\text{l=1} \\ \downarrow \\ \text{First moment}}}.$$

The incident radiation G_η and radiative heat flux $\mathbf{q}$ are defined as

$$G_\eta(\mathbf{r}) = \int_{4\pi} I(\mathbf{r}, \hat{\mathbf{s}})\, d\Omega, \quad \mathbf{q} = \int_{4\pi} I(\mathbf{r}, \hat{\mathbf{s}})\hat{\mathbf{s}}\, d\Omega,$$

The intensity can be expressed

$$I(\mathbf{r}, \hat{\mathbf{s}}) = \frac{1}{4\pi}\left[G(\mathbf{r}) + 3\mathbf{q}(\mathbf{r}) \cdot \hat{\mathbf{s}}\right].$$



P_1 and SP_3 - Approximations

The equations that constitute the P_1-approximation are given by

$$\nabla_\tau G = -(3 - A_1\omega)\mathbf{q}, \qquad \nabla_\tau \cdot \mathbf{q} = (1-\omega)(4\pi I_b - G)$$

or as a single elliptic second-order PDE for the incident radiation

$$\frac{1}{3\kappa}\nabla \cdot \left(\frac{1}{\beta - A_1\sigma_s/3}\nabla G\right) - G = -4\pi I_{\eta b},$$

expressed in terms of $\omega \equiv \sigma_\eta/\beta_\eta$, the absorption, scattering, and extinction coefficients.

P_1 and SP_3 - Approximations

The equations that constitute the P_1-approximation are given by

$$\nabla_\tau G = -(3 - A_1\omega)\mathbf{q}, \qquad \nabla_\tau \cdot \mathbf{q} = (1-\omega)(4\pi I_b - G)$$

or as a single elliptic second-order PDE for the incident radiation

$$\frac{1}{3\kappa}\nabla \cdot \left(\frac{1}{\beta - A_1\sigma_s/3}\nabla G\right) - G = -4\pi I_{\eta b},$$

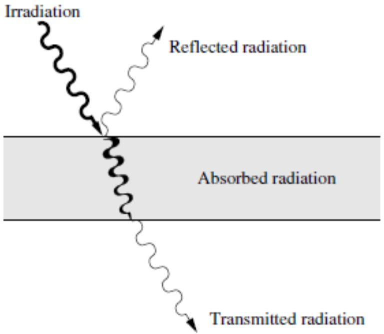
expressed in terms of $\omega \equiv \sigma_\eta/\beta_\eta$, the absorption, scattering, and extinction coefficients.

The radiative heat flux can be expressed as

$$\mathbf{q} = -\frac{1}{3 - A_1\omega}\nabla_\tau G.$$

and the Marshak boundary conditions for the P_1-approximation

$$\frac{2-\epsilon}{\epsilon}G - \frac{1}{\epsilon}\nabla \cdot \mathbf{q} = 4\pi I_{bw}.$$



Source: Modest M., Radiative heat transfer

P_1 and SP_3 - Approximations

SP_3 - Approximations

The $I(\mathbf{r}, \hat{s})$ is truncated beyond $l = 3$ with $m = \pm 3\,, \pm 2, \pm 1, 0$

$$I(\mathbf{r}, \hat{s}) = J_o(\mathbf{r}) + J_2(\mathbf{r}) \cdot P_2(\hat{s}),$$

where $J_0(\mathbf{r})$ is the zeroth moment, $J_2(\mathbf{r})$ the second moment, and $P_2(\hat{\mathbf{s}}) = 1/2\,(3\cos^2\theta - 1)$

P_1 and SP_3 - Approximations

SP_3 - Approximations

The $I(\mathbf{r}, \hat{s})$ is truncated beyond $l = 3$ with $m = \pm 3\,, \pm 2, \pm 1, 0$

$$I(\mathbf{r}, \hat{s}) = J_o(\mathbf{r}) + J_2(\mathbf{r}) \cdot P_2(\hat{s}),$$

where $J_0(\mathbf{r})$ is the zeroth moment, $J_2(\mathbf{r})$ the second moment, and $P_2(\hat{\mathbf{s}}) = 1/2\,(3\cos^2\theta - 1)$

P_1 and SP_3 - Approximations

SP_3 - Approximations

- The Simplified P_N (SP_N) approximation reduces the complexity of P_N equations.
- Odd-order terms are treated as vectors with divergence operators, and even-order terms remain scalar with gradients.

