

**UNA**

Universität Augsburg  
Mathematisch-Naturwissenschaftlich-  
Technische Fakultät

# Computer aided Analytical Calculations for Physical Many-Body Problems

- Project Work Presentation -

**Jonas Kell**  
Chair for theoretical Physics III

15<sup>th</sup> of Mai 2024

Computer aided calculations

2024-12-08

1. Welcome
2. Presentation of "Project Work" (Projektarbeit)
3. Runs parallel to the "Practical Training" (Fachpraktikum)
4. Presentation in this Group-Slot

# Outline

## 1 Introduction of the problem



### └ Outline

1. Introduction of theme worked on in Project-Work, Practical Training and Master Thesis
  - The mathematical Problem of Many-Body Physics
2. Focus of the report is on the mathematical details. I have a history in Computer-Science, therefore would like here to present some techniques I learned in my work as well as studies as a Computer scientist, that I think might be quite useful to apply in the context of such a working group.
  - The computational/notational problem of "doing maths easily"
3. What was there to make it easier (Off the shelf solutions)
4. What I did:
  - Math Manipulator
  - Tricks to use to improve your Python
  - Latex for presentations
5. Gist: after I (somewhat of a computer scientist by trade) learned the process of being a theoretical physicist, I want to bring back some tools/workflows to maybe improve someones life here at TP III

# Outline

- 1** [Introduction of the problem](#)
- 2** [Solutions: Off the shelf](#)



## └ Outline

1. Introduction of theme worked on in Project-Work, Practical Training and Master Thesis
  - The mathematical Problem of Many-Body Physics
2. Focus of the report is on the mathematical details. I have a history in Computer-Science, therefore would like here to present some techniques I learned in my work as well as studies as a Computer scientist, that I think might be quite useful to apply in the context of such a working group.
  - The computational/notational problem of "doing maths easily"
3. What was there to make it easier (Off the shelf solutions)
4. What I did:
  - Math Manipulator
  - Tricks to use to improve your Python
  - Latex for presentations
5. Gist: after I (somewhat of a computer scientist by trade) learned the process of being a theoretical physicist, I want to bring back some tools/workflows to maybe improve someones life here at TP III

# Outline

## 1 Introduction of the problem

## 2 Solutions: Off the shelf

## 3 Math-Manipulator

- What did I do?
- How can you use it
- Interactive Demonstration



## └ Outline

1. Introduction of theme worked on in Project-Work, Practical Training and Master Thesis
  - The mathematical Problem of Many-Body Physics
2. Focus of the report is on the mathematical details. I have a history in Computer-Science, therefore would like here to present some techniques I learned in my work as well as studies as a Computer scientist, that I think might be quite useful to apply in the context of such a working group.
  - The computational/notational problem of "doing maths easily"
3. What was there to make it easier (Off the shelf solutions)
4. What I did:
  - Math Manipulator
  - Tricks to use to improve your Python
  - Latex for presentations
5. Gist: after I (somewhat of a computer scientist by trade) learned the process of being a theoretical physicist, I want to bring back some tools/workflows to maybe improve someones life here at TP III

# Outline

- 1** **Introduction of the problem**
- 2** **Solutions: Off the shelf**
- 3** **Math-Manipulator**
  - What did I do?
  - How can you use it
  - Interactive Demonstration
- 4** **Custom Python Scripts (SymPy)**



## Computer aided calculations

2024-12-08

### └ Outline

1. Introduction of theme worked on in Project-Work, Practical Training and Master Thesis
  - The mathematical Problem of Many-Body Physics
2. Focus of the report is on the mathematical details. I have a history in Computer-Science, therefore would like here to present some techniques I learned in my work as well as studies as a Computer scientist, that I think might be quite useful to apply in the context of such a working group.
  - The computational/notational problem of "doing maths easily"
3. What was there to make it easier (Off the shelf solutions)
4. What I did:
  - Math Manipulator
  - Tricks to use to improve your Python
  - Latex for presentations
5. Gist: after I (somewhat of a computer scientist by trade) learned the process of being a theoretical physicist, I want to bring back some tools/workflows to maybe improve someones life here at TP III

# Outline

## 1 Introduction of the problem

## 2 Solutions: Off the shelf

## 3 Math-Manipulator

- What did I do?
- How can you use it
- Interactive Demonstration

## 4 Custom Python Scripts (SymPy)

## 5 Presentations? - Custom Beamer Template



### └ Outline

1. Introduction of theme worked on in Project-Work, Practical Training and Master Thesis
  - The mathematical Problem of Many-Body Physics
2. Focus of the report is on the mathematical details. I have a history in Computer-Science, therefore would like here to present some techniques I learned in my work as well as studies as a Computer scientist, that I think might be quite useful to apply in the context of such a working group.
  - The computational/notational problem of "doing maths easily"
3. What was there to make it easier (Off the shelf solutions)
4. What I did:
  - Math Manipulator
  - Tricks to use to improve your Python
  - Latex for presentations
5. Gist: after I (somewhat of a computer scientist by trade) learned the process of being a theoretical physicist, I want to bring back some tools/workflows to maybe improve someones life here at TP III

# Physics of what we calculate

## Introduction of the problem

- Content of my Practical Training (Fachpraktikum)

$$\mathcal{H} = \mathcal{H}_0 + \hat{V}$$

$$\mathcal{H}_0 = U \cdot \sum_l \hat{n}_{l,\uparrow} \hat{n}_{l,\downarrow} + \sum_{l,\sigma} \underbrace{\left( \vec{E} \cdot \vec{r}_l \right)}_{\varepsilon_l} \hat{n}_{l,\sigma}$$

$$\hat{V} = -J \cdot \sum_{\langle l,m \rangle, \sigma} \left( \hat{c}_{l,\sigma}^\dagger \hat{c}_{m,\sigma} + \hat{c}_{m,\sigma}^\dagger \hat{c}_{l,\sigma} \right)$$

2024-12-08

Computer aided calculations  
 └ Introduction of the problem  
 └ Physics of what we calculate

- Content of my Practical Training (Fachpraktikum)

$$\begin{aligned} \mathcal{H} &= \mathcal{H}_0 + \hat{V} \\ \mathcal{H}_0 &= U \cdot \sum_l \hat{n}_{l,\uparrow} \hat{n}_{l,\downarrow} + \sum_{l,\sigma} \left( \vec{E} \cdot \vec{r}_l \right) \hat{n}_{l,\sigma} \\ \hat{V} &= -J \cdot \sum_{\langle l,m \rangle, \sigma} \left( \hat{c}_{l,\sigma}^\dagger \hat{c}_{m,\sigma} + \hat{c}_{m,\sigma}^\dagger \hat{c}_{l,\sigma} \right) \end{aligned}$$

1. Hubbard-Model Hamiltonian with externally applied electric field
2. Hard-Core Bosonic Operators (will be explained in later slide, do not need to elaborate on)
3. Spin-dependent lattice Model
4. External electric field acts symmetric on both spin directions
  - This will be important later, multiple times
  - All terms always need to be symmetric in terms of spin up/down

# Physics of what we calculate

## Introduction of the problem

- Content of my Practical Training (Fachpraktikum)
- Hubbard Model Hamiltonian

$$\mathcal{H} = \mathcal{H}_0 + \hat{V}$$

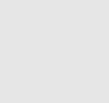
$$\mathcal{H}_0 = U \cdot \sum_l \hat{n}_{l,\uparrow} \hat{n}_{l,\downarrow} + \sum_{l,\sigma} \underbrace{\left( \vec{E} \cdot \vec{r}_l \right)}_{\varepsilon_l} \hat{n}_{l,\sigma}$$

$$\hat{V} = -J \cdot \sum_{\langle l,m \rangle, \sigma} \left( \hat{c}_{l,\sigma}^\dagger \hat{c}_{m,\sigma} + \hat{c}_{m,\sigma}^\dagger \hat{c}_{l,\sigma} \right)$$

Computer aided calculations  
 └ Introduction of the problem  
 └ Physics of what we calculate

2024-12-08

1. Hubbard-Model Hamiltonian with externally applied electric field
2. Hard-Core Bosonic Operators (will be explained in later slide, do not need to elaborate on)
3. Spin-dependent lattice Model
4. External electric field acts symmetric on both spin directions
  - This will be important later, multiple times
  - All terms always need to be symmetric in terms of spin up/down



$$\mathcal{H} = \mathcal{H}_0 + \hat{V}$$

■ Content of my Practical Training (Fachpraktikum)  
 ■ Hubbard Model Hamiltonian

$$\mathcal{H}_0 = U \cdot \sum_l \hat{n}_{l,\uparrow} \hat{n}_{l,\downarrow} + \sum_{l,\sigma} \left( \vec{E} \cdot \vec{r}_l \right) \hat{n}_{l,\sigma}$$

$$\hat{V} = -J \cdot \sum_{\langle l,m \rangle, \sigma} \left( \hat{c}_{l,\sigma}^\dagger \hat{c}_{m,\sigma} + \hat{c}_{m,\sigma}^\dagger \hat{c}_{l,\sigma} \right)$$

$$\mathcal{H} = \mathcal{H}_0 + \hat{V}$$

$$\mathcal{H}_0 = U \cdot \sum_l \hat{n}_{l,\uparrow} \hat{n}_{l,\downarrow} + \sum_{l,\sigma} \underbrace{\left( \vec{E} \cdot \vec{n} \right)}_{\varepsilon_l} \hat{n}_{l,\sigma}$$

$$\hat{V} = -J \cdot \sum_{\langle l,m \rangle, \sigma} \left( \hat{c}_{l,\sigma}^\dagger \hat{c}_{m,\sigma} + \hat{c}_{m,\sigma}^\dagger \hat{c}_{l,\sigma} \right)$$

# Physics of what we calculate

## Introduction of the problem

- Content of my Practical Training (Fachpraktikum)
- Hubbard Model Hamiltonian
- System under influence of external electric field

$$\mathcal{H} = \mathcal{H}_0 + \hat{V}$$

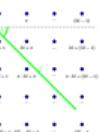
$$\mathcal{H}_0 = U \cdot \sum_l \hat{n}_{l,\uparrow} \hat{n}_{l,\downarrow} + \sum_{l,\sigma} \underbrace{\left( \vec{E} \cdot \vec{n} \right)}_{\varepsilon_l} \hat{n}_{l,\sigma}$$

$$\hat{V} = -J \cdot \sum_{\langle l,m \rangle, \sigma} \left( \hat{c}_{l,\sigma}^\dagger \hat{c}_{m,\sigma} + \hat{c}_{m,\sigma}^\dagger \hat{c}_{l,\sigma} \right)$$

2024-12-08

Computer aided calculations  
 └ Introduction of the problem  
 └ Physics of what we calculate

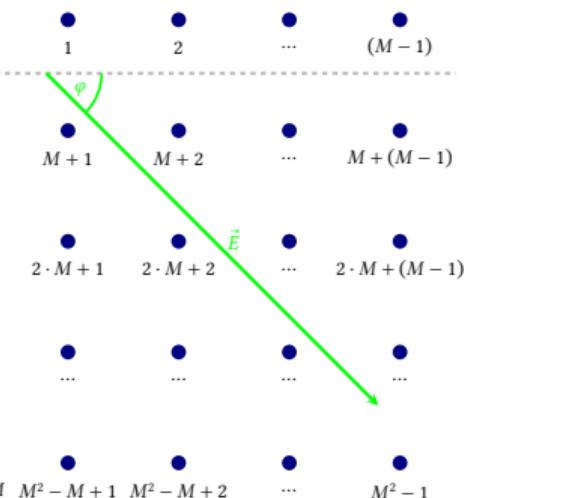
1. Hubbard-Model Hamiltonian with externally applied electric field
2. Hard-Core Bosonic Operators (will be explained in later slide, do not need to elaborate on)
3. Spin-dependent lattice Model
4. External electric field acts symmetric on both spin directions
  - This will be important later, multiple times
  - All terms always need to be symmetric in terms of spin up/down



# Physics of what we calculate

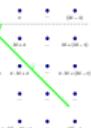
## Introduction of the problem

- 2-dimensional geometry
- Square lattice arrangement
- Computation for general size and field angle



2024-12-08  
Computer aided calculations  
└ Introduction of the problem  
    └ Physics of what we calculate

1. 2-dimensional square geometry
2. Program supports already for general size inputs
3. Thanks to the Monte-Carlo Sampling we can even efficiently evaluate larger systems already

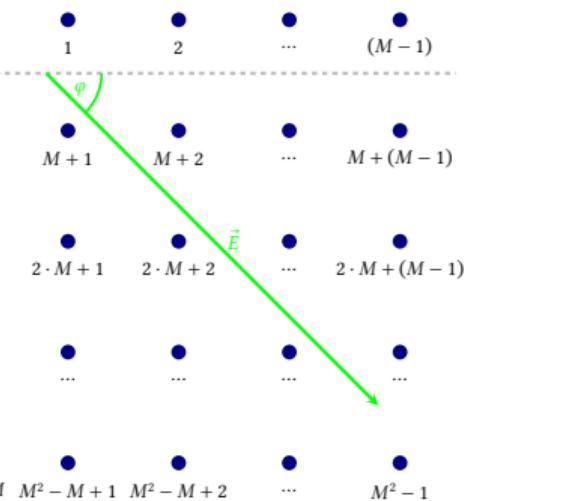


# Physics of what we calculate

## Introduction of the problem

- 2-dimensional geometry
- Square lattice arrangement
- Computation for general size and field angle

