263-5210-00: Probabilistic Artificial Intelligence

Conceptual Summary

Swiss Federal Institute of Technology Zurich (ETH Zürich)

Last Edit Date: 01/22/2025

Disclaimer and Term of Use:

1. We do not guarantee the accuracy and completeness of the summary content. Some of the course material may not be included, and some of the content in the summary may not be correct. You should use this file properly and legally. We are not responsible for any results from using this file.
2. This personal note is adapted (derived) from the slides and/or codes from *Professor Andreas Krause*. Please contact us to delete this file if you think your rights have been violated.
3. This work is licensed under a Creative Commons Attribution 4.0 International License.

1. Preliminaries

I. Useful Probability Density Functions

Gaussian: $p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$

Beta: $p(x) = (1/B(r,s))x^{r-1}(1-x)^{s-1}$

Gamma: $p(x) = ((\lambda t)^{r-1}/\Gamma(x))\lambda\, e^{-\lambda t}$

II. Expectation & Variance Matrix

$E[\boldsymbol{AX}+b] = \boldsymbol{A}E[\boldsymbol{X}] + b$; $E[\boldsymbol{X}+\boldsymbol{Y}] = E[\boldsymbol{X}] + E[\boldsymbol{Y}]$; $\mathrm{Var}(\boldsymbol{AX}+b) = \boldsymbol{A}\mathrm{Var}(\boldsymbol{X})\boldsymbol{A}^T + b$; $\mathrm{Var}(\boldsymbol{X}+\boldsymbol{Y}) = \mathrm{Var}(\boldsymbol{X}) + \mathrm{Var}(\boldsymbol{Y}) + 2\mathrm{Cov}(\boldsymbol{X},\boldsymbol{Y})$ if independent $\mathrm{Cov} = 0$; $E[\boldsymbol{X}\boldsymbol{T}^T] = E[\boldsymbol{X}]E[\boldsymbol{Y}]^T$ if independent; $E[g(\boldsymbol{X})] = \int g(x)f(x)dx$ if $\boldsymbol{X} \in R$, $g: R^d \to R$; $E[E[\boldsymbol{X}|\boldsymbol{Y}]] = E[\boldsymbol{X}]$; $\mathrm{Var}(\boldsymbol{X}) = E[\mathrm{Var}(\boldsymbol{X}|\boldsymbol{Y})] + \mathrm{Var}(E[\boldsymbol{X}|\boldsymbol{Y}])$; $E(f(x)) \le f(E(x))$ Jensen's inequality f convex.

IV. Change of Variable

g is differentiable and invertible. $\boldsymbol{Y} = g(\boldsymbol{X})$ we have $p_{\boldsymbol{Y}}(y) = p_{\boldsymbol{X}}(g^{-1}(y)) \cdot \left|\det(Dg^{-1}(y))\right|$ $Dg^{-1}(y)$ is the Jacobian of g^{-1} evaluated at y.

V. Bayes' Rule

$p(\boldsymbol{x}|\boldsymbol{y}) = \frac{p(y|\boldsymbol{x})p(\boldsymbol{x})}{p(y)}$; $p(\boldsymbol{y}) = \sum_i p(\boldsymbol{y}|x_i)p(x_i)$ marginal likelihood, $p(\boldsymbol{x}|\boldsymbol{y})$ posterior, $p(\boldsymbol{x})$ prior, $p(\boldsymbol{y}|\boldsymbol{x})$ conditional likelihood.

VI. Gaussian Random Variables

$\boldsymbol{Y} = \boldsymbol{X} + \boldsymbol{X}' \sim N(\mu_Y + \mu_Y', \boldsymbol{\Sigma}_{VV} + \boldsymbol{\Sigma}_{VV}')$

$\boldsymbol{Y} = \boldsymbol{MX} \sim N(\boldsymbol{M}\mu_V, \boldsymbol{M}\boldsymbol{\Sigma}_{VV}\boldsymbol{M}^T)$

$\boldsymbol{X_A}|\boldsymbol{X_B} \sim N(\mu_A + \boldsymbol{\Sigma}_{AB}\boldsymbol{\Sigma}_{BB}^{-1}(x_B - \mu_B), \boldsymbol{\Sigma}_{AA} - \boldsymbol{\Sigma}_{AB}\boldsymbol{\Sigma}_{BB}^{-1}\boldsymbol{\Sigma}_{BA})$

* Under Gaussian, affine transformations are also Gaussian; uncorrelated ↔ independent.

