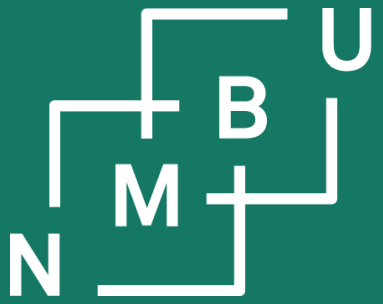
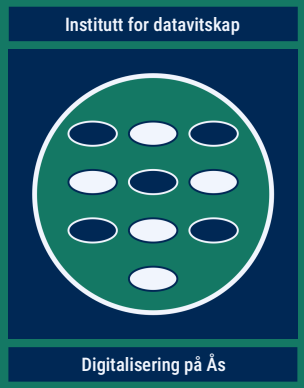


Noregs miljø- og biovitskaplege universitet



INF203

June advanced programming project

3 Uncertainty, validation, and testing

Plan for part 3: Validation

Important: Do you not know your **colloquium** time slot? Let us sort it now!

Thursday, 12th June 2025

- Part 3.1: Unit testing
- Part 3.2: Reproducibility and validation

Informal discussion / chat in room epsilon with …

- Group 1 (at 13:15)
- Group 2 (at 13:30)
- Group 4 (at 13:45)
- Group 6 (at 14:15)
- Group 12 (at 14:30)

Plan for part 3: Validation

Friday, 13th June 2025

- Part 3.3: Formal program analysis
- Part 3.4: Holistic validation
- Part 3.5: Auto- and decorrelation
- Part 3.6: Time series block averaging

Monday, 16th June 2025

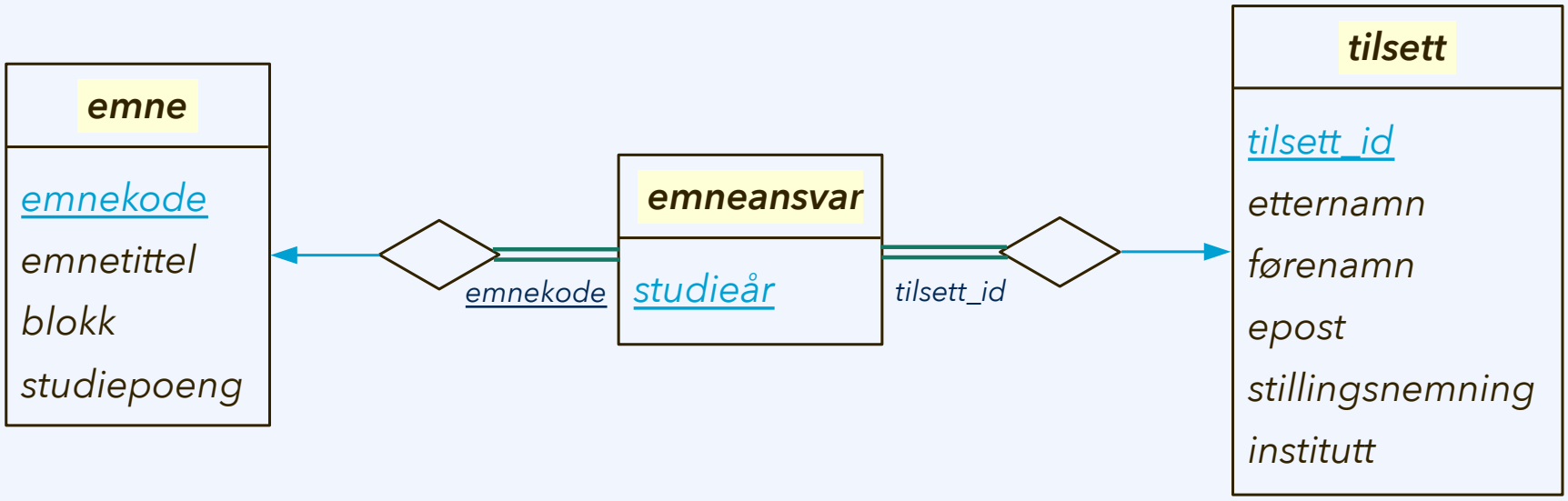
Presentations by …

- Group 1
- Group 2
- Group 4
- Group 6
- Group 12

Submission of the third worksheet by Monday, 16th June 2025, 17.30 CEST.

