Gaussian processes and the common ground of decision making under uncertainty

Javier González
Amazon Cambridge

Warwick, 2019, UK

March 8, 2019

Purpose of this talk

Bandits, Bayesian optimization, Active learning, Bayesian quadrature and Model based RL

- Machine learning targets automatic decision making.
- Sequential decision methods are usually studied separately.
- Less common to look/implement these methods all together.
- This is the perspective of this talk.

Key elements:

Data efficient belief representations + policies/utilities

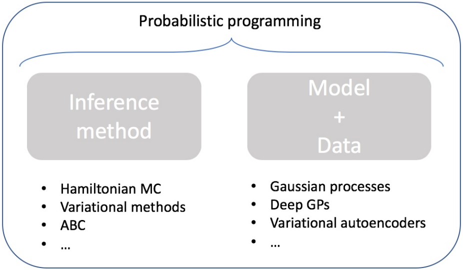
Goal:

General recipe to create and prototype new methods.

Talk inspired on some the work of Marc Toussaint on POMDPs. Here we focus more on the belief models used.
Marc Toussaint. The Bayesian Search Game. Theory and Principled Methods for the Design of Meta. 2014.

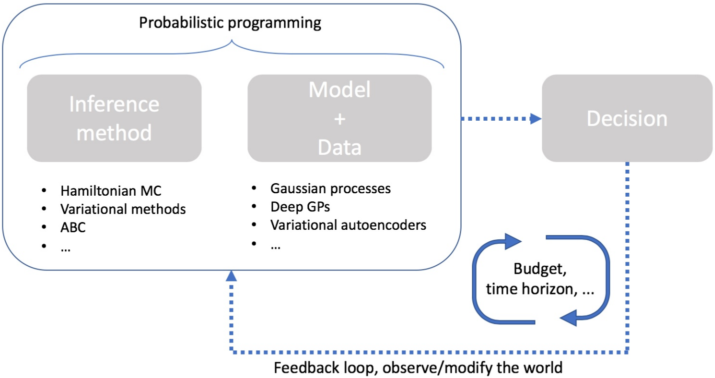
Probabilistic machine learning

Data + model (inference) → predictions → decisions



Probabilistic machine learning and decision making

Data + model (inference) $\rightarrow$ predictions $\rightarrow$ decisions



Other fields have different versions of this recipe.

- ML focuses primarily on the data + modeling hypothesis.
- OR, for instance, focuses more on mechanisms.

Decisions under uncertainty

From inference to 'static' decisions making.

Inference (belief)

- Things that I know:

$$y$$

- Things that I don't know:

$$y*$$

- Description of the world:

$$p(y^*, y)$$

- What I need:

$$p(y^*|y)$$

Decisions (policy)

- Actions I can take:

$$a \in \mathcal{A}$$

- Reward I gain:

$$R(a|y, y^*)$$

- 'Optimal' decision:

$$a^* = arg \max_{\mathcal{A}} \alpha(a; R, p)$$

$$\alpha(a; R, p) = \mathbb{E}_p \left[R(a|y, y^*) \right]$$

Decisions under uncertainty

From inference to ‘static’ decisions making.

Inference (belief)

- Things that I know:

$$y$$

- Things that I don’t know:

$$y*$$

- Description of the world:

$$p(y^*, y)$$

- What I need:

$$p(y^*|y)$$

Decisions (policy)

- Actions I can take:

$$a \in \mathcal{A}$$

- Reward I gain:

$$R(a|y, y^*)$$

- ‘Optimal’ decision:

$$a^* = arg \max_{\mathcal{A}} \alpha(a; R, p)$$

$$\alpha(a; R, p) = \mathbb{E}_p \left[R(a|y, y^*) \right]$$

Bandits

Bandits

As an archetype of sequential decision methods

Problem definition:

- We can play T times on n machines.
- Each machine provides a reward $y = p(y;\theta)$.
- Parameter θ is unknown (but fixed) for all machines.

Applications in marketing, health, etc.

