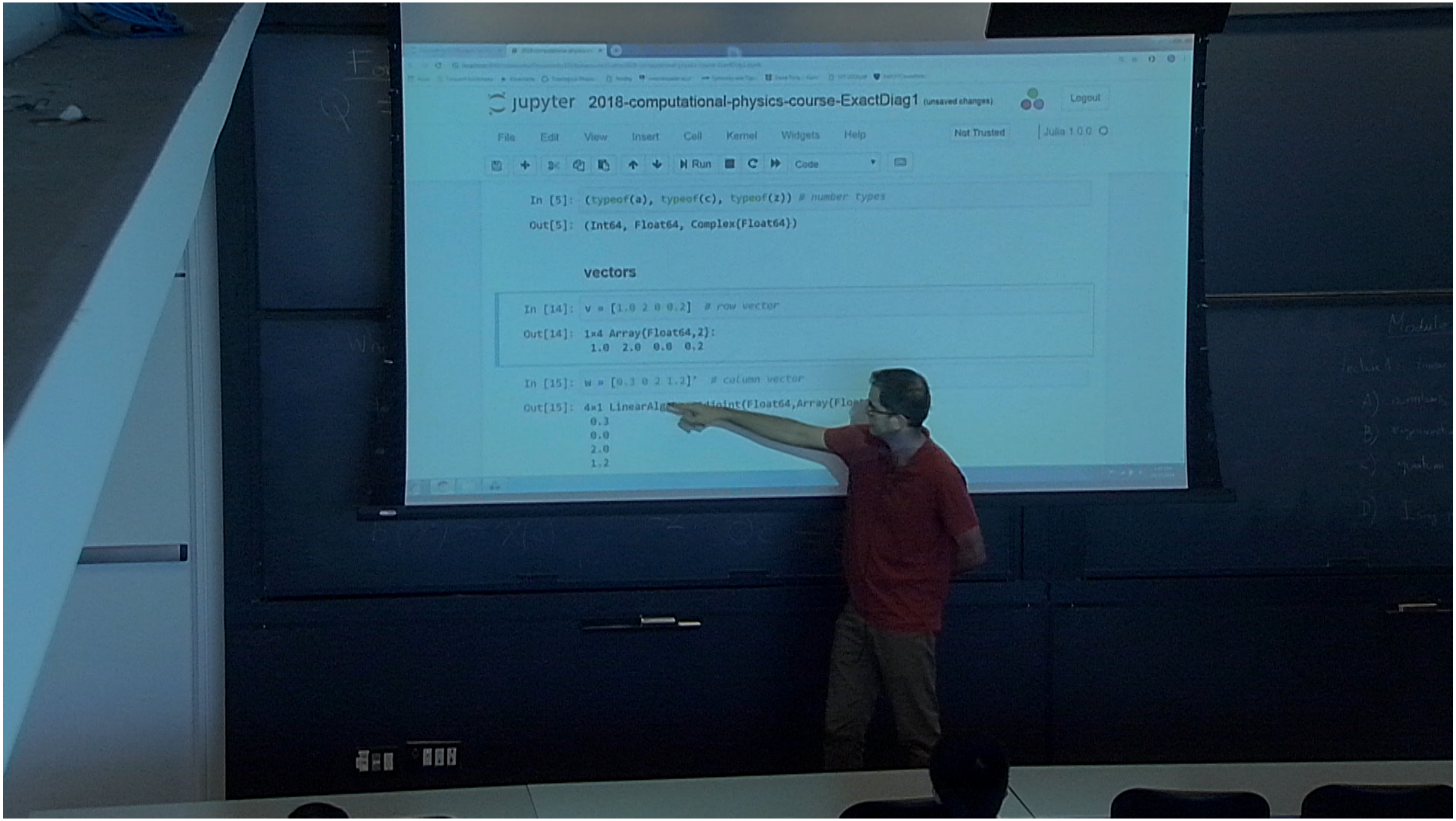
Title: Computational Physics - Lecture 19

Date: Nov 14, 2018 01:00 PM

URL: http://pirsa.org/18110045

Abstract:


jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Julia 1.0.0
Run
Code
In [5]: (typeof(a), typeof(c), typeof(z)) # number types
Out[5]: (Int64, Float64, Complex{Float64})
vectors
In [14]: v = [1.0 2 0 0.2] # row vector
Out[14]: 1×4 Array{Float64,2}:
1.0 2.0 0.0 0.2
In [15]: w = [0.3 0 2 1.2]' # column vector
0.3
0.0
2.0
1.2

jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)

```
In [5]: (typeof(a), typeof(c), typeof(z)) # number types

Out[5]: (Int64, Float64, Complex{Float64})
```

vectors

```
In [*]: v = [1.0 2 0 0.2]  # row vector

In [15]: w = [0.3 0 2 1.2]'  # column vector

Out[15]: 4×1 LinearAlgebra.Adjoint{Float64,Array{Float64,2}}:
 0.3
 0.0
 2.0
 1.2

In [16]: w = randn(4,1) + im*randn(4,1) # random column vector --> try randn(1,4) to creat
```

Documents/2018juliacourse/Gui × 2018-computational-physics-co ×

localhost:8888/notebooks/Documents/2018juliacourse/Guifre/2018-computational-physics-course-ExactDiag1.ipynb

Apps Frequent bookmarks Bookmarks Topological Phases fendley www.lancaster.ac.uk Symmetry and Topo David Tong -- Camb CFT-2014.pdf AdSCFTCourseNotes

jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

Run Code

```
 -0.4288384710133761 + 0.533447899128118im
 -1.2355316579953635 + 0.6877862363268086im
  -2.347742453257248 + 0.5361118233472415im
```

In [17]:
```
w[2] # access component 2 of the vector w
```

Out[17]:
```
0.013788021878399444 - 1.812236091312663im
```

In [18]:
```
v*w # scalar product as a 1x1 array
```

Out[18]:
```
1×1 Array{Complex{Float64},2}:
 0.9129387992719952 - 4.687110701636254im
```

In [19]:
```
(v*w)[1] # scalar product as a single number
```

Out[19]:
```
0.9129387992719952 - 4.687110701636254im
```

In [20]:
```
sqrt((w'*w)[1]) # norm of w
```

Out[20]:
```
2.4810720321289126 + 0.0im
```

1:08 PM 14/11/2018

jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)

```
 -0.4288384710133761 + 0.533447899128118im
 -1.2355316579953635 + 0.6877862363268086im
  -2.347742453257248 + 0.5361118233472415im

In [12]: w[2] # access component 2 of the vector w

Out[12]: -0.4288384710133761 + 0.533447899128118im

In [13]: v*w # scalar product as a 1x1 array

Out[13]: 1×1 Array{Complex{Float64},2}:
          -1.4562674511799953 + 1.4511718170798624im

In [14]: (v*w)[1] # scalar product as a single number

Out[14]: -1.4562674511799953 + 1.4511718170798624im

In [20]: sqrt((w'*w)[1]) # norm of w

Out[20]: 2.4810720321289126 + 0.0im
```

localhost:8888/notebooks/Documents/2018juliacourse/Guifre/2018-computational-physics-course-ExactDiag1.ipynb

jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

```
 0.661698+0.260063im    1.01979+1.58037im   -0.619617+0.505106im
```

In [24]:
```
M' # hermitian conjugate
```

Out[24]:
```
3×3 LinearAlgebra.Adjoint{Complex{Float64},Array{Complex{Float64},2}}:
 0.352758+0.810127im   0.380101+0.469409im   0.661698-0.260063im
  1.06993+0.314252im  -0.401652+0.771544im    1.01979-1.58037im
  1.06289-1.89936im   -0.420209+0.596753im  -0.619617-0.505106im
```

