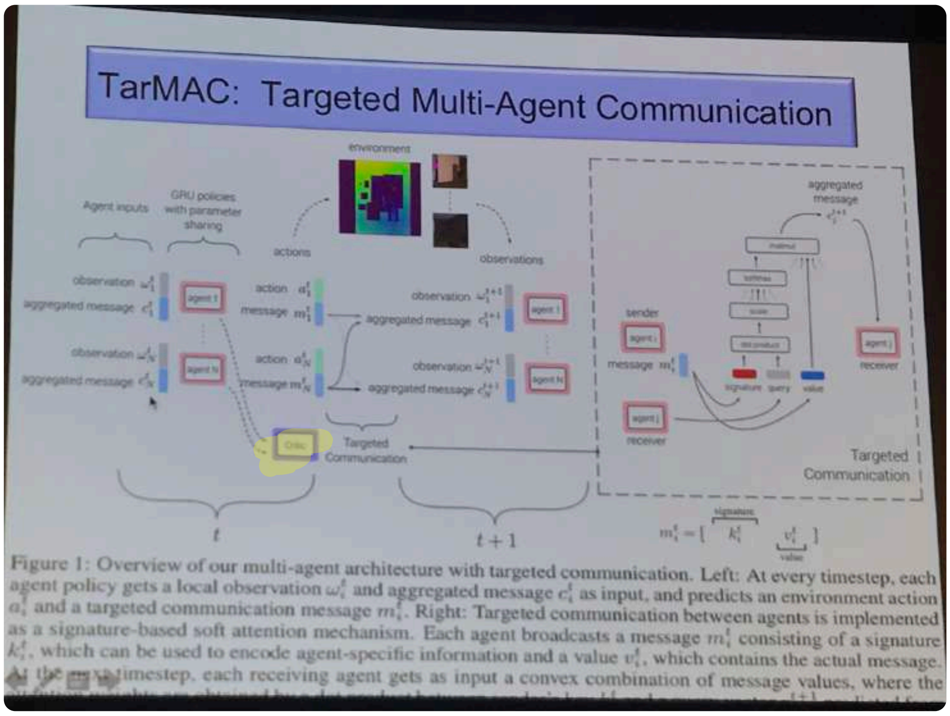
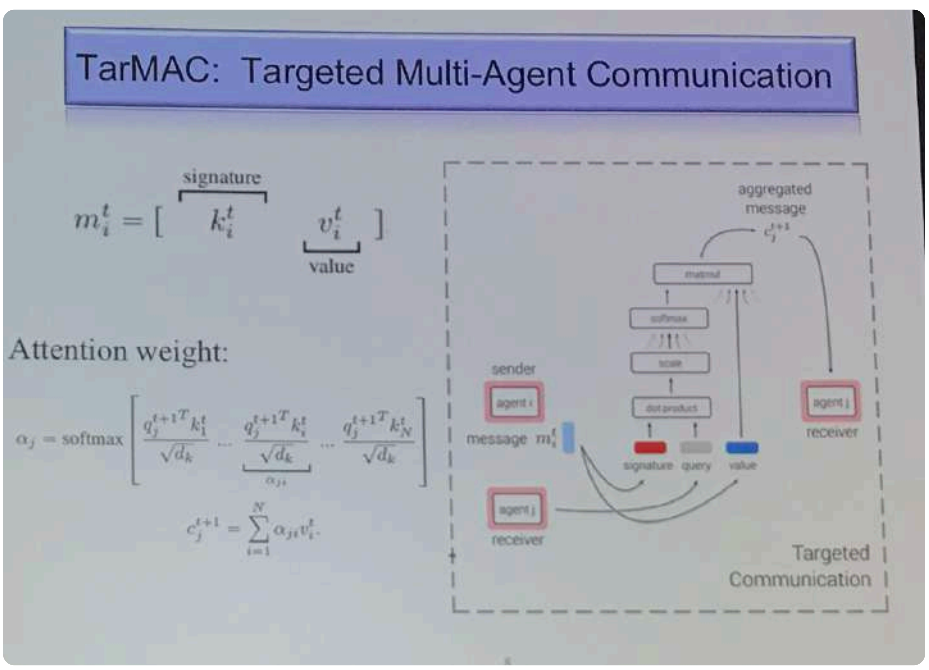
NIPS - DAY 6 — Workshop on RL in Partial Obs. Setting

