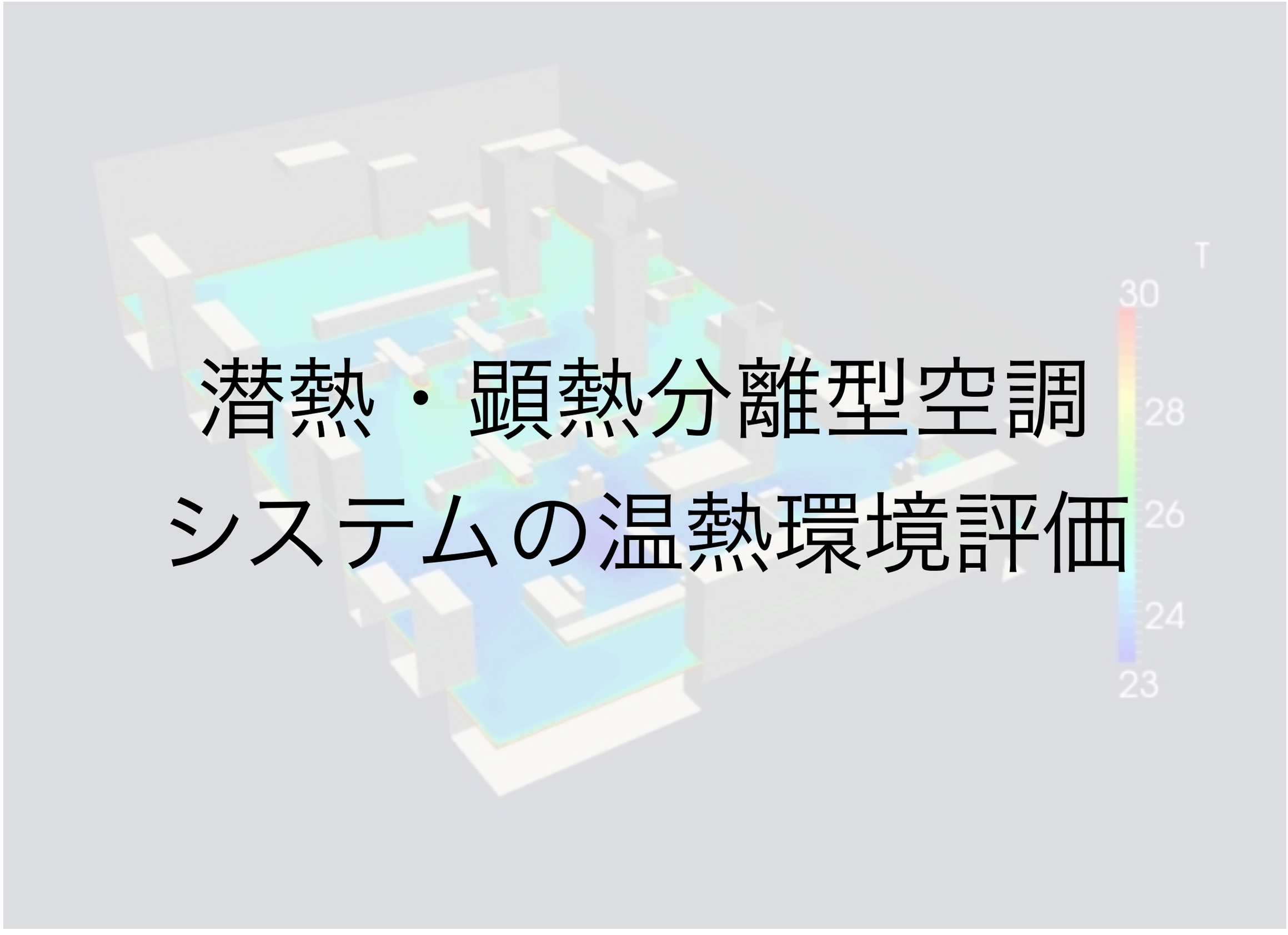
輸送方程式のソース項の実装

Ver-2.1の新機能Field sources

今野 雅(東京大学)

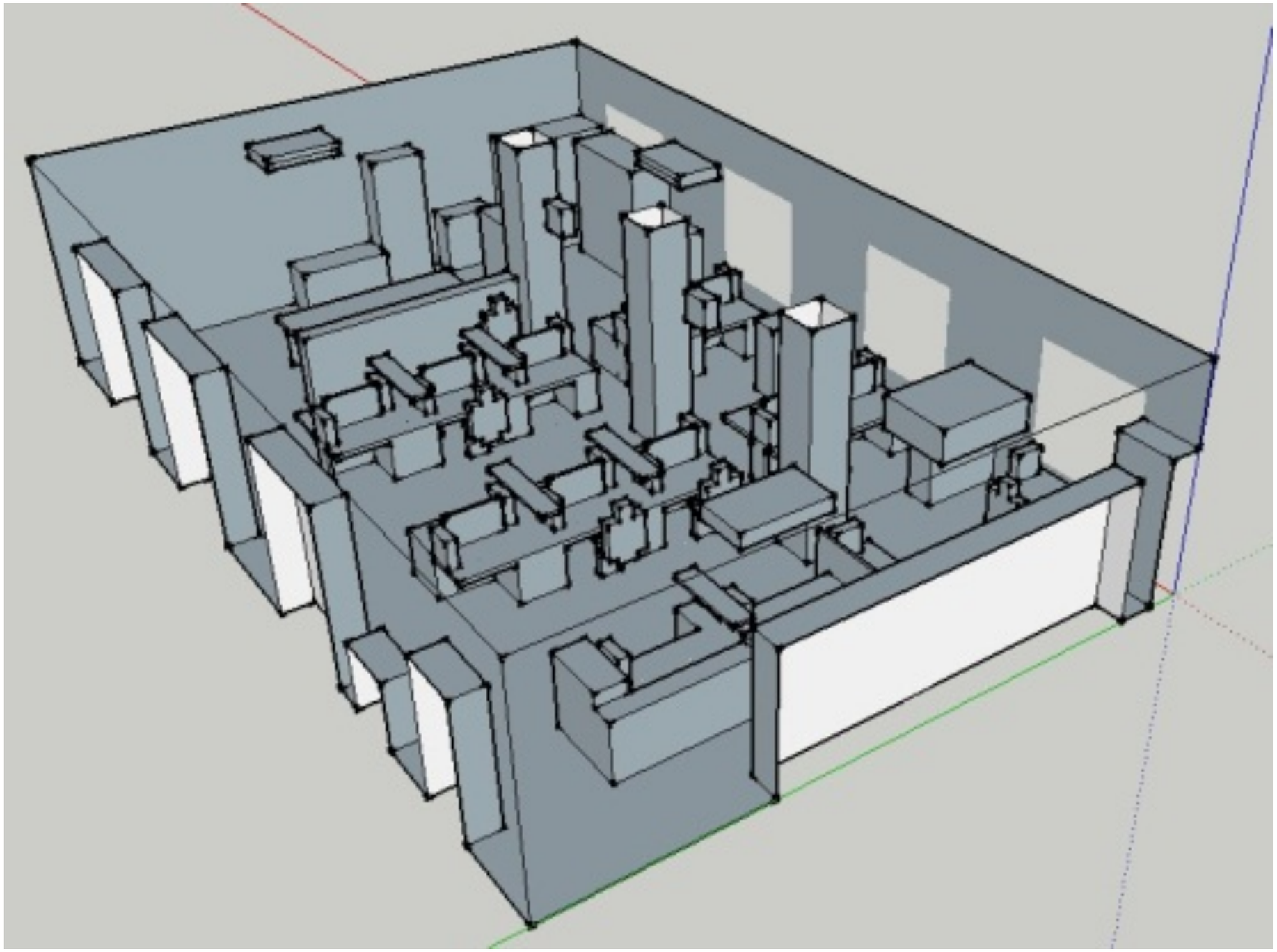
潜熱・顕熱分離型空調システムの温熱環境評価



研究の目的

- 空調設備の省エネと快適性の両立への要求が高り、潜熱・顕熱を個別に処理可能なデシカント空調システムが注目されている
- 東京大学の院生室に導入し、温熱環境測定やエネルギー消費を調査
- CFD解析で温熱環境予測

