Title: LECTURE: Generative Modelling

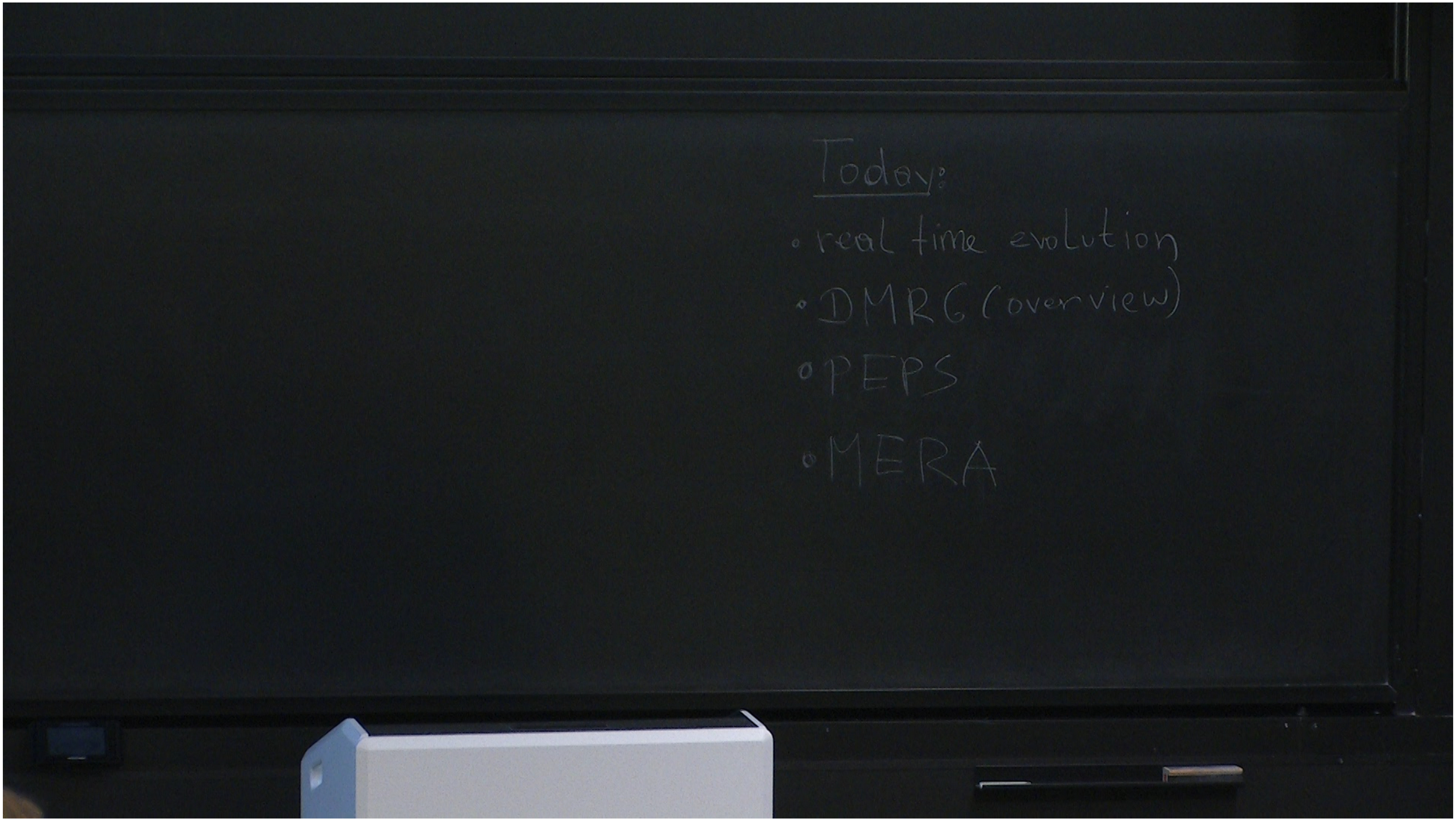
Speakers: Mohamed Hibat Allah

Collection: Quantum and AI Career Trajectories Mini-Course: Computational Methods and their Applications

Date: May 11, 2023 - 11:00 AM

URL: https://pirsa.org/23050096

Abstract: TBC


Today:
• real time evolution
• DMRG (overview)
• PEPS
• MERA

Lecture 4: Generative Models

* Last lectures (Quantum state reconstruction).
  * Restricted Boltzmann Machines (RBM) $\rightsquigarrow$ Approximate likelihood Model.
  * Neural Autoregressive Density Estimator (NADE) $\rightsquigarrow$ Exact likelihood Model.

$p(\vec{\sigma})$

ve

→ Approximate likelihood Model.

) → Exact likelihood Model.

$p(\vec{\sigma})$

* This lecture

Using "Recurrent Neural Networks" (RNN) to study Quantum many-body systems.

$\underbrace{\text{Recurrent Neural Networks}}_{\text{Autoregressive model.}}$

* Outline

1. RNNs
2. Variational Monte Carlo with RNNs
3. Gated Recurrent Unit (GRU).

① RNNs:

↳ Originally built for language processing.

Application: Machine translation, speech recognition, DNA sequence analysis.

