

Решение задачи: «Моделирование течения жидкости в трещинном/поровом коллекторе» в GNU Octave (Matlab)

Цель работы: ознакомление с средствами и методическими приемами выполнения научных расчетов гидродинамики в средах научного программирования (на примере GNU Octave (Matlab)) для построения двумерной модели течения нефти в породе.

Задача:

- 1) Внимательно изучить теоретические основы рассматриваемого метода CFD, приводимые в методическом пособии, *порядок перехода от ситуации реального мира к модельной*;
- 2) прочитать комментарии исходного кода, запустить исходный код на выполнение с имеющимися параметрами;
- 3) изучить параметры своего варианта работы №6, выполнить необходимые преобразование данных для применение их в модели, правильно выбрать масштаб домена и его размеры;
- 4) Применить численную модель к ситуации своего варианта, ознакомиться с результатами, проинтерпретировать, ответить на контрольные вопросы.

Программное обеспечение. Для выполнения настоящей работы используется бесплатная программа GNU Octave (<http://www.octave.org>), имеющая существенную совместимость с Matlab. Предоставляются исходные файлы – LMB-скрипт Matlab (авторы Sukor, M.C., Thorne, D.T., FIU, 2007).

Возможности модели

Предлагаемая модель LBM позволяет обнаружить вероятные пути протекания жидкости через каналы сложной морфологии, выявить зоны максимальных и минимальных скоростей течения.

Описание модели

Для моделирования течения жидкости в тонких цилиндрических трубках, трубах небольшого сечения и капиллярных каналах, описываемых [законом Пуазейля](#) применимы модели так называемых решеток Больцмана, названных так по имени Людвига Больцмана, основателя молекулярно-кинетической теории.

Метод использует дискретное уравнение Больцмана (1):

$$f_a(\mathbf{x} + \mathbf{e}_a \Delta t, t + \Delta t) = f_a(\mathbf{x}, t) - \frac{[f_a(\mathbf{x}, t) - f_a^{eq}(\mathbf{x}, t)]}{\tau} \quad (1),$$

где \mathbf{e}_a – микроскопическая скорость (скорость в определенном направлении a), Δt – шаг времени, f_a – массовая плотность частиц, движущаяся от одного узла решетки к другому в направлении a , f_a^{eq} – массовая плотность частиц, движущаяся в направлении a в момент равновесия, \mathbf{x} – вектор направления, τ – время релаксации.

Работа модели разделена на два повторяющихся шага: шаг перемещения ($f_a(\mathbf{x} + \mathbf{e}_a \Delta t, t + \Delta t) = f_a(\mathbf{x}, t)$) и шаг столкновения ($(f_a(\mathbf{x}, t) - f_a^{eq}(\mathbf{x}, t)) / \tau$). Эта модель

может приняться для моделирования течения несжимаемой жидкости с малыми числами Рейнольдса ($Re \ll 1$).

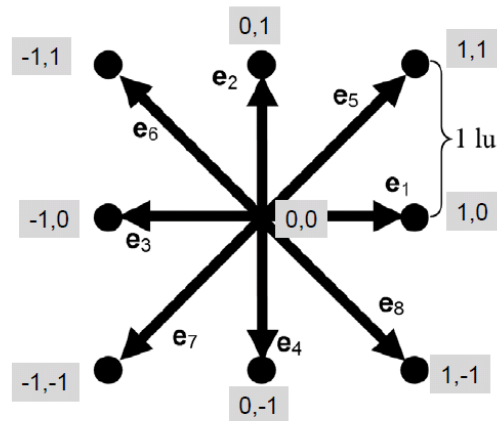


Рис. 1. Модель D2Q9, x, y – компоненты скорости

Значения параметров модели по умолчанию: время релаксации $\tau=1$, микроскопические скорости \mathbf{e}_a принимают значения:

$$\mathbf{e}_a = \begin{cases} (0,0) \\ (\pm 1, 0) \\ (0, \pm 1) \\ (\pm 1, \pm 1) \end{cases}$$

Вязкость и плотность жидкости выражаются в модели с помощью времени релаксации (2):

$$\tau = \mu \frac{d\mathbf{u}}{dx} = \mu \frac{\rho}{\rho} \frac{d\mathbf{u}}{dx} = \nu \frac{d\mathbf{p}}{dx} \quad (2),$$

где μ – динамическая вязкость, ν – кинематическая вязкость, ρ – плотность, \mathbf{p} – импульс ($\rho \cdot \mathbf{u}$).

