Title: PSI 2019/2020 - Computational Physics - Lecture 14

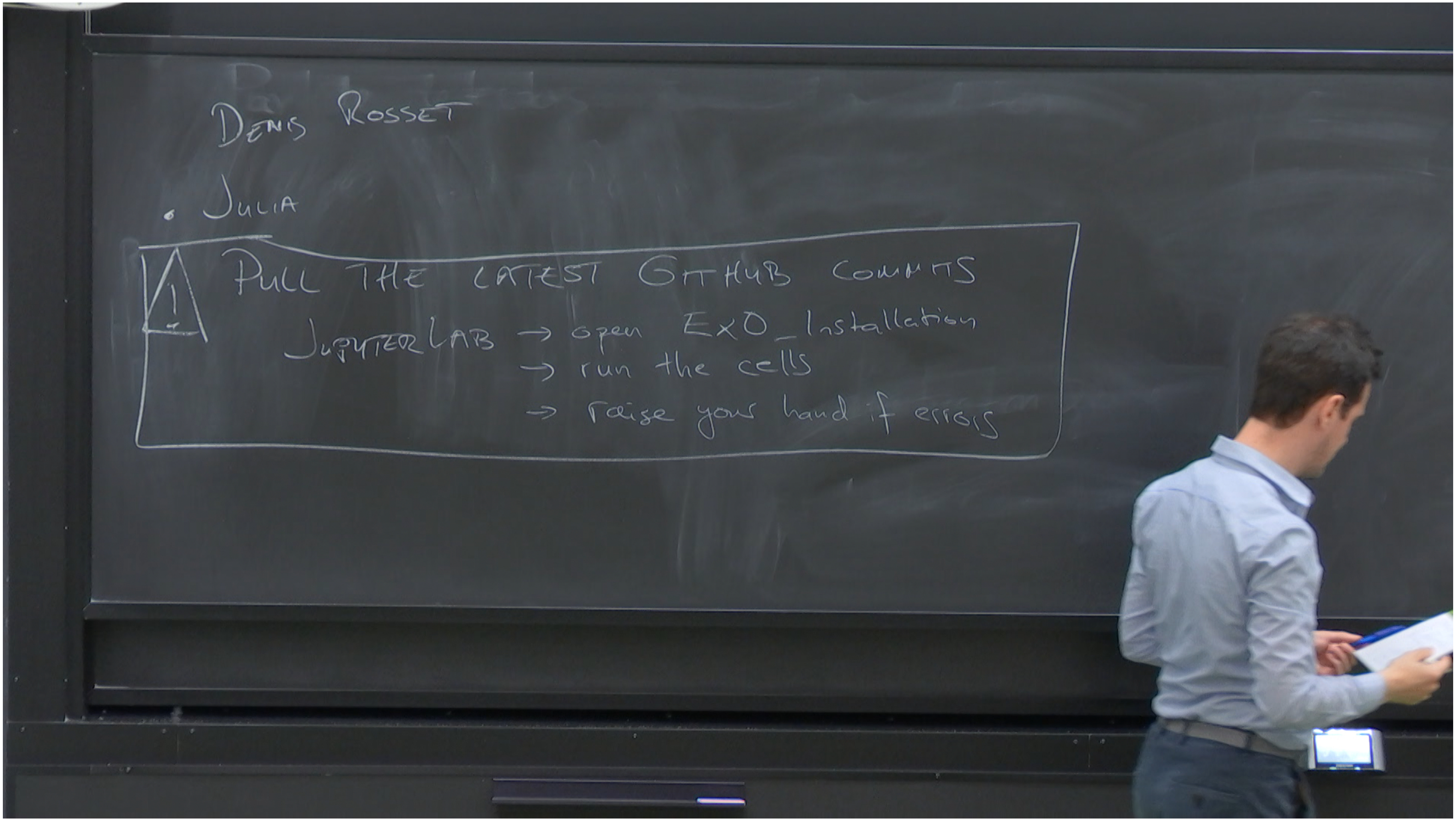
Speakers: Erik Schnetter

Collection: PSI 2019/2020 - Computational Physics

Date: March 09, 2020 - 1:30 PM

URL: http://pirsa.org/20030031

PERIMETER PI INSTITUTE FOR THEORETICAL PHYSICS

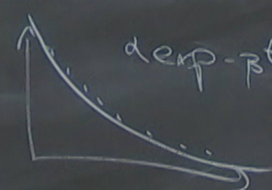

DENIS ROSSET
• JULIA
PULL THE LATEST GITHUB COMMITS
JUPYTERLAB → open ExO_Installation
→ run the cells
→ raise your hand if errors

PAST (Convex) optimization

- $\exists\, \vec{x} \in \mathbb{R}^n \quad F(\vec{x}) = 0$

  $F: \mathbb{R}^n \longrightarrow \mathbb{R}^m$

- Nonlinear fit



$\alpha \exp -\beta t$

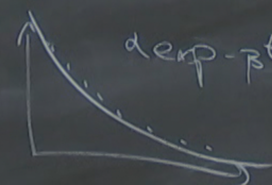
- SAT (REDUCE to normal form)

PAST (Convex) optimization

- $\exists \vec{x} \in \mathbb{R}^n \quad F(\vec{x}) = 0$

  $F: \mathbb{R}^n \to \mathbb{R}^m$

- Nonlinear fit



$a\exp(-\beta t)$

- SAT (REDUCE to normal form)