VII. MLE in Linear Regression

$\widehat{\boldsymbol{w}}_{MLE} = \mathrm{argmax}_w \sum_{i=1}^n \log p(y_i|\boldsymbol{x}_i, \boldsymbol{w}) = \mathrm{argmin}_w \sum_{i=1}^n (y_i - \boldsymbol{w}^T x_i)^2 = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{Y}$,

$\boldsymbol{w} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, $\boldsymbol{\Sigma} = \left(\sigma_n^{-2}\boldsymbol{X}^T\boldsymbol{X} + \sigma_p^{-2}\boldsymbol{I}\right)^{-1}$, $\boldsymbol{\mu} = \sigma_n^{-2}\boldsymbol{\Sigma}\boldsymbol{X}^T\boldsymbol{y}$. $\widehat{\boldsymbol{w}}_{ridge} = (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^T\boldsymbol{Y}$.

L1: $\widehat{\boldsymbol{w}}_{MLE} + \lambda\|\boldsymbol{w}\|_1$; L2: $\widehat{\boldsymbol{w}}_{MLE} + \lambda\|\boldsymbol{w}\|_2^2$.

2. Bayesian Linear Regression

I. Maximum a Posteriori Estimation

$\widehat{\boldsymbol{w}}_{MAP} = \mathrm{argmax}_{\mathrm{w}} \log p(y_i|\boldsymbol{x}_i, \boldsymbol{w}) + \log p(\boldsymbol{w})$, where $\log p(\boldsymbol{w})$ is the regularization term.

$\widehat{\boldsymbol{w}}_{MAP,L2} = \mathrm{argmin}_{\mathrm{w}}\|\boldsymbol{y} - \boldsymbol{Xw}\|_2^2 + \frac{\sigma_n^2}{\sigma_p^2}\|\boldsymbol{w}\|_2^2$

$\widehat{\boldsymbol{w}}_{MAP,L1} = \mathrm{argmin}_{\mathrm{w}}\|\boldsymbol{y} - \boldsymbol{Xw}\|_2^2 + \frac{\sigma_n^2}{l}\|\boldsymbol{w}\|_1$

Predict *test* $f^*|\boldsymbol{x}^* \to N(\boldsymbol{\mu}^T\boldsymbol{x}^*, \boldsymbol{x}^{*T}\boldsymbol{\Sigma}\boldsymbol{x}^*)$

Predict *new* $y^*|\boldsymbol{x}^* \to N(\boldsymbol{\mu}^T\boldsymbol{x}^*, \boldsymbol{x}^{*T}\boldsymbol{\Sigma}\boldsymbol{x}^* + \sigma_n^2)$

$p^t(w) = N(w; \mu^t, \Sigma^t)$ recursive probabilistic inference with $O(d^2)$ parameters, time-independent $O(d)$ memory, $O(nd^2)$ round.

II. Aleatoric & Epistemic Uncertainty

$\mathrm{Var}(y^*|x^*) = E_\theta[\mathrm{Var}_{y^*}(y^*|x^*, \theta)] + \mathrm{Var}_\theta(E_{y^*}[y^*|x^*, \theta])$, where the first term is aleatoric uncertainty (data noise, irreducible), the second term is epistemic uncertainty (model noise, due to lack of data).

