Radiation heat transfer in OpenFoam

Presentation for the course
«CFD with OpenSource Software»

Alexey Vdovin
Chalmers University of Technology

Agenda(1)

- Introduction
- Addition of the radiation heat transfer to the solver
- Radiation models in the OpenFoam
  - Abstract class for radiation models
  - NoRadiation model
  - Finite Volume Discrete Ordinates Model(fvDOM)
  - P1 model
  - Radiation constants
  - Scatter models
  - Absorbtion-emission models
  - Radiation boundary conditions

Agenda(2)

- P1 radiation model files
- hotRoom case set up
- Case files modification to include radiation effects
- Results
- Conclusions

Introduction

- Radiation
  - electromagnetic waves mainly in the infrared region
  - requires no medium in contrast with a heat transfer by conduction and convection
  - the intermediaries are photons which travel at the speed of light

Addition radiation heat transfer to the solver(1)

buoyantSimpleFoam - steady-state solver for buoyant, turbulent flow of compressible fluids.

Modifications to make it work with radiation:

- *buoyantSimpleRadiationFoam.C file:*

```
#include "radiationModel.H“
#include "createRadiationModel.H“
```

- *Make/options file:*

```
EXE_INC = \
    -I$(LIB_SRC)/thermophysicalModels/radiation/lnInclude \
EXE_LIBS = \
    -lradiation \
```

Addition radiation heat transfer to the solver(2)

- *hEqn.H:*

```
        fvScalarMatrix hEqn
          (    fvm::div(phi, h)
            - fvm::Sp(fvc::div(phi), h)
            - fvm::laplacian(turbulence->alphaEff(), h)
           ==
             fvc::div(phi/fvc::interpolate(rho)*fvc::interpolate(p))
            - p*fvc::div(phi/fvc::interpolate(rho))
            + radiation->Sh(thermo)
        ...
        radiation->correct();
          );
```

Radiation models in the OpenFoam

- Abstract class
- noRadiation model
- fvDOM model
- P1 model

Abstract class for radiation models(1)

(radiation\radiationModel\radiationModel\radiationModel.C)

- Sh() member function:

```
volScalarField& h = thermo.h();
  const volScalarField cp = thermo.Cp();
  const volScalarField T3 = pow3(T_);
  return
  (
     Ru()
   - fvm::Sp(4.0*Rp()*T3/cp, h)
   - Rp()*T3*(T_ - 4.0*h/cp)
 );
```

$$Sh() = Ru() - 4Rp() * \frac{T^3 h}{C_p} - Rp()T^4 + 4Rp() * \frac{T^3 h}{C_p} = Ru() - Rp()T^4$$

Abstract class for radiation models(2)

- correct() member function:

```
void Foam::radiation::radiationModel::correct()
{
if (!radiation_)
    {
      return;
   }
    if (time_.timeIndex() % solverFreq_ == 0)
    {
    calculate();
    }
}
```

NoRadiation model

- Ru() and Rp() are set to zero
- Thereby Sh() is equal to zero
- No additional term for the enthalpy equation

Finite Volume Discrete Ordinates Model(fvDOM)

In this model the radiative transfer equation is solved for a discrete number of finite solid angles.

- Advantages of fvDOM model:
  - Conservative method leads to heat balance for coarse discretization, the accuracy can be increased by using a finer discretization,
  - Most comprehensive radiation model: Accounts for scattering, semi-transparent media, specular surfaces, and wavelength-dependent transmission using banded-gray option.
- Limitations of fvDOM model:
  - Solving a problem with a large number of ordinates is CPU-intensive.

P1 radiation model

The directional dependence in radiative transfer equation is integrated out, resulting in a diffusion equation for incident radiation.

- Advantages of P1 model:
  - Radiative transfer equation easy to solve with little CPU demand,
  - Includes effect of scattering, effects of particles, droplets, and soot can be included,
  - Works reasonably well for applications where the optical thickness is large, where L = distance between objects(e.g. model can be used in combustion).
- Limitations of P1 model:
  - Assumes all surfaces are diffuse,
  - May result in loss of accuracy (depending on the complexity of the geometry) if the optical thickness is small,
  - Tends to overpredict radiative fluxes from localized heat sources or sinks.