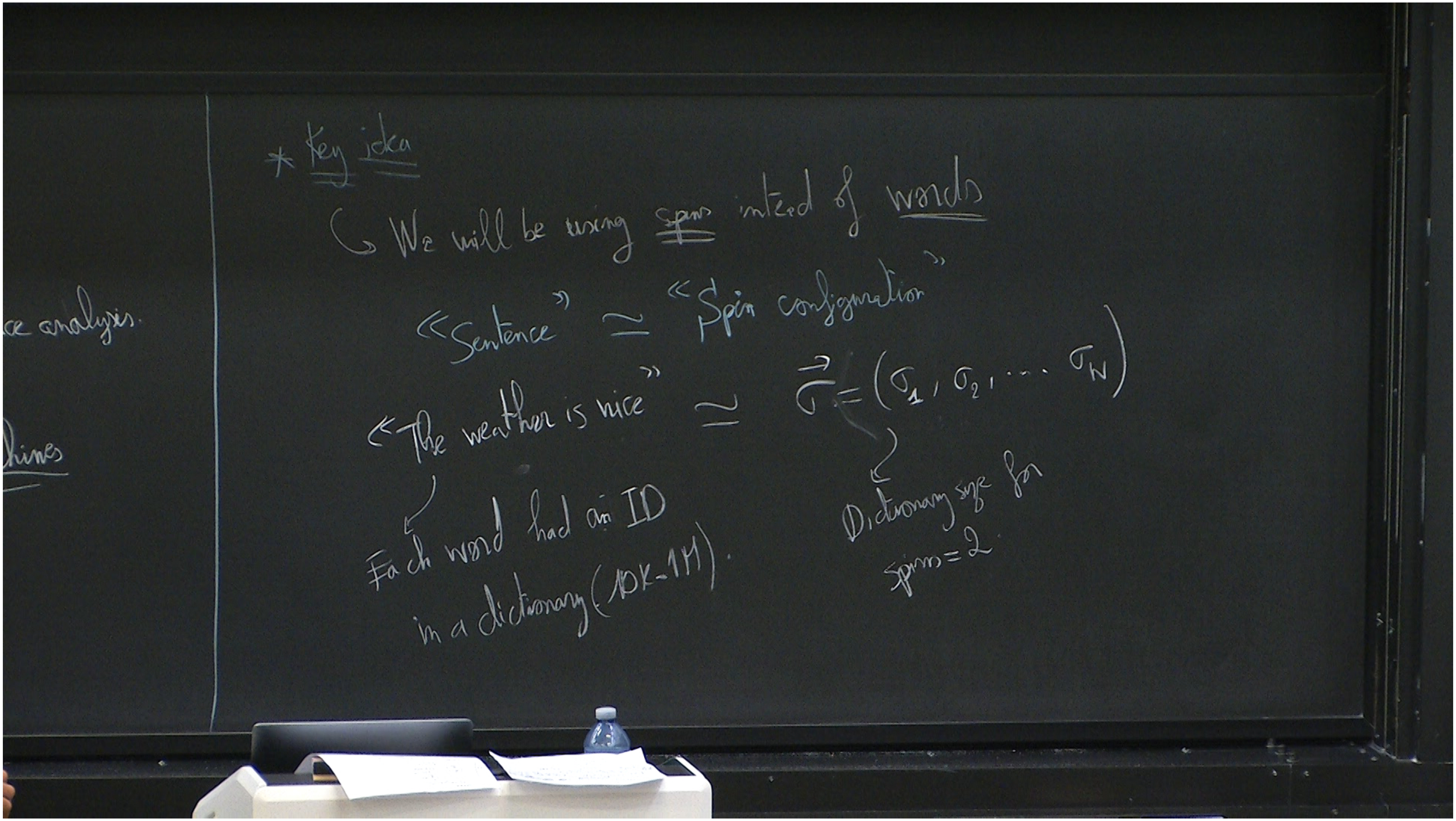
Music generation.

(1) RNNs:

↳ Originally built for language processing.

Application: Machine translation, speech recognition, DNA sequence analysis.
Music generation (AIVA)

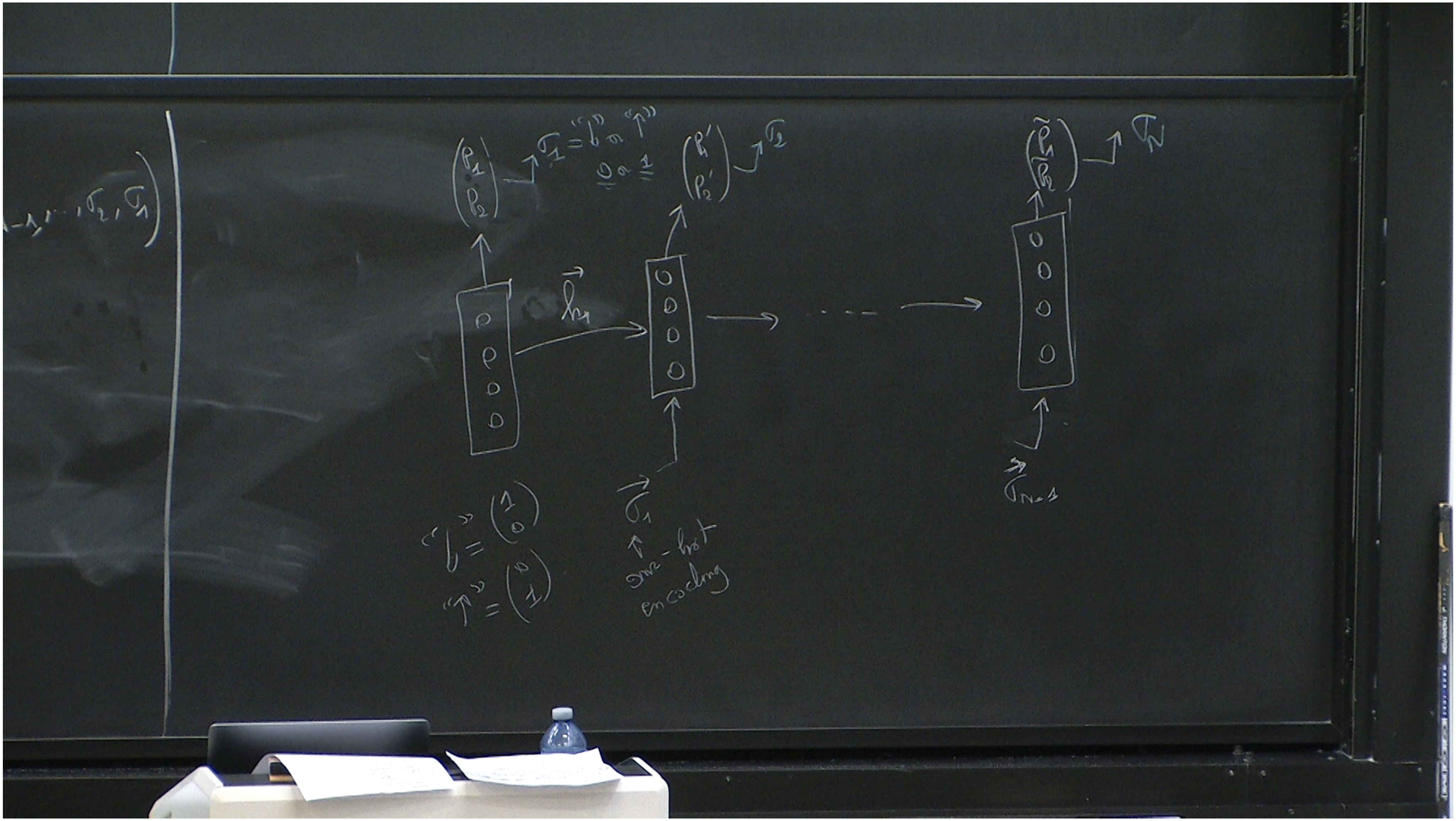
* Universal approximators. * Can simulate Turing Machines

