

Can MgO and CaO grow on each other?

Group 4

Generated by Doxygen 1.8.11



# Contents

<b>1</b>	<b>Getting started</b>	<b>1</b>
<b>2</b>	<b>Todo List</b>	<b>3</b>
<b>3</b>	<b>Namespace Index</b>	<b>5</b>
3.1	Namespace List . . . . .	5
<b>4</b>	<b>File Index</b>	<b>7</b>
4.1	File List . . . . .	7
<b>5</b>	<b>Namespace Documentation</b>	<b>9</b>
5.1	constants Module Reference . . . . .	9
5.1.1	Detailed Description . . . . .	9
5.1.2	Variable Documentation . . . . .	9
5.1.2.1	dp . . . . .	9
5.1.2.2	pi . . . . .	9
5.1.2.3	tau . . . . .	10
5.2	convex_hull Namespace Reference . . . . .	10
5.2.1	Variable Documentation . . . . .	10
5.2.1.1	convex_hull . . . . .	10
5.2.1.2	data . . . . .	10
5.2.1.3	file_name . . . . .	10
5.2.1.4	hv_new . . . . .	10
5.2.1.5	max_loc . . . . .	10
5.2.1.6	n . . . . .	10
5.2.1.7	theta . . . . .	10
5.2.1.8	theta_max . . . . .	10

<b>6 File Documentation</b>	<b>11</b>
6.1 constants.f90 File Reference . . . . .	11
6.2 convex_hull.py File Reference . . . . .	11
6.3 lattice.f90 File Reference . . . . .	12
6.3.1 Function/Subroutine Documentation . . . . .	12
6.3.1.1 cube_init(L, cube, prop) . . . . .	12
6.3.1.2 lattice . . . . .	12
6.3.1.3 sheet_init(L, V, sheet) . . . . .	12
6.3.1.4 write_cube(cube, L, prop, fileno) . . . . .	13
6.3.1.5 write_sheet(sheet, L, V, fileno) . . . . .	13
6.4 README.md File Reference . . . . .	13
<b>Index</b>	<b>15</b>

# Chapter 1

## Getting started

*Group 4 term 3*

1. Generate a lattice using `lattice.f90`
2. Run the simulation using `run_castep.sh`
3. Find the convex hull using `convex_hull.py`

To run on Physlogin use `run_castep_ebor.sh`.

**For full documentation, see** [latex/refman.pdf](#).

Generate your own copy of this documentation, including an interactive HTML version, by running `makedoc.sh` (requires [doxygen](#)).



## Chapter 2

## Todo List

### Subprogram [lattice](#)

Produce a pair of sheets with a vacuum between them





## Chapter 3

# Namespace Index

### 3.1 Namespace List

Here is a list of all namespaces with brief descriptions:

<a href="#">constants</a>	Module contains definitions of useful constants . . . . .	<a href="#">9</a>
<a href="#">convex_hull</a>	. . . . .	<a href="#">10</a>



## Chapter 4

# File Index

### 4.1 File List

Here is a list of all files with brief descriptions:

<a href="#">constants.f90</a>	.....	<a href="#">11</a>
<a href="#">convex_hull.py</a>	.....	<a href="#">11</a>
<a href="#">lattice.f90</a>	.....	<a href="#">12</a>



## Chapter 5

# Namespace Documentation

### 5.1 constants Module Reference

Module contains definitions of useful constants.

#### Variables

- integer, parameter `dp = selected_real_kind(15, 300)`  
*Double-precision real kind.*
- `real(kind=dp)`, parameter `pi = 4.0_dp*atan(1.0_dp)`  
*The circle constant, pi.*
- `real(kind=dp)`, parameter `tau = 2.0_dp*pi`  
*2\*pi*

#### 5.1.1 Detailed Description

Module contains definitions of useful constants.

#### 5.1.2 Variable Documentation

##### 5.1.2.1 integer, parameter `constants::dp = selected_real_kind(15, 300)`

Double-precision real kind.

