参考文献：

[1]P. Descouvemont and D. Baye, Rep. Prog. Phys. 73 036301(2010)
[2]D. Baye, Phys. Rep. 565 1(2015)

# R矩阵求解束缚态与散射态

大多数幻灯片取自S. Quaglioni, Talent Course 6, MSU, 2019

# Schrodinger equations for local potential

$$\left(\frac{d^2}{dr^2}-\frac{l(l+1)}{r^2}-\frac{2\mu}{\hbar^2}V(r)+k^2\right)u_l(r)=0 \quad ; \quad k^2=2\mu E/\hbar^2$$

$$\left(T_l(r)+V(r)-E\right)u_l(r)=0 \quad ; \quad T_l(r)=-\frac{\hbar^2}{2\mu}\left(\frac{d^2}{dr^2}-\frac{l(l+1)}{r^2}\right)$$

- More in general, accounting for also the spin and isospin d.o.f

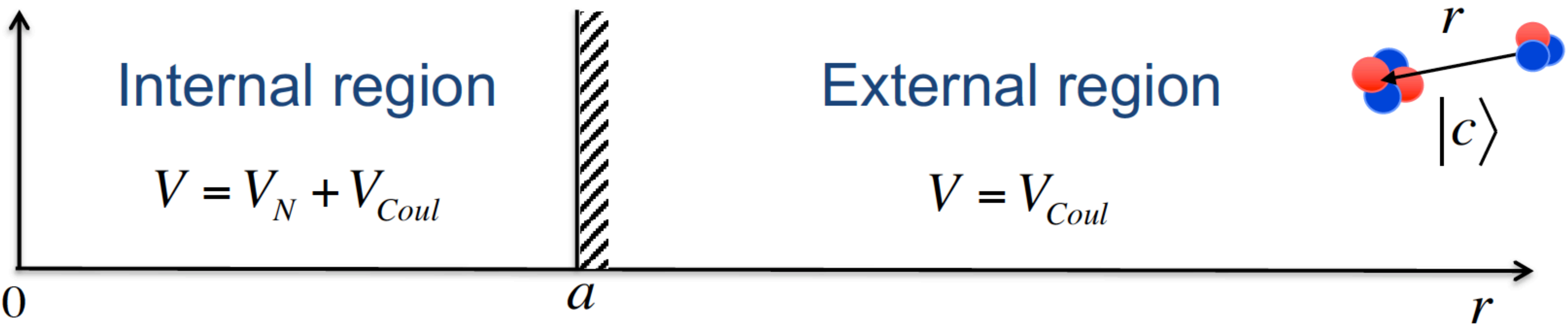
$$(T_c(r)-E)u_c(r)+\sum_{c'}V_{cc'}(r)u_{c'}(r)=0 \quad ; \quad c=\{l,s,j,t\}$$

