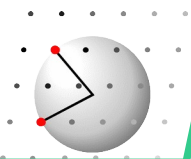


Introduction to syntax



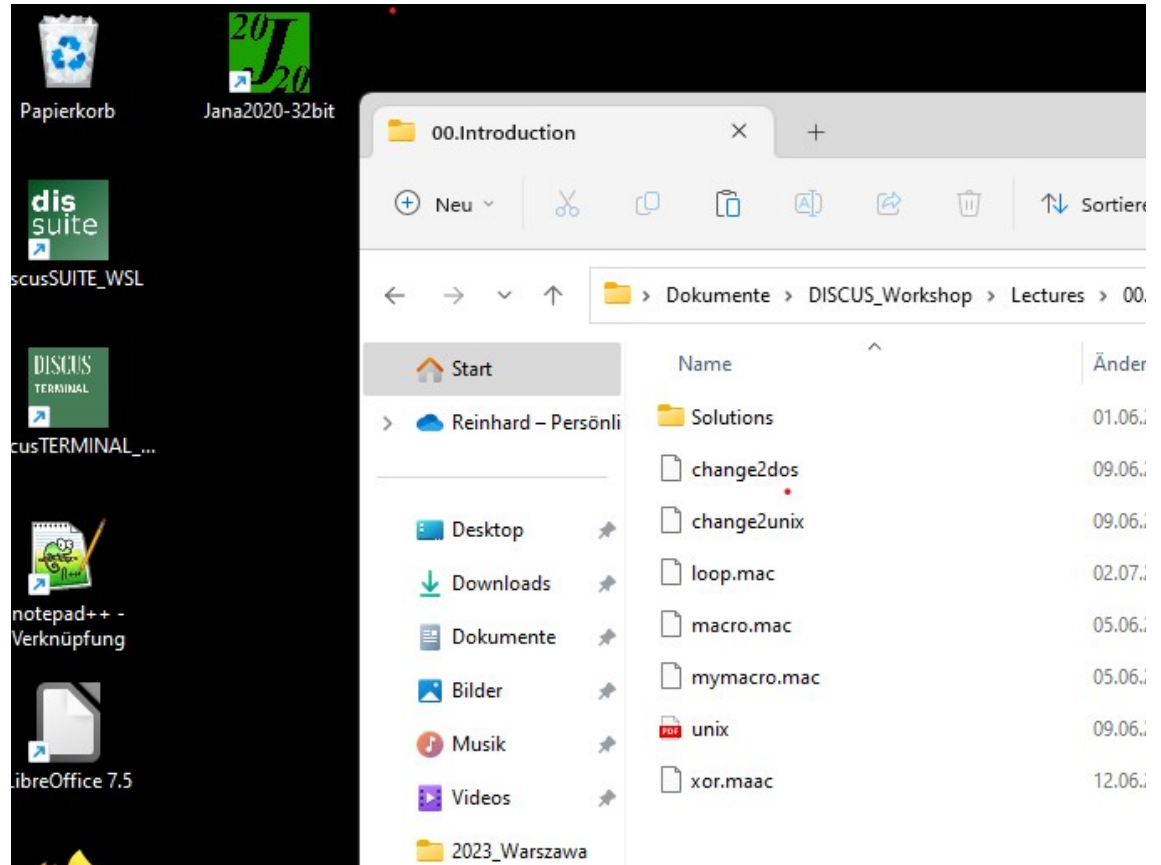
Simulate a crystal structure

You should see:

Open in Windows Explorer:

Lectures\00.Introduction

Start DISCUS_SUITE



Simulate a crystal structure

You should see:

Open in Windows Explorer:

Lectures\00.Introduction

Start DISCUS_SUITE

DISCUS_SUITE
needs to be in correct folder

type:

```
*****  
*          D I S C U S - S U I T E  Version 6.15.00          *  
*  
*      OpenMP is active; maximum threads    6                *  
*      Created : 2023-06-27                                   *  
*-----*  
* (c) R.B. Neder  (reinhard.neder@fau.de)                     *  
* (c) T. Proffen  (tproffen@ornl.gov)                          *  
*****  
  
Start directory : C:\Users\neder\  
Access manuals at each section with      : manual  
Access help at each section/menu with    : help  
News at each section/Command_lang in     : help News  
Change font size with                    : CTRL + /CTRL -  
Preferences: Right click + Preferences:  
  
suite >
```



Simulate a crystal structure

You should see:

Open in Windows Explorer:

Lectures\00.Introduction

Start DISCUS_SUITE

DISCUS_SUITE
needs to be in correct folder

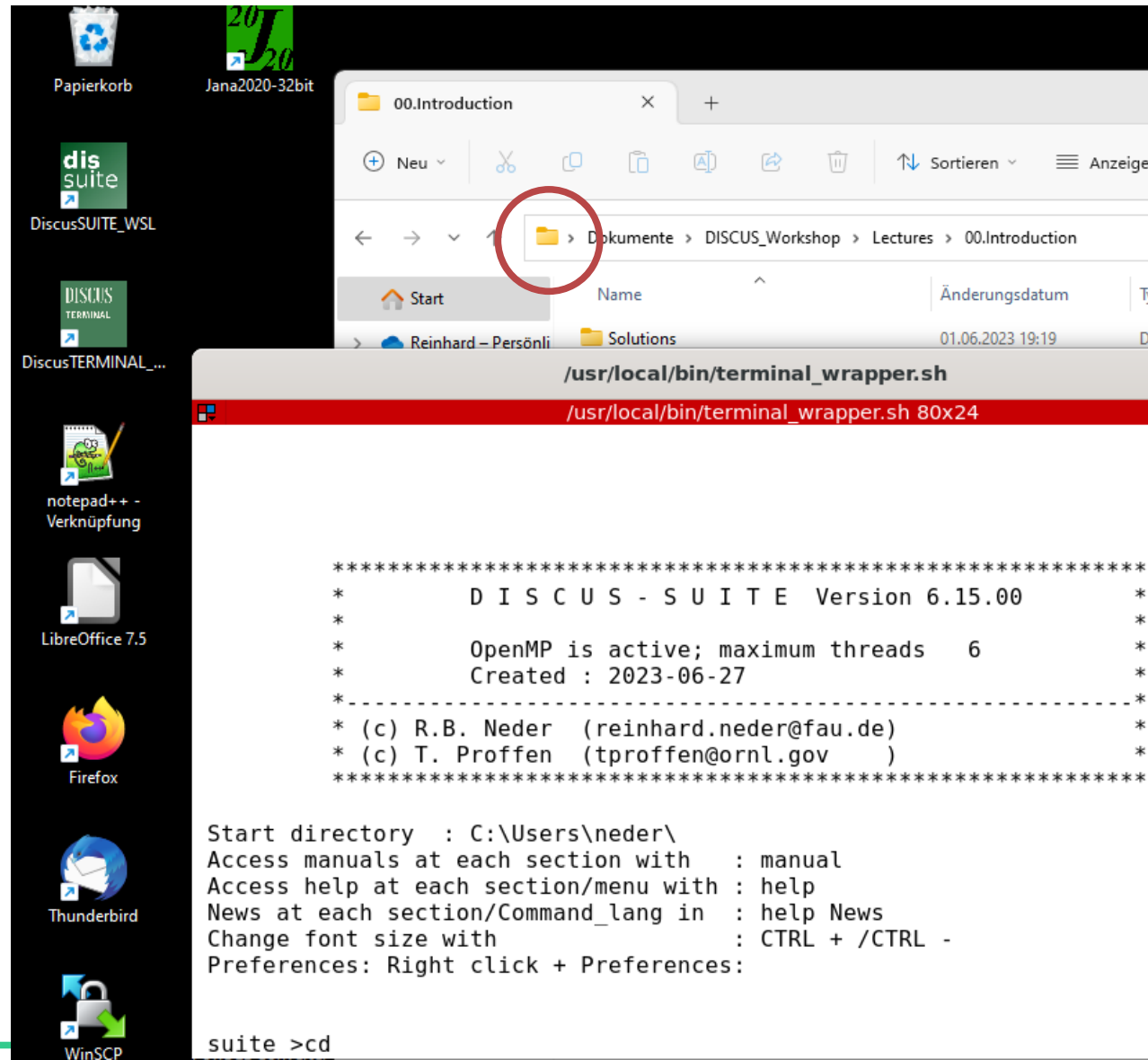
type:

discus > cd

NO RETURN YET !!

Left click on folder icon
Full path will be in blue
Copy with CTRL c

Activate DISCUS_SUITE Window
Paste by SHIFT + right mouse
button



Simulate a crystal structure

You should see:

Open in Windows Explorer:

Lectures\00.Introduction

Start DISCUS_SUITE

DISCUS_SUITE
needs to be in correct folder

type:

discus > cd

NO RETURN YET !!