In [25]:
```
M = (M+M')/2 # Let us make M Hermitian!
```

Out[25]:
```
3×3 Array{Complex{Float64},2}:
 0.352758+0.0im          0.725014+0.0775789im   0.862292+0.819646im
 0.725014-0.0775789im  -0.401652+0.0im          0.299789-1.08856im
 0.862292-0.819646im    0.299789+1.08856im    -0.619617+0.0im
```

eigenvector decomposition

jupyter 2018-computational-physics-course-ExactDiag1 (autosaved)

eigenvector decomposition

In [26]:
```
using LinearAlgebra
```

In [27]:
```
D,U = eigen(M) # eigenvalue decomposition
```

Out[27]:
```
Eigen{Complex{Float64},Float64,Array{Complex{Float64},2},Array{Float64,1}}
eigenvalues:
3-element Array{Float64,1}:
 -0.7877943732613641
 -0.08480037193106327
  1.7132106705651966
eigenvectors:
3×3 Array{Complex{Float64},2}:
 -0.26138+0.210345im  -0.583143-0.508061im  -0.537762-0.00810858im
 -0.61876-0.161217im   0.279073+0.485567im  -0.524431+0.0495952im
 0.691795+0.0im        0.296919+0.0im       -0.658224-0.0im
```

jupyter 2018-computational-physics-course-ExactDiag1 (autosaved)

eigenvector decomposition

In [26]:
```julia
using LinearAlgebra
```

In [27]:
```julia
D,U = eigen(M) # eigenvalue decomposition
```

Out[27]:
```
Eigen{Complex{Float64},Float64,Array{Complex{Float64},2},Array{Float64,1}}
eigenvalues:
3-element Array{Float64,1}:
 -0.7877943732613641
 -0.08480037193106327
  1.7132106705651966
eigenvectors:
3×3 Array{Complex{Float64},2}:
 -0.26138+0.210345im  -0.583143-0.508061im  -0.537762-0.00810858im
 -0.61876-0.161217im   0.279073+0.485567im  -0.524431+0.0495952im
 0.691795+0.0im        0.296919+0.0im       -0.658224-0.0im
```

In [28]:
```julia
D  # eigenvalues
```

$$M = M^\dagger$$

$$M = \mathcal{U} D \mathcal{U}^\dagger$$

$\mathcal{U}$ unitary $\quad \mathcal{U}\mathcal{U}^\dagger = \mathcal{U}^\dagger \mathcal{U} = \mathbb{1}$

D diagonal $\begin{pmatrix} \lambda_1 & & \\ & \lambda_2 & \\ & & \ddots \end{pmatrix} \in \mathbb{R}$

jupyter 2018-computational-physics-course-ExactDiag1 (autosaved)

eigenvector decomposition

```
In [26]: using LinearAlgebra
```

```
In [27]: D,U = eigen(M) # eigenvalue decomposition
```

```
Out[27]: Eigen{Complex{Float64},Float64,Array{Complex{Float64},2},Array{Float64,1}}
eigenvalues:
3-element Array{Float64,1}:
 -0.7877943732613641
 -0.08480037193106327
  1.7132106705651966
eigenvectors:
3×3 Array{Complex{Float64},2}:
 -0.26138+0.210345im  -0.583143-0.508061im  -0.537762-0.00810858im
 -0.61876-0.161217im   0.279073+0.485567im  -0.524431+0.0495952im
 0.691795+0.0im        0.296919+0.0im       -0.658224-0.0im
```

```
In [28]: D  # eigenvalues
```

jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help | Not Trusted | Julia 1.0.0

Out[35]:

```
4×4 Array{Float64,2}:
 0.0  -2.2144  0.0        0.0
 0.0   0.0     0.0605539  0.0
 0.0   0.0     0.0        1.48533
 0.0   0.0     0.0        0.0
```

In [34]:

```julia
N = U*diagm(0=>D)*U' # Let us build UDU'
display(M) # Are M and
display(N) # N the same?
```

```
DimensionMismatch("A has dimensions (6,3) but B has dimensions (4,4)")

Stacktrace:
 [1] gemm_wrapper!(::Base.ReinterpretArray{Float64,2,Complex{Float64},Array{Complex{Float64},2}}, ::Char, ::Char, ::Base.ReinterpretArray{Float64,2,Complex{Float64},Array{Complex{Float64},2}}, ::Array{Float64,2}) at C:\cygwin\home\Administrator\buildbot\worker\package_win64\build\usr\share\julia\stdlib\v1.0\LinearAlgebra\src\matmul.jl:436
 [2] mul!(::Array{Complex{Float64},2}, ::Array{Complex{Float64},2}, ::Array{Float64,2}) at C:\cygwin\home\Administrator\buildbot\worker\package_win64\build\usr\share\julia\stdlib\v1.0\LinearAlgebra\src\matmul.jl:149
```

Is it true that $M = U \cdot D \cdot U^\dagger$?

```
In [36]: diagm(0=>D)

Out[36]: 4×4 Array{Float64,2}:
           0.0      0.0        0.0      0.0
          -2.2144   0.0        0.0      0.0
           0.0      0.0605539  0.0      0.0
           0.0      0.0        1.48533  0.0
```

```
In [34]: N = U*diagm(0=>D)*U' # Let us build UDU'
         display(M) # Are M and
         display(N) # N the same?
```

```
DimensionMismatch("A has dimensions (6,3) but B has dimensions (4,4)")

Stacktrace:
 [1] gemm_wrapper!(::Base.ReinterpretArray{Float64,2,Complex{Float64},Array{Com
plex{Float64},2}}, ::Char, ::Char, ::Base.ReinterpretArray{Float64,2,Complex{Fl
```

jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)

Out[37]:
```
3×3 Array{Float64,2}:
 -2.2144  0.0        0.0
  0.0     0.0605539  0.0
  0.0     0.0        1.48533
```

In [34]:
```julia
N = U*diagm(0=>D)*U' # Let us build UDU'
display(M) # Are M and
display(N) # N the same?
```

```
DimensionMismatch("A has dimensions (6,3) but B has dimensions (4,4)")

Stacktrace:
 [1] gemm_wrapper!(::Base.ReinterpretArray{Float64,2,Complex{Float64},Array{Complex{Float64},2}}, ::Char, ::Char, ::Base.ReinterpretArray{Float64,2,Complex{Float64},Array{Complex{Float64},2}}, ::Array{Float64,2}) at C:\cygwin\home\Administrator\buildbot\worker\package_win64\build\usr\share\julia\stdlib\v1.0\LinearAlgebra\src\matmul.jl:436
 [2] mul!(::Array{Complex{Float64},2}, ::Array{Complex{Float64},2}, ::Array{Float64,2}) at C:\cygwin\home\Administrator\buildbot\worker\package_win64\build\usr\share\julia\stdlib\v1.0\LinearAlgebra\src\matmul.jl:149
 [3] *(::Array{Complex{Float64},2}, ::Array{Float64,2}, ::Adjoint{Complex{Float
```

localhost:8888/notebooks/Documents/2018juliacourse/Guifre/2018-computational-physics-course-ExactDiag1.ipynb

jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help | Not Trusted | Julia 1.0.0

```
 0.725014-0.0775789im  -0.401652+0.0im        0.299789-1.08856im
 0.862292-0.819646im    0.299789+1.08856im   -0.619617+0.0im

3×3 Array{Complex{Float64},2}:
 0.352758+0.0im          0.725014+0.0775789im     0.862292+0.819646im
 0.725014-0.0775789im  -0.401652-1.73472e-18im   0.299789-1.08856im
 0.862292-0.819646im    0.299789+1.08856im      -0.619617+0.0im
```

In [31]:

```
display(M-N) # yes, up to numerical precision
println("The norm of M-N is ", norm(M-N))
```

```
3×3 Array{Complex{Float64},2}:
 -1.05471e-15+0.0im           …  1.55431e-15-1.80411e-16im
 -8.88178e-16+0.0im              1.66533e-15+1.38778e-17im
  1.55431e-15+1.66533e-16im      -2.498e-15+0.0im

The norm of M-N is 4.592945026354102e-15
```