CFD解析モデル



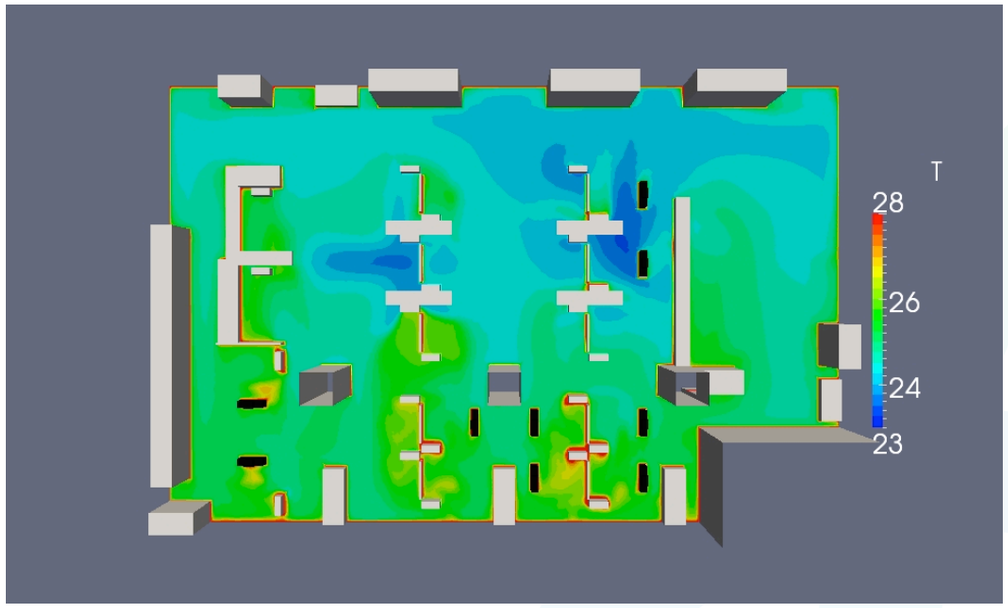
Google

Sketchup

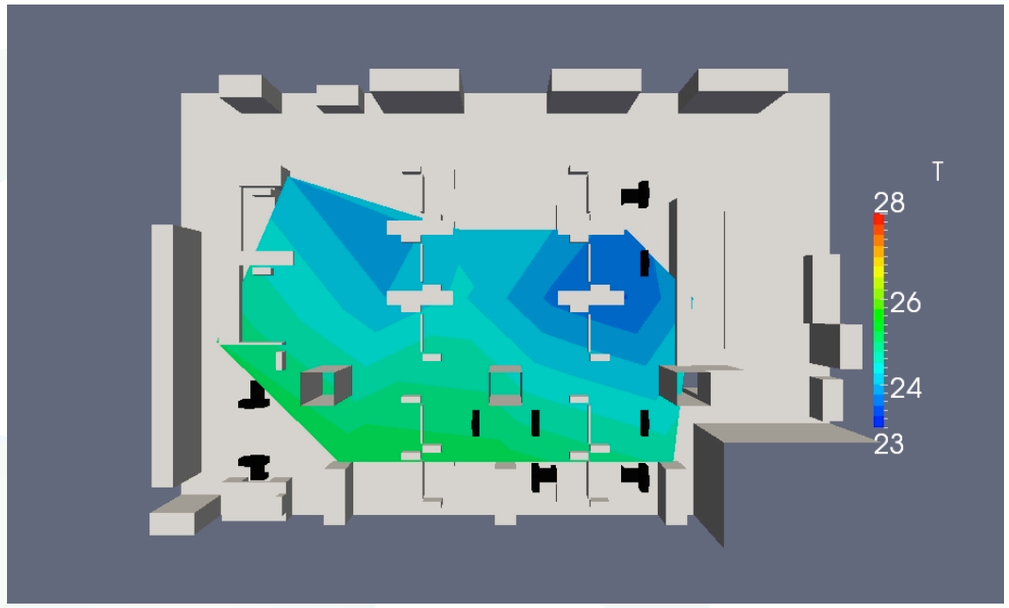
Pluginを

使ってSTL化

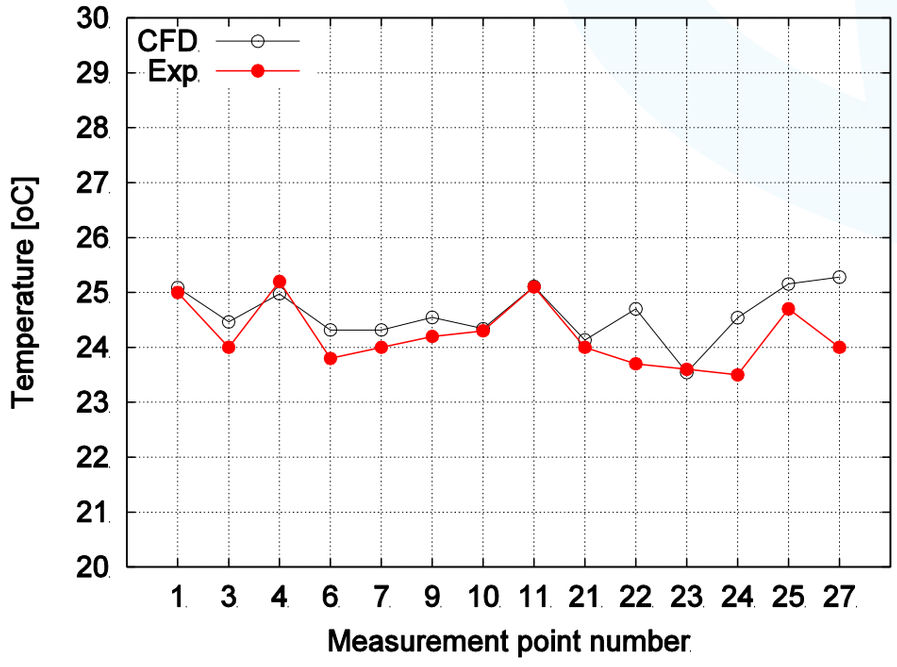
温度の実測結果との比較



CFD解析結果

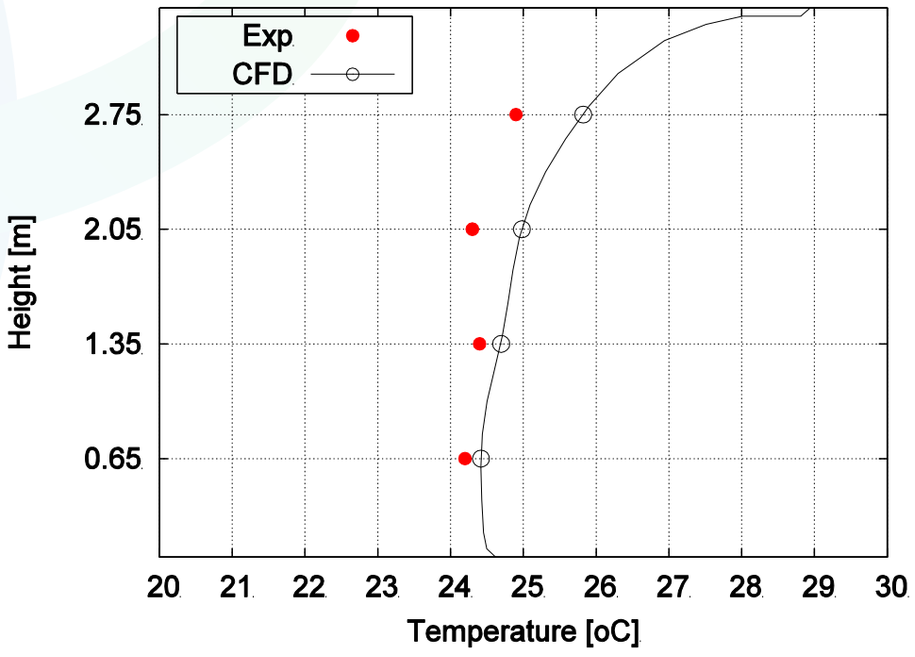


実測結果 (高さ0.8m)

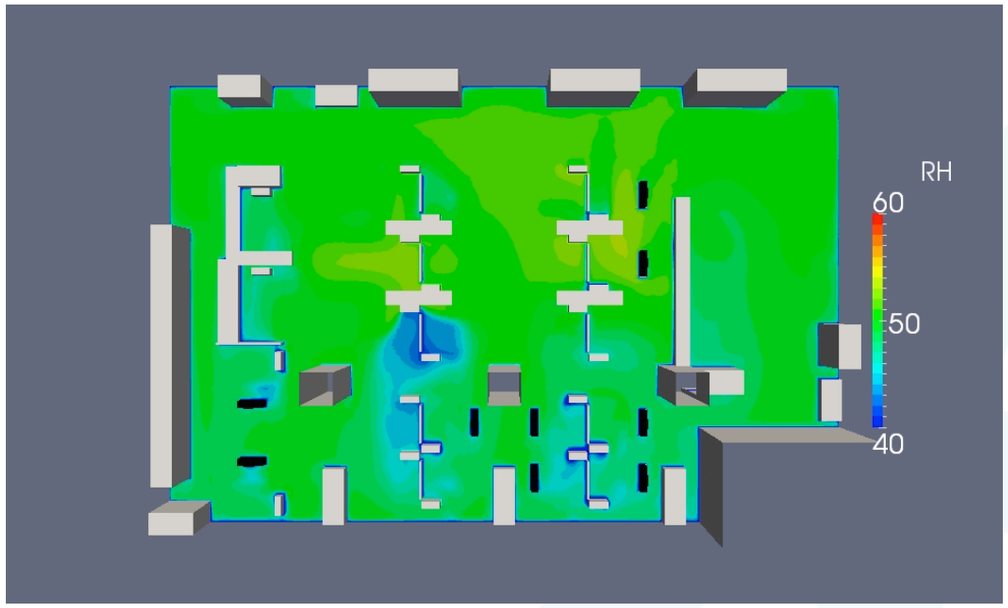


水平分布比較
(高さ0.8m)