① J. Pineau — TarMAC: Targeted Multi-Agent Communication

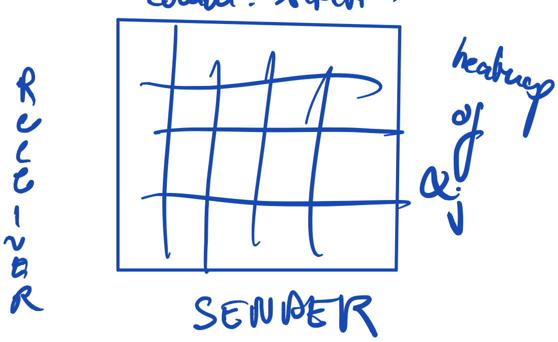
- cooperative, bandwidth-limitation (sending and receiving)!
  → no supervision on communication
- decentralized POMDP setting → agent-specific policy $\pi_{\theta_i}(a_i|\omega_i)$ with actor-critic setup → centralized critic (joint reward fct.)
- Motivation: Self-driving cars → not everyone can communicate at same time



- Trained via backprop through complete architecture
- → aggregate message: compression of all messages
- → agent gets localized observation
- message-signature: intend to speak to receiver!!
- message-attention: intend to listen to speaker!



- Test in shapes domain
  - ↳ agents meet around particular goal
  - ↳ experiment with capped communication channel
    - ↳ one agent dedicated Comm. Attention



↳ if goal is reached by one agent ↳ spike in communication from one agent to others! ←

- Also: traffic junction experiments and Havoc 3D env.
- 1stage / 2stage: multiple rounds of communication → passes!
- Critic: architecture not good at planning! ⇒ deep RL method problem

- push env. complexity and layer abstractions

Comparison to related work

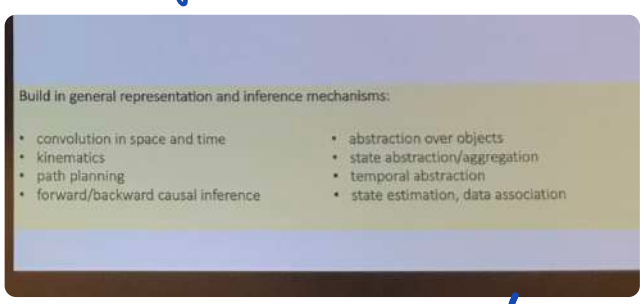
	Decentralized Execution	Targeted Communication	Multi-Stage Decisions	Reinforcement Learning
DIAL (Foerster et al., 2016)	Yes	No	No	Yes (Q-Learning)
CommNets (Sukhbaatar et al., 2016)	No	No	Yes	Yes (REINFORCE)
VAIN (Hoshen, 2017)	No	Yes	Yes	No (Supervised)
ATOC (Jiang & Lu, 2018)	Yes	No	No	Yes (Actor-Critic)
TarMAC (this paper)	Yes	Yes	Yes	Yes (Actor-Critic)

(2) L. Kaelbling – Partial Observability ⇒ Physical Robots (MIT)

- Stigmergic policies (1989) → (Learn policies with external memory)
  → learn traces in env that remind parts of deeper memory
- Generalization over different envs
  (1) Reverse-eng. human
  (2) Recapitulate evolution
  (3) Solve POMDP in 'robot factory' of life!
  (4) Optimal policy as RL POMDP algorithm

→ general purpose & slow

- Biological intelligence
  ↳ Spelke & Kinzler "core knowledge"
  ↳ invariants in world
  ↳ human constraints in engineering loop

