Programming by Feedback

Riad Akrour, **Marc Schoenauer**, **Jean-Christophe Souplet**, **Michèle Sebag**
TAO, CNRS — INRIA — LRI, Université Paris-Sud, France

Motivations: It is time for a 3rd programming age

1970s Specifications — **Languages & thm proving**

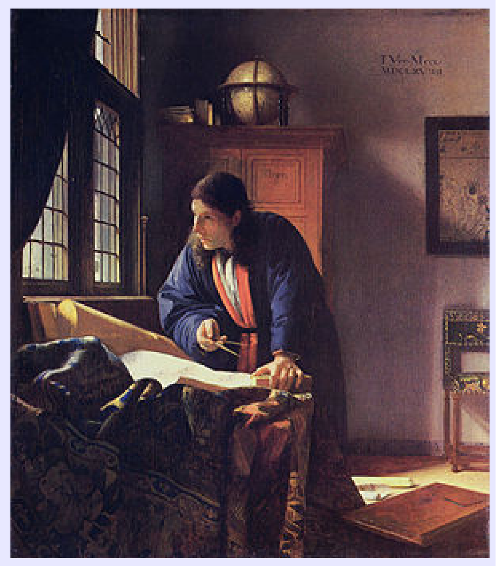
1990s Programming by Examples — **Pattern recognition & ML**

2010s Interactive Learning and Optimization

- Visual rendering — Brochu et al. 2010
- Information retrieval — Joachims et al., 2012
- Robotics — Knox et al. 2010, Akrour et al., 2012; Wilson et al., 2012; Saxena et al. 2013

Programming by Feedback, overview

Active Computer — Critic User



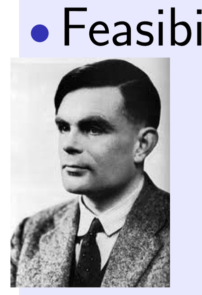
Knowledge-constrained — **Computation, memory-constrained**

Algorithm: Iterate

1. Computer presents the user with a pair of behaviors y_{t_1}, y_{t_2}
2. User emits preferences $y_{t_1} \succ y_{t_2}$
3. Computer updates User's utility function
4. Computer searches for behavior *with best expected posterior utility*

Conclusion and Perspectives

- Feasibility of the **Programming by Feedback** paradigm.



One could carry through the organization of an intelligent machine with only two interfering inputs, one for pleasure or reward, and the other for pain or punishment.

- Importance of noise: all users make mistakes. The computer must trust the user to a limited extent. Beware that computer distrust increases the user mistakes.
- Next: Identifying the sub-behaviors responsible for the expert's like/dislikes, taking inspiration from — Wilson et al. 2012
- Next: Accounting for the variance of the behaviors associated to a solution (multi-objective optimization).

Formally

$\mathcal{X}$ ($\mathbb{R}^D$) Search space, solution space (controllers in RL)

$\mathcal{Y}$ ($\mathbb{R}^d$) Evaluation space (behaviors, trajectories, demonstrations)

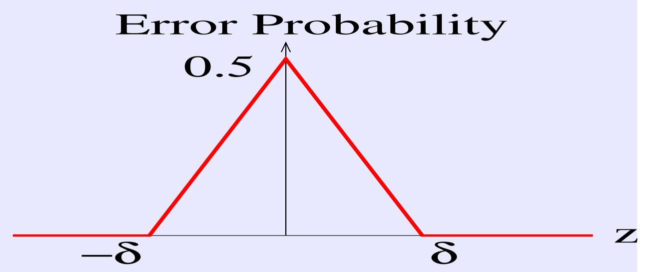
True utility function U^* (with unknown w^* in W):

$$U : \mathcal{Y} \mapsto \mathbb{R}, U(y) = \langle w^*, y \rangle$$

Modelling the user's competence: Noise model $\delta \sim U[0, M]$

Given preference margin $z = \langle \mathbf{w}^*, y - y' \rangle$

$$P(y \prec y' \mid \mathbf{w}^*, \delta) = \begin{cases} 0 & \text{if } z < -\delta \\ 1 & \text{if } z > \delta \\ \frac{\delta + z}{2\delta} & \text{otherwise} \end{cases}$$



Learning the user's utility function — find θ_t posterior on W

Proposition. Given evidence $\mathcal{U}_t = \{y_0, y_1, \ldots; (y_{i_1} \succ y_{i_2}), i = 1 \ldots t\}$,

$$\theta_t(\mathbf{w}) \propto \prod_{i=1,t} P(y_{i_1} \succ y_{i_2} \mid \mathbf{w}) = \prod_{i=1,t} \left(\frac{1}{2} + \frac{\mathbf{w}_i}{2M} \left(1 + \log \frac{M}{|\mathbf{w}_i|} \right) \right)$$

with $\mathbf{w}_i = \langle \mathbf{w}, y_{i_1} - y_{i_2} \rangle$, capped to $[-M, M]$.