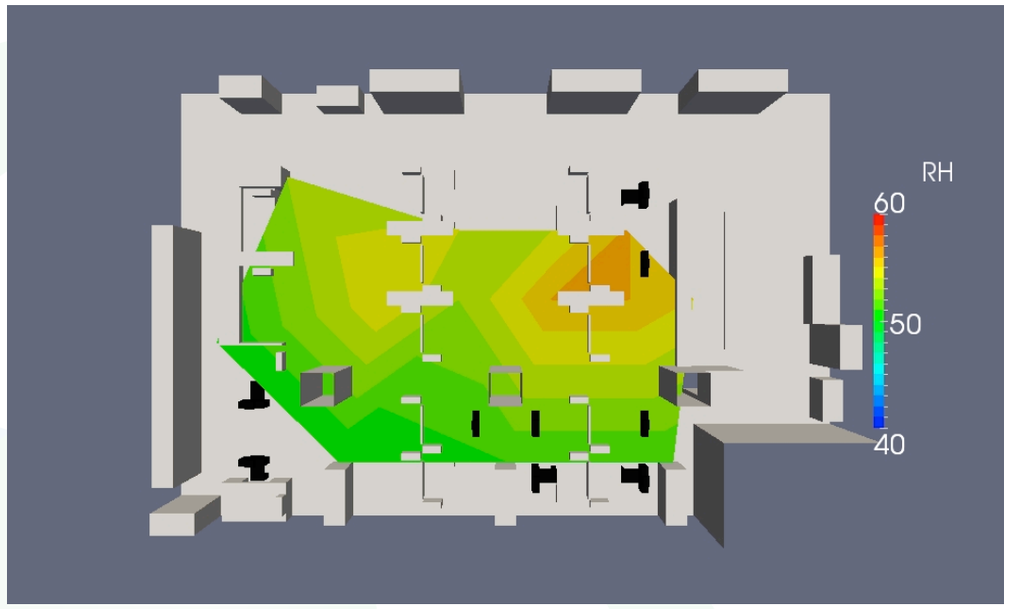
鉛直分布比較
(室中央)



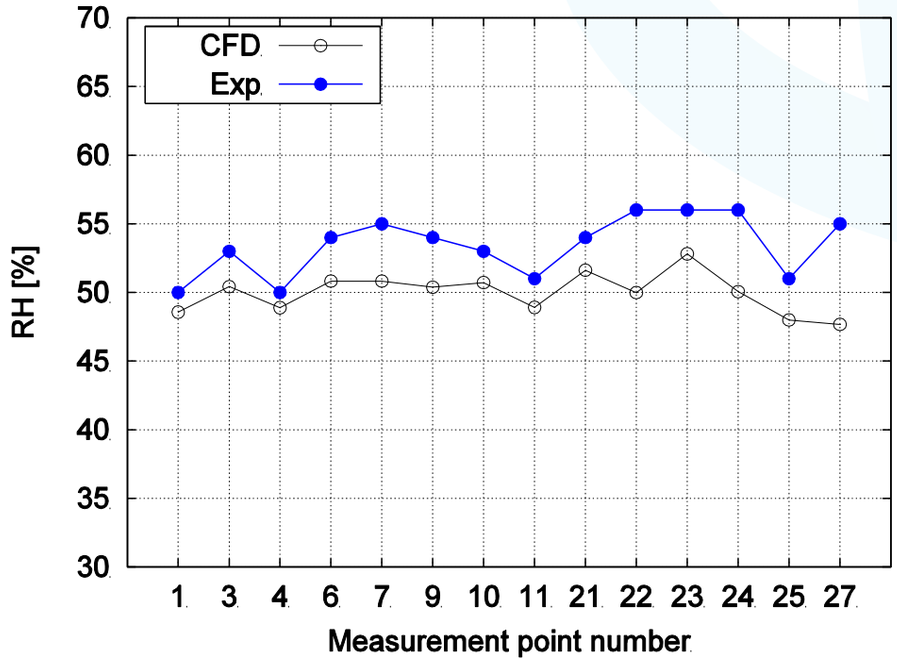
湿度の実測結果との比較



CFD解析結果

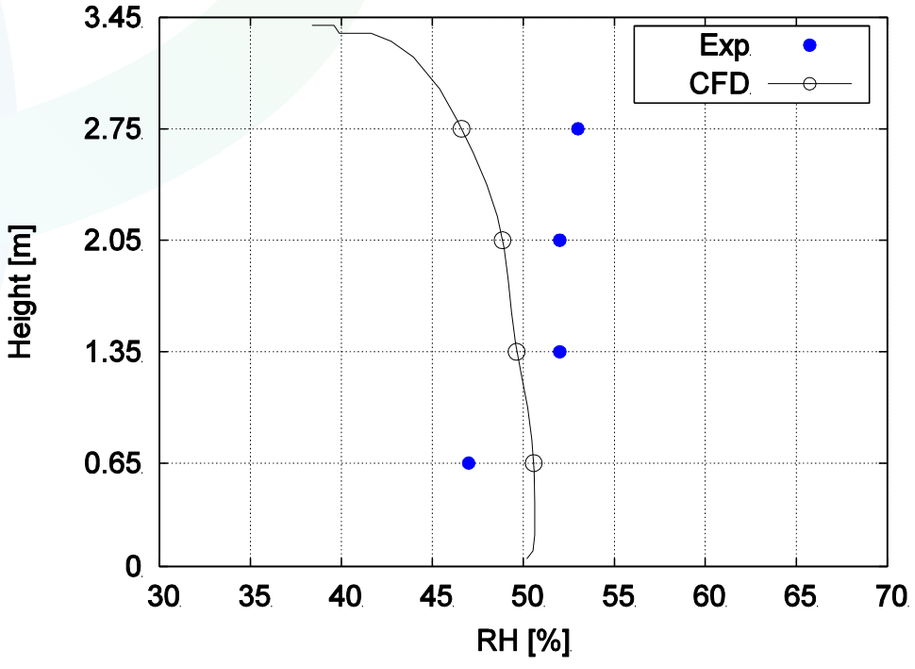


実測結果 (高さ0.8m)