In [32]:

```
evalue1 = D[1] # first eigenvalue of matrix M
display(U)
evector1 = U[:,1] # first eigenvector of matrix M
```

jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help | Not Trusted | Julia 1.0.0

```
 3.33067e-16+8.88178e-16im     5.55112e-16-1.9984e-15im
 1.77636e-15-1.44329e-15im    -4.21885e-15+0.0im

The norm of M-N is 6.563709735601526e-15
```

In [32]:

```julia
evalue1 = D[1] # first eigenvalue of matrix M
display(U)
evector1 = U[:,1] # first eigenvector of matrix M
```

```
3×3 Array{Complex{Float64},2}:
 -0.26138+0.210345im  -0.583143-0.508061im  -0.537762-0.00810858im
 -0.61876-0.161217im   0.279073+0.485567im  -0.524431+0.0495952im
 0.691795+0.0im        0.296919+0.0im       -0.658224-0.0im
```

Out[32]:

```
3-element Array{Complex{Float64},1}:
  -0.261379955705327 + 0.21034487093814058im
 -0.6187600454178341 - 0.16121734745982072im
  0.6917951482180529 + 0.0im
```

Is it true that $M \cdot \vec{v}_1 = \lambda_1 \vec{v}_1$?

jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)

```
3×3 Array{Complex{Float64},2}:
 -0.214142-0.382299im   -0.476417+0.263338im  -0.663063-0.268361im
 -0.0137253+0.563976im  -0.408939-0.575111im  -0.353472+0.24252im
  0.699809+0.0im         0.453537+0.0im        -0.55188-0.0im
```

Out[42]:
```
3-element Array{Complex{Float64},1}:
   -0.2141420256285332 - 0.3822992827398361im
 -0.013725325529155508 + 0.5639764369260704im
    0.6998090063064409 + 0.0im
```

Is it true that $M \cdot \vec{v}_1 = \lambda_1 \vec{v}_1$?

In [33]:
```
display(M*evector1) #
display(evalue1*evector1) #
```

```
3-element Array{Complex{Float64},1}:
 0.20591365838796288 - 0.16570850576945528im
  0.4874556821791179 + 0.1270061192009691im
 -0.5449923252156972 - 5.551115123125783e-17im

3-element Array{Complex{Float64},1}:
```

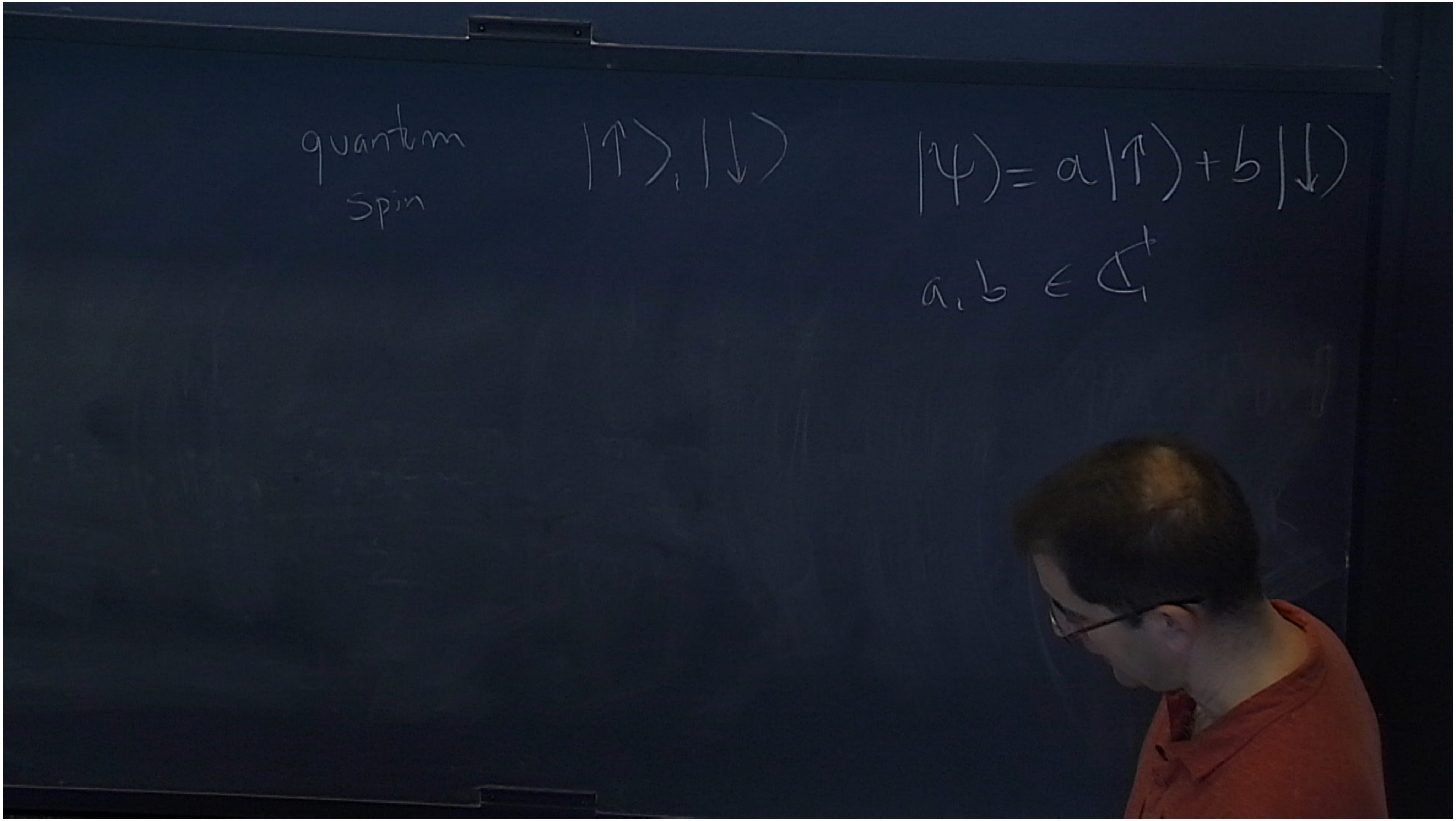
jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)