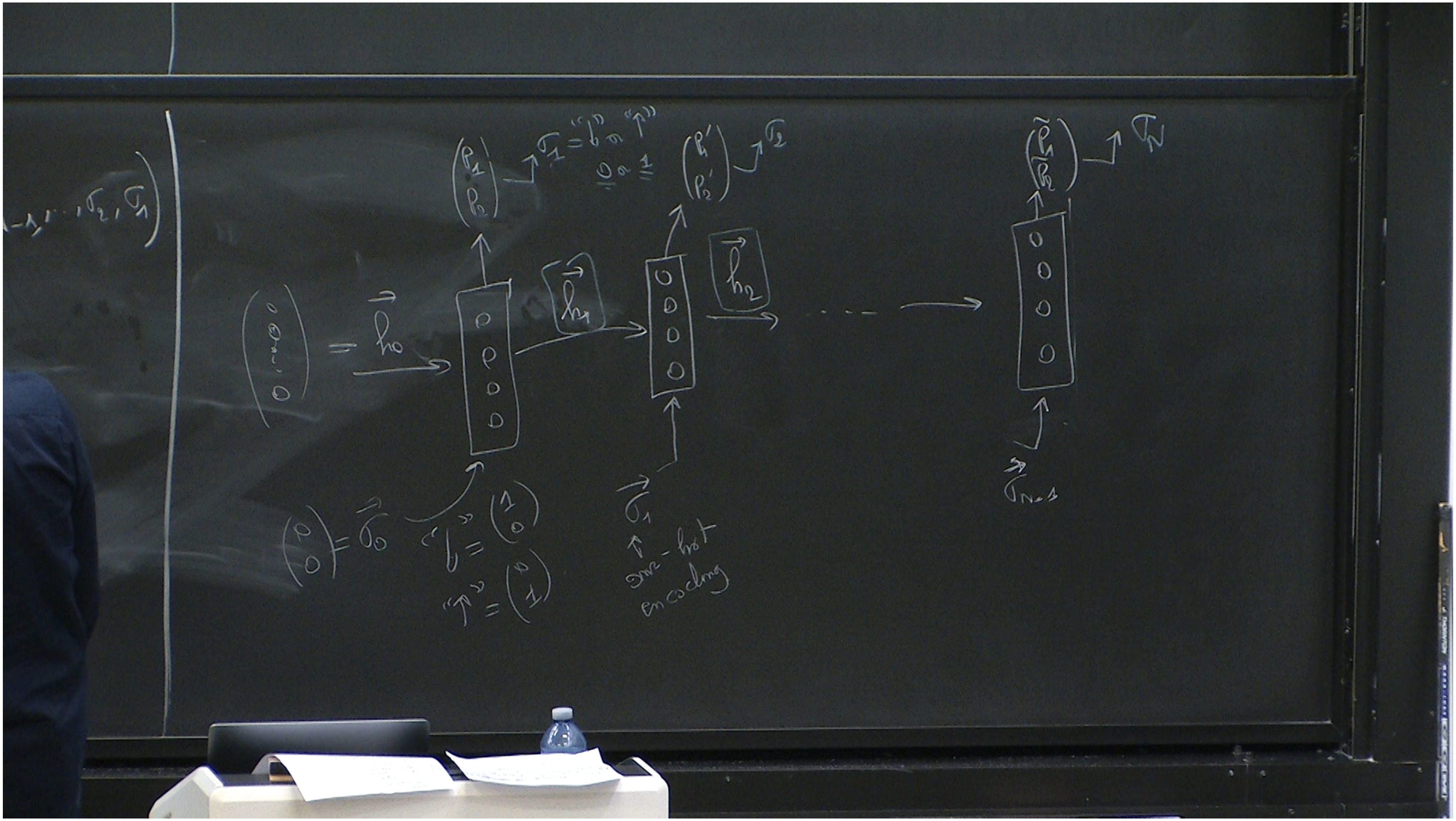

* Key idea
↳ We will be using spins instead of words
«Sentence» ≃ «Spin configuration»
«The weather is nice» ≃ $\vec{\sigma} = (\sigma_1, \sigma_2, \dots \sigma_N)$
Each word had an ID in a dictionary (10K-1M).
Dictionary size for spins = 2.
ce analysis.
chines

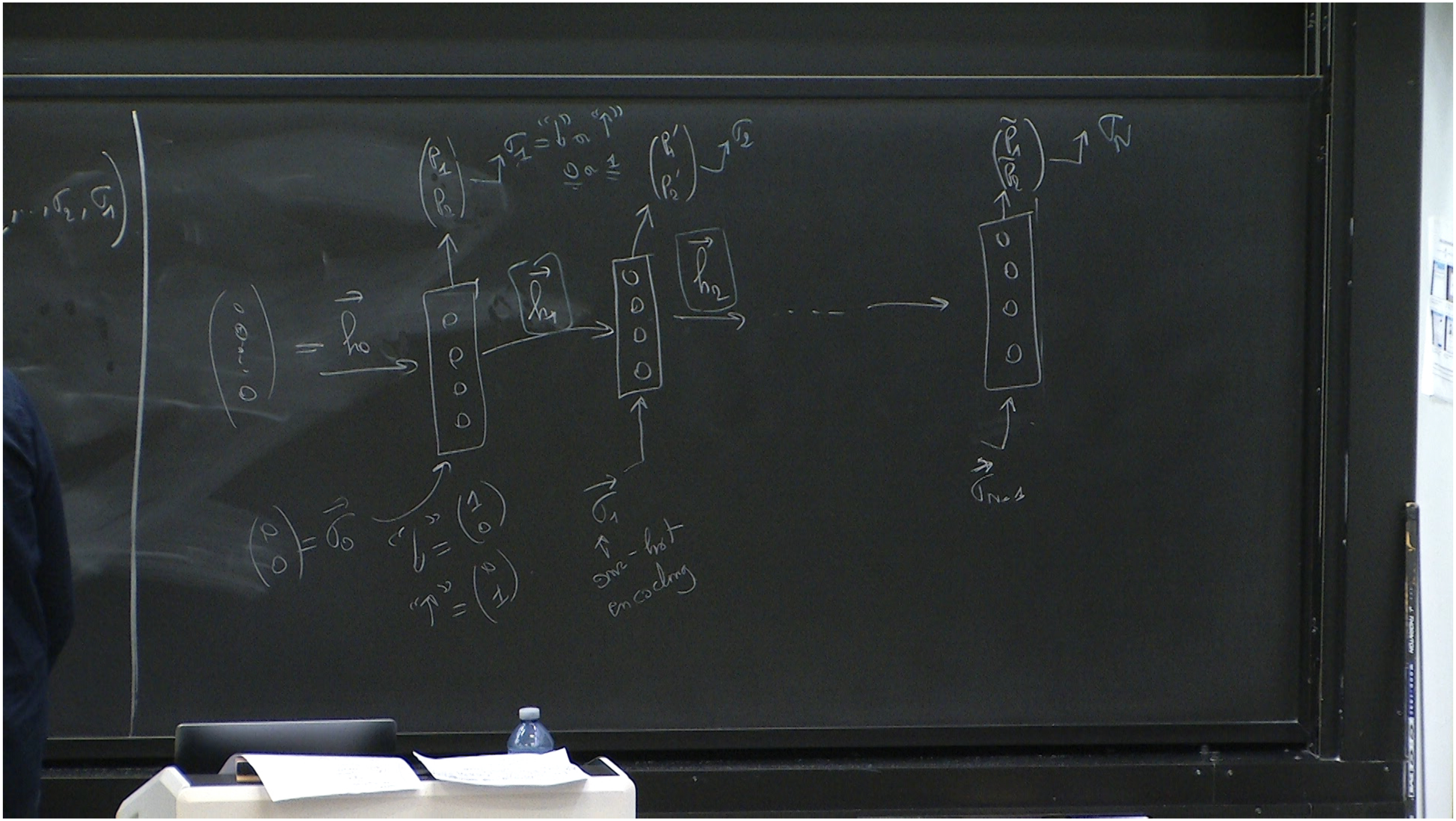
* Probability chain rule:

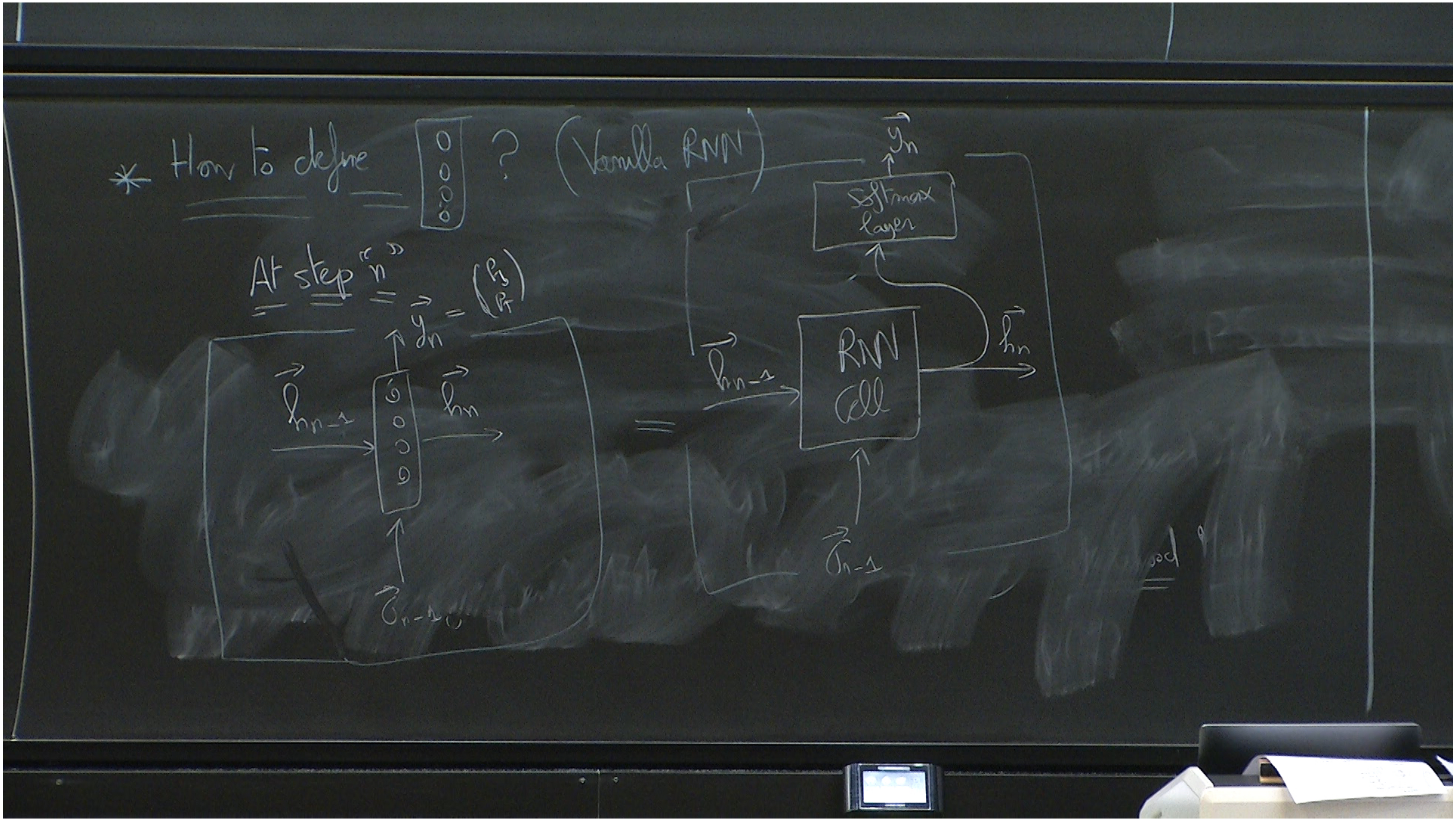
$$P(\sigma_1, \sigma_2, \dots, \sigma_N) = P(\sigma_1)\, P(\sigma_2 | \sigma_1) \dots P(\sigma_N | \sigma_{N-1}, \dots, \sigma_2, \sigma_1)$$