III. Non-linearity & Kernels

Define $\boldsymbol{\Phi} = \phi(\boldsymbol{X})$, kernel, with a gaussian prior we have $\boldsymbol{f}|\boldsymbol{X} \sim N(\boldsymbol{\Phi}E[\boldsymbol{w}], \boldsymbol{\Phi}\mathrm{Var}(\boldsymbol{w})\boldsymbol{\Phi}^T) = N(0, \sigma_p^2\boldsymbol{\Phi}\boldsymbol{\Phi}^T) = N(0, \boldsymbol{K})$. The kernel function is $k(\boldsymbol{x}, \boldsymbol{x}') = \sigma_p^2\phi(\boldsymbol{x})^T\phi(\boldsymbol{x}') = \mathrm{Cov}(f(\boldsymbol{x}), f(\boldsymbol{x}'))$. A kernel matrix looks like $\boldsymbol{K} = \begin{bmatrix} k(\boldsymbol{x}_1,\boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_1,\boldsymbol{x}_n) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_n,\boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_n,\boldsymbol{x}_n) \end{bmatrix}$. Predict with kernels: test point $\widetilde{\boldsymbol{f}}|\boldsymbol{X}, \boldsymbol{x}^* \sim N(0, \widetilde{\boldsymbol{K}})$; new $\widetilde{\boldsymbol{y}}|\boldsymbol{X}, \boldsymbol{x}^* \sim N(0, \widetilde{\boldsymbol{K}} + \sigma_n^2\boldsymbol{I})$, where $\widetilde{\boldsymbol{K}} = \sigma_p^2\widetilde{\boldsymbol{\Phi}}\widetilde{\boldsymbol{\Phi}}^T$, and $\widetilde{\boldsymbol{\Phi}} = [\boldsymbol{\Phi}\ \boldsymbol{\Phi}(\boldsymbol{x}^*)^T]^T$.

3. Gaussian Processes

I. Basics & Definition

A Gaussian process is an **infinite set of random variables** such that any **finite number of them a jointly Gaussian**. Suppose we have $\mu: \chi \to R$, $k: \chi \times \chi \to R$ such that $A = \{x_1, \cdots, x_m\} \subseteq \chi$, have $\boldsymbol{f}_A = [f_{x_1} \cdots f_{x_m}]^T \sim N(\boldsymbol{\mu}_A, \boldsymbol{K}_{AA})$, where $\boldsymbol{\mu}_A = [\mu(x_1) \cdots \mu(x_m)]^T$, $\boldsymbol{K}_{AA} = \begin{bmatrix} k(\boldsymbol{x}_1,\boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_1,\boldsymbol{x}_m) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_m,\boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_m,\boldsymbol{x}_m) \end{bmatrix}$. $f \sim GP(\mu, k)$.

Given a mean function, covariance function, and homoscedastic assumption, we have that $\boldsymbol{y}^*|\boldsymbol{x}^*, \mu, k \sim N(\mu(\boldsymbol{x}^*), k(\boldsymbol{x}^*, \boldsymbol{x}^*) + \sigma_n^2)$.

II. Prediction & Inference

Predict with test x^*, $\begin{bmatrix} \boldsymbol{y} \\ f^* \end{bmatrix}|\boldsymbol{x}^*, \boldsymbol{x}_{1:n} \sim N(\widetilde{\boldsymbol{\mu}}, \widetilde{\boldsymbol{K}})$,

$\widetilde{\boldsymbol{\mu}} = \begin{bmatrix} \boldsymbol{\mu}_A \\ \mu(\boldsymbol{x}^*) \end{bmatrix}$, $\widetilde{\boldsymbol{K}} = \begin{bmatrix} \boldsymbol{K}_{AA} + \sigma_n^2\boldsymbol{I} & \boldsymbol{k}_{x^*,A} \\ \boldsymbol{k}_{x^*,A}^T & k(\boldsymbol{x}^*, \boldsymbol{x}^*) \end{bmatrix}$, $\boldsymbol{k}_{x,A} = \begin{bmatrix} k(\boldsymbol{x}, \boldsymbol{x}_1) \\ \vdots \\ k(\boldsymbol{x}, \boldsymbol{x}_n) \end{bmatrix}$.