**Goal:** Approximate evaluation of time-evolution using Monte-Carlo Sampling



1. 2-dimensional square geometry
2. Program supports already for general size inputs
3. Thanks to the Monte-Carlo Sampling we can even efficiently evaluate larger systems already

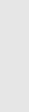
$$\begin{aligned} [\hat{c}_{l,\sigma}, \hat{c}_{l',\sigma'}] &= [\hat{c}_{l,\sigma}^\dagger, \hat{c}_{l',\sigma'}^\dagger] = 0 \\ [\hat{c}_{l,\sigma}, \hat{c}_{l',\sigma'}^\dagger] &= \delta(l,l') \cdot \delta(\sigma,\sigma') \end{aligned}$$

# Physics of what we calculate

## Introduction of the problem

### ■ Hard-Core Bosonic operators

$$\begin{aligned} [\hat{c}_{l,\sigma}, \hat{c}_{l',\sigma'}] &= [\hat{c}_{l,\sigma}^\dagger, \hat{c}_{l',\sigma'}^\dagger] = 0 \\ [\hat{c}_{l,\sigma}, \hat{c}_{l',\sigma'}^\dagger] &= \delta(l,l') \cdot \delta(\sigma,\sigma') \end{aligned}$$



2024-12-08  
Computer aided calculations  
└ Introduction of the problem  
    └ Physics of what we calculate

1. Operators are hard-core bosonic
  - standard commutation relations
  - occupations are only either zero or one
2. fermionic operators (anti-commutation) would work equivalent
3. other notation to rewrite the two spin degrees of freedom into two operators of different dof makes it more readable and comfortable to apply computationally in some cases

$$\begin{aligned} [\hat{c}_{l,\sigma}, \hat{c}_{l',\sigma'}] &= [\hat{c}_{l,\sigma}^\dagger, \hat{c}_{l',\sigma'}^\dagger] = 0 \\ [\hat{c}_{l,\sigma}, \hat{c}_{l',\sigma'}^\dagger] &= \delta(l,l') \cdot \delta(\sigma,\sigma') \end{aligned}$$

# Physics of what we calculate

## Introduction of the problem

### ■ Hard-Core Bosonic operators

- (Would work analogously with Fermionic operators)

$$\begin{aligned} [\hat{c}_{l,\sigma}, \hat{c}_{l',\sigma'}] &= [\hat{c}_{l,\sigma}^\dagger, \hat{c}_{l',\sigma'}^\dagger] = 0 \\ [\hat{c}_{l,\sigma}, \hat{c}_{l',\sigma'}^\dagger] &= \delta(l,l') \cdot \delta(\sigma,\sigma') \end{aligned}$$

Computer aided calculations  
└ Introduction of the problem  
    └ Physics of what we calculate

2024-12-08

1. Operators are hard-core bosonic
  - standard commutation relations
  - occupations are only either zero or one
2. fermionic operators (anti-commutation) would work equivalent
3. other notation to rewrite the two spin degrees of freedom into two operators of different dof makes it more readable and comfortable to apply computationally in some cases

$$\begin{aligned} [\hat{c}_{l,\sigma}, \hat{c}_{l',\sigma'}] &= [\hat{c}_{l,\sigma}^\dagger, \hat{c}_{l',\sigma'}^\dagger] = 0 \\ [\hat{c}_{l,\sigma}, \hat{c}_{l',\sigma'}^\dagger] &= \delta(l,l') \cdot \delta(\sigma,\sigma') \end{aligned}$$

$\hat{c}_{l,\sigma}^{(1)} \leftrightarrow \hat{c}_l^{(1)}$      $\hat{c}_{l,\sigma}^{(1)} \leftrightarrow \hat{d}_l^{(1)}$

$\hat{c}_{l,\sigma}^{(2)} \leftrightarrow \hat{c}_l^{(2)}$      $\hat{c}_{l,\sigma}^{(2)} \leftrightarrow \hat{d}_l^{(2)}$

$H_0 = U \cdot \sum_l \hat{c}_l^\dagger \hat{c}_l \hat{d}_l^\dagger \hat{d}_l + \sum_l \varepsilon_l \hat{c}_l^\dagger \hat{c}_l \sum_l \varepsilon_l \hat{d}_l^\dagger \hat{d}_l$

$\hat{V} = -J \cdot \sum_{\langle l,m \rangle} (\hat{c}_l^\dagger \hat{c}_m + \hat{c}_m^\dagger \hat{c}_l + \hat{d}_l^\dagger \hat{d}_m + \hat{d}_m^\dagger \hat{d}_l)$

# Physics of what we calculate

## Introduction of the problem

### ■ Hard-Core Bosonic operators

- (Would work analogously with Fermionic operators)

### ■ Two spin-degrees of freedom rewritten in alternate notation

$$[\hat{c}_{l,\sigma}, \hat{c}_{l',\sigma'}] = [\hat{c}_{l,\sigma}^\dagger, \hat{c}_{l',\sigma'}^\dagger] = 0$$

$$[\hat{c}_{l,\sigma}, \hat{c}_{l',\sigma'}^\dagger] = \delta(l,l') \cdot \delta(\sigma,\sigma')$$

$$\hat{c}_{l,\uparrow}^{(\dagger)} \leftrightarrow \hat{c}_l^{(\dagger)} \quad \hat{c}_{l,\downarrow}^{(\dagger)} \leftrightarrow \hat{d}_l^{(\dagger)}$$

$$H_0 = U \cdot \sum_l \hat{c}_l^\dagger \hat{c}_l \hat{d}_l^\dagger \hat{d}_l + \sum_l \varepsilon_l \hat{c}_l^\dagger \hat{c}_l \sum_l \varepsilon_l \hat{d}_l^\dagger \hat{d}_l$$

$$\hat{V} = -J \cdot \sum_{\langle l,m \rangle} (\hat{c}_l^\dagger \hat{c}_m + \hat{c}_m^\dagger \hat{c}_l + \hat{d}_l^\dagger \hat{d}_m + \hat{d}_m^\dagger \hat{d}_l)$$

### 1. Operators are hard-core bosonic

- standard commutation relations
- occupations are only either zero or one

### 2. fermionic operators (anti-commutation) would work equivalent

### 3. other notation to rewrite the two spin degrees of freedom into two operators of different dof makes it more readable and comfortable to apply computationally in some cases

# Working on paper: A Computer-Scientists view

## Introduction of the problem

Doing Theoretical Physics often requires lengthy analytical calculations

2024-12-08  
Computer aided calculations  
└ Introduction of the problem  
└ Working on paper: A Computer-Scientists view

1. Comparison of advantages/disatvantages of doing calculations "by hand"
2. Advantages
  - No skills required to learn, writing is natural, every new symbol can just be written
  - Operational Libery (cross out stuff as please, swap whatever without "proof")
  - For completely different approaches no new tool is needed (lern that integral cannot be solved with a specific strategy that you tool doesn't support: out of luck)
  - Hard to loose access to taken notes, produce them without internet/power
3. Disatvantages
  - Many tasks are repetitive, trivial and unfunny to do, paper cannot automate
  - Sign errores, copy errors happen too often
  - Sharing and archiving is non-standart. Many standarts exist for e.g. sharing latex
  - No version-control (git): hard for people that are used to it like me. Cannot just do stuff and later revert if it didn't work
  - Need to write the pretty version in the computer anyway



# Working on paper: A Computer-Scientists view

## Introduction of the problem

Doing Theoretical Physics often requires lengthy analytical calculations

### Advantages of working on paper:

- No barrier to entry (required Skill & technology)
- Maximum liberty how to operate
- Fast iteration for high variation workload
- "Offline" available

1. Comparison of advantages/disadvantages of doing calculations "by hand"
2. Advantages
  - No skills required to learn, writing is natural, every new symbol can just be written
  - Operational Liberty (cross out stuff as please, swap whatever without "proof")
  - For completely different approaches no new tool is needed (lern that integral cannot be solved with a specific strategy that you tool doesn't support: out of luck)
  - Hard to loose access to taken notes, produce them without internet/power
3. Disadvantages
  - Many tasks are repetitive, trivial and unfunny to do, paper cannot automatize
  - Sign errores, copy errors happen too often
  - Sharing and archiving is non-standart. Many standarts exist for e.g. sharing latex
  - No version-control (git): hard for people that are used to it like me. Cannot just do stuff and later revert if it didn't work
  - Need to write the pretty version in the computer anyway

# Working on paper: A Computer-Scientists view

## Introduction of the problem

Doing Theoretical Physics often requires lengthy analytical calculations

### Advantages of working on paper:

- No barrier to entry (required Skill & technology)
- Maximum liberty how to operate
- Fast iteration for high variation workload
- "Offline" available

### Problems of working on paper:

- Repetitive tasks not automizable
- Error-prone and time-consuming to fix mistakes
- Difficult & non-standard to share/archive
- No version-control
- Need to produce final version anyway



## Computer aided calculations

### └ Introduction of the problem

#### └ Working on paper: A Computer-Scientists view

### 1. Comparison of advantages/disadvantages of doing calculations "by hand"

#### 2. Advantages

- No skills required to learn, writing is natural, every new symbol can just be written
- Operational Liberty (cross out stuff as please, swap whatever without "proof")
- For completely different approaches no new tool is needed (lern that integral cannot be solved with a specific strategy that you tool doesn't support: out of luck)
- Hard to loose access to taken notes, produce them without internet/power

#### 3. Disadvantages

- Many tasks are repetitive, trivial and unfunny to do, paper cannot automate
- Sign errores, copy errors happen too often
- Sharing and archiving is non-standart. Many standards exist for e.g. sharing latex
- No version-control (git): hard for people that are used to it like me. Cannot just do stuff and later revert if it didn't work
- Need to write the pretty version in the computer anyway

- No barrier to entry (required Skill & technology)
- Maximum liberty how to operate
- Error-prone and time-consuming to fix mistakes
- Difficult & non-standard to share/archive
- "Offline" available
- Need to produce final version anyway

- Repetitive tasks not automizable
- Error-prone and time-consuming to fix mistakes
- Need to produce final version anyway

# Examples from "Theoretical Solid State Physics"

## Introduction of the problem

$$\begin{aligned}
 [S, H] &= [c_1^t c_2 - c_2^t c_1, \epsilon c_1^t c_1 + \Delta (c_1^t c_2 + c_2^t c_1)] \quad \text{→ } [c_1^t c_2, c_1^t c_1] = 0, \text{ obvious, usu.} \\
 &= \epsilon [c_1^t c_2, c_1^t c_1] - \epsilon [c_2^t c_1, c_1^t c_1] + \Delta [c_1^t c_2, c_1^t c_1] - \Delta [c_2^t c_1, c_1^t c_1] \\
 &\quad \left| \begin{array}{l} c_1^t c_2 + c_1^t c_1 - c_2^t c_1 - c_1^t c_1 = c_1^t c_1 \\ \cancel{+ c_1^t c_1} \quad \cancel{- c_1^t c_1} \end{array} \right| \quad \left| \begin{array}{l} c_1^t c_2 + c_1^t c_1 - c_2^t c_1 - c_1^t c_1 = \\ + c_1^t c_1 \quad \cancel{- c_1^t c_1} \end{array} \right. \\
 &\quad \cancel{+ c_1^t c_1} \quad \cancel{- c_1^t c_1} \\
 &\quad = c_1^t c_1 - c_2^t c_2 - c_2^t c_1 + c_1^t c_2 = \\
 &\quad \cancel{- c_1^t c_1} \quad \cancel{- c_1^t c_1} \\
 &\quad = c_1^t c_1 + c_1^t c_2 - c_2^t c_1 - c_2^t c_2 = c_1^t c_1 - c_2^t c_2 \\
 &\quad \rightarrow \epsilon c_1^t c_2 - \epsilon c_2^t c_1 + 2\Delta (c_1^t c_1 - c_2^t c_1) \\
 &\quad = \epsilon (c_1^t c_2 - c_2^t c_1) + 2\Delta (c_1^t c_1 - c_2^t c_1) \\
 &\quad = \frac{\epsilon}{\Delta} V + 2\Delta (c_1^t c_1 - c_2^t c_1)
 \end{aligned}$$

