برنامه نویسی در محیط نرم افزار اپن فوم (محیط متخلخل)

- توصیف حلگر **laplacianFoam**
- ایجاد حلگر **darcyFoam** برای حل جریان در محیط متخلخل
- انتقال حرارت در محیط متخلخل به صورت تک معادله ای و دو معادله ای

۱-۱- حلگر laplacianFoam

در حقیقت یکی از عوامل قدرتمند بودن نرم افزار اپن فوم آسان بودن برنامه نویسی معادلات با مشتقات جزیی در آن میباشد:

$$\frac{\partial \rho \mathbf{U}}{\partial t} + \nabla \cdot \phi \mathbf{U} - \nabla \cdot \mu \nabla \mathbf{U} = -\nabla p$$

```
solve
(
      fvm::ddt(rho,U)
    + fvm::div(phi,U)
    - fvm::laplacian(mu,U)
      ==
    - fvc::grad(p)
);
```

- در این قسمت به بررسی حلگر laplacianFoam میپردازیم:

پشت پرده laplacianFoam :

laplacianFoam.C:

$ gedit laplacianFoam.C

```
#include "fvCFD.H"
#include "simpleControl.H"

// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

int main(int argc, char *argv[])
{
    #include "setRootCase.H"

    #include "createTime.H"
    #include "createMesh.H"
    #include "createFields.H"

    simpleControl simple(mesh);

    // * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

    Info<< "\nCalculating temperature distribution\n" << endl;

    while (simple.loop())
    {
        Info<< "Time = " << runTime.timeName() << nl << endl;

        while (simple.correctNonOrthogonal())
        {
            solve
            (
                fvm::ddt(T) - fvm::laplacian(DT, T)
            );
        }

        #include "write.H"

        Info<< "ExecutionTime = " << runTime.elapsedCpuTime() << " s"
            << "  ClockTime = " << runTime.elapsedClockTime() << " s"
            << nl << endl;
    }

    Info<< "End\n" << endl;

    return 0;
}
```

فراخوانی کتابخانه های اپن فوم

$$\frac{\partial T}{\partial t} = \nabla \cdot (D_T \nabla T)$$

ایجاد ماتریس:

fvm: ترم های ضمنی (implicit)

fVC: ترم های صریح (explicit)

متغیرهای T و DT در createField.H تعریف شده اند.

$ gedit createFields.H

```
Info<< "Reading field T\n" << endl;

volScalarField T
(
    IOobject
    (
        "T",
        runTime.timeName(),
        mesh,
        IOobject::MUST_READ,
        IOobject::AUTO_WRITE
    ),
    mesh
);

Info<< "Reading transportProperties\n" << endl;

IOdictionary transportProperties
(
    IOobject
    (
        "transportProperties",
        runTime.constant(),
        mesh,
        IOobject::MUST_READ_IF_MODIFIED,
        IOobject::NO_WRITE
    )
);

Info<< "Reading diffusivity DT\n" << endl;

dimensionedScalar DT
(
    transportProperties.lookup("DT")
);
```

میدان دمایی T تعریف شده است به عنوان یک نمونه از شی volScalarField.

دما یک میدان اسکالر است که در مرکز سلول تعریف شده است.

این متغیر باید در زمان اولیه خوانده شود.

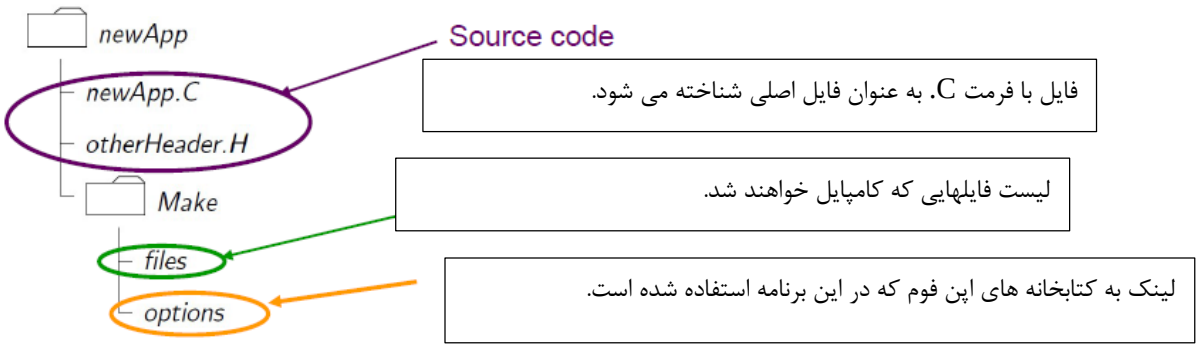
ابعاد (واحد) آن در 0/T تعریف شده است.

