模型-计算与模拟-计算机模拟-元胞自动机【hxy】

1. 模型名称

元胞自动机（Cellular Automaton Method，CA）

2. 适用范围

仿真局部规则和局部联系（如森林火灾模拟，传染病传播模拟等）

3. 形式

用一个四元组表示

$$CA = (L_d, S, N, f)$$

其中，L代表了一个规则划分的元胞空间[1]，每个网格空间就是一个元胞[2]，d为L的维数，理论上可以为任意正整数的规则空间；S代表一个离散的有限集合，用来表达各个元胞的状态s；N代表元胞的邻居[3]集合，即包含n个不同元胞状态的一个空间矢量，$N = (s_1, s_2, s_3, \Lambda s_n), s_i \in Z, i \in (1, \Lambda n)$; f表示一个映射函数$S_t^n \to S_{t+1}$，即根据t时刻某个元胞的所有邻居的状态组合来确定$t+1$时刻该元胞状态值

4. 求解过程

4.1 Conway的生命游戏机

- 规则
  - 对周围的8个近邻的元胞状态求和
  - 如果总和为2的话，则下一时刻的状态不改变
  - 如果总和为3的话，则下一时刻的状态为1
  - 否则状态为0
- 核心代码

```
% 更新元胞范围
x = 2:n - 1;
y = 2:n - 1;
% 邻居规则
sum(x,y) = cells(x,y-1) + cells(x,y+1) + ...
  cells(x-1,y) + cells(x+1,y) + ...
  cells(x-1,y-1) + cells(x-1,y+1) + ...
  cells(x+1,y-1) + cells(x+1,y+1);
% CA规则
cells = (sum==3)|(sum==2&cells);
```