$$\begin{aligned}
 [S, H] &= [c_1^t c_2 - c_2^t c_1, \epsilon (c_1^t c_1 + \Delta (c_1^t c_2 + c_2^t c_1))] \quad \text{→ } [c_1^t c_2, c_1^t c_1] = 0, \text{ obvious, usu.} \\
 &= \epsilon [c_1^t c_2, c_1^t c_1] - \epsilon [c_2^t c_1, c_1^t c_1] + \Delta [c_1^t c_2, c_1^t c_1] - \Delta [c_2^t c_1, c_1^t c_1] \\
 &\quad \cancel{+ c_1^t c_2 + c_1^t c_1 - c_2^t c_1 - c_1^t c_1} \\
 &\quad = c_1^t c_2 + c_1^t c_1 - c_2^t c_1 - c_1^t c_1 \\
 &\quad = c_1^t c_1 - c_2^t c_2 - c_2^t c_1 + c_1^t c_2 = \\
 &\quad \cancel{- c_1^t c_1} \quad \cancel{- c_1^t c_1} \\
 &\quad = c_1^t c_1 + c_1^t c_2 - c_2^t c_1 - c_2^t c_2 = c_1^t c_1 - c_2^t c_2 \\
 &\quad \rightarrow \epsilon c_1^t c_2 - \epsilon c_2^t c_1 + 2\Delta (c_1^t c_1 - c_2^t c_1) \\
 &\quad = \epsilon (c_1^t c_2 - c_2^t c_1) + 2\Delta (c_1^t c_1 - c_2^t c_1) \\
 &\quad = \frac{\epsilon}{\Delta} V + 2\Delta (c_1^t c_1 - c_2^t c_1)
 \end{aligned}$$

$$\begin{aligned}
 [S, H] &= [c_1^t c_2 - c_2^t c_1, \epsilon (c_1^t c_1 + \Delta (c_1^t c_2 + c_2^t c_1))] \quad \text{→ } [c_1^t c_2, c_1^t c_1] = 0, \text{ obvious, usu.} \\
 &= \epsilon [c_1^t c_2, c_1^t c_1] - \epsilon [c_2^t c_1, c_1^t c_1] + \Delta [c_1^t c_2, c_1^t c_1] - \Delta [c_2^t c_1, c_1^t c_1] \\
 &\quad \cancel{+ c_1^t c_2 + c_1^t c_1 - c_2^t c_1 - c_1^t c_1} \\
 &\quad = c_1^t c_2 + c_1^t c_1 - c_2^t c_1 - c_1^t c_1 \\
 &\quad = c_1^t c_1 - c_2^t c_2 - c_2^t c_1 + c_1^t c_2 = \\
 &\quad \cancel{- c_1^t c_1} \quad \cancel{- c_1^t c_1} \\
 &\quad = c_1^t c_1 + c_1^t c_2 - c_2^t c_1 - c_2^t c_2 = c_1^t c_1 - c_2^t c_2 \\
 &\quad \rightarrow \epsilon c_1^t c_2 - \epsilon c_2^t c_1 + 2\Delta (c_1^t c_1 - c_2^t c_1) \\
 &\quad = \epsilon (c_1^t c_2 - c_2^t c_1) + 2\Delta (c_1^t c_1 - c_2^t c_1) \\
 &\quad = \frac{\epsilon}{\Delta} V + 2\Delta (c_1^t c_1 - c_2^t c_1)
 \end{aligned}$$

- From the lecture "Theoretical Solid State Physics" (Theoretische Festkörperphysik), held by Markus Heyl in WS 22/23
- Sheet 3, Problem 2: Schrieffer-Wolf transformation
- Rather simple calculation, still contains sing mistakes (that here luckily averaged out)

$$\begin{aligned} \hat{H}_0 &= \sum_{\mathbf{k}} \epsilon_{\mathbf{k}} n_{\mathbf{k}} + \frac{U}{N} \sum_{\mathbf{k}_1, \mathbf{k}_2, \mathbf{k}_3, \mathbf{k}_4} \delta(\mathbf{k}_1 - \mathbf{k}_2 + \mathbf{k}_3 - \mathbf{k}_4) \frac{c_{\mathbf{k}_1}^\dagger c_{\mathbf{k}_2} c_{\mathbf{k}_3}^\dagger c_{\mathbf{k}_4}}{\epsilon_{\mathbf{k}_1} - \epsilon_{\mathbf{k}_2} + \epsilon_{\mathbf{k}_3} - \epsilon_{\mathbf{k}_4}} \\ &\quad + \text{higher order terms} \end{aligned}$$

# Examples from "Theoretical Solid State Physics"

## Introduction of the problem

Final transform?

$$c_{4\sigma}^\dagger(t) = e^{iHt} c_{4\sigma}^\dagger e^{-iHt} \rightarrow \text{complicated}$$

$$c_{4\sigma}^\dagger(t) = e^{iHt} c_{4\sigma}^\dagger e^{-iHt} \rightarrow \text{easier}$$

$$c_{4\sigma}^\dagger = e^S c_{4\sigma}^\dagger e^{-S} \approx c_{4\sigma}^\dagger + [S, c_{4\sigma}^\dagger] + O(U^2)$$

$$[S, c_{4\sigma}^\dagger] = \frac{U}{N} \sum'_{\mathbf{k}_1, \mathbf{k}_2, \mathbf{k}_3, \mathbf{k}_4} \delta(\mathbf{k}_1 - \mathbf{k}_2 + \mathbf{k}_3 - \mathbf{k}_4) [c_{4\sigma}^\dagger, c_{4\sigma}^\dagger c_{k_3}^\dagger c_{k_4}, c_{4\sigma}] =$$

$$c_{4\sigma}^\dagger c_{4\sigma}^\dagger c_{4\sigma} c_{4\sigma} + c_{4\sigma}^\dagger c_{4\sigma}^\dagger c_{k_3}^\dagger c_{k_4} + c_{4\sigma}^\dagger c_{k_3}^\dagger c_{k_4} c_{4\sigma}$$

$$c_{4\sigma}^\dagger = e^S c_{4\sigma}^\dagger e^{-S} = c_{4\sigma}^\dagger + \frac{U}{N} \sum'_{\mathbf{k}_1, \mathbf{k}_2, \mathbf{k}_3} \frac{\delta(\mathbf{k}_1 - \mathbf{k}_2 + \mathbf{k}_3 - \mathbf{k}_4)}{\epsilon_{\mathbf{k}_1} - \epsilon_{\mathbf{k}_2} + \epsilon_{\mathbf{k}_3} - \epsilon_{\mathbf{k}_4}} \quad | \text{Change index } k_2 \leftrightarrow k_3, k_4 \leftrightarrow k_3$$

$$= c_{4\sigma}^\dagger + \frac{U}{N} \sum'_{\mathbf{k}_1, \mathbf{k}_2, \mathbf{k}_3} \frac{\delta(\mathbf{k}_1 - \mathbf{k}_2 + \mathbf{k}_3 - \mathbf{k}_4)}{\epsilon_{\mathbf{k}_1} - \epsilon_{\mathbf{k}_2} + \epsilon_{\mathbf{k}_3} - \epsilon_{\mathbf{k}_4}} \delta_{\mathbf{k}_1, \mathbf{k}_2} c_{k_3}^\dagger c_{k_4} \quad (\mathbf{k}_4 \rightarrow \mathbf{k}_1 - \mathbf{k}_2 + \mathbf{k}_3)$$

$$= c_{4\sigma}^\dagger + \frac{U}{N} \sum'_{\mathbf{k}_1, \mathbf{k}_2, \mathbf{k}_3} \frac{1}{\epsilon_{\mathbf{k}_1} - \epsilon_{\mathbf{k}_2} + \epsilon_{\mathbf{k}_3} - \epsilon_{\mathbf{k}_4}} c_{4\sigma}^\dagger c_{k_3}^\dagger c_{k_4} \quad (\mathbf{k}_4 \rightarrow \mathbf{k}_1 - \mathbf{k}_2 + \mathbf{k}_3)$$

$$\bar{H} = \alpha H_u + e^S H c^{-S} = H + [S, H_0] + O(U^2) \quad [\text{S.V}] \rightarrow \text{second order}$$

$$[S, H_0] = \frac{U}{N} \sum'_{\mathbf{k}_1, \mathbf{k}_2, \mathbf{k}_3, \mathbf{k}_4} \frac{\delta(\mathbf{k}_1 - \mathbf{k}_2 + \mathbf{k}_3 - \mathbf{k}_4)}{\epsilon_{\mathbf{k}_1} - \epsilon_{\mathbf{k}_2} + \epsilon_{\mathbf{k}_3} - \epsilon_{\mathbf{k}_4}} [c_{4\sigma}^\dagger c_{4\sigma} c_{k_3}^\dagger c_{k_4}, \sum_{\mathbf{k}} \epsilon_{\mathbf{k}} n_{\mathbf{k}}]$$

$$\rightarrow \text{use } [c_{4\sigma}^\dagger, h_{k_3}^\dagger] = -(\epsilon_{k_1} - \epsilon_{k_2} + \epsilon_{k_3} - \epsilon_{k_4}) c_{4\sigma}^\dagger c_{k_2}^\dagger c_{k_3}^\dagger c_{4\sigma}$$

## Computer aided calculations

### Introduction of the problem

#### Examples from "Theoretical Solid State Physics"

- From the lecture "Theoretical Solid State Physics" (Theoretische Festkörperphysik), held by Markus Heyl in WS 22/23
- Sheet 8, Problem 1: Schrieffer-Wolf transformation of the Hubbard Model
- Just written down steps done by someone else (Tutor/other pupil, often shared the terms to calculate them all, still time)

# Examples from the Practical Training

## Introduction of the problem

- Goal: produce time evolution of operator

$$\hat{V} = -J \cdot \sum_{\langle l,m \rangle} (\hat{c}_l^\dagger \hat{c}_m + \hat{c}_m^\dagger \hat{c}_l + \hat{d}_l^\dagger \hat{d}_m + \hat{d}_m^\dagger \hat{d}_l)$$

2024-12-08

Computer aided calculations  
└ Introduction of the problem  
    └ Examples from the Practical Training

1. Want to calculate the Time-Evolution of the operator  $V$  in the interaction Picture
2. Basically comes down to inserting definitions (however already derived from MM)
3. Distributing and re-ordering is a painful, time-intensive process
4. How to time-evolve operator in the Interaction Picture  $\{\cdot\}(t) = e^{iH_0 t} \{\cdot\} e^{-iH_0 t}$



# Examples from the Practical Training

## Introduction of the problem

- Goal: produce time evolution of operator
- Uses calculation in Interaction Picture

$$\hat{V} = -J \cdot \sum_{\langle l,m \rangle} \left( \hat{c}_l^\dagger \hat{c}_m + \hat{c}_m^\dagger \hat{c}_l + \hat{d}_l^\dagger \hat{d}_m + \hat{d}_m^\dagger \hat{d}_l \right)$$

$$\hat{c}_m^{\dagger I}(t) = e^{i\epsilon_m t} \left( 1 + (e^{iUt} - 1) \hat{d}_m^{\dagger S} \hat{d}_m^S \right) \hat{c}_m^{\dagger S}$$

$$\hat{c}_m^I(t) = e^{-i\epsilon_m t} \left( 1 + (e^{-iUt} - 1) \hat{d}_m^{\dagger S} \hat{d}_m^S \right) \hat{c}_m^S$$

$$\hat{d}_m^{\dagger I}(t) = e^{i\epsilon_m t} \left( 1 + (e^{iUt} - 1) \hat{c}_m^{\dagger S} \hat{c}_m^S \right) \hat{d}_m^{\dagger S}$$

$$\hat{d}_m^I(t) = e^{-i\epsilon_m t} \left( 1 + (e^{-iUt} - 1) \hat{c}_m^{\dagger S} \hat{c}_m^S \right) \hat{d}_m^S$$

Computer aided calculations  
 └ Introduction of the problem  
 └ Examples from the Practical Training

2024-12-08

- Goal: produce time evolution of operator
- Uses calculation in Interaction Picture

$$\begin{aligned}\hat{V} &= -J \cdot \sum_{\langle l,m \rangle} \left( \hat{c}_l^\dagger \hat{c}_m + \hat{c}_m^\dagger \hat{c}_l + \hat{d}_l^\dagger \hat{d}_m + \hat{d}_m^\dagger \hat{d}_l \right) \\ \hat{d}_m^{\dagger S}(t) &= e^{i\omega_m t} \left( 1 + (e^{iUt} - 1) \hat{d}_m^{\dagger S} \hat{d}_m^S \right) \hat{d}_m^S \\ \hat{d}_m^S(t) &= e^{-i\omega_m t} \left( 1 + (e^{-iUt} - 1) \hat{d}_m^{\dagger S} \hat{d}_m^S \right) \hat{d}_m^S \\ \hat{d}_m^S(t) &= e^{i\omega_m t} \left( 1 + (e^{iUt} - 1) \hat{c}_m^{\dagger S} \hat{c}_m^S \right) \hat{d}_m^S \\ \hat{d}_m^S(t) &= e^{-i\omega_m t} \left( 1 + (e^{-iUt} - 1) \hat{c}_m^{\dagger S} \hat{c}_m^S \right) \hat{d}_m^S\end{aligned}$$