В простейшей двумерной модели D2Q9 кинематическая вязкость и время релаксации связаны выражением (3):

$$\nu = \frac{1}{3} \left(\tau - \frac{1}{2} \right) \quad (3)$$

Отсюда (4):

$$\tau = 3\nu + \frac{1}{2} \quad (4)$$

Заданное число Рейнольдса потока позволяет рассчитать среднюю скорость в модели в соответствии с соотношением (5):

$$Re = \frac{u \cdot 2a}{\nu} \quad (4)$$

, где u – средняя макроскопическая скорость потока, $2a$ – ширина канала, к котором осуществляется протекание [в единицах решетки]. Таким образом, для заданного числа Рейнольдса Re , средняя скорость потока будет равна (5):

$$u = \frac{Re \cdot \nu}{2a} \quad (5)$$

Гравитационное ускорение g в модели рассчитывается по формуле:

$$g = \frac{3uv}{a^2} \quad (6)$$

Порядок действий при моделировании с помощью LBM:

Процедура моделирования течения жидкости в трубе, пористом или трещиноватом домене сводится к следующему:

1. Разобрать описание работы модели, исходный код с комментариями ниже и прилагаемые исходные коды течения в трубе и трещинном домене;
2. Необходимо рассчитать присущие жидкости варианта переменные модели g , τ ;
3. Инкорпорировать в исходный код параметры ρ , g , τ , изображение пористой структуры из работы №6;
4. Запустить программу, сохранить картину протекания жидкости через структуру;
5. Рассчитать скорость протекания жидкости, как векторную сумму u_x и u_y . Показать и сохранить, как матрицу с цветовой индикацией.

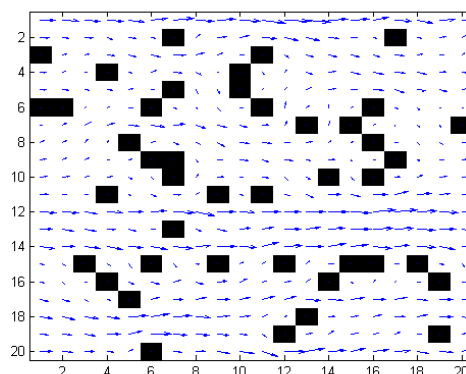


Рис. 2. Линии потока в пористом домене

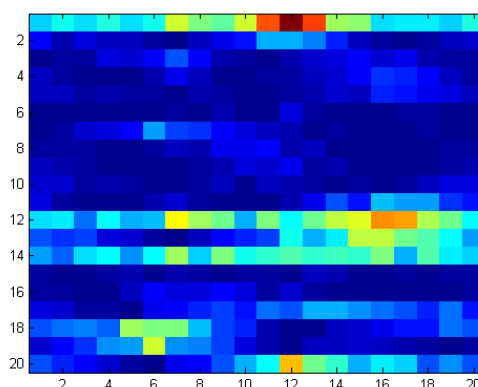


Рис. 3. Распределение скорости потока жидкости $(u_x.^2+u_y.^2).^0.5$

Ниже приводится описание модели Matlab в виде комментариев исходного кода Matlab. Необходимо ознакомиться с ними для понимания принципов и хода работы программы.

Контрольные вопросы

1. Как рассчитать и графически, в виде цветовой индикации, показать распределение интегральной скорости u в домене, при условии, что известны ix и iy ?
2. Какие максимальная скорость u_{max} и минимальная скорость u_{min} в домене? Показать области эллипсом на изображении.
3. Насколько средняя скорость в домене отличается от рассчитанной по формуле при постановке задачи? Почему?