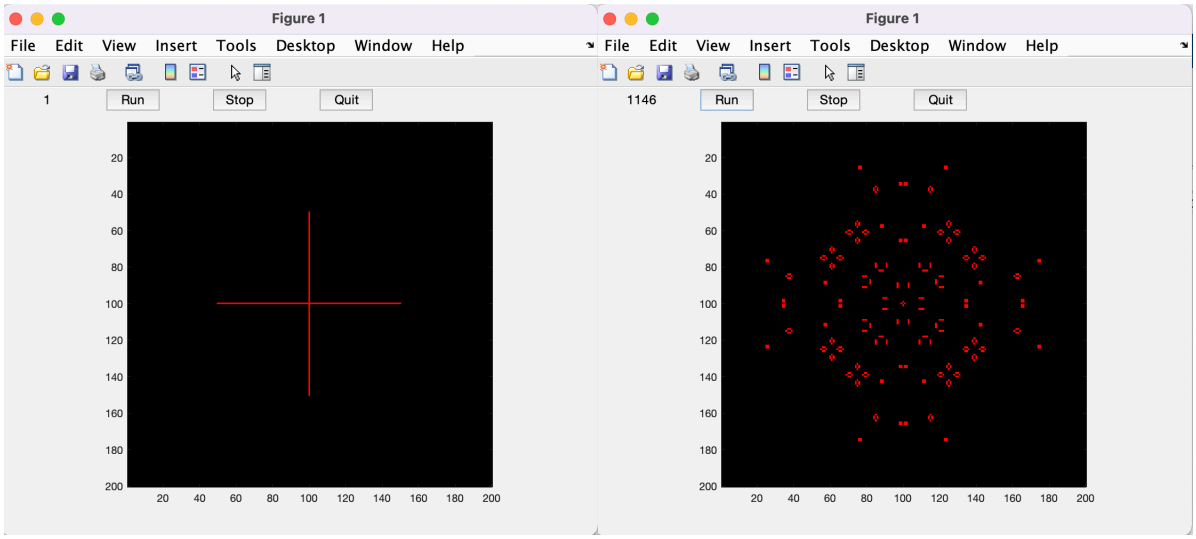
- 实现代码

  CA.m

```
% 加入用户界面：三个按钮（运行，停止，退出），一个文本框（仿真运行次数）
plotbutton = uicontrol('Style','pushbutton','String','Run',...
    'FontSize',12,'Position',[100,400,50,20],'Callback','run=1');
erasebutton = uicontrol("Style",'pushbutton','String','Stop',...
    'FontSize',12,'Position',[200,400,50,20],'Callback','freeze=1');
quitbutton = uicontrol('Style','pushbutton','String','Quit',...
    'FontSize',12,'Position',[300,400,50,20],'Callback','stop=1;close');
number = uicontrol('Style','text','String','1',...
    'FontSize',12,'Position',[20,400,50,20]);

% 初始化：元胞状态为0，中心十字形的元胞状态为1
n = 200;
z = zeros(n,n);
cells = z;
cells(n/2, .25*n:.75*n) = 1;
cells(.25*n:.75*n, n/2) = 1;
sum = z;
stop = 0;
run = 0;
freeze = 0;

% 更新元胞范围
x = 2:n-1;
y = 2:n-1;

% 建立RGB图像，返回句柄
        imh = image(cat(3,cells,z,z));
        set(imh,'erasemode','none');
        axis equal
        axis tight

% 循环
while(stop == 0)
    if(run == 1)
```

```
        % 邻居规则
        sum(x,y) = cells(x,y-1) + cells(x,y+1) + ...
            cells(x-1,y) + cells(x+1,y) + ...
            cells(x-1,y-1) + cells(x-1,y+1) + ...
            cells(x+1,y-1) + cells(x+1,y+1);
        % CA规则
        cells = (sum == 3) | (sum == 2 & cells);
        % 画图
        set(imh, 'cdata', cat(3,cells,z,z));
        % 更新模拟次数
        stepnumber = 1 + str2double(get(number,'string'));
        set(number,'string',num2str(stepnumber));
    end
    if(freeze == 1)
        run = 0;
        freeze = 0;
    end
drawnow
end
```

- 结果



4.2.2 表面张力

- 规则
  - 对周围的8近邻的元胞以及自身的状态求和
  - 如果总和<4或$=5$，下一时刻的状态为0
  - 否则状态为1
- 核心代码

```
% 更新元胞范围
x = 2:n-1;
y = 2:n-1;
% 邻居规则
sum(x,y) = cells(x,y-1) + cells(x,y+1) + ...
  cells(x-1,y) + cells(x+1,y) + ...
  cells(x-1,y-1) + cells(x-1,y+1) + ...
  cells(x+1,y-1) + cells(x+1,y+1) + ...
  cells(x,y);
% CA规则
cells = ~((sum<4)|(sum==5));
```

