



# Применение СПО для решения задач аэро- и газовой динамики

## Семинар 2. Запуск задачи обтекания цилиндра

Преподаватель: Романова Дарья Игоревна

Институт системного программирования им. В.П. Иванникова РАН, Москва

2023

# Схема задачи обтекания квадратного цилиндра

Математическая модель:

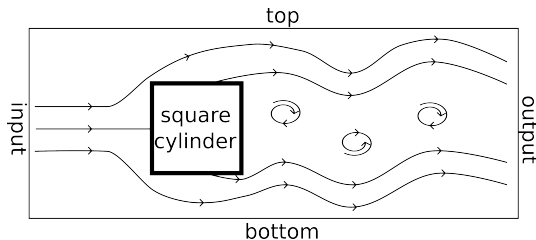
$$\nabla \cdot \mathbf{U} = 0,$$

$$\frac{\partial \mathbf{U}}{\partial t} + \nabla \cdot \mathbf{U} \mathbf{U} = -\nabla p + \nabla \cdot \boldsymbol{\tau},$$

где  $\mathbf{U}$  — скорость,  $\boldsymbol{\tau} = 2\nu \mathbf{s}$  — тензор вязких напряжений,  $\mathbf{s} = 0.5(\nabla \mathbf{U} + (\nabla \mathbf{U})^T)$  — тензор скоростей деформаций,  $\nu$  — кинематическая вязкость,  $p$  — давление.

Геометрия расчётной области:

- ▶ Сторона квадрата  $D = 1$
- ▶ Высота расчётной области  $H = 30D$
- ▶ Длина расчётной области  $L = 80D$
- ▶ Толщина  $B = D$



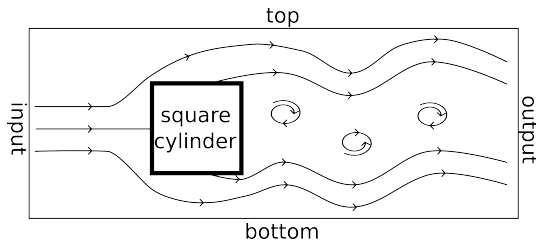
# Начальные и граничные условия

Течение ламинарное:

$$Re = \frac{UD}{\nu} = \frac{1 \cdot 1}{0.01} = 100$$

Начальные условия

$\mathbf{U} = 1$	$p = 0$	$\nu = 0.01$
------------------	---------	--------------



Граничные условия

Граница	$\mathbf{U}$	$p$
inlet	Fixed value $\mathbf{U} = 1$	Zero gradient
outlet	Zero gradient	Fixed value $p = 0$
top/bottom/front/back	Zero gradient	Zero gradient
cylinder	Fixed value $\mathbf{U} = 0$	Zero gradient

## Скачиваем кейс

Кейс лежит на GitHub

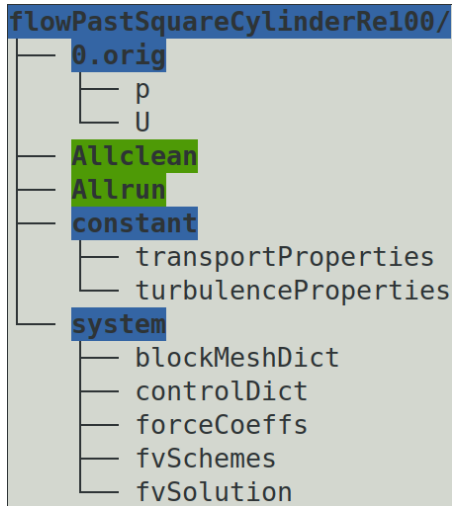
<https://github.com/RomanovaDI/freeSoftwareTrainingCourse>

Скачать можно по ссылке

<https://github.com/RomanovaDI/freeSoftwareTrainingCourse/archive/refs/heads/main.zip>