# These equations can be solved using R−matrix theory

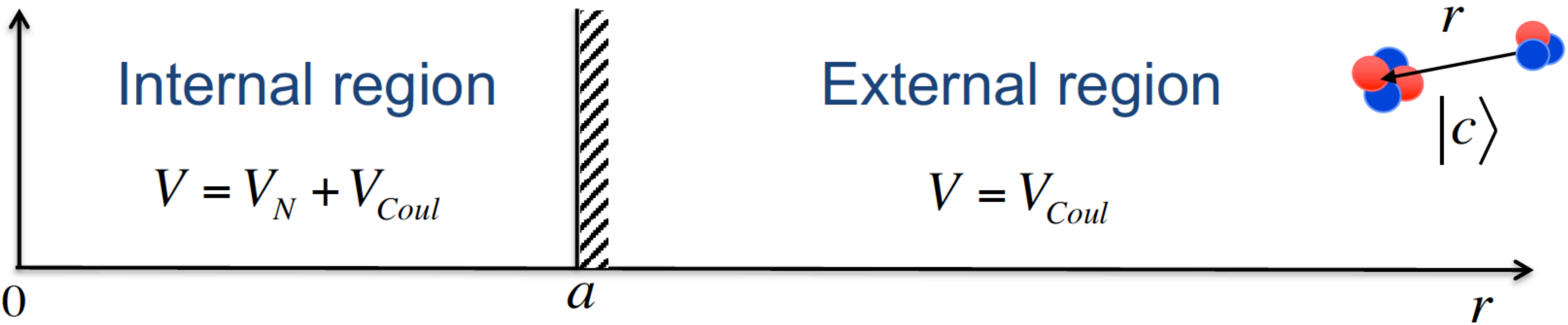


# These equations can be solved using R−matrix theory





# These equations can be solved using R−matrix theory




**Expansion on a basis (square-integrable)**

$$u_c(r) = \sum_n A_{cn} f_n(r)$$

**Bound state asymptotic behavior**

$$u_c(r) = C_c W(k_c r)$$

**Scattering state asymptotic behavior**

$$u_c(r) = \frac{i}{2} v_c^{-\frac{1}{2}} \left[ \delta_{ci} I_c(k_c r) - S_{ci} O_c(k_c r) \right]$$

# Internal region: Lagrange basis functions

- Internal region:

$$u_c(r) = \sum_n A_{cn} f_n(r) \; ; \quad N \text{ Lagrange basis functions } f_n(r)$$

associated with a Lagrange mesh of $N$ points $ax_n \in [0,a]$

$r_n = ax_n$

$x_n$ ... zero of shifted Legendre polynomials: $P_N(2x_n - 1) = 0$

$$f_n(r) = (-1)^{N-n} a^{-1/2} \sqrt{\frac{1-x_n}{x_n}} \frac{r}{r - ax_n} P_N(\tfrac{2r}{a} - 1)$$

$$f_{n'}(ax_n) = \frac{1}{\sqrt{a\lambda_n}} \delta_{n,n'} \quad \ldots \quad \text{zero at all mesh points except one}$$

- $\lambda_n$ … weights of the Gauss-Legendre quadrature approx. of integral

$$\int_0^1 g(x)dx \approx \sum_{n=1}^{N} \lambda_n g(x_n)$$

# Bloch operator



- R-matrix formalism conveniently expressed with the help of the Bloch surface operator

$$L_c = \frac{\hbar^2}{2\mu_c}\delta(r-a)\left(\frac{d}{dr} - \frac{B_c}{r}\right)$$

Boundary parameters

- System of Bloch-Schrödinger equations:

$$(T_{rel}(r) + L_c - E)u_c(r) + \sum_{c'} V_{cc'}(r)u_{c'}(r) = L_c u_c(r) \quad ;$$

$$u_c(r) = \sum_n A_{cn} f_n(r)$$

asymptotic form for large $r$

# Bound states



- We can choose:

$$L_c = \frac{\hbar^2}{2\mu_c}\delta(r-a)\left(\frac{d}{dr} - \frac{B_c}{r}\right)$$

$$B_c = k_c a \frac{W'(k_c a)}{W(k_c a)} \Rightarrow L_c u_c^{ext}(r) = 0$$

- After projection (from the left) on the basis $f_n(r)$:

$$\sum_{c'n'} \left[ C_{cn,c'n'} - E\,\delta_{cn,c'n'} \right] A_{c'n'} = 0$$

$$\langle f_n | T_{rel}(r) + L_c + V_{Coul}(r) | f_{n'} \rangle \delta_{cc'} + \langle f_n | V_{cc'}^N(r) | f_{n'} \rangle$$

# Bound states



- We can choose:

$$B_c = k_c a \frac{W'(k_c a)}{W(k_c a)} \Rightarrow L_c u_c^{ext}(r) = 0$$

- After projection (from the left) on the basis $f_n(r)$:

$$\sum_{c'n'} \left[ C_{cn,c'n'} - E\,\delta_{cn,c'n'} \right] A_{c'n'} = 0$$

Eigenvalue problem

- Start with $E = 0$ and solve iteratively ($k_c$ depends on the energy!)
- Convergence in few iterations