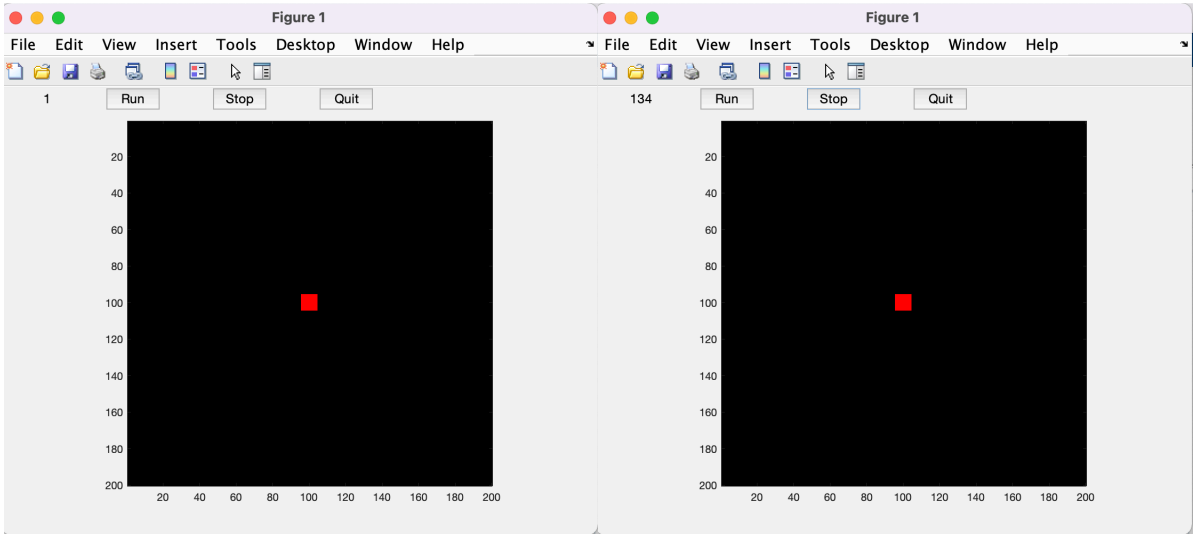
- 实现代码

  CA2.m

```
% 加入用户界面：三个按钮（运行，停止，退出），一个文本框（仿真运行次数）
plotbutton = uicontrol('Style','pushbutton','String','Run',...
    'FontSize',12,'Position',[100,400,50,20],'Callback','run=1');
erasebutton = uicontrol("Style",'pushbutton','String','Stop',...
    'FontSize',12,'Position',[200,400,50,20],'Callback','freeze=1');
quitbutton = uicontrol('Style','pushbutton','String','Quit',...
    'FontSize',12,'Position',[300,400,50,20],'Callback','stop=1;close');
number = uicontrol('Style','text','String','1',...
    'FontSize',12,'Position',[20,400,50,20]);

% 初始化：元胞状态为0，中心十字形的元胞状态为1
n = 200;
z = zeros(n,n);
cells = z;
cells(n/2-4:n/2+4, n/2-4:n/2+4) = 1;
sum = z;
stop = 0;
run = 0;
freeze = 0;

% 更新元胞范围
x = 2:n-1;
y = 2:n-1;

% 建立RGB图像，返回句柄
        imh = image(cat(3,cells,z,z));
        set(imh,'erasemode','none');
        axis equal
        axis tight

% 循环
while(stop == 0)
    if(run == 1)
```

```
        % 邻居规则
        sum(x,y) = cells(x,y-1) + cells(x,y+1) + ...
          cells(x-1,y) + cells(x+1,y) + ...
          cells(x-1,y-1) + cells(x-1,y+1) + ...
          cells(x+1,y-1) + cells(x+1,y+1) + ...
          cells(x,y);
        % CA规则
        cells = ~((sum<4)|(sum==5));
        % 画图
        set(imh, 'cdata', cat(3,cells,z,z));
        % 更新模拟次数
        stepnumber = 1 + str2double(get(number,'string'));
        set(number,'string',num2str(stepnumber));
    end
    if(freeze == 1)
        run = 0;
        freeze = 0;
    end
drawnow
end
```

- 结果



4.2.3 渗流集群

- 规则
  - 对周围相邻的8邻居求和（元胞只有两种状态，0或1），元胞也有一个单独的状态参量（所谓“记录”）记录他们之前是否有非零状态的邻居
  - 在0与1之间产生一个随机数r
  - 如果总和>0（至少一个邻居）并且$r>$阈值，或者元胞从未有过一个邻居，则元胞为1
  - 如果总和>0则设置“记录”的标志，记录这些元胞有一个非零的邻居
- 核心代码

```
% 变量a和b是图像的尺寸
ax = axes('units','pixels','position',[1 1 500 400],'color','k');
```

```
text('units','pixels','position',[130,255,0],...
  'string','MCM','color','w','fontname','helvetica','fontsize',100);
text('units','pixels','position',[10,120,0],...
'string','Cellular Automata','color','w','fontname','helvetica','fontsize',50);
initial = getframe(gca); % 用getframe把他们写入一个矩阵
% 初始化
[a,b,c] = size(initial.cdata);
z = zeros(a,b);
cells = double(initial.cdata(:,:,1)==255);
visit = z;
sum = z;
threshold = 0.5;
% 邻居规则
sum(2:a-1,2:b-1) = cells(2:a-1,1:b-2) + cells(2:a-1,3:b) + ...
            cells(1:a-2,2:b-1) + cells(3:a,2:b-1) + ...
            cells(1:a-2,1:b-2) + cells(1:a-2,3:b) + ...
            cells(3:a,1:b-2) + cells(3:a,3:b);
% CA规则
pick = rand(a,b);
cells = cells | ((sum>=1) & (pick>=threshold) & (visit==0));
visit = (sum>=1);
```