# Структура кейса

- ▶ 0.orig — каталог с начальными условиями
  - ▶ p — начальные условия на давление
  - ▶ U — начальные условия на скорость
- ▶ Allclean — скрипт очистки директории кейса
- ▶ Allrun — скрипт запуска расчёта
- ▶ constant — каталог с константными параметрами
  - ▶ transportProperties — реологические свойства
  - ▶ turbulenceProperties — параметры турбулентности
- ▶ system — каталог с параметрами расчёта задачи
  - ▶ blockMeshDict — описание сетки
  - ▶ controlDict — параметры управления расчётом
  - ▶ forceCoeffs — описание расчета сил на цилиндре
  - ▶ fvSchemes — схемы аппроксимации
  - ▶ fvSolution — параметры решателя



# Скрипты выполнения

Allclean

```
0 #!/bin/sh
1 cd "${0%/*}" || exit # Run from this directory
2 . ${WM_PROJECT_DIR:?}/bin/tools/CleanFunctions # Tutorial clean functions
3 #-----
4
5 cleanCase0 # Чистим директорию кейса от старых расчётов
6
7 #-----
```

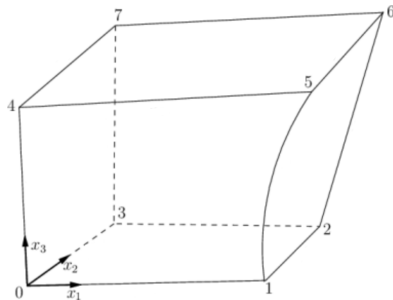
# Скрипты выполнения

Allrun

```
0 #!/bin/sh
1 cd "${0%/*}" || exit # Run from this directory
2 . ${WM_PROJECT_DIR:?}/bin/tools/RunFunctions # Tutorial run functions
3 #-----
4
5 touch flowPastSquareCylinderRe100.foam # Создаем файл для визуализации
6 runApplication blockMesh # Создаем сетку
7 restore0dir # Создаем нулевой момент
8 runApplication $(getApplication) # Запускаем расчёт
9
10 #-----
```

## Сетка. Куб. blockMeshDict

- ▶ Система координат должна быть правосторонней.
- ▶ Вершины объявляются в соответствии с выбранной системой координат.
- ▶ При объявлении блока порядок перечисления вершин блока строго определён и показан на рисунке. Он зависит от системы координат.
- ▶ Объявляются внешние грани расчетной области, нормаль должна смотреть наружу (определяется по принципу правой руки).