# Matrix elements on Lagrange basis functions



- Lagrange basis functions orthonormal within the quadrature approximation

$$\int_0^a f_n(r) f_{n'}(r)\,dr \approx \delta_{n,n'}$$

- Potential:

$$r_n = a x_n$$

$$\langle f_n | V | f_{n'} \rangle = \int_0^a f_n(r) V(r) f_{n'}(r)\,dr \approx V(a x_n)\,\delta_{n,n'}$$

$$L_c(B) = L_c(0) - \frac{\hbar^2}{2\mu}\delta(r-a)\frac{B}{r}$$

- Kinetic energy ($n = n'$)

$$\langle f_n | T_{l=0}(r) + L_c(0)\, | f_n \rangle = \frac{\hbar^2}{2\mu}\frac{1}{3a^2 x_n(1-x_n)}\left[4N(N+1) + 3 + \frac{1-6x_n}{x_n(1-x_n)}\right]$$

# Matrix elements on Lagrange basis functions



- Kinetic energy ($n \neq n'$)

$$T_{rel} = T_{l=0} + \frac{\hbar^2 l(l+1)}{2\mu r^2}$$

$$L_c(B) = L_c(0) - \frac{\hbar^2}{2\mu}\delta(r-a)\frac{B}{r}$$

$$\langle f_n | T_{l=0}(r) + L_c(0) \, | f_{n'} \rangle$$

$$= \frac{\hbar^2}{2\mu} \frac{(-1)^{n+n'}}{a^2\sqrt{x_n x_{n'}(1-x_n)(1-x_{n'})}} \left[ N(N+1) + 1 + \frac{x_n + x_{n'} - 2x_n x_{n'}}{(x_n - x_{n'})^2} - \frac{1}{1-x_n} - \frac{1}{1-x_{n'}} \right]$$

- Centrifugal barrier

$$\left\langle f_n \left| \frac{\hbar^2 l(l+1)}{2\mu r^2} \right| f_{n'} \right\rangle = \frac{\hbar^2}{2\mu} \frac{l(l+1)}{a^2 x_n^2} \delta_{nn'}$$

# Matrix elements on Lagrange basis functions



$$T_{rel} = T_{l=0} + \frac{\hbar^2 l(l+1)}{2\mu r^2}$$

$$L_c(B) = L_c(0) - \frac{\hbar^2}{2\mu}\delta(r-a)\frac{B}{r}$$

- B-dependent part of Bloch operator

$$\left\langle f_n \middle| -\frac{\hbar^2}{2\mu}\delta(r-a)\frac{B}{r} \middle| f_{n'} \right\rangle = -\frac{\hbar^2}{2\mu}\frac{B(-1)^{n+n'}}{a^2\sqrt{x_n x_{n'}(1-x_n)(1-x_{n'})}}$$

- Other useful quantity

$$f_n(a) = \frac{(-1)^n}{\sqrt{a}\sqrt{x_n\,(1-x_n)}}$$

# Bound–state algorithm



$$E_0 = 0 \;\Rightarrow k_0 = 0$$

Iterations: 1,2,3, … *i*

1) $E_i = E_{i-1} \Rightarrow k_i = \sqrt{\frac{-2\mu E_i}{\hbar^2}},\, B_i = k_i a \frac{W'(k_i a)}{W(k_i a)}$

2) *Compute*

$$C_{nn'} = \langle f_n | T_{rel}(r) + L_c(\,B_i) + V_{Coul}(r) | f_{n'} \rangle \delta_{cc'} + \langle f_n \big| V^N_{cc'}(r) \big| f_{n'} \rangle$$

3) *Diagonalize the matrix:* $C - EI = 0$

4) *Compute* $\varepsilon = |E_i - E|$

5) *If* $\varepsilon \leq$ *tolerance: stop* ➔ *Binding energy = E*

6) *Else, repeat until convergence!*

# Scattering states

- We can choose $B_c = 0$
- After projection (from the left) on the basis $f_n(r)$:

$$\sum_{c'n'} \left[ C_{cn,c'n'} - (E - E_c)\delta_{cn,c'n'} \right] A_{c'n'} = \frac{\hbar^2 k_c}{2\mu_c v_c^{1/2}} \left\langle f_n \middle| L_c \middle| I_c \delta_{ci} - S_{ci} O_c \right\rangle$$

