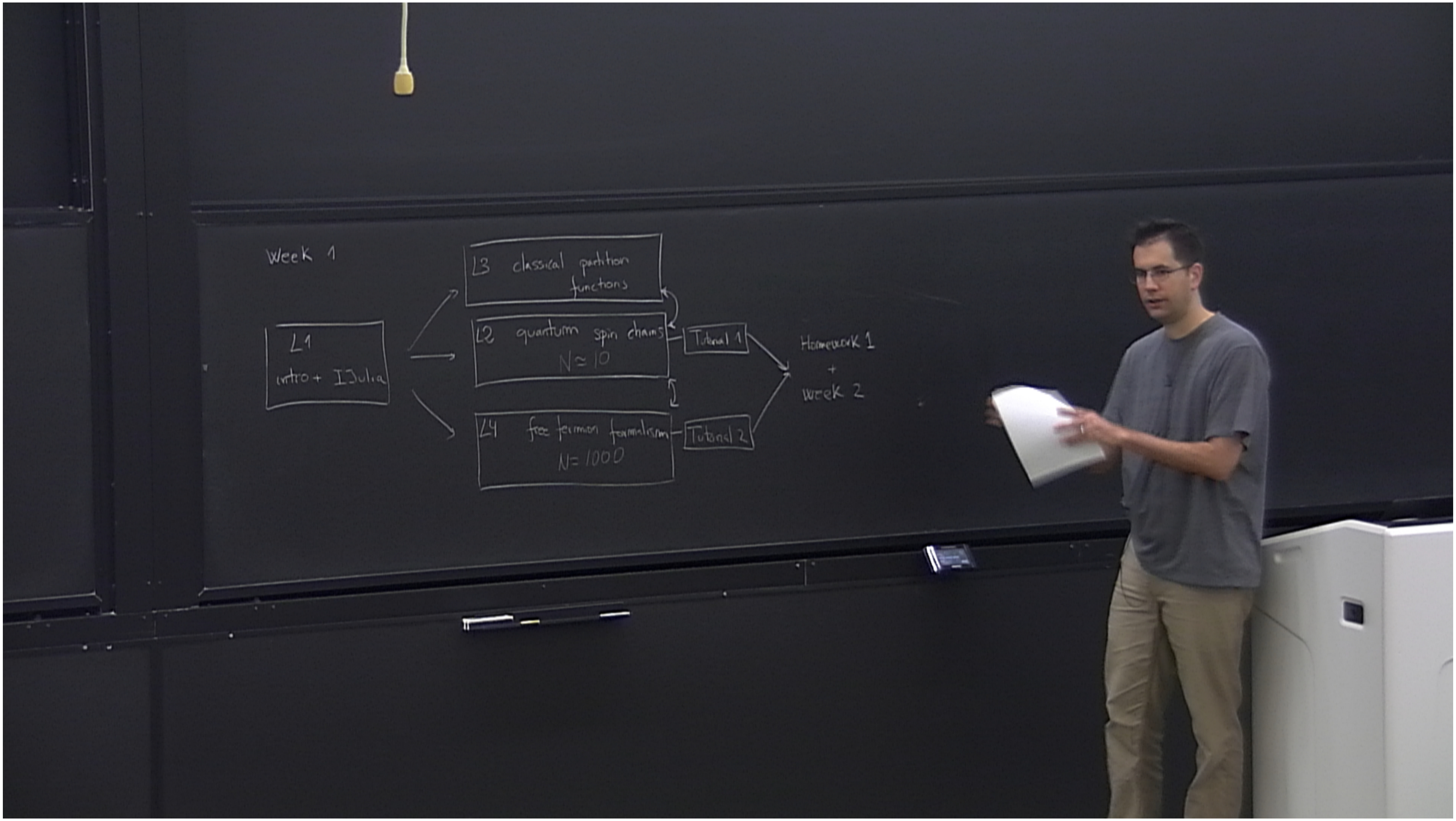
Title: PSI 2015/2016 Explorations in Condensed Matter - Guifre Vidal - 4

Date: Mar 24, 2016 10:15 AM

URL: http://pirsa.org/16030133

Abstract:


Week 1
L1
intro + IJulia
L3 classical partition functions
L2 quantum spin chains
N≃10
L4 free fermion formalism
N=1000
Tutorial 1
Tutorial 2
Homework 1
+
week 2

$\psi_1 \quad \psi_2$

$\psi_i^\dagger = \psi_i$ Hermitian

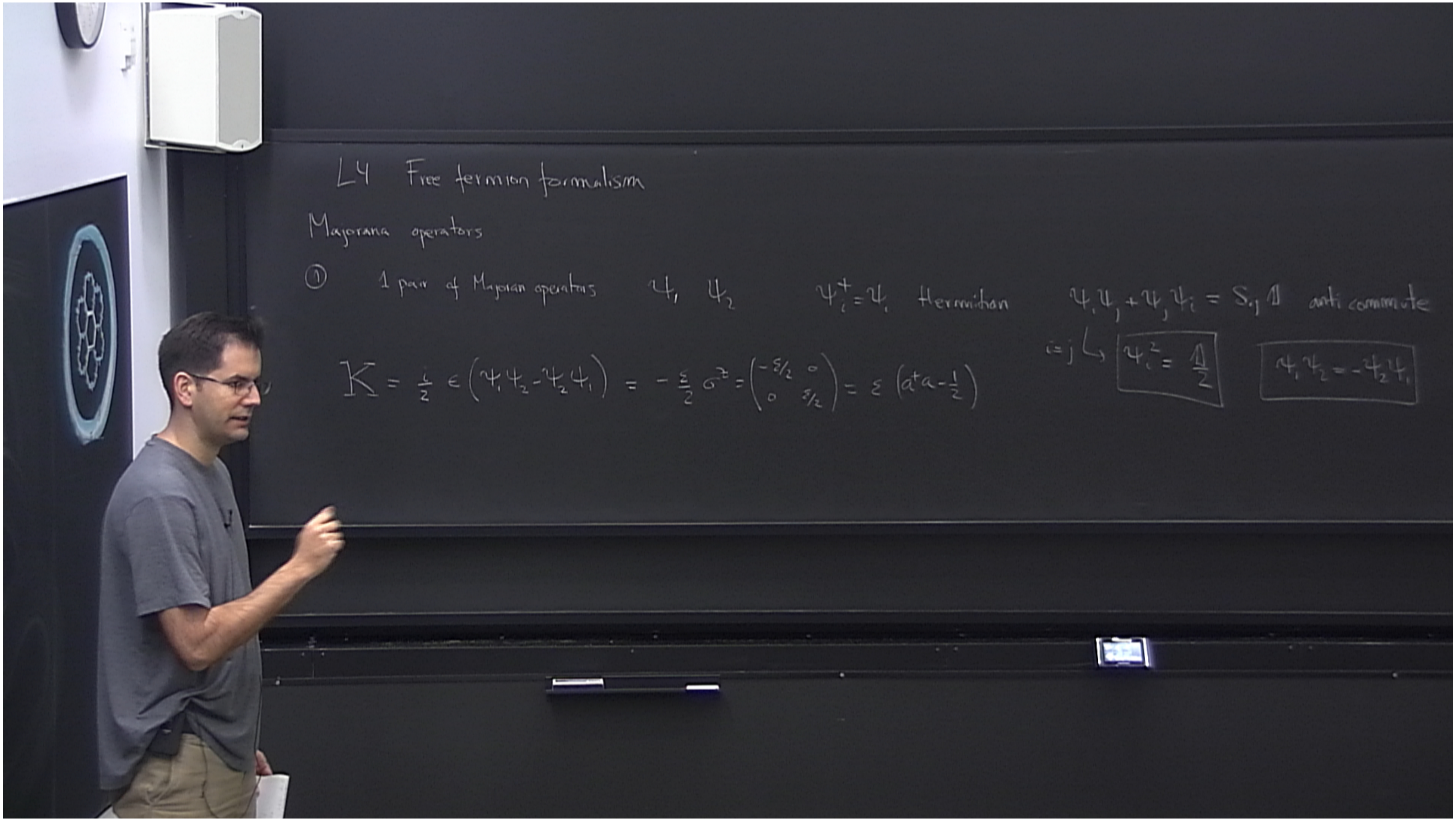
$\psi_i\psi_j + \psi_j\psi_i = \delta_{ij}\mathbb{1}$ anti commute