# Examples from the Practical Training

## Introduction of the problem

- Goal: produce time evolution of operator
- Uses calculation in Interaction Picture
- Workflow:
  - Inserting Definitions
  - Expanding
  - Ordering
  - Rewriting efficiently

$$\hat{V} = -J \cdot \sum_{\langle l, m \rangle} \left( \hat{c}_l^\dagger \hat{c}_m + \hat{c}_m^\dagger \hat{c}_l + \hat{d}_l^\dagger \hat{d}_m + \hat{d}_m^\dagger \hat{d}_l \right)$$

$$\hat{c}_m^{\dagger I}(t) = e^{i \cdot \varepsilon_m \cdot t} \left( 1 + (e^{i \cdot U \cdot t} - 1) \hat{d}_m^{\dagger S} \hat{d}_m^S \right) \hat{c}_m^{\dagger S}$$

$$\hat{c}_m^I(t) = e^{-i \cdot \varepsilon_m \cdot t} \left( 1 + (e^{-i \cdot U \cdot t} - 1) \hat{d}_m^{\dagger S} \hat{d}_m^S \right) \hat{c}_m^S$$

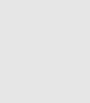
$$\hat{d}_m^{\dagger I}(t) = e^{i \cdot \varepsilon_m \cdot t} \left( 1 + (e^{i \cdot U \cdot t} - 1) \hat{c}_m^{\dagger S} \hat{c}_m^S \right) \hat{d}_m^{\dagger S}$$

$$\hat{d}_m^I(t) = e^{-i \cdot \varepsilon_m \cdot t} \left( 1 + (e^{-i \cdot U \cdot t} - 1) \hat{c}_m^{\dagger S} \hat{c}_m^S \right) \hat{d}_m^S$$

2024-12-08

- Computer aided calculations  
 └ Introduction of the problem  
 └ Examples from the Practical Training

1. Want to calculate the Time-Evolution of the operator V in the interaction Picture
2. Basically comes down to inserting definitions (however already derived from MM)
3. Distributing and re-ordering is a painful, time-intensive process
4. How to time-evolve operator in the Interaction Picture  $\{\cdot\}(t) = e^{iH_0 t} \{\cdot\} e^{-iH_0 t}$



■ Goal: produce time evolution of operator  
 ■ Uses calculation in Interaction Picture  
 ■ Workflow:
 

- Inserting Definitions
- Expanding
- Ordering
- Rewriting efficiently

$$\begin{aligned}\hat{V} &= -J \cdot \sum_{\langle l, m \rangle} \left( \hat{c}_l^{\dagger} \hat{c}_m + \hat{c}_m^{\dagger} \hat{c}_l + \hat{d}_l^{\dagger} \hat{d}_m + \hat{d}_m^{\dagger} \hat{d}_l \right) \\ \hat{d}_m^{\dagger I}(t) &= e^{i \cdot \varepsilon_m \cdot t} \left( 1 + (e^{i \cdot U \cdot t} - 1) \hat{d}_m^{\dagger S} \hat{d}_m^S \right) \hat{d}_m^S \\ \hat{d}_m^I(t) &= e^{-i \cdot \varepsilon_m \cdot t} \left( 1 + (e^{-i \cdot U \cdot t} - 1) \hat{d}_m^{\dagger S} \hat{d}_m^S \right) \hat{d}_m^S \\ \hat{d}_m^{\dagger I}(t) &= e^{i \cdot \varepsilon_m \cdot t} \left( 1 + (e^{i \cdot U \cdot t} - 1) \hat{c}_m^{\dagger S} \hat{c}_m^S \right) \hat{d}_m^S \\ \hat{d}_m^I(t) &= e^{-i \cdot \varepsilon_m \cdot t} \left( 1 + (e^{-i \cdot U \cdot t} - 1) \hat{c}_m^{\dagger S} \hat{c}_m^S \right) \hat{d}_m^S\end{aligned}$$

# Paper-like writing on digital devices

## Solutions: Off the shelf

- Can give benefits that paper lacks
  - Copy/Paste
  - Collaborate
  - Convert to text
  - Searching and ordering large amounts of notes

2024-12-08 Computer aided calculations

└ Solutions: Off the shelf

└ Paper-like writing on digital devices

1. Taking notes digitally can have many benefits
2. OneNote, Xournalpp, Notion and many more examples
3. Still lacks "true" automatization
4. Great danger of Vendor-Locking, losing access
  - Cloud-Driven offers do not work without internet
  - Often times even tied to the device (Apple-Notes I-pad, personal experience with one-note)
  - Countless stories about providers ending support, users getting blocked and more
5. Conclusion: very great choice to boost productivity. But TIP: archive everything as pdf if you ever plan to use it later. It may save you weeks of work/complete loss of data.



# Paper-like writing on digital devices

## Solutions: Off the shelf

- Can give benefits that paper lacks
  - Copy/Paste
  - Collaborate
  - Convert to text
  - Searching and ordering large amounts of notes
- HOWEVER
  - Still no proper automatization
  - Easily access is lost when using proprietary technology
  - "Vendor-Locking"

2024-12-08 Computer aided calculations

└ Solutions: Off the shelf

└ Paper-like writing on digital devices

1. Taking notes digitally can have many benefits
2. OneNote, Xournalpp, Notion and many more examples
3. Still lacks "true" automatization
4. Great danger of Vendor-Locking, losing access
  - Cloud-Driven offers do not work without internet
  - Often times even tied to the device (Apple-Notes I-pad, personal experience with one-note)
  - Countless stories about providers ending support, users getting blocked and more
5. Conclusion: very great choice to boost productivity. But TIP: archive everything as pdf if you ever plan to use it later. It may save you weeks of work/complete loss of data.



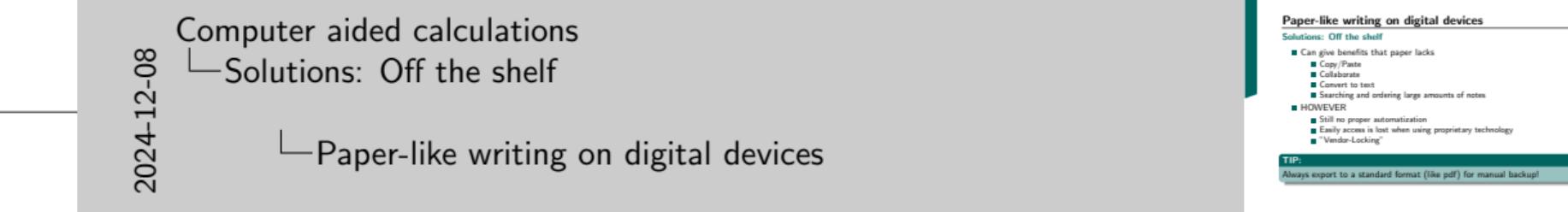
# Paper-like writing on digital devices

## Solutions: Off the shelf

- Can give benefits that paper lacks
  - Copy/Paste
  - Collaborate
  - Convert to text
  - Searching and ordering large amounts of notes
- HOWEVER
  - Still no proper automatization
  - Easily access is lost when using proprietary technology
  - "Vendor-Locking"

### TIP:

Always export to a standard format (like pdf) for manual backup!



1. Taking notes digitally can have many benefits
2. OneNote, Xournalpp, Notion and many more examples
3. Still lacks "true" automatization
4. Great danger of Vendor-Locking, losing access
  - Cloud-Driven offers do not work without internet
  - Often times even tied to the device (Apple-Notes I-pad, personal experience with one-note)
  - Countless stories about providers ending support, users getting blocked and more
5. Conclusion: very great choice to boost productivity. But TIP: archive everything as pdf if you ever plan to use it later. It may save you weeks of work/complete loss of data.



# Computer-Algebra Systems

## Solutions: Off the shelf

- Proprietary Software (Often payed):

- Computer aided calculations
  - Solutions: Off the shelf
    - Computer-Algebra Systems

- Standalone Online Calculators

1. There exist already countless possible attempts at solving the worlds problems
2. All of them are great at what they are designed for, I will not attempt competing with any of the m in terms of any metric except maybe usability (and probably still fail)
3. Chat-GPT is included in the list, because it can do many calculations by now, however is extremely unreliable
4. Central, reoccurring Problems (multiple ore some for everything)
  - Paid access
  - Non-reliable
  - Difficult to "cite" / use in paper/thesis
  - Limited in scope, because software is very spezialized to the task
  - Require training/ prior (export) knowlede to properly take advantage of



# Computer-Algebra Systems

## Solutions: Off the shelf

- Proprietary Software (Often payed):
  - Wolfram-Alpha
  - Mathematica
  - Chat-GPT
  - ...
- Standalone Online Calculators

# Computer-Algebra Systems

## Solutions: Off the shelf

- Proprietary Software (Often payed):
  - Wolfram-Alpha
  - Mathematica
  - Chat-GPT
  - ...
- Standalone Online Calculators
  - Integralrechner.de (Integral-Calculator)
  - Matrix-Calculators
  - ...

- Proprietary Software (Often payed):
  - Wolfram-Alpha
  - Mathematica
  - Chat-GPT
  - ...
- Standalone Online Calculators
  - Integralrechner.de (Integral-Calculator)
  - Matrix-Calculators
  - ...
- ... and many more

# Computer-Algebra Systems

## Solutions: Off the shelf

- Proprietary Software (Often payed):
  - Wolfram-Alpha
  - Mathematica
  - Chat-GPT
  - ...
- Standalone Online Calculators
  - Integralrechner.de (Integral-Calculator)
  - Matrix-Calculators
  - ...
- ... and many more

Computer aided calculations

└ Solutions: Off the shelf

    └ Computer-Algebra Systems

2024-12-08



What is Math-Manipulator?  
Math-Manipulator - What did I do?

# What is Math-Manipulator?

## Math-Manipulator - What did I do?

```
sum((n = 0); 100; int(-inf; inf; (123+(A*4)/100); x))
```

$$\sum_{n=0}^{100} \int_{-\infty}^{\infty} \left( 123 + \frac{(A \cdot 4)}{100} \right) dx$$

[Sel. Parent](#) [Sel. First Child](#) [Sel. Previous Sibling](#) [Sel. Next Sibling](#)

[Replace](#) [Replace All Equal](#) [Replace \(With Export\)](#) [Define Variable](#) [Commute with subsequent](#) [Fold Numbers](#) [Cleanup Terms](#) [Pull Out Minus](#) [Reduce](#)

[Rename With Swap](#) [Pack Same Variables](#) [Replace All With](#)

[Selected Export to Clipboard](#) [Show Export Structure \(UUIDs\)](#) [Show Export Structure](#) [Show Latex](#)

T

$$\sum_{n=0}^{100} \int_{-\infty}^{\infty} (123 + T) dx$$



Computer aided calculations

└ Math-Manipulator

  └ What did I do?

    └ What is Math-Manipulator?

1. I wrote a custom tool that should help with the process of doing math paper-like, but with the help of computers

### 2. Features/Properties

- Relatively smart initial text-input
  - Omit implied multiplication
  - Omit implied zeros
  - Parameter autoboxing
  - Bracket culling
  - → Way less stringent as e.g. input to many Computer-Algebra-Systems requires
- Once initial input is set, operations with mouse
  - Rename, Pack, Move elements with mouse interactions
  - Execute operator-specific operations (distribute, delta/qm-op stuff)
- Many pre-defined Functions/Symbols
- Automatic Macro and Variable system
- In-Browser Help with interactive Playground

# In the Browser

## Math-Manipulator - How can you use it

- Available open-source on GitHub
- Self-hosting possible
- Hosted version available in every modern browser

Computer aided calculations  
└ Math-Manipulator  
  └ How can you use it  
    └ In the Browser

2024-12-08

1. Source code available
2. (Relatively) easily extensible, tutorial and issues available
3. Instantly usable, offline and online



- Available open-source on GitHub
- Self-hosting possible
- Hosted version available in every modern browser

Available here  
<https://jonas-kell.github.io/math-manipulator/>

# In the Browser

## Math-Manipulator - How can you use it

- Available open-source on GitHub
- Self-hosting possible
- Hosted version available in every modern browser

Available here

<https://jonas-kell.github.io/math-manipulator/>

2024-12-08  
Computer aided calculations  
└ Math-Manipulator  
  └ How can you use it  
    └ In the Browser

1. Source code available
2. (Relatively) easily extensible, tutorial and issues available
3. Instantly usable, offline and online



- Offline-Use after installed once
- Store progress in files
- Directly integrated in powerful IDE, and with that version control

# Usage as VS-Code Extension

## Math-Manipulator - How can you use it

- Offline-Use after installed once
- Store progress in files
- Directly integrated in powerful IDE, and with that version control

Computer aided calculations  
└ Math-Manipulator  
  └ How can you use it  
    └ Usage as VS-Code Extension

2024-12-08

1. For more serious use, integrated in the code-editor VS-Code as an Extension
2. Allows for all features, but includes
  - Offline use out of the box
  - Storage of files (with that version control)
  - Multiple parallel projects
3. Has already taken entry into my personal workflow by now
  - Probably I will never make back my WHOLE time-investment
  - But I already saved hours upon hours when I noticed sign mistakes, symmetry issues or similar in calculations I did with it



# Usage as VS-Code Extension

## Math-Manipulator - How can you use it

- Offline-Use after installed once
- Store progress in files
- Directly integrated in powerful IDE, and with that version control