SP_N - Approximations

- The Simplified P_N (SP_N) approximation reduces the complexity of P_N equations.
- Odd-order terms are treated as vectors with divergence operators, and even-order terms remain scalar with gradients.

$$I(\tau_\eta, \mu) \approx \sum_{l=0}^{N} I_l(\tau_\eta) P_l(\mu),$$

where $I_l(\tau_\eta)$ represents coefficients that depend only on the optical thickness τ_η. By substituting this expression into the RTE and applying the orthogonality properties of Legendre polynomials, a system of coupled differential equations for the coefficients $I_l(\tau_\eta)$ is obtained.

$$\frac{k+1}{2k+1} I'_{k-1}(\tau_\eta) + \frac{k}{2k-1} I'_{k-1}(\tau_\eta) + \left(1 - \frac{\omega A_k}{2k+1}\right) I_k(\tau_\eta) = (1-\omega) I_b(\tau_\eta) \delta_{0k},$$

where $k = 0, 1, \ldots, N$, and ω is the scattering albedo of the medium.

P_1 and SP_3 - Approximations

P_3 - Approximations

- Odd-order terms vanish due to the dependence of $P_n^m(\cos\theta)$ on odd powers of $\cos\theta$, resulting in $I_n^m = 0$.
- Symmetry constraints eliminate terms with $m > n$ and z-derivatives in 2D problems.
- Higher-order terms like I_4^m are excluded in P_3, simplifying to four key equations.

P_1 and SP_3 - Approximations

P_3 - Approximations

- Odd-order terms vanish due to the dependence of $P_n^m(\cos\theta)$ on odd powers of $\cos\theta$, resulting in $I_n^m = 0$.
- Symmetry constraints eliminate terms with $m > n$ and z-derivatives in 2D problems.
- Higher-order terms like I_4^m are excluded in P_3, simplifying to four key equations.

SP_3 - Approximations

- SP_3 removes cross-terms and complex derivatives present in P_3, resulting in a more efficient and practical formulation.
- Unlike P_3, SP_3 captures second-order anisotropies while maintaining simplicity, making it ideal for multidimensional systems and optically thick media.

$$\sum_{k\,\text{even}} p^0_{k,2i-1} I_k - \sum_{k\,\text{odd}} \frac{p^0_{k,2i-1}}{\alpha_k} \left[\frac{k}{2k-1}\, \hat{n} \cdot \nabla_\tau I_{k-1} + \frac{k+1}{2k+3}\, \hat{n} \cdot \nabla_\tau I_{k+1} \right] = \frac{p^0_{0,2i-1}}{\pi} J_w,$$
$$i = 1, 2, \ldots, \frac{1}{2}(N+1).$$

P_1 and SP_3 - Approximations

The SP_3 model results in a system of two coupled equations

$$\frac{1}{3}\nabla \cdot \left(\frac{1}{\kappa}\nabla J_0\right) - \kappa\left(J_0 - \frac{2}{3}J_2 - I_b\right) = 0,$$
$$\frac{3}{7}\nabla \cdot \left(\frac{1}{\kappa}\nabla J_2\right) + \frac{2}{3}\nabla \cdot \left(\frac{1}{\kappa}\nabla J_0\right) - \kappa J_2 = 0.$$

The incident radiation G and the radiative flux $\mathbf{q}$

$$G = 4\pi\left(J_0 - \frac{2}{3}J_2\right), \qquad \mathbf{q} = -\frac{1}{3\kappa}\nabla J_0,$$

where the $-\frac{2}{3}J_2$ factor arises from the contribution of the quadratic term $P_2^0(\hat{\mathbf{s}})$.