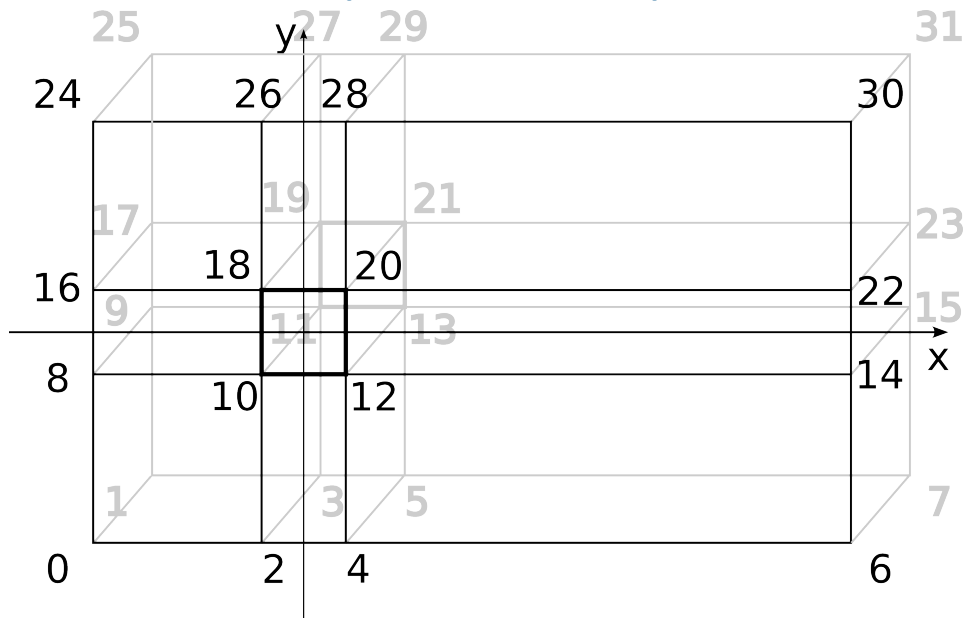


## Сетка. Куб. blockMeshDict

```
0 scale    0.1;
1 vertices
2 (
3     (0 0 0)
4     (1 0 0)
5     (1 1 0)
6     (0 1 0)
7     (0 0 0.1)
8     (1 0 0.1)
9     (1 1 0.1)
10    (0 1 0.1)
11 );
12 blocks
13 (
14     hex (0 1 2 3 4 5 6 7) (20 20 1) simpleGrading (1 1 1)
15 );
16 edges
17 ();
```

```
18 boundary
19 (
20     walls {
21         type wall;
22         faces
23         (
24             (4 5 6 7) // top
25             (0 3 2 1) // bottom
26             (0 4 7 3) // left
27             (2 6 5 1) // right
28         ); }
29     frontAndBack {
30         type empty;
31         faces
32         (
33             (1 5 4 0) // front
34             (3 7 6 2) // back
35         );}
36 );
```

## Сетка. Обтекание квадратного цилиндра. blockMeshDict



## Сетка. blockMeshDict

```
0 scale      1.0;
1
2 DHalf      0.5;
3 HHalf      15.5;
4 LHalf      15.5;
5 LRHalf     65.5;
6 BHalf      0.5;
7 nB         1;
8 nD         5;
9 nHHalf     75;
10 nLHalf     75;
11 nLRHalf   325;
12
13 vertices
14 (
15     (-$LHalf -$HHalf  $BHalf) // 0
16     (-$LHalf -$HHalf -$BHalf) // 1
17     (-$DHalf  -$HHalf  $BHalf) // 2
```

```
18      (-$DHalf -$HHalf -$BHalf) // 3
19      ( $DHalf -$HHalf  $BHalf) // 4
20      ( $DHalf -$HHalf -$BHalf) // 5
21      ( $LRHalf -$HHalf  $BHalf) // 6
22      ( $LRHalf -$HHalf -$BHalf) // 7
23      (-$LLHalf -$DHalf  $BHalf) // 8
24      (-$LLHalf -$DHalf -$BHalf) // 9
25      (-$DHalf -$DHalf  $BHalf) // 10
26      (-$DHalf -$DHalf -$BHalf) // 11
27      ( $DHalf -$DHalf  $BHalf) // 12
28      ( $DHalf -$DHalf -$BHalf) // 13
29      ( $LRHalf -$DHalf  $BHalf) // 14
30      ( $LRHalf -$DHalf -$BHalf) // 15
31      (-$LLHalf  $DHalf  $BHalf) // 16
32      (-$LLHalf  $DHalf -$BHalf) // 17
33      (-$DHalf  $DHalf  $BHalf) // 18
34      (-$DHalf  $DHalf -$BHalf) // 19
35      ( $DHalf  $DHalf  $BHalf) // 20
36      ( $DHalf  $DHalf -$BHalf) // 21
37      ( $LRHalf  $DHalf  $BHalf) // 22
```

```

38      ( $LRHalf $DHalf -$BHalf) // 23
39      (-$LLHalf $HHalf $BHalf) // 24
40      (-$LLHalf $HHalf -$BHalf) // 25
41      (-$DHalf $HHalf $BHalf) // 26
42      (-$DHalf $HHalf -$BHalf) // 27
43      ( $DHalf $HHalf $BHalf) // 28
44      ( $DHalf $HHalf -$BHalf) // 29
45      ( $LRHalf $HHalf $BHalf) // 30
46      ( $LRHalf $HHalf -$BHalf) // 31
47 );
48
49 blocks
50 (
51     hex (1 3 11 9 0 2 10 8) ($nLLHalf $nHHalf $nB) simpleGrading (1 1
52     1)
53     hex (3 5 13 11 2 4 12 10) ($nD $nHHalf $nB) simpleGrading (1 1 1)
54     hex (5 7 15 13 4 6 14 12) ($nLRHalf $nHHalf $nB) simpleGrading (1
55     1 1)
56     hex (9 11 19 17 8 10 18 16) ($nLLHalf $nD $nB) simpleGrading (1 1
57     1)