### Note

Do not allow a spellchecker to run on the file, it will break everything performance-wise

Computer aided calculations

└ Math-Manipulator

  └ How can you use it

    └ Usage as VS-Code Extension

2024-12-08

Usage as VS-Code Extension  
Math-Manipulator - How can you use it

- Offline-Use after installed once
- Store progress in files
- Directly integrated in powerful IDE, and with that version control

**Note**  
Do not allow a spellchecker to run on the file, it will break everything performance-wise

$$\begin{aligned} &= \epsilon [c_1^t c_2, c_1^t c_2] - \epsilon [c_2 + c_3 - c_1^t c_1] + D [c_1^t c_2, c_2^t c_3] - D [c_1^t c_3, c_3^t c_2] \\ &= \epsilon (c_1^t c_2 - c_2^t c_1) + 2D (c_1^t c_3 - c_3^t c_2) \end{aligned}$$

$$\begin{aligned} &[c_{41}^t c_{42}^t c_{43}^t c_{44}^t, \sum_{k=1}^4 \epsilon_k u_{kk}] \\ &- (c_{41} - \epsilon_{42} - \epsilon_{43} - \epsilon_{44}) c_{41}^t c_{42}^t c_{43}^t c_{44}^t \end{aligned}$$

# Demo Problems from before

## Math-Manipulator - Interactive Demonstration

$$\begin{aligned} &= \epsilon [c_1^t c_2, c_1^t c_2] - \epsilon [c_2 + c_3 - c_1^t c_1] + D [c_1^t c_2, c_2^t c_3] - D [c_1^t c_3, c_3^t c_2] \\ &= \epsilon (c_1^t c_2 - c_2^t c_1) + 2D (c_1^t c_3 - c_3^t c_2) \end{aligned}$$

$$\begin{aligned} &[c_{41}^t c_{42}^t c_{43}^t c_{44}^t, \sum_{k=1}^4 \epsilon_k u_{kk}] \\ &- (c_{41} - \epsilon_{42} - \epsilon_{43} - \epsilon_{44}) c_{41}^t c_{42}^t c_{43}^t c_{44}^t \end{aligned}$$

Computer aided calculations  
└ Math-Manipulator  
  └ Interactive Demonstration  
    └ Demo Problems from before

2024-12-08

1. Go to Demo in VS-Code folder math-manipulator-calculations in the presentation subfolder
2. Think of disabling spellchecker
3. Think to first split deltas before evaluating !!!!!
4. Macros and other optimizations are possible, but not really necessary
5. Input for simple demo: distribute, cleanup, order op string,  
 $(x+y)(4*x -z+y )(-z + 1+ x)$
6. Input for simple operators:

```
fc("") 1)fa("") 1) ;|; fc("") 1)fa("") m) ;|; fc("") 1)fa("") 2) ;|; fa("") 2)fc("") 2);|; fc("") 1)fa("") 1)fc("") 3)fa("") 3)fa("") 2)fc("") 1)
```

# Usage for the Practical Training

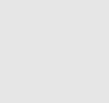
## Math-Manipulator - Interactive Demonstration

$$\begin{aligned} \hat{V}^I(t) &= \left\{ \hat{V}^S \right\}(t) \stackrel{2.11}{=} -J \cdot \sum_{[l,m]} \left\{ \left( \hat{c}_l^{\dagger S} \hat{c}_m^S + \hat{d}_l^{\dagger S} \hat{d}_m^S \right) \right\}(t) \\ &= -J \cdot \sum_{[l,m]} \left( \hat{c}_l^{\dagger l}(t) \hat{c}_m^l(t) + \hat{d}_l^{\dagger l}(t) \hat{d}_m^l(t) \right) \\ &\stackrel{MM}{=} -J \cdot \sum_{[l,m]} \left[ e^{i(\varepsilon_m - \varepsilon_l)t} \cdot \hat{V}_{\text{Part A}}(l, m) + e^{i(\varepsilon_m - \varepsilon_l + U)t} \cdot \hat{V}_{\text{Part B}}(l, m) + e^{i(\varepsilon_m - \varepsilon_l - U)t} \cdot \hat{V}_{\text{Part C}}(l, m) \right] \end{aligned}$$

Computer aided calculations  
 └ Math-Manipulator  
 └ Interactive Demonstration  
 └ Usage for the Practical Training

2024-12-08

1. Go to Demo in VS-Code submodule mm-calculations.
2. Think of disabling spellchecker
3. Show process of how V is calculated
4. Story: at the end at first result was no longer symmetrical in up and down, but it should have been. Mistake was in the very first input. But because of the upfront configuration (and in my case programming) effort, re-calculating the endresult took 20min instead of one week



$$\hat{V}(t) = \left\{ \hat{V}^S \right\}(t) \stackrel{2.11}{=} -J \cdot \sum_{[l,m]} \left\{ \left( \hat{c}_l^{\dagger S} \hat{c}_m^S + \hat{d}_l^{\dagger S} \hat{d}_m^S \right) \right\}(t)$$

$$\hat{V}_{\text{Part A}}(l, m) \stackrel{\text{MM}}{=} (1 \cdot \hat{c}_l^S \hat{c}_m^{\dagger S}) + (1 \cdot \hat{d}_l^S \hat{d}_m^{\dagger S}) + (2 \cdot \hat{c}_l^S \hat{c}_m^S \hat{c}_l^{\dagger S} \hat{c}_m^{\dagger S} \hat{d}_l^S \hat{d}_m^{\dagger S}) + \\ + (-3 \cdot \hat{c}_l^S \hat{c}_l^{\dagger S} \hat{d}_m^S \hat{d}_m^{\dagger S}) + (-3 \cdot \hat{c}_m^S \hat{c}_m^{\dagger S} \hat{d}_l^S \hat{d}_l^{\dagger S}) + (-3 \cdot \hat{c}_l^S \hat{c}_m^S \hat{d}_l^S \hat{d}_l^{\dagger S}) + (-3 \cdot \hat{c}_l^S \hat{c}_m^{\dagger S} \hat{d}_m^S \hat{d}_m^{\dagger S})$$

$$\hat{V}_{\text{Part B}}(l, m) \stackrel{\text{MM}}{=} (-2 \cdot \hat{c}_l^S \hat{c}_m^{\dagger S}) + (-2 \cdot \hat{d}_l^S \hat{d}_m^{\dagger S}) + (-1 \cdot \hat{c}_l^S \hat{c}_m^S \hat{c}_l^{\dagger S} \hat{c}_m^{\dagger S} \hat{d}_l^S \hat{d}_m^{\dagger S}) + \\ + (1 \cdot \hat{c}_l^S \hat{c}_l^{\dagger S} \hat{d}_m^S \hat{d}_m^{\dagger S}) + (2 \cdot \hat{c}_m^S \hat{c}_m^{\dagger S} \hat{d}_l^S \hat{d}_l^{\dagger S}) + (1 \cdot \hat{c}_l^S \hat{c}_m^S \hat{d}_l^S \hat{d}_l^{\dagger S}) + (2 \cdot \hat{c}_l^S \hat{c}_m^{\dagger S} \hat{d}_m^S \hat{d}_m^{\dagger S})$$

$$\hat{V}_{\text{Part C}}(l, m) \stackrel{\text{MM}}{=} (-2 \cdot \hat{c}_l^S \hat{c}_m^{\dagger S}) + (-2 \cdot \hat{d}_l^S \hat{d}_m^{\dagger S}) + (-1 \cdot \hat{c}_l^S \hat{c}_m^S \hat{c}_l^{\dagger S} \hat{c}_m^{\dagger S} \hat{d}_l^S \hat{d}_m^{\dagger S}) + \\ + (2 \cdot \hat{c}_l^S \hat{c}_l^{\dagger S} \hat{d}_m^S \hat{d}_m^{\dagger S}) + (1 \cdot \hat{c}_m^S \hat{c}_m^{\dagger S} \hat{d}_l^S \hat{d}_l^{\dagger S}) + (2 \cdot \hat{c}_m^S \hat{c}_m^{\dagger S} \hat{d}_l^S \hat{d}_l^{\dagger S}) + (1 \cdot \hat{c}_l^S \hat{c}_m^{\dagger S} \hat{d}_m^S \hat{d}_m^{\dagger S})$$

# Usage for the Practical Training

## Math-Manipulator - Interactive Demonstration

$$\hat{V}^I(t) = \left\{ \hat{V}^S \right\}(t) \stackrel{2.11}{=} -J \cdot \sum_{[l,m]} \left\{ \left( \hat{c}_l^{\dagger S} \hat{c}_m^S + \hat{d}_l^{\dagger S} \hat{d}_m^S \right) \right\}(t)$$

$$= -J \cdot \sum_{[l,m]} \left( \hat{c}_l^{\dagger I}(t) \hat{c}_m^I(t) + \hat{d}_l^{\dagger I}(t) \hat{d}_m^I(t) \right)$$

$$\stackrel{\text{MM}}{=} -J \cdot \sum_{[l,m]} \left[ e^{i(\varepsilon_m - \varepsilon_l) \cdot t} \cdot \hat{V}_{\text{Part A}}(l, m) + e^{i(\varepsilon_m - \varepsilon_l + U) \cdot t} \cdot \hat{V}_{\text{Part B}}(l, m) + e^{i(\varepsilon_m - \varepsilon_l - U) \cdot t} \cdot \hat{V}_{\text{Part C}}(l, m) \right]$$

$$\hat{V}_{\text{Part A}}(l, m) \stackrel{\text{MM}}{=} (5 \cdot \hat{c}_l^S \hat{c}_m^{\dagger S}) + (5 \cdot \hat{d}_l^S \hat{d}_m^{\dagger S}) + (2 \cdot \hat{c}_l^S \hat{c}_m^S \hat{c}_l^{\dagger S} \hat{c}_m^{\dagger S} \hat{d}_l^S \hat{d}_m^{\dagger S}) + (2 \cdot \hat{c}_l^S \hat{c}_m^{\dagger S} \hat{d}_l^S \hat{d}_m^S \hat{d}_l^{\dagger S} \hat{d}_m^{\dagger S}) + \\ + (-3 \cdot \hat{c}_l^S \hat{c}_l^{\dagger S} \hat{d}_m^S \hat{d}_m^{\dagger S}) + (-3 \cdot \hat{c}_m^S \hat{c}_m^{\dagger S} \hat{d}_l^S \hat{d}_l^{\dagger S}) + (-3 \cdot \hat{c}_l^S \hat{c}_m^S \hat{d}_l^S \hat{d}_l^{\dagger S}) + (-3 \cdot \hat{c}_l^S \hat{c}_m^{\dagger S} \hat{d}_m^S \hat{d}_m^{\dagger S})$$

$$\hat{V}_{\text{Part B}}(l, m) \stackrel{\text{MM}}{=} (-2 \cdot \hat{c}_l^S \hat{c}_m^{\dagger S}) + (-2 \cdot \hat{d}_l^S \hat{d}_m^{\dagger S}) + (-1 \cdot \hat{c}_l^S \hat{c}_m^S \hat{c}_l^{\dagger S} \hat{c}_m^{\dagger S} \hat{d}_l^S \hat{d}_m^{\dagger S}) + (-1 \cdot \hat{c}_l^S \hat{c}_m^{\dagger S} \hat{d}_l^S \hat{d}_m^S \hat{d}_l^{\dagger S} \hat{d}_m^{\dagger S}) + \\ + (1 \cdot \hat{c}_l^S \hat{c}_l^{\dagger S} \hat{d}_m^S \hat{d}_m^{\dagger S}) + (2 \cdot \hat{c}_m^S \hat{c}_m^{\dagger S} \hat{d}_l^S \hat{d}_l^{\dagger S}) + (1 \cdot \hat{c}_l^S \hat{c}_m^{\dagger S} \hat{d}_l^S \hat{d}_l^{\dagger S}) + (2 \cdot \hat{c}_l^S \hat{c}_m^{\dagger S} \hat{d}_m^S \hat{d}_m^{\dagger S})$$

$$\hat{V}_{\text{Part C}}(l, m) \stackrel{\text{MM}}{=} (-2 \cdot \hat{c}_l^S \hat{c}_m^{\dagger S}) + (-2 \cdot \hat{d}_l^S \hat{d}_m^{\dagger S}) + (-1 \cdot \hat{c}_l^S \hat{c}_m^S \hat{c}_l^{\dagger S} \hat{c}_m^{\dagger S} \hat{d}_l^S \hat{d}_m^{\dagger S}) + (-1 \cdot \hat{c}_l^S \hat{c}_m^{\dagger S} \hat{d}_l^S \hat{d}_m^S \hat{d}_l^{\dagger S} \hat{d}_m^{\dagger S}) + \\ + (2 \cdot \hat{c}_l^S \hat{c}_l^{\dagger S} \hat{d}_m^S \hat{d}_m^{\dagger S}) + (1 \cdot \hat{c}_m^S \hat{c}_m^{\dagger S} \hat{d}_l^S \hat{d}_l^{\dagger S}) + (2 \cdot \hat{c}_m^S \hat{c}_m^{\dagger S} \hat{d}_l^S \hat{d}_l^{\dagger S}) + (1 \cdot \hat{c}_l^S \hat{c}_m^{\dagger S} \hat{d}_m^S \hat{d}_m^{\dagger S})$$