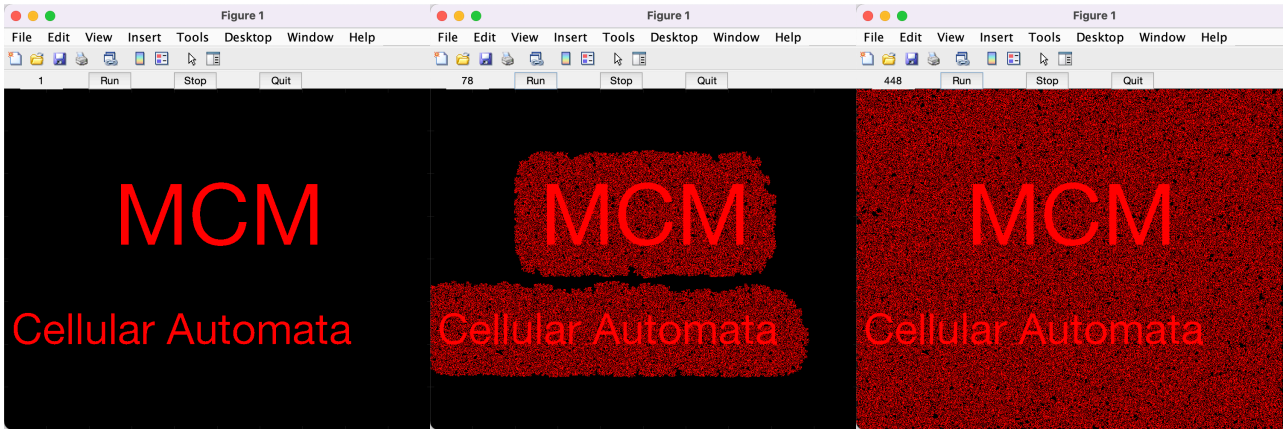
- 实现代码

  CA3.m

```
clc, clear;
% 加入用户界面：三个按钮（运行，停止，退出），一个文本框（仿真运行次数）
plotbutton = uicontrol('Style','pushbutton','String','Run',...
    'FontSize',12,'Position',[100,400,50,20],'Callback','run=1');
erasebutton = uicontrol("Style",'pushbutton','String','Stop',...
    'FontSize',12,'Position',[200,400,50,20],'Callback','freeze=1');
quitbutton = uicontrol('Style','pushbutton','String','Quit',...
    'FontSize',12,'Position',[300,400,50,20],'Callback','stop=1;close');
number = uicontrol('Style','text','String','1',...
    'FontSize',12,'Position',[20,400,50,20]);

% 设定坐标系是一个固定的尺寸，把坐标系里写入文本，然后获得并返回坐标内的内容
ax = axes('units','pixels','position',[1 1 500 400],'color','k');
text('units','pixels','position',[130,255,0],...
  'string','MCM','color','w','fontname','helvetica','fontsize',100);
text('units','pixels','position',[10,120,0],...
'string','Cellular Automata','color','w','fontname','helvetica','fontsize',50);
initial = getframe(gca); % 用getframe把他们写入一个矩阵

% 初始化
[a,b,c] = size(initial.cdata);
z = zeros(a,b);
cells = double(initial.cdata(:,:,1)==255);
```

```
visit = z;
sum = z;
threshold = 0.5;

% 初始化状态
stop = 0;
run = 0;
freeze = 0;

% 建立RGB图像，返回句柄
        imh = image(cat(3,cells,z,z));
        set(imh,'erasemode','none');
        axis equal
        axis tight

% 循环
while(stop == 0)
    if(run == 1)
        % 邻居规则
        sum(2:a-1,2:b-1) = cells(2:a-1,1:b-2) + cells(2:a-1,3:b) + ...
            cells(1:a-2,2:b-1) + cells(3:a,2:b-1) + ...
            cells(1:a-2,1:b-2) + cells(1:a-2,3:b) + ...
            cells(3:a,1:b-2) + cells(3:a,3:b);
        % CA规则
        pick = rand(a,b);
        cells = cells | ((sum>=1) & (pick>=threshold) & (visit==0));
        visit = (sum>=1);
        % 画图
        set(imh, 'cdata', cat(3,cells,z,z));
        % 更新模拟次数
        stepnumber = 1 + str2double(get(number,'string'));
        set(number,'string',num2str(stepnumber));
    end
    if(freeze == 1)
        run = 0;
        freeze = 0;
    end
drawnow
end
```

- 结果



5. 参考资料

1. CA代码及应用-Matlab（超多代码实例）
2. CA案例-Matlab
3. 数模官网-CA

---

1. 元胞空间：元胞所分布的空间网店集合，理论上可以是任意维数的欧几里得空间规则划分 ↩

2. 元胞：单元或基元，是元胞自动机最基本的组成部分 ↩

3. 基于邻居规则，通常有几种形式：1. 冯诺依曼型（一个元胞的上、下、左、右、为元胞的邻居，$N_{Neumann}=\{v_1=(v_{ix},v_{iy}||v_{ix}-v_{ax}|+|v_{iy}-v_{oy}|\leq 1,(v_{ix,v_{iy}})\in Z^2)\}$）；2. 摩尔型（一个元胞的上、下、左、右、左上、右上、右下、左下为元胞的邻居）； 3. 扩展的摩尔型（将以上的邻居半径扩展为2或者更大）； 4. 马哥勒斯型（每次将一个$2*2$的元胞块做统一处理） ↩