- MATLAB / MAPLE
  - fmincon
  - fminunc ← nonlinear optimization

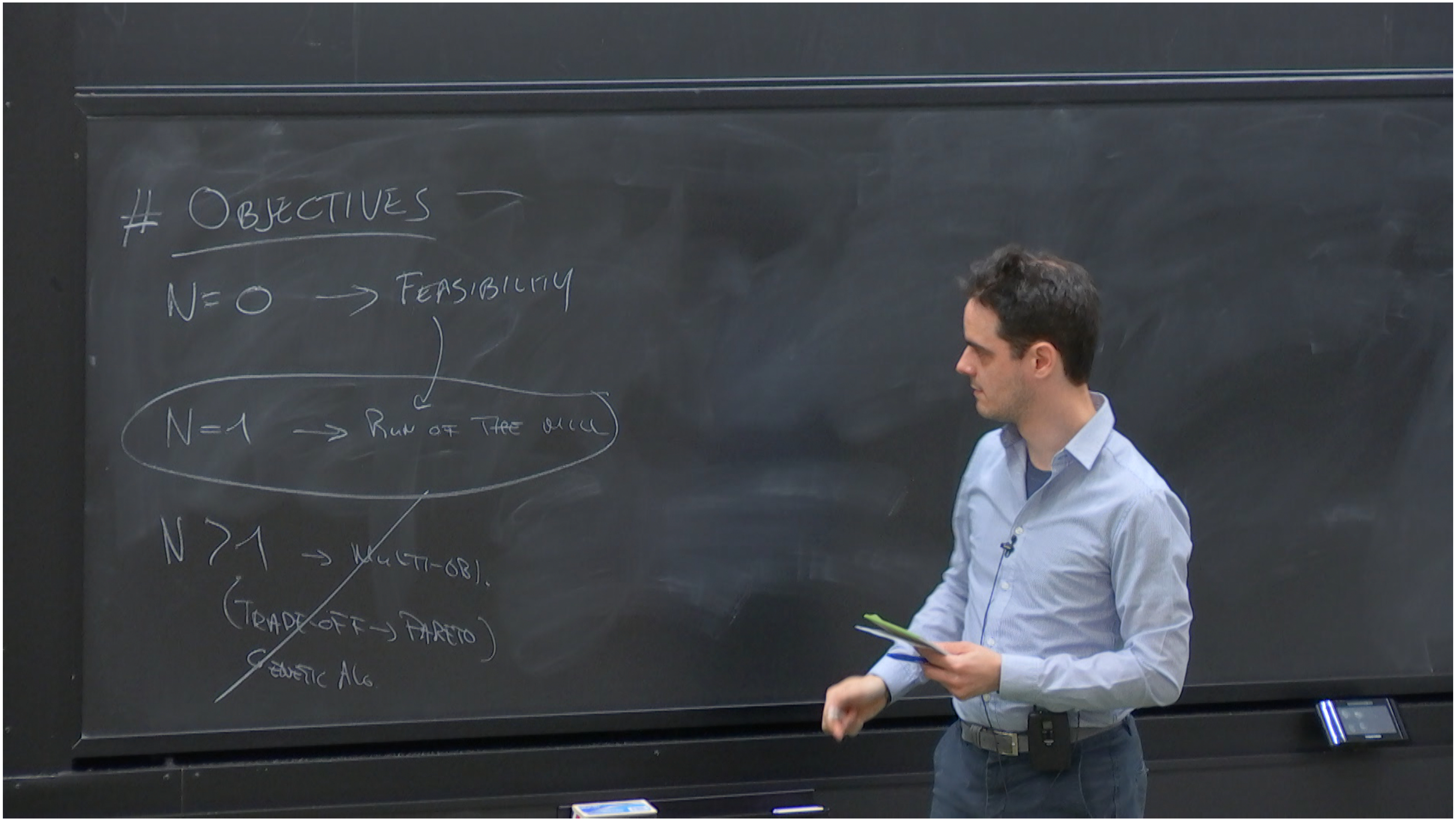
FUTURE

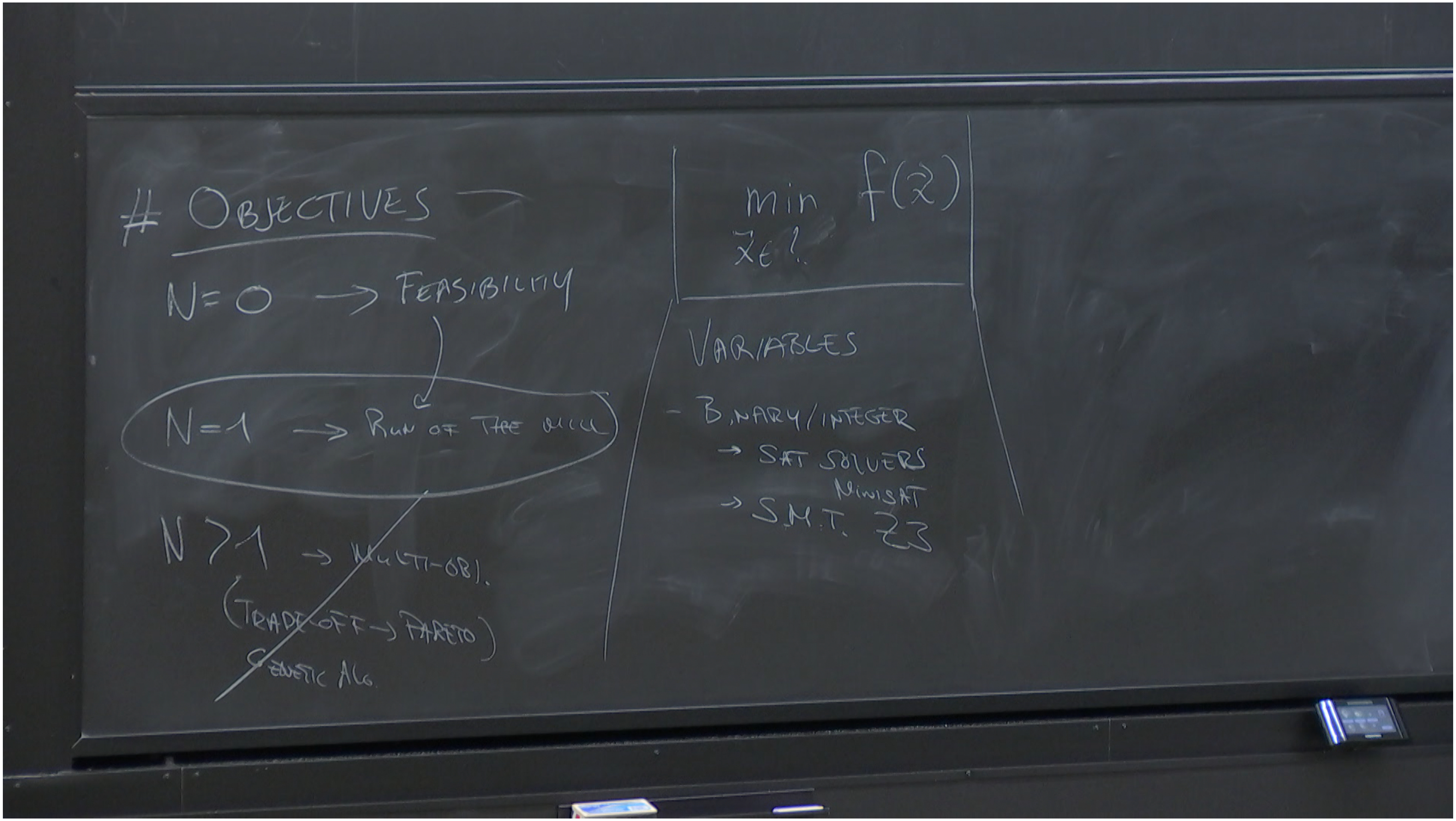
- SDP

LECTURE

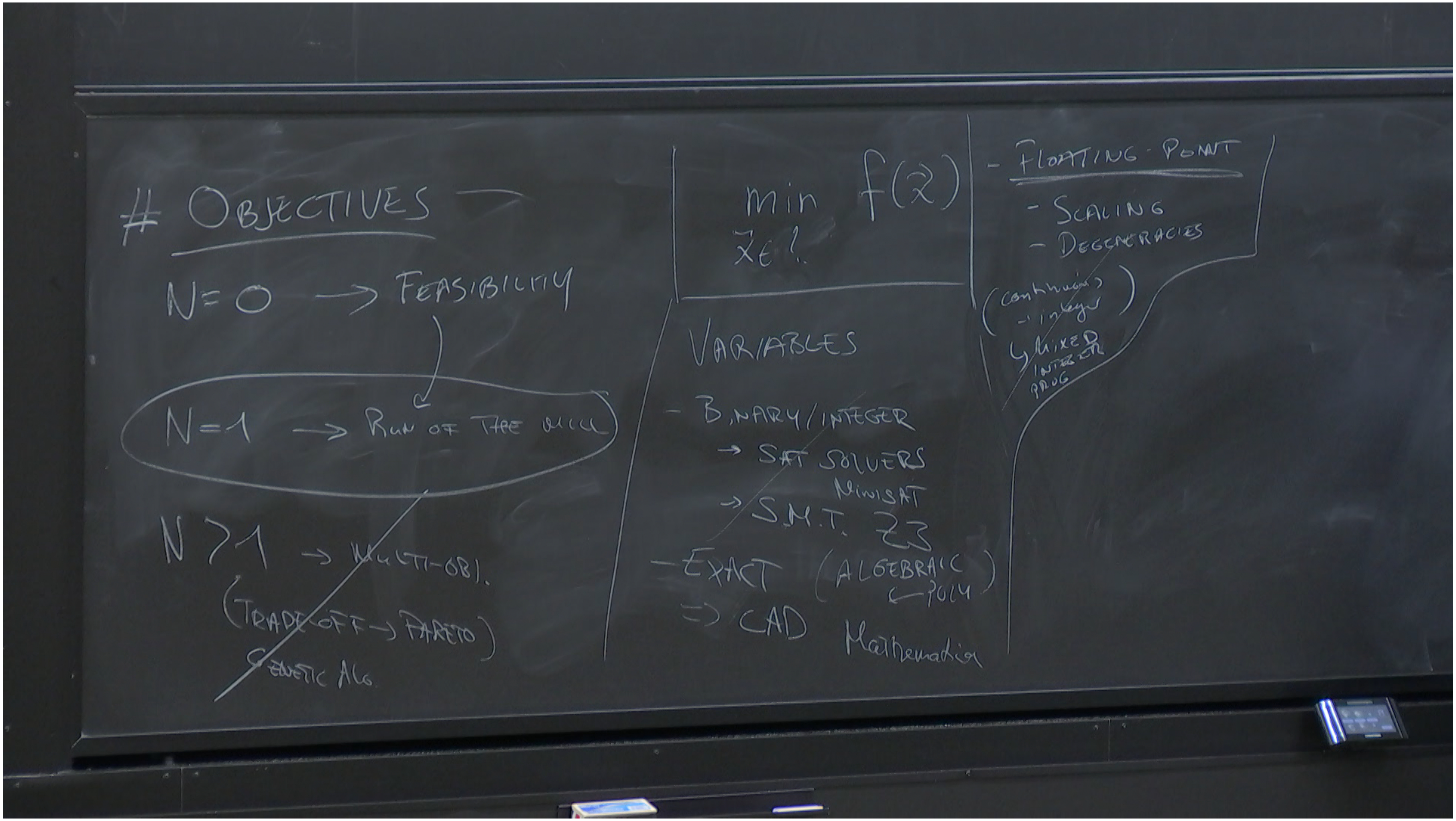
QUANTUM INFO

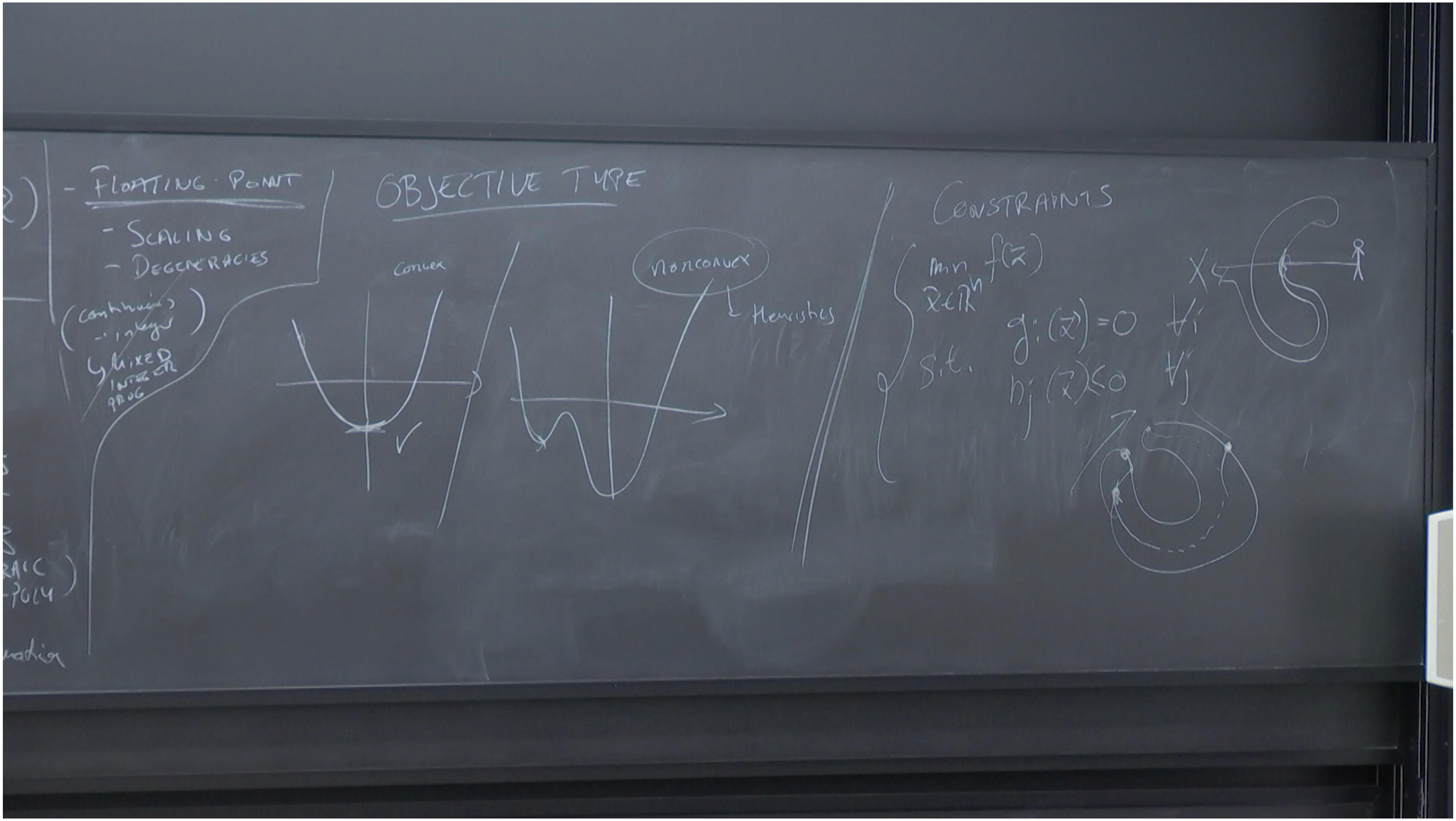
– EXAMPLES TOY PROBLEMS

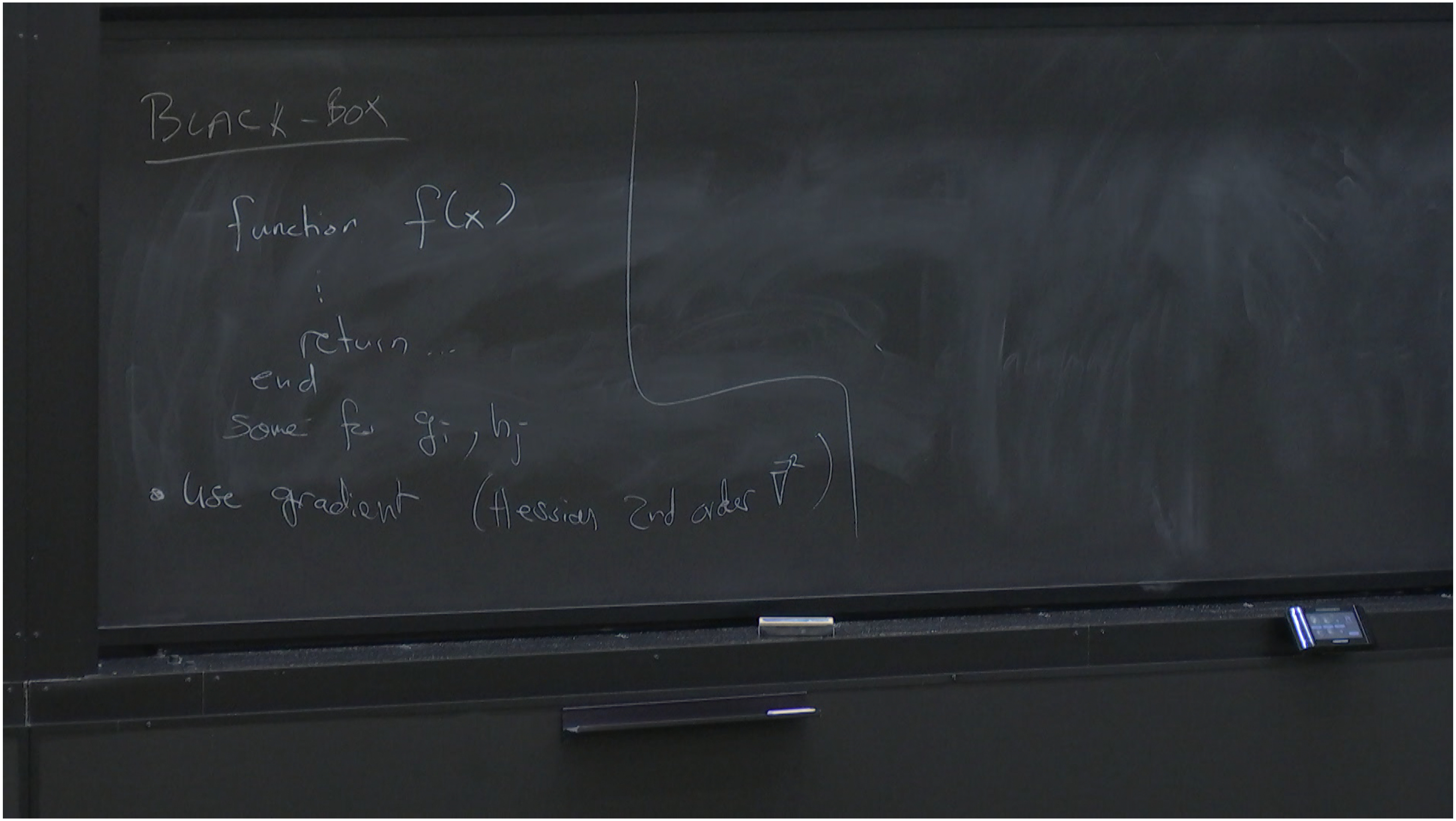

# OBJECTIVES
N=0 → FEASIBILITY