GP posterior: $f|\boldsymbol{x}_{1:n}, y_{1:n} \sim GP(\mu', k')$, where $\mu'(\boldsymbol{x}) = \mu(\boldsymbol{x}) + \boldsymbol{k}_{x,A}^T(\boldsymbol{K}_{AA} + \sigma_n^2\boldsymbol{I})^{-1}(\boldsymbol{y}_A - \boldsymbol{\mu}_A)$ and $k'(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x}, \boldsymbol{x}') - \boldsymbol{k}_{x,A}^T(\boldsymbol{K}_{AA} + \sigma_n^2\boldsymbol{I})^{-1}\boldsymbol{k}_{x',A}$. So $\boldsymbol{y}^*|\boldsymbol{x}^*, \boldsymbol{x}_{1:n}, y_{1:n} \sim N(\mu^*, k^* + \sigma_n^2)$ is the predictive posterior at test point x^*.

* Not interested to get full predictive posterior distribution, but to sample, but with $O(n^3)$.

III. Properties of Kernels

Symmetry $k(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x}', \boldsymbol{x})$ for $\boldsymbol{x}, \boldsymbol{x}' \in \boldsymbol{\chi}$; $\boldsymbol{K_{AA}}$ is **positive semi-definite** for $\boldsymbol{A} \subseteq \boldsymbol{\chi}$; **stationary** if $\tilde{k}(\boldsymbol{x} - \boldsymbol{x}') = k(\boldsymbol{x}, \boldsymbol{x}')$; **isotropic** if $\tilde{k}\left(\|\boldsymbol{x} - \boldsymbol{x}'\|_2\right) = k(\boldsymbol{x}, \boldsymbol{x}')$.

* Isotropic implies stationary.

Compositions: $k(\boldsymbol{x}, \boldsymbol{x}') = k_1(\boldsymbol{x}, \boldsymbol{x}') + k_2(\boldsymbol{x}, \boldsymbol{x}')$; $k(\boldsymbol{x}, \boldsymbol{x}') = k_1(\boldsymbol{x}, \boldsymbol{x}') \cdot k_2(\boldsymbol{x}, \boldsymbol{x}')$; $k(\boldsymbol{x}, \boldsymbol{x}') = c \cdot k_1(\boldsymbol{x}, \boldsymbol{x}')$ for a scale $c > 0$; $k(\boldsymbol{x}, \boldsymbol{x}') = f(k_1(\boldsymbol{x}, \boldsymbol{x}'))$ for polynomial f with positive coefficients or $f = \exp\cdot$.

* If $f_1 \sim GP(\mu_1, k_1)$, $f_2 \sim GP(\mu_2, k_2)$ then $f = f_1 + f_2 \sim GP(\mu_1 + \mu_2, k_1 + k_2)$.

IV. Useful Kernels

Linear: $k(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\phi}) = \boldsymbol{\phi}(\boldsymbol{x})^T\phi(\boldsymbol{x}')$, a GP with linear kernel is just Bayesian linear regression.

RBF: $k(\boldsymbol{x}, \boldsymbol{x}'; l) = \exp\left(-\|\boldsymbol{x} - \boldsymbol{x}'\|_2^2/2l^2\right)$

Laplace: $k(\boldsymbol{x}, \boldsymbol{x}'; l) = \exp\left(-\|\boldsymbol{x} - \boldsymbol{x}'\|_2/l\right)$, less smooth compare to RBF (gaussian) kernel.

Matern: $k(\boldsymbol{x}, \boldsymbol{x}'; \nu, l) = \frac{2^{1-\nu}}{\Gamma(\nu)}\left(\frac{\sqrt{2\nu}\|\boldsymbol{x} - \boldsymbol{x}'\|_2}{l}\right)^\nu \cdot K_\nu\left(\frac{\sqrt{2\nu}\|\boldsymbol{x} - \boldsymbol{x}'\|_2}{l}\right)$, Γ gamma function, K_ν Bessel function of second kind, $\nu = 1/2$ get Laplace, $\nu \to \infty$ get RBF.

* l is length scale, larger l smoother function.

V. Model Selection

Validation set optimization collapses the uncertainty in f into a point estimate.

Maximizing the marginal likelihood avoids it. $\hat{\theta}_{MLE} = \mathrm{argmax}_\theta \int p(y_{1:n}|x_{1:n}, f, \theta)p(f|\theta)df$

VI. Approximation to Avoid $O(n^3)$

1. Local Method

When sampling at $\boldsymbol{x}$ only condition on the samples $\boldsymbol{x}'$ that are close, i.e. $|k(\boldsymbol{x}, \boldsymbol{x}')| \ge \tau$ for $\tau > 0$, instead of all samples. If τ is chosen too large, samples become essentially independent.