Questions about E-R diagrams

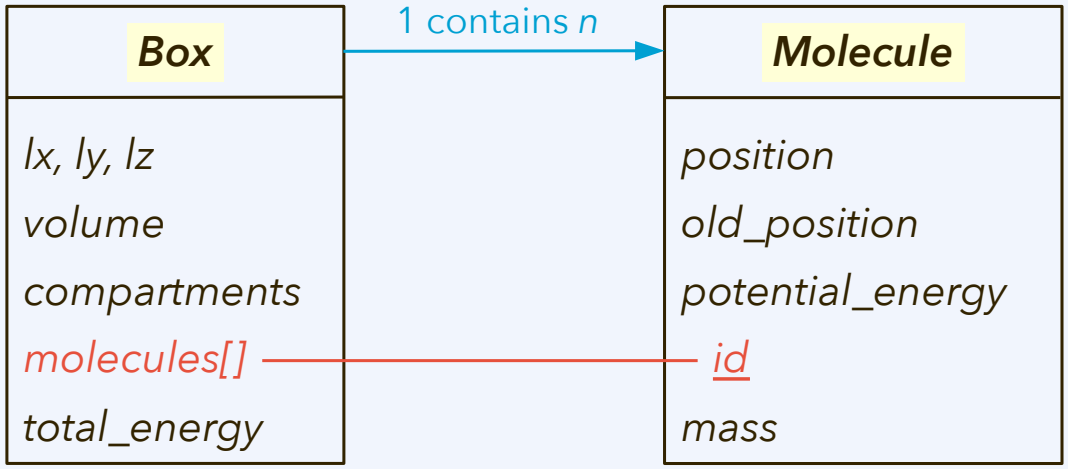
Confer *e.g.* the example:

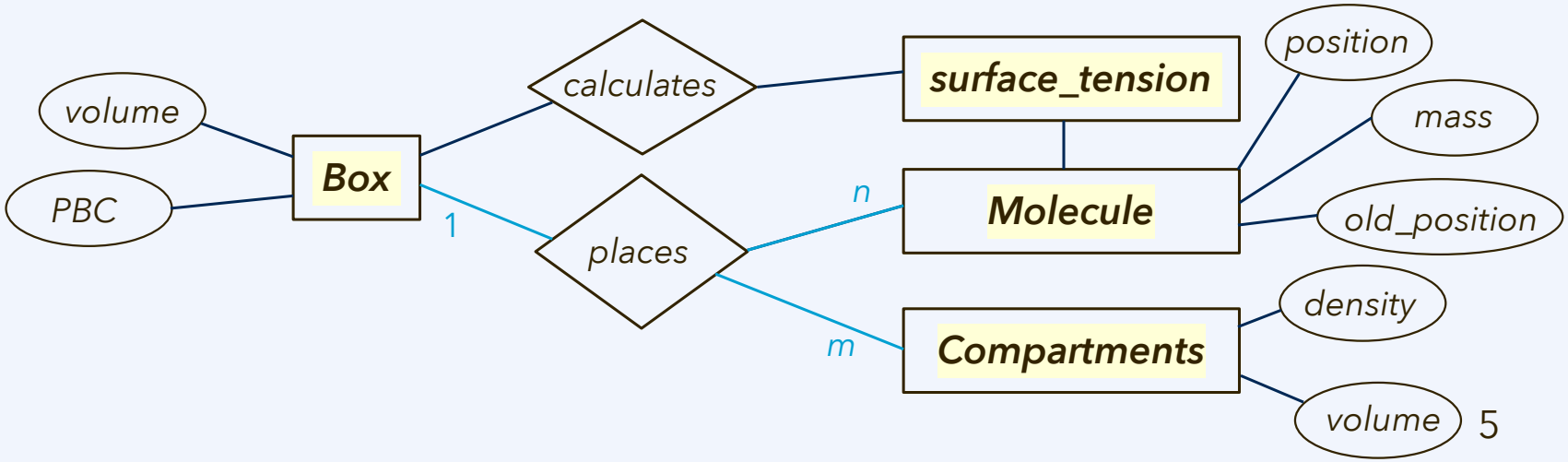


How can we use E-R diagram notation to express our object-oriented data structure design as clearly as possible?