N=1 → RUN OF THE MILL
N>1 → MULTI-OBJ.
(TRADE-OFF → PARETO)
GENETIC ALG

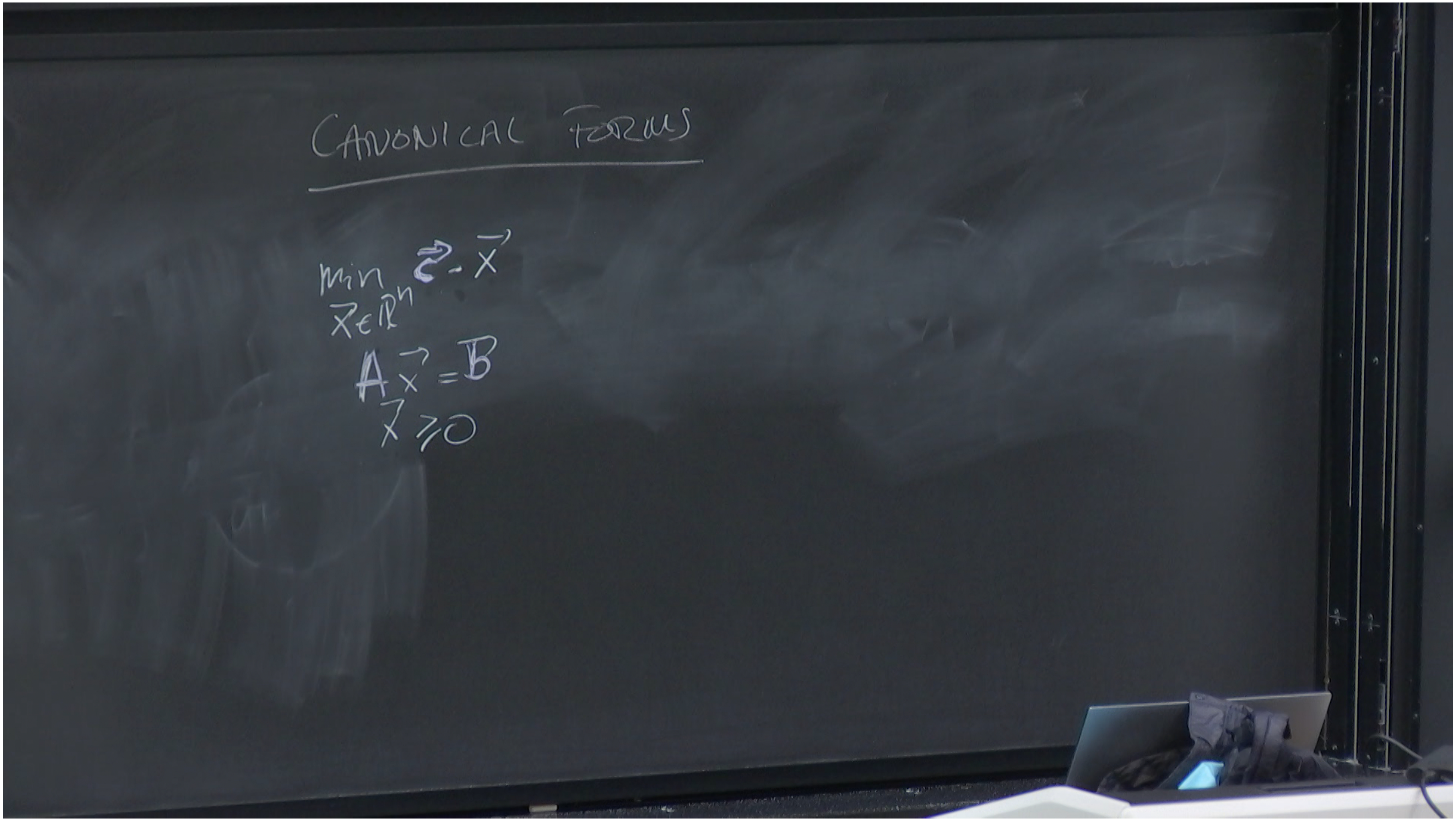


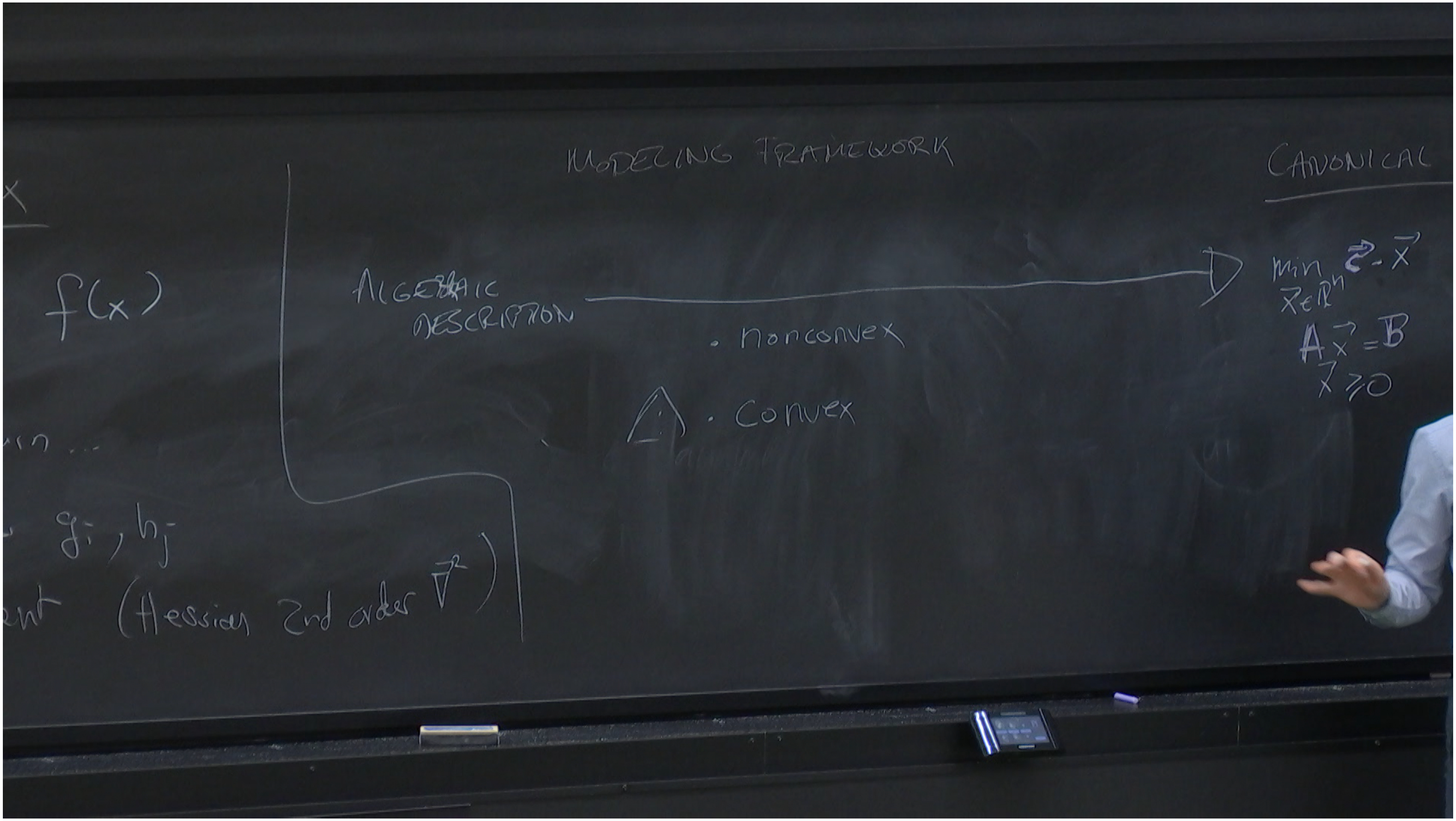
# OBJECTIVES →
N=0 → FEASIBILITY
N=1 → RUN OF THE MILL
N>1 → MULTI-OBJ.
(TRADE-OFF → PARETO)
GENETIC ALG.
min f(x)
x∈?
VARIABLES
- BINARY/INTEGER
→ SAT SOLVERS
MINISAT
→ S.M.T. Z3