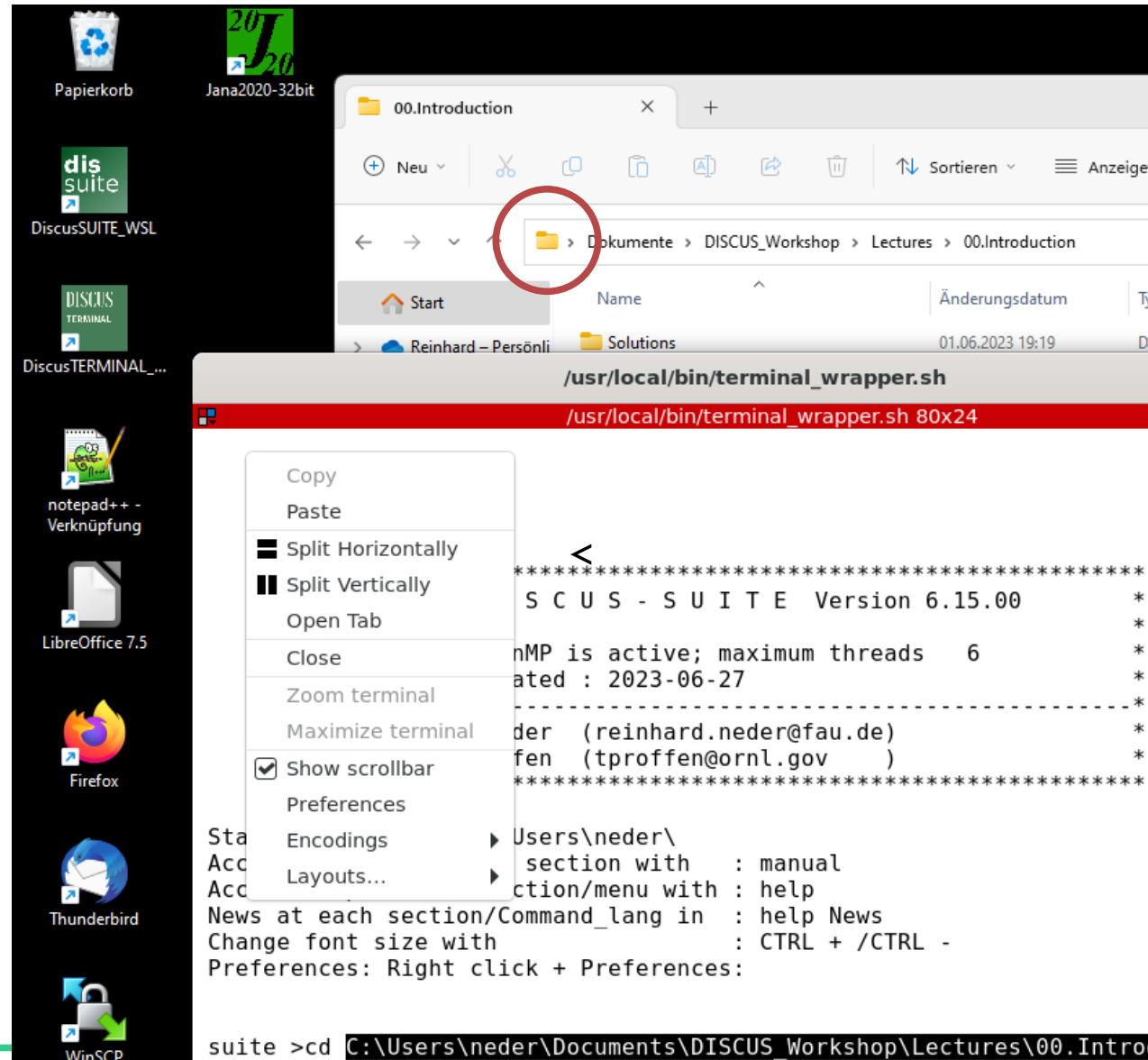
Left click on folder icon

Full path will be in blue

Copy with CTRL c

Activate DISCUS_SUITE Window

Paste by SHIFT + right mouse
button



Simulate a crystal structure

You should see:

Open in Windows Explorer:

Start DISCUS_SUITE

DISCUS_SUITE
needs to be in correct folder

type:

discus > cd

NO RETURN YET !!