The Marshak Boundary Conditions

$$i = 1: \qquad \frac{1}{3\alpha_1}\hat{\mathbf{n}} \cdot \nabla_\eta J_0 = \frac{1}{2}\left(J_0 - \frac{J_w}{\pi}\right) + \frac{1}{8}J_2,$$
$$i = 2: \qquad \frac{1}{7\alpha_3}\hat{\mathbf{n}} \cdot \nabla_\eta J_2 = -\frac{1}{8}\left(J_0 - \frac{J_w}{\pi}\right) + \frac{7}{24}J_2.$$

Implementation

The structure of this directory.

The radiation functionality in OpenFOAM is found under the thermophysicalModels directory.

```
thermophysicalModels
└── radiation
    ├── Make
    ├── derivedFvPatchFields
    │   ├── MarshakRadiation
    │   ├── SP3MarshakRadiation
    │   │   ├── SP3MarshakRadiationFvPatchScalarField.C
    │   │   └── SP3MarshakRadiationFvPatchScalarField.H
    │   ├── MarshakRadiationFixedTemperature
    │   └── ..
    ├── radiationModels
    │   ├── P1
    │   └── SP3
    │       ├── SP3.C
    │       └── SP3.H
    └── ..
```

Implementation

```
Foam::radiation::SP3::SP3(const volScalarField& T):
J2_(IOobject(
    "J2",
    mesh_.time().timeName(),
    mesh_,
    IOobject::NO_READ,
    IOobject::NO_WRITE
    ),mesh_),
a1_(IOobject(
    "a1",
    mesh_.time().timeName(),
    mesh_,
    IOobject::NO_READ,
    IOobject::AUTO_WRITE
    ),mesh_,dimensionedScalar(dimless/dimLength, Zero)),
a2_(IOobject(
    "a2",
    mesh_.time().timeName(),
    mesh_,
    IOobject::NO_READ,
    IOobject::AUTO_WRITE
    ),mesh_,dimensionedScalar(dimless/dimLength, Zero)),
//...
```

SP3.C - Constructor

`Foam::radiation::SP3::SP3`

- Initializes the diffusivity and absorption coefficients a, a_1, a_2, a_3, the second moment J_2, and the radiative flux q_r across domain boundaries.
- This ensures proper boundary treatment and overall energy conservation.

Implementation

SP3.C - calculate()

The code initializes coefficients and emission terms while introducing a small stabilization constant a_0 to ensure numerical stability in the radiative transfer calculations.

```
a_ = dimensionedScalar("alpha0", coeffs_); // Absorption coefficient for J0
a1_ = dimensionedScalar("alpha1", coeffs_); // Diffusivity for J0
a2_ = dimensionedScalar("alpha2", coeffs_); // Absorption coefficient for J2
a3_ = dimensionedScalar("alpha3", coeffs_); // Diffusivity for J2
e_ = absorptionEmission_->e();
E_ = absorptionEmission_->E();
```

Implementation

SP3.C - calculate()

The code initializes coefficients and emission terms while introducing a small stabilization constant a_0 to ensure numerical stability in the radiative transfer calculations.

```
a_ = dimensionedScalar("alpha0", coeffs_); // Absorption coefficient for J0
a1_ = dimensionedScalar("alpha1", coeffs_); // Diffusivity for J0
a2_ = dimensionedScalar("alpha2", coeffs_); // Absorption coefficient for J2
a3_ = dimensionedScalar("alpha3", coeffs_); // Diffusivity for J2
e_ = absorptionEmission_->e();
E_ = absorptionEmission_->E();
```

```
const volScalarField gamma0(
    IOobject("gamma0Rad", G_.mesh().time().timeName(), G_
    .mesh(),
              IOobject::NO_READ, IOobject::NO_WRITE),
    1.0/(3.0*a1_ + sigmaEff + a0));
const volScalarField gamma2(
    IOobject("gamma2Rad", G_.mesh().time().timeName(), G_
    .mesh(),
              IOobject::NO_READ, IOobject::NO_WRITE),
    3.0/(7.0*a3_ + sigmaEff + a0));
```