Требования к отчету

Файл программы Octave/Matlab, содержащий все необходимые комментарии и выполняющий автоматизированные расчеты по заданию. Графические изображения зависимости, построенные в программе (не скриншоты!) с подписями.

Основная литература

1. Sukop, M.C., Thorne, D.T., 2007. Lattice Boltzmann Modeling. An Introduction for Geoscientists and Engineers. Springer-Verlag Berlin Heidelberg, 177 pp.
2. Mike Sukop Introduction to Lattice Boltzmann Method // [Электронный ресурс]. Florida International University URL: <http://faculty.fiu.edu/~sukopm/GLY5835/GLY5835.htm> (Дата обращения: 09.09.2017)

Дополнительная литература

3. Поршнева С. В. Компьютерное моделирование физических процессов в пакете MATLAB / М.:, 2003. 593 с.
4. Алексеев Е.Р. , Чеснокова О.В. Введение в Octave для инженеров и математиков: / Е.Р. Алексеев, О.В.Чеснокова М.: ALT Linux, 2012. 368 с.
5. Материалы по продуктам MATLAB & Toolboxes // [Электронный ресурс]: Математический сайт Exponenta.ru. Веб-сайт. URL: <http://matlab.exponenta.ru/index.php> (Дата обращения: 05.11.2015)

Варианты заданий

Номер варианта	Плотность, ρ [т/м ³]	Кинематическая вязкость, ν [м ² /с] [Ст]
1	0.786	0,67
2	0.805	0,52
3	0.759	1,23
4	0.815	1,28
5	0.746	1,34
6	0.739	0,46

7	0.817	0,46
8	0.805	0,35
9	0.743	0,31
10	0.750	1,4

Приложение. Исходный код с комментариями

```
%Sukop and Thorne's (c) Lattice Boltzmann Modeling. 2006
close('all');clear('all') %очистка памяти

display('initialize')
LY=20 %параметры домена - ширина и высота
LX=20
tau = 1; %время релаксации
g=0.001 %гравитационное ускорение
rho=1; %плотность жидкости

%фрагмент кода ниже устанавливает границы сверху и снизу
is_solid_node=zeros(LY,LX);
for i=1:LX
    is_solid_node(1,i)=1;
    is_solid_node(LY,i)=1;
end

display('solid nodes')
%этот фрагмент кода генерирует 30% препятствий в домене
is_solid_node=(rand(LY,LX))>0.7;

%задание начальной плотности rho и f (массовой плотности частиц, перемещающихся из одного
%узла к другому)
rho=ones(LY,LX)*rho; %распределение плотности жидкости в домене
f(:, :, 1) = (4./9.)*rho; %f - плотность частиц,
f(:, :, 2) = (1./9.)*rho; %взвешенная в соответствии с направлениями потока
f(:, :, 3) = (1./9.)*rho;
f(:, :, 4) = (1./9.)*rho;
f(:, :, 5) = (1./9.)*rho;
f(:, :, 6) = (1./36.)*rho;
f(:, :, 7) = (1./36.)*rho;
f(:, :, 8) = (1./36.)*rho;
f(:, :, 9) = (1./36.)*rho;

%определение векторов скорости e (как показано на рисунке ниже)
```