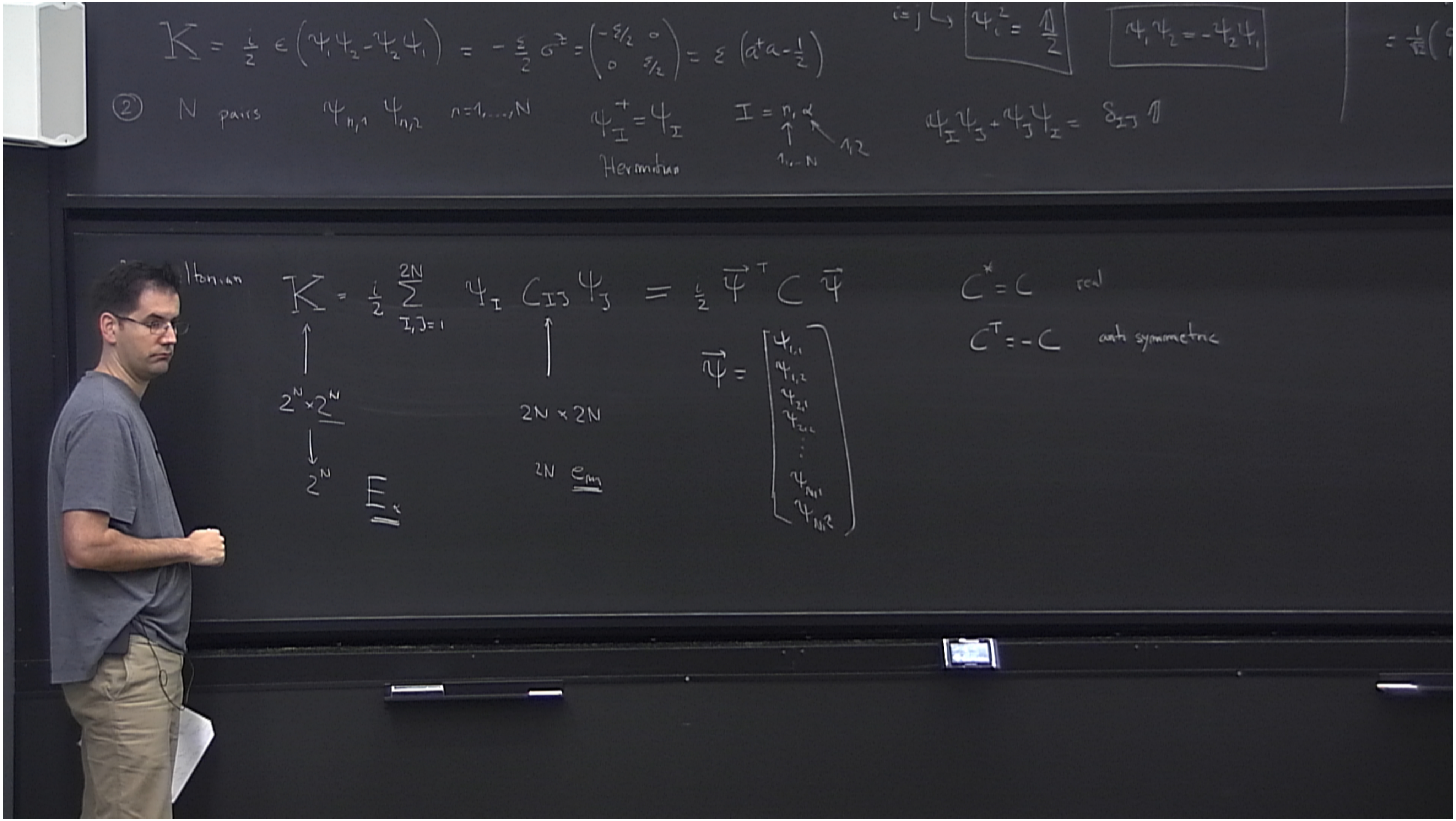
$i=j \to \boxed{\psi_i^2 = \frac{\mathbb{1}}{2}}$ $\quad \boxed{\psi_1\psi_2 = -\psi_2\psi_1}$

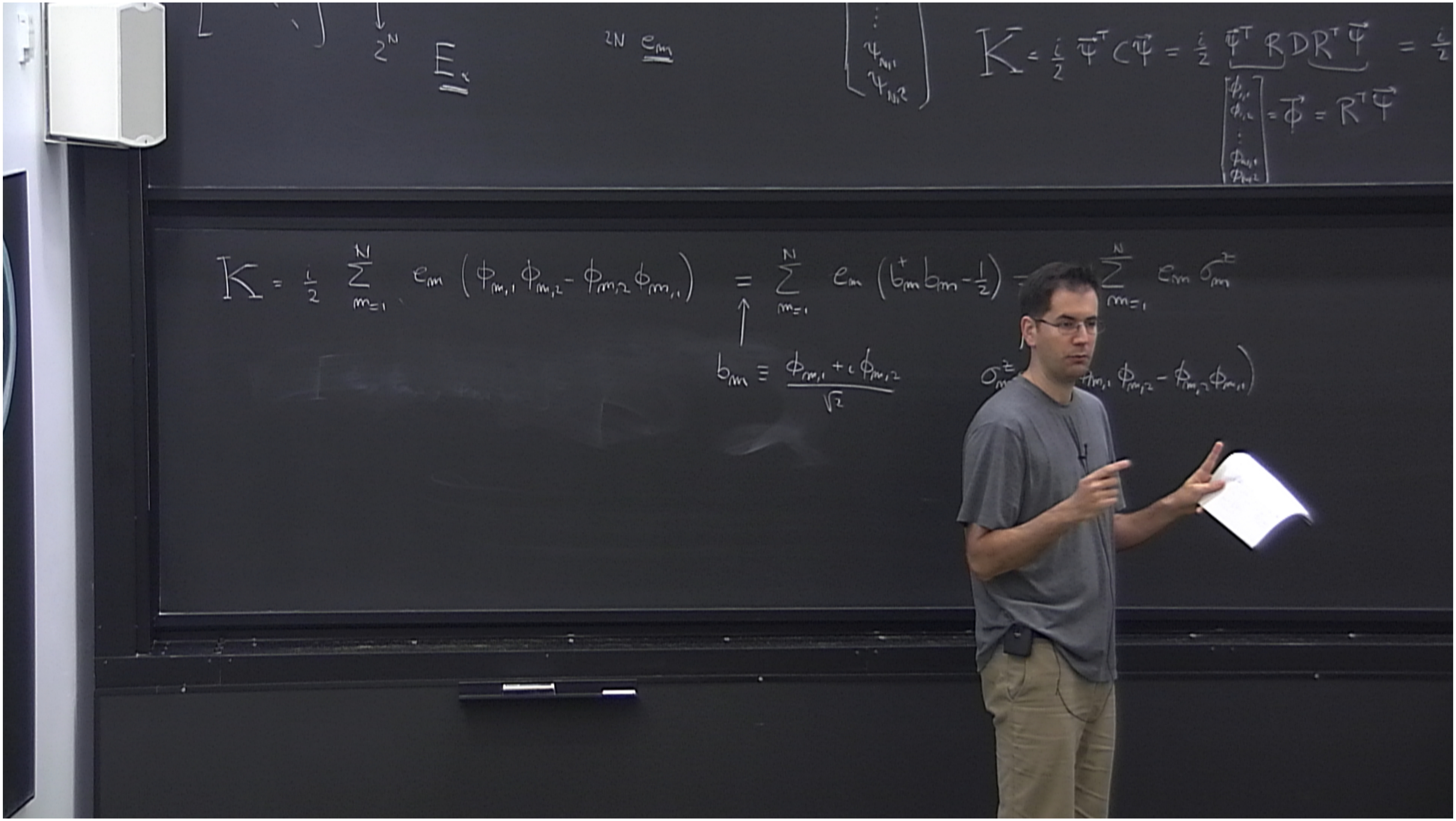
$a = \frac{\psi_1 + i\psi_2}{\sqrt{2}}$ fermionic Annihilation operator