Questions about E-R diagrams

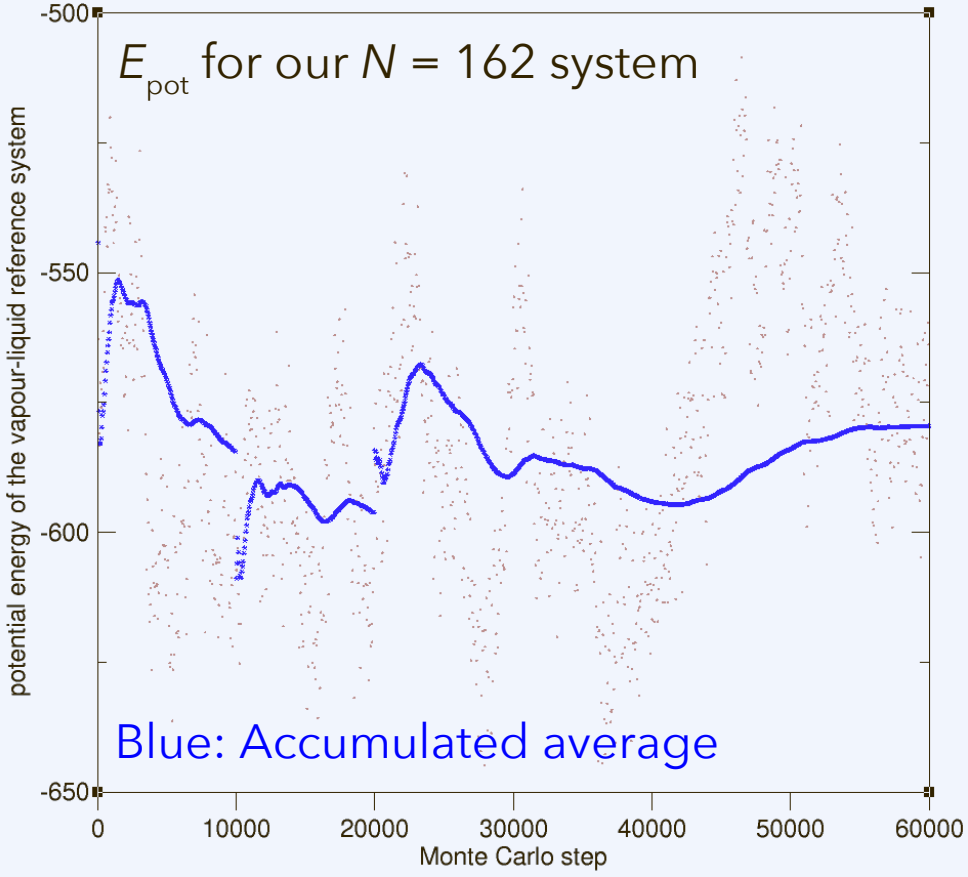
Let us look at some concrete discussion items, *e.g.*, a submission looking like:





Question about potential energy

Yesterday some raised the question: Why is E_{pot} negative – what does it mean?