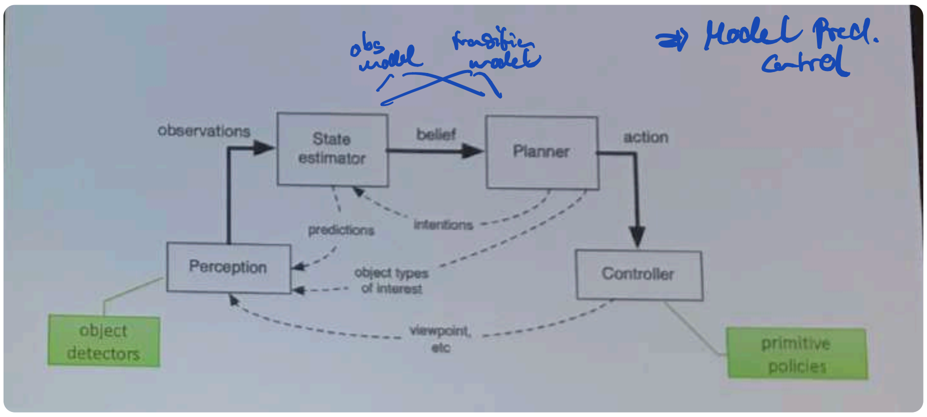


Hand-built vs. Learned
↳ transition models, inference rules, search control

WHAT CONSTITUTES RL?!

State Est. → beliefs
Planning → plans
Action Policy

⇒ Belief-Space Model-Pred. Control with model adaptation



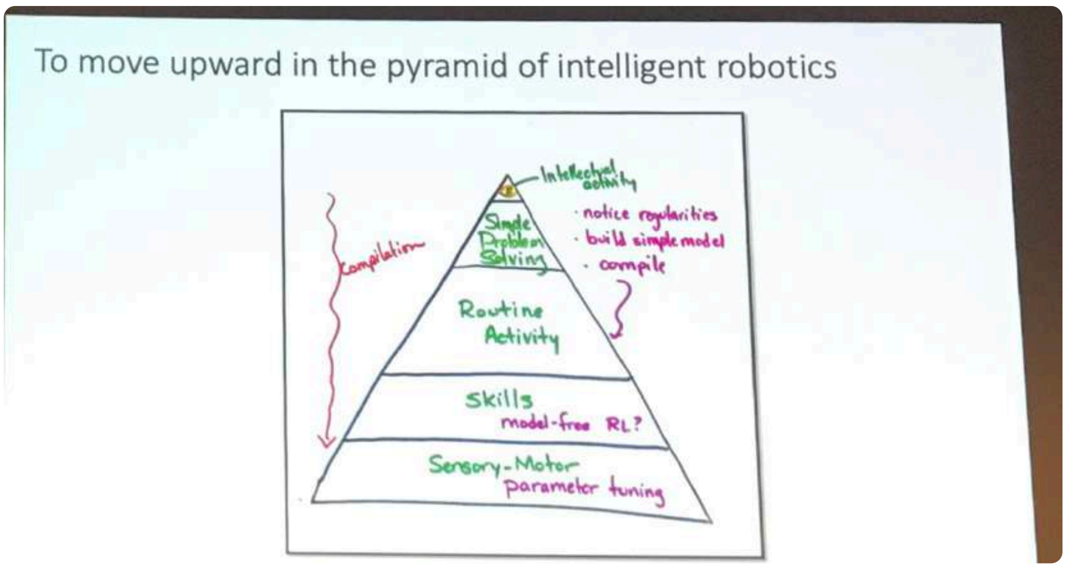
• Observations ①

what action to take → effect on world
→ effect on my own belief of the world

meta-cognition

• Occlusion vs. Noise! → epistemic / irreducible noise!
↓
hard seq. exploration problem on its own

• Dresden (Made Up Novels) books



③ J. Gehring – High-level Strategy StarCraft (FAIR)

- "Fog of war" = incomplete info
- Auxiliary task ⇒ jointly learn hidden state est. along w/ objective
  - ↳ e.g. depth estimation ⇒ bootstraps learning!
  - ↳ here: predict ground-truth opponent unit counts by type
  - → optimize Huber loss → small MLP on top of LSTM

(4) D. Silver – Adaptive Deep RL (DeepMind)

- Combat sensitivity of RL to hyperparameters ⇒ an very brought deep!
  - ↳ sensitivity to changes in env
  - ↳ within one lifetime!
  - → Online adaptation of meta-params
    - large number of resources required!
    - meta = level
    - hyper = not level

Meta-Parameters

- A deep RL algorithm is defined by **update function** f and **meta-parameters** η

$$\theta_{k+1} = \theta_k + f(x, \theta_k, \eta)$$

x: seq. of experience

- e.g. TD(λ) adjusts parameters θ to minimise prediction error w.r.t. the λ-return

$$G_t^\lambda = R_{t+1} + \gamma(\lambda G_{t+1}^\lambda + (1-\lambda) v_\theta(s_{t+1}))$$