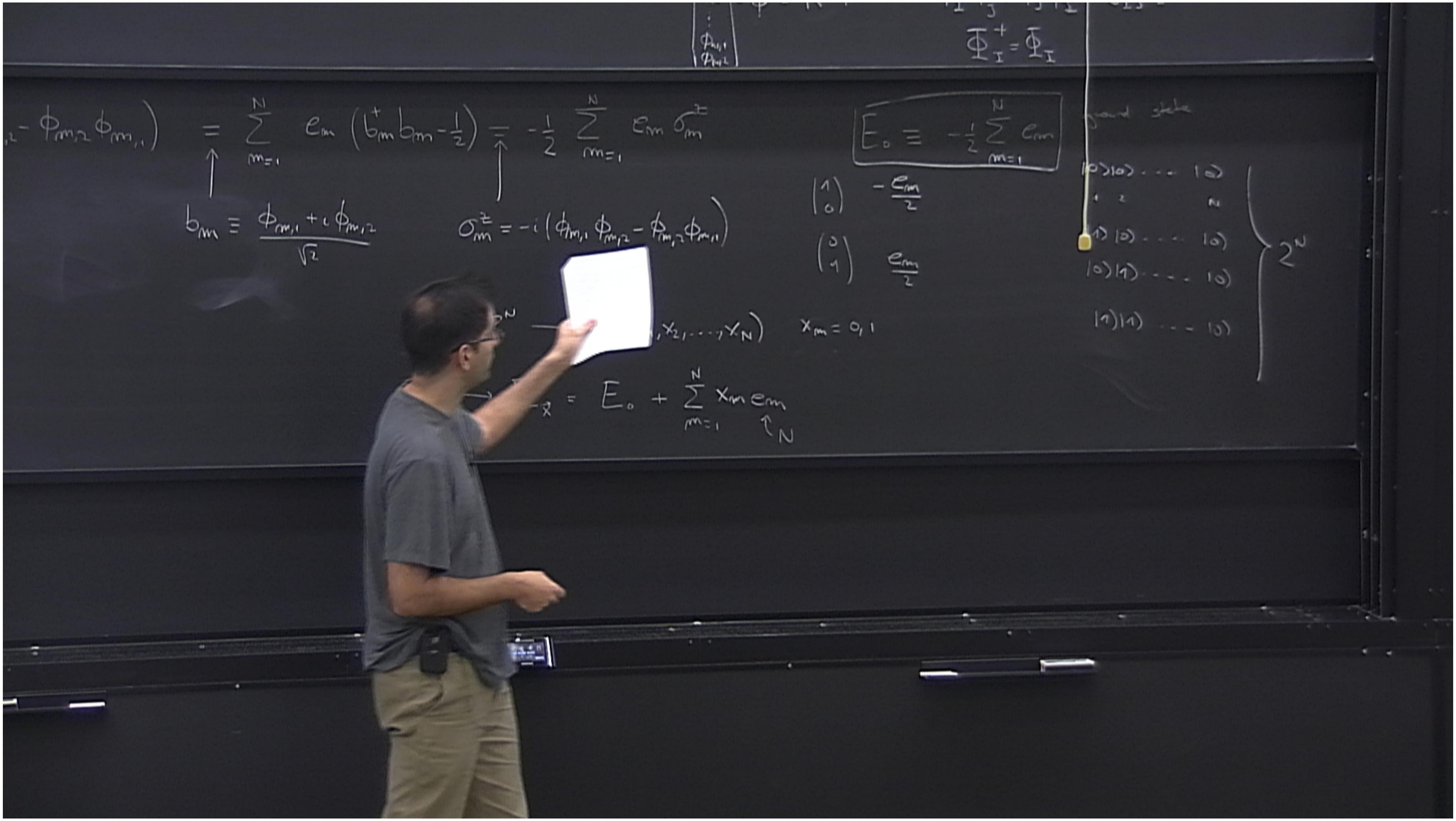
$a^\dagger = \frac{\psi_1 - i\psi_2}{\sqrt{2}}$

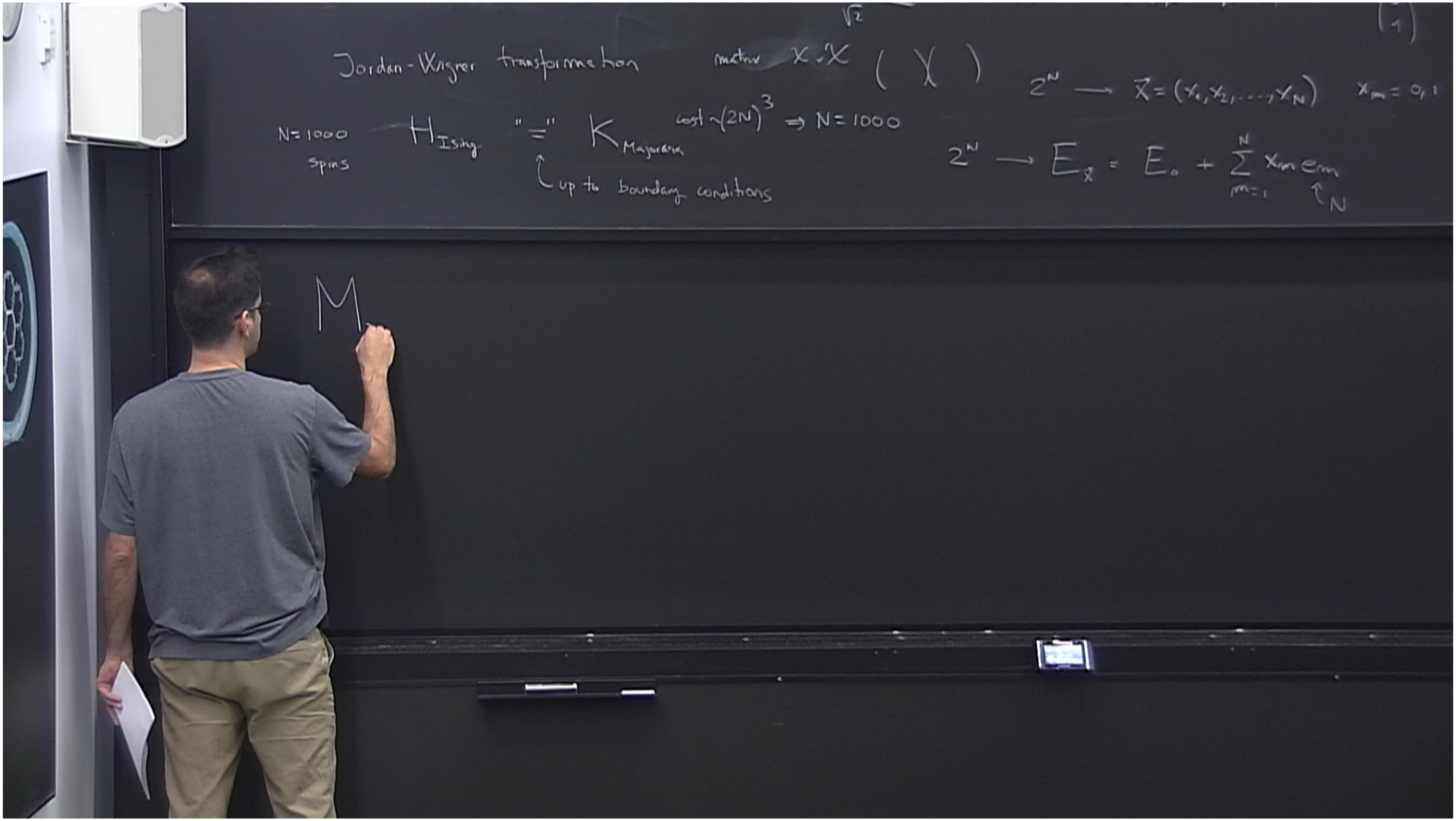
$\psi_1 = \frac{\sigma^x}{\sqrt{2}} = \frac{1}{\sqrt{2}}\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ $\quad \psi_2 = \frac{\sigma^y}{\sqrt{2}} = \frac{1}{\sqrt{2}}\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$ Pauli matrices