```

```
55     hex (13 15 23 21 12 14 22 20) ($nLRHalf $nD $nB) simpleGrading (1
56     1 1)
57     hex (17 19 27 25 16 18 26 24) ($nLLHalf $nHHalf $nB) simpleGrading
58     (1 1 1)
59     hex (19 21 29 27 18 20 28 26) ($nD $nHHalf $nB) simpleGrading (1 1
60     1)
61     hex (21 23 31 29 20 22 30 28) ($nLRHalf $nHHalf $nB) simpleGrading
62     (1 1 1)
63 );
64
65 edges
66 (
67     inlet
68     {
69         type patch;
70         faces
```

```
71      (  
72          (0 8 9 1)  
73          (8 16 17 9)  
74      (16 24 25 17)  
75      );  
76  }  
77  outlet  
78  {  
79      type patch;  
80      faces  
81      (  
82          (6 7 15 14)  
83          (14 15 23 22)  
84      (22 23 31 30)  
85      );  
86  }  
87  hole  
88  {  
89      type wall;  
90      faces
```



```
91         (  
92             (10 11 19 18)  
93             (18 19 21 20)  
94             (12 20 21 13)  
95             (10 12 13 11)  
96         );  
97     }  
98     topAndBottom  
99     {  
100         type wall;  
101         faces  
102         (  
103             (24 26 27 25)  
104             (26 28 29 27)  
105             (28 30 31 29)  
106             (0 1 3 2)  
107             (2 3 5 4)  
108             (4 5 7 6)  
109         );  
110     }
```

```
111 );  
112  
113 mergePatchPairs  
114 (  
115 );  
116  
117  
118 // ***** //
```

## Начальные условия. p

```
0 FoamFile
1 {
2     version      2.0;
3     format       ascii;
4     class        volScalarField;
5     object       p;
6 }
7 dimensions      [0 2 -2 0 0 0 0];
8 internalField    uniform 0;
9 boundaryField
10 {
11     hole
12     {
13         type      zeroGradient;
14     }
15     inlet
16     {
17         type      zeroGradient;
```

```
18     }
19     outlet
20     {
21         type          fixedValue;
22         value          uniform 0;
23     }
24     topAndBottom
25     {
26         type          zeroGradient;
27     }
28     defaultFaces
29     {
30         type          empty;
31     }
32 }
```

## Начальные условия. U

```
0 FoamFile
1 {
2     version      2.0;
3     format       ascii;
4     class        volVectorField;
5     object       U;
6 }
7 dimensions      [0 1 -1 0 0 0 0];
8 internalField    uniform (1 0 0);
9 boundaryField
10 {
11     hole
12     {
13         type      fixedValue;
14         value      uniform (0 0 0);
15     }
16     inlet
17     {
```

```
18         type          fixedValue;
19         value          uniform (1 0 0);
20     }
21     outlet
22     {
23         type          zeroGradient;
24     }
25     topAndBottom
26     {
27         type          zeroGradient;
28     }
29     defaultFaces
30     {
31         type          empty;
32     }
33 }
```

## Константные параметры. transportProperties

```
0 FoamFile
1 {
2     version      2.0;
3     format       ascii;
4     class        dictionary;
5     location     "constant";
6     object       transportProperties;
7 }
8 // * * * * *
9
10 transportModel  Newtonian;
11
12 nu              0.01;
13
14 // ***** //
```