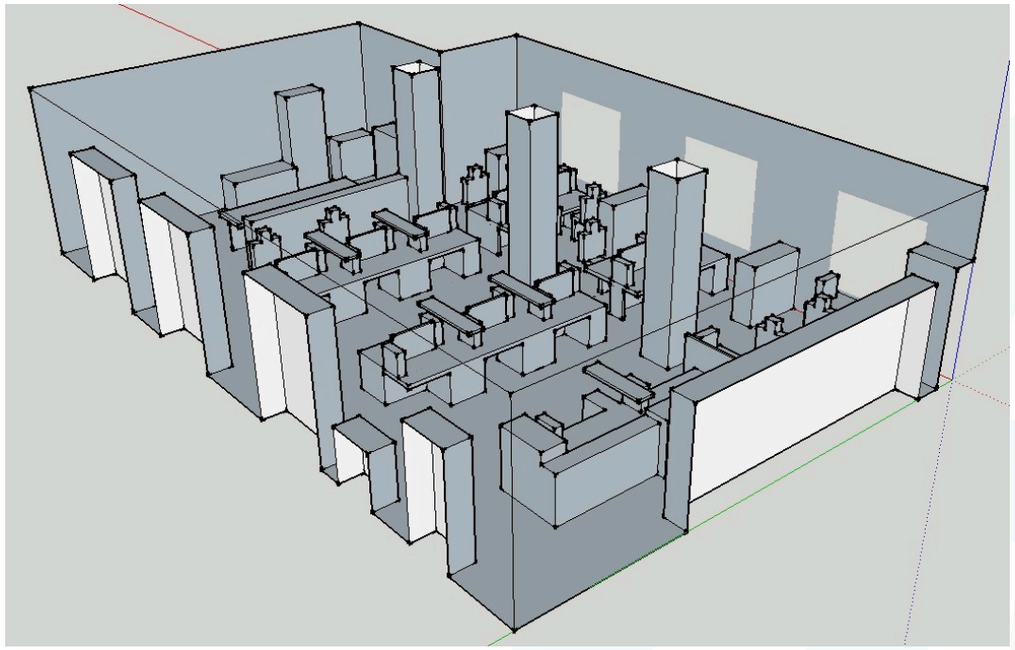


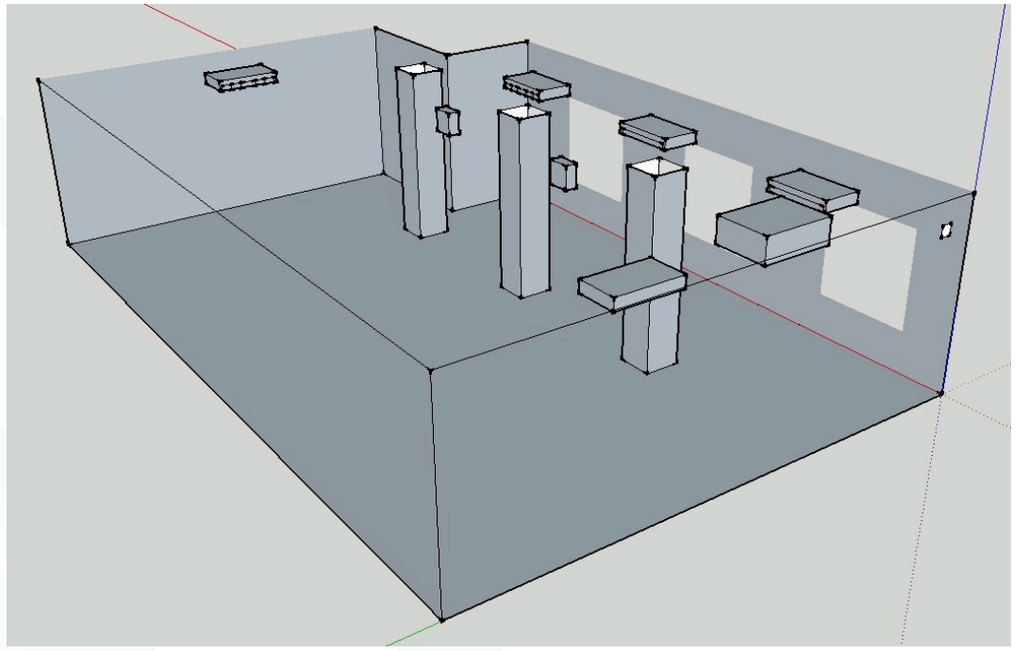
水平分布比較
(高さ0.8m)

鉛直分布比較
(室中央)



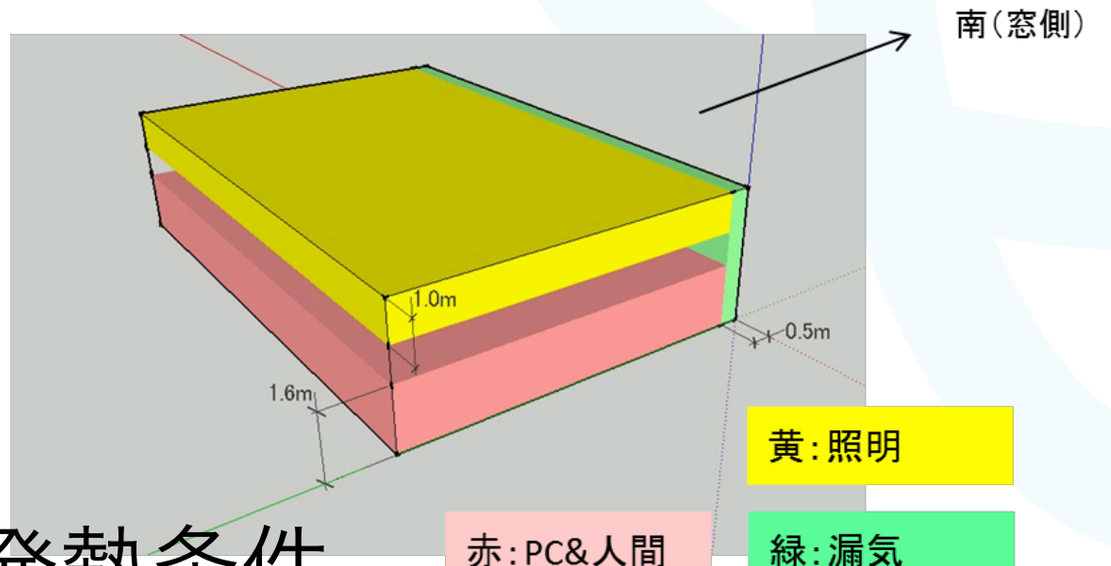
計算条件簡略化



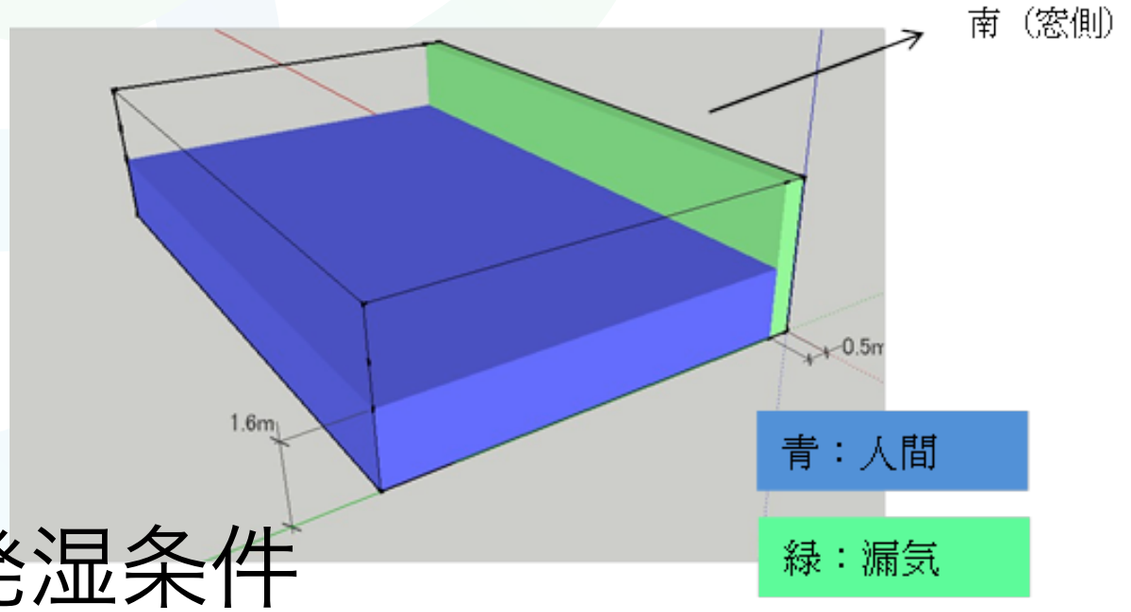


什器有り+面発熱湿(PC、人体)