Thermodynamic property

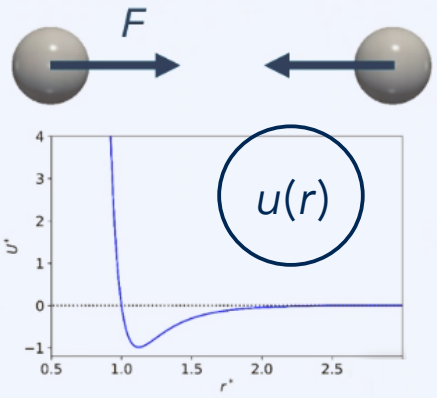
=

Ideal contribution

+

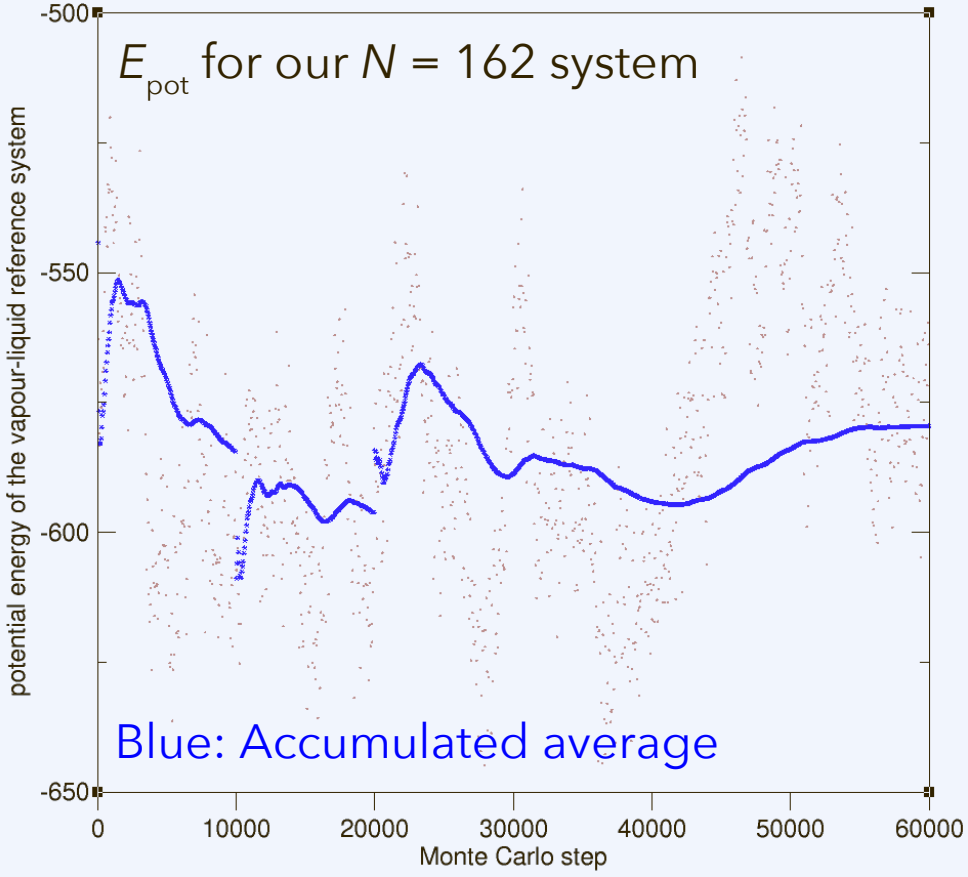
Residual contribution

Modeling of molecular interaction



Question about potential energy

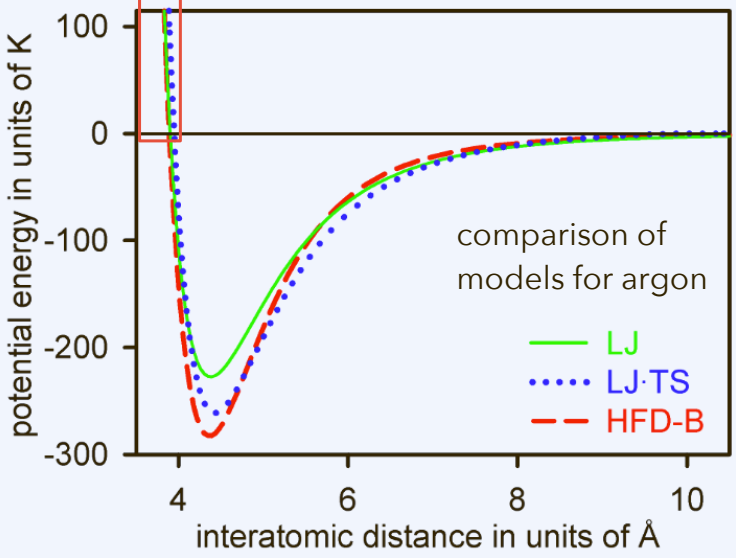
Yesterday some raised the question: Why is E_{pot} negative – what does it mean?



The pair potential $u(r)$ is only positive for extremely short distances. Really, only when there is a collision of atoms!

E_{pot} is the sum over all $u(r)$.

So it is usually negative.



Worksheet no. 3

Norwegian University of Life Sciences

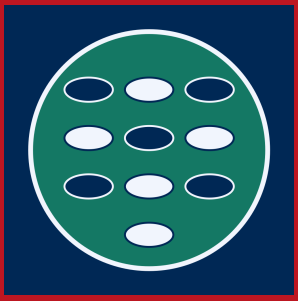
This phase of the project runs until Monday, 16th June 2025, 17.30 CEST.

- File output in XYZ format
- Virtual distortion of the simulation volume
- Implementing the test area method
- Simulation output from the test area method
- Simulation run for the reference system

This is the interesting worksheet, where we actually do something with the MC.

Noregs miljø- og biovitskaplege universitet

Institutt for datavitskap



Digitalisering på Ås

3 Validation

3.1 Unit testing

(slides from INF202)

3.2 Validation & reproducibility

INF203

12th June 2025

Types of tests

Unit tests

Test **one piece of code**, *e.g.*, one method, for right *arguments* → *return value*.

Integration tests

Test concrete interactions between parts of the code, do they fit together?

Acceptance tests

Holistic validation: Run the complete code/system, do y/n correctness checks.

Regression tests

Added once a bug is detected and fixed. Check that the bug does not return.

Library for orchestrating tests

In general, pytest is invoked with the command `pytest` (see below for other ways to invoke pytest). This will execute all tests in all files whose names follow the form `test_*.py` or `\*_test.py` in the current directory and its subdirectories. More generally, pytest follows standard test discovery rules.

Specifying which tests to run

Pytest supports several ways to run and select tests from the command-line or from a file (see below for reading arguments from file).

Run tests in a module

```
pytest test_mod.py
```

Run tests in a directory

```
pytest testing/
```

Run tests by keyword expressions

```
pytest -k 'MyClass and not method'
```

Multiple test inputs using “parametrize”

```
import pytest
from src.packageName.fileName import className

@pytest.mark.parametrize(
        "input1, input2, input3",
        [(1, 2, 3), (1, 3, 4), (3.1, 4.0, 7.1)]
)
def testAddition(input1, input2, input3):
        assert input1 + input2 == pytest.approx(input3)
```

The sequence of terms in the first argument of parametrize must match the parameter names of the function (here, input1 *etc.*).

This is followed by a list of tuples of argument values for the function.