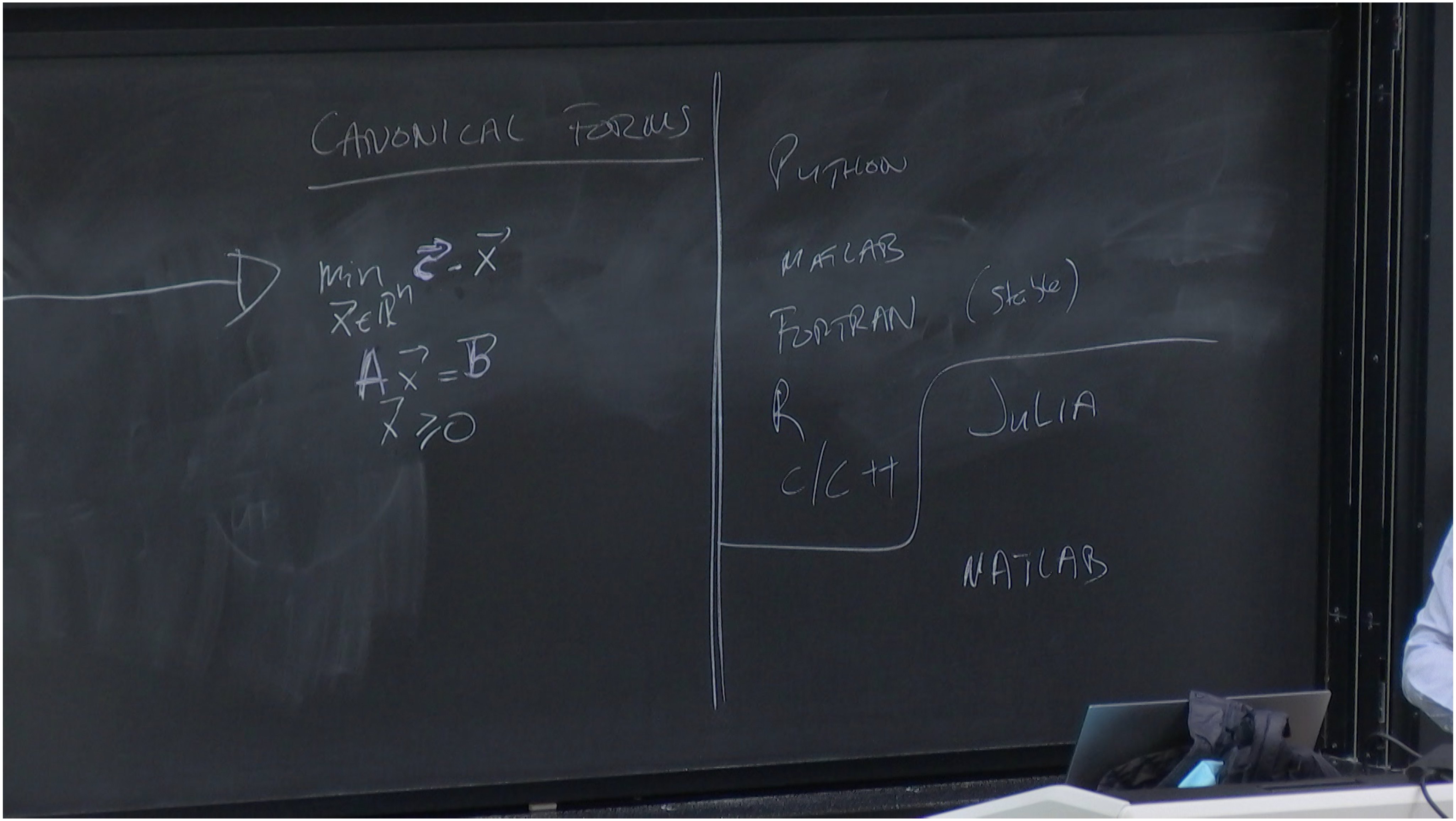

# OBJECTIVES
N=0 → FEASIBILITY
N=1 → RUN OF THE MILL
N>1 → MULTI-OBJ.
(TRADE-OFF → PARETO)
GENETIC ALG.
min f(x)
x ∈ ?
VARIABLES
- BINARY/INTEGER
→ SAT SOLVERS
MINISAT
→ S.M.T. Z3
- EXACT (ALGEBRAIC)
← POLY
⇒ CAD Mathematica
- FLOATING-POINT
- SCALING
- DEGENERACIES
(continuous + integer)
MIXED INTEGER PROG

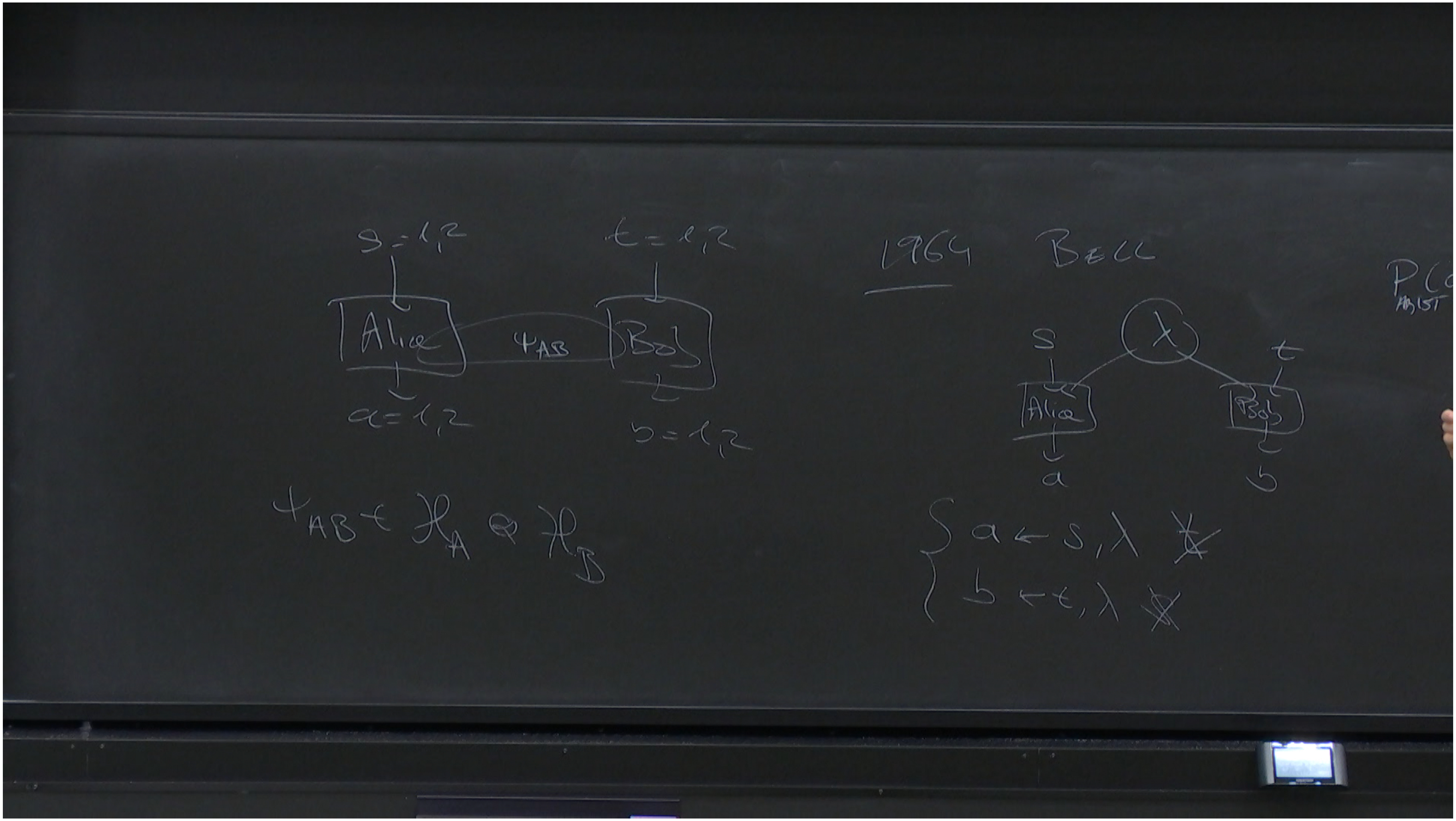

- FLOATING POINT
- SCALING
- DEGENERACIES
(continuous
- integer)
MIXED INTEGER PROG
OBJECTIVE TYPE
convex
nonconvex
Heuristics
CONSTRAINTS
min f(x)
x∈ℝ^n
s.t. g_i(x) = 0 ∀i
h_j(x) ≤ 0 ∀j