Bandits

What drives the decision of what machine to play?

- $a_t \in \{1, \ldots, n\}$ is the chosen machine at time time t.
- $y_t \in \mathbb{R}$ is the reward after choosing a_t.

Policy:
Maps from history to a new choice a_t:

$$\pi : [(a_1, y_1), (a_2, y_2), \ldots, (a_{t-1}, y_{t-1})] \to a_t$$

Goal:
Find $\pi^\star$ that maximizes the cumulative (or other) reward:

$$\pi^\star = \arg\max_\pi \mathbb{E}_\pi \left[\sum_{t=t}^{T} y_t \right]$$

The belief state

Probabilistic representation of our 'knowledge' about the system

Knowledge can be represented in two ways:

- As the full history/dataset at time t:

$$\mathcal{D}_t = [(a_1, y_1), (a_2, y_2), \ldots, (a_{t-1}, y_{t-1})]$$

- As the belief (data + prior) computed used probability rules:

$$\mathcal{B}(\theta) = p(\theta|\mathcal{D}_t) \propto p(\theta)p(\mathcal{D}_t|\theta)$$

where $\theta = (\theta_1, \ldots, \theta_n)$ are the parameters of all the machines.

Example

Belief state in independent Gaussian bandits with fixed noise

$$\mathcal{B}(\theta) = p(\theta|\mathcal{D}_t) = \prod_{i=1}^{n} b_i(\mu_i|\mathcal{D}_t) = \prod_{i=1}^{n} \mathcal{N}(\mu_i|\bar{y}_i, \bar{s}_i)$$

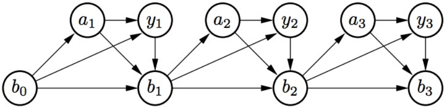
In this case the belief is <u>multivariate Gaussian</u>.

- Other beliefs are possible (Beta-binomial model).
- Gaussian belief $\rightarrow$ central role of Gaussian processes.

Value function and optimal belief planning

Usual terminology in RL, not so much in BO, BQ, etc.

Markov decision process (MDP), decisions affect rewards:



Value function, total reward under the optimal policy given $\mathcal{B}_{t-1}$:

$$
\begin{aligned}
V_{t-1}(\mathcal{B}_{t-1}(\theta)) &= \max_{\pi} \mathbb{E}_{\pi}\left[\sum_{t=t}^{T} y_t\right] \\
&= \max_{a_t} \int [y_t + V_t(\mathcal{B}_{t-1}(\theta; y_t, a_t))] p(y_t | a_t, \mathcal{B}_{t-1}) dy_t
\end{aligned}
$$

where $\mathcal{B}_{t-1}(\theta; y_t, a_t)$ is the updated belief given y_t and a_t.

Image source: Toussaint 2013, MLSS.

Notes on the value function

$$V_{t-1}(\mathcal{B}_{t-1}(\theta)) = \max_{a_t} \int [y_t + V_t(\mathcal{B}_{t-1}(\theta; y_t, a_t))] p(y_t | a_t, \mathcal{B}_{t-1}) dy_t$$

- y_t, reward of selecting a_t on the next step.
- $V_t(\mathcal{B}_{t-1}(\theta; y_t, a_t))$, future 'value' of have selected a_t.

Considerations:

- It tell us how to 'optimally optimize' our policy.
- Intractable, requires roll-out into the future.
- In practice: myopic approximation + domain specific belief.

Notes on the value function

$$V_{t-1}(\mathcal{B}_{t-1}(\theta)) = \max_{a_t} \int [y_t + V_t(\mathcal{B}_{t-1}(\theta; y_t, a_t))] p(y_t|a_t, \mathcal{B}_{t-1}) dy_t$$

- y_t, reward of selecting a_t on the next step.
- $V_t(\mathcal{B}_{t-1}(\theta; y_t, a_t))$, future 'value' of have selected a_t.

Considerations:

- It tell us how to 'optimally optimize' our policy.
- Intractable, requires roll-out into the future.
- In practice: myopic approximation + domain specific belief.