This allows us to define multiple test arguments for the same test; *e.g.*, above we would test whether 1 + 2 = 3, 1 + 3 = 4, 3.1 + 4.0 = 7.1 work correctly.

Check if errors are raised (pytest.raises)

It can make sense to implement unit tests for checking that, in cases where something is going wrong, your code raises the errors you expect it to raise.

Example from INF202:

```
import pytest
from src.packageName.fileName import className

@pytest.mark.parametrize("input1, input2", [(1, 0), (2, 0), (3.1, 0)])
def testDivision(input1, input2):
    with pytest.raises(ZeroDivisionError) as excinfo:
        a = input1 / input2
    assert str(excinfo.value) == "division by zero"
```

Fixtures

Fixtures are execution states (complete, of the whole program, or partial, as contexts relevant to a certain element of the code) for which we **arrange tests**.

Example given by pytest:

```
class Fruit:
    def __init__(self, name):
        self.name = name

    def __eq__(self, other):
        return self.name == other.name
```

```
@pytest.fixture
def my_fruit():
    return Fruit("apple")

@pytest.fixture
def fruit_basket(my_fruit):
    return [Fruit("banana"), my_fruit]
```

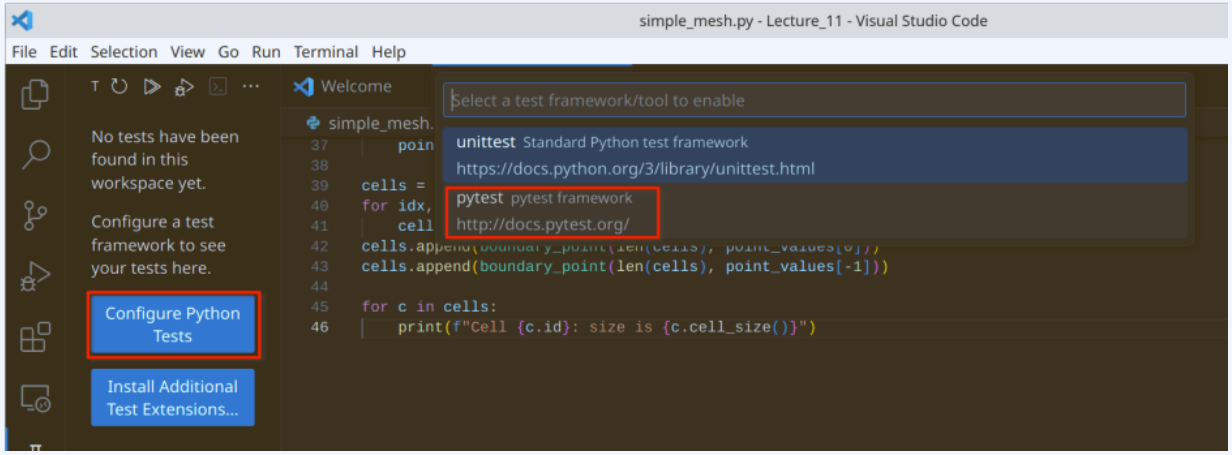
```
def test_my_fruit_in_basket(my_fruit, fruit_basket):
    assert my_fruit in fruit_basket
```

For users of VSCode

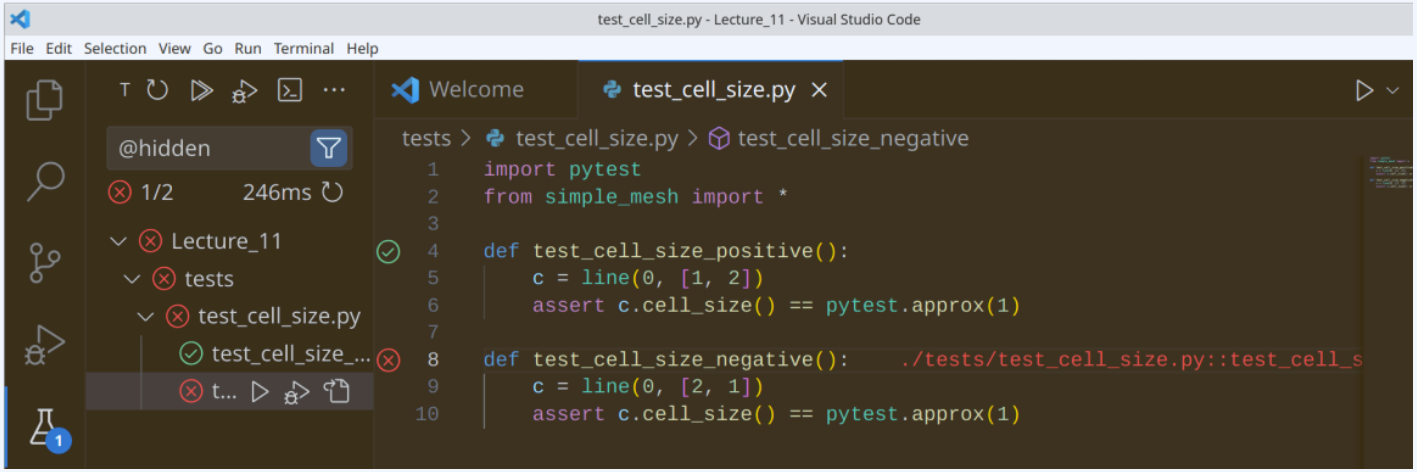
pytest can be difficult to use for VSCode users.