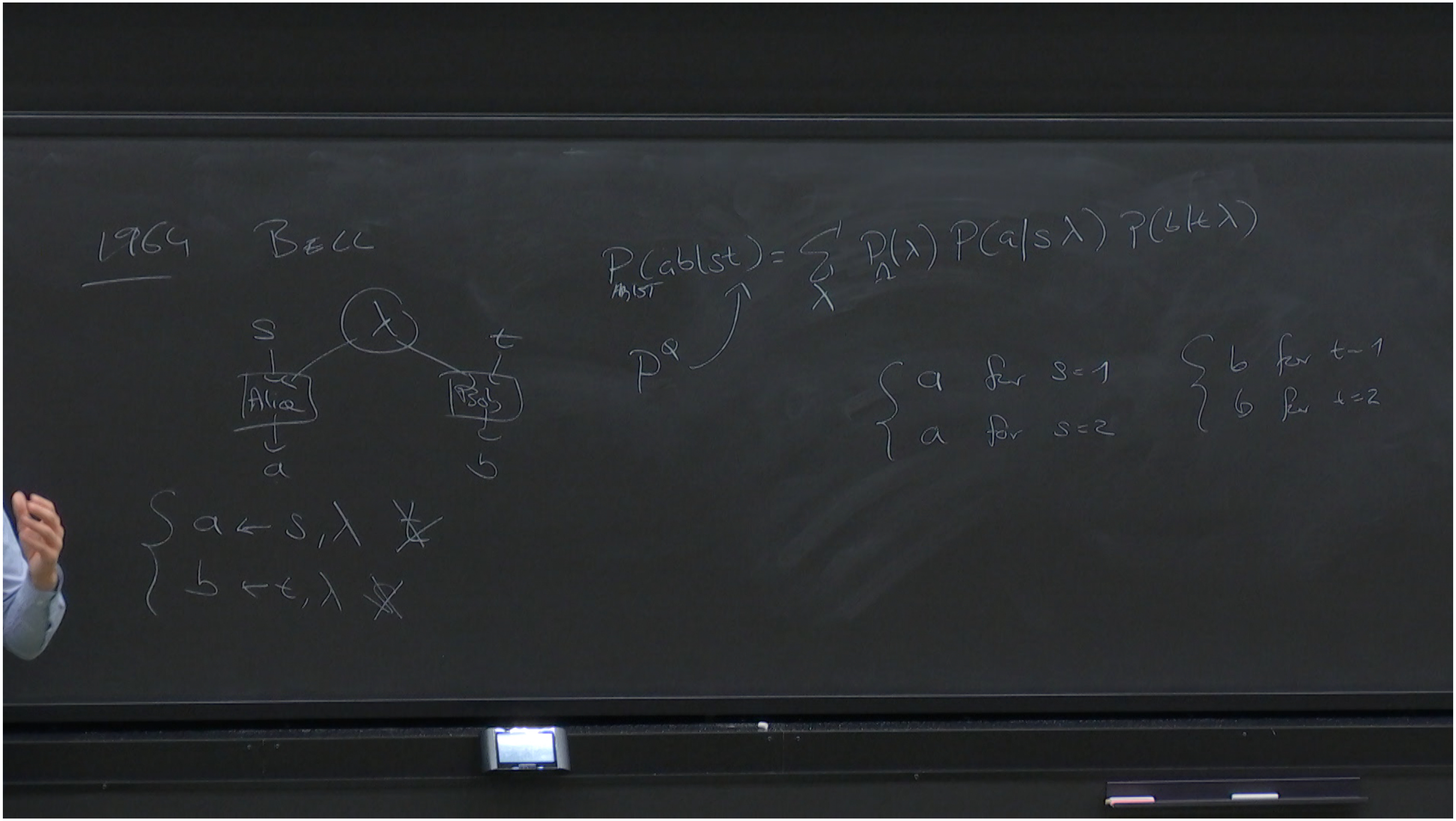

BLACK-BOX
function f(x)
return ...
end
Some fn g_i, h_j
• Use gradient (Hessian 2nd order ∇^2)


CANONICAL FORMS
$\min_{\vec{x} \in \mathbb{R}^n} \vec{c} \cdot \vec{x}$
$A\vec{x} = \vec{b}$
$\vec{x} \geq 0$


MODELING FRAMEWORK
CANONICAL
f(x)
ALGEBRAIC DESCRIPTION
• nonconvex
• convex
min $\vec{c} \cdot \vec{x}$
$\vec{x} \in \mathbb{R}^n$
$A\vec{x} = B$
$\vec{x} \geq 0$
g_i, h_j
(Hessian 2nd order $\vec{\nabla}^2$)


CANONICAL FORMS
min c · x
x ∈ R^n
Ax = B
x ≥ 0
PYTHON
MATLAB
FORTRAN
(stable)
R
C/C++
JULIA
MATLAB


s = 1,2
t = 1,2
Alice
ψAB
Bob
a = 1,2
b = 1,2
ψAB ∈ HA ⊗ HB
1964
BELL
λ
s
t
Alice
Bob
a
b
a ← s, λ
b ← t, λ

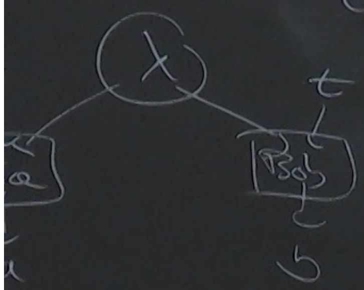

1964 BELL
s
λ
t
Alice
Bob
a
b
a ← s, λ
b ← t, λ
P(ab|st) = Σ_λ P(λ) P(a|s λ) P(b|t λ)
P^Q
a for s=1
a for s=2
b for t=1
b for t=2

BELL

$P(ab|st)$ is $\overset{\text{BELL}}{\text{LOCAL}}$ if $\exists\ P_\Lambda(\lambda),\ P(a|s\lambda),\ P(b|t\lambda)$

$$(*)\quad P_{AB|ST}(ab|st) = \sum_{\lambda} P_\Lambda(\lambda)\, P(a|s\lambda)\, P(b|t\lambda)$$

P^Q



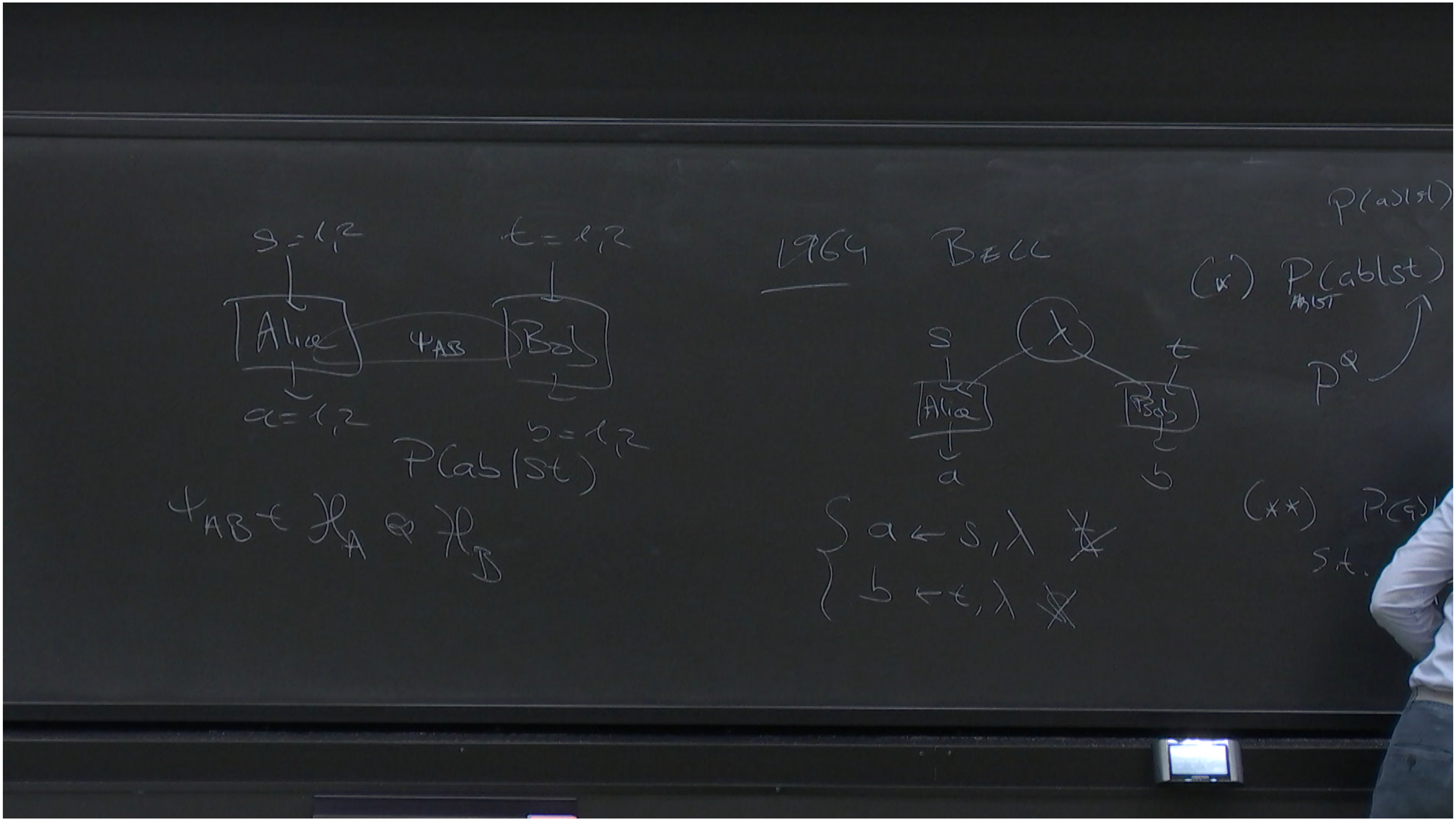
$$\begin{cases} a & \text{for } s=1 \\ a & \text{for } s=2 \end{cases} \qquad \begin{cases} b & \text{for } t=1 \\ b & \text{for } t=2 \end{cases}$$

s, λ

t, λ

$(**)$ $P(ab|st)$ is LOCAL if $\exists\ P_{A_1 A_2 B_1 B_2}(a_1, a_2, b_1, b_2)$

s.t. $$P(ab|st) = \sum_{a_1 a_2 b_1 b_2} P(a_1, a_2, b_1, b_2)\, \delta[a = a_s]\, \delta(b = b_t)$$

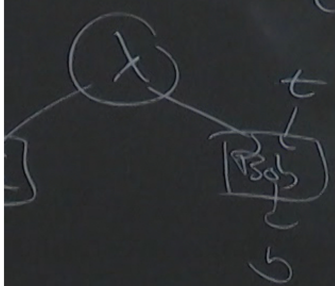

s = 1,2
t = 1,2
Alice
ψAB
Bob
a = 1,2
b = 1,2
P(ab|st)
ψAB ∈ HA ⊗ HB
1964 Bell
λ
s
t
Alice
Bob
a
b
a ← s, λ
b ← t, λ
P(ab|st)
(*) PAB|ST(ab|st)
P^Q
(**)
s.t.