Myopic, 1-step look-ahead heuristics

Thompson sampling

Given $\mathcal{B}_{t-1}(\theta)$, use the following heuristic:

1. Sample from the Gaussians in each arm, $s_1, \ldots, s_n$.
2. Play arm $i(t) := \arg\max s_i$ and observe the reward y_t.
3. Update the belief.

Properties:

- Simple a fast heuristic.
- Possible to analyze its theoretical properties.

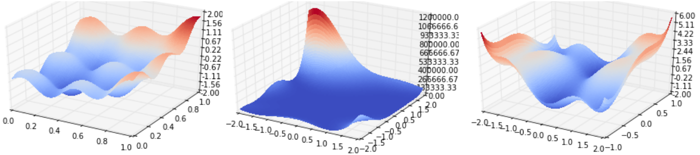
J. Honda A. Takemura. Optimality of Thompson Sampling for Gaussian Bandits Depends on Priors. AISTATS 2014.

Bayesian optimization

Problem definition

$f : \mathcal{X} \to \mathbb{R}$ where $\mathcal{X} \subseteq \mathbb{R}^D$ is 'well behaved' function is a bounded domain. Find

$$x_M = \arg\min_{x \in \mathcal{X}} f(x).$$



- f is explicitly unknown and multimodal.
- Evaluations of f may be perturbed by noise.
- Evaluations of f are expensive.

Applications to hyper-parameter optimization, robotics, intractable likelihoods, molecules design, etc.

Connection to bandits

- Infinitely-many arms with $y_t = f(x_t)$.
- 'Machines' are correlated.
- $\mathcal{D}_t = [(x_1, y_1), (x_2, y_2), \ldots, (x_{t-1}, y_{t-1})]$.
- Same reward as in the bandits case, $\sum_{t=t}^{T} y_t$.

Value function:

$$V_{t-1}(\mathcal{B}_{t-1}(f)) = \max_{a_t} \int [y_t + V_t(\mathcal{B}_{t-1}(f; x_t, y_t))] p(y_t | x_t, \mathcal{B}_{t-1}) dy_t$$

Belief model:

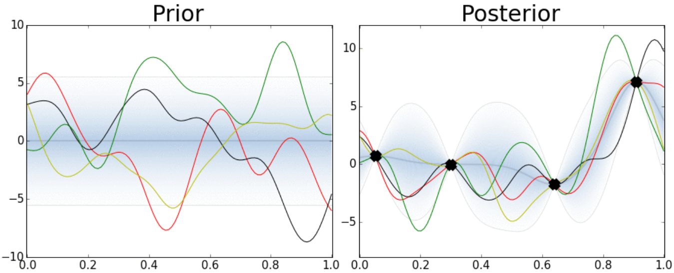
Multivariate Gaussian (n machines) $\rightarrow$ Gaussian process (f).

$$\mathcal{B}_t(f) \sim \mathcal{GP}(f : m, k)$$

Gaussian process as belief

Infinite-dimensional probability density, such that each linear finite-dimensional restriction is multivariate Gaussian.

$$f(x) \sim \mathcal{GP}(m(x), k(x, x'))$$



Posterior mean and variance can be computed in closed form:

- $m(x; \mathcal{D}) = k(x, X)(k(X, X) - \sigma^2 I)^{-1} y$
- $k(x, x'; \mathcal{D}) = k(x, x') - k(x, X)(k(X, X) - \sigma^2 I)^{-1} k(X, x')$.