* Autoregressive sampling

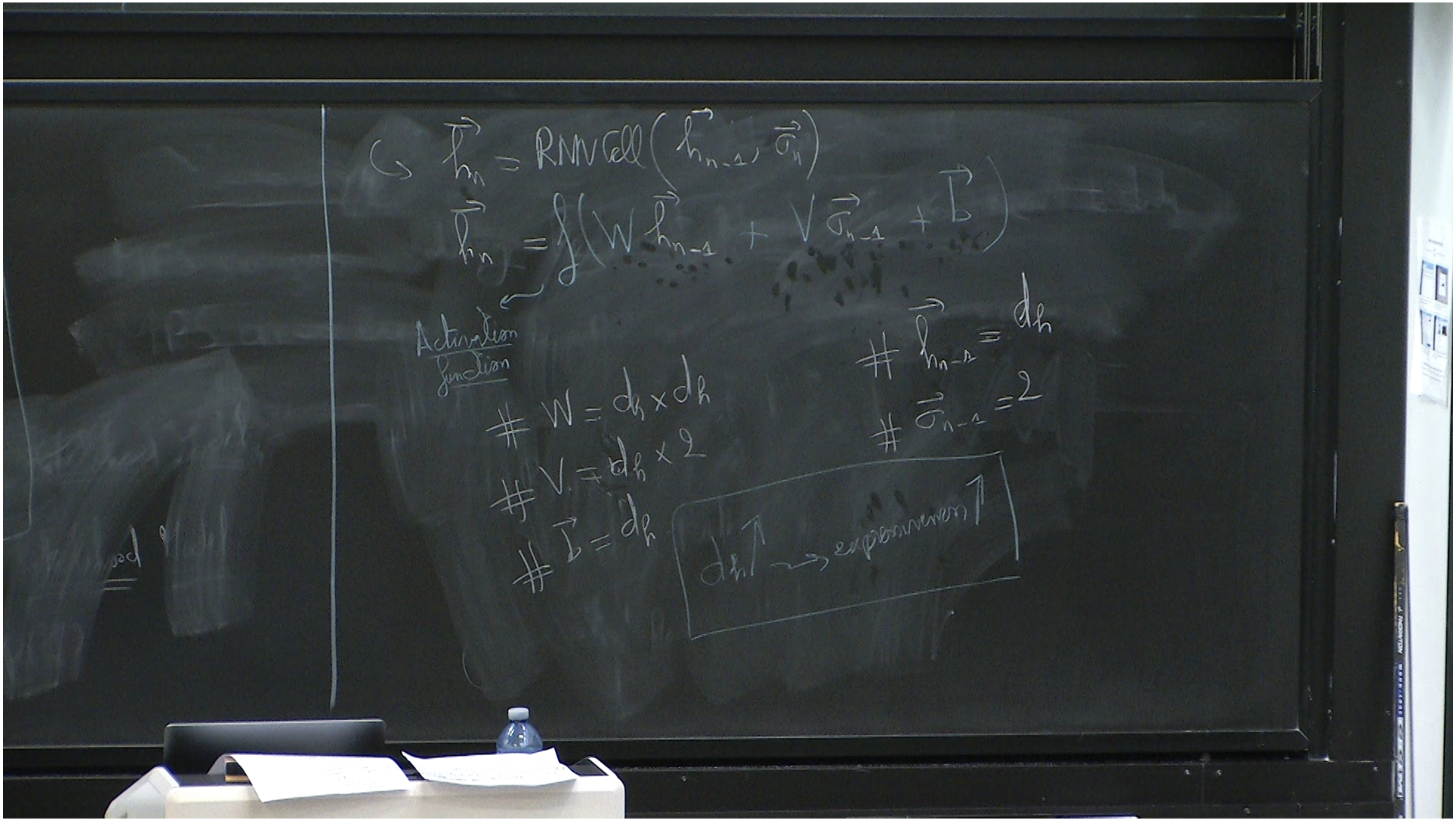

one-hot
encoding


one - hot
encoding


one-hot
encoding



* How to define ?
(Vanilla RNN)
At step "n"
y_n
h_{n-1}
h_n
σ_{n-1}
Softmax layer
RNN Cell


$\vec{h}_n = \text{RNNCell}(\vec{h}_{n-1}, \vec{\sigma}_n)$
$\vec{h}_n = f(W\vec{h}_{n-1} + V\vec{\sigma}_{n-1} + \vec{b})$
Activation function
$W = d_h \times d_h$
$V = d_h \times 2$
$\vec{b} = d_h$
$\vec{h}_{n-1} = d_h$
$\vec{\sigma}_{n-1} = 2$
$d_h \uparrow \rightarrow$ expressiveness $\uparrow$

$$\vec{y}_n = \text{Softmaxlayer}(\vec{h}_n) = \text{Softmax}(U\vec{h}_n + \vec{c})$$

$$\vec{v} = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}. \quad \text{Softmax}(\vec{v}) = \begin{pmatrix} \dfrac{e^{v_1}}{e^{v_1}+e^{v_2}} \\ \dfrac{e^{v_2}}{e^{v_1}+e^{v_2}} \end{pmatrix} \rightsquigarrow \text{probabilistic interpretation of } \vec{y}_n$$

* Key ide

$$\vec{y}_n = \text{Softmaxlayer}(\vec{h}_n) = \text{Softmax}(U\vec{h}_n + \vec{c})$$

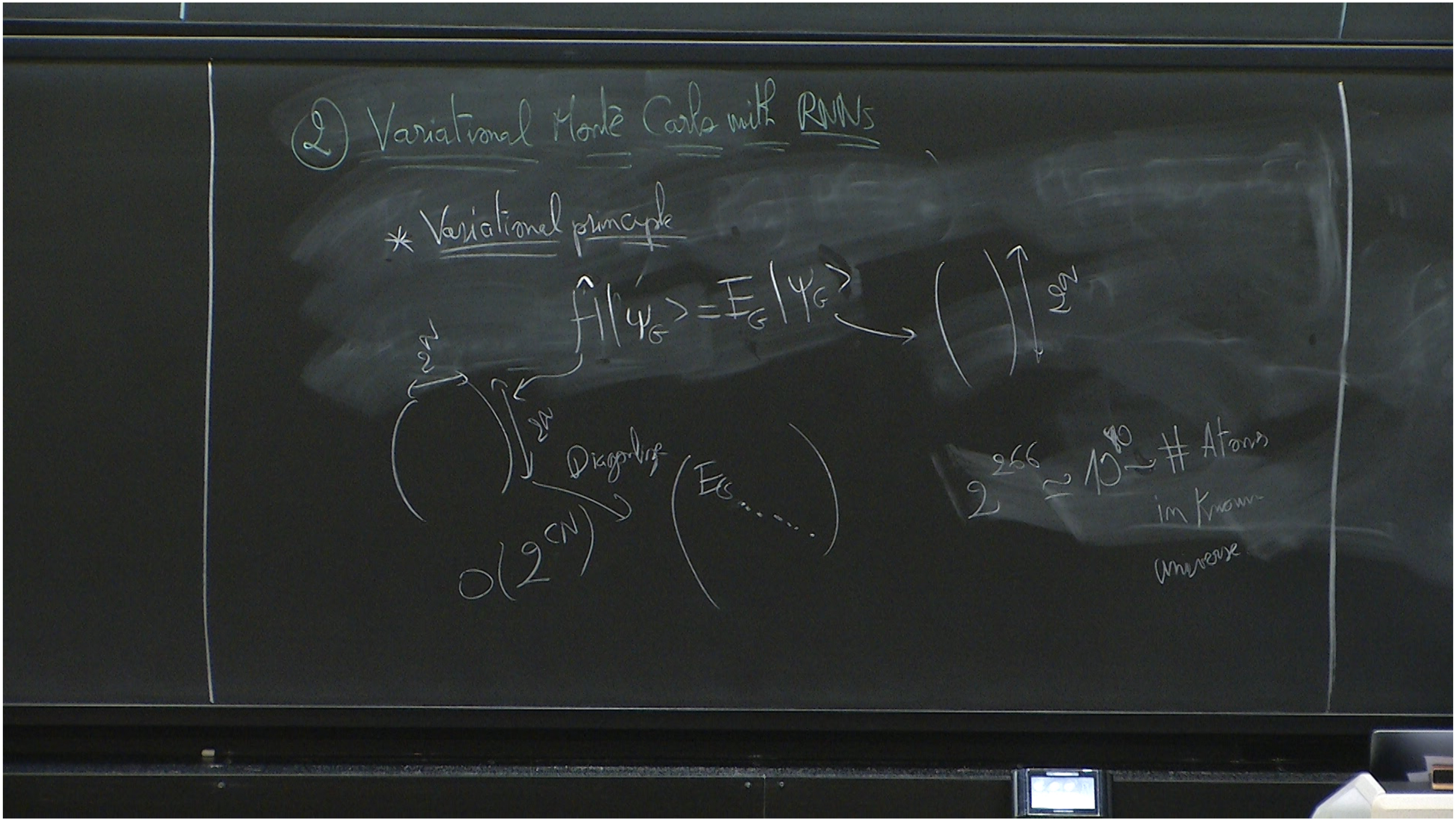
$$\vec{v} = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}. \quad \text{Softmax}(\vec{v}) = \begin{pmatrix} \frac{e^{v_1}}{e^{v_1}+e^{v_2}} \\ \frac{e^{v_2}}{e^{v_1}+e^{v_2}} \end{pmatrix}$$ $\longrightarrow$ probabilistic interpretation of $\vec{y}_n$