Most informative demonstrations (y, y') ?

Expected utility of selection:

$$EUS(y, y') = \mathbb{E}_{\theta_t}[\langle \mathbf{w}, y - y' \rangle > 0] \; . \; U(\theta_t^+, y) + \mathbb{E}_{\theta_t}[\langle \mathbf{w}, y - y' \rangle < 0] \; . \; U(\theta_t^-, y')$$

Expected posterior utility:

$$\begin{aligned} EPU(y, y') &= \mathbb{E}_{\theta_t}[\langle \mathbf{w}, y - y' \rangle > 0] \; . \; max_y U(\theta^+, y) + \mathbb{E}_{\theta_t}[\langle \mathbf{w}, y - y' \rangle < 0] \; . \; max_y U(\theta^-, y) \\ &= \mathbb{E}_{\theta_t}[\langle \mathbf{w}, y - y' \rangle > 0] \; . \; U(\theta^+, y^*) + \mathbb{E}_{\theta_t}[\langle \mathbf{w}, y - y' \rangle < 0] \; . \; U(\theta^-, y'^*) \end{aligned}$$

Therefore — Viappiani & Boutilier 10

Find argmax $EUS(y, y')$

Optimization in the demonstration space

Proposition. $EUS^{noiseless}(y, y') - L \leq EUS^{noise}(y, y') \leq EUS^{noiseless}(y, y')$

Proposition. $EUS_t^{*,noiseless} - L \leq EPU_t^{*,noise} \leq EUS_t^{*,noiseless} + L$

Optimization in the solution space

- Find argmax $EUS(y_t^*, y)$ — decreases cognitive burden
- Given the mapping Φ: Solution $\mapsto$ Demonstration space,

$$\mathbb{E}_\Phi[EUS^{NL}(\Phi(x), y_t^*)] \geq EUS^{NL}(\bar{y}, y_t^*)$$

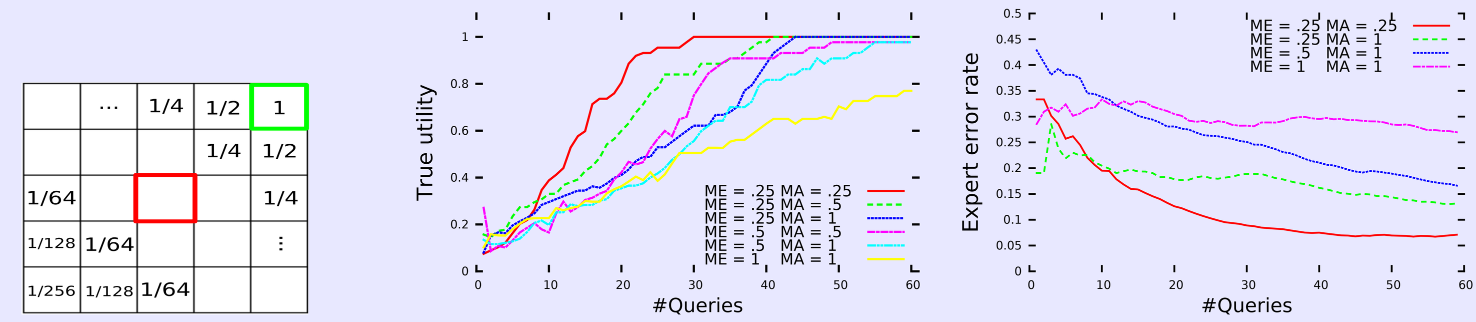
- Draw $\mathbf{w}_0 \sim \theta_t$ and let $\mathbf{x}_1 =$ argmax $\langle \mathbf{w}_0, \bar{y} \rangle$
Iteratively, find $\mathbf{x}_{i+1} =$ argmax $\langle \mathbb{E}_{\theta_i}[\mathbf{w}], \bar{y} \rangle$, with θ_i posterior with $\bar{y}_i > \bar{y}_t^*$.

Proposition. The sequence monotonically converges toward a local optimum of $EUS^{noiseless}$.

