Описание функции `computeHeatFluxJump` с примерами из кода

Назначение функции

Функция `computeHeatFluxJump` вычисляет разность тепловых потоков на границе между жидкой и паровой фазами:

$$R = q_{\text{vap}}^{(n)} - q_{\text{liq}}^{(n)} = -\lambda_{\text{vap}} \frac{\partial T}{\partial n}_{\text{vap}} + \lambda_{\text{liq}} \frac{\partial T}{\partial n}_{\text{liq}}$$

Она возвращает двумерное поле R с невязками потока в ячейках, находящихся на фазовой границе.

Код:

```
function R = computeHeatFluxJump(fields, params, phase, dx, dy)
```

1. Загрузка полей и параметров

Считывается размерность сетки, температурное поле и формируется поле теплопроводности в зависимости от фазы.

Код:

```
[M, N] = size(phase);
T = fields.T;
lambda = params.lambda_liquid * (phase == 0) + ...
params.lambda_vapor * (phase == 1);
```

2. Вычисление градиентов температуры

Для расчёта нормальной компоненты теплового потока нужно знать $\partial T/\partial x$ и $\partial T/\partial y$.

Код:

```
[dTdx, dTdy] = gradient(T, dx, dy);
```

3. Подготовка обхода по соседям

Проход по 4 направлениям: влево, вверх, вправо, вниз. Для каждой стороны задаются сдвиг и нормальный вектор.

Код:

```
shifts  = {[-1, 0], [0, -1], [1, 0], [0, 1]};
normals = {[0, -1], [-1, 0], [0, 1], [1, 0]};
```

4. Поиск межфазных ячеек

Определяется, какие пары ячеек находятся по разные стороны фазовой границы. Проверяется условие `phase1` $\neq$ `phase2`.

Код:

```
p1 = phase(idx1);
p2 = phase(idx2);
is_interface = p1 ~= p2;
gas_left  = is_interface & (p1 == 1);
gas_right = is_interface & (p2 == 1);
```

5. Расчёт нормальных градиентов температуры

Вычисляется $\nabla T \cdot \vec{n}$ — скалярное произведение градиента и нормали к границе:

$$\frac{\partial T}{\partial n} = \frac{\partial T}{\partial x} n_x + \frac{\partial T}{\partial y} n_y$$

Код:

```
gradT1_n = dTdx(idx1) * ni + dTdy(idx1) * nj;
gradT2_n = dTdx(idx2) * ni + dTdy(idx2) * nj;
```

6. Построение тепловых потоков

В каждой точке вычисляется тепловой поток по закону Фурье:

$$q^{(n)} = -\lambda \frac{\partial T}{\partial n}$$

Код:

```
k1 = lambda(idx1); k2 = lambda(idx2);
q1 = -k1 .* gradT1_n;
q2 = -k2 .* gradT2_n;
```

7. Вычисление локальной невязки потока

Для межфазных ячеек вычисляется $q_{\text{vap}} - q_{\text{liq}}$ в зависимости от стороны, с которой находится пар:

Код:

```
dQ = zeros(size(idx1));
dQ(gas_left)  = q1(gas_left) - q2(gas_left);
dQ(gas_right) = q2(gas_right) - q1(gas_right);
```

8. Агрегация результата в поле R

Невязка потока добавляется в результирующее поле R:

Код:

```
R(idx1) = R(idx1) + dQ;
```

Физический смысл

Функция проверяет выполнение граничного условия на тепловой поток через фазовую границу:

$$\lambda \frac{\partial T}{\partial n}\bigg|_{\text{liq}} = \lambda \frac{\partial T}{\partial n}\bigg|_{\text{vap}}$$

Отклонение от этого условия численно проявляется в виде ненулевого R.