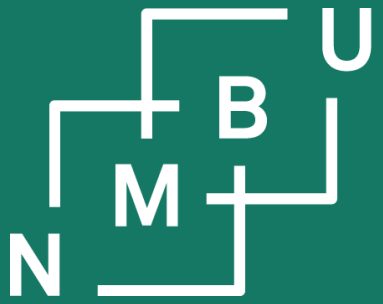
Jonas' recommendations: **1.** Add an empty "**__init__.py**" file so that tests are detected.

Set up test environment

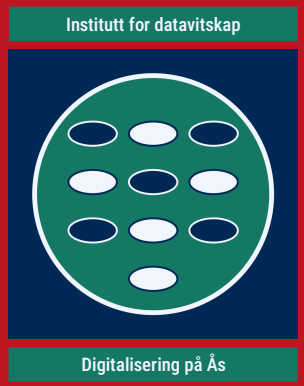


2. To reconfigure tests, run "Python: Configure Tests" in the command palette.





Noregs miljø- og biovitskaplege universitet



3 Validation

3.1 Unit testing (slides from INF202)

3.2 Validation & reproducibility

INF203

12th June 2025

What even is reproducibility?

Reproducibility definitions: Discussed in a review by Plesser.[1]

Consider the case where a validator *b* contradicts findings by *a*:

1) Reseacher *a* did κ and found φ.

2) Reseacher *b* did γ and found $\zeta \neq \varphi$.

[1]H. E. Plesser, *Frontiers Neuroinform.* **11**: 76, doi:10.3389/fninf.2017.00076, **2018**.

What even is reproducibility?

Reproducibility definitions: Discussed in a review by Plesser.[1]

Common formulation and schema for reproducibility claims (RCs):

«Whenever research process κ'' is carried out, it must lead to the outcome φ''.»

Consider the case where a validator *b* contradicts findings by *a*:

1) Reseacher *a* did κ (consistent with κ'') and found φ (consistent with φ'').

   Here, *a* also made the **positive reproducibility claim $\psi = \Box(\varphi'' \mid \kappa'')$.**

2) Reseacher *b* did γ, **consistent with κ''**, and found ζ, **inconsistent with φ''**.

   Here, *b* made the **negative reproducibility claim $\Diamond(\neg\varphi'' \mid \kappa'') \equiv \neg\Box(\varphi'' \mid \kappa'') \equiv \neg\psi$.**

3) What is relevant there is the **contradiction between ψ and $\neg\psi$**.

Claim ψ is usually implicit, ascribed to *a* based on unwritten community rules.[2]

[1]H. E. Plesser, *Frontiers Neuroinform.* **11**: 76, doi:10.3389/fninf.2017.00076, **2018**.

[2]In *Proc. FOIS 2023*, *pp.* 302–317, doi:10.3233/faia231136, **2023**.

Case study on claims made in molecular modelling

Epistemic metadata and their documentation were explored for the domain of molecular modelling and simulation within engineering thermodynamics:

First stage report (10 cases), doi:10.5281/zenodo.7516532, **2023**.

Discussion of *five papers each* from *two research groups* (London, Berlin) without involving the papers' authors. Obtained a tentative **taxonomy for epistemic metadata**, later implemented into the PIMS-II ontology.

Second stage report (12 claims), doi:10.5281/zenodo.7608074, **2023**.

Discussion of *two claims each* from *six papers*, with two papers each from three research groups (London, Berlin, Kaiserslautern), involving the papers' authors. Discussed aspects such as the **grounding of knowledge claims** with authors.