什器無し+領域発熱湿

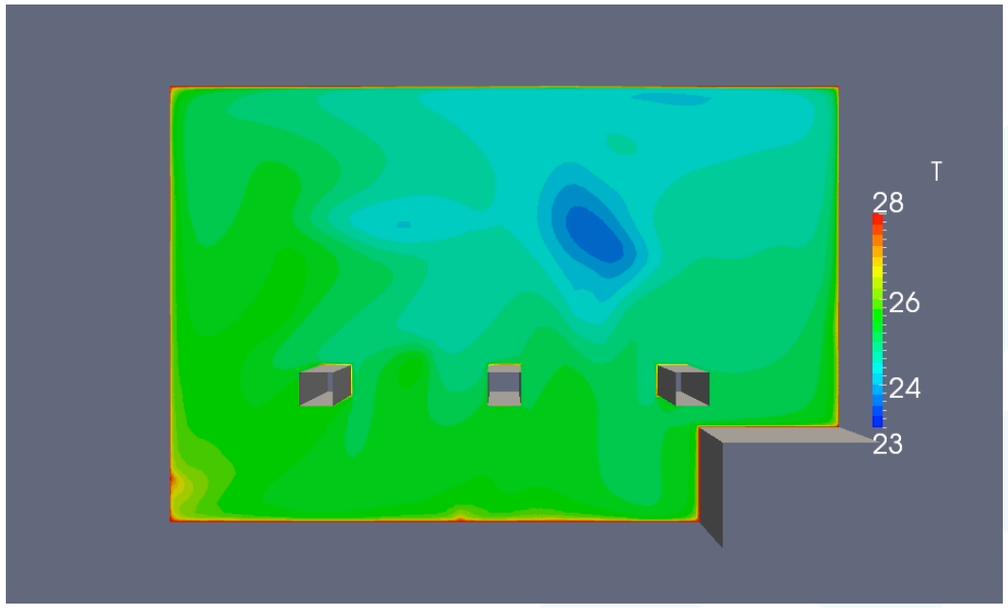


発熱条件

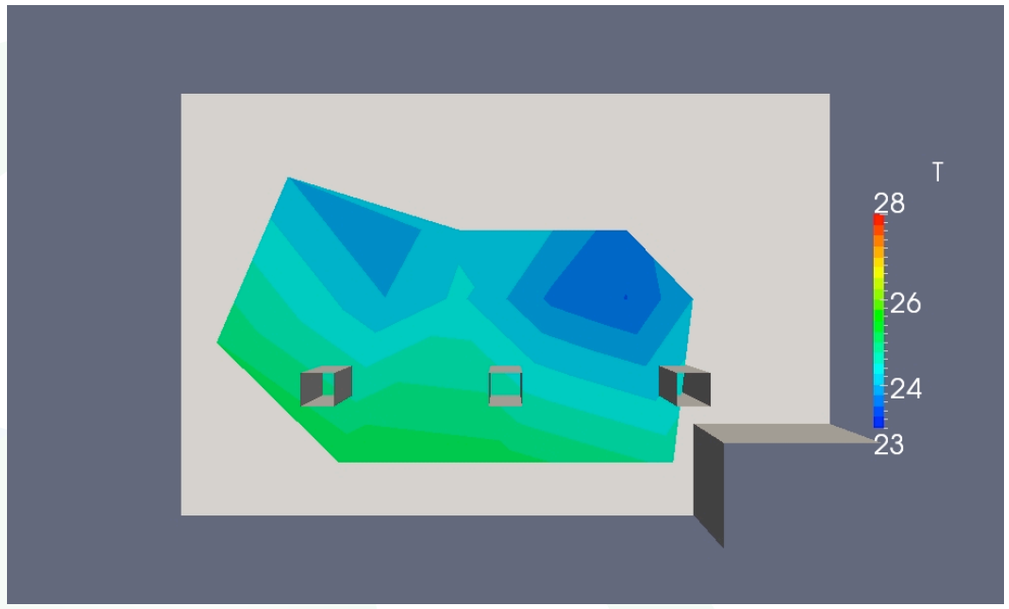


発湿条件

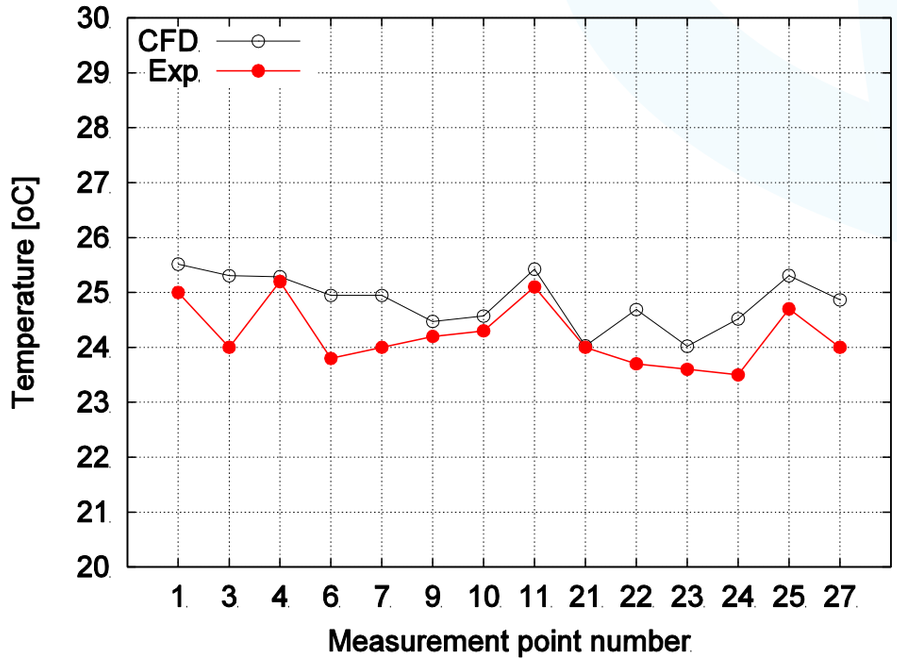
温度の実測結果との比較



CFD解析結果

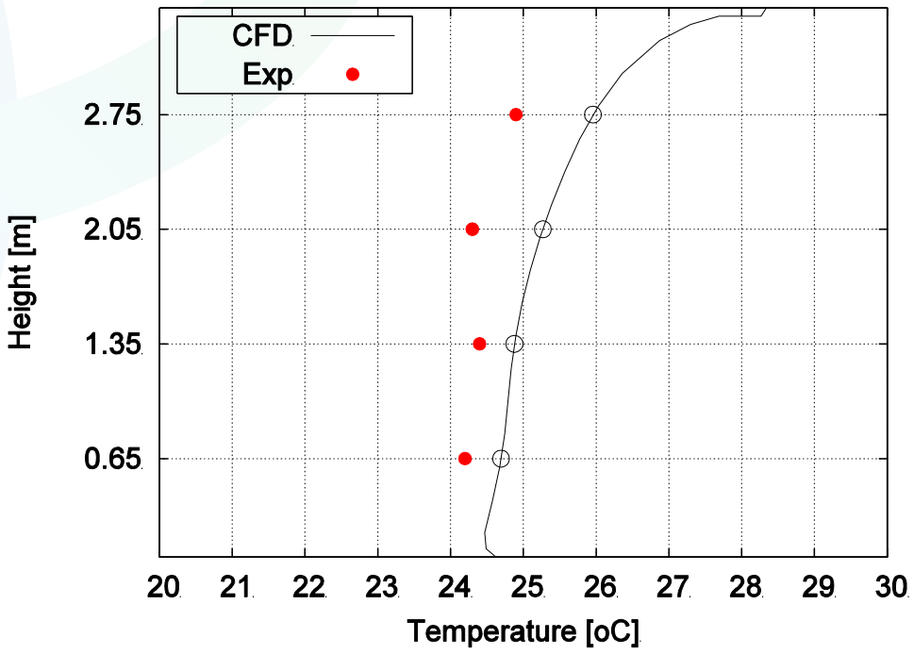


実測結果 (高さ0.8m)

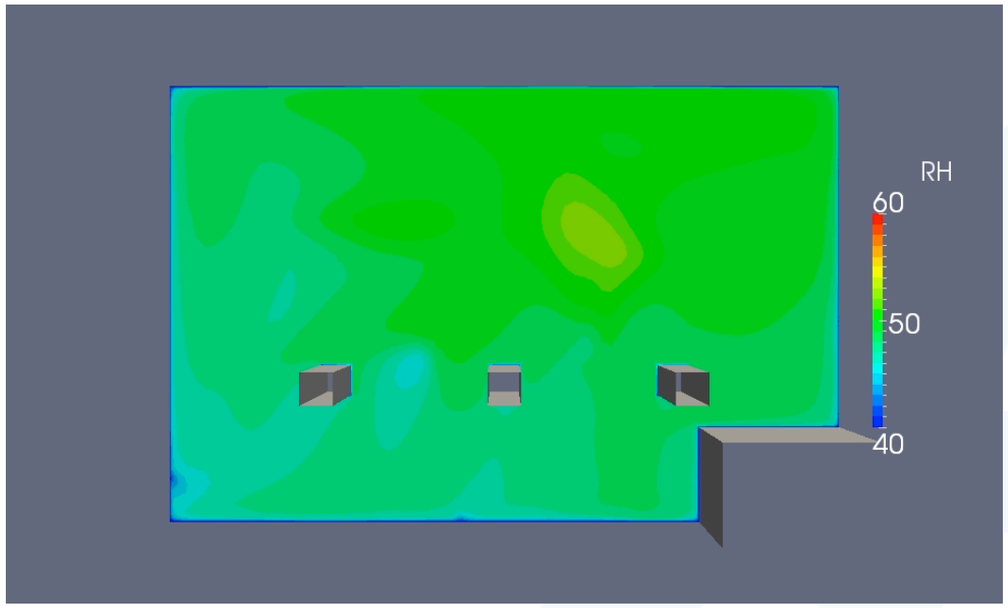


水平分布比較
(高さ0.8m)