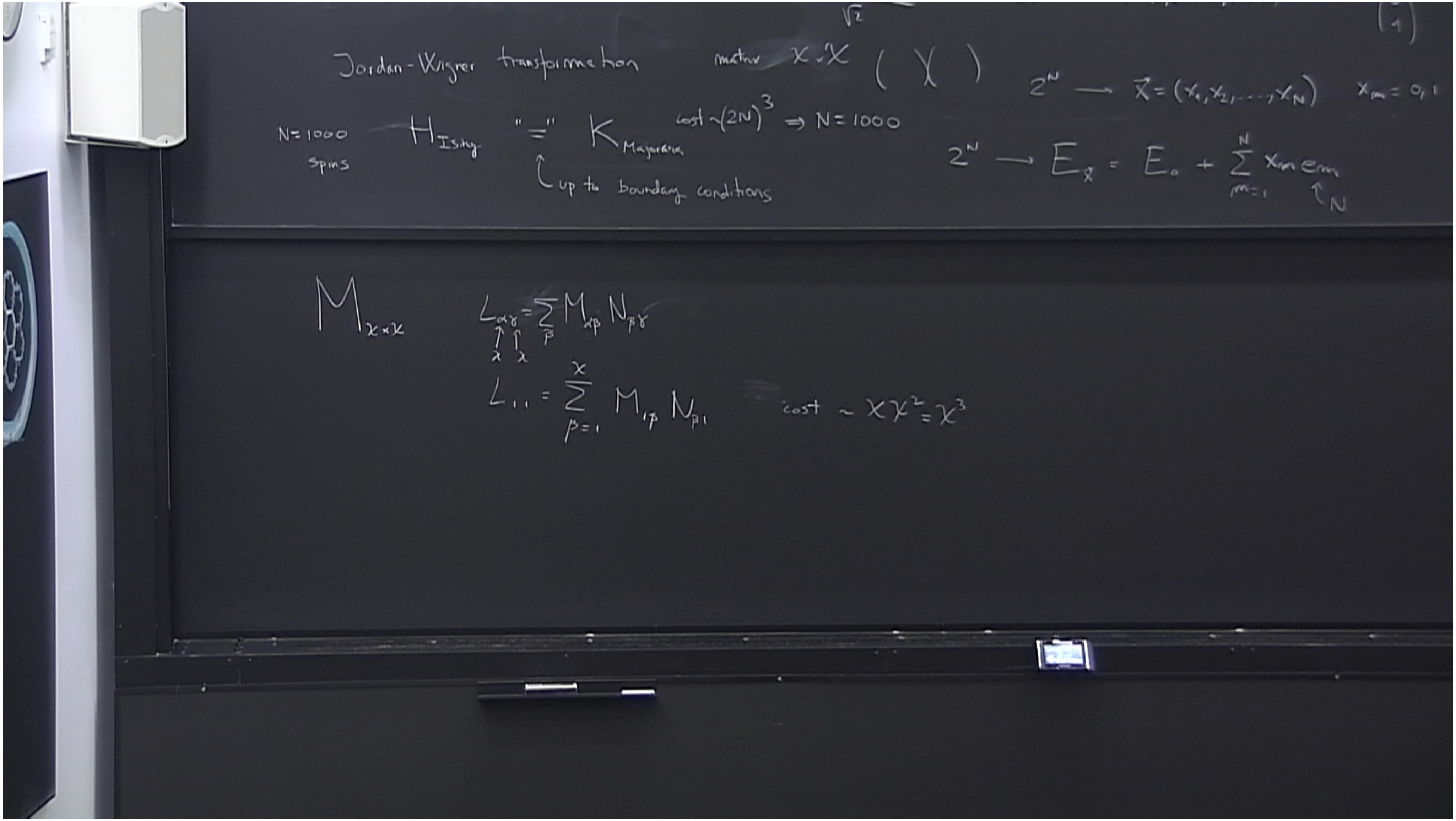

L4 Free fermion formalism
Majorana operators
① 1 pair of Majoran operators $\psi_1$ $\psi_2$
$\psi_i^\dagger = \psi_i$ Hermitian
$\psi_i\psi_j + \psi_j\psi_i = \delta_{ij} \mathbb{1}$ anti commute
$i=j$ ↳ $\boxed{\psi_i^2 = \frac{\mathbb{1}}{2}}$
$\boxed{\psi_1\psi_2 = -\psi_2\psi_1}$
$K = \frac{i}{2}\epsilon(\psi_1\psi_2 - \psi_2\psi_1) = -\frac{\varepsilon}{2}\sigma^z = \begin{pmatrix} -\varepsilon/2 & 0 \\ 0 & \varepsilon/2 \end{pmatrix} = \varepsilon\left(a^\dagger a - \frac{1}{2}\right)$

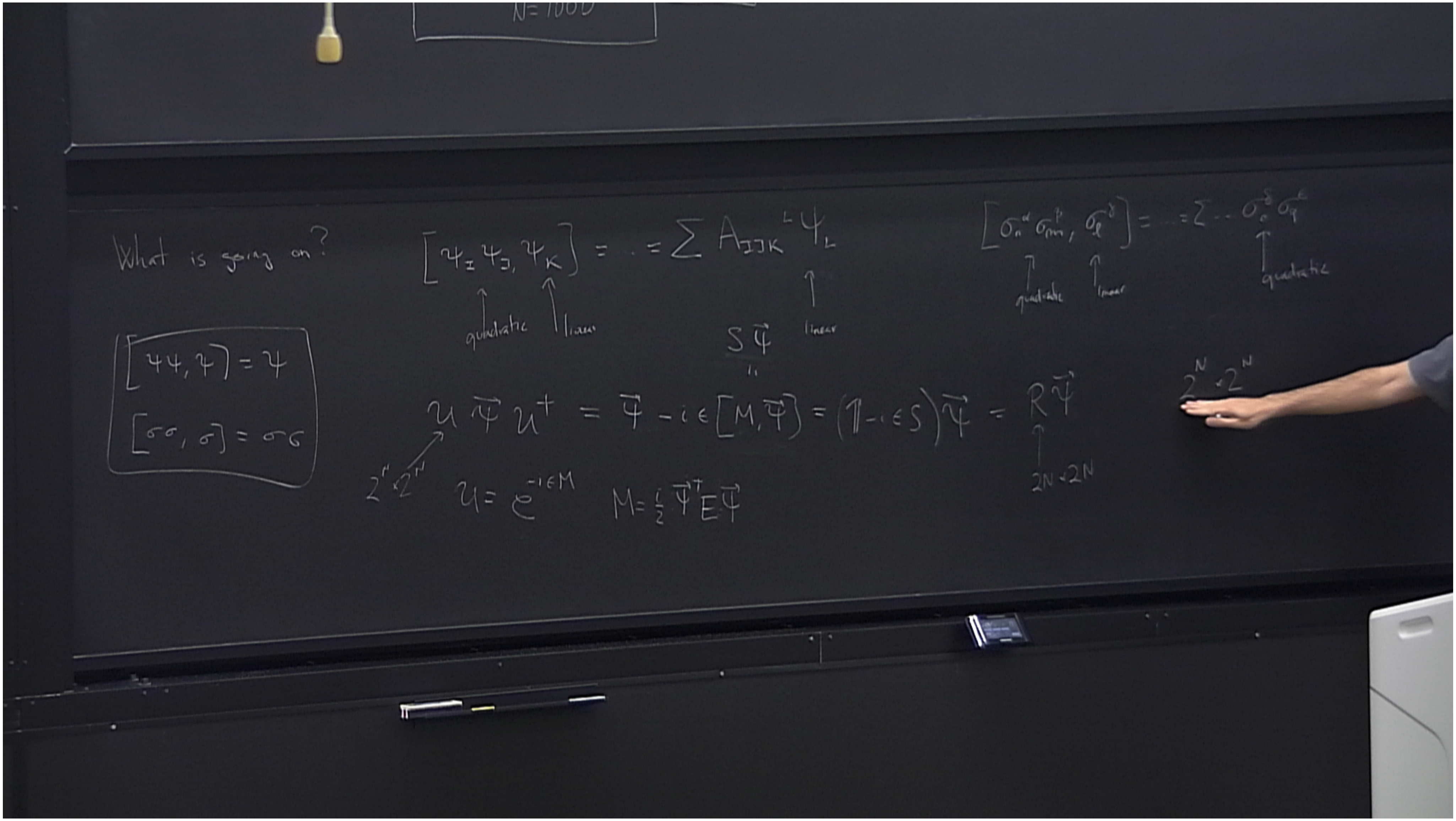

$K = \frac{i}{2}\epsilon(\psi_1\psi_2 - \psi_2\psi_1) = -\frac{\varepsilon}{2}\sigma^z = \begin{pmatrix} -\varepsilon/2 & 0 \\ 0 & \varepsilon/2 \end{pmatrix} = \varepsilon(a^+a - \frac{1}{2})$
$i=j$ $\boxed{\psi_i^2 = \frac{1}{2}}$
$\boxed{\psi_1\psi_2 = -\psi_2\psi_1}$
② N pairs $\psi_{n,1}$ $\psi_{n,2}$ $n=1,\dots,N$
$\psi_I^+ = \psi_I$
Hermitian
$I = n, \alpha$
$1,\dots,N$
$1,2$
$\psi_I\psi_J + \psi_J\psi_I = \delta_{IJ}\mathbb{1}$
$K = \frac{i}{2}\sum_{I,J=1}^{2N} \psi_I C_{IJ} \psi_J = \frac{i}{2}\vec{\psi}^T C \vec{\psi}$
$2^N \times 2^N$
$2^N$
$E_\alpha$
$2N \times 2N$
$2N$ $e_m$
$\vec{\psi} = \begin{bmatrix} \psi_{1,1} \\ \psi_{1,2} \\ \psi_{2,1} \\ \psi_{2,2} \\ \vdots \\ \psi_{N,1} \\ \psi_{N,2} \end{bmatrix}$
$C^* = C$ real
$C^T = -C$ anti symmetric