Definition at line 8 of file constants.f90.

##### 5.1.2.2 `real(kind=dp)`, parameter `constants::pi = 4.0_dp*atan(1.0_dp)`

The circle constant, pi.

Definition at line 11 of file constants.f90.

5.1.2.3 `real(kind=dp), parameter constants::tau = 2.0_dp*pi`

`2*pi`

Definition at line 12 of file constants.f90.

## 5.2 `convex_hull` Namespace Reference

### Variables

- string `file_name` = 'madeup.dat'
- `data` = `np.genfromtxt(file_name, usecols = (0, 1))`
- `n` = `len(data)`
- `convex_hull` = `open('convex_hull.dat', 'w')`
- int `hv_new` = 0
- float `theta_max` = -2.0
- int `theta` = -1
- `max_loc` = i

### 5.2.1 Variable Documentation

5.2.1.1 `convex_hull.convex_hull = open('convex_hull.dat', 'w')`

Definition at line 18 of file `convex_hull.py`.

5.2.1.2 `convex_hull.data = np.genfromtxt(file_name, usecols = (0, 1))`

Definition at line 12 of file `convex_hull.py`.

5.2.1.3 `string convex_hull.file_name = 'madeup.dat'`

Definition at line 11 of file `convex_hull.py`.

5.2.1.4 `convex_hull.hv_new = 0`

Definition at line 22 of file `convex_hull.py`.

5.2.1.5 `convex_hull.max_loc = i`

Definition at line 35 of file `convex_hull.py`.

5.2.1.6 `convex_hull.n = len(data)`

Definition at line 15 of file `convex_hull.py`.

5.2.1.7 `int convex_hull.theta = -1`

Definition at line 31 of file `convex_hull.py`.

5.2.1.8 `convex_hull.theta_max = -2.0`

Definition at line 27 of file `convex_hull.py`.

## Chapter 6

# File Documentation

### 6.1 constants.f90 File Reference

#### Modules

- module `constants`  
*Module contains definitions of useful constants.*

#### Variables

- integer, parameter `constants::dp` = `selected_real_kind(15, 300)`  
*Double-precision real kind.*
- real(kind=dp), parameter `constants::pi` = `4.0_dp*atan(1.0_dp)`  
*The circle constant, pi.*
- real(kind=dp), parameter `constants::tau` = `2.0_dp*pi`  
*2\*pi*

### 6.2 convex\_hull.py File Reference

#### Namespaces

- `convex_hull`

#### Variables

- string `convex_hull.file_name` = 'madeup.dat'
- `convex_hull.data` = `np.genfromtxt(file_name, usecols = (0, 1))`
- `convex_hull.n` = `len(data)`
- `convex_hull.convex_hull` = `open('convex_hull.dat', 'w')`
- int `convex_hull.hv_new` = 0
- float `convex_hull.theta_max` = -2.0
- int `convex_hull.theta` = -1
- `convex_hull.max_loc` = i

## 6.3 lattice.f90 File Reference

### Functions/Subroutines

- program [lattice](#)  
*Fortran 2003 program to generate an initial lattice.*
- subroutine [sheet\\_init](#) (L, V, sheet)  
*Initialises a thin sheet of randomly-arranged atoms.*
- subroutine [cube\\_init](#) (L, cube, prop)  
*Initialises a cube of randomly-arranged atoms.*
- subroutine [write\\_cube](#) (cube, L, prop, fileno)  
*Writes a generated cube to a CASTEP cell file.*
- subroutine [write\\_sheet](#) (sheet, L, V, fileno)  
*Writes a generated sheet to a CASTEP cell file.*

### 6.3.1 Function/Subroutine Documentation

**6.3.1.1** subroutine `lattice::cube_init` ( integer, intent(in) *L*, integer, dimension(:, :, :), intent(inout), allocatable *cube*, real(kind=dp), intent(inout) *prop* )

Initialises a cube of randomly-arranged atoms.