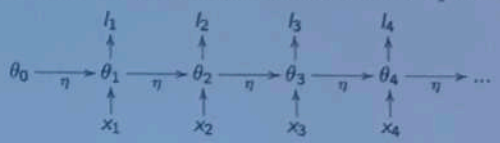
$$\theta_{t+1} = \theta_t + \alpha \frac{\partial (G_t^\lambda - v_\theta(s_t))^2}{\partial \theta}$$

- The nature of the update is determined by meta-parameters $\eta = \{\alpha, \gamma, \lambda\}$
- How can we adapt meta-parameters to optimise an **extrinsic objective**?

- Black box – brute force optimization
  - ↳ population-based training
    - ↳ extrinsic objective = max. return
  - → evolution nested around RL (Stanley & Clune, 91)
- Differentiable memory, hierarchical temporal...

Meta-Gradient Reinforcement Learning (Xu et al.)

Trace effect of meta-parameters on future parameters θ_k and extrinsic losses l_k



The gradient of the extrinsic losses w.r.t. η can be accumulated online

$$\theta_{k+1} = \theta_k + f(x, \theta_k, \eta)$$

$$\frac{d\theta_k}{d\eta} \approx \frac{d\theta_{k-1}}{d\eta} + \frac{\partial f(x, \theta_{k-1}, \eta)}{\partial \eta} \qquad \frac{\partial l_k}{\partial \eta} = \frac{\partial l_k}{\partial \theta_k} \frac{d\theta_k}{d\eta}$$

- → trace effects of meta-params!
  - ↳ "exact RTRL" ⇒ weights
    - ↳ shared params = meta parameters
- Testbed on specifically designed MC ⇒ shortcut alter in noisy-nets
  - ↳ beneficial to switch distant paths!

↳ additive structure → online update – gradients can also simply be accumulated

- Extrinsic loss – policy gradients → meta-meta gradients?!
  - → need to condition on hyperparameters ⇒ as input
    - ↳ similar UVFA!

Conditioned Policy and Value Function

- The meta-parameters (e.g. γ) may determine the **semantics** of the return
- If the meta-parameters change, the semantics of the policy/value change
- It is therefore important to condition the policy and value function on η
  - Provide η as an input to the policy and value function
  - Learn a "universal" policy/value across different meta-parameters

$$v_\theta^\eta(S) = v_\theta([S; \mathbf{e}_\eta]) \qquad \pi_\theta^\eta(S) = \pi_\theta([S; \mathbf{e}_\eta])$$

- This was necessary for strong performance

- DQN ⇒ hardcoded! → not all days help!
  - ↳ change of problem definition!!
  - ↳ Hessel et al ⇒ adaptive DQN

Learning the RL Algorithm

- Deep RL learns **representations**
  - Value/policy represented by a deep neural network
  - Allowing for flexible and powerful representations
  - Efficiently learned by gradient-based optimisation
- Meta-gradient RL learns **update functions**
  - Update rule can be represented by a deep neural network
  - Allowing for flexible and powerful updates
  - Efficiently learned by gradient-based optimisation
- The ultimate goal:
  - A meta-RL algorithm that learns its own RL update rule
  - Adapted during the agent's lifetime in its environment
  - Meta-gradient cost: O(#parameters x #meta-parameters)

→ Unlikely scalability!

⑤ A. Grishin – Joint Belief Tracking and Focused Opt. via Approx. Inf.

Frameworks

MDP: state $\xrightarrow{policy}$ action

POMDP: observation $\xrightarrow{encoder}$ **state representation** $\xrightarrow{policy}$ action

Usual approach in POMDP

Encoder: compress history or maintain belief over state
Policy: predict optimal action based on state representation