1) Solve for $A_{cn}$

2) Match internal and external solutions at channel radius, $a$

$$\sum_{c'} R_{cc'} \frac{k_{c'} a}{\sqrt{\mu_{c'} v_{c'}}} \left[ I'_{c'}(k_{c'}a)\delta_{ci} - S_{c'i} O'_{c'}(k_{c'}a) \right] = \frac{1}{\sqrt{\mu_c v_c}} \left[ I_c(k_c a)\delta_{ci} - S_{ci} O_c(k_c a) \right]$$

# Scattering states

- In the process introduce R-matrix, projection of the Green's function operator on the channel-surface functions

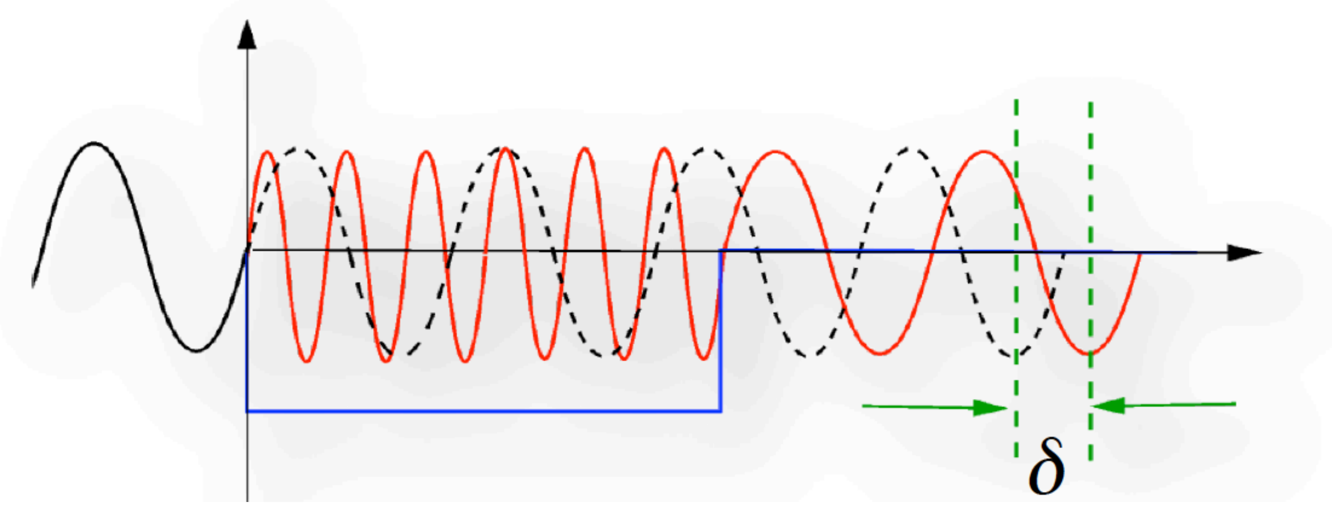
$$R_{cc'} = \sum_{nn'} \frac{\hbar}{\sqrt{2\mu_c a}} f_n(a) \left[C - EI\right]^{-1}_{cn,c'n'} \frac{\hbar}{\sqrt{2\mu_{c'} a}} f_{n'}(a)$$

3) Solve for the scattering matrix: $S = Z^{-1} Z^{*}$

with: $Z_{cc'} = \left(k_{c'} a\right)^{-1} \left[ O_c(k_c a)\delta_{cc'} - k_{c'} a \, R_{cc'} \, O'_{c'}(k_{c'} a) \right]$

- Phase shifts are extracted from the scattering matrix elements

$$S = \exp(2i\delta)$$



# Scattering–state algorithm

Energy steps: 1,2,3, … *i*

1) $E = E_i \Rightarrow k_i = \sqrt{\frac{-2\mu E_i}{\hbar^2}}$, $B_i = 0$

2) *Compute*

$$C_{cn,c'n'} = \langle f_n | T_{rel}(r) + L_c(0) + V_{Coul}(r) | f_{n'} \rangle \delta_{cc'} + \langle f_n | V^N_{cc'}(r) | f_{n'} \rangle$$

3) *Compute* $(C - EI)_{cn,cn'} = C_{cn,c'n'} - E\delta_{cc'}\delta_{nn'}$

4) *Invert the matrix:* $C - EI$

5) *Compute (real) matrix* $R_{cc'}$ *& (complex) functions* $O_c(k_i a)$, $O'_c(k_i a)$