Radiation constants

(radiation\radiationConstants\radiationConstants.C)

```
const Foam::dimensionedScalar Foam::radiation::sigmaSB
(
    Foam::dimensionedConstant
    (
        "sigmaSB",
        dimensionedScalar
        (
            "sigmaSB",
            dimensionSet(1, 0, -3, -4, 0, 0, 0),
            5.670E-08
        )
    )
);
```

Stefan-Boltzmann constant: $\sigma_{SB} = 5.670 \cdot 10^{-8} \; {}^{W}/_{m^2} \cdot K^4$

Scatter models

(*radiation\submodels\scatterModel*)

ConstantScatter model:

```
Foam::tmp<Foam::volScalarField>
Foam::radiation::constantScatter::sigmaEff() const
{
    return tmp<volScalarField>
    (
        new volScalarField
        (
            IOobject
            (
                "sigma",
                mesh_.time().timeName(),
                mesh_,
                IOobject::NO_READ,
                IOobject::NO_WRITE,
                false
            ),
            mesh_,
            sigma_*(3.0 - C_)
        )  );
  }
```

$$sigmaEff = sigma(3.0 - C)$$

Absorption-Emission Models

(radiation\submodels\absorptionEmissionModel)

- Three coefficients that are used in radiation models to take into account absorption and emission:

$$a\ \left[{}^{1}/_{m}\right] - \mathit{absorbtion\ coefficient},$$

$$e\ \left[{}^{1}/_{m}\right] - \mathit{emission\ coefficient},$$

$$E\ \left[{}^{W}/_{m^3}\right] - \mathit{emission\ contribution}.$$

Radiation boundary conditions

(radiation\derivedFvPatchFields)

- *MarshakRadiation*

Uses calculated temperature from the case as a temperature for deriving radiation intensity.

- *MarshakRadiationFixedT*

Radiation temperature needs to be specified. This value is used for calculating G.

P1 radiation model files(1)

(radiation/radiationModel/P1/)

- P1.C :
  - *gamma* calculation:

```
const volScalarField gamma
    (
        IOobject
        (
            "gammaRad",
            G_.mesh().time().timeName(),
            G_.mesh(),
            IOobject::NO_READ,
            IOobject::NO_WRITE
        ),
        1.0/(3.0*a_ + sigmaEff)
    );
```

$$gamma = \frac{1}{3a + sigmaEff}$$

P1 radiation model files(2)

- Incident radiation intensity *G* transport equation:

```
solve
(
   fvm::laplacian(gamma, G_)
  - fvm::Sp(a_, G_)
 ==
  - 4.0*(e_*radiation::sigmaSB*pow4(T_) + E_)
);
```

- In terms of equations:

$$\nabla \cdot gamma \nabla G - aG = -4(e\sigma_{SB}T^4 + E)$$

P1 radiation model files(3)

- *Rp()* member function:

  *4.0*absorptionEmission_->eCont()*radiation::sigmaSB*

  - In terms of equations:

$$Rp(\,) = 4e\sigma_{SB}$$

- Ru() member function:

  *return a*G - 4.0*E;*

  - In terms of equations:

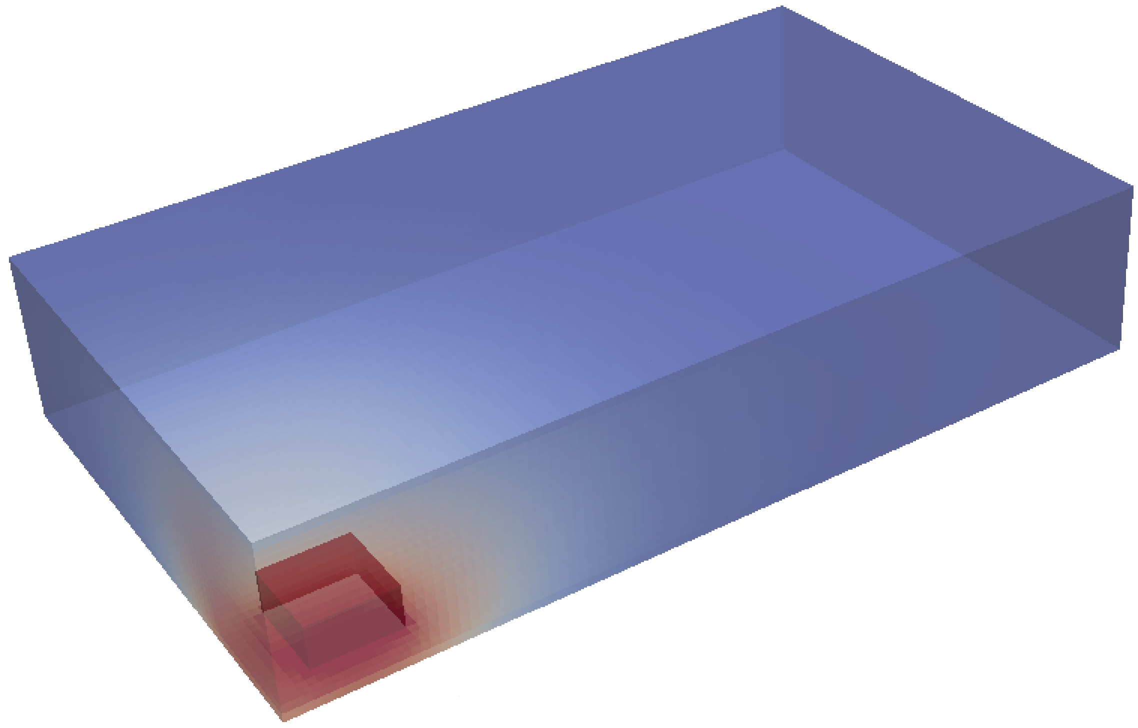
$$Ru(\,) = aG - 4E$$

- Finally the additional term *Sh() for the enthalpy equation:*

$$Sh(\,) = aG - 4(e\sigma_{SB}T^4 + E)$$

Case set up

- A room 10x6x2 meters with heater 1x1x0.5 meters in dimensions



- Temperature of the room is 300K, of the heater – 500K.

Modification of the case files to make it work with radiation(1)

- Adding *constant**radiationProperties* file:

```
radiation       on;
radiationModel  P1;
...
absorptionEmissionModel constantAbsorptionEmission;
constantAbsorptionEmissionCoeffs
{    a       a [ 0 -1 0 0 0 0 0 ] 0.5;
     e       e [ 0 -1 0 0 0 0 0 ] 0.5;
     E      E [ 1 -1 -3 0 0 0 0 ] 0;
}

scatterModel    constantScatter;
constantScatterCoeffs
{    sigma    sigma [ 0 -1 0 0 0 0 0 ] 0;
        C             C [ 0 0 0 0 0 0 0 ] 0;
}
```

Modification of the case files to make it work with radiation(2)

- Setting up boundary conditions for the radiation in 0/*G* file:

```
FoamFile
{   version     2.0;
    format      ascii;
    class       volScalarField;
    object      G;}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
dimensions      [1 0 -3 0 0 0 0];
internalField   uniform 0;

boundaryField
{
    floor
    {   type            MarshakRadiation;
        T               T;
        emissivity      1;
        value           uniform 0;    }
  ...
}
```

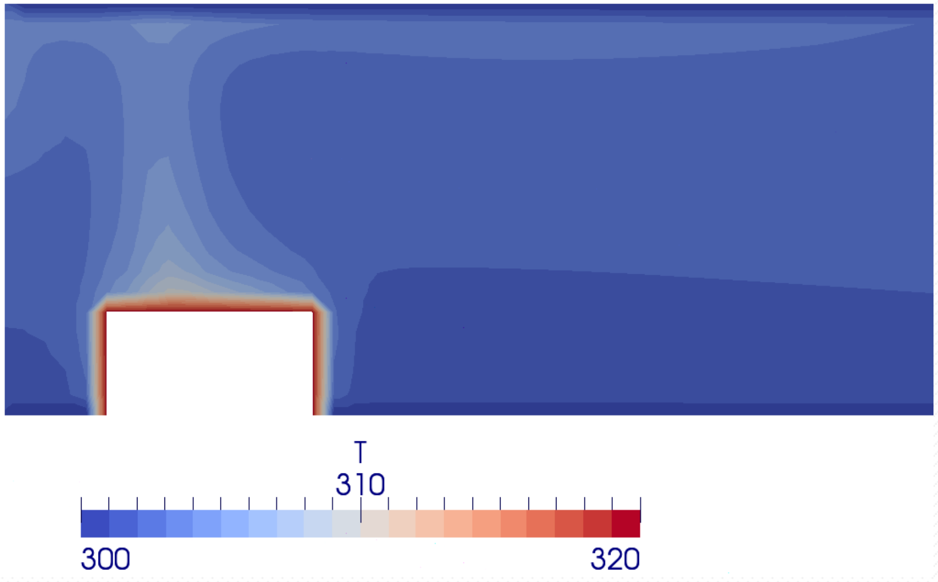
Modification of the case files to make it work with radiation(3)