The diffusion coefficients are defined as

$$\gamma_0 = \frac{1}{3(a_1 + \sigma_{\text{eff}} + a_0)},$$
$$\gamma_2 = \frac{3}{7(a_3 + \sigma_{\text{eff}} + a_0)},$$

where a_1 and a_3 are the diffusivity terms associated with J_0 and J_2

Implementation

Coupled equations

The SP_3 model is based on two coupled equations to solve the RTE, which will be implemented in the `calculate()` function.

```
solve
(
    fvm::laplacian(gamma0, G)                   // Diffusion term
  - fvm::Sp(kappa_, G)                          // Absorption term
  == -(2.0 / 3.0) * J2_                         // Coupling term with J2
     - 4.0 * E                                  // Blackbody emission
);
```

The code corresponds at the equation G,

$$\frac{1}{3}\nabla \cdot \left(\frac{1}{\kappa}\nabla J_0\right) - \kappa\left(J_0 - \frac{2}{3}J_2 - I_b\right) = 0$$

Implementation

The equation for the second moment J_2 is

$$\frac{1}{3}\nabla\cdot\left(\frac{1}{\kappa}\nabla J_2\right)+\frac{2}{3}\nabla\cdot\left(\frac{1}{\kappa}\nabla J_0\right)-\kappa J_2=0$$

```
solve
(
    fvm::laplacian(gamma2, J2_)             // Diffusion term
  - fvm::Sp(a2_, J2_)                       // Absorption term
  == (2.0 / 3.0) * couplingTerm             // Coupling with G
);
```

```
volScalarField couplingTerm(
    IOobject
    (
        "couplingTerm",
        G_.mesh().timeName(),
        G_.mesh(),
        IOobject::NO_READ,
        IOobject::NO_WRITE
    ),
    fvc::div(fvc::grad(G) / a1_));
```

The coupling term $\frac{2}{3}\nabla\cdot\left(\frac{1}{\kappa}\nabla J_0\right)$ explicitly connects G with J_2

Implementation

Radiative Heat Flux on Boundaries

The boundary fields in the code are `qrBf` for the radiative flux q_r, `GBf` for the zeroth moment G, `J2Bf` for the second moment J_2, `gamma0Bf` and `gamma2Bf` for diffusivities γ_0 and γ_2.

```
    volScalarField::Boundary& qrBf = qr_.boundaryFieldRef();
    const volScalarField::Boundary& GBf = G_.boundaryField();
    const volScalarField::Boundary& J2Bf = J2_.boundaryField();
    const volScalarField::Boundary& gamma0Bf = gamma0.boundaryField();
    const volScalarField::Boundary& gamma2Bf = gamma2.boundaryField();

    // Iterate through all boundary patches
    forAll(G_.mesh().boundary(), patchI)
    {
        const fvPatch& patch = G_.mesh().boundary()[patchI];
        if (!GBf[patchI].coupled())
        {   // Compute radiative flux using gradient terms from J0 (G_) and J2
            qrBf[patchI] =
                -gamma0Bf[patchI] * GBf[patchI].snGrad()
                -gamma2Bf[patchI] * J2Bf[patchI].snGrad();
        }
        //...
    }
```

SP3MarshakRadiationFvPatchScalarField.C - Boundary conditions

The boundary field for J_2 (J2Boundary) is initialized from the file system during the simulation setup.

```
    // Diffusivity - created by radiation model's ::updateCoeffs()
    const auto& gamma0 = patch().lookupPatchField<volScalarField>("gamma0Rad");
    const auto& gamma2 = patch().lookupPatchField<volScalarField>("gamma2Rad");
    const auto& J2Boundary = patch().lookupPatchField<volScalarField>("J2");
```

Implementation

SP3MarshakRadiationFvPatchScalarField.C - Boundary conditions

The boundary field for J_2 (J2Boundary) is initialized from the file system during the simulation setup.

```
    // Diffusivity - created by radiation model's ::updateCoeffs()
    const auto& gamma0 = patch().lookupPatchField<volScalarField>("gamma0Rad");
    const auto& gamma2 = patch().lookupPatchField<volScalarField>("gamma2Rad");
    const auto& J2Boundary = patch().lookupPatchField<volScalarField>("J2");
```