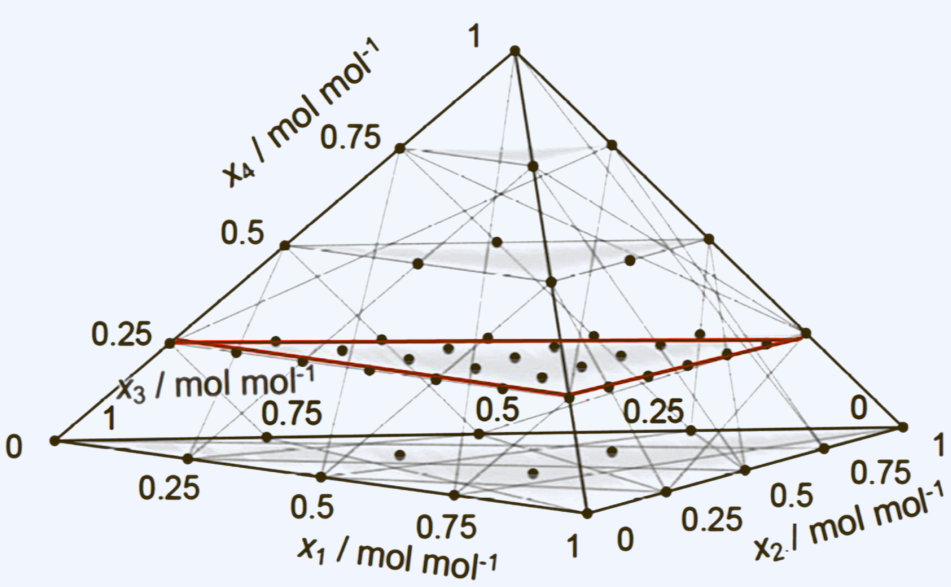
Imperial College London

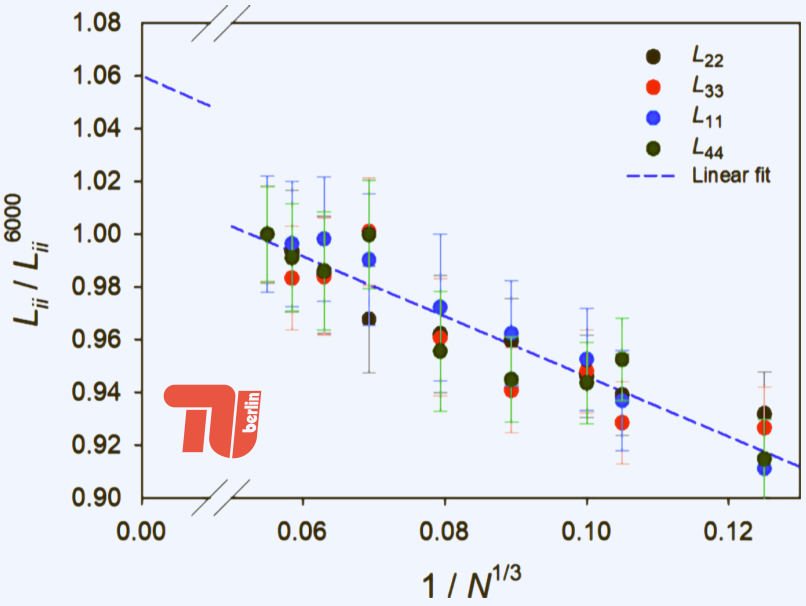
RPTU Rheinland-Pfälzische Technische Universität

Science and Technology Facilities Council

The first stage of the case study

Example: The work by Guevara *et al.*[1] (2020) was considered at both stages.





[1]G. Guevara Carrión, R. Fingerhut, J. Vrabec, «Fick diffusion coefficient matrix of a quaternary liquid mixture by molecular dynamics», *J. Phys. Chem. B* **124**(22): 4527–4535, doi:10.1021/acs.jpcb.0c01625, **2020**.

Guevara *et al.* (2020) paper:[1] First-stage analysis[2]

Question: What is a good methodology for obtaining Fick diffusion coefficients in multicomponent mixtures by [equilibrium molecular dynamics] simulation?