Left click on folder icon

Full path will be in blue

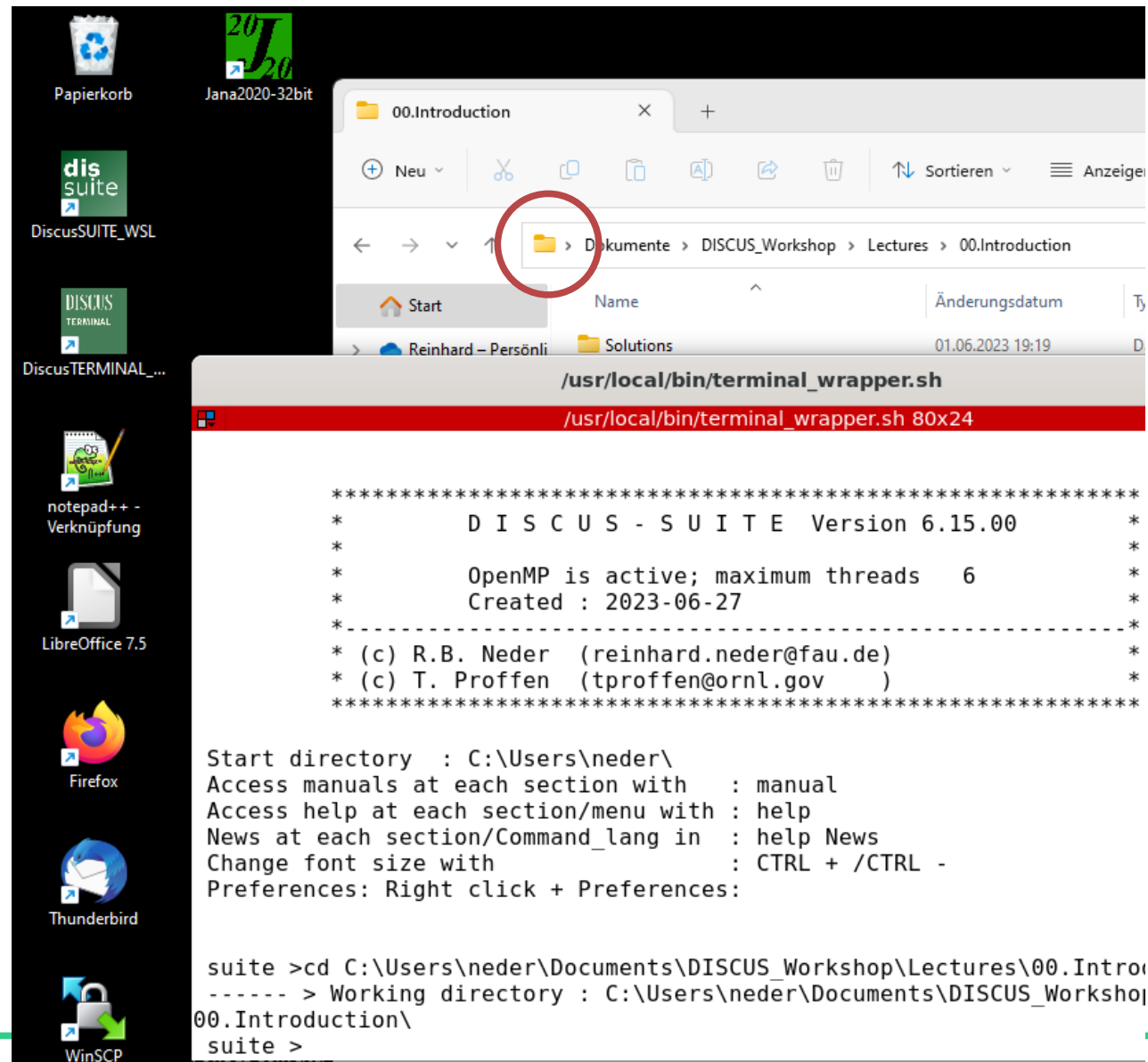
Copy with CTRL c

Activate DISCUS_SUITE Window

Paste by SHIFT + right mouse
button

HIT RETURN

Lectures\00.Introduction



Simulate a crystal structure

You should see:

Open in Windows Explorer:

Start DISCUS_SUITE

DISCUS_SUITE
needs to be in correct folder

type:

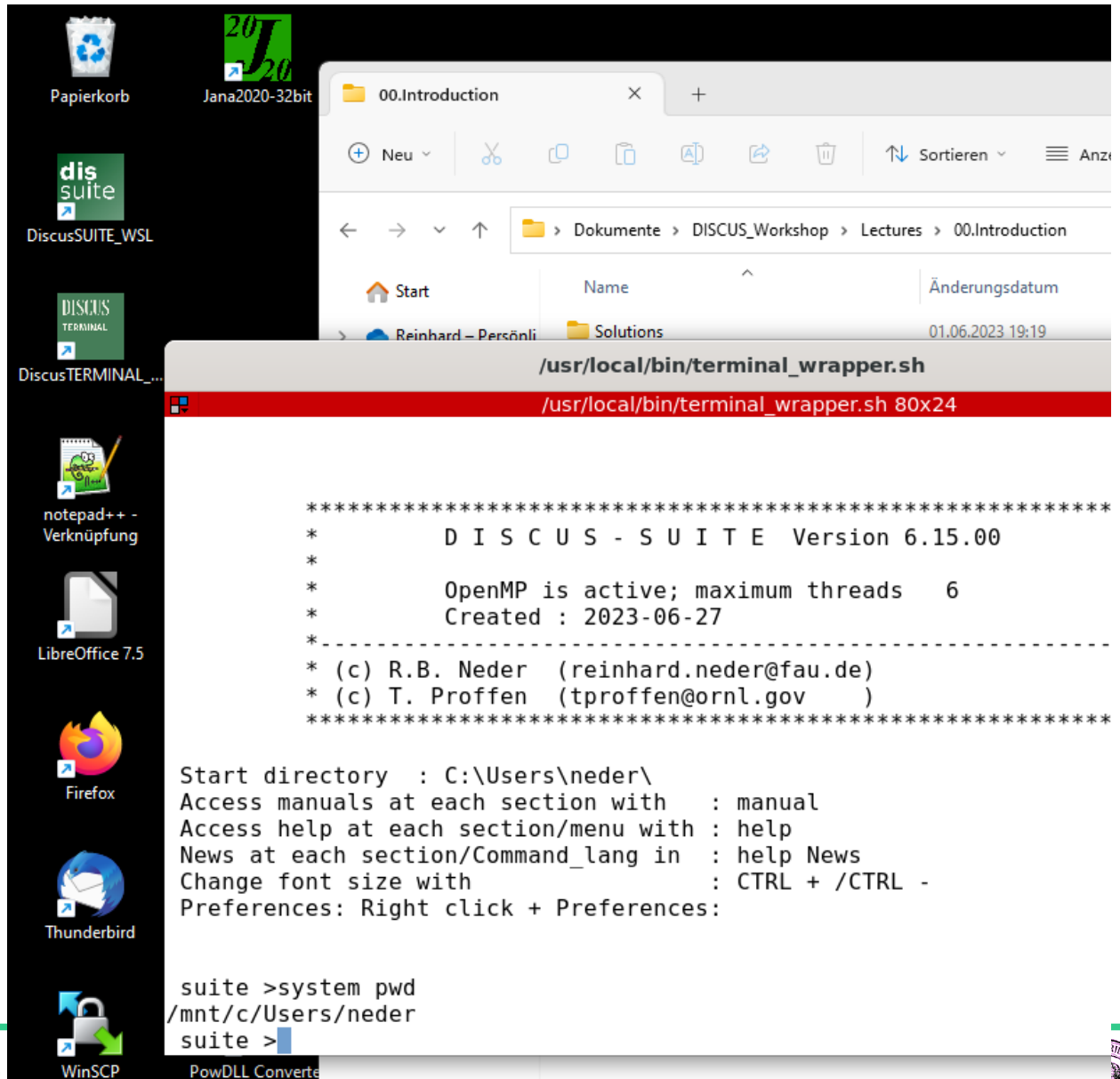
discus > `system pwd`

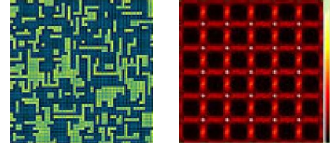
Default folder is:
C:\Users\your_user_name

Within this folder you have:
Downloads
Documents
Etc. etc

Change to a folder via
`cd path_to new_folder`

Lectures\00.Introduction

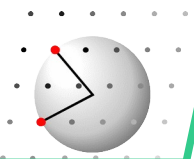


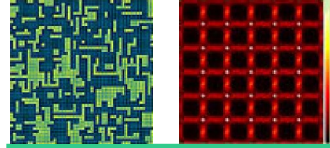


DISCUS, main commands



to start:	Type	<i>discus</i>	at the main discus_suite prompt
help			Enters help menu
help <command>			gives help on specific command leave help by simple return
exit			ends DISCUS section Warning, nothing is saved automatically, DISCUS does NOT ask
@macro			executes commands listed in file „macro.mac“
learn <filename.mac>			Start to learn everything you type into a macro called <filename.mac>
lend			stops the learn process the macro may now be used as: @<filename>
manual section:<name>			Opens the manual on section <name> suite, discus, diffev, kuplot, package, refine