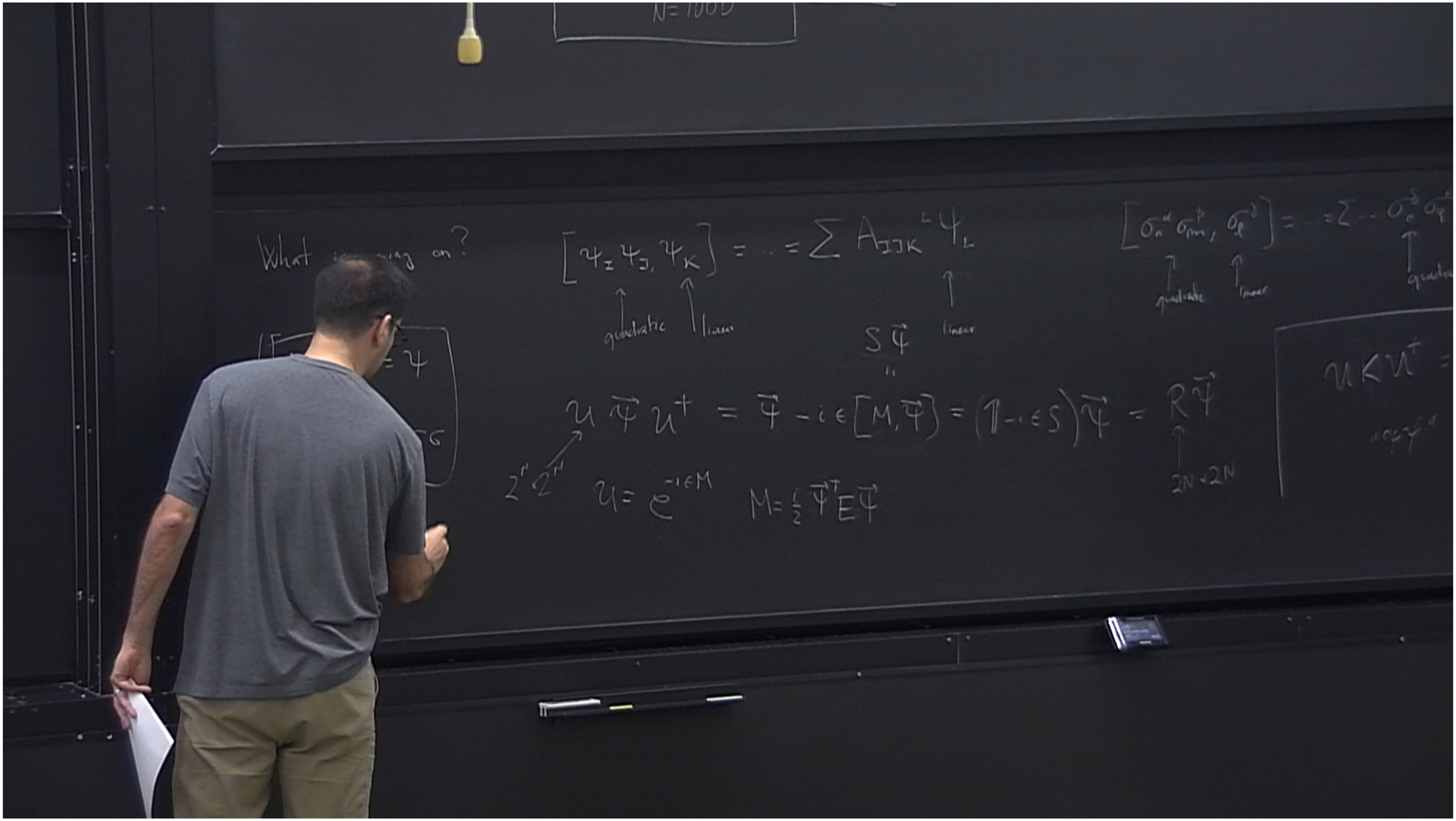

K = i/2 Σ_{m=1}^{N} e_m (φ_{m,1} φ_{m,2} − φ_{m,2} φ_{m,1}) = Σ_{m=1}^{N} e_m (b_m^† b_m − 1/2)
b_m ≡ (φ_{m,1} + i φ_{m,2}) / √2
K = i/2 Ψ^T C Ψ = i/2 Ψ^T R D R^T Ψ
Φ = R^T Ψ

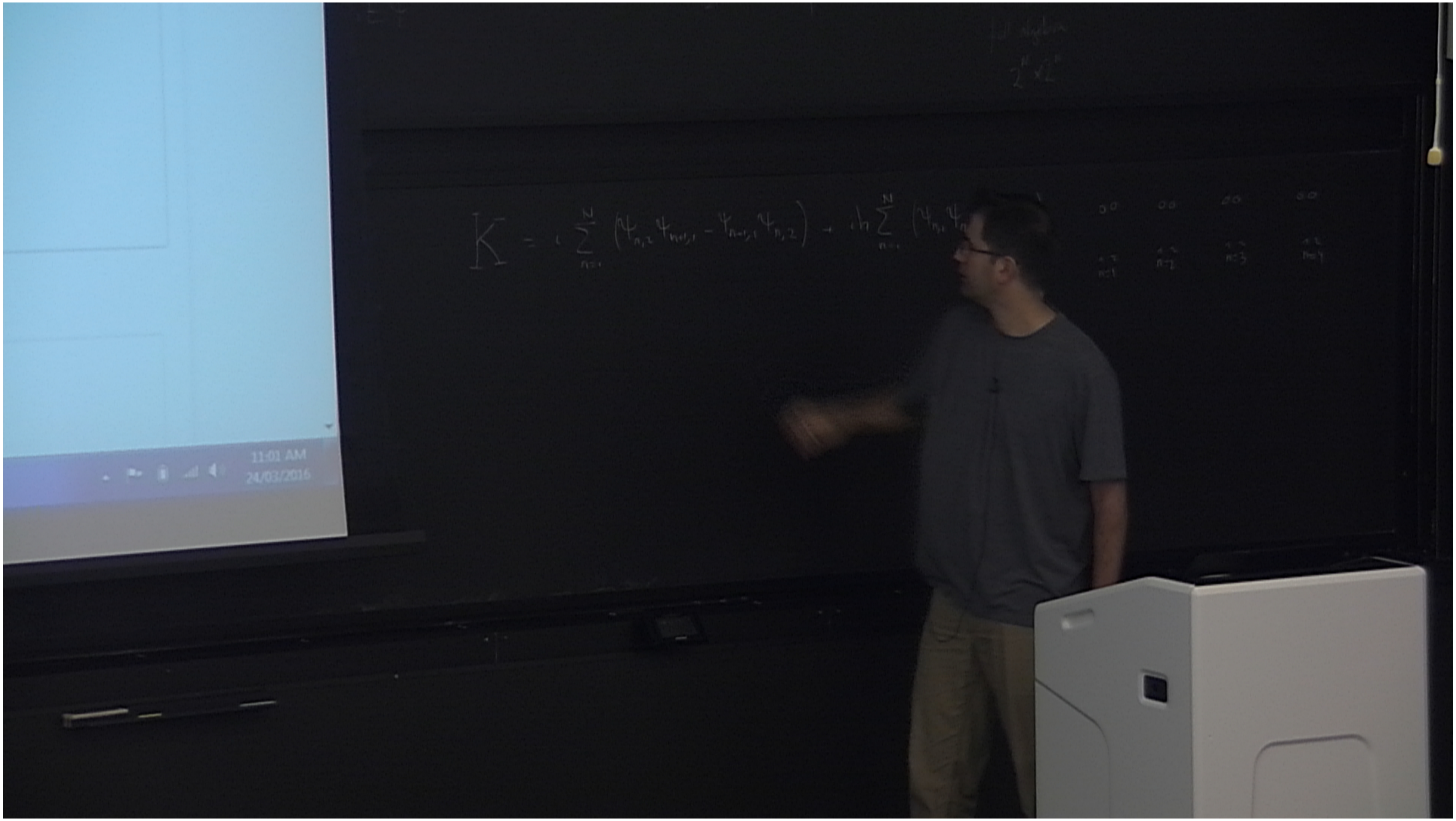

$= \sum_{m=1}^{N} e_m (b_m^\dagger b_m - \frac{1}{2}) = -\frac{1}{2} \sum_{m=1}^{N} e_m \sigma_m^z$
$E_0 \equiv -\frac{1}{2} \sum_{m=1}^{N} e_m$
$b_m \equiv \frac{\phi_{m,1} + i\phi_{m,2}}{\sqrt{2}}$
$\sigma_m^z = -i(\phi_{m,1}\phi_{m,2} - \phi_{m,2}\phi_{m,1})$
ground state
$x_m = 0, 1$
$= E_0 + \sum_{m=1}^{N} x_m e_m$
$2^N$