The valueFraction() is calculated to incorporate both γ_0 and γ_2, and The factor of 0.5 is used due to the normalization and orthogonality of the Legendre polynomial $P_2(\cos\theta)$.

```
    // Effective emissivity factor for SP3
    const scalarField Ep = emissivity / (2.0 * (scalar(2) - emissivity));

    // Calculate value fraction for SP3 (depends on gamma0 and gamma2 contributions)
    valueFraction() = 1.0 / (1.0 + (gamma0 + gamma2) * patch().deltaCoeffs() / Ep);

    // Update boundary values for J2 using the read boundary field
    const scalar reflectionFactor = 0.5; // Reflection factor for SP3
    patchInternalField() =
        reflectionFactor * J2Boundary; // Use the boundary field directly
```

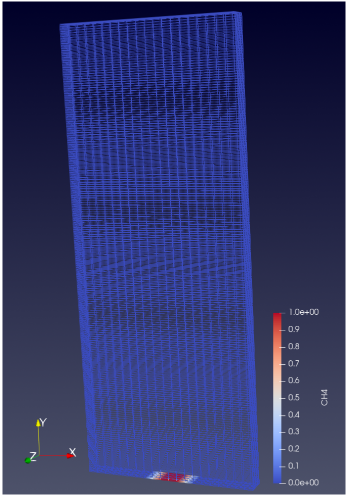
Case Study: smallPoolFire2D

Case Study: smallPoolFire2D

smallPoleFire2D

```
cp -r $FOAM_TUTORIALS/combustion/fireFoam/smallPoleFire2D $FOAM_RUN
run
cd smallPoleFire2D
```

- It is used to compare P_1 and SP_3-approximations to predict radiative heat transfer and temperature distributions.
- In this case, methane (CH_4) is used as a gaseous fuel to simulate the radiative effects.
- Critical parameters, such as the Heat Release Rate (HRR) and incident radiative flux, are calculated.



smallPoleFire2D

```
cp -r $FOAM_TUTORIALS/combustion/fireFoam/smallPoleFire2D $FOAM_RUN
run
cd smallPoleFire2D
```

- It is used to compare P_1 and SP_3-approximations to predict radiative heat transfer and temperature distributions.
- In this case, methane (CH_4) is used as a gaseous fuel to simulate the radiative effects.
- Critical parameters, such as the Heat Release Rate (HRR) and incident radiative flux, are calculated.

Physical and Numerical Models for the PoolFire2D Case

Model	**Choice**
Turbulence	`LES` (Large Eddy Simulation)
Combustion	`EDM` (Eddy Dissipation Model)
Radiative Heat Transfer	$\mathtt{P}_1$ and $\mathtt{SP}_3$ radiation models
Emission and Absorption	`constantAbsorptionEmission` with case-dependent coefficients
Soot Formation	None

Case Study: smallPoolFire2D

constant directory: radiationProperties

The file radiationProperties specifies the general settings for the radiation model. Here, the SP_3 model is activated with the following entries

```

radiation on;

radiationModel  SP3;

SP3Coeffs
{
alpha0 [0 -1 0 0 0 0 0] 1.0;     // [1/m]
alpha1 [0 -1 0 0 0 0 0] 0.8;    // [1/m]
alpha2 [0 -1 0 0 0 0 0] 0.8;    // [1/m]
alpha3 [0 -1 0 0 0 0 0] 0.9;    // [1/m]
}
solverFreq 10;
\...
```

Case Study: smallPoolFire2D

system directory: `fvSolution`

- Solver configurations for G and J_2.
- The `GAMG` solver ensures efficient convergence with the `DICGaussSeidel` smoother for both moments.

```
"(G|J2)"
{
    solver          GAMG;
    tolerance       1e-06;
    relTol          0.1;
    smoother        DICGaussSeidel;
};

"(G|J2)Final"
{
    solver          GAMG;
    tolerance       1e-08;
    relTol          0;
};