C. Rasmussen and C. Williams. Gaussian Processes for Machine Learning. MIT Press.

Myopic, 1-step look-ahead heuristics

Theoretical results that link these heuristics to different reward functions exist

Lower Confidence bound:

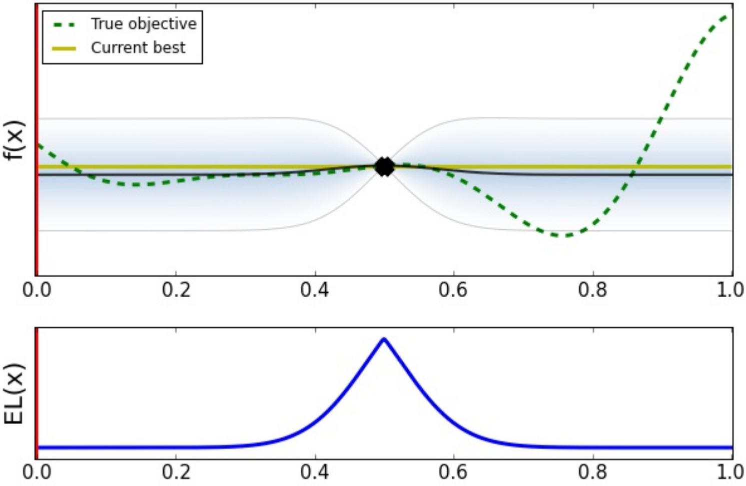
$$\alpha_{LCB}(x;\mathcal{D}) = -\mu(x;\mathcal{D}) + \beta_t \sigma(x;\mathcal{D})$$

Expected improvement:

$$\alpha_{EI}(x;\mathcal{D}) = \int_y \max(0, y_{best} - y) p(y|x;\mathcal{D}) dy$$

B. Shahriari, K. Swersky, Z. Wang, R. P. Adams and N. de Freitas. Taking the Human Out of the Loop: A Review of Bayesian Optimization. Proc. IEEE 104 (1) (January): 148-175

Using the expected loss to minimize a function



Exploration vs. exploitation

In each action we can do two things:

- **Exploit**: select a_t (or x_t) that maximizes reward $\mathbb{E}[y_{a_t}]$.
- **Explore**: select the action that minimizes the expected entropy of the belief, $\mathbb{E}[H(\mathcal{B}_t)]$.

Heuristics choose the balance between these terms.

Wait, how do we know how to optimally select this balance?

Optimally optimize! $\rightarrow$ Approximate the value function!

Exploration vs. exploitation

In each action we can do two things:

- **Exploit**: select a_t (or x_t) that maximizes reward $\mathbb{E}[y_{a_t}]$.
- **Explore**: select the action that minimizes the expected entropy of the belief, $\mathbb{E}[H(\mathcal{B}_t)]$.

<u>Heuristics choose the balance between these terms.</u>

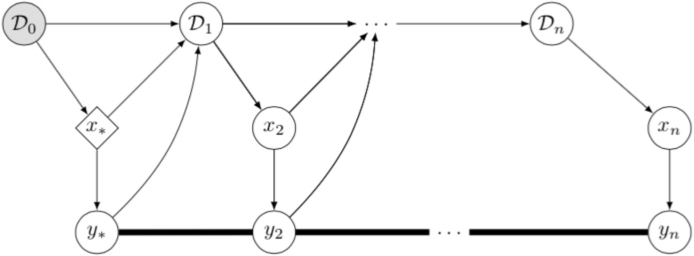
Wait, how do we know how to optimally select this balance?

Optimally optimize! $\rightarrow$ Approximate the value function!

Non-myopic Bayesian optimization

Approximating directly the value function

GLASSES: Global optimisation with Look-Ahead through Stochastic Simulation and Expected-loss Search



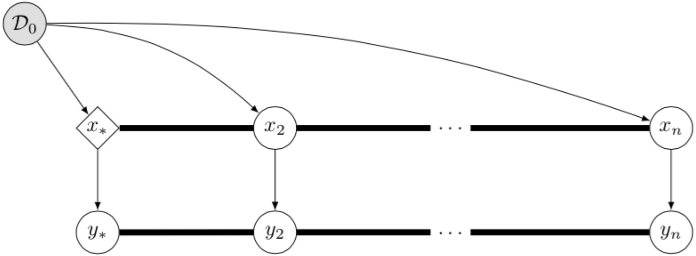
Approximate the computation of the value function for each action by sparsifying the MDP.