Tasks are set **separately**

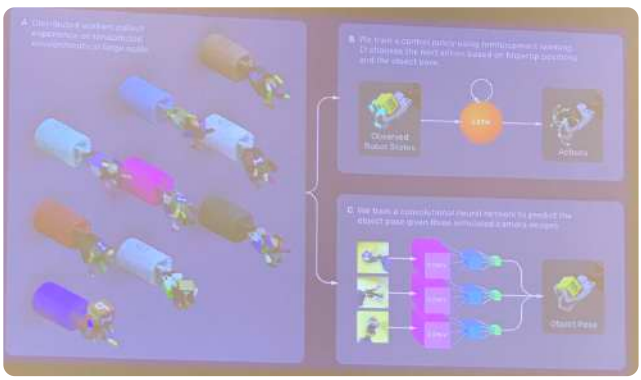
Our approach

Joint probabilistic model for
- belief tracking (rewards as observation)
- policy optimization (rewards as target)

⑥ L. Way – Learning Dexterity (OpenAI)

- High-dim control dexterity hand ⇒ 24 dof, 20 actuators
  → partial observability → finger tip markers occluded by object!
- Sim2Real: train in simulation and deploy in reality
  - phase space tracking of finger positions! ⇓ transfer!
    → physics + visual!
- Domain randomization ⇒ train for robustness!

① Asher Model LSTM
② Control Model LSTM
③ Vision Model multi-camera CNN



⑦ P. Abeel – RL under Env. Uncertainty

(1)

- Model-based ⇒ more sample efficient → but gets to lower level
  - ↳ overfitting: policy opt. exploits regions where insufficient data available
    ⇒ model-bias → solution: model-ensemble TRPO (ME-TRPO)
- Here: Model-based Meta-Policy Optimization
  - ↳ adaptive policy robust to dynamics randomization & also robust in real deployments
    - ↳ learn ensemble simulators to capture dynamics of the world

Model-Based RL via Meta Policy Optimization (MB-MPO)

for iter = 1, 2, …
- collect data under current adaptive policies $\pi_{\phi_1}, \ldots, \pi_{\phi_K}$
- learn **ENSEMBLE** of K dynamics models from all past data
- **meta-policy optimization over ENSEMBLE**
  - → new meta-policy π_θ
  - → new adaptive policies $\pi_{\phi_1}, \ldots, \pi_{\phi_K}$ → meant ub

(2)

- Representation Learning ⇒ for exploration based on prior knowledge
  - ↳ representations of behaviors not memory states! ⇒ generic!
  - ↳ Gupta et al (2018) ⇒ Model-agnostic (MESN)

(3)

- Fast RL via slow RL
  - ↳ envs are all mathematical defined ⇒ how to leverage advantage of prior physics learn
  - ↳ Meta-learning: learn the learning algo → how do envs need to be like?!
  - ↳ Duan et al (2016): train = test → every time new env is generated
    - ↳ both rules/outputs are probabilistic!

⑧ E. Todorov – Solving estimation and control problems with one machinery

Linearly-solvable MDPs

Todorov 2006-2010

In traditional MDPs the controller chooses actions u which in turn specify the transition probabilities $p(x'|x,u)$. We can obtain a linearly-solvable MDP (LMDP) by allowing the controller to specify these probabilities directly:

$x' \sim u(\cdot\|x)$	controlled dynamics
$x' \sim p(\cdot\|x)$	passive dynamics
$p(x'\|x)=0 \Rightarrow u(x'\|x)=0$	feasible control set

The running cost is in the form

$$\ell(x,u(\cdot|x)) = q(x) + D_{KL}(u(\cdot|x)\,||\,p(\cdot|x))$$

Thus the controller can impose any dynamics it wishes, however it pays a price (KL divergence control cost) for pushing the system away from its passive dynamics.

(related results in continuous time)

Optimal control law:

$$u^*(x'|x) = \frac{p(x'|x)\,z(x')}{\mathcal{P}[z](x)}$$

$$z(x) \triangleq \exp(-v(x))$$

$$\mathcal{P}[z](x) \triangleq \sum_{x'} p(x'|x)\,z(x')$$

Linear Bellman equation:

$$z(x) = \exp(-q(x))\,\mathcal{P}[z](x)$$

linear operator

→ still can't solve due to high-dim.!

Estimation-control duality

The probability of a trajectory under the *passive* dynamics is

$$p(x_1, x_2, \cdots x_T|x_0) = \prod_{t=0}^{T-1} p(x_{t+1}|x_t)$$

The probability of a trajectory under the *optimally-controlled* dynamics is

$$\mu(x_1, x_2, \cdots x_T|x_0) \propto \exp\left(-\sum_{t=1}^{T} q(x_t)\right) p(x_1, x_2, \cdots x_T|x_0)$$

This is equivalent to Bayesian inference where p is the prior, μ the posterior, and q the negative log-likelihood (of some unspecified measurements).