## Константные параметры. turbulenceProperties

```
0 FoamFile
1 {
2     version      2.0;
3     format       ascii;
4     class        dictionary;
5     location     "constant";
6     object       turbulenceProperties;
7 }
8 // * * * * *
9
10 simulationType laminar;
11
12 // ***** //
```



## Управление расчетом. controlDict

```
0 application      pimpleFoam;
1 startFrom        latestTime;
2 startTime        0;
3 stopAt           endTime;
4 endTime          10;
5 deltaT           0.004;
6 writeControl      timeStep;
7 writeInterval     25;
8 writeFormat       binary;
9 writePrecision    8;
10 writeCompression off;
11 timeFormat        general;
12 timePrecision     6;
13 runTimeModifiable true;
14
15
16
17
```

```

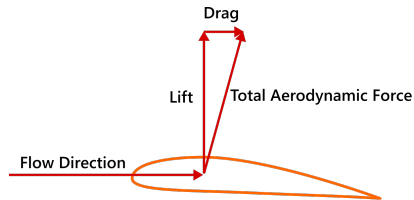
18 functions
19 {
20     forces
21     {
22         type            forces;
23         libs            (forces);
24         writeControl    writeTime;
25
26         patches        (hole);
27         // Centre of rotation for moment calculations
28         CofR           (0 0 0);
29         writeFields    yes;
30
31         rho            rhoInf;
32         rhoInf         1.0;
33     }
34
35     #include "forceCoeffs"
36 }

```

# Аэродинамическая сила.

Полная аэродинамическая сила делится на подъёмную силу и силу лобового сопротивления:

$$\mathbf{F} = \mathbf{F}_L + \mathbf{F}_D.$$



Расчёт полной аэродинамической силы происходит через интеграл по поверхности тела от давления и вязких напряжений на поверхности тела:

$$\mathbf{F} = \mathbf{F}_p + \mathbf{F}_v,$$

$$\mathbf{F}_p = \oint_S \rho(p - p_{ref}) d\mathbf{S}, \quad \mathbf{F}_v = \oint_S \boldsymbol{\tau}^{(d)} \cdot d\mathbf{S},$$

где  $p$  — давление,  $p_{ref}$  — давление на бесконечности,  $\boldsymbol{\tau}^{(d)}$  — девиатор тензора вязких напряжений.

## Коэффициент аэродинамического сопротивления и коэффициент подъёмной силы.

Коэффициент подъёмной силы — безразмерная величина, характеризующая подъёмную силу крыла определённого профиля при известном угле атаки.

$$C_L = \frac{F_L}{0.5U^2S},$$

где  $S$  — характерная площадь, чаще всего площадь миделевого сечения (наибольшее по площади поперечное сечение тела, движущегося в воде или воздухе),  $U$  — скорость набегающего потока в случае, если используется связанная система координат.

Коэффициент аэродинамического сопротивления — коэффициент, характеризующий способность тела преодолевать аэродинамическое сопротивление воздуха.

$$C_D = \frac{F_D}{0.5U^2S}.$$

## $C_D$ и $C_L$ . forceCoeffs

```
0 forceCoeffs1
1 {
2     type                forceCoeffs;
3     libs                (forces);
4     writeControl        timeStep;
5     timeInterval        1;
6     log                 yes;
7     patches              (hole);
8     rho                 rhoInf;          // Indicates incompressible
9     rhoInf              1.0;           // Required when rho = rhoInf
10    liftDir              (0 1 0);
11    dragDir              (1 0 0);
12    CofR                 (0 0 0);      // Axle midpoint on ground
13    pitchAxis            (0 0 1);
14    magUInf              1;
15    lRef                 1;             // Wheelbase length
16    Aref                 1;             // Estimated
17 }
```

СПАСИБО ЗА ВНИМАНИЕ!

# Список литературы