J. González, M. Osborne, N. Lawrence. GLASSES: Relieving the myopia of Bayesian optimisation. AISTATS 2016.
J. González, Z, Dai, P. Hennig, N. Lawrence. Batch Bayesian optimization via local penalization. AISTATS 2016.

Non-myopic Bayesian optimization

Approximating directly the value function

GLASSES: Global optimisation with Look-Ahead through Stochastic Simulation and Expected-loss Search



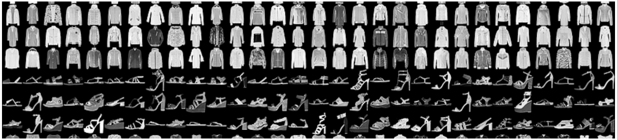
Approximate the computation of the value function for each action by sparsifying the MDP $\rightarrow$ Automatic exploration/exploitation.

J. González, M. Osborne, N. Lawrence. GLASSES: Relieving the myopia of Bayesian optimisation. AISTATS 2016.
J. González, Z, Dai, P. Hennig, N. Lawrence. Batch Bayesian optimization via local penalization. AISTATS 2016.

Active learning

Motivation

The goal in active learning is to 'learn' as fast as possible about about a function of interest f.



Examples

- Given a dataset of labeled and unlabeled images, select what image to label that improves the error of a given classifier.
- Experimental design.

Active learning

- Similar to BO but now we want to learn about f.
- $\mathcal{D}_t = [(x_1, y_1), (x_2, y_2), \dots, (x_{t-1}, y_{t-1})]$.
- Gaussian process belief: $\mathcal{B}_t(f) \sim \mathcal{GP}(f : m, k)$

Goal:
Minimize the entropy of the belief at the end of the search:

$$\pi^\star = \arg\max_\pi \mathbb{E}_\pi \left[-H(\mathcal{B}_T(f)) \right]$$

Value function

Value function, maximum entropy reduction

$$
\begin{aligned}
V_{t-1}(\mathcal{B}_{t-1}(f)) &= \max_{\pi} \mathbb{E}_{\pi}\left[-H(\mathcal{B}_T(f))\right] \\
&= \max_{x_t} \int V_t(\mathcal{B}_{t-1}(f; y_t, x_t)) p(y_t | x_t, \mathcal{B}_{t-1}) dy_t
\end{aligned}
$$

- For Gaussian belief it does not depend on the values of y_t.
- 'Pure exploration' compared to what is done in BO.
- Intractable objective.

Myopic, 1-step look-ahead heuristics

What to do in cases where the belief is a Gaussian process?

(Reminder!) In Bayesian optimization we balance:

- **Exploit**: select the action a_t that maximizes reward $\mathbb{E}[y_{a_t}]$.
- **Explore**: select the action that minimizes the expected entropy of the belief, $\mathbb{E}[H(\mathcal{B}_t)]$.

Myopic, 1-step look-ahead heuristics

What to do in cases where the belief is a Gaussian process?

In Active learning:

- ~~**Exploit**: select the action a_t that maximizes reward $\mathbb{E}[y_{a_t}]$.~~
- **Explore**: select the action that minimizes the expected entropy of the belief, $\mathbb{E}[H(\mathcal{B}_t)]$.

Myopic, 1-step look-ahead heuristics

What to do in cases where the belief is a Gaussian process?

In Active learning:

- ~~**Exploit**: select the action a_t that maximizes reward $\mathbb{E}[y_{a_t}]$.~~
- **Explore**: select the action that minimizes the expected entropy of the belief, $\mathbb{E}[H(\mathcal{B}_t)]$.