Experimental study

Grid world: Discrete Case, no Generative Model

25 states, 5 actions, horizon 300, 50% transition motionless



True $\mathbf{w}^*$ on gridworld — True utility of $\mathbf{x}_t$ — user's mistakes

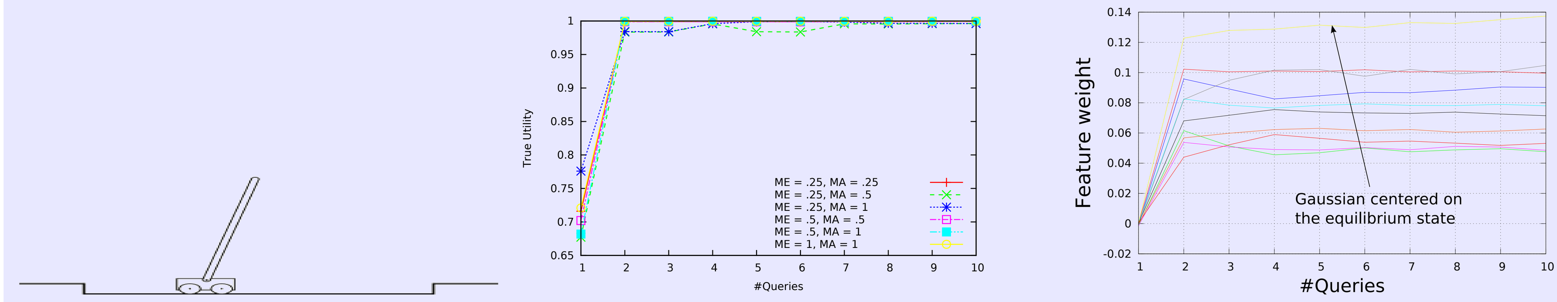
Sensitivity study wrt user's competence (M_E) and computer trust (M_A):

a cumulative (dis)advantage phenomenon

The number of (emulated) user mistakes *increases* as the computer underestimates the user's competence. For low M_A, the computer learns faster, submits more relevant demonstrations to the user, thus priming a virtuous educational process.

The Cartpole: Continuous Case, no Generative Model

State space $\mathbb{R}^2$ (the angle and angular velocity of the pendulum), 3 actions; demonstration length 3,000.

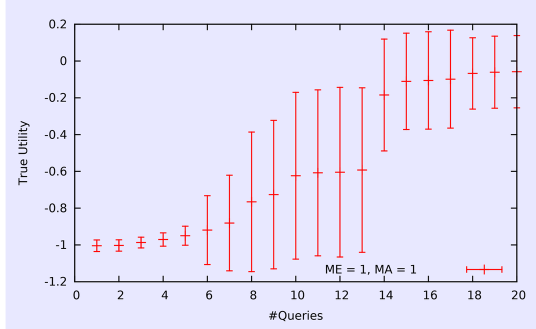


Cartpole — True utility of $\mathbf{x}_t$ — Estimated utility of features

Demonstration space $\mathcal{Y} = \mathbb{R}^9$ (feature = Gaussian in state space). Simulated user's feedback: best demonstration is the longest one (+ noise). True utility: fraction of the demonstration in equilibrium.

Two interactions required on average to solve the cartpole problem, irrespective of the noise model hyper-parameters

The Bicycle: Continuous Case, with Generative Model



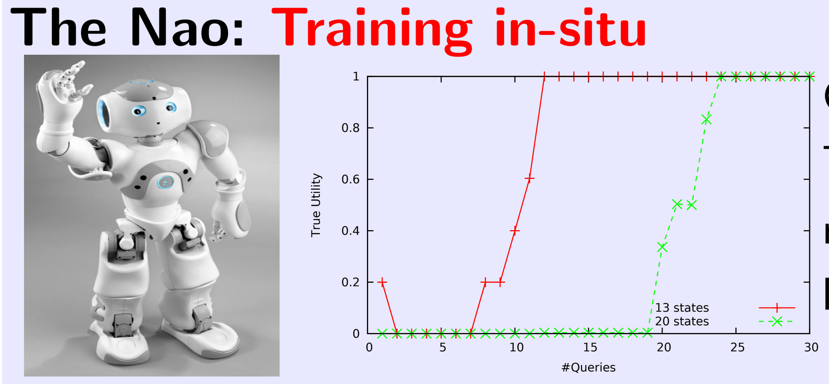
True utility of $\mathbf{x}_t$

State space $\mathbb{R}^4$, action space $\mathbb{R}^2$, demonstration length $\leq 30,000$. Solution space $\mathcal{X} \subseteq \mathbb{R}^{210}$ (weight vector of a 1-layer feedforward NN with 4 input, 29 hidden neurons and 2 output). Optimization component: CMA-ES black box optimization (Hansen et al., 2001) as LSPI fails with the estimated utility function.

15 interactions required on average to solve the bicycle problem for the low noise setting ($M_E = M_A = 1$).

Improves on the state of the art: circa 20 queries required with discrete action space in Wilson et al. 2012; explained from the more compact search space (V as opposed to Q).

The Nao: Training in-situ



The Nao robot — True utility of $\mathbf{x}_t$

Goal: reaching a given state. Transition matrix estimated from 1,000 random (s, a, s') triplets. Demonstration length 10, initial state is fixed.