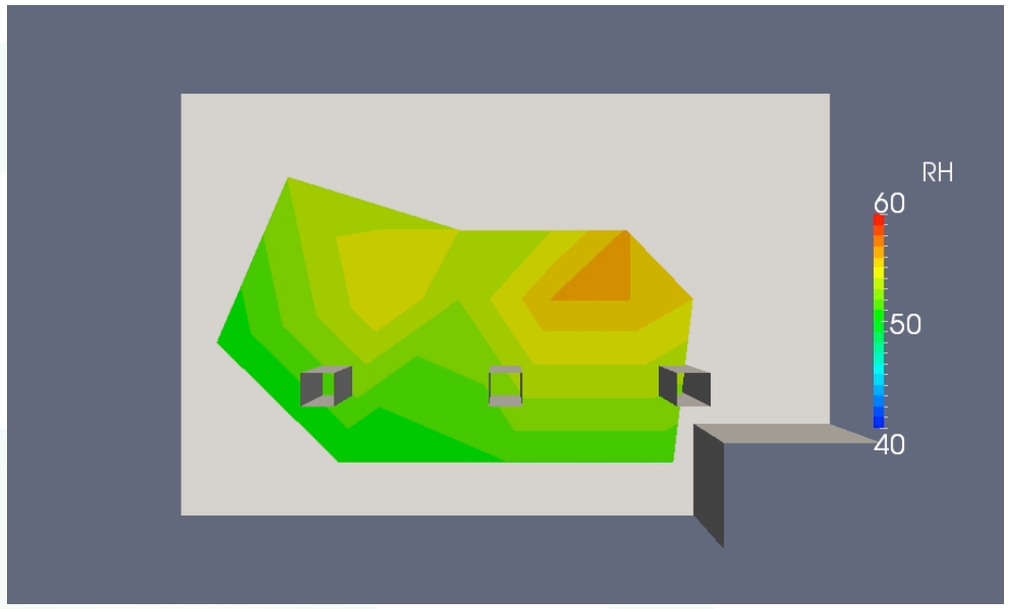
鉛直分布比較
(室中央)



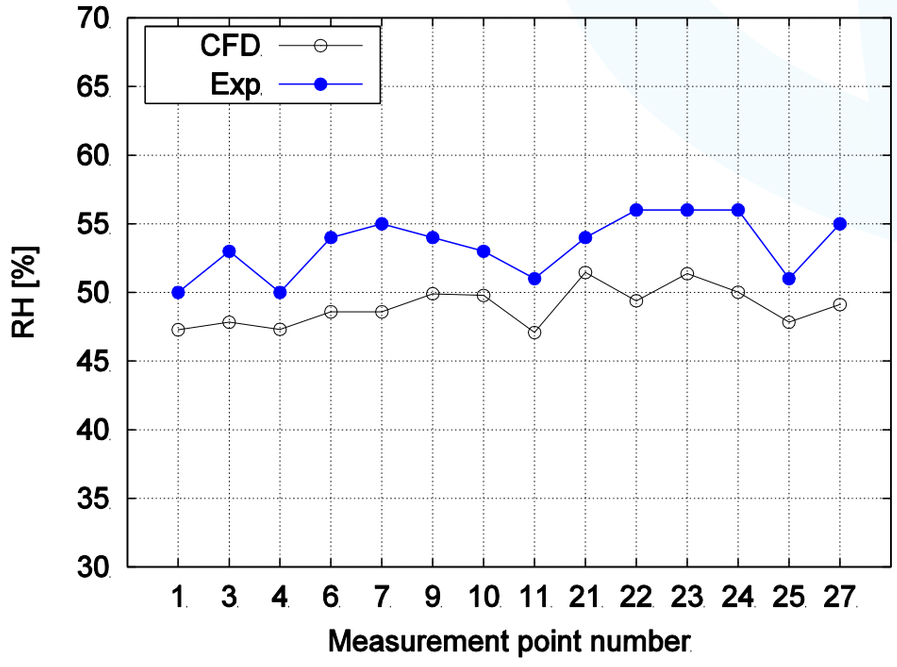
湿度の実測結果との比較



CFD解析結果

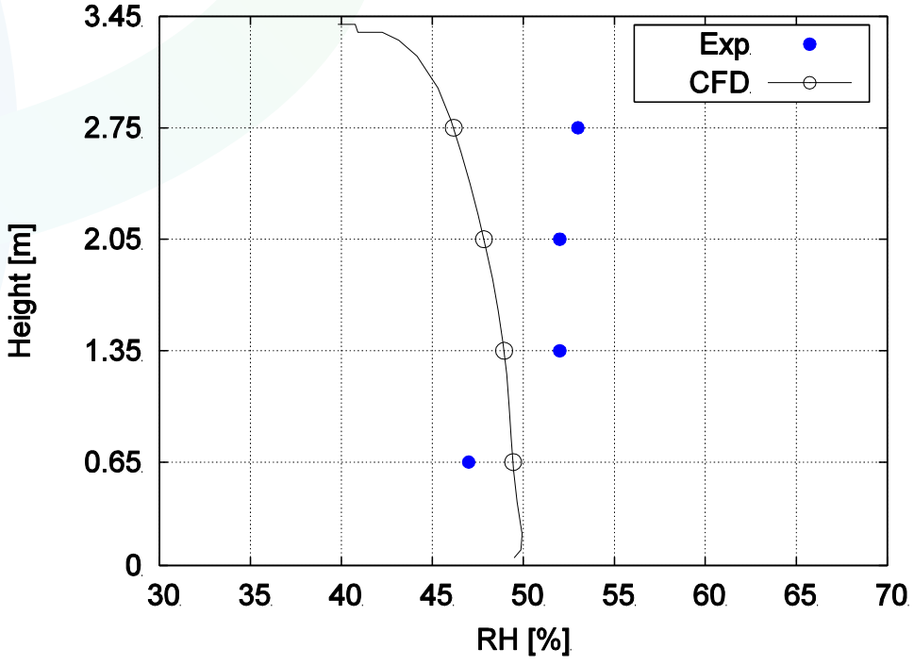


実測結果 (高さ0.8m)

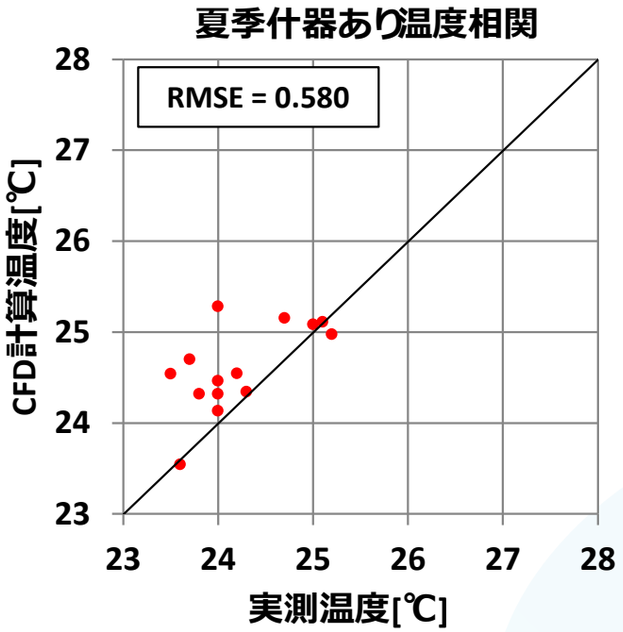


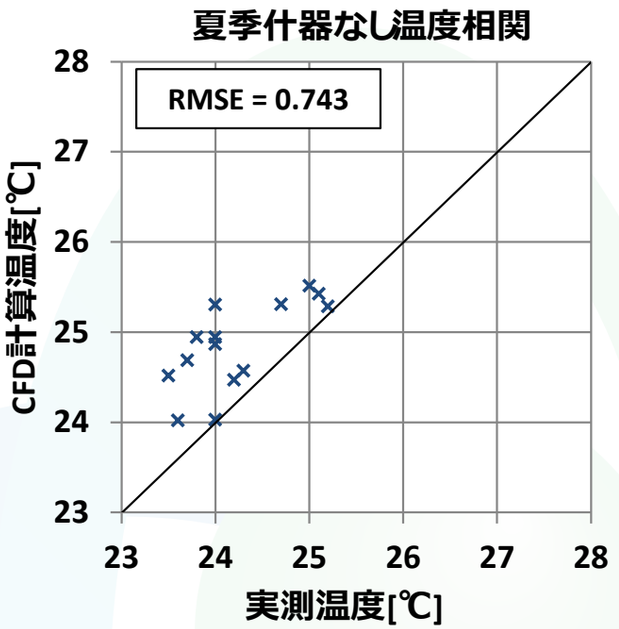
水平分布比較
(高さ0.8m)

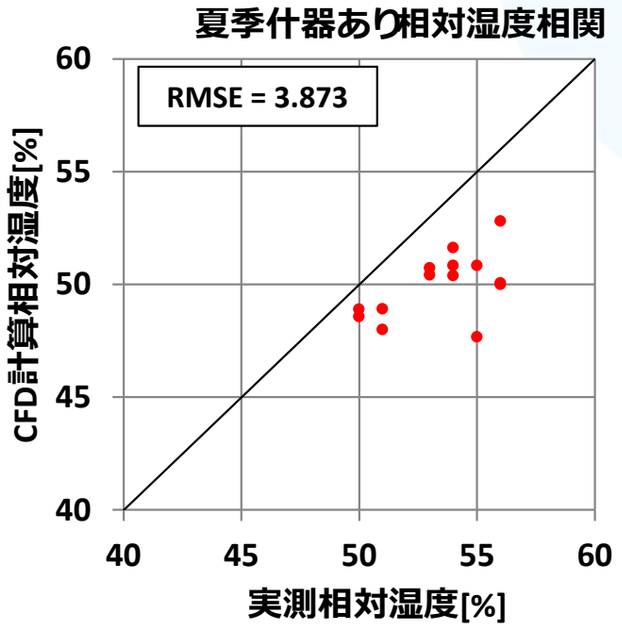
鉛直分布比較
(室中央)