ELL

$P(ab|st)$ is LOCAL (BELL) if $\exists\ P_\Lambda(\lambda),\ P(a|s\lambda),\ P(b|t\lambda)$

$$(*)\quad P_{AB|ST}(ab|st) = \sum_\lambda P_\Lambda(\lambda)\, P(a|s\lambda)\, P(b|t\lambda)$$

P^Q



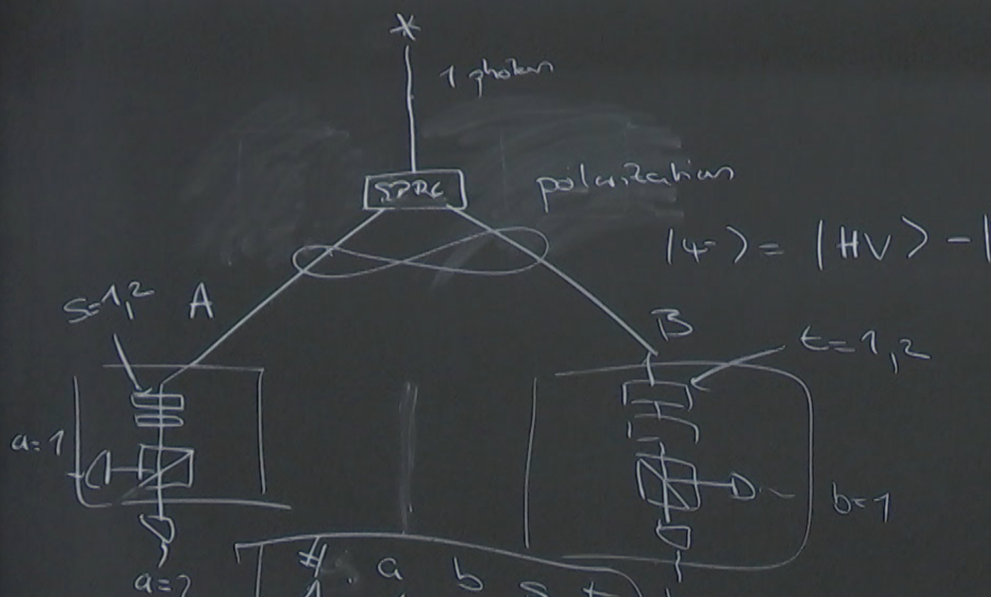
$$\begin{cases} a & \text{for } s=1 \\ a & \text{for } s=2 \end{cases} \qquad \begin{cases} b & \text{for } t=1 \\ b & \text{for } t=2 \end{cases}$$

(**) $P(ab|st)$ is LOCAL if $\exists\ P_{A_1A_2B_1B_2}(a_1,a_2,b_1,b_2)$

s.t. $$P(ab|st) = \sum_{a_1a_2b_1b_2} P(a_1a_2b_1b_2)\,\delta[a=a_s]\,\delta(b=b_t)$$