DISCUS, main commands



Syntax of DISCUS commands:

command

just a command name, which may be abbreviated

help
echo
fourier

command parameter, parameter

evaluate 4+5
insert si, 0, 0, 0 , 1.0

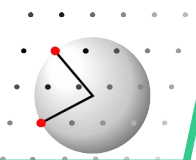
a command name followed by at least one blank
and a list of parameters, all separated by comma

command parameter, parameter # The rest of the line is a comment

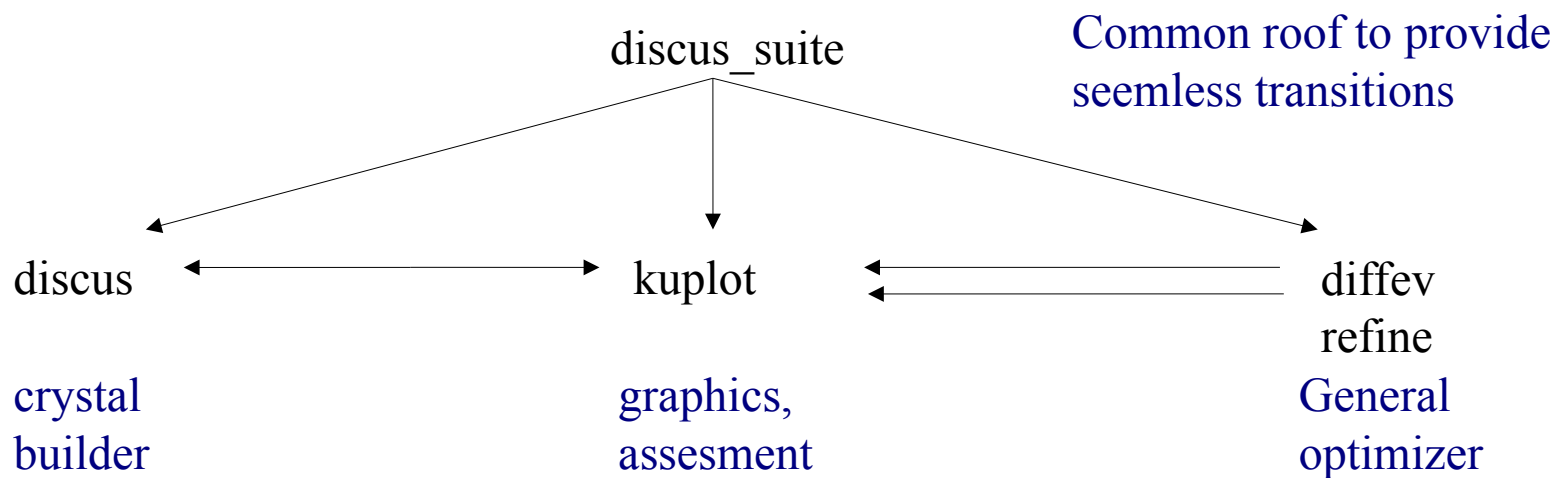
command parameter, parameter ! The rest of the line is a comment

text following a '#' is treated as comment and ignored

text following a '!' is treated as comment and ignored



DISCUS, overview



Build in parallel options MPI /OpenMP; **no** GPU yet

Availability:

Linux, Unix, Mac-OS,
Windows

Single computer
Massive parallel farms



DISCUS, overview



discus

crystal builder

Common tasks structures into menus

Read / Save a structure

read

import

save

plot

Modify a structure

mmc

! Monte-Carlo-modifications

domain

! Introduce (extended) domains into host

stack

! Build crystal with stacking faults

surface

! Cut crystal into finite shape

decorate

! Place molecules onto surface

wave

! Modulated structure

symmetry

! Apply a symmetry operation

shear

! Apply a general affine operation

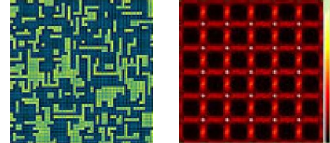
rmc

! Reverse-Monte-Carlo

transform

! Change unit cell description





DISCUS, overview



discus

crystal builder

Common tasks structured into menus

Information on a structure

chem

! Content / average

connect

! Create a connectivity table

property

! Set / clear reaction to properties

Fourier a structure

fourier
powder

! Single crystal diffraction pattern
! powder diffraction pattern

output

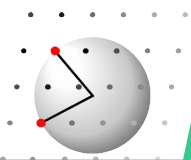
! Write diffraction pattern to disk
! Create powder PDF
! Create 3D- Δ -PDF

patterson
inverse
diff-four

! Patterson function
! Inverse Fourier (single crystal)
! Difference Fourier

(pdf)