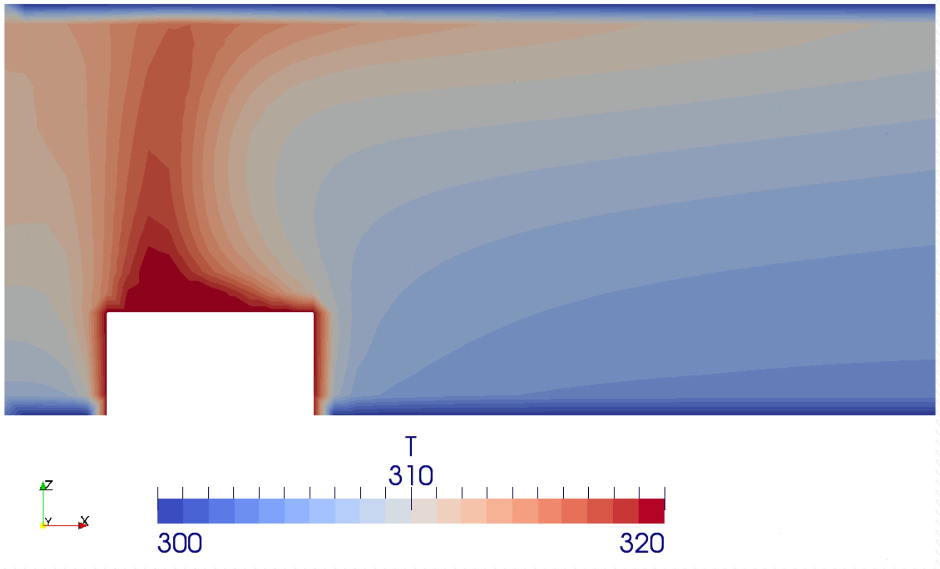
- Adding solver and relaxation factors for G to the *system/fvSolution* file:

```
solvers
{
    ...
    G
    {
        solver          PCG;
        preconditioner  DIC;
        tolerance       1e-05;
        relTol          0.1;
    }
}
...
relaxationFactors
{
...
G               0.7;
}
```

Results(1)

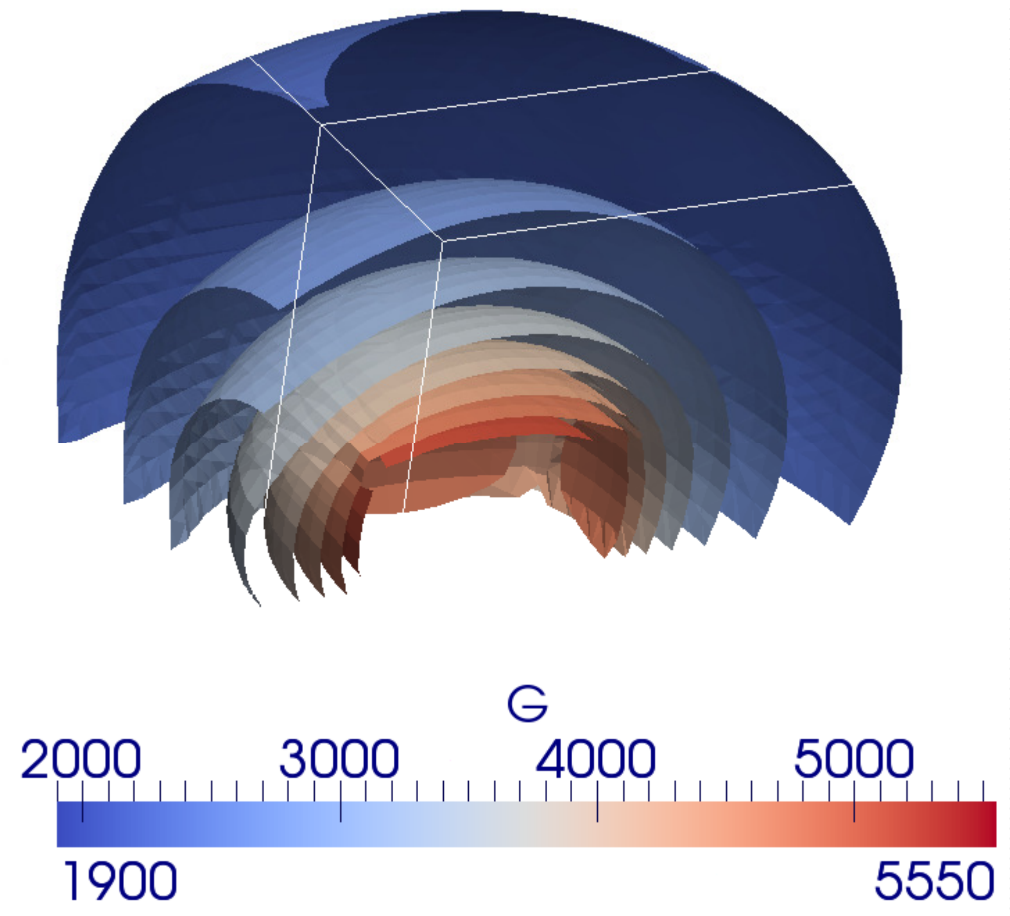
- Temperature gradient with and without radiation heat transfer