## Computer aided calculations

### Math-Manipulator

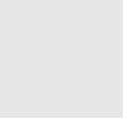
#### Interactive Demonstration

##### Usage for the Practical Training

2024-12-08



1. Go to Demo in VS-Code submodule mm-calculations.
2. Think of disabling spellchecker
3. Show process of how V is calculated
4. Story: at the end at first result was no longer symmetrical in up and down, but it should have been. Mistake was in the very first input. But because of the upfront configuration (and in my case programming) effort, re-calculating the endresult took 20min instead of one week



# Optimization: Difference of $V(t)$ for Hopping

## Custom Python Scripts (SymPy)

- Monte-Carlo sampling requires transition probabilities between "adjacent" states

$$\begin{aligned}\alpha &= \frac{P(\tilde{N}, t)}{P(N, t)} = \frac{f(\tilde{N}, t)}{f(N, t)} \stackrel{2.39}{=} \frac{\left| e^{\mathcal{H}_{\text{eff}}(\tilde{N}, t)} \right|^2 |\Psi_{\tilde{N}}|^2}{\left| e^{\mathcal{H}_{\text{eff}}(N, t)} \right|^2 |\Psi_N|^2} \\ &= \frac{|\Psi_{\tilde{N}}|^2 e^{\Re(\mathcal{H}_{\text{eff}}(\tilde{N}, t)) + i\Im(\mathcal{H}_{\text{eff}}(\tilde{N}, t)) + \Re(\mathcal{H}_{\text{eff}}(\tilde{N}, t)) - i\Im(\mathcal{H}_{\text{eff}}(\tilde{N}, t))}}{|\Psi_N|^2 e^{\Re(\mathcal{H}_{\text{eff}}(N, t)) + i\Im(\mathcal{H}_{\text{eff}}(N, t)) + \Re(\mathcal{H}_{\text{eff}}(N, t)) - i\Im(\mathcal{H}_{\text{eff}}(N, t))}} \\ &= \frac{|\Psi_{\tilde{N}}|^2}{|\Psi_N|^2} e^{2\Re(\mathcal{H}_{\text{eff}}(\tilde{N}, t)) - 2\Re(\mathcal{H}_{\text{eff}}(N, t))} \\ &= \frac{|\Psi_{\tilde{N}}|^2}{|\Psi_N|^2} e^{2\Re(\mathcal{H}_{\text{eff}}(\tilde{N}, t) - \mathcal{H}_{\text{eff}}(N, t))}\end{aligned}$$

Computer aided calculations

└ Custom Python Scripts (SymPy)

└ Optimization: Difference of  $V(t)$  for Hopping

2024-12-08

Monte-Carlo sampling requires transition probabilities between "adjacent" states

$$\begin{aligned}\alpha &= \frac{P(\tilde{N}, t)}{P(N, t)} = \frac{f(\tilde{N}, t)}{f(N, t)} \frac{\left| e^{\mathcal{H}_{\text{eff}}(\tilde{N}, t)} \right|^2 |\Psi_{\tilde{N}}|^2}{\left| e^{\mathcal{H}_{\text{eff}}(N, t)} \right|^2 |\Psi_N|^2} \\ &= \frac{|\Psi_{\tilde{N}}|^2 e^{\Re(\mathcal{H}_{\text{eff}}(\tilde{N}, t)) + i\Im(\mathcal{H}_{\text{eff}}(\tilde{N}, t)) + \Re(\mathcal{H}_{\text{eff}}(\tilde{N}, t)) - i\Im(\mathcal{H}_{\text{eff}}(\tilde{N}, t))}}{|\Psi_N|^2 e^{\Re(\mathcal{H}_{\text{eff}}(N, t)) + i\Im(\mathcal{H}_{\text{eff}}(N, t)) + \Re(\mathcal{H}_{\text{eff}}(N, t)) - i\Im(\mathcal{H}_{\text{eff}}(N, t))}} \\ &= \frac{|\Psi_{\tilde{N}}|^2}{|\Psi_N|^2} e^{2\Re(\mathcal{H}_{\text{eff}}(\tilde{N}, t)) - 2\Re(\mathcal{H}_{\text{eff}}(N, t))} \\ &= \frac{|\Psi_{\tilde{N}}|^2}{|\Psi_N|^2} e^{2\Re(\mathcal{H}_{\text{eff}}(\tilde{N}, t) - \mathcal{H}_{\text{eff}}(N, t))}\end{aligned}$$

- "Adjacent" states because from one Monte-Carlo step to the next we require a small change of one or two particle-sites. Could come from hopping or flipping.
- The  $E_0$  differences are quickly calculated by hand in the report



# Optimization: Difference of $V(t)$ for Hopping

## Custom Python Scripts (SymPy)

- Monte-Carlo sampling requires transition probabilities between "adjacent" states
- These require differences of  $\hat{V}^I(t)$  and  $E_0$

$$\begin{aligned}\alpha &= \frac{P(\tilde{N}, t)}{P(N, t)} = \frac{f(\tilde{N}, t)}{f(N, t)} \stackrel{2.39}{=} \frac{\left| e^{H_{\text{eff}}(\tilde{N}, t)} \right|^2 |\Psi_{\tilde{N}}|^2}{\left| e^{H_{\text{eff}}(N, t)} \right|^2 |\Psi_N|^2} \\ &= \frac{|\Psi_{\tilde{N}}|^2 e^{\Re(H_{\text{eff}}(\tilde{N}, t)) + i\Im(H_{\text{eff}}(\tilde{N}, t)) + \Re(H_{\text{eff}}(\tilde{N}, t)) - i\Im(H_{\text{eff}}(\tilde{N}, t))}}{|\Psi_N|^2 e^{\Re(H_{\text{eff}}(N, t)) + i\Im(H_{\text{eff}}(N, t)) + \Re(H_{\text{eff}}(N, t)) - i\Im(H_{\text{eff}}(N, t))}} \\ &= \frac{|\Psi_{\tilde{N}}|^2}{|\Psi_N|^2} e^{2\Re(H_{\text{eff}}(\tilde{N}, t)) - 2\Re(H_{\text{eff}}(N, t))} \\ &= \frac{|\Psi_{\tilde{N}}|^2}{|\Psi_N|^2} e^{2\Re(H_{\text{eff}}(\tilde{N}, t) - H_{\text{eff}}(N, t))}\end{aligned}$$

Computer aided calculations  
└ Custom Python Scripts (SymPy)

└ Optimization: Difference of  $V(t)$  for Hopping

2024-12-08

1. "Adjacent" states because from one Monte-Carlo step to the next we require a small change of one or two particle-sites. Could come from hopping or flipping.
2. The  $E_0$  differences are quickly calculated by hand in the report

$$\begin{aligned}\alpha &= \frac{P(\tilde{N}, t)}{P(N, t)} = \frac{f(\tilde{N}, t)}{f(N, t)} \stackrel{2.39}{=} \frac{\left| e^{H_{\text{eff}}(\tilde{N}, t)} \right|^2 |\Psi_{\tilde{N}}|^2}{\left| e^{H_{\text{eff}}(N, t)} \right|^2 |\Psi_N|^2} \\ &= \frac{|\Psi_{\tilde{N}}|^2 e^{\Re(H_{\text{eff}}(\tilde{N}, t)) + i\Im(H_{\text{eff}}(\tilde{N}, t)) + \Re(H_{\text{eff}}(\tilde{N}, t)) - i\Im(H_{\text{eff}}(\tilde{N}, t))}}{|\Psi_N|^2 e^{\Re(H_{\text{eff}}(N, t)) + i\Im(H_{\text{eff}}(N, t)) + \Re(H_{\text{eff}}(N, t)) - i\Im(H_{\text{eff}}(N, t))}} \\ &= \frac{|\Psi_{\tilde{N}}|^2}{|\Psi_N|^2} e^{2\Re(H_{\text{eff}}(\tilde{N}, t)) - 2\Re(H_{\text{eff}}(N, t))} \\ &= \frac{|\Psi_{\tilde{N}}|^2}{|\Psi_N|^2} e^{2\Re(H_{\text{eff}}(\tilde{N}, t) - H_{\text{eff}}(N, t))}\end{aligned}$$

# Optimization: Difference of $V(t)$ for Hopping

## Custom Python Scripts (SymPy)

- Monte-Carlo sampling requires transition probabilities between "adjacent" states
- These require differences of  $\hat{V}^I(t)$  and  $E_0$
- Too many terms for simple evaluation by hand
- Intelligent pre-computation required to speed up processing

$$\begin{aligned}\alpha &= \frac{P(\tilde{N}, t)}{P(N, t)} = \frac{f(\tilde{N}, t)}{f(N, t)} \stackrel{2.39}{=} \frac{\left| e^{H_{\text{eff}}(\tilde{N}, t)} \right|^2 |\Psi_{\tilde{N}}|^2}{\left| e^{H_{\text{eff}}(N, t)} \right|^2 |\Psi_N|^2} \\ &= \frac{|\Psi_{\tilde{N}}|^2 e^{\Re(H_{\text{eff}}(\tilde{N}, t)) + i\Im(H_{\text{eff}}(\tilde{N}, t)) + \Re(H_{\text{eff}}(\tilde{N}, t)) - i\Im(H_{\text{eff}}(\tilde{N}, t))}}{|\Psi_N|^2 e^{\Re(H_{\text{eff}}(N, t)) + i\Im(H_{\text{eff}}(N, t)) + \Re(H_{\text{eff}}(N, t)) - i\Im(H_{\text{eff}}(N, t))}} \\ &= \frac{|\Psi_{\tilde{N}}|^2}{|\Psi_N|^2} e^{2\Re(H_{\text{eff}}(\tilde{N}, t)) - 2\Re(H_{\text{eff}}(N, t))} \\ &= \frac{|\Psi_{\tilde{N}}|^2}{|\Psi_N|^2} e^{2\Re(H_{\text{eff}}(\tilde{N}, t) - H_{\text{eff}}(N, t))}\end{aligned}$$

## Computer aided calculations

### Custom Python Scripts (SymPy)

#### Optimization: Difference of $V(t)$ for Hopping

2024-12-08

Monte-Carlo sampling requires transition probabilities between "adjacent" states

These require differences of  $\hat{V}^I(t)$  and  $E_0$

Too many terms for simple evaluation by hand

Intelligent pre-computation required to speed up processing

$$\begin{aligned}\alpha &= \frac{P(\tilde{N}, t)}{P(N, t)} = \frac{f(\tilde{N}, t)}{f(N, t)} \stackrel{2.39}{=} \frac{\left| e^{H_{\text{eff}}(\tilde{N}, t)} \right|^2 |\Psi_{\tilde{N}}|^2}{\left| e^{H_{\text{eff}}(N, t)} \right|^2 |\Psi_N|^2} \\ &= \frac{|\Psi_{\tilde{N}}|^2 e^{\Re(H_{\text{eff}}(\tilde{N}, t)) + i\Im(H_{\text{eff}}(\tilde{N}, t)) + \Re(H_{\text{eff}}(\tilde{N}, t)) - i\Im(H_{\text{eff}}(\tilde{N}, t))}}{|\Psi_N|^2 e^{\Re(H_{\text{eff}}(N, t)) + i\Im(H_{\text{eff}}(N, t)) + \Re(H_{\text{eff}}(N, t)) - i\Im(H_{\text{eff}}(N, t))}} \\ &= \frac{|\Psi_{\tilde{N}}|^2}{|\Psi_N|^2} e^{2\Re(H_{\text{eff}}(\tilde{N}, t)) - 2\Re(H_{\text{eff}}(N, t))} \\ &= \frac{|\Psi_{\tilde{N}}|^2}{|\Psi_N|^2} e^{2\Re(H_{\text{eff}}(\tilde{N}, t) - H_{\text{eff}}(N, t))}\end{aligned}$$

1. "Adjacent" states because from one Monte-Carlo step to the next we require a small change of one or two particle-sites. Could come from hopping or flipping.
2. The  $E_0$  differences are quickly calculated by hand in the report



$$\begin{aligned} E_0(N) - E_0(\tilde{N}) &= U \sum_l n_{l,\downarrow} n_{l,\uparrow} - U \sum_l \tilde{n}_{l,\downarrow} \tilde{n}_{l,\uparrow} + \sum_{l,\sigma} \varepsilon_l n_{l,\sigma} - \sum_{l,\sigma} \varepsilon_l \tilde{n}_{l,\sigma} \\ &\stackrel{2.45}{=} (\varepsilon_i - \varepsilon_j) (n_{i,\sigma_i} - n_{j,\sigma_j}) + U \cdot \begin{cases} (n_{i,\uparrow} - n_{j,\uparrow}) (n_{i,\downarrow} - n_{j,\downarrow}) & : \sigma_i = \sigma_j \\ (n_{i,\uparrow} - n_{j,\downarrow}) (n_{i,\downarrow} - n_{j,\uparrow}) & : \sigma_i \neq \sigma_j \end{cases} \\ &= (\varepsilon_i - \varepsilon_j) (n_{i,\sigma_i} - n_{j,\sigma_j}) + U (n_{i,\sigma_i} - n_{j,\sigma_j}) (n_{i,\bar{\sigma}_i} - n_{j,\bar{\sigma}_j}) \end{aligned}$$