#### Parameters

in	<i>L</i>	(integer) length of a side of the cube
out	<i>cube</i>	(integer array) the generated cube
in	<i>prop</i>	(real) proportion of metal ions that are calcium

Definition at line 92 of file `lattice.f90`.

**6.3.1.2** program `lattice` ( )

Fortran 2003 program to generate an initial lattice.

**Todo** Produce a pair of sheets with a vacuum between them

This program creates a CASTEP cell file consisting of a random arrangement of Ca and Mg atoms in a crystal structure with oxygen, according to a user specification.

Definition at line 8 of file `lattice.f90`.

**6.3.1.3** subroutine `lattice::sheet_init` ( integer, intent(in) *L*, integer, intent(in) *V*, integer, dimension(:, :, :), intent(inout), allocatable *sheet* )

Initialises a thin sheet of randomly-arranged atoms.



## Parameters

in	<i>L</i>	(integer) length of a side of the sheet
in	<i>V</i>	(integer) total height of the structure
out	<i>sheet</i>	(integer array) the generated sheet

Definition at line 54 of file lattice.f90.

6.3.1.4 subroutine `lattice::write_cube` ( integer, dimension(:, :, :), intent(in), allocatable *cube*, integer, intent(in) *L*, real(kind=dp), intent(in) *prop*, integer, intent(in) *fileno* )

Writes a generated cube to a CASTEP cell file.

## Parameters

in	<i>cube</i>	(integer array) the cube of atoms
in	<i>L</i>	(real) length of a side of the cube
in	<i>prop</i>	(real) proportion of metal ions that are calcium
in	<i>fileno</i>	(integer) the memory unit corresponding to the file to which to write

Definition at line 161 of file lattice.f90.

6.3.1.5 subroutine `lattice::write_sheet` ( integer, dimension(:, :, :), intent(in), allocatable *sheet*, integer, intent(in) *L*, integer, intent(in) *V*, integer, intent(in) *fileno* )

Writes a generated sheet to a CASTEP cell file.

## Parameters

in	<i>sheet</i>	(integer array) the sheet of atoms
in	<i>L</i>	(real) length of a side of the sheet
in	<i>V</i>	(integer) total height of the structure
in	<i>fileno</i>	(integer) the memory unit corresponding to the file to which to write

Definition at line 228 of file lattice.f90.

## 6.4 README.md File Reference



# Index

- constants, [9](#)
  - dp, [9](#)
  - pi, [9](#)
  - tau, [9](#)
- constants.f90, [11](#)
- convex\_hull, [10](#)
  - convex\_hull, [10](#)
  - data, [10](#)
  - file\_name, [10](#)
  - hv\_new, [10](#)
  - max\_loc, [10](#)
  - n, [10](#)
  - theta, [10](#)
  - theta\_max, [10](#)
- convex\_hull.py, [11](#)
- cube\_init
  - lattice.f90, [12](#)
- data
  - convex\_hull, [10](#)
- dp
  - constants, [9](#)
- file\_name
  - convex\_hull, [10](#)
- hv\_new
  - convex\_hull, [10](#)
- lattice
  - lattice.f90, [12](#)
- lattice.f90, [12](#)
  - cube\_init, [12](#)
  - lattice, [12](#)
  - sheet\_init, [12](#)
  - write\_cube, [13](#)
  - write\_sheet, [13](#)
- max\_loc
  - convex\_hull, [10](#)
- n
  - convex\_hull, [10](#)
- pi
  - constants, [9](#)
- README.md, [13](#)
- sheet\_init
  - lattice.f90, [12](#)

- tau
  - constants, [9](#)
- theta
  - convex\_hull, [10](#)
- theta\_max
  - convex\_hull, [10](#)
- write\_cube
  - lattice.f90, [13](#)
- write\_sheet
  - lattice.f90, [13](#)