```

system directory: fvSchemes

Gradient and divergence schemes for G and J_2:

```
divSchemes{
    default none;
    div(phi,U) Gauss LUST grad(U);
    div(U) Gauss linear;
    div(phi,K) Gauss linear;
    div(phi,k) Gauss limitedLinear 1;
    div(phi,FSDomega) Gauss limitedLinear 1;
    div(phi,Yi_h) Gauss multivariateSelection
    {
        O2 limitedLinear01 1;
        CH4 limitedLinear01 1;
        N2 limitedLinear01 1;
        H2O limitedLinear01 1;
        CO2 limitedLinear01 1;
        h limitedLinear 1;
    };
    div(((rho*nuEff)*dev2(T(grad(U))))) Gauss linear;
    div(Ji,Ii_h) Gauss upwind;          //moment - J
    div((grad(G)|a1)) Gauss linear;}     //moment - G
```

Case Study: smallPoolFire2D

0/: J_2 Initialization

The Initialization of J_2 has the `SP3MarshakRadiation` boundary condition and manages radiative flux interactions with the domain boundaries.

```
dimensions      [1 0 -3 0 0 0 0];

internalField   uniform 0.000001;

boundaryField
{
    ".*"
    {
        type            SP3MarshakRadiation;
        value           uniform 0;
    }

    frontAndBack
    {
        type            empty;
    }
}

```

Case Study: smallPoolFire2D

`controlDict` - Heat Release Rate (HRR)

The `volFieldValue` calculates the Heat Release Rate (HRR) by integrating the radiative heat flux (`Qdot`) over the entire computational domain.

```
HRR
{
    type              volFieldValue;
    libs             ("libfieldFunctionObjects.so");
    log               true;
    writeControl      timeStep;
    writeInterval     1;
    writeFields       false;
    regionType        all;
    operation         volIntegrate;
    enabled           true;
    fields
    (
        Qdot
    );
}
```

Case Study: smallPoolFire2D

controlDict - Thermocouple measurements

The thermoCoupleProbes function simulates thermocouples to measure temperature at selected positions within the plume.

```
thermoCouple
{   type                thermoCoupleProbes;
    libs                (utilityFunctionObjects);
    writeControl        timeStep;
    writeInterval       1;

    solver              Euler;
    absTol              1e-4;
    relTol              1e-1;

    interpolationScheme cell;

    rho                 8908;
    Cp                  440;
    d                   1e-3;
    epsilon             0.9;
\...
```

Case Study: smallPoolFire2D

controlDict - Thermocouple measurements

The `thermoCoupleProbes` function simulates thermocouples to measure temperature at selected positions within the plume.

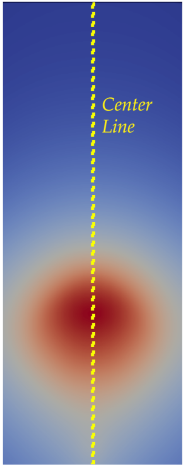
```
\...

    radiationField  G;

    probeLocations
    (
        (  0.00 0.5 0.0)
        (  0.25 0.5 0.0)
        (  0.50 0.5 0.0)
    );
    fields
    (
        T
    );
}
```

controlDict - Incident radiation

The surfaces function samples the incident radiation (G) along a vertical line from $(0.1, 0.0, 0.0)$ to $(0.1, 5.0, 0.0)$, with 100 evenly distributed points.



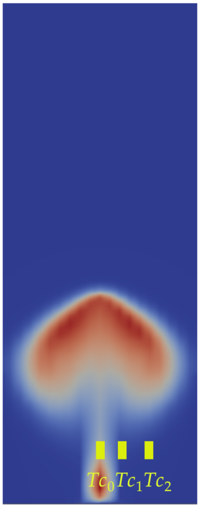
G field

```
surfaces
{   type              sets;
    libs              (sampling);
    writeControl      writeTime;
    setFormat   raw;
    interpolationScheme cellPoint;
    fields            (G);
    sets
    (   line
        {
            type    uniform;
            axis    xyz;
            start   (0.1 0.0 0.0);
            end     (0.1 5.0 0.0);
            nPoints 100;
        }
    );
}
```