Setup difference

# The process: Difference of V(t) for Hopping

## Custom Python Scripts (SymPy)

$$\begin{aligned} E_0(N) - E_0(\tilde{N}) &= U \sum_l n_{l,\downarrow} n_{l,\uparrow} - U \sum_l \tilde{n}_{l,\downarrow} \tilde{n}_{l,\uparrow} + \sum_{l,\sigma} \varepsilon_l n_{l,\sigma} - \sum_{l,\sigma} \varepsilon_l \tilde{n}_{l,\sigma} \\ &\stackrel{2.45}{=} (\varepsilon_i - \varepsilon_j) (n_{i,\sigma_i} - n_{j,\sigma_j}) + U \cdot \begin{cases} (n_{i,\uparrow} - n_{j,\uparrow}) (n_{i,\downarrow} - n_{j,\downarrow}) & : \sigma_i = \sigma_j \\ (n_{i,\uparrow} - n_{j,\downarrow}) (n_{i,\downarrow} - n_{j,\uparrow}) & : \sigma_i \neq \sigma_j \end{cases} \\ &= (\varepsilon_i - \varepsilon_j) (n_{i,\sigma_i} - n_{j,\sigma_j}) + U (n_{i,\sigma_i} - n_{j,\sigma_j}) (n_{i,\bar{\sigma}_i} - n_{j,\bar{\sigma}_j}) \end{aligned}$$

### ■ Setup difference

## Computer aided calculations

### └ Custom Python Scripts (SymPy)

#### └ The process: Difference of V(t) for Hopping

1. The  $V$  differences are  $3 \cdot 8 = 32 \cdot 4 = 96 \cdot 2 = 192$  terms
2. The process is relatively straight forward:
  - write down the terms
  - replace
  - simplify difference
3. Still quite complicated and a lot of work, but:
  - way faster to write because non-repetitive logic
  - verifiably correct calculations
  - multiple times re-done on changes to underlying code, each time only took seconds
  - sometimes brute-force NEEDS to work, before optimization can be found



$$\begin{aligned} E(N) - E(\tilde{N}) &= U \sum_l n_{l,\downarrow} n_{l,\uparrow} - U \sum_l \tilde{n}_{l,\downarrow} \tilde{n}_{l,\uparrow} + \sum_{l,\sigma} \varepsilon_l n_{l,\sigma} - \sum_{l,\sigma} \varepsilon_l \tilde{n}_{l,\sigma} \\ &\stackrel{2.45}{=} (\varepsilon_i - \varepsilon_j) (n_{i,\sigma_i} - n_{j,\sigma_j}) + U \cdot \begin{cases} (n_{i,\uparrow} - n_{j,\uparrow}) (n_{i,\downarrow} - n_{j,\downarrow}) & : \sigma_i = \sigma_j \\ (n_{i,\uparrow} - n_{j,\downarrow}) (n_{i,\downarrow} - n_{j,\uparrow}) & : \sigma_i \neq \sigma_j \end{cases} \quad \tilde{n}_{l,\sigma} = \begin{cases} n_{i,\sigma_i} & : l = j \wedge \sigma = \sigma_j \\ n_{j,\sigma_j} & : l = i \wedge \sigma = \sigma_i \\ n_{l,\sigma} & : \text{else} \end{cases} \\ &= (\varepsilon_i - \varepsilon_j) (n_{i,\sigma_i} - n_{j,\sigma_j}) + U (n_{i,\sigma_i} - n_{j,\sigma_j}) (n_{i,\bar{\sigma}_i} - n_{j,\bar{\sigma}_j}) \end{aligned}$$

Setup difference  
Replace one by appropriate new occupation

# The process: Difference of V(t) for Hopping

## Custom Python Scripts (SymPy)

$$E_0(N) - E_0(\tilde{N}) = U \sum_l n_{l,\downarrow} n_{l,\uparrow} - U \sum_l \tilde{n}_{l,\downarrow} \tilde{n}_{l,\uparrow} + \sum_{l,\sigma} \varepsilon_l n_{l,\sigma} - \sum_{l,\sigma} \varepsilon_l \tilde{n}_{l,\sigma}$$

$$\stackrel{2.45}{=} (\varepsilon_i - \varepsilon_j) (n_{i,\sigma_i} - n_{j,\sigma_j}) + U \cdot \begin{cases} (n_{i,\uparrow} - n_{j,\uparrow}) (n_{i,\downarrow} - n_{j,\downarrow}) & : \sigma_i = \sigma_j \\ (n_{i,\uparrow} - n_{j,\downarrow}) (n_{i,\downarrow} - n_{j,\uparrow}) & : \sigma_i \neq \sigma_j \end{cases} \quad \tilde{n}_{l,\sigma} = \begin{cases} n_{i,\sigma_i} & : l = j \wedge \sigma = \sigma_j \\ n_{j,\sigma_j} & : l = i \wedge \sigma = \sigma_i \\ n_{l,\sigma} & : \text{else} \end{cases}$$

- Setup difference
- Replace one by appropriate new occupation

## Computer aided calculations

### Custom Python Scripts (SymPy)

#### The process: Difference of V(t) for Hopping

2024-12-08

# The process: Difference of $V(t)$ for Hopping

## Custom Python Scripts (SymPy)

$$\begin{aligned} E_0(N) - E_0(\tilde{N}) &= U \sum_l n_{l,\downarrow} n_{l,\uparrow} - U \sum_l \tilde{n}_{l,\downarrow} \tilde{n}_{l,\uparrow} + \sum_{l,\sigma} \varepsilon_l n_{l,\sigma} - \sum_{l,\sigma} \varepsilon_l \tilde{n}_{l,\sigma} \\ &\stackrel{2.45}{=} (\varepsilon_i - \varepsilon_j) (n_{i,\sigma_i} - n_{j,\sigma_j}) + U \cdot \begin{cases} (n_{i,\uparrow} - n_{j,\uparrow})(n_{i,\downarrow} - n_{j,\downarrow}) & : \sigma_i = \sigma_j \\ (n_{i,\uparrow} - n_{j,\downarrow})(n_{i,\downarrow} - n_{j,\uparrow}) & : \sigma_i \neq \sigma_j \end{cases} \quad \tilde{n}_{l,\sigma} = \begin{cases} n_{i,\sigma_i} & : l = j \wedge \sigma = \sigma_j \\ n_{j,\sigma_j} & : l = i \wedge \sigma = \sigma_i \\ n_{l,\sigma} & : \text{else} \end{cases} \\ &= (\varepsilon_i - \varepsilon_j) (n_{i,\sigma_i} - n_{j,\sigma_j}) + U (n_{i,\sigma_i} - n_{j,\sigma_j}) (n_{i,\bar{\sigma}_i} - n_{j,\bar{\sigma}_j}) \end{aligned}$$

- Setup difference
- Replace one by appropriate new occupation
- Simplify as much as possible
- Sum over all possible neighbor combinations

2024-12-08

## Computer aided calculations

- └ Custom Python Scripts (SymPy)

- └ The process: Difference of  $V(t)$  for Hopping

1. The  $V$  differences are  $3 \cdot 8 = 32 \cdot 4 = 96 \cdot 2 = 192$  terms
2. The process is relatively straight forward:
  - write down the terms
  - replace
  - simplify difference
3. Still quite complicated and a lot of work, but:
  - way faster to write because non-repetitive logic
  - verifiably correct calculations
  - multiple times re-done on changes to underlying code, each time only took seconds
  - sometimes brute-force NEEDS to work, before optimization can be found

The process: Difference of  $V(t)$  for Hopping  
Custom Python Scripts (SymPy)

$$\begin{aligned} E(N) - E(\tilde{N}) &= U \sum_l n_{l,\sigma} \cdot \tilde{n}_{l,\sigma} + \sum_l \tilde{n}_{l,\sigma} \cdot n_{l,\sigma} \\ &\stackrel{!}{=} (\varepsilon_i - \varepsilon_j) (n_{i,\sigma_i} - n_{j,\sigma_j}) + U \cdot \begin{cases} (n_{i,\uparrow} - n_{j,\uparrow})(n_{i,\downarrow} - n_{j,\downarrow}) & : \sigma_i = \sigma_j \\ (n_{i,\uparrow} - n_{j,\downarrow})(n_{i,\downarrow} - n_{j,\uparrow}) & : \sigma_i \neq \sigma_j \end{cases} \quad \tilde{n}_{l,\sigma} = \begin{cases} n_{i,\sigma_i} & : l = j \wedge \sigma = \sigma_j \\ n_{j,\sigma_j} & : l = i \wedge \sigma = \sigma_i \\ n_{l,\sigma} & : \text{else} \end{cases} \\ &= (\varepsilon_i - \varepsilon_j) (n_{i,\sigma_i} - n_{j,\sigma_j}) + U (n_{i,\sigma_i} - n_{j,\sigma_j}) (n_{i,\bar{\sigma}_i} - n_{j,\bar{\sigma}_j}) \end{aligned}$$

- Setup difference
- Replace one by appropriate new occupation
- Simplify as much as possible
- Sum over all possible neighbor combinations



# The process: Difference of $V(t)$ for Hopping

## Custom Python Scripts (SymPy)

$$\begin{aligned} E_0(N) - E_0(\tilde{N}) &= U \sum_l n_{l,\downarrow} n_{l,\uparrow} - U \sum_l \tilde{n}_{l,\downarrow} \tilde{n}_{l,\uparrow} + \sum_{l,\sigma} \varepsilon_l n_{l,\sigma} - \sum_{l,\sigma} \varepsilon_l \tilde{n}_{l,\sigma} \\ &\stackrel{2.45}{=} (\varepsilon_i - \varepsilon_j) (n_{i,\sigma_i} - n_{j,\sigma_j}) + U \cdot \begin{cases} (n_{i,\uparrow} - n_{j,\uparrow})(n_{i,\downarrow} - n_{j,\downarrow}) & : \sigma_i = \sigma_j \\ (n_{i,\uparrow} - n_{j,\downarrow})(n_{i,\downarrow} - n_{j,\uparrow}) & : \sigma_i \neq \sigma_j \end{cases} \quad \tilde{n}_{l,\sigma} = \begin{cases} n_{i,\sigma_i} & : l = j \wedge \sigma = \sigma_j \\ n_{j,\sigma_j} & : l = i \wedge \sigma = \sigma_i \\ n_{l,\sigma} & : \text{else} \end{cases} \\ &= (\varepsilon_i - \varepsilon_j) (n_{i,\sigma_i} - n_{j,\sigma_j}) + U (n_{i,\sigma_i} - n_{j,\sigma_j}) (n_{i,\bar{\sigma}_i} - n_{j,\bar{\sigma}_j}) \end{aligned}$$

- Setup difference
- Replace one by appropriate new occupation
- Simplify as much as possible
- Sum over all possible neighbor combinations

Problem: number of terms/combinations:  $3 \cdot 8 \cdot 4 \cdot 2 = 192$



## Computer aided calculations

### Custom Python Scripts (SymPy)

#### The process: Difference of $V(t)$ for Hopping

2024-12-08

The process: Difference of  $V(t)$  for Hopping  
Custom Python Scripts (SymPy)

$$E(N) - E(\tilde{N}) = U \sum_l n_{l,\sigma} \cdot \tilde{n}_{l,\sigma} + \sum_l \tilde{n}_{l,\sigma} \cdot n_{l,\sigma}$$
$$\stackrel{!}{=} (\varepsilon_i - \varepsilon_j) (n_{i,\sigma_i} - n_{j,\sigma_j}) + U \cdot \begin{cases} (n_{i,\uparrow} - n_{j,\uparrow})(n_{i,\downarrow} - n_{j,\downarrow}) & : \sigma_i = \sigma_j \\ (n_{i,\uparrow} - n_{j,\downarrow})(n_{i,\downarrow} - n_{j,\uparrow}) & : \sigma_i \neq \sigma_j \end{cases}$$
$$\tilde{n}_{l,\sigma} = \begin{cases} n_{i,\sigma_i} & : l = j \wedge \sigma = \sigma_j \\ n_{j,\sigma_j} & : l = i \wedge \sigma = \sigma_i \\ n_{l,\sigma} & : \text{else} \end{cases}$$

Setup difference  
Replace one by appropriate new occupation  
Simplify as much as possible  
Sum over all possible neighbor combinations

Problem: number of terms/combinations:  $3 \cdot 8 \cdot 4 \cdot 2 = 192$

# Script-Generation: Difference of V(t) for Hopping

## Custom Python Scripts (SymPy)

### Generating Script

simplificationtermhelper.py

### Generated Script

analyticalcalcfuctions.py

Computer aided calculations

└ Custom Python Scripts (SymPy)