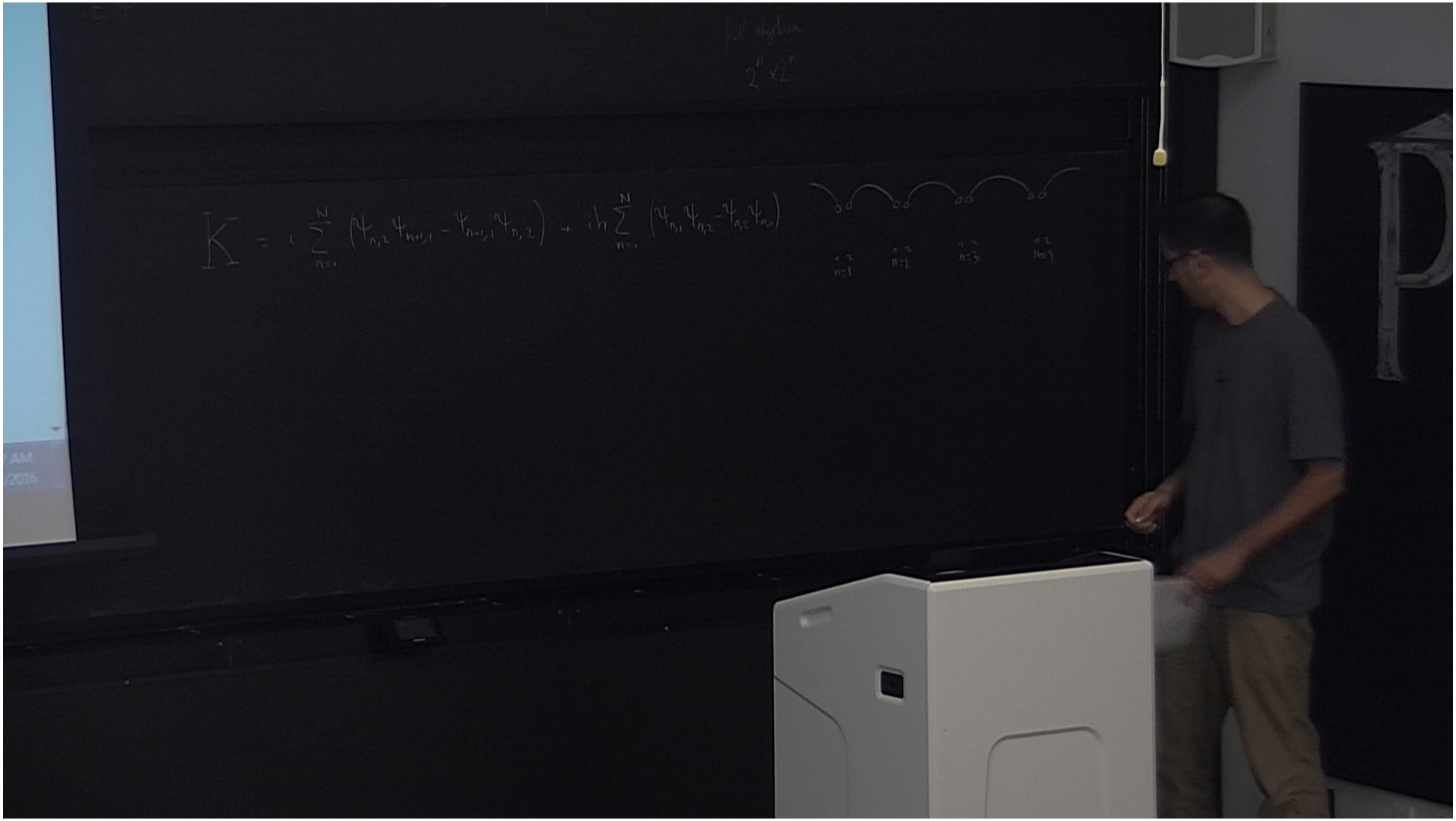

Jordan-Wigner transformation
matrix
N=1000
spins
H_Ising "=" K_Majorana
cost ~(2N)^3 ⇒ N=1000
up to boundary conditions
2^N → X = (x_1, x_2, ..., x_N)
x_m = 0, 1
2^N → E_X = E_0 + Σ_{m=1}^{N} x_m e_m
N
M


Jordan-Wigner transformation
matrix $\chi \times \chi$
$2^N \longrightarrow \vec{x} = (x_1, x_2, \ldots, x_N)$ $x_m = 0, 1$
N = 1000 spins
$H_{Ising}$ "=" $K_{Majorana}$
cost $\sim (2N)^3 \Rightarrow N = 1000$
up to boundary conditions
$2^N \longrightarrow E_{\vec{x}} = E_0 + \sum_{m=1}^{N} x_m e_m$
N
$M_{\chi \times \chi}$
$L_{\alpha\gamma} = \sum_\beta M_{\alpha\beta} N_{\beta\gamma}$
$\chi$ $\chi$
$L_{11} = \sum_{\beta=1}^{\chi} M_{1\beta} N_{\beta 1}$
cost $\sim \chi \chi^2 = \chi^3$