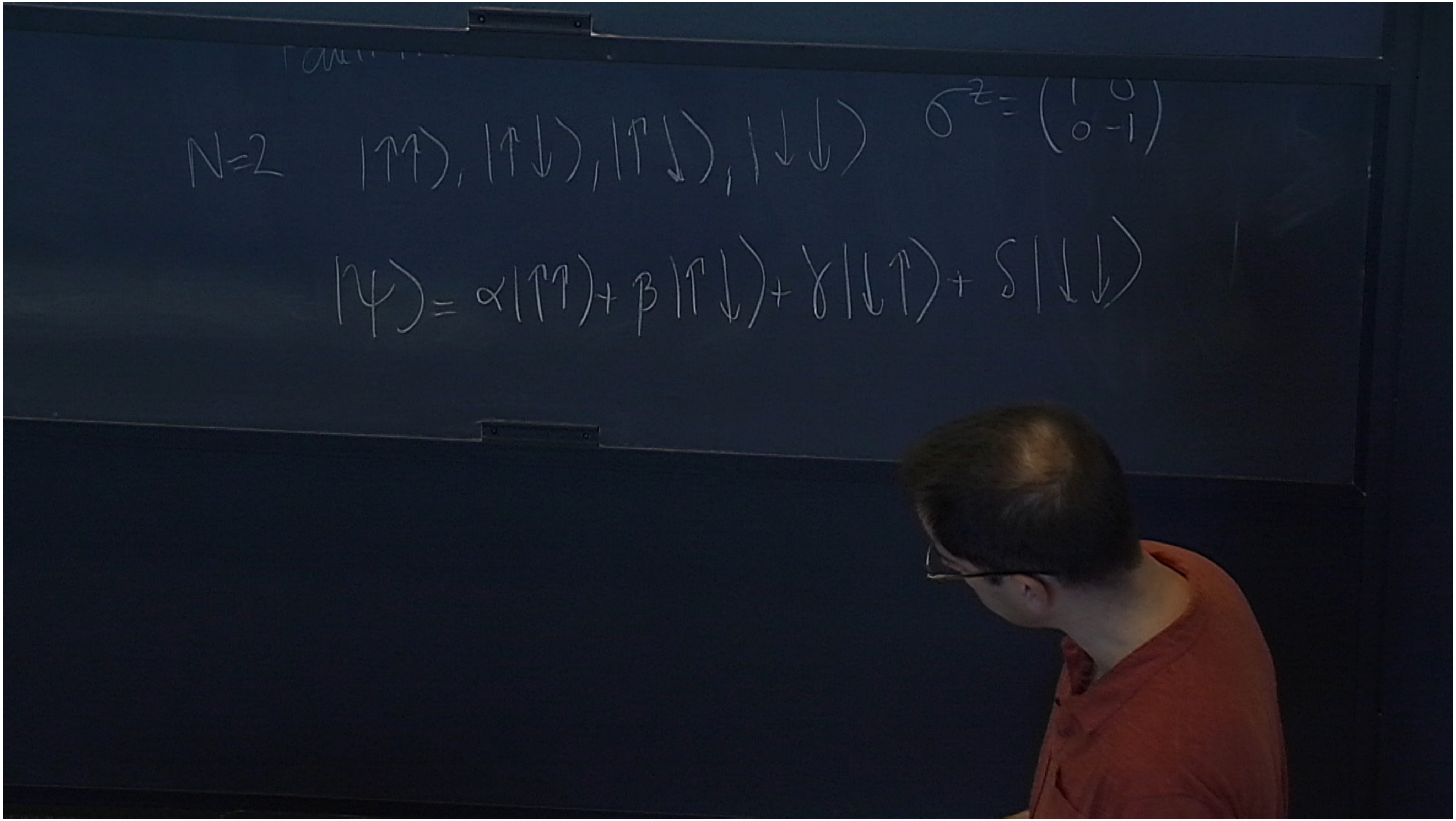
File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

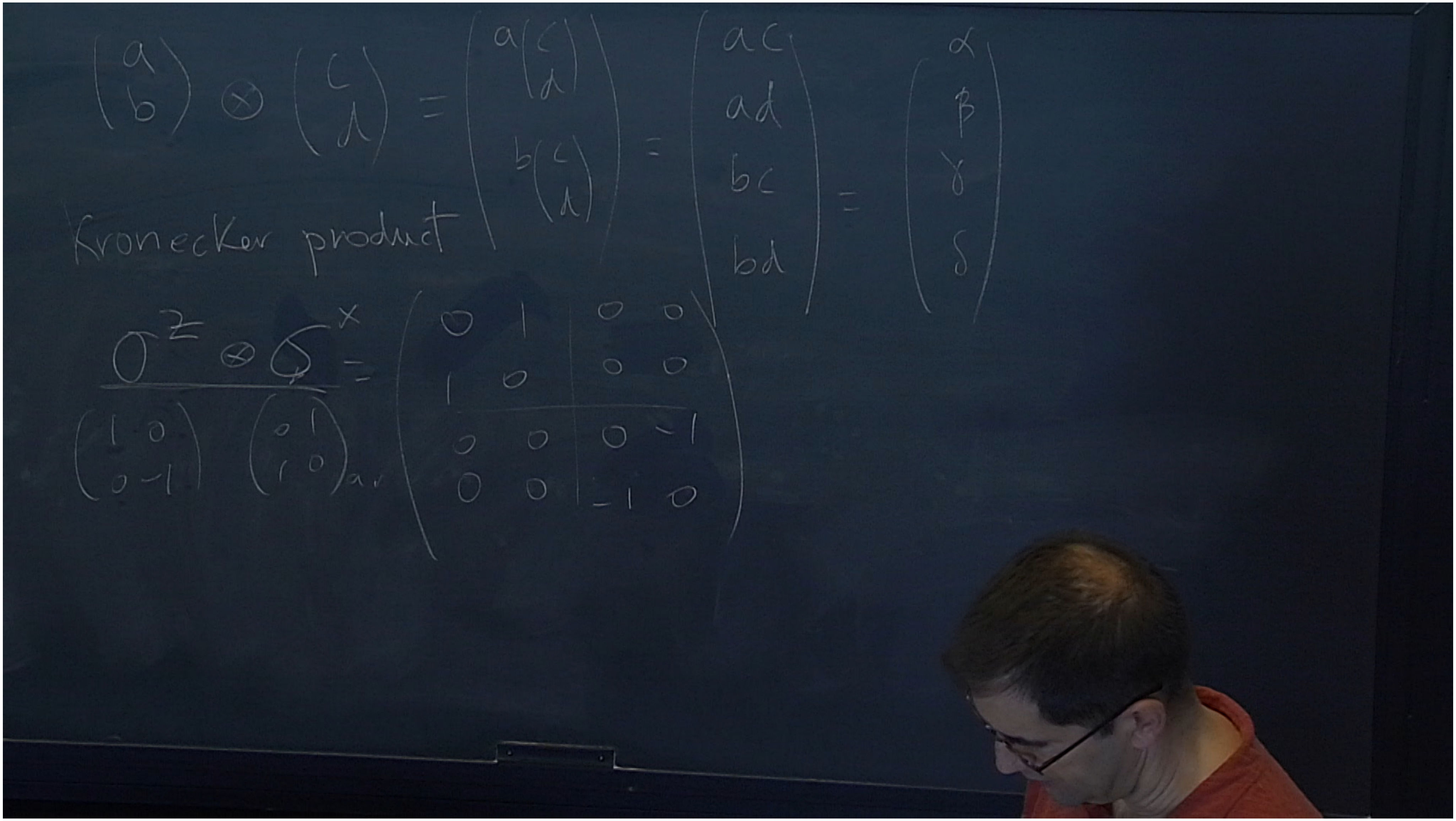
```
3-element Array{Complex{Float64},1}:
   0.4741956370961859 + 0.8465627025248235im
 0.030393331082711805 - 1.2488681987126957im
   -1.549655545739428 - 5.551115123125783e-17im

3-element Array{Complex{Float64},1}:
  0.4741956370961839 + 0.8465627025248218im
 0.03039333108271109 - 1.2488681987126935im
  -1.549655545739423 - 0.0im
```

Kronecker product $$A \otimes B = \begin{bmatrix} a_{11}B & a_{12}B & \cdots & a_{1n}B \\ a_{21}B & a_{22}B & \cdots & a_{2n}B \\ \vdots & \vdots & \cdots & \vdots \\ a_{n1}B & a_{n2}B & \cdots & a_{nn}B \end{bmatrix}$$


quantum
spin
|↑⟩, |↓⟩
|ψ⟩ = a|↑⟩ + b|↓⟩
a, b ∈ ℂ


N=2 |↑↑⟩, |↑↓⟩, |↑↓⟩, |↓↓⟩
σ^z = (1 0 / 0 -1)
|ψ⟩ = α|↑↑⟩ + β|↑↓⟩ + γ|↓↑⟩ + δ|↓↓⟩


$\begin{pmatrix} a \\ b \end{pmatrix} \otimes \begin{pmatrix} c \\ d \end{pmatrix} = \begin{pmatrix} a\begin{pmatrix} c \\ d \end{pmatrix} \\ b\begin{pmatrix} c \\ d \end{pmatrix} \end{pmatrix} = \begin{pmatrix} ac \\ ad \\ bc \\ bd \end{pmatrix} = \begin{pmatrix} \alpha \\ \beta \\ \gamma \\ \delta \end{pmatrix}$
Kronecker product
$\sigma^z \otimes \sigma^x = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 \end{pmatrix}$
$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \quad \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$

jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help | Not Trusted | Julia 1.0.0