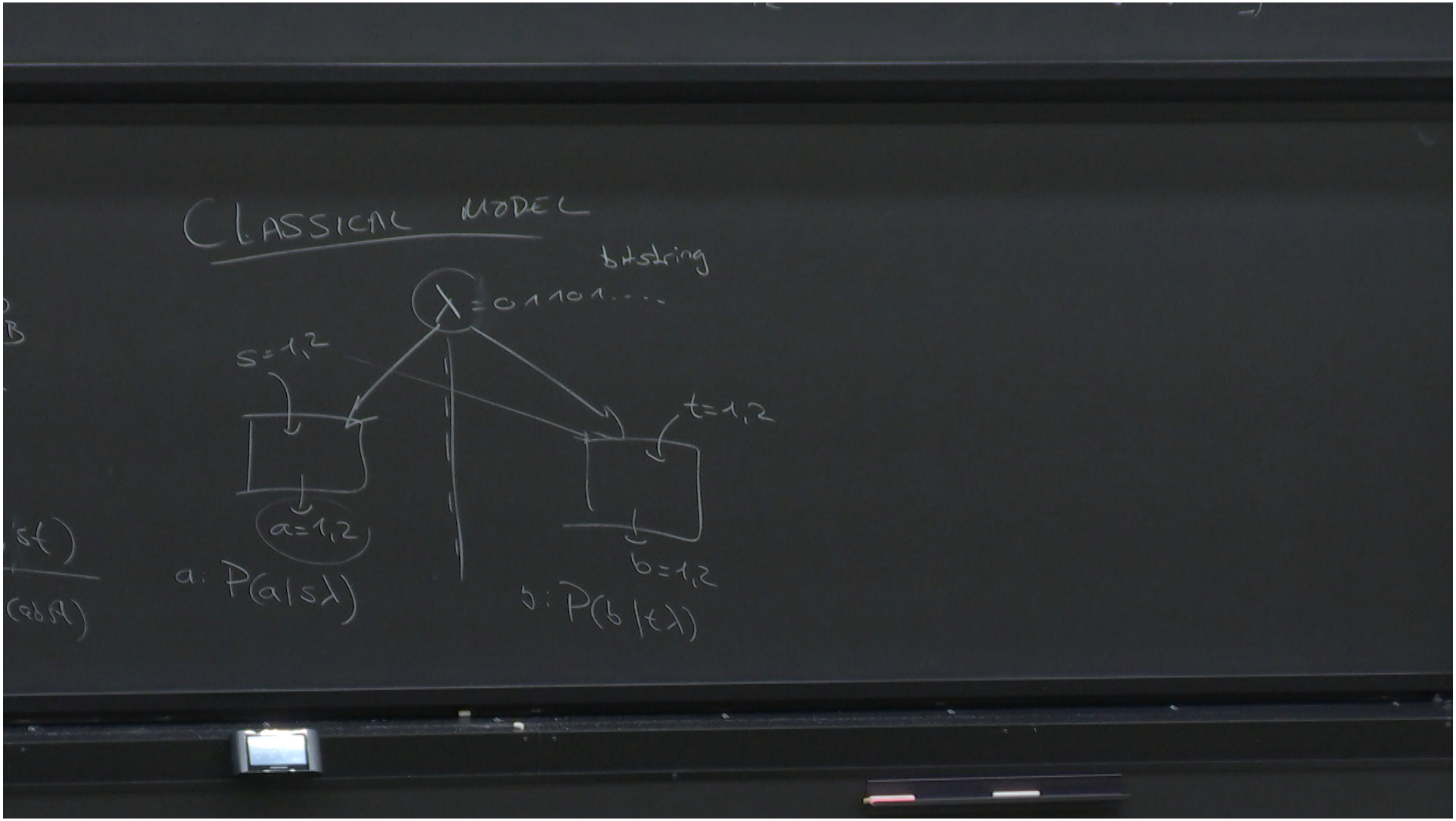
s, λ

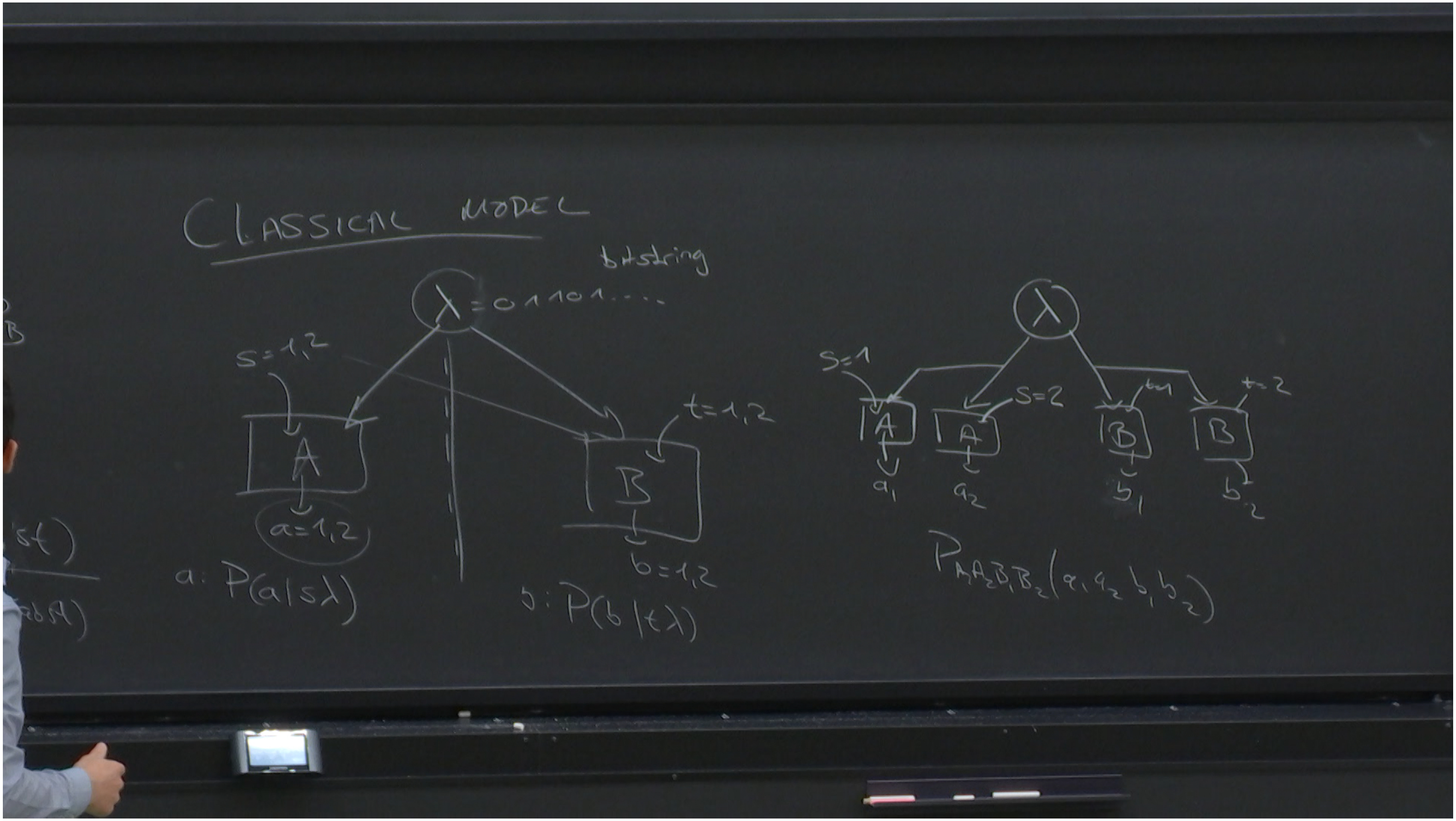
$, \lambda$

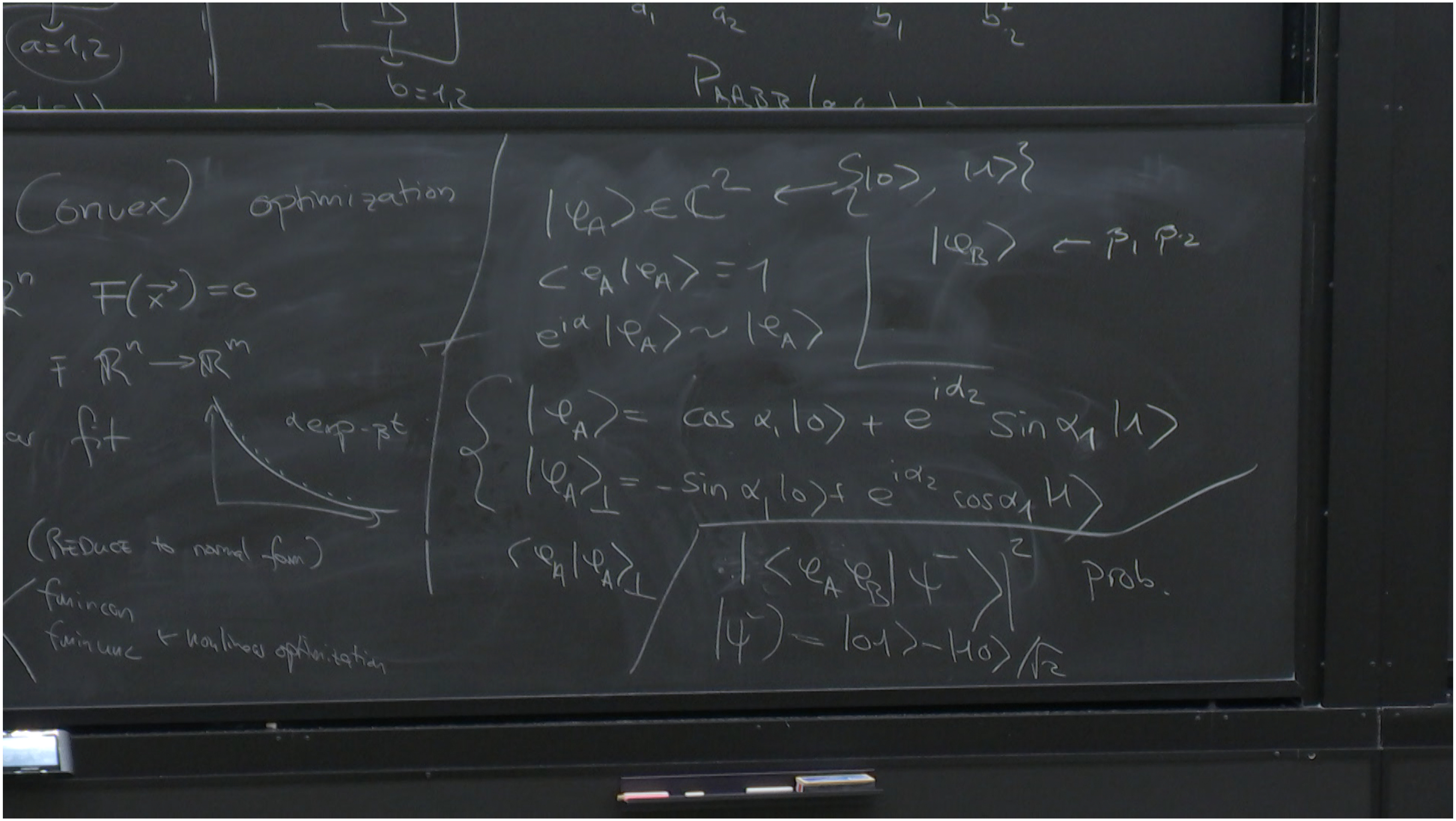


#	a	b	s	t
1	1	1	1	1
2	1	2	1	1
3	2	1	2	1
4	1	1	1	2

$$P(ab|st) = \frac{N(ab|st)}{\sum_{ab} N(ab|st)}$$


CLASSICAL MODEL
bitstring
λ = 0 1 1 0 1 ....
s = 1,2
t = 1,2
a = 1,2
b = 1,2
a: P(a|sλ)
b: P(b|tλ)


CLASSICAL MODEL
bitstring
λ = 01101....
s = 1,2
A
a = 1,2
a: P(a|sλ)
t = 1,2
B
b = 1,2
b: P(b|tλ)
λ
s=1
s=2
t=1
t=2
A
A
B
B
a1
a2
b1
b2
P A1A2B1B2 (a1 a2 b1 b2)


a=1,2
b=1,2
(Convex) optimization
F(x)=0
F: R^n → R^m
fit
exp -βt
(REDUCE to normal form)
fmincon
fminunc ← nonlinear optimization
|φ_A⟩ ∈ C^2 ← {|0⟩, |1⟩}
⟨φ_A|φ_A⟩ = 1
e^{iα}|φ_A⟩ ~ |φ_A⟩
|φ_B⟩ ← β_1 β_2
|φ_A⟩ = cos α_1|0⟩ + e^{iα_2} sin α_1|1⟩
|φ_A⟩_⊥ = −sin α_1|0⟩ + e^{iα_2} cos α_1|1⟩
⟨φ_A|φ_A⟩_⊥
|⟨φ_A φ_B|ψ^−⟩|^2 prob.
|ψ^−⟩ = |01⟩ − |10⟩ / √2

File Edit View Run Kernel Git Tabs Settings Help

/ ··· / convexcourse / Module1 /

Name	Last Modified
Ex0_Installation.ipynb	3 hours ago
Ex1a_Nonlinear.ipynb	3 hours ago
Ex1b_Nonlinear_JuMP.ipynb	3 hours ago
Ex2_Visibility.ipynb	3 hours ago
octave-workspace	3 hours ago

Ex0_Installation.ipynb × Ex1_Nonlinear.ipynb × Ex2_Visibility.ipynb × Ex1a_Nonlinear.ipynb ×

Markdown git Julia 1.3.1

Packages:

- NLopt Powell's derivative-free algorithms
- Optim Gradient-based algorithms, autodifferentiation support

[1]:
```
using NLopt
using Optim
using Printf
```

Nonlinear unconstrained optimization

Example: projective measurements on the singlet state $|\Psi^-\rangle = |01 - 10\rangle/\sqrt{2} \in \mathcal{H}_\mathrm{A} \otimes (\mathcal{H}_\mathrm{B})$.

A for Alice and B for Bob.

We parameterize local measurements by qubit states.

$$\left|\vec{\phi}\right\rangle = \begin{pmatrix} \cos\phi_1 \\ e^{i\phi_2}\sin\phi_1 \end{pmatrix}, \qquad \left|\vec{\phi}\right\rangle_\perp = \begin{pmatrix} -\sin\phi_1 \\ e^{i\phi_2}\cos\phi_1 \end{pmatrix}$$

There are two measurement settings indexed by $s = 1, 2$ on $\mathcal{H}_\mathrm{A}$, corresponding to two projectors $\{|\alpha_1\rangle, |\alpha_1\rangle_\perp\}$ and $\{|\alpha_2\rangle, |\alpha_2\rangle_\perp\}$, and two measurement settings indexed by $t = 1, 2$ on $\mathcal{H}_\mathrm{B}$, corresponding to $\{|\beta_1\rangle, |\beta_1\rangle_\perp\}$ and $\{|\beta_2\rangle, |\beta_2\rangle_\perp\}$.