└ Script-Generation: Difference of V(t) for Hopping

2024-12-08

1. simplificationtermhelper.py script to generate the script
2. analyticalcalcfuctions.py script is generated and used in the code
3. Generated file: 1556 lines, script that generates: 534 lines
4. Show the script that generates/is generated → simplification with Sympy
5. Parallel to computer-science practice "Vendored" code



# Beamer: LaTeX way of writing Presentations

## Presentations? - Custom Beamer Template

- Write presentations like your papers/thesis in  $\text{\LaTeX}$

2024-12-08

Computer aided calculations

└ Presentations? - Custom Beamer Template

    └ Beamer: LaTeX way of writing Presentations

▪ Write presentations like your papers/thesis in  $\text{\LaTeX}$

Beamer [2] Template [3]



# Beamer: LaTeX way of writing Presentations

## Presentations? - Custom Beamer Template

- Write presentations like your papers/thesis in  $\text{\LaTeX}$
- Reuse formulas/images/code/sources



2024-12-08

Computer aided calculations

└ Presentations? - Custom Beamer Template

└ Beamer: LaTeX way of writing Presentations

1. Re-use tooling, editor, setup (even this on overleaf alone much better than shared powerpoint presentation)
2. This presentation, report and thesis all live in one git-repository and share resources
3. Everything I ever work on is hosted in Git.
  - Without Version Control it is no longer possible for me to work effectively
  - Every workflow is massively ensured by it
  - Also for single-person work, but built in collaboration, synchronization and backup

# Beamer: LaTeX way of writing Presentations

## Presentations? - Custom Beamer Template

- Write presentations like your papers/thesis in  $\text{\LaTeX}$
- Reuse formulas/images/code/sources
- Consistent style & references



1. Re-use tooling, editor, setup (even this on overleaf alone much better than shared powerpoint presentation)
2. This presentation, report and thesis all live in one git-repository and share resources
3. Everything I ever work on is hosted in Git.
  - Without Version Control it is no longer possible for me to work effectively
  - Every workflow is massively ensured by it
  - Also for single-person work, but built in collaboration, synchronization and backup

# Beamer: LaTeX way of writing Presentations

## Presentations? - Custom Beamer Template

- Write presentations like your papers/thesis in  $\text{\LaTeX}$
- Reuse formulas/images/code/sources
- Consistent style & references
- Version control



# Beamer: LaTeX way of writing Presentations

## Presentations? - Custom Beamer Template

- Write presentations like your papers/thesis in  $\text{\LaTeX}$
- Reuse formulas/images/code/sources
- Consistent style & references
- Version control
- Easier collaboration



2024-12-08  
Computer aided calculations  
└ Presentations? - Custom Beamer Template  
    └ Beamer: LaTeX way of writing Presentations

- Write presentations like your papers/thesis in  $\text{\LaTeX}$
- Reuse formulas/images/code/sources
- Consistent style & references
- Version control
- Easier collaboration

# Minimal example for beamer presentation

## Presentations? - Custom Beamer Template

```
1 \documentclass[aspectratio=169]{beamer}
2
3 \usetheme[neutralbackground]{uniamntf}
4
5 \title[]{Title}
6 \subtitle{subtitle}
7 \author{Jonas Kell}
8 \institute[TP III]{Chair for theoretical Physics III}
9 \date[01.05.2024]{\$1^{\text{st}}\$ of Mai 2024}
10
11 \acknowledgement{
12   Jonas Kell\\ Augsburg University\\
13   jonas.kell@student.uni-augsburg.de\\ www.uni-augsburg.de
14 }
15
16 \begin{document}
17
18 \begin{frame}[t,plain]
19   \maketitle
20 \end{frame}
21
22 \section*{Summary \& Conclusion}
23 {
24   \setbeamertemplate{frametitle}{uniamntfack}
25   \begin{frame}[plain]{Acknowledgment}
26     Thank you for your kind attention
27   \end{frame}
28 }
29 \end{document}
```



Computer aided calculations

└ Presentations? - Custom Beamer Template

└ Minimal example for beamer presentation

2024-12-08

1. 29 lines gets you a basic presentation
2. First time very much slower as putting together by hand in PowerPoint
3. If you want to do crazy stuff, possible but really hard
4. Actually quite a timesaver when you have an example/template to work of and do not require crazy levels of features
5. When it works, perfectly portable (just a pdf), high performant and versatile (output rendering of presentation, notes, animations from one source)
6. If proper presentation tools, has all the bells and whistles (presentation timer, pointer, multi-view presentation and more)

Minimal example for beamer presentation  
Presentations? - Custom Beamer Template



Thank you for your kind attention



Jonas Kell  
University of Augsburg  
[jonas.kell@student.uni-augsburg.de](mailto:jonas.kell@student.uni-augsburg.de)  
[www.uni-augsburg.de](http://www.uni-augsburg.de)



Universität Augsburg  
Mathematisch-Naturwissenschaftlich-  
Technische Fakultät



# References I

## Summary & Conclusion

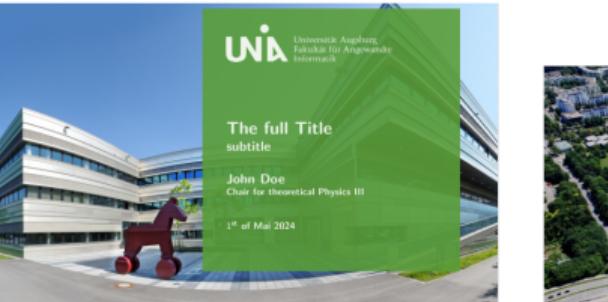
- [1] J. Kell, *Math-manipulator tool - info and repository*, (2024) <https://github.com/jonas-kell/math-manipulator>.
- [2] CTAN: package beamer, <https://ctan.org/pkg/beamer> (visited on 04/11/2024).
- [3] J. Kell, [Https://github.com/jonas-kell/beamer-theme-unia-mntf](https://github.com/jonas-kell/beamer-theme-unia-mntf), (2024) <https://github.com/jonas-kell/math-manipulator>.
- [4] J. Kell, *Latex sources for this presentation, the thesis and report*, (2024) <https://github.com/jonas-kell/master-thesis-documents>.
- [5] J. Kell, *Thesis code implementation and reference*, (2024) <https://github.com/jonas-kell/master-thesis-code>.
- [6] J. Kell, *Supplementary calculations perfomed with math-manipulator*, (2024) <https://github.com/jonas-kell/master-thesis-mm-calculations>.





# Theme alternative: Faculty of App. Computer Science

## Extra slides



**Random Text**  
Testsection

because of the [t] option, text starts at the top. Default is centered.

Hello

This is Text with a delayed show animation

**Title-Here**  
Testsection Hello

This is a text in first column.  
 $E = mc^2$

This text will be in the second column and on a second thoughts, this is a nice looking layout in some cases.

First item  
Second item  
and  
and  
and  
and  
and

**Outline**

- 1 Testsection
- 2 Testsection Hello
- 3 Testsection Test
  - sub-1
  - sub-2
- 4 Summary

John Doe (TP III) small size 15.05.2024 A / 10

**Outline**

- 1 Testsection
- 2 Testsection Hello
- 3 Testsection Test
  - sub-1
  - sub-2
- 4 Summary

Thank you for your kind attention

John Doe (TP III)

John Doe  
Augsburg University  
john.doe@uni-augsburg.de  
www.uni-augsburg.de

John Doe (TP III) small size 15.05.2024 A / 10

# **Backup-Solution Problem 1**

xtra slides

aided calculations  
slides

Backup-Solution Problem 1





```


$$\hat{V}_{\text{part A}}(l, m) \stackrel{\text{MM}}{=} \sum_{\sigma \in \{\uparrow, \downarrow\}} \hat{c}_{l, \sigma}^S \hat{c}_{m, \sigma}^{\dagger S} (1 + 2 \cdot \hat{n}_{l, \bar{\sigma}}^S \hat{n}_{m, \bar{\sigma}}^S - \hat{n}_{l, \bar{\sigma}}^S - \hat{n}_{m, \bar{\sigma}}^S)$$


$$= (-2 \cdot c_l c_m^{\dagger}) + (-1 \cdot c_l c_m^{\dagger} d_l d_m d_l^{\dagger} d_m^{\dagger}) + (2 \cdot c_l c_m^{\dagger} d_l d_l^{\dagger}) + (1 \cdot c_l c_m^{\dagger} d_m d_m^{\dagger})$$


$$+ (-2 \cdot c_l c_m^{\dagger}) + (-1 \cdot c_l c_m^{\dagger} d_l d_m d_l^{\dagger} d_m^{\dagger}) + (2 \cdot c_l c_m^{\dagger} \cdot (1 + d_l^{\dagger} d_l)) + (1 \cdot c_l c_m^{\dagger} d_m d_m^{\dagger})$$


$$+ (-2 \cdot c_l c_m^{\dagger}) + (-1 \cdot c_l c_m^{\dagger} d_l d_m d_l^{\dagger} d_m^{\dagger}) + (2 \cdot c_l c_m^{\dagger} \cdot (1 + d_m^{\dagger} d_m)) + (c_l c_m^{\dagger} \cdot (1 + d_m^{\dagger} d_m))$$


$$+ (-2 \cdot c_l c_m^{\dagger}) - (1 \cdot c_l c_m^{\dagger} d_l d_m d_m^{\dagger}) + (2 \cdot c_l c_m^{\dagger} \cdot (1 + d_l^{\dagger} d_l)) + (c_l c_m^{\dagger} \cdot (1 + d_m^{\dagger} d_m))$$


$$+ (-2 \cdot c_l c_m^{\dagger}) - (c_l c_m^{\dagger} \cdot (1 + d_l^{\dagger} d_l) \cdot (1 + d_m^{\dagger} d_m)) + (2 \cdot c_l c_m^{\dagger} \cdot (1 + d_l^{\dagger} d_l)) + (c_l c_m^{\dagger} \cdot (1 + d_m^{\dagger} d_m))$$


$$+ (-2 \cdot c_l c_m^{\dagger}) - (c_l c_m^{\dagger} \cdot 1 \cdot 1) - (c_l c_m^{\dagger} \cdot 1 \cdot d_m^{\dagger} d_m) - (c_l c_m^{\dagger} d_l^{\dagger} d_l \cdot 1) - c_l c_m^{\dagger} d_l^{\dagger} d_l d_m^{\dagger} d_m + (2 \cdot c_l c_m^{\dagger} \cdot (1 + d_l^{\dagger} d_l)) + (c_l c_m^{\dagger} \cdot (1 + d_m^{\dagger} d_m))$$


$$+ (-2 \cdot c_l c_m^{\dagger}) - (c_l c_m^{\dagger} \cdot 1 \cdot 1) - (c_l c_m^{\dagger} \cdot 1 \cdot d_m^{\dagger} d_m) - (c_l c_m^{\dagger} d_l^{\dagger} d_l \cdot 1) - c_l c_m^{\dagger} d_l^{\dagger} d_l d_m^{\dagger} d_m + (2 \cdot c_l c_m^{\dagger} \cdot 1) + (2 \cdot c_l c_m^{\dagger} d_l^{\dagger} d_l) + (c_l c_m^{\dagger} \cdot 1) + (c_l c_m^{\dagger} d_m^{\dagger} d_m)$$


$$+ (-2 \cdot c_l c_m^{\dagger}) - (c_l c_m^{\dagger} \cdot 1 \cdot 1) - (c_l c_m^{\dagger} \cdot 1 \cdot d_m^{\dagger} d_m) - (c_l c_m^{\dagger} d_l^{\dagger} d_l \cdot 1) - c_l c_m^{\dagger} d_l^{\dagger} d_l d_m^{\dagger} d_m + (2 \cdot c_l c_m^{\dagger} \cdot 1) + (2 \cdot c_l c_m^{\dagger} d_l^{\dagger} d_l) + (c_l c_m^{\dagger} \cdot 1) + (c_l c_m^{\dagger} d_m^{\dagger} d_m)$$


$$= (-c_l c_m^{\dagger} d_l^{\dagger} d_l - c_l c_m^{\dagger} d_l^{\dagger} d_l d_m^{\dagger} d_m + (2 \cdot c_l c_m^{\dagger} d_l^{\dagger} d_l))$$


```

Variables:

<b>Macros:</b>	
c <sup>+</sup>	<code>b0["c", #0]</code>
c	<code>ba["c", #0]</code>
d <sup>+</sup>	<code>b0["d", #0]</code>
d	<code>ba["d", #0]</code>
displayl	<code>\mathit{displayl}</code>
displaym	<code>\mathit{displaym}</code>
l	<code>dist(displayl)</code>
m	<code>dist(displaym)</code>

# Backup: Simplification V-Parts

## Extra slides

## Computer aided calculations

- Extra slides

- Backup: Simplification V-Parts

2024-12-08



1. Better optimization was found, rewriting in number operators DOES save on terms (however only for B and C, A still same number of terms and computational requirement, even if it looks nicer)
2. Doesn't help at all with computing more efficiently, so python script still exceptionally useful
3. Wouldn't have come so far, if simple "brute-force" solution wasn't done in the first place