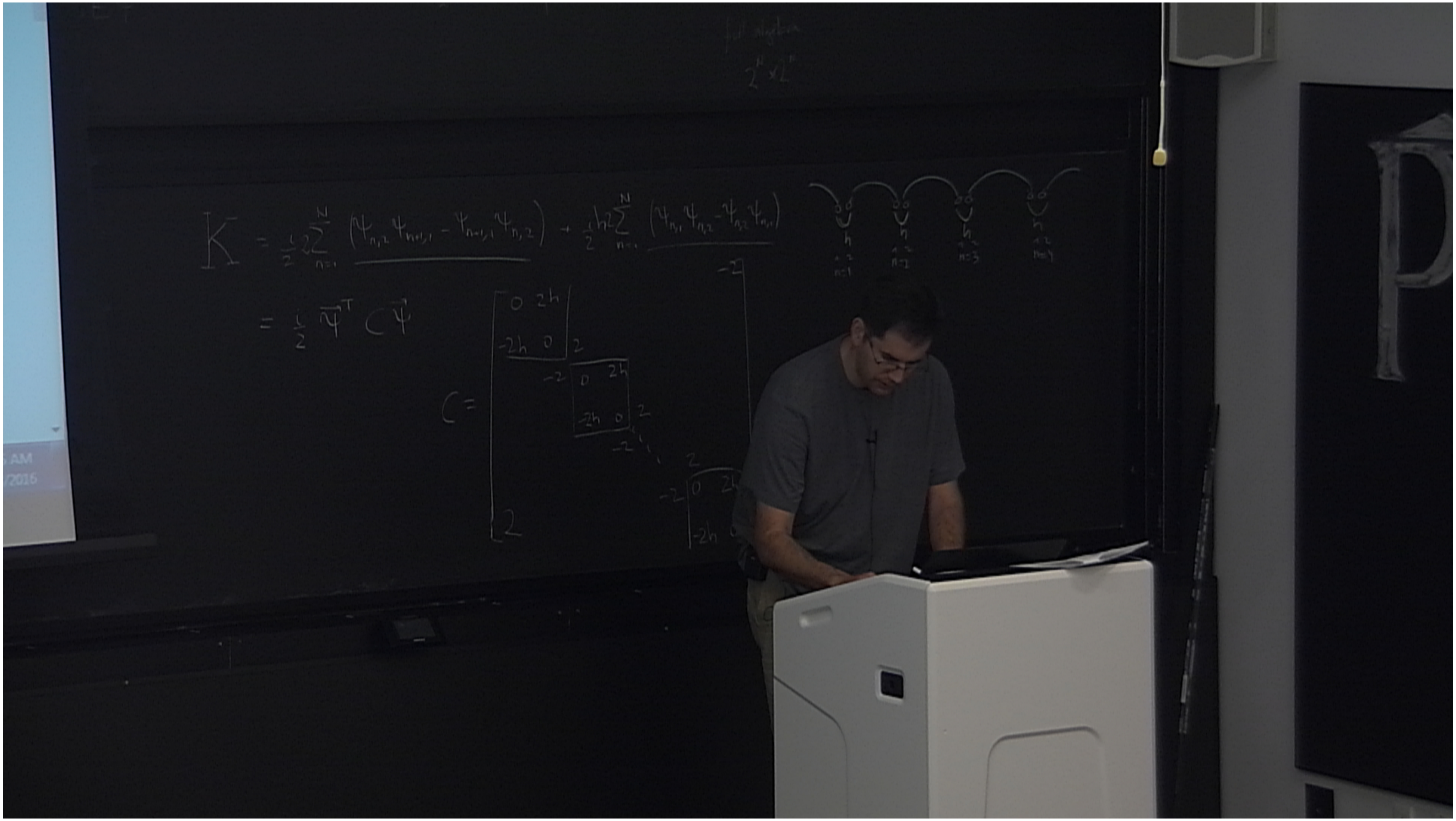

N=1000
What is going on?
$[\psi_I \psi_J, \psi_K] = \ldots = \sum A_{IJK}{}^L \psi_L$
quadratic
linear
linear
$[\sigma_n^\alpha \sigma_m^\beta, \sigma_l^\gamma] = \ldots = \sum \ldots \sigma_p^\delta \sigma_q^\epsilon$
quadratic
linear
quadratic
$[\psi\psi, \psi] = \psi$
$[\sigma\sigma, \sigma] = \sigma\sigma$
$S\vec{\psi}$
$\mathcal{U} \vec{\psi} \mathcal{U}^\dagger = \vec{\psi} - i\epsilon[M, \vec{\psi}] = (\mathbb{1} - i\epsilon S)\vec{\psi} = R\vec{\psi}$
$2^N \times 2^N$
$2N \times 2N$
$2^N \times 2^N$
$\mathcal{U} = e^{-i\epsilon M}$
$M = \frac{i}{2} \vec{\psi}^T E \vec{\psi}$


N=1000
What is going on?
[ψ_I ψ_J, ψ_K] = ... = Σ A_IJK^L ψ_L
quadratic
linear
linear
S ψ
u ψ u† = ψ − iε[M, ψ] = (1 − iεS) ψ = R ψ
2^N × 2^N
2N × 2N
u = e^{−iεM}
M = ½ ψ^T E ψ
[σ σ, σ] = ... = Σ ... σ σ
quadratic
linear
u K u†


11:01 AM
24/03/2016



full algebra
$2^N \times 2^N$
$K = i \sum_{n=1}^{N} (\psi_{n,2}\psi_{n+1,1} - \psi_{n+1,1}\psi_{n,2}) + ih \sum_{n=1}^{N} (\psi_{n,1}\psi_{n,2} - \psi_{n,2}\psi_{n,1})$
n=1
n=2
n=3
n=4



localhost:8888/notebooks/Documents/PSI%202016/PSI2016IJuliaIV.ipynb

jupyter PSI2016IJuliaIV Last Checkpoint: an hour ago (unsaved changes)

File Edit View Insert Cell Kernel Help Julia 0.4.0

```
    end
    for i=1:N-1
        J[2i,2i+1] = 2
        J[2i+1,2i] = -2
    end
    J[1,2N] = -2*bc
    J[2N,1] = 2*bc
    return C = Z+J
end

C = buildC(N,h)
```

Out[2]:

```
8x8 Array{Float64,2}:
  0.0   2.0   0.0   0.0   0.0   0.0   0.0  -2.0
 -2.0   0.0   2.0   0.0   0.0   0.0   0.0   0.0
  0.0  -2.0   0.0   2.0   0.0   0.0   0.0   0.0
  0.0   0.0  -2.0   0.0   2.0   0.0   0.0   0.0
  0.0   0.0   0.0  -2.0   0.0   2.0   0.0   0.0
  0.0   0.0   0.0   0.0  -2.0   0.0   2.0   0.0
  0.0   0.0   0.0   0.0   0.0  -2.0   0.0   2.0
  2.0   0.0   0.0   0.0   0.0   0.0  -2.0   0.0
```

#### Fermionic translation operator Tf

In [ ]:

```
# Translation operator Tf
```

jupyter PSI2016IJuliaIV Last Checkpoint: an hour ago (unsaved changes)

## Fermionic translation operator Tf

In [ ]:
```julia
# Translation operator Tf

function buildTf(N,bc=1) # choose bc=1 (default) for PBC and bc=-1 for APBC
    Tf = zeros(2N,2N)
    for i=1:2N-2
        Tf[i,i+2] = 1
    end
    Tf[2N-1,1] = bc
    Tf[2N,2] = bc
    return(Tf)
end

Tf = buildTf(N)
```

In [ ]:
```julia
# Test: do C and Tf commute?
C = buildC(N,h,1)
Tf = buildTf(N,1)
isequal(C*Tf,Tf*C)
```

## Simultaneous diagonalization of Hamiltonian C and translation operator Tf

jupyter PSI2016IJuliaIV Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Help — Julia 0.4.0

```
C = buildC(N,n,1)
Tf = buildTf(N,1)
isequal(C*Tf,Tf*C)
```

Out[5]: true

### Simultaneous diagonalization of Hamiltonian C and translation operator Tf

### Single particle energies $e_\alpha$ and momenta $k_\alpha$

In [ ]:

```
C1 = buildC(N,1,1)  # periodic boundary conditions PBC
Tf1 = buildTf(N,1)
D,U = eig(C1*(eye(2N)+0.13*Tf1))
e1 = real(diag(im*U'*C1*U))
k1 = angle(diag(U'*Tf1*U))
figure(figsize=(10,4))
plot(k1,e1, marker = "o", linestyle = "", color = "b")

[e1 k1]
```

In [ ]:

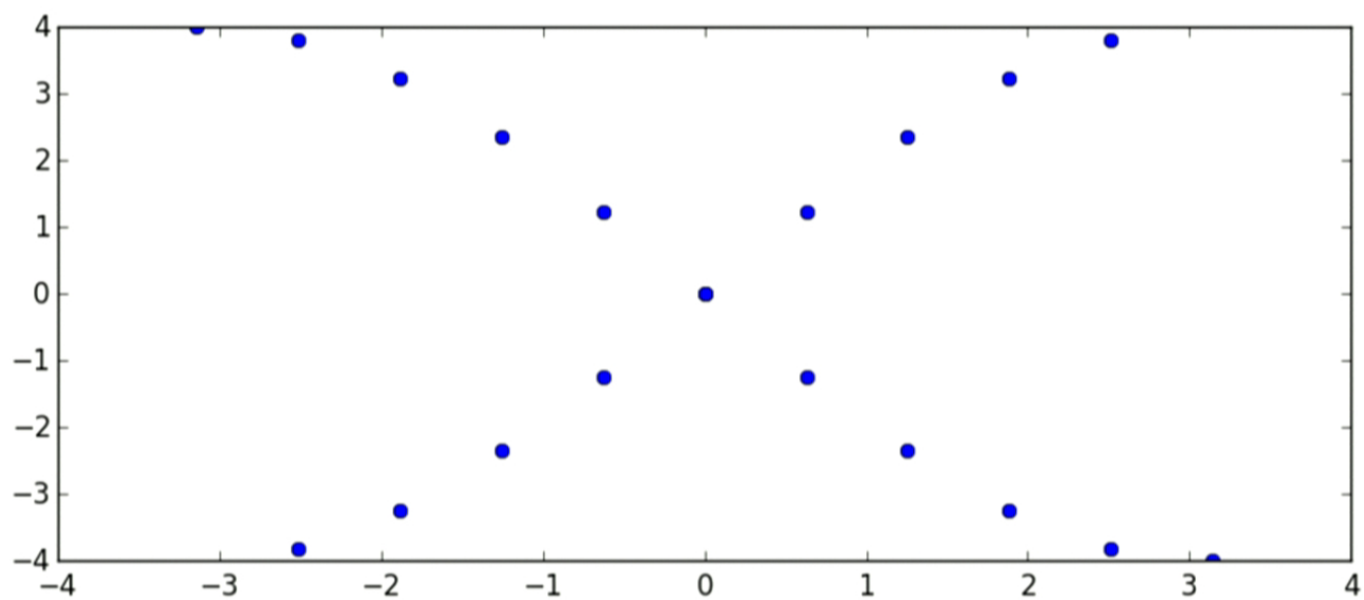
```
perm = sortperm(e1) # let us sort out the single particle energies
e1 = e1[perm]
k1 = k1[perm]
[e1 k1]
```