The measurement outcomes are 1-based (Julia, Matlab), so that $a = 1$ corresponds to $|\alpha_s\rangle$ and $a = 2$ to $|\alpha_s\rangle_\perp$. The same for Bob: for $b = 1$ corresponds to $|\beta_t\rangle$ and $b = 2$ to $|\beta_t\rangle_\perp$

Then we consider the joint, conditional probability distribution

$$P_\mathrm{AB|ST}(a, b|s, t)$$

. We compute using Born's rule:

$$\begin{aligned}
P_\mathrm{AB|ST}(1, 1|s, t) &= (\langle\alpha_s| \otimes \langle\beta_t|)\left|\Psi^-\right\rangle \\
P_\mathrm{AB|ST}(2, 1|s, t) &= \left(\langle\alpha_s|_\perp \otimes \langle\beta_t|\right)\left|\Psi^-\right\rangle \\
P_\mathrm{AB|ST}(1, 2|s, t) &= \left(\langle\alpha_s| \otimes \langle\beta_t|_\perp\right)\left|\Psi^-\right\rangle \\
P_\mathrm{AB|ST}(2, 2|s, t) &= \left(\langle\alpha_s|_\perp \otimes \langle\beta_t|_\perp\right)\left|\Psi^-\right\rangle
\end{aligned}$$

We want to maximize the CHSH expression:

$$C = \sum_{abst} (-1)^{(a-1)+(b-1)+(s-1)(t-1)} P_\mathrm{AB|ST}(a, b|s, t)$$

[10]:
```
function singlet_proj_prob(x)
# Computes the joint probability distribution given by two projective measurements on each subsystem of a singlet state
    P = zeros(eltype(x), (2, 2, 2, 2));
    A = zeros(Complex{eltype(x)}, (2, 2, 2));
    B = zeros(Complex{eltype(x)}, (2, 2, 2));
    A[:,1,1] = [ cos(x[1])
                 sin(x[1])];
    A[:,2,1] = [-sin(x[1])
```

0 5 Julia 1.3.1 | Idle Mode: Command Ln 1, Col 1 Ex1a_Nonlinear.ipynb

File Edit View Run Kernel Git Tabs Settings Help

Ex0_Installation.ipynb | Ex1_Nonlinear.ipynb | Ex2_Visibility.ipynb | Ex1a_Nonlinear.ipynb

/ ··· / convexcourse / Module1 /

Name

- Ex0_Installation.ipynb
- Ex1a_Nonlinear.ipynb
- Ex1b_Nonlinear_Ju...
- Ex2_Visibility.ipynb
- octave-workspace

Code git Julia 1.3.1

abst

[10]:

```julia
function singlet_proj_prob(x)
# Computes the joint probability distribution given by two projective measurements on each subsystem of a singlet state
    P = zeros(eltype(x), (2, 2, 2, 2));
    A = zeros(Complex{eltype(x)}, (2, 2, 2));
    B = zeros(Complex{eltype(x)}, (2, 2, 2));
    A[:,1,1] = [ cos(x[1])
                 sin(x[1])];
    A[:,2,1] = [-sin(x[1])
                 cos(x[1])];
    A[:,1,2] = [ cos(x[2])
                 sin(x[2])];
    A[:,2,2] = [-sin(x[2])
                 cos(x[2])];
    B[:,1,1] = [ cos(x[3])
                 sin(x[3])];
    B[:,2,1] = [-sin(x[3])
                 cos(x[3])];
    B[:,1,2] = [ cos(x[4])
                 sin(x[4])];
    B[:,2,2] = [-sin(x[4])
                 cos(x[4])];
    for s = 1:2
        for t = 1:2
            for a = 1:2
                for b = 1:2
                    ov = kron(A[:,a,s], B[:,b,t])' * [0; 1; -1; 0]/sqrt(2);
                    P[a,b,s,t] = real(conj(ov)*ov);
                end
            end
        end
    end
    return P
end
```

0 5 Julia 1.3.1 | Idle Mode: Edit Ln 19, Col 29 Ex1a_Nonlinear.ipynb

File Edit View Run Kernel Git Tabs Settings Help

Ex0_Installation.ipynb | Ex1_Nonlinear.ipynb | Ex2_Visibility.ipynb | Ex1a_Nonlinear.ipynb

/ ··· / convexcourse / Module1 /

Name

- Ex0_Installation.ipynb
- Ex1a_Nonlinear.ipynb
- Ex1b_Nonlinear_Ju...
- Ex2_Visibility.ipynb
- octave-workspace

Markdown | git | Julia 1.3.1

$$\left|\vec{\phi}\right\rangle = \begin{pmatrix} \cos\phi_1 \\ e^{i\phi_2}\sin\phi_1 \end{pmatrix}, \qquad \left|\vec{\phi}\right\rangle_\perp = \begin{pmatrix} -\sin\phi_1 \\ e^{i\phi_2}\cos\phi_1 \end{pmatrix}$$

There are two measurement settings indexed by $s = 1, 2$ on $\mathcal{H}_A$, corresponding to two projectors $\{|\alpha_1\rangle, |\alpha_1\rangle_\perp\}$ and $\{|\alpha_2\rangle, |\alpha_2\rangle_\perp\}$, and two measurement settings indexed by $t = 1, 2$ on $\mathcal{H}_B$, corresponding to $\{|\beta_1\rangle, |\beta_1\rangle_\perp\}$ and $\{|\beta_2\rangle, |\beta_2\rangle_\perp\}$.

The measurement outcomes are 1-based (Julia, Matlab), so that $a = 1$ corresponds to $|\alpha_s\rangle$ and $a = 2$ to $|\alpha_s\rangle_\perp$. The same for Bob: for $b = 1$ corresponds to $|\beta_t\rangle$ and $b = 2$ to $|\beta_t\rangle_\perp$

Then we consider the joint, conditional probability distribution

$$P_{AB|ST}(a, b|s, t)$$

. We compute using Born's rule:

$$\begin{aligned}
P_{AB|ST}(1, 1|s, t) &= (\langle\alpha_s| \otimes \langle\beta_t|)\,|\Psi^-\rangle \\
P_{AB|ST}(2, 1|s, t) &= \left(\langle\alpha_s|_\perp \otimes \langle\beta_t|\right)|\Psi^-\rangle \\
P_{AB|ST}(1, 2|s, t) &= \left(\langle\alpha_s| \otimes \langle\beta_t|_\perp\right)|\Psi^-\rangle \\
P_{AB|ST}(2, 2|s, t) &= \left(\langle\alpha_s|_\perp \otimes \langle\beta_t|_\perp\right)|\Psi^-\rangle
\end{aligned}$$

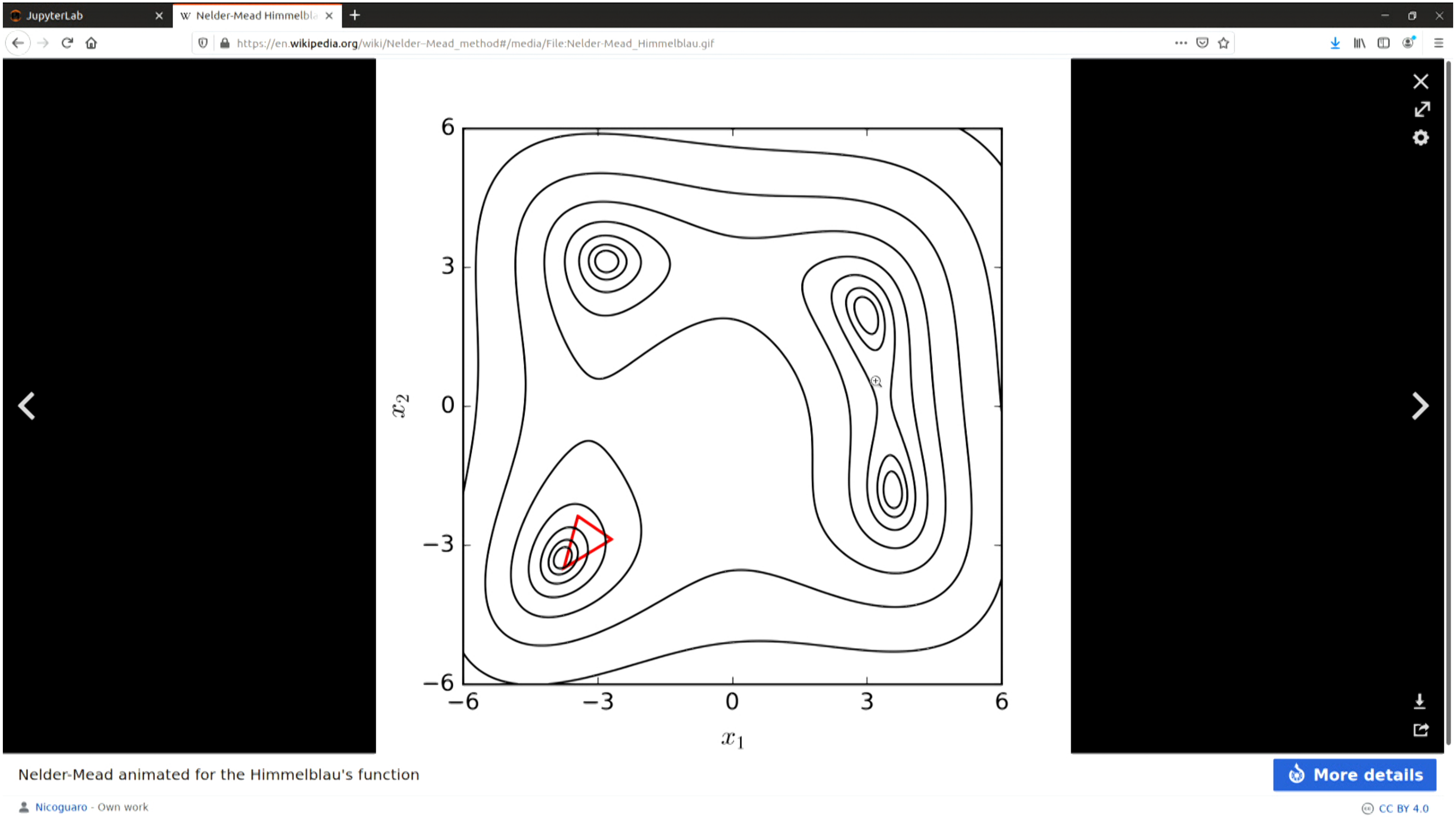
We want to maximize the CHSH expression:

$$C = \sum_{abst} (-1)^{(a-1)+(b-1)+(s-1)(t-1)} P_{AB|ST}(a, b|s, t)$$

[10]:

```julia
function singlet_proj_prob(x)
# Computes the joint probability distribution given by two projective measurements on each subsystem of a singlet state
    P = zeros(eltype(x), (2, 2, 2, 2));
    A = zeros(Complex{eltype(x)}, (2, 2, 2));
    B = zeros(Complex{eltype(x)}, (2, 2, 2));
    A[:,1,1] = [ cos(x[1])
                 sin(x[1])];
```

0 | 5 | Julia 1.3.1 | Idle — Mode: Command — Ln 1, Col 1 — Ex1a_Nonlinear.ipynb


JupyterLab
Nelder-Mead Himmelbla
https://en.wikipedia.org/wiki/Nelder-Mead_method#/media/File:Nelder-Mead_Himmelblau.gif
6
3
0
−3
−6
x_2
−6
−3
0
3
6
x_1
Nelder-Mead animated for the Himmelblau's function
More details
Nicoguaro - Own work
CC BY 4.0