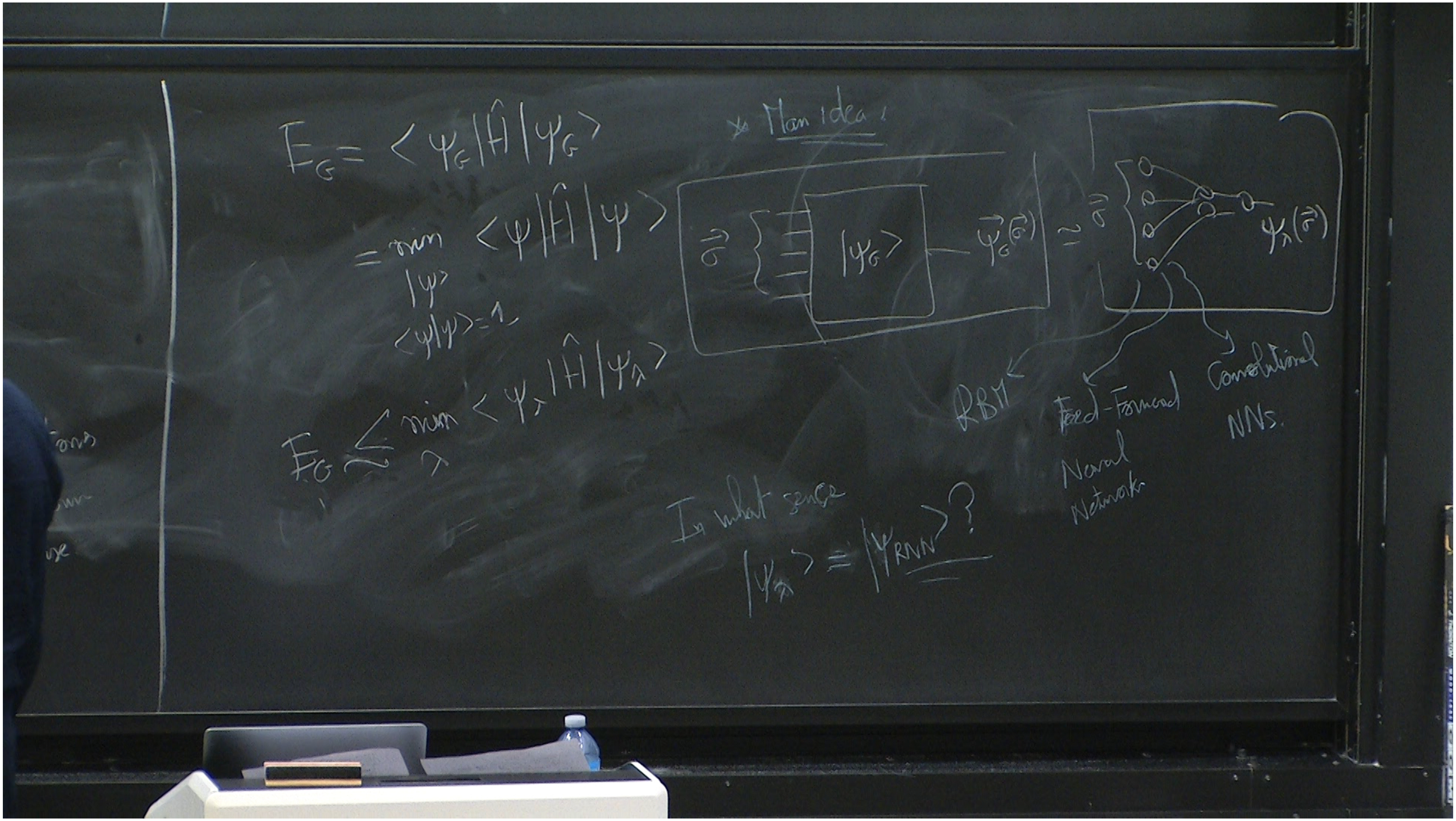
* $P_{RNN} = P_1 \times P_2 \times \dots \times P_N$

↳ Normalized probability distribution

* $P_{RNN} = |\Psi_{RNN}|^2$

↳ RNN wave function


② Variational Monte Carlo with RNNs
* Variational principle
Ĥ|ψG> = EG|ψG>
2^N
Diagonalize
O(2^cN)
EG
2^266 ≃ 10^80 ~ # Atoms in known universe