This is equivalent to maximize:

$$\alpha(x; \mathcal{D}) = k(x, x) - k(x, X)(k(X, X) - \sigma^2 I)^{-1} k(X, x)$$

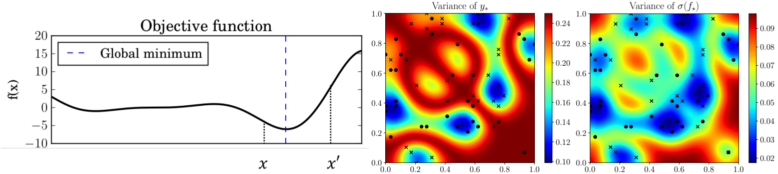
- Independent of the outputs.
- Nice connections with other techniques like determinantal point processes.

Active learning for preferential learning

Be careful with the uncertainty that you reduce...

- Find the minimum of a latent function $g(x), x \in \mathcal{X}$.
- The outcomes are binary and represent the preference.
- Classification model for duels:

$$p(y_\star = 1 | \mathcal{D}, [\mathbf{x}, \mathbf{x}'], \theta) = \int \sigma(f_\star) p(f_\star | \mathcal{D}, [\mathbf{x}_\star, \mathbf{x}'_\star], \theta) df_\star$$



$$\alpha_{DTS}(\mathbf{x}; \mathcal{D}) = \int \left(\sigma(f_\star) - E[\sigma(f_\star)]\right)^2 p(f_\star | \mathcal{D}, [\mathbf{x}, \mathbf{x}']) df_\star$$

J. Gonzalez, Z. Dai, A. Damianou, N. Lawrence. Preferential Bayesian Optimization. ICML 2017.

Bayesian Quadrature

Problem definition

In general, we want to estimate an integral

$$\mathcal{I}(f) = \int_{\mathcal{X}} f(x)p(x)dx.$$

We are interested in cases where:

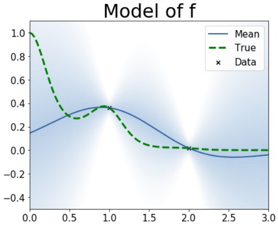
- The primitive of f is unknown.
- Evaluations of f are expensive.
- $p(x)$ is some measure of interest.

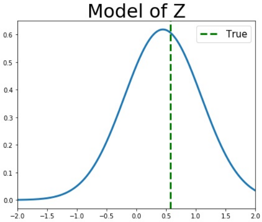
Applications in any operation in Bayesian inference.

Belief model

Indirect belief model over the integral via f

- Gaussian process on the integrand, $f \sim \mathcal{GP}(f : m, k)$.
- The belief over f induces a belief over $Z = \mathcal{I}(f)$.





$$p\left(\int_{\mathcal{X}} f(x)dx\right) = \mathcal{N}\left(Z; \int_{\mathcal{X}} m(x)dx, \int_{\mathcal{X}} k(x,x')dxdx'\right)$$

Bayesian Quadrature

- Similar to AL but now we want to learn about $\mathcal{I}(f)$.
- $\mathcal{D}_t = [(x_1, y_1), (x_2, y_2), \ldots, (x_{t-1}, y_{t-1})]$.
- Gaussian belief over the integral $\mathcal{B}(\mathcal{I}(f)) \sim \mathcal{N}(\mathcal{I}(f); m_{\mathcal{I}}, \sigma^2_{\mathcal{I}})$.

Goal:

$$\pi^\star = \arg\max_\pi \mathbb{E}_\pi \left[-H(\mathcal{B}_T(\mathcal{I}(f))) \right]$$

Value function:

$$V_{t-1}(\mathcal{B}_{t-1}(\mathcal{I}(f))) = \max_{x_t} \int V_t(\mathcal{B}_{t-1}(\mathcal{I}(f); y_t, x_t)) p(y_t | a_t, \mathcal{B}_{t-1}) dy_t$$

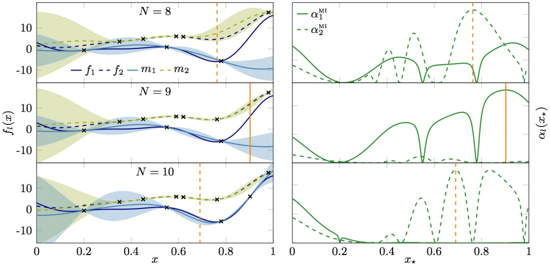
Myopic, 1-step look-ahead heuristics

Somehow similar to the active learning case

Vanilla approach, reduce variance about $\mathcal{I}(f)$.