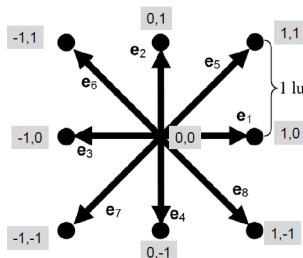


Figure 22. D2Q9 x,y velocity components.

```
ex(0+1)= 0; ey(0+1)= 0; %(0+1) потому, что индекс в Matlab начинается с 0
ex(1+1)= 1; ey(1+1)= 0;
ex(2+1)= 0; ey(2+1)= 1;
ex(3+1)=-1; ey(3+1)= 0;
ex(4+1)= 0; ey(4+1)=-1;
ex(5+1)= 1; ey(5+1)= 1;
ex(6+1)=-1; ey(6+1)= 1;
ex(7+1)=-1; ey(7+1)=-1;
ex(8+1)= 1; ey(8+1)=-1;

for ts=1:300 %цикл шага времени построения модели
    ts %счетчик времени
    % расчет макроскопических плотности rho и скорости u=(ux,uy).
    for j=1:LY %шаг по Y

        for i=1:LX %шаг по X
```

```

u_x(j,i) = 0.0; %определение начальных значений узлов (j,i)
                %скорости и плотности в соответствующих матрицах
u_y(j,i) = 0.0;
rho(j,i) = 0.0;

if ~is_solid_node(j,i) %если узел (j,i) не является препятствием
    for a=0:8 %для каждого из 8 направлений векторов возможного движения частицы
        %по формуле микроскопической плотности

```

$$\rho = \sum_{a=0}^8 f_a$$

```

rho(j,i) = rho(j,i) + f(j,i,a+1);

```

%т.к. макроскопическая скорость \mathbf{u} является средним значением макроскопических скоростей \mathbf{e}_a взвешенных относительно плотности по направлению f_a :

$$\mathbf{u} = \frac{1}{\rho} \sum_{a=0}^8 f_a \mathbf{e}_a$$

```

    u_x(j,i) = u_x(j,i) + ex(a+1)*f(j,i,a+1); %взвесить fa в соответствии
    с микроскопической плотностью
    u_y(j,i) = u_y(j,i) + ey(a+1)*f(j,i,a+1);

```

```

end

```

```

    u_x(j,i) = u_x(j,i)/rho(j,i); %поделить на rho для u_x and u_y в
    соответствии с микроскопической скоростью
    u_y(j,i) = u_y(j,i)/rho(j,i);

```

```

end

```

```

%определяем матрицы x и y для вывода направлений потока
x(j,i)=i;
y(j,i)=j;

```

```

end

```

```

end

```

% расчет функции равновесия, f_{eq} .

$$\underbrace{f_a(\mathbf{x} + \mathbf{e}_a \Delta t, t + \Delta t)}_{\text{Streaming}} = f_a(\mathbf{x}, t) - \underbrace{\frac{[f_a(\mathbf{x}, t) - f_a^{eq}(\mathbf{x}, t)]}{\tau}}_{\text{Collision}}$$

```

f1=3.; %веса плотности
f2=9./2.;
f3=3./2.;

```

%столкновение частиц жидкости рассматривается как релаксация в сторону локального равновесия, формула функции локального равновесия f_{eq} приведена ниже:

$$f_a^{eq}(\mathbf{x}) = w_a \rho(\mathbf{x}) \left[1 + 3 \frac{\mathbf{e}_a \cdot \mathbf{u}}{c^2} + \frac{9}{2} \frac{(\mathbf{e}_a \cdot \mathbf{u})^2}{c^4} - \frac{3}{2} \frac{u^2}{c^2} \right] \quad (17)$$

```

for j=1:LY

```

```

    for i=1:LX

```

```

        if ~is_solid_node(j,i)

```

```

            rt0 = (4./9.)*rho(j,i); %W_a*rho(x)
            rt1 = (1./9.)*rho(j,i);
            rt2 = (1./36.)*rho(j,i);

```

```

            ueqxij = u_x(j,i)+tau*g;
            ueqyij = u_y(j,i);

```

```

            %ниже рассчитывается макроскопическая скорость
            %расчет квадратической скорости (as vector addition)