Let $\mu_k(x)$ be the marginal of the trajectory density at time step k, and define $z_k(x) = \mu_k(x)/z_0(x)$. Then

$$\begin{aligned} z_k(x) &= \exp(-q(x)) \sum_{x'} p(x'|x)\, z_{k+1}(x') \\ z_{k+1}(x') &= \sum_x \exp(-q(x))\, p(x'|x)\, z_k(x) \\ \mu_k(x) &= z_k(x)\, z_k(x) \end{aligned}$$

This is equivalent to recursive Bayesian inference where z_k is the filtering density, z_k the backward filtering density, and μ_k the posterior.

General case:

$$x_{t+1} \sim u(\cdot|x_t)$$

$$\ell(x,u) = q(x) + D_{KL}(u(\cdot|x)\,||\,\bar{p}(\cdot|x))$$

is dual to

$$x_{t+1} \sim p(\cdot|x_t)$$

$$y_t \sim \text{Bernoulli}(r(x_t))$$

when

$$q(x) = -\log(r(x))$$

Linear case:

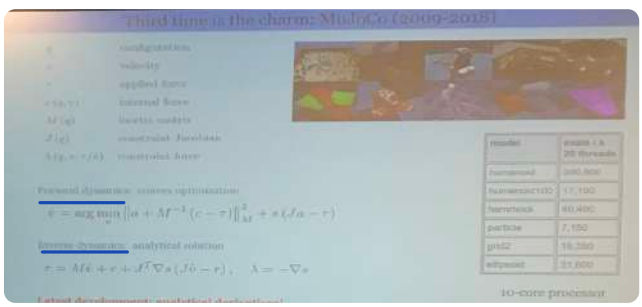
LQR is dual to Information filter

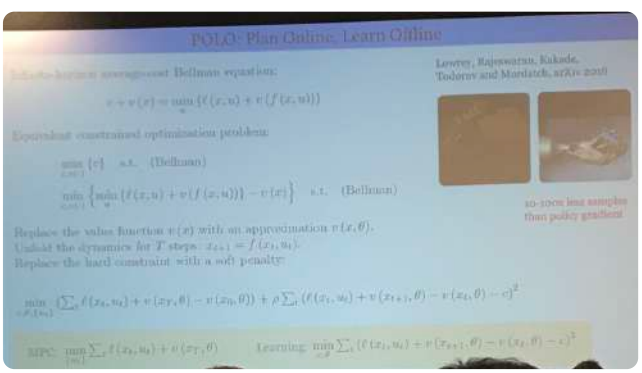
Kalman

Stochastic ↑ Det.

↳ solution same for linear systems

- MuJoCo → based on bad prev. solutions
- Bayesian posterior vs. optimal values





- Optico SDK
  Costs, derivatives, optimizers

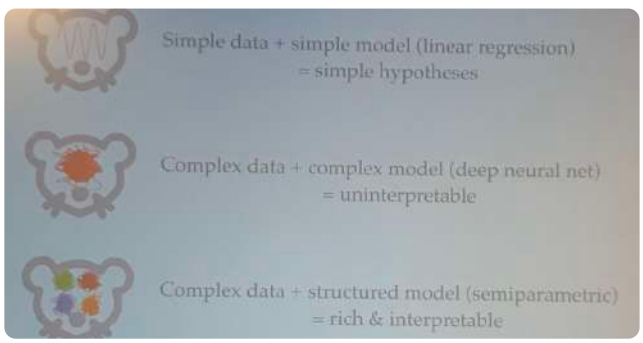
③ R. Adams – Inference and Control of Learning Behavior in Rodents

- Task learning in rats (Brody lab Princeton)
  - → high / low tones ⇒ L together with a lag → What is rat's algo?
  - → Roy et al (2018 – NIPS)! ⇒ E-E, disengagement
  - → observation: slow learning in rodents