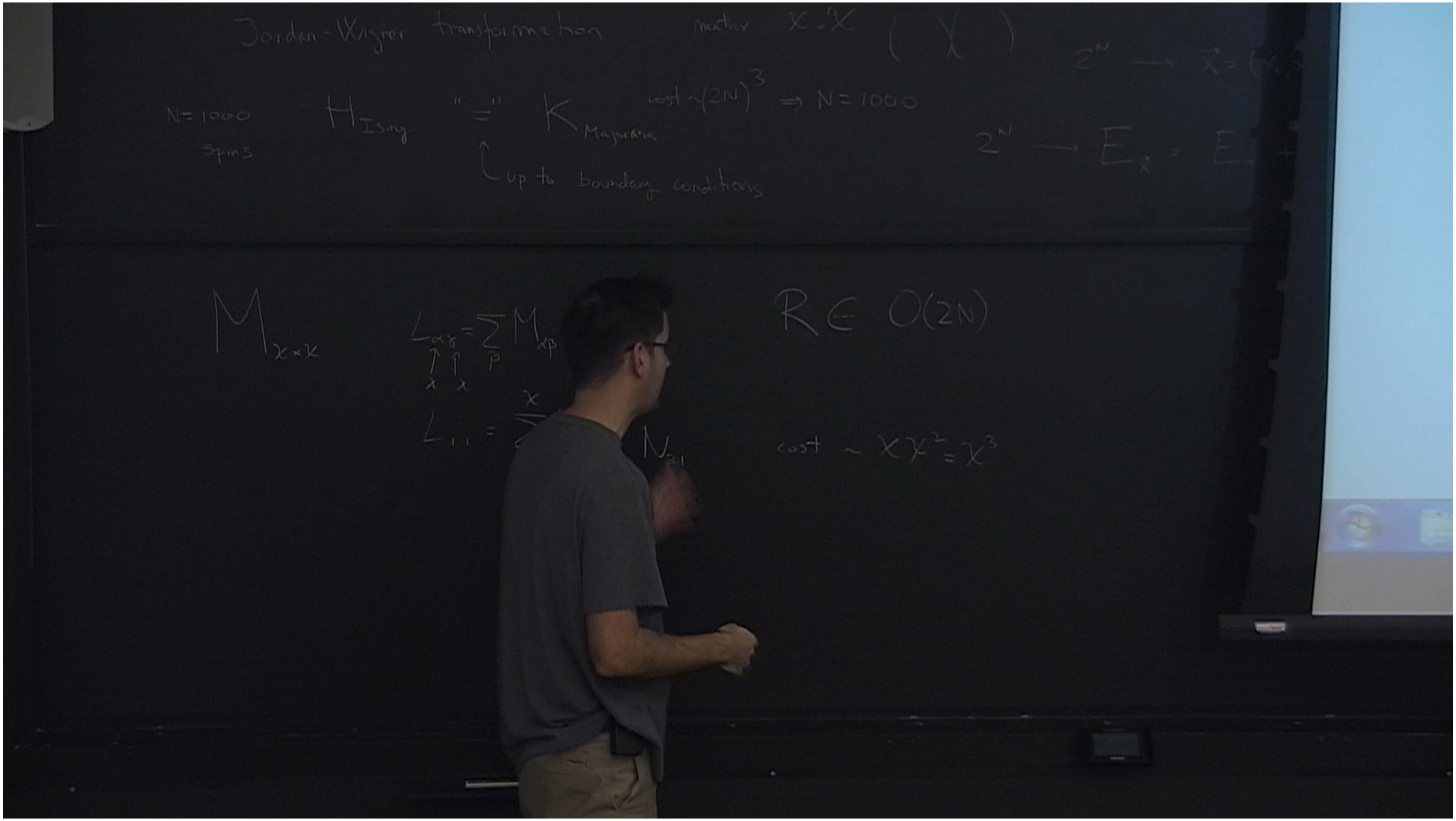
jupyter PSI2016IJuliaIV Last Checkpoint: an hour ago (unsaved changes)

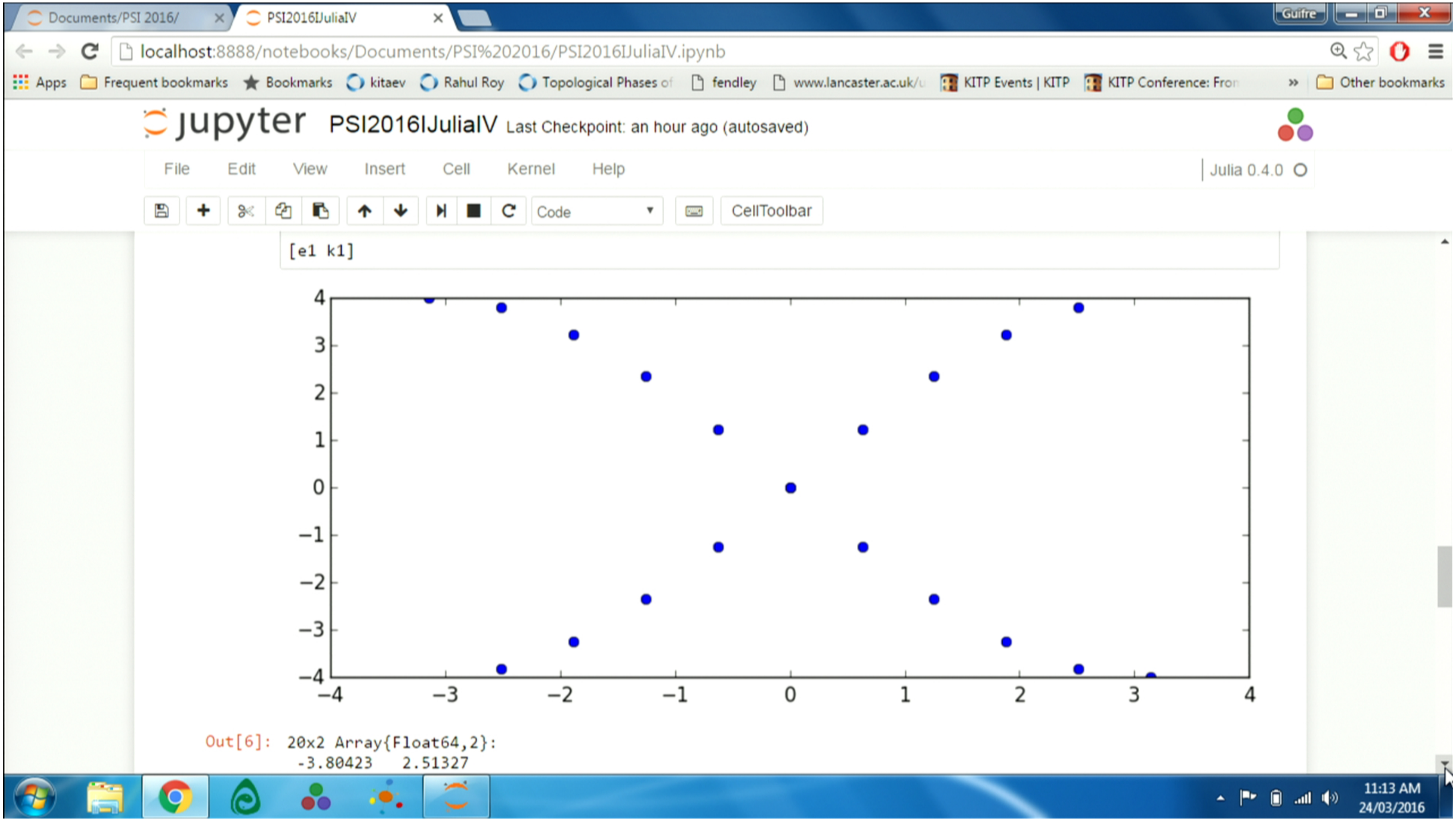
File Edit View Insert Cell Kernel Help Julia 0.4.0

```
figure(figsize=(10,4))
plot(k1,e1, marker = "o", linestyle = "", color = "b")

[e1 k1]
```




Jordan-Wigner transformation
N=1000
spins
up to boundary conditions
cost ~(2N)^3 ⇒ N=1000
R ∈ O(2N)
cost


Documents/PSI 2016/
PSI2016IJuliaIV
localhost:8888/notebooks/Documents/PSI%202016/PSI2016IJuliaIV.ipynb
Apps
Frequent bookmarks
Bookmarks
kitaev
Rahul Roy
Topological Phases of
fendley
www.lancaster.ac.uk/
KITP Events | KITP
KITP Conference: Fro
Other bookmarks
jupyter
PSI2016IJuliaIV
Last Checkpoint: an hour ago (autosaved)
File
Edit
View
Insert
Cell
Kernel
Help
Julia 0.4.0
Code
CellToolbar
[e1 k1]
4
3
2
1
0
−1
−2
−3
−4
−4
−3
−2
−1
0
1
2
3
4
Out[6]:
20x2 Array{Float64,2}:
-3.80423   2.51327
11:13 AM
24/03/2016