```

```

uxsq = ueqxij * ueqxij;
uysq = ueqyij * ueqyij;

%расчет скорости движения по диагонали (dx dy addition)
uxuy5 = ueqxij + ueqyij;
uxuy6 = -ueqxij + ueqyij;
uxuy7 = -ueqxij + -ueqyij;
uxuy8 = ueqxij + -ueqyij;

usq = uxsq + uysq; %расчет квадрата скорости


$$f_a^{eq}(\mathbf{x}) = w_a \rho(\mathbf{x}) \left[ 1 + 3 \frac{\mathbf{e}_a \cdot \mathbf{u}}{c^2} + \frac{9 (\mathbf{e}_a \cdot \mathbf{u})^2}{2 c^4} - \frac{3 \mathbf{u}^2}{2 c^2} \right]$$


%f1=3.; %веса плотности
%f2=9./2.;
%f3=3./2.;
%c - базовая скорость в модели, 1

feq(j,i,0+1) = rt0*( 1. - f3*usq);
feq(j,i,1+1) = rt1*( 1. + f1*ueqxij + f2*uxsq - f3*usq);
feq(j,i,2+1) = rt1*( 1. + f1*ueqyij + f2*uysq - f3*usq);
feq(j,i,3+1) = rt1*( 1. - f1*ueqxij + f2*uxsq - f3*usq);
feq(j,i,4+1) = rt1*( 1. - f1*ueqyij + f2*uysq - f3*usq);
feq(j,i,5+1) = rt2*( 1. + f1*uxuy5 + f2*uxuy5*uxuy5 - f3*usq);
feq(j,i,6+1) = rt2*( 1. + f1*uxuy6 + f2*uxuy6*uxuy6 - f3*usq);
feq(j,i,7+1) = rt2*( 1. + f1*uxuy7 + f2*uxuy7*uxuy7 - f3*usq);
feq(j,i,8+1) = rt2*( 1. + f1*uxuy8 + f2*uxuy8*uxuy8 - f3*usq);

end
end
end
%на шаге коллизии и отражения рассчитывается массовая плотность в каждом направлении
% столкновение
for j=1:LY
for i=1:LX

if is_solid_node(j,i);

% обычный отскок от препятствия

temp = f(j,i,1+1); f(j,i,1+1) = f(j,i,3+1); f(j,i,3+1) = temp;
temp = f(j,i,2+1); f(j,i,2+1) = f(j,i,4+1); f(j,i,4+1) = temp;
temp = f(j,i,5+1); f(j,i,5+1) = f(j,i,7+1); f(j,i,7+1) = temp;
temp = f(j,i,6+1); f(j,i,6+1) = f(j,i,8+1); f(j,i,8+1) = temp;

else
% обычное столкновение

for a=1:9

f(j,i,a) = f(j,i,a)-( f(j,i,a) - feq(j,i,a))/tau;
end

end
end
%шаг течения, некоторые изменения здесь для индексации
%индексы jn и jp добавлены из-за перехода с конца на начало сетки
for j=1:LY

if j>1
jn = j-1;
else
jn = LY;
end

if j<LY
jp = j+1;
else
jp = 1;
end

for i=1:LX

```

```

    if i>1
        in = i-1;
    else
        in = LX;
    end
    if i<LX
        ip = i+1;
    else
        ip = 1;
    end

    ftemp(j,i,0+1) = f(j,i,0+1);
    ftemp(j,ip,1+1) = f(j,i,1+1);
    ftemp(jp,i,2+1) = f(j,i,2+1);
    ftemp(j,in,3+1) = f(j,i,3+1);
    ftemp(jn,i,4+1) = f(j,i,4+1);
    ftemp(jp,ip,5+1) = f(j,i,5+1);
    ftemp(jp,in,6+1) = f(j,i,6+1);
    ftemp(jn,in,7+1) = f(j,i,7+1);
    ftemp(jn,ip,8+1) = f(j,i,8+1);

    end
end

f(:, :, :) = ftemp(:, :, :);

end %конец цикла времени

figure
imagesc(~is_solid_node); hold on; colormap gray;
quiver(x,y,u_x,u_y)

width=LY-2

figure
%поток Пуазейля в трубе сечения 2а. Результаты работы модели и аналитически
plot(y(:,1)-1.5-width/2,u_x(:,1),'ro')

hold on

%Профиль скоростей Пуазейля как голубая линия
nu=1/3*(tau-1/2)
plot(y(:,1)-1.5-width/2,g/(2*nu)*((width/2)^2-(y(:,1)-1.5-width/2).^2))

```