Kronecker product $$A \otimes B = \begin{bmatrix} a_{21}B & a_{22}B & \cdots & a_{2n}B \\ \vdots & \vdots & \cdots & \vdots \\ a_{n1}B & a_{n2}B & \cdots & a_{nn}B \end{bmatrix}$$

In [46]:
```
E = [1 0; 0 1]
```

Out[46]:
```
2×2 Array{Int64,2}:
 1  0
 0  1
```

In [47]:
```
X = [0 1; 1 0]
```

Out[47]:
```
2×2 Array{Int64,2}:
 0  1
 1  0
```

In [57]:
```
A = [1 2; 3 4]
```

jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)

```
Out[48]: 2×2 Array{Int64,2}:
          1  2
          3  4
```

Example:

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} a & b & 0 & 0 \\ c & d & 0 & 0 \\ 0 & 0 & a & b \\ 0 & 0 & c & d \end{bmatrix}$$

```
In [59]: kron(E,A)

Out[59]: 4×4 Array{Int64,2}:
          1  2  0  0
          3  4  0  0
          0  0  1  2
          0  0  3  4
```

jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)

```
3  4
```

Example:

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} a & b & 0 & 0 \\ c & d & 0 & 0 \\ 0 & 0 & a & b \\ 0 & 0 & c & d \end{bmatrix}$$

In [49]:
```
kron(E,A)
```

Out[49]:
```
4x4 Array{Int64,2}:
 1  2  0  0
 3  4  0  0
 0  0  1  2
 0  0  3  4
```

In [38]:
```
kron(X,A)
```

jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help — Not Trusted — Julia 1.0.0

Out[51]:
```
4×4 Array{Int64,2}:
 1  0  2  0
 0  1  0  2
 3  0  4  0
 0  3  0  4
```

In [52]:
```
kron(A,A)
```

Out[52]:
```
4×4 Array{Int64,2}:
 1   2   2   4
 3   4   6   8
 3   6   4   8
 9  12  12  16
```

quantum Ising model on N=3 spins

$$H = \sigma_1^x \otimes \sigma_2^x + \sigma_2^x \otimes \sigma_3^x + \sigma_3^x \otimes \sigma_1^x + h\left(\sigma_1^z + \sigma_2^z + \sigma_3^z\right)$$

In [41]:
```
ones(2)
```

jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)

```
 3  0  4  0
 0  3  0  4
```

In [52]:
```julia
kron(A,A)
```

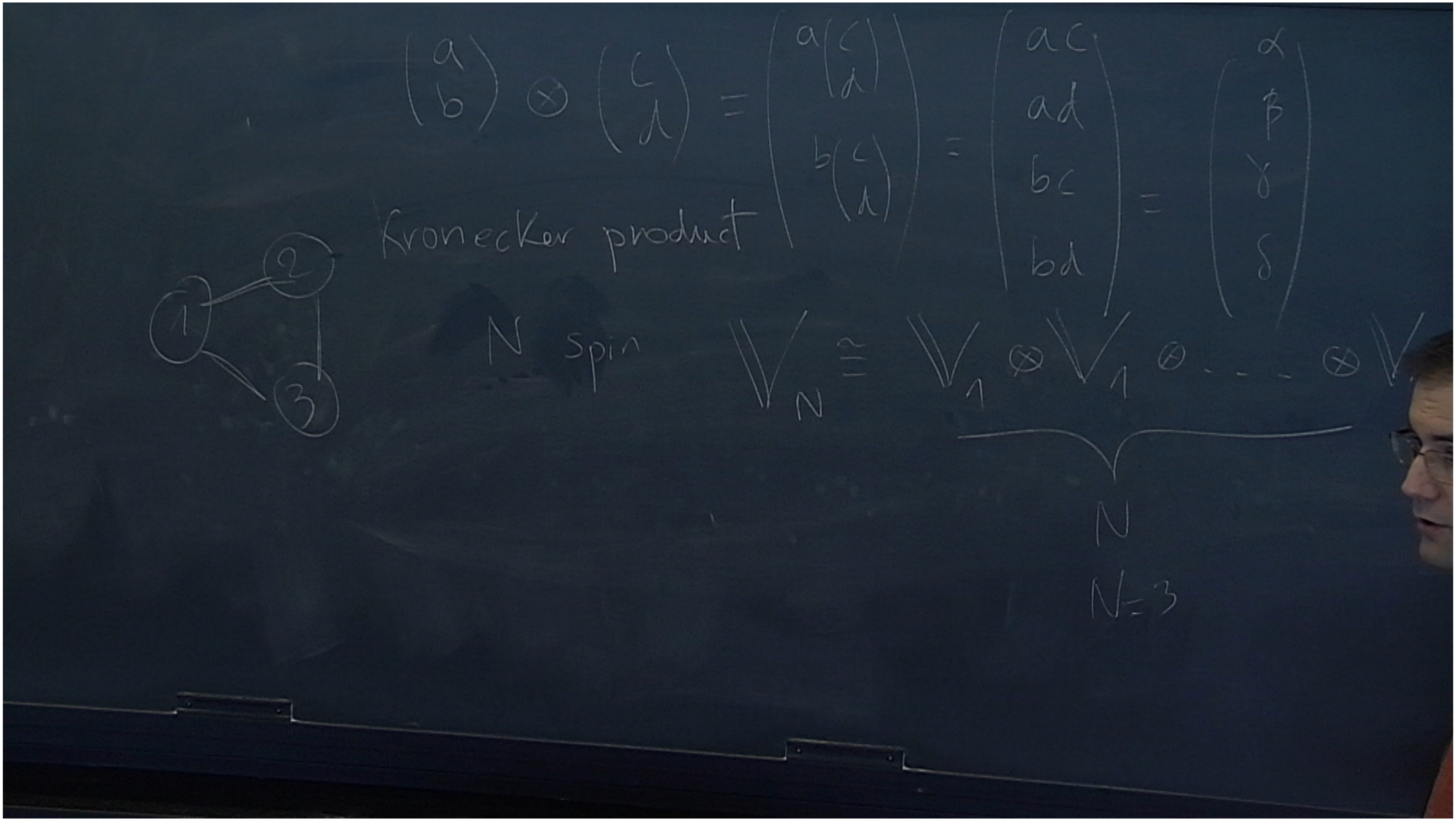
Out[52]:
```
4×4 Array{Int64,2}:
 1   2   2   4
 3   4   6   8
 3   6   4   8
 9  12  12  16
```