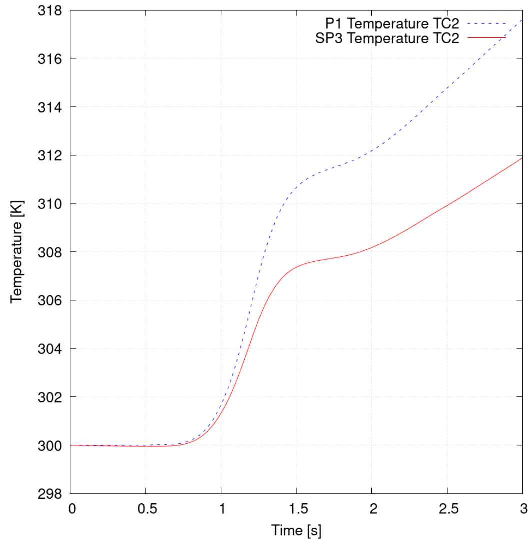
Case Study: smallPoolFire2D

Results

The temperature variation over time for P_1 and SP_3 models.



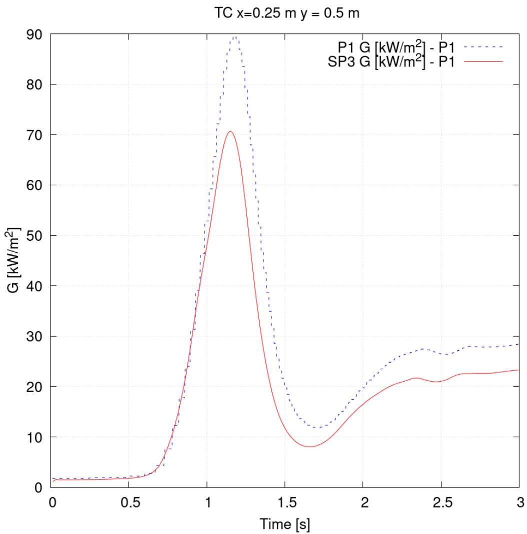
Temperature field with probes positions

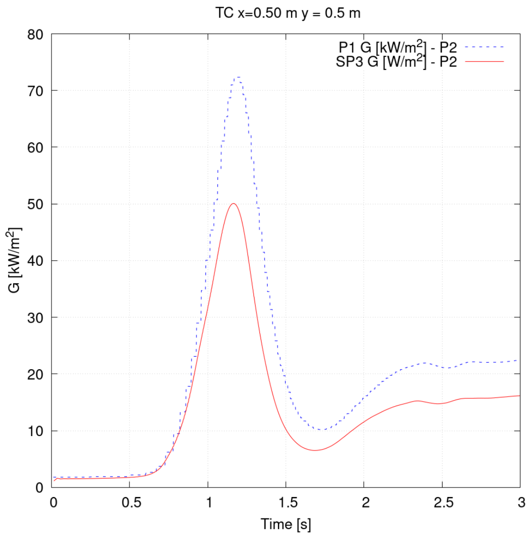


Case study

Results

Incident radiance measured in Position 1: $(0.25, 0.50, 0.0)$ and Position 2: $(0.50, 0.50, 0.0)$ using P_1 and SP_3 model





Conclusions and Future Works

Conclusions and Future Works

Conclusions

- SP_3 better predicts radiative diffusion and angular dependencies, reducing P_1's overestimation of intensity near the plume core.
- SP_3 captures gradual intensity decreases farther from the plume center, matching physical flux attenuation.
- SP_3 is more computationally demanding but offers superior accuracy for detailed fire safety and combustion simulations.

Future Works

- **Angular Validation:** Add measurements at various angles/distances to validate SP_3, focusing on regions dominated by radiative transfer.
- **Advanced Boundary Conditions:** Incorporate reflective, semi-transparent, or mixed boundary conditions to enhance SP_3 accuracy.
- **Turbulence Models:** Compare LES-SP_3 for transient dynamics and RANS-SP_3 for steady-state cases to optimize efficiency and accuracy.

LULEÅ
TEKNISKA
UNIVERSITET