! Old, obsolete Powder PDF



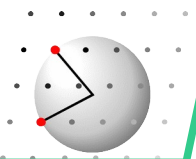
The command language



Variables

SUITE	$i[0] = 3$	$i[0]$ to $i[500]$ is an array of integer variables
DISCUS	$i[500] = i[0] + 2$	
KUPLOT		
DIFFEV	$r[0] = 3.1415$	$r[0]$ to $r[500]$ is an array of real variables
REFINE	$r[500] = -1.4e5$	
	variable real, pi variable real, length, 10.0 variable integer, number	further variables may be defined by the user and used just as the predefined variables
	pi = 3.1415 length = length * 2 number = 4	

DISCUS	n[1] n[2] x[index] y[index] z[index]	Total number of atoms in the crystal number of different atom types x position of atom no. index y position of atom no. index z position of atom no. index
--------	--	--





SUITE $i[0] = 3$
DISCUS $i[500] = i[0] + 2**4$
KUPLOT
DIFFEV $r[0] = 3.1415$
 $r[i[0]] = \cos(r[0])$

 evaluate $\text{acosd}(60.00)$

 variable real, pi
 $\text{pi} = 4*\text{atan}(1.0)$

 evaluate $1./(6.+4.)$

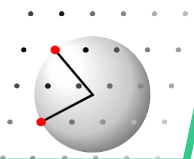
Variables and fixed numbers may be used in arithmetic operations

the usual hierarchy of operation applies

() brackets may be used to change the sequence of operations

$r[0]$ to $r[500]$ is an array of real variables

evaluate just calculates the result of an expression and displays the result on the screen



Homework 1



start a learn process into a file called `homework1.mac`

define an integer valued variable called „product“

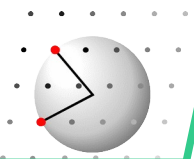
store into this variable the result of the product of numbers 1, 2, 3, 4, 5

evaluate the result of the variable

Stop the learn process

Execute the macro `homework1.mac` to verify your commands

At the discus prompt type: `@homework1`



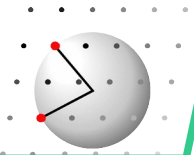
Homework 1 solution



```
variable integer, product
```

```
product = 1 * 2 * 3 * 4 * 5
```

```
evaluate product
```





DISCUS $\sin(\arg)$ $\text{sin}d(\arg)$
KUPLOT $\cos(\arg)$ $\text{cos}d(\arg)$
DIFFEV $\tan(\arg)$ $\text{tan}d(\arg)$

usual trigonometric functions
argument in radian
or in degrees $\text{cos}d(\arg)$

$\text{asin}(\arg)$...

inverse trigonometric functions

$\text{sqrt}(\arg)$
 $\text{exp}(\arg)$
 $\ln(\arg)$
 $\text{abs}(\arg)$

square root of argument
exponential function e^{\arg}
natural logarithm
absolute value

$\text{int}(\arg)$
 $\text{nint}(\arg)$
 $\text{frac}(\arg)$
 $\text{min}(\arg1, \arg2)$

conversion to integer $1.6 \implies 1$
conversion to next integer $1.6 \implies 2$
fractional part $1.6 \implies 0.6$
minimum of the two or more arguments
(also max)

$\text{mod}(\arg1, \arg2)$

modulo function rest after integer division
 $\arg1 - \text{int}(\arg1/\arg2)$

$\text{ran}(\arg)$
 $\text{gran}(\arg)$
 $\text{logn}(\arg, \arg)$
 $\text{pois}(\arg)$

uniform random number
Gaussian distributed random number
lognormaly distributed random number
Poisson distributed random number





DISCUS
KUPLOT
DIFFEV

```
if( condition ) then
    block1
elseif( condition2) then
    block2
...
else
    block3
endif
```

condition:
expression operator expression

operators:

< .lt.
= .le.
== .eq.
>= .ge.
> .gt.