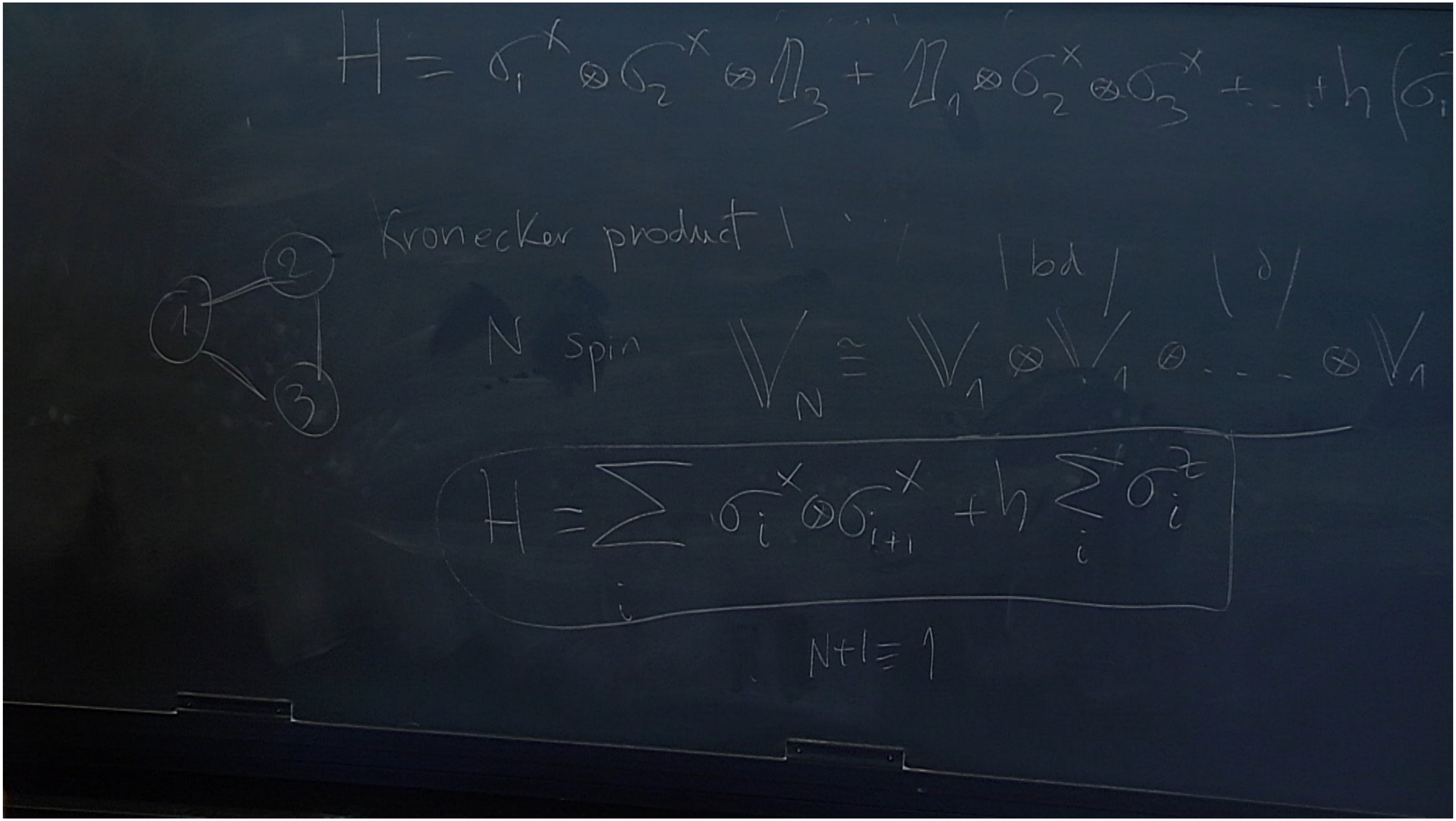
quantum Ising model on N=3 spins

$$H = \sigma_1^x \otimes \sigma_2^x + \sigma_2^x \otimes \sigma_3^x + \sigma_3^x \otimes \sigma_1^x + h\left(\sigma_1^z + \sigma_2^z + \sigma_3^z\right)$$

In [41]:
```julia
ones(2)
```

Out[41]:
```
2-element Array{Float64,1}:
 1.0
```


(a b) ⊗ (c d) = (a(c d) b(c d)) = (ac ad bc bd) = (α β γ δ)
Kronecker product
1
2
3
N spin
V_N ≅ V_1 ⊗ V_1 ⊗ . . . ⊗ V
N
N=3


H = σ1x ⊗ σ2x ⊗ 𝟙3 + 𝟙1 ⊗ σ2x ⊗ σ3x + ... + h (σ1
Kronecker product
bd
N spin
VN ≅ V1 ⊗ V1 ⊗ ... ⊗ V1
H = Σi σix ⊗ σi+1x + h Σi σiz
N+1 ≡ 1

Documents/2018juliacourse/Gu... × 2018-computational-physics-co... ×

localhost:8888/notebooks/Documents/2018juliacourse/Guifre/2018-computational-physics-course-ExactDiag1.ipynb

Apps Frequent bookmarks Bookmarks Topological Phases fendley www.lancaster.ac.uk Symmetry and Topo David Tong -- Camb CFT-2014.pdf AdSCFTCourseNotes

jupyter 2018-computational-physics-course-ExactDiag1 (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

Run Markdown

In [52]: kron(A,A)

Out[52]:
```
4×4 Array{Int64,2}:
 1   2   2   4
 3   4   6   8
 3   6   4   8
 9  12  12  16
```

quantum Ising model on N=3 spins

$$H = \sigma_1^x \otimes \sigma_2^x + \sigma_2^x \otimes \sigma_3^x + \sigma_3^x \otimes \sigma_1^x + h\left(\sigma_1^z + \sigma_2^z + \sigma_3^z\right)$$

In [41]:
```
ones(2)
```

Out[41]:
```
2-element Array{Float64,1}:
 1.0
 1.0
```

In [62]:
```
E = diagm(0=>ones(2))
```

1:41 PM 14/11/2018

jupyter 2018-computational-physics-course-ExactDiag1 (autosaved)

Kronecker product $A \otimes B = \begin{bmatrix} a_{21}B & a_{22}B & \cdots & a_{2n}B \\ \vdots & \vdots & \cdots & \vdots \\ a_{n1}B & a_{n2}B & \cdots & a_{nn}B \end{bmatrix}$

In [46]:
```
E = [1 0; 0 1]
```
Out[46]:
```
2×2 Array{Int64,2}:
 1  0
 0  1
```

In [47]:
```
X = [0 1; 1 0]
```
Out[47]:
```
2×2 Array{Int64,2}:
 0  1
 1  0
```

In [48]:
```
A = [1 2; 3 4]
```

jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)

