模拟铁铬合金在500°C下一周内的旋节分解

参考教程：<https://mooseframework.inl.gov/modules/phase_field/Tutorial.html>

## 网格自适应

基本网格为25×25个元素。这定义了网格的最大粗糙度。uniform\_refine选项在第一步将网格细化为100×100个元素的网格。如果我们不启用网格自适应，网格将在整个模拟过程中保持100×100，并且与我们已经使用的网格完全相同。

[Mesh]

type = GeneratedMesh

dim = 2

elem\_type = QUAD4

nx = 25

ny = 25

nz = 0

xmin = 0

xmax = 25

ymin = 0

ymax = 25

zmin = 0

zmax = 0

uniform\_refine = 2

[]

为了添加网格自适应性，我们回到执行器块，添加自适应子块，并告诉它我们希望最大细化级别 比允许的最粗网格 精细两个因素。

[Adaptivity]

coarsen\_fraction = 0.1

refine\_fraction = 0.7

max\_h\_level = 2

[]

## 初始条件：

初始条件可以非常方便的定义，如下图

### 固定的

[ICs]

*#* Use a bounding box IC at equilibrium concentrations to make sure the

*#* model behaves as expected.

  [./testIC]

    type = BoundingBoxIC

    variable = c

    x1 = 5

    x2 = 20

    y1 = 5

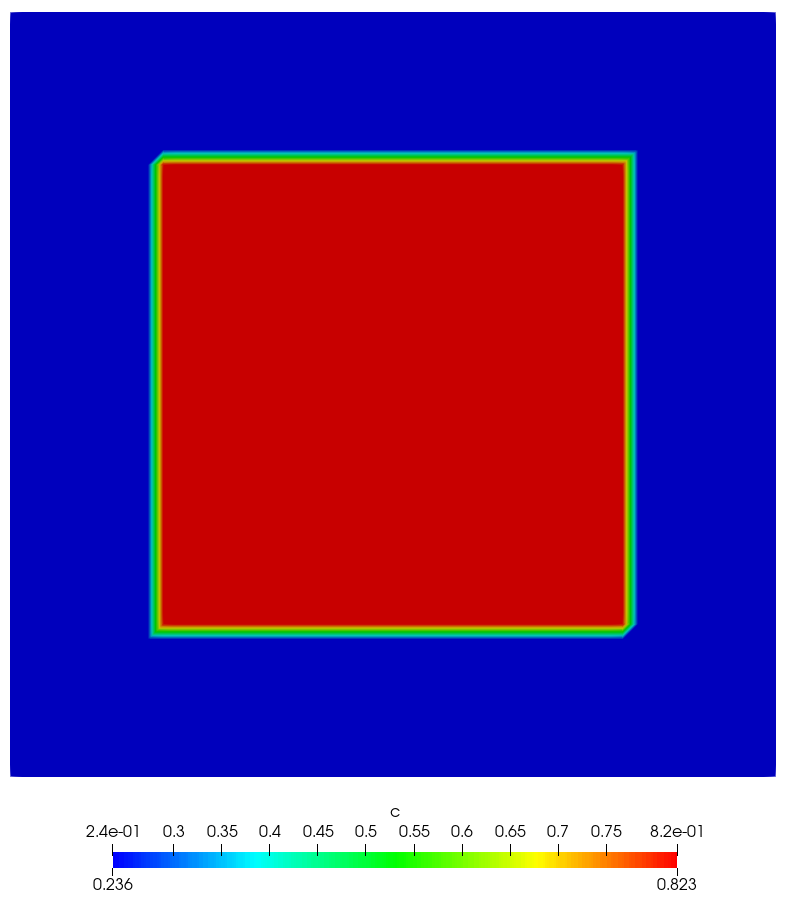
    y2 = 20

    inside = 0.823

    outside = 0.236

  [../]

[]



### 随机的

[concentrationIC] # 46.774 mol% Cr with variations

type = RandomIC

min = 0.44774

max = 0.48774

seed = 210

variable = c

[]

## 循环边界条件的定义

[BCs]

*#* periodic BC as is usually done on phase-field models

  [./Periodic]

    [./c\_bcs]

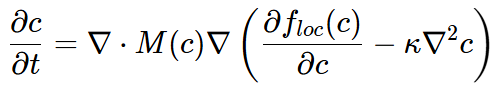
      auto\_direction = 'x y'

    [../]

  [../]

[]

## 核



[Kernels]

*#* See wiki page "Developing Phase Field Models" for more information on Split

*#* Cahn-Hilliard equation kernels.

*#* <http://mooseframework.org/wiki/PhysicsModules/PhaseField/DevelopingModels/>

  [./w\_dot]

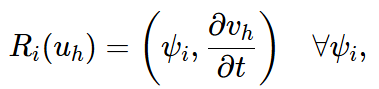
    variable = w

    v = c

    type = CoupledTimeDerivative

  [../]

左边第一项的弱形式：



为什么使用CoupledTimeDerivative？而不是直接TimeDerivative？因为w（c）即（floc）是c的函数，如果不少别的函数的自变量，则使用TimeDerivative即可

  [./coupled\_res]

    variable = w

    type = SplitCHWRes

    mob\_name = M

  [../]

  [./coupled\_parsed]

    variable = c

    type = SplitCHParsed

    f\_name = f\_loc

    kappa\_name = kappa\_c

    w = w

  [../]

[]

## 输出模拟累积残差计算与时间

[Postprocessors]

[./step\_size] # Size of the time step

type = TimestepSize

[../]

[./iterations] # Number of iterations needed to converge timestep

type = NumNonlinearIterations

[../]

[./nodes] # Number of nodes in mesh

type = NumNodes

[../]

[./evaluations] # Cumulative residual calculations for simulation

type = NumResidualEvaluations

[../]

[./total\_energy] # Total free energy at each timestep

type = ElementIntegralVariablePostprocessor

variable = f\_density

execute\_on = 'initial timestep\_end'

[../]

[./num\_features] # Number of precipitates formed

type = FeatureFloodCount

variable = c

threshold = 0.6

[../]

[./precipitate\_area] # Fraction of surface devoted to precipitates

type = ElementIntegralMaterialProperty

mat\_prop = prec\_indic

[../]

[./active\_time] # Time computer spent on simulation

type = PerfGraphData

section\_name = "Root"

data\_type = total

[../]

[]

## 残差缩放

还有另一种选择可以提高收敛速度。如果所有变量的残差大小相同，则系统收敛最佳。MOOSE有一个称为缩放的选项，它允许我们将其中一个变量的残差乘以某个因子，以使残差达到相同的幅度。但在我们这样做之前，我们需要知道残差是什么。我们添加一个Debug块来告诉我们每次迭代后的残差。