$$\alpha(x;\mathcal{D}) = \mathbb{V}ar(\mathcal{I}(f)|\mathcal{D}) - \mathbb{E}_{p(y|x,\mathcal{D})}\left[\mathbb{V}ar(\mathcal{I}(f)|\mathcal{D}\cup\{x,y\})|\mathcal{D},x\right]$$

Changing the belief: Active multi-fidelity Bayesian quadrature.



A. Gessner, J. González and M. Mahsereci. On acquisition functions for active multi-source Bayesian quadrature. NeurIPS Workshop in Bayesian non-parametrics, 2018.

Model Based Reinforcement Learning

Motivation

RL is a slightly different beast...

- An agent makes decisions $a_t \in \mathcal{A}$ to optimize some reward.
- All previous problems: static environment.
- In RL: the environment changes, there is an <u>state</u>.
- <u>Actions influence the state</u>, s.

Comparison with Bayesian optimization

- BO: finds a solution (vector) that optimizes the function.
- RL: learns an optimal function that outputs a 'best' action for every possible state.

Reinforcement learning

Elements:

- Initial state distribution, $p(s_0)$.
- State transition probabilities: $p(s'|s,a)$.
- Reward probabilities: $p(y|s,a)$.
- Policy $\pi : s \rightarrow a$.

Goal: maximize the reward:

$$R = \sum_{t=t}^{T} \gamma^t y_t$$

.

Value function (written in terms of the state):

$$V_{t-1}(s) = \max_a [\mathbb{E}[y|s,a] + \gamma \sum_{s'} p(s'|s,a) V_t(s')]$$

Belief models in Reinforcement learning

Knowledge is given as:

$$\mathcal{D}_t = [(s_1, a_1, y_1), (s_2, a_2, y_2), \ldots, (s_{t-1}, a_{t-1}, y_{t-1}), s_t]$$

Belief model over the system dynamics:

- Use a GP to model model $p(s_{t+1}|s_t, a_t)$.
- PILCO: probabilistic dynamics model for long term planning.

Belief model over the reward, use some parametric policy $\pi(\theta)$

- Use a GP to model model $p(R|\theta)$.
- Bayesian optimization for reinforcement learning!

M. Deisenroth, C. Rasmussen. PILCO: A Model-Based and Data-Efficient Approach to Policy Search. ICML 2011
A. Wilson, A. Fern, P. Tadepalli. Using Trajectory Data to Improve Bayesian Optimization for Reinforcement Learning. JMLR, 2014.

Summary and final connections

Summary and final connections

Bandits, Bayesian optimization, Bayesian quadrature, Active learning and Model based RL

Method	Action set	History	Reward	Belief
Bandits	$a_i \in \{1, \dots, n\}$	$\{(a_i, y_i)\}_{i=1}^{t-1}$	$\sum_{t=t}^{T} y_t$	$\mathcal{N}(\theta; \mu, \sigma^2)$
Bayesian Optimization	$x \in \mathcal{X} \subseteq \mathbb{R}^D$	$\{(x_i, y_i)\}_{i=1}^{t-1}$	$\sum_{t=t}^{T} y_t$	$\mathcal{GP}(f; \mu, K)$
Active Learning	$x \in \mathcal{X} \subseteq \mathbb{R}^D$	$\{(x_i, y_i)\}_{i=1}^{t-1}$	$-H(\mathcal{B}(f))$	$\mathcal{GP}(f; \mu, K)$
Bayesian Quadrature	$x \in \mathcal{X} \subseteq \mathbb{R}^D$	$\{(x_i, y_i)\}_{i=1}^{t-1}$	$-H(\mathcal{B}(\mathcal{I}(f)))$	$\mathcal{N}(\mathcal{I}(f); \mu, \sigma^2)$
Reinforcement Learning	$a_i \in \mathcal{A}$	$\{(s_i, a_i, y_i)\}_{i=1}^{t-1}$	$\sum_{t=t}^{T} \gamma^t y_t$	$\mathcal{GP}(s; \mu, K)$