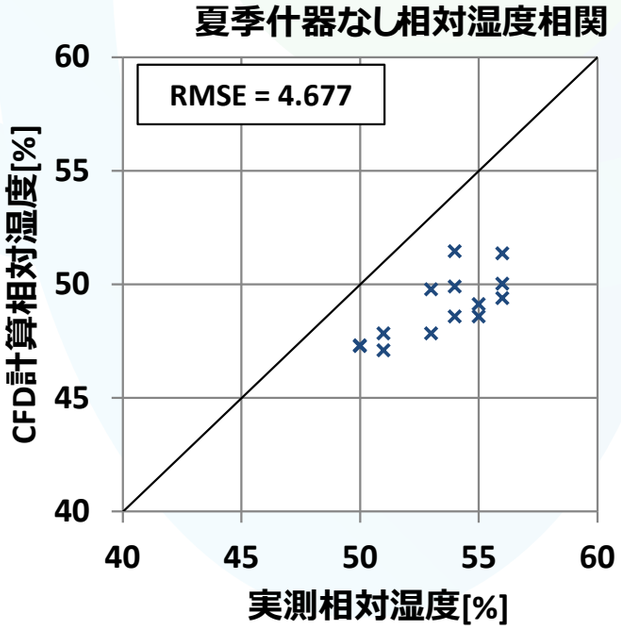


什器の有無による誤差の変化









解析結果と実測値との誤差(RMSE)

	温度[℃]	相対湿度[%]
什器あり	0.58	3.87
什器なし	0.74	4.68

発熱ソースの実装(Ver-2.1以前)

熱流体ソルバーを改造(これはVer-2.1以降でも変わらない)

- buoyantBoussinesqSimpleFoam：定常問題用
- buoyantBoussinesqPimpleFoam：非定常問題用

ソルバーの変更点

- createFields.H：温度輸送方程式のソース(ST)の場を読むコードを加える
- TEqn.H：温度輸送方程式にソース(ST)の項を加える

ソルバー実行前の準備

- setFieldsやfunkySetFieldsなどユーティリティを用いて、ST場に対して、発熱領域がある領域内に発熱密度を与える

ソルバーの改造(Ver-2.1以前)

createFields.H

```
Info<< "Reading field ST\n"
<< endl;
volScalarField ST
(
    IOobject
    (
        "ST",
        runTime.timeName(),
        mesh,
        IOobject::MUST_READ,
        IOobject::AUTO_WRITE
    ),
    mesh
);
```

TEqn.H

```
fvScalarMatrix TEqn
(
    fvm::div(phi, T)
  - fvm::Sp(fvc::div(phi), T)
  - fvm::laplacian(kappaEff, T)
  - ST // 温度輸送方程式にソース項を
加える
);
```

赤：追加

青：コメント

ソース項の準備(Ver-2.1以前)

system/setFieldsDict

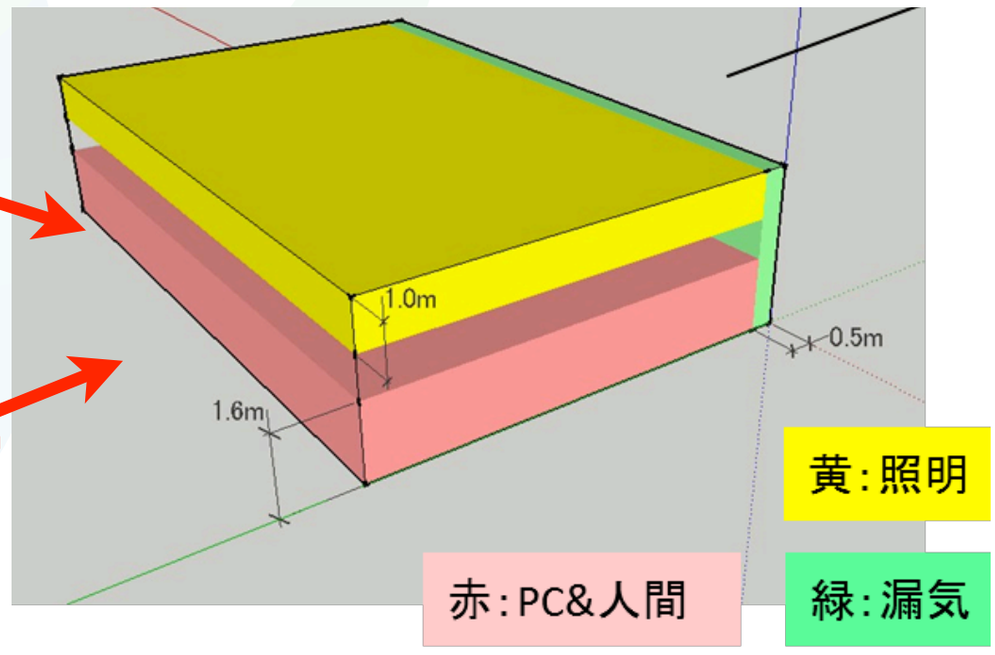
```
defaultFieldValues
(
    volScalarFieldValue ST 0
);                              デフォルト値

regions
(
    boxToCell //PC&人間。照明等も同様
    { //
      box    (0 0 0.5) (10 10 1.6);
                          発熱領域
      fieldValues
      (
          volScalarFieldValue ST 1
      );                  発熱密度/熱容量
    }
);
```

0/ST を用意して

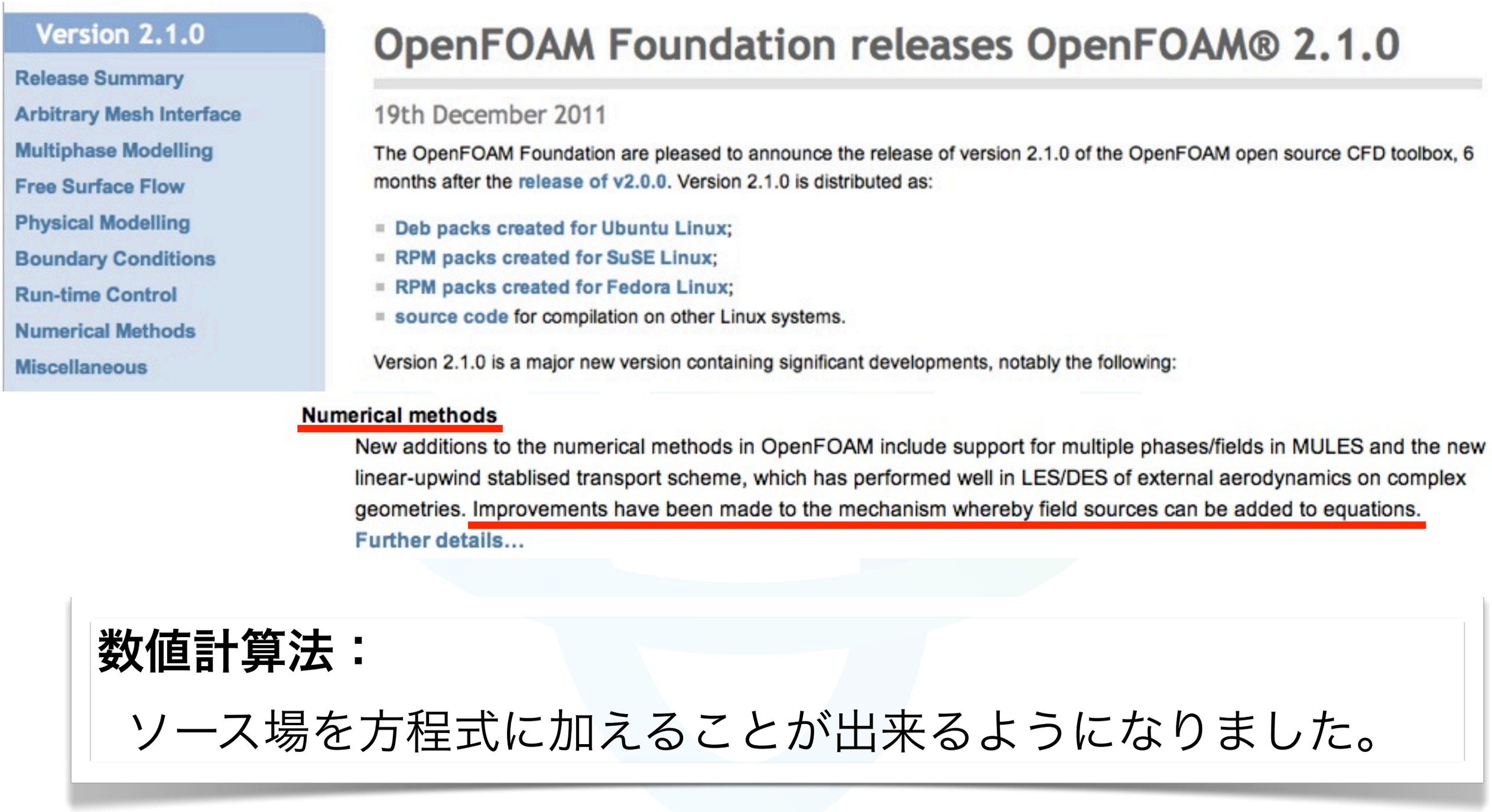
(境界条件は通常勾配0)、

以下を実行

```
setFields
```



Ver-2.1の新機能: Field sources



Version 2.1.0

Release Summary
Arbitrary Mesh Interface
Multiphase Modelling
Free Surface Flow
Physical Modelling
Boundary Conditions
Run-time Control
Numerical Methods
Miscellaneous