T در هر گام زمانی در فولدر مورد نظر باید نوشته شود (runtime.timeName()).

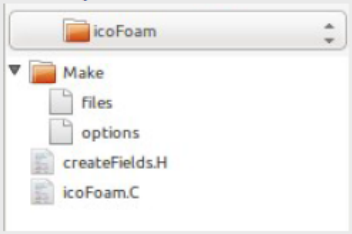
این شی شامل شرایط مرزی میشود که در پوشه 0/T مشخص شده اند.

دیکشنری transportProperties از فایل ورودی constant/transportProperties لود میشود.

تعریف متغیر DT.ابعاد و اندازه آن تعریف شده اند در فایل ورودی constant/transportProperties.



Example of the *icoFoam* solver




```
$ cd    $FOAM_APP/solvers/incompressible/icoFoam
$ ls
```

ایجاد دایرکتوری (پوشه) برای برنامه های شخصی ایجاد شده(این مرحله فقط یک بار لازم به اجرا است).

```
$ mkdir   -p    $WM_PROJECT_USER_DIR/applications/solvers/
```


۱-۲- ایجاد حلگر دارسی (Darcy solver)

هدف : توسعه دادن یک برنامه که جریان در یک محیط متخلخل کاملا اشباع شده را با استفاده از قانون دارسی حل میکند.

$$\nabla.\mathbf{U} = 0 \qquad (۱-۱)$$

$$\mathbf{U} = -\frac{k}{\mu}\nabla p \qquad (۱-۲)$$

چگونه این مساله ریاضی را حل کنیم؟

با ترکیب کردن معادلات ۱ و ۲ معادله نفوذ برای میدان فشار بدست می آید:

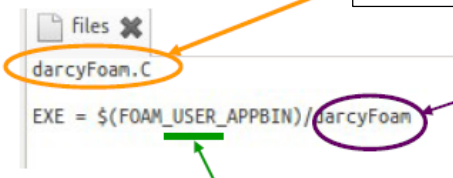
$$\nabla.\frac{k}{\mu}\nabla p = 0 \qquad (۱-۳)$$

ما قصد داریم که حلگر خود را بر روی laplacianFoam بنویسیم:

```
$ cd    $WM_PROJECT_USER_DIR/applications/solvers/
$ cp    –r    $FOAM_APP/solvers/basic/laplacianFoam darcyFoam
```

پس از اینکه حلگر laplacianFoam در دایرکتوری مورد نظر کپی شد،ما فیل اصلی را تغییر نام میدهیم و سپس make/files را edit میکنیم.

```
$ cd     darcyFoam
$ mv     laplacianFoam.C     darcyFoam.C
$ gedit   Make/files
```



لیست فایلهایی که باید کامپایل شوند(فقط فایل های .C)

نام برنامه جدید

در اینجا USER را اضافه میکنیم برای اینکه مشخص کنیم که برنامه جدید در پوشه USER کامپایل خواهد شد(بدون USER کامپایل کردن با error مواجه خواهد شد).

اکنون ما با دستور wclean اثرات کامپایل کردن قبلی را پاک میکنیم و برنامه جدید را با wmake مجددا کامپایل میکنیم.

```
$ wclean
$ wmake
```

در این مرحله ما یک حلگر جدید به نام darcyFoam داریم که دقیقا یک کپی از laplacianFoam است.

توصیه میشود که غالبا در طول مراحل برنامه نویسی در اپن فوم از wmake استفاده شود.(بدون wclean)