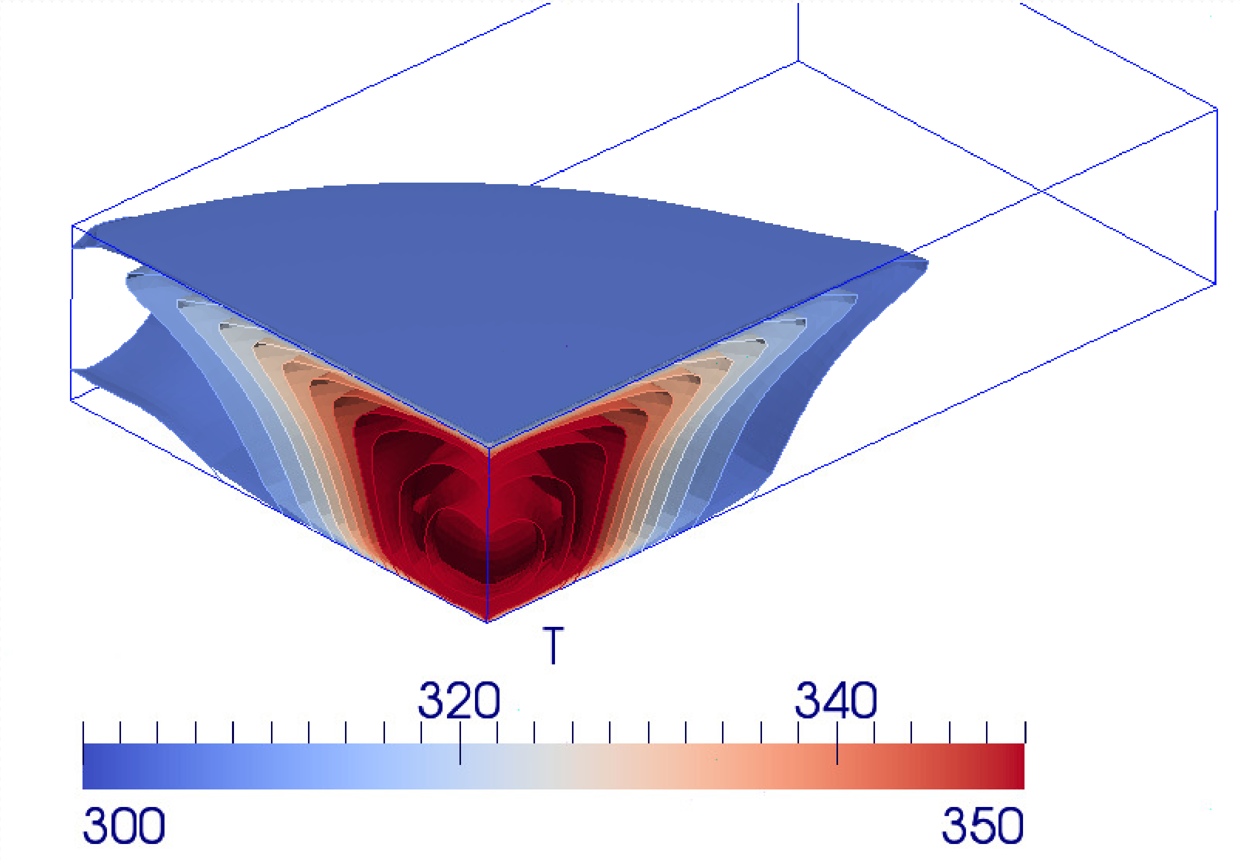
Results(2)

- Incident radiation intensity G field:



Results(3)

- Results with decreased convection effects by setting gravity constant to zero:



Conclusions

- The effect of radiation heat transfer is significant and cannot be ignored
- The addition of radiation heat transfer to the solver and case files is not a very complicated task
- Heat transfer model should be selected depending on the needs and computational power available
- Before running the case a proper scatter model, boundary conditions and absorption-emission model need to be selected

Thank you for listening