6) *Compute and invert (complex) matrix* $Z_{cc'}$

7) *Compute (complex) matrix* $S = Z^{-1}Z^*$

# 程序说明



## 已知np相互作用势，求解np束缚态能量

## 完善lagrange_mesh.f文件

```
c********************************************************************************
      subroutine T_operator(mu)
      ! this subroutine is used to compute the kinetic and Bloch operator
      ! in the matrix form
      ! the formulas in this subroutine used the forms of Sofia's notes of
      ! talent course
c********************************************************************************
      use precision
      use constants
      implicit none
      integer :: ir, irp
      real*8 :: f1,f2,f3
      real*8 ::  xi, xj
      real*8 :: mu


      if(allocated(kinetic_matrix)) deallocate(kinetic_matrix)
      allocate(kinetic_matrix(1:nr, 1:nr))

c********************************************************************************
c********************************************************************************
c********************************************************************************
c*************************完善T_operator的计算程序*************************
c*************************储存在kinetic_matrix中*************************
c********************************************************************************
c********************************************************************************
```

# 程序说明

## 已知np相互作用势，求解np束缚态能量

## 完善lagrange_mesh.f文件

```fortran
c***********************************************************************
       subroutine Bloch_bound(mu,Bc,B_operator)
       ! this subroutine is used to compute the kinetic and Bloch operator
       ! in the matrix form
       ! note for the bound state the Bloch operator is not zero
       ! the formulas in this subroutine used the forms of Sofia's notes of
       ! talent course
c***********************************************************************
       use precision
       use constants
       implicit none
       integer :: ir, irp
       real*8 ::  xi, xj
       real*8 :: mu
       real*8 :: Bc
       real*8 :: f1
       real*8,dimension(1:nr, 1:nr) :: B_operator
       B_operator=0.0_dpreal
c***********************************************************************
c***********************************************************************
c***********************************************************************
c************************完善Blochoperator的计算程序******************************
c************************储存在B_operator中******************************
c***********************************************************************
c***********************************************************************
```

# 程序说明

## 已知np相互作用势，求解np束缚态能量

## 完善lagrange_mesh.f文件

```
c-----------------------------------------------------------------------
c***********************************************************************
      subroutine centrifugal_barrier (l,mu)
      ! this subroutine is used to compute centrifugal_barrier in
      ! lagrange mesh basis
      ! V_l = \frac{\hbarc^2 l (l+1)}{2 * mu * r}
      ! input : mu  in MeV
      !         l
c***********************************************************************
      use constants
      use precision
      implicit none
      integer :: l , ir
      real*8 :: mu ,xi
      if(allocated(l_barrier_matrix)) deallocate(l_barrier_matrix)
      allocate(l_barrier_matrix(1:nr, 1:nr))
      l_barrier_matrix=0.0_dpreal

c***********************************************************************
c***********************************************************************
c***********************************************************************
c************************完善centrifugal_barrier的计算程序****************
c************************储存在l_barrier_matrix中************************
c***********************************************************************
c***********************************************************************
```

# 程序说明

## 已知np相互作用势，求解np束缚态能量

## 完善lagrange_mesh.f文件

```
      subroutine lagrange_V(Vpot)
      ! this subroutine is used to compute the V-matrix in the Lagrange
      ! mesh basis
      ! input V is given in the mesh point in the step of hcm
      ! this subroutine interpolate the Vpot to give the correct mesh se
c************************************************************************
      use interpolation
      use precision
      implicit none
      real*8,dimension(0:irmatch),intent(in) :: vpot
      integer :: ir,irr
      real*8 ::  r
      logical :: nonlocal
      complex*16, dimension(1:nr,1:nr) :: nlpot

      if(allocated(V_matrix)) deallocate(V_matrix)
      allocate(V_matrix(1:nr,1:nr))
      V_matrix=0.0_dpreal

c************************************************************************
c************************************************************************
c************************************************************************
c************************完善V_operator的计算程序************************
c************************储存在V_matrix中********************************
c************************需要对vpot进行内插******************************
c************************************************************************
c************************************************************************
```