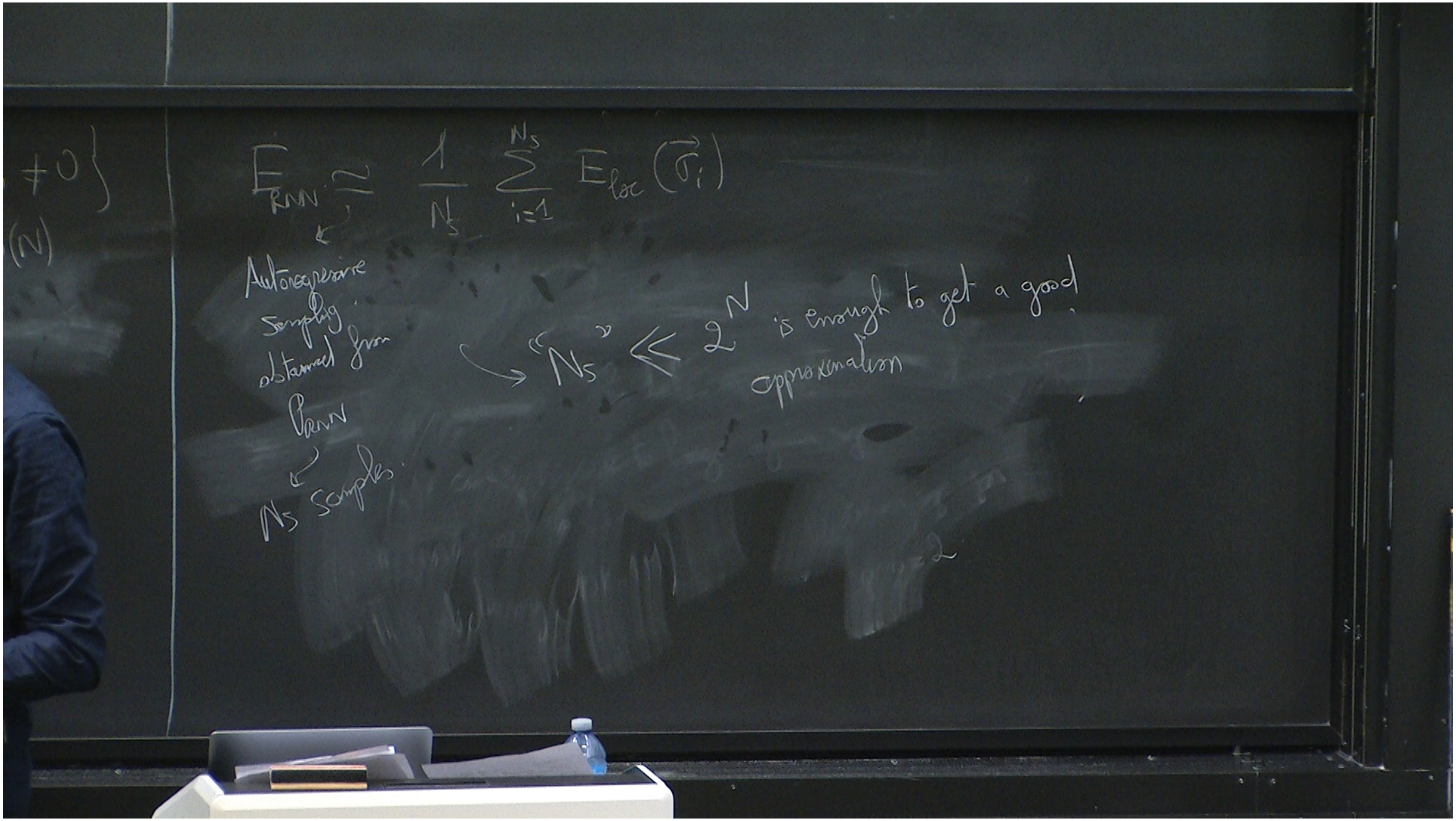


$E_G = \langle \psi_G | \hat{H} | \psi_G \rangle$
$= \min_{|\psi\rangle, \langle\psi|\psi\rangle = 1} \langle \psi | \hat{H} | \psi \rangle$
$E_G \leq \min_{\lambda} \langle \psi_\lambda | \hat{H} | \psi_\lambda \rangle$
Main idea:
$\vec{\sigma}$
$|\psi_G\rangle$
$\psi_G(\vec{\sigma})$
$\simeq$
$\psi_\lambda(\vec{\sigma})$
RBM
Feed-Forward Neural Network
Convolutional NNs.
In what sense
$|\psi_\lambda\rangle \simeq |\psi_{RNN}\rangle$?

$$E_{RNN} = \langle \psi_{RNN} | \hat{H} | \psi_{RNN} \rangle$$

$$= \sum_{\vec{\sigma}, \vec{\sigma}'} \psi^*_{RNN}(\vec{\sigma}) \, H_{\vec{\sigma}\vec{\sigma}'} \, \psi_{RNN}(\vec{\sigma}')$$

$$= \sum_{\vec{\sigma}} \underbrace{|\psi_{RNN}(\vec{\sigma})|^2}_{p_{RNN}(\vec{\sigma})} \underbrace{\left(\sum_{\vec{\sigma}'} H_{\vec{\sigma}\vec{\sigma}'} \frac{\psi_{RNN}(\vec{\sigma}')}{\psi_{RNN}(\vec{\sigma})} \right)}_{\text{Local energy } E_{loc}(\vec{\sigma})}$$


$E_{RNN} \simeq \frac{1}{N_s} \sum_{i=1}^{N_s} E_{loc}(\vec{\sigma}_i)$
Autoregressive sampling obtained from P_{RNN}
N_s samples
"N_s" $\ll 2^N$ is enough to get a good approximation

* How to estimate $\psi_{RNN}(\vec{\sigma})$?

↳ Stoquastic Hamiltonians

$$H_{\vec{\sigma}',\vec{\sigma}} = \langle \vec{\sigma}' | \hat{H} | \vec{\sigma} \rangle \leq 0 \quad \text{when } \vec{\sigma} \neq \vec{\sigma}'$$

$$\psi_G(\vec{\sigma}) \geq 0$$

$$\hat{H} = -\underbrace{\sum_{\langle i,j \rangle} \sigma_i^z \sigma_j^z}_{\text{Diagonal}} - h \underbrace{\sum_i \sigma_i^x}_{\text{off diagonal}}$$

$$\psi_{RNN}(\vec{\sigma}) = \sqrt{P_{RNN}(\vec{\sigma})}$$

$\boxed{\text{Positive RNN wavefunctions}}$

* How to estimate $\psi_{RNN}(\vec{\sigma})$?

↳ Stoquastic Hamiltonians

$$H_{\vec{\sigma}',\vec{\sigma}} = \langle \vec{\sigma}' | \hat{H} | \vec{\sigma} \rangle \leq 0$$