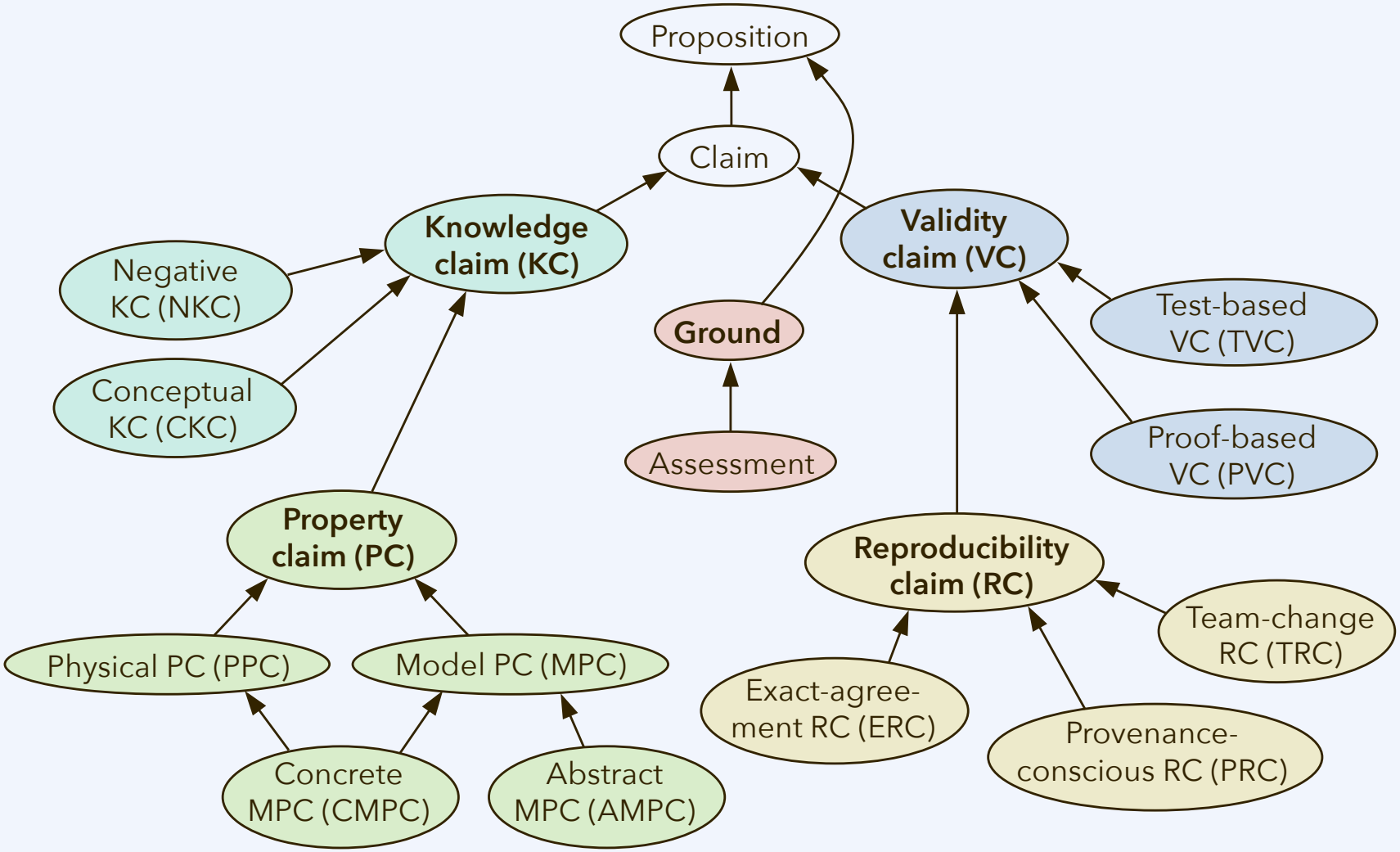
Object of research: The object of research is the Fick diffusion coefficient matrix as such.

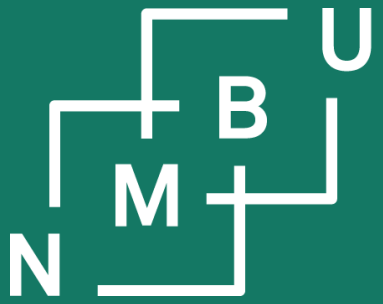
Knowledge claim: […] methodology […] first, the explicit inclusion of a finite-size correction, where it is specifically novel that this correction is applied to the Onsager coefficients, and second, obtaining the Darken correction from [Kirkwood-Buff] integrals.

Grounding: KB part […] validated against "the Wilson excess Gibbs energy model […]" […] not clear what should make us accept the finite-size methodology […]. It yields a correction of 6% […] whereas the "[…] following Yeh and Hummer would have led to corrections of around 15%." It is based on a linear regression in $N^{-1/3}$ […] ad hoc fit.

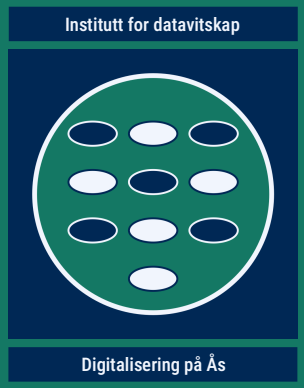
[1]G. Guevara Carrión, R. Fingerhut, J. Vrabec, «Fick diffusion coefficient matrix of a quaternary liquid mixture by molecular dynamics», *J. Phys. Chem. B* **124**(22): 4527–4535, doi:10.1021/acs.jpcb.0c01625, **2020**.

Taxonomy from 2nd stage of case study





Noregs miljø- og biovitskaplege universitet



INF203

June advanced programming project

3 Uncertainty, validation, and testing

3.1 Unit testing
3.2 Reproducibility
3.3 Formal analysis
3.4 Holistic validation
3.5 Autocorrelated data
3.6 Block averaging