.and.
.or.
.not.

if condition 1 is true then
statements in block 1 are executed
otherwise, if condition 2 is true then
statements in block 2 are executed

otherwise, (no condition was true)
statements in block 3 are executed
final line of an if block

conditions are a logical comparison or
an arithmetic comparison

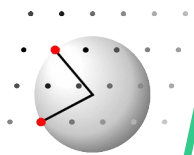
less than; less than or equal; equal;
greater than or equal; greater than

EXAMPLE:

variable real,value
value = ...

...

if(3.14 > asin(value) .or. value == 0.0) then





DISCUS
KUPLOT
DIFFEV

```
do counter= start,end,increment  
    block1  
enddo
```

Loop with fixed number of cycles

```
do while(logical condition)  
    block2  
enddo
```

loop is executed as long as the condition is true

```
do  
    block3  
enddo until (condition)
```

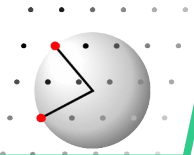
loop is executed until the condition is true

EXAMPLE:

```
do i[0]=1,10,1  
    evaluate i[0]  
enddo
```

EXAMPLE:

```
i[0]=1  
do while (i[0] < 11)  
    evaluate i[0]  
    i[0] = i[0] + 1  
enddo
```



The command language Macros



DISCUS variable integer, counter
KUPLOT variable integer, start, 1
DIFFEV variable integer, ende, 10

do counter = start, ende
 evaluate counter
enddo

@loop.mac

A macro file contains DISCUS commands,
typed as explicitly needed
here in file loop.mac

the „@“ starts reading the commands from
the macro file

MODIFICATION:

@loop.mac 1,10

@loop.mac 2, 99

do counter=\$1,\$2
 evaluate counter
enddo

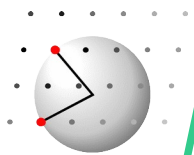
string „\$1“ is replaced by the first parameter

@trigo.mac cosd, 10

@trigo.mac sind, 60

File trigo.mac:

evaluate \$1(\$2)



Homework 2

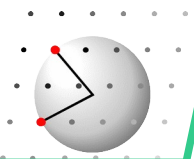


Create a macro that

takes one parameter, a number n

calculates the product $1 * 2 * \dots * n$ and stores this into a variable

evaluates the value of the variable



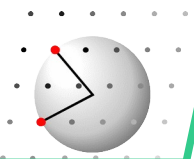
Homework 2



Create a macro that
takes one number as parameter n
calculates the product $1 * 2 * \dots * n$ and stores this into a variable
evaluates the value of the variable

```
variable integer, product
variable integer, loop
variable integer, start
variable integer, finish
product = 1
start = 1
finish = $1
do loop = start, finish
  product = product * loop
enddo

evaluate product
```



Homework 3



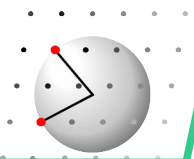
write a macro that

takes one parameter, a real valued number

if the number is larger than zero calculate the square root

if the number is less than zero calculate the square

display the number and the result



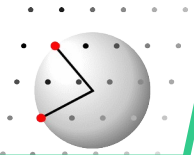
Homework 3



```
variable real, result

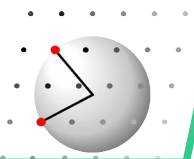
if ( $1 > 0.0) then
    result = sqrt( $1)
elseif ($1 < 0.0) then
    result = $1**2
else
    result = 0.0
endif

evaluate $1
evaluate result
```





DISCUS	<code>date(0)</code>	date in format CCYYMMDDhhmmss.ss
KUPLOT	<code>fdate(0)</code>	Date in format Day Mon DD hh:mm:ss CCYY
DIFFEV		
	<code>fmodt('name')</code>	File modification date
	<code>getcwd()</code>	Current directory / folder
	<code>getenv('name')</code>	Get environment variable
	<code>length(<string>)</code>	Length of character string
	<code>index(<string>, <sub> [, "BACK"])</code>	Position of strign <sub> in <string>
	<code>isvar(<string>)</code>	Is <string> a user variable Yes/No
	<code>isexp(<string>)</code>	Is <string> a valid expression Yes/No



Exercise



Run the macro „loop.mac“ with a single parameter,
an arbitrary whole number

```
suite> @loop.mac 40
```

Run macro with HUGE number

```
suite> @loop.mac 123456789
```

Interrupt with a CTRL-c