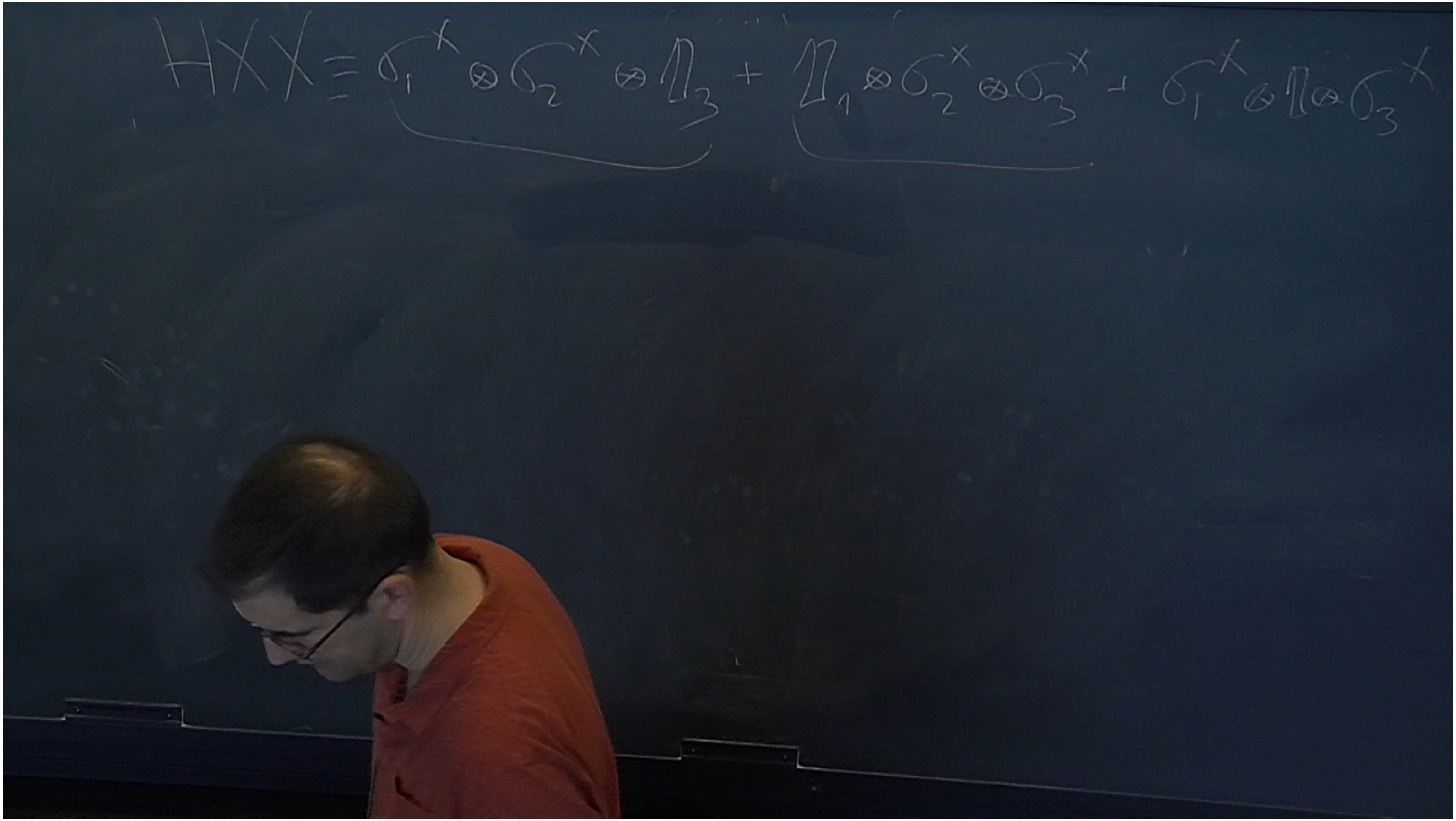
```
 1.0
 1.0
```

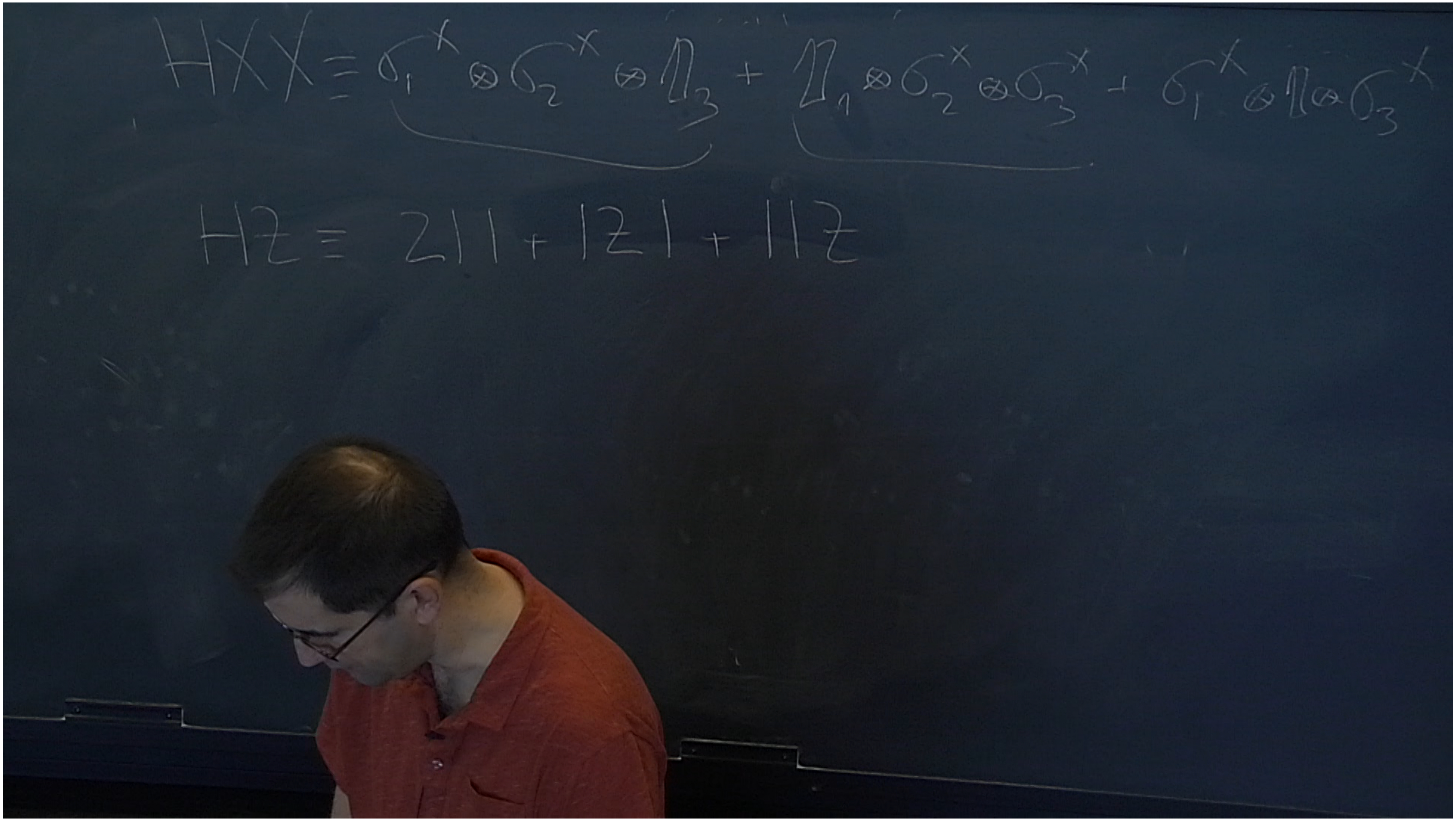
In [54]:
```julia
E = diagm(0=>ones(2))
X = [0. 1; 1 0]
Z = [1. 0; 0 -1]
EE = diagm(0=>ones(4))
XX = kron(X,X)
```

Out[54]:
```
4×4 Array{Float64,2}:
 0.0  0.0  0.0  1.0
 0.0  0.0  1.0  0.0
 0.0  1.0  0.0  0.0
 1.0  0.0  0.0  0.0
```

In [63]:
```julia
HXX = kron(XX,E) + kron(E, XX) + kron(X,kron(E,X))
```

Out[63]:
```
8×8 Array{Float64,2}:
 0.0  0.0  0.0  1.0  0.0  1.0  1.0  0.0
 0.0  0.0  1.0  0.0  1.0  0.0  0.0  1.0
 0.0  1.0  0.0  0.0  1.0  0.0  0.0  1.0
```


$H_{XX} \equiv \sigma_1^x \otimes \sigma_2^x \otimes \mathbb{1}_3 + \mathbb{1}_1 \otimes \sigma_2^x \otimes \sigma_3^x + \sigma_1^x \otimes \mathbb{1} \otimes \sigma_3^x$


HXX ≡ σ1x ⊗ σ2x ⊗ 13 + 11 ⊗ σ2x ⊗ σ3x + σ1x ⊗ 1 ⊗ σ3x
HZ ≡ Z11 + 1Z1 + 11Z

jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

In [52]:

```
H_max = 2
Nsteps=20
h = collect(0:H_max/(Nsteps-1):H_max)
```

Out[52]:

```
20-element Array{Float64,1}:
 0.0
 0.10526315789473684
 0.21052631578947367
 0.3157894736842105
 0.42105263157894735
 0.5263157894736842
 0.631578947368421
 0.7368421052631579
 0.8421052631578947
 0.9473684210526315
 1.0526315789473684
 1.1578947368421053
 1.263157894736842
 1.368421052631579
 1.4736842105263157
 1.5789473684210527
```

jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

```
8x8 Array{Float64,2}:
  0.0        1.38778e-16   6.66134e-16  …   0.0        0.866025
 -0.149429  -5.55112e-17   1.11022e-16     -0.557678   2.77556e-17
 -0.149429   5.55112e-17   1.11022e-16     -0.557678   2.77556e-17
  0.0       -0.707107     -0.408248         0.0        0.288675
 -0.149429  -1.2326e-32    1.11022e-16     -0.557678   1.11022e-16
  0.0        0.707107     -0.408248     …   0.0        0.288675
  0.0       -9.61481e-17   0.816497         0.0        0.288675
  0.965926   0.0           0.0             -0.258819   0.0
```

In [ ]:

In [52]:

```julia
H_max = 2
Nsteps=20
h = collect(0:H_max/(Nsteps-1):H_max)
```

Out[52]:

```
20-element Array{Float64,1}:
 0.0
 0.10526315789473684
 0.21052631578947367
 0.3157894736842105
```

localhost:8888/notebooks/Documents/2018juliacourse/Guifre/2018-computational-physics-course-ExactDiag1.ipynb

jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help | Not Trusted | Julia 1.0.0

```
H = HXX + h*HZ
```

Out[60]:

```
8×8 Array{Float64,2}:
 3.0  0.0  0.0   1.0  0.0   1.0   1.0   0.0
 0.0  1.0  1.0   0.0  1.0   0.0   0.0   1.0
 0.0  1.0  1.0   0.0  1.0   0.0   0.0   1.0
 1.0  0.0  0.0  -1.0  0.0   1.0   1.0   0.0
 0.0  1.0  1.0   0.0  1.0   0.0   0.0   1.0
 1.0  0.0  0.0   1.0  0.0  -1.0   1.0   0.0
 1.0  0.0  0.0   1.0  0.0   1.0  -1.0   0.0
 0.0  1.0  1.0   0.0  1.0   0.0   0.0  -3.0
```

In [ ]:

```
D,U = eigen(H)
```

In [52]:

```
H_max = 2
Nsteps=20
h = collect(0:H_max/(Nsteps-1):H_max)
```

Out[52]:

```
20-element Array{Float64,1}:
```

localhost:8888/notebooks/Documents/2018juliacourse/Guifre/2018-computational-physics-course-ExactDiag1.ipynb

jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help   Not Trusted   Julia 1.0.0

```
 -0.149429   5.55112e-17  1.11022e-16      -0.557678  2.77556e-17
  0.0       -0.707107    -0.408248          0.0       0.288675
 -0.149429  -1.2326e-32   1.11022e-16      -0.557678  1.11022e-16
  0.0        0.707107    -0.408248     …    0.0       0.288675
  0.0       -9.61481e-17  0.816497          0.0       0.288675
  0.965926   0.0          0.0              -0.258819  0.0
```

In [52]:
```
H_max = 2
Nsteps=20
h = collect(0:H_max/(Nsteps-1):H_max)
```

Out[52]:
```
20-element Array{Float64,1}:
 0.0
 0.10526315789473684
 0.21052631578947367
 0.3157894736842105
 0.42105263157894735
 0.5263157894736842
 0.631578947368421
 0.7368421052631579
 0.8421052631578947
```

jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

```
1.5789473684210527
1.6842105263157894
1.7894736842105263
1.894736842105263
2.0
```

In [64]:

```
E0 = zeros(Nsteps)
for i=1:Nsteps
    H = HXX + h[i]*HZ
    D,U = eigen(H)
    E0[i] = D[1]
end
```

In [54]:

```
using PyPlot
plot(h,E0, marker=".", color = "b")
grid("on")
title("Ground state energy: quantum Ising model for N=3 spins")
xlabel("magnetic field h")
ylabel("Energy");
```

jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help | Not Trusted | Julia 1.0.0

```
1.7894736842105263
1.894736842105263
2.0
```

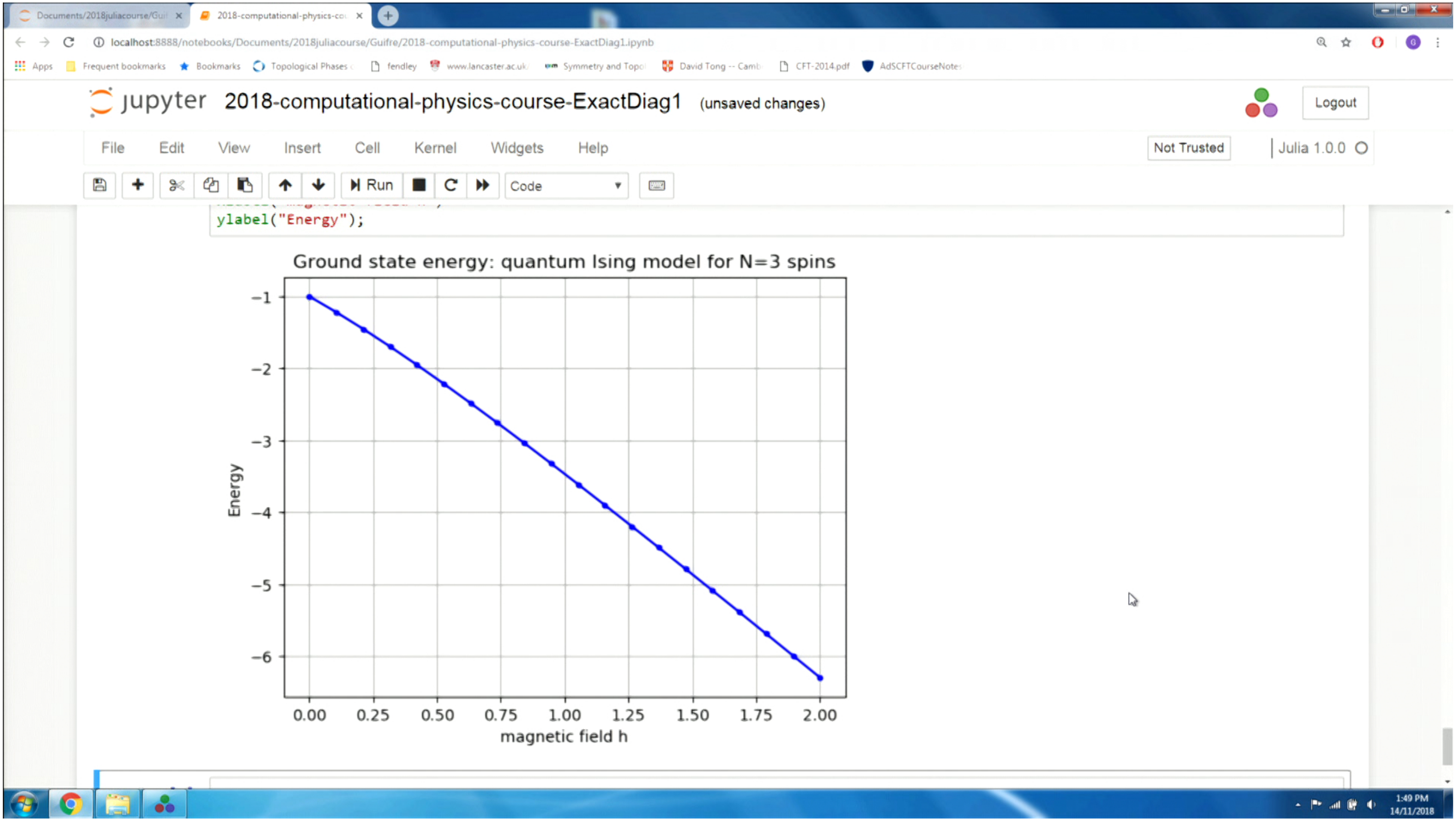
In [64]:

```
E0 = zeros(Nsteps)
for i=1:Nsteps
    H = HXX + h[i]*HZ
    D,U = eigen(H)
    E0[i] = D[1]
end
```

In [*]:

```
using PyPlot
plot(h,E0, marker=".", color = "b")
grid("on")
title("Ground state energy: quantum Ising model for N=3 spins")
xlabel("magnetic field h")
ylabel("Energy");
```

In [ ]:


localhost:8888/notebooks/Documents/2018juliacourse/Guifre/2018-computational-physics-course-ExactDiag1.ipynb
jupyter 2018-computational-physics-course-ExactDiag1 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Julia 1.0.0
Run
Code
ylabel("Energy");
Ground state energy: quantum Ising model for N=3 spins
Energy
magnetic field h
−1
−2
−3
−4
−5
−6
0.00 0.25 0.50 0.75 1.00 1.25 1.50 1.75 2.00