OpenFOAM Foundation releases OpenFOAM® 2.1.0

19th December 2011

The OpenFOAM Foundation are pleased to announce the release of version 2.1.0 of the OpenFOAM open source CFD toolbox, 6 months after the release of v2.0.0. Version 2.1.0 is distributed as:

- Deb packs created for Ubuntu Linux;
- RPM packs created for SuSE Linux;
- RPM packs created for Fedora Linux;
- source code for compilation on other Linux systems.

Version 2.1.0 is a major new version containing significant developments, notably the following:

Numerical methods

New additions to the numerical methods in OpenFOAM include support for multiple phases/fields in MULES and the new linear-upwind stablised transport scheme, which has performed well in LES/DES of external aerodynamics on complex geometries. Improvements have been made to the mechanism whereby field sources can be added to equations.

Further details...

数値計算法：

ソース場を方程式に加えることが出来るようになりました。

Ver-2.1の新機能: Field sources

Field sources

Improvements have been made to the mechanism in which sources can be added to fields in equations. Sources can now generally be applied using a *constant/sourcesProperties* dictionary with entries like the following.

```
massSource1
{
    type            scalarExplicitSource;
    active          true;
    timeStart       0.2;
    duration        2.0;
    selectionMode   points;
    points
    (
        (2.75 0.5 0)
    );

    scalarExplicitSourceCoeffs
    {
        volumeMode      absolute;
        injectionRate
        {
            rho         1e-4; // kg/s
            H2O         1e-4; // kg/s
        }
    }
}
```

simpleFoam/UEqn.H

```
tmp<fvVectorMatrix> UEqn
(
    fvm::div(phi, U)
  + turbulence->divDevReff(U)
  ==
    sources(U)
);

UEqn().relax();

sources.constrain(UEqn());

solve(UEqn() == -fvc::grad(p));
```

Ver-2.1の新機能: Field sources

tutorials/lagrangian/coalChemistryFoam/simplifiedSiwek/constant/sourcesProperties

```
source1
{
    type                    scalarExplicitSource;
//続く
```

typeの種類

場の型	Explicit	SetValue Source
scalar sphericalTensor symmTensor tensor vector		

pressureGradientExplicitSource
actuationDiskSource
radialActuationDiskSource

設定した平均風速を維持するよう圧力勾配を維持(channelFoam)

SetValue:
領域の値を規定

Source:
領域の発生量を規定

Ver-2.1の新機能: Field sources

```
//続き
    active          true; // 有効・無効
    timeStart       0.15; // 開始時間
    duration        0.2;  // 持続時間
    selectionMode   cellSet; // all,cellSet,cellZone,points
    cellSet         ignitionCells; // ソース領域のcellSet名

    scalarExplicitSourceCoeffs
    {
        volumeMode  absolute; // absolute,specific([X/m3])
        injectionRate         // 発生量
        {
            hs                20000; // 場の名前と発生量
        }
    }
}
```

発熱ソースの実装(Ver-2.1以降)

熱流体ソルバーを改造(熱流体ソルバーにはソース項が無いため)

- buoyantBoussinesqSimpleFoam：定常問題用
- buoyantBoussinesqPimpleFoam ：非定常問題用

ソルバーの変更点

- buoyantBoussinesq{S,P}impleFoam.C：IObasicSourceList クラス用のヘッダをインクルード
- createFields.H：ソース項IObasicSourceListの変数を宣言
- TEqn.H：温度輸送方程式にソース項を加える等

ソルバー実行前の準備

- constant/sourcesProperties を作成する

ソルバーの改造(Ver-2.1以降)

buoyantBoussinesqExplicitSourceSimpleFoam.C

```
#include "IObasicSourceList.H"
```

createFields.H

```
IObasicSourceList sources(mesh);
```

TEqn.H

```
fvScalarMatrix TEqn
(
    fvm::div(phi, T)
  - fvm::Sp(fvc::div(phi), T)
  - fvm::laplacian(kappaEff, T)
  - sources(T)
);
TEqn.relax();
sources.constrain(TEqn);
```

ソース項の準備(Ver-2.1以降)

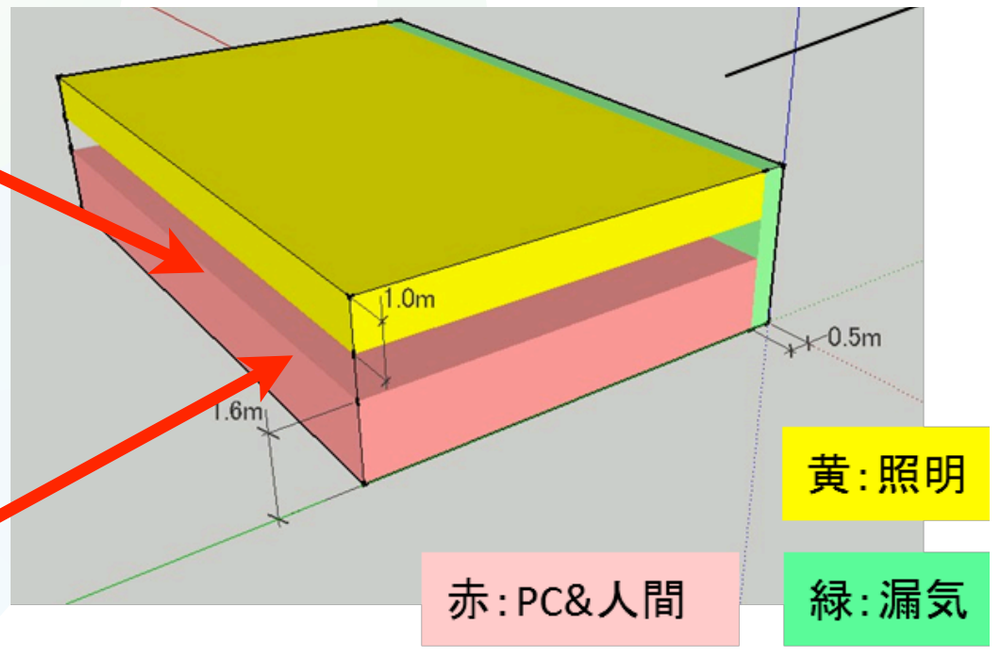
constant/sourcesProperties

```
Tsource
{
    type            scalarExplicitSource;
    active          true;
    timeStart       0;
    duration        1000000;
    selectionMode   cellZone;
    cellZone        all;

    scalarExplicitSourceCoeffs
    {
        volumeMode      absolute;
        injectionRate
        {
          T         1;
        }
    }
}
```

定常解：大きな値、非定常解：実際の時間

発熱領域

発熱密度/熱容量



Run-time code compilation

Run-time code compilation

A shortcut for the #codeStream **syntax** has been introduced to perform one line calculations. The shortcut uses the #calc directive, *e.g.* to do an algebraic operation between two scalars, a and b, the syntax is:

```
a 1.0;     // force a to be a scalar
b 3;
c #calc "$a/$b";
```

```
    scalarExplicitSourceCoeffs
    {
        volumeMode      absolute;
        injectionRate
        {
          T             #calc "1200/(1006.0*1.184)"; // 1200[W]
        }
    }
}
```

実行時にコードがコンパイルされて値が設定される

Ver-2.1の新機能ソース場の特徴

利点

- ソース場のファイルを用意する必要がない
- 予めsetFields等のユーティリティを用いてソース場の分布を設定しなくても良く、ソルバーで分布を設定できる
- ソースが有効となる開始時刻や継続時間が設定できる
- 体積単位の発生量ではなく、発生量が指定できる
- 併記圧力勾配制御など特殊なソースが指定できる

欠点

- ソース機構が組み込まれていないソルバーは改造が必要
- 複雑なソースの空間分布を指定する場合には、ソース場タイプの改造をしてfunkySetFields等で設定が必要
- 現時点では時系列変化の設定が階段状にしかできない