Method	Value function (given the reward)	Heuristic(s)
Bandits	$V_{t-1}(\mathcal{B}_{t-1}) = \max_{a_t} \int [y_t + V_t(\mathcal{B}_{t-1}(\theta; y_t, a_t))] dp y_t$	UCB, TS
Bayesian Optimization	$V_{t-1}(\mathcal{B}_{t-1}) = \max_{x_t} \int [y_t + V_t(\mathcal{B}_{t-1}(f; y_t, x_t))] dp y_t$	EI, MPI, UCB
Active Learning	$V_{t-1}(\mathcal{B}_{t-1}) = \max_{x_t} \int [V_t(\mathcal{B}_{t-1}(f; y_t, x_t))] dp y_t$	Variance reduction
Bayesian Quadrature	$V_{t-1}(\mathcal{B}_{t-1}) = \max_{x_t} \int [V_t(\mathcal{B}_{t-1}(\mathcal{I}(f); y_t, x_t))] dp y_t$	Integral variance reduction
Reinforcement Learning	$V_{t-1}(s) = \max_a [\mathbb{E}[y \mid s, a] + \gamma \sum_{s'} p(s' \mid s, a) V_t(s')]$	PILCO, BO, others

Summary and final connections

Bandits, Bayesian optimization, Bayesian quadrature, Active learning and Model based RL

- They are all sequential decision processes.
- The belief is key to reason about optimal policies.
- Gaussian process are a common and flexible model the belief.
- The decisions influence the rewards in Bandits, BO, AL and BQ and in RL decisions also influence the state.
- An optimal although often intractable solution usually exist but in practice tractable myopic heuristics are used.
- Heuristics show an exploration/exploitation trade off that is automatic when the value function is approximated.

Recipe

To make your own decision making method

1. Define the reward.
2. Define the resources.
3. (X) Build a model of your belief.
4. Write down the optimal policy.
5. (X) Define a heuristic that balances the use of your resources and the approximation to the optimal policy.

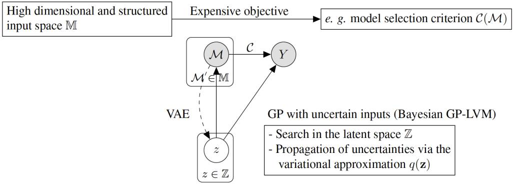
(X) = key!

The best methods are always:

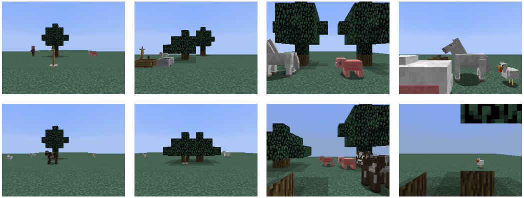
- Use **domain knowledge** to define the belief.
- Define a policy that makes use of the **properties of the belief**.

Example of the recipe

Semi-supervised Bayesian optimization



- Optimization on context free grammar.
- Learn a probabilistic manifold using a VAE (structured belief).
- Propagation of uncertainty to the search (tailored heuristic).
- Application to image understanding.



X. Lu, J. Gonzalez, Z. Dai and N. Lawrence. Structured Variationally Auto-encoded optimization. ICML 2018.

Emukit

Python platform for quick prototyping of decision making methods

- Probabilistic programing (DP) provides a framework to automate the constructions of probabilistic models.
- Emukit provides a framework to plug-and-play components of several decision making methods.
- Separates model and decision. You can use your own modeling framework, TensorFlow, MXnet, GPy, etc.

Many thanks to!

Neil Lawrence, Zhenwen Dai, Andreas Damianou, Xiaoyu Lu, Mark Pullin, Andrei Paleyes, Maren Mahsereci, Alexandra Gessner.