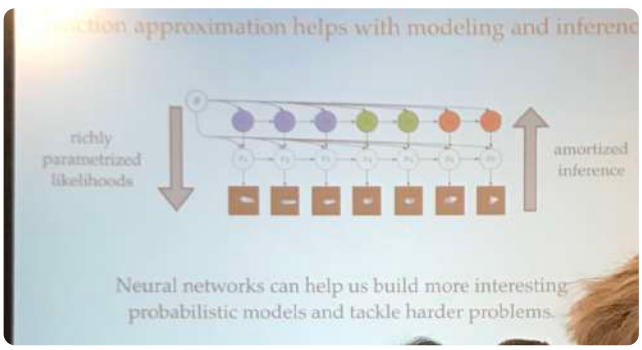
- Wiltschko et al (2015)

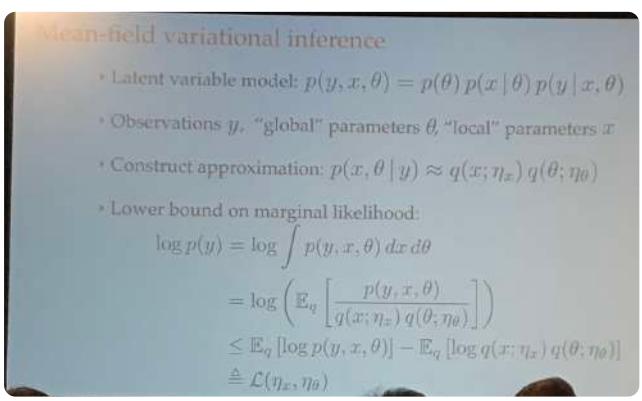
→ unsupervised discovery of language of behavior

→ mouse ≈ switching LDS
⇒ Kalman Filter

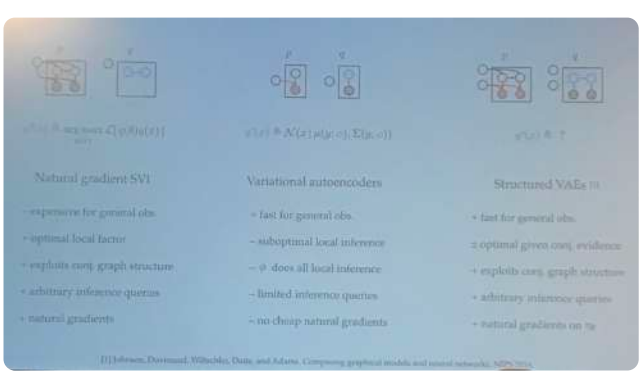


→ can also use VAE indeed!



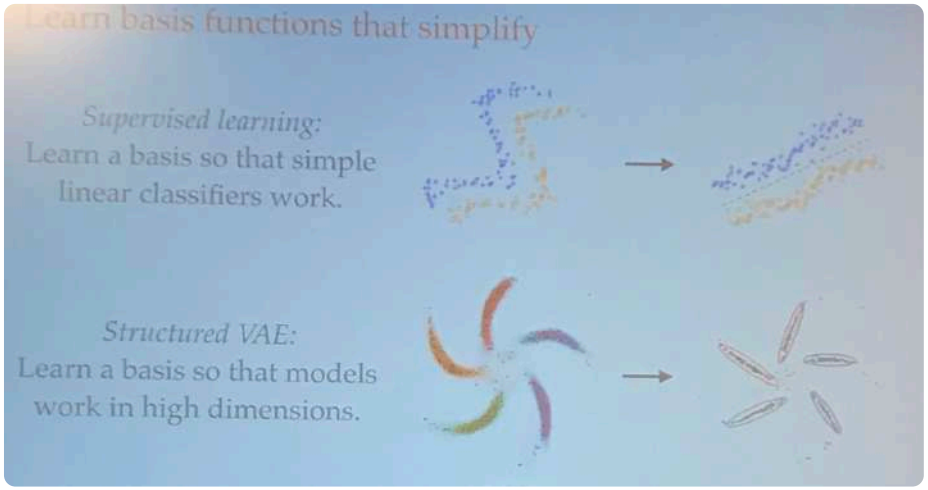


→ turn into Stochastic VI on global level!



Amortized inference for non-conjugate models

↳ Kingma and Welling
(Auto-encoding variational Bayes)
(2014)



→ Finn (2018) talk by Gordon Berman

Links also useful for annotation of movement generative models!