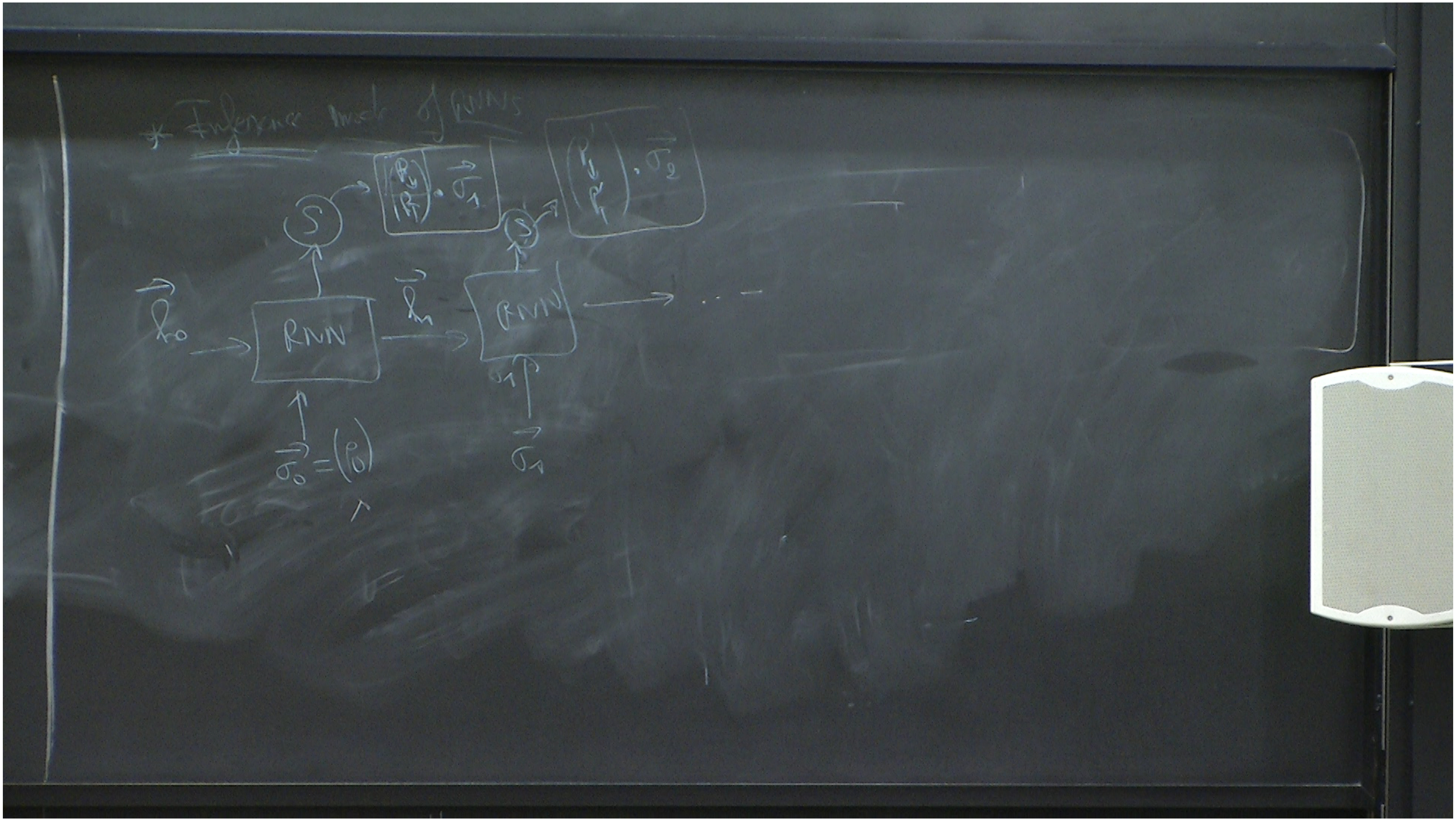
when $\vec{\sigma} \neq \vec{\sigma}'$

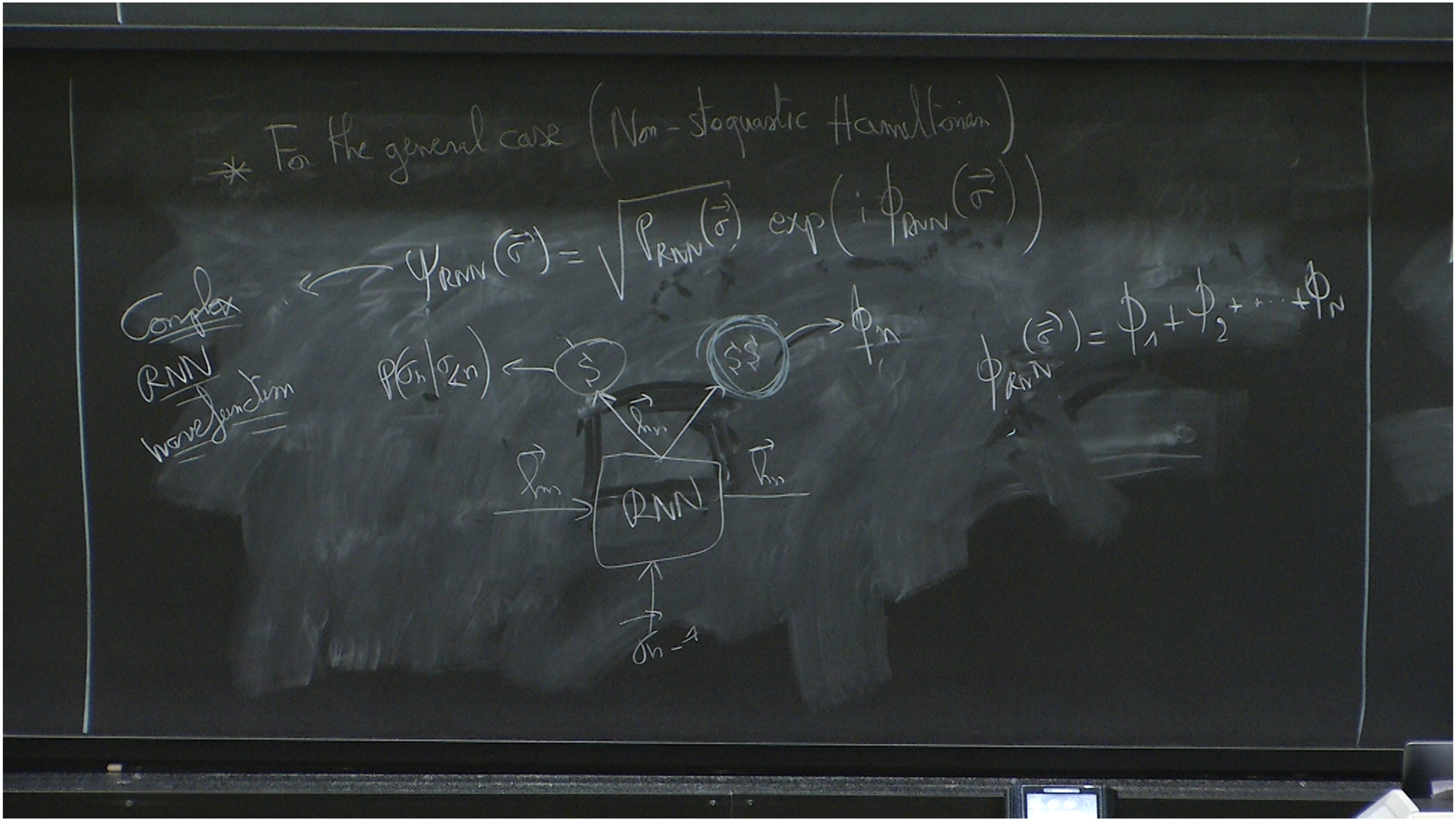
$$\psi_G(\vec{\sigma}) \geq 0$$

$$\hat{H} = \underbrace{-\sum_{\langle i,j \rangle} \sigma_i^z \sigma_j^z}_{\text{Diagonal}} \underbrace{- h \sum_i \sigma_i^x}_{\text{off diagonal}}$$

$$\psi_{RNN}(\vec{\sigma}) = \sqrt{P_{RNN}(\vec{\sigma})}$$

Positive RNN wavefunctions


* Inference mode of RNNs
RNN
RNN


* For the general case (Non-stoquastic Hamiltonian)
Complex RNN wavefunction
$\psi_{RNN}(\vec{\sigma}) = \sqrt{P_{RNN}(\vec{\sigma})} \exp(i \phi_{RNN}(\vec{\sigma}))$
$\phi_{RNN}(\vec{\sigma}) = \phi_1 + \phi_2 + \dots + \phi_N$
$P(\sigma_n | \sigma_{<n})$
S
SS
ϕ_n
$\vec{h}_n$
RNN
$\vec{\sigma}_{n-1}$

$$\vec{\phi}_n = \$\$(\vec{h}_n) = \pi\,\mathrm{Softsign}\left(U'\vec{h}_n + \vec{c}\,'\right)$$

$$\mathrm{Softsign}(x) = \frac{x}{1+|x|} \in (-1,1)$$