2. Kernel Approximation

Construct a low dimensional feature map $\phi: R^d \to R^m$ that approximates the kernel $k(\boldsymbol{x}, \boldsymbol{x}') \approx \phi(\boldsymbol{x})^T\phi(\boldsymbol{x}')$. Then apply Bayesian linear regression getting $O(nm^2 + m^3)$.

Achieved by *Fourier transformation*: a stationary kernel k can be interpreted as a function in one variable, and has an associated Fourier transform denoted by $p(\boldsymbol{\omega})$: $k(\boldsymbol{x} - \boldsymbol{x}') = \int p(\boldsymbol{\omega})e^{i\boldsymbol{\omega}^T(\boldsymbol{x}-\boldsymbol{x}')}d\omega$. **Bochner's theorem**: A continuous stationary kernel on R^d is positive definite if and only if its Fourier transform $p(\boldsymbol{\omega})$ is *non-negative*. For gaussian kernel $p(\boldsymbol{\omega}) = (2l^2\pi)^{d/2}\exp\left(-l^2\frac{\|\boldsymbol{\omega}\|_2^2}{2}\right)$. We can interpret a kernel as an expectation $k(\boldsymbol{x} - \boldsymbol{x}') = E\left[e^{i\boldsymbol{\omega}^T(\boldsymbol{x}-\boldsymbol{x}')}\right] = E[\cos(\boldsymbol{\omega}^T\boldsymbol{x} - \boldsymbol{\omega}^T\boldsymbol{x}') + i\sin(\boldsymbol{\omega}^T\boldsymbol{x} - \boldsymbol{\omega}^T\boldsymbol{x}')] = \boldsymbol{z}(\boldsymbol{x})^T\boldsymbol{z}(\boldsymbol{x}')$, $\boldsymbol{z}(\boldsymbol{x}) = \frac{1}{\sqrt{m}}\left[z_{\omega^{(1)},b^{(1)}}(\boldsymbol{x}), \cdots, z_{\omega^{(m)},b^{(m)}}(\boldsymbol{x})\right]^T$, here $\omega^{(i)} \sim^{iid} p$ and $b^{(i)} \sim^{iid} Unif([0, 2\pi])$.

4. Variational Inference

I. Goal & Basics

Approximate the true (**intractable**) posterior with a **simpler posterior** that is easy to sample: $p(\boldsymbol{\theta}|\boldsymbol{x}_{1:n}, y_{1:n}) = \frac{1}{Z}p(\boldsymbol{\theta}, y_{1:n}|\boldsymbol{x}_{1:n}) \approx q(\theta|\lambda) = q_\lambda(\theta)$, where Z is intractable here.

II. Laplace Approximation

The idea is to use a Gaussian approximation (a second-order Taylor approximation) of the posterior distribution around its **mode**. The Laplace approximation q of p is $q(\boldsymbol{\theta}) = N(\boldsymbol{\theta}; \widehat{\boldsymbol{\theta}}, \boldsymbol{\Lambda}^{-1}) = \exp\left(\widehat{\psi}(\boldsymbol{\theta})\right) = \exp\left(\log N\left(\boldsymbol{\theta}; \widehat{\boldsymbol{\theta}}, -\boldsymbol{H}_\psi(\widehat{\boldsymbol{\theta}})^{-1}\right)\right)$, where $\boldsymbol{\Lambda} = -\boldsymbol{H}_\psi(\widehat{\boldsymbol{\theta}}) = -\boldsymbol{H}_\theta \log p(\theta|x_{1:n}, y_{1:n})|_{\theta=\widehat{\theta}}$, the covariance matrix $\boldsymbol{\Lambda}^{-1}$ needs to be symmetric and positive semi-definite.

III. Predictions with a Variational Posterior

$p(y^*|\boldsymbol{x}^*, \boldsymbol{x}_{1:n}, y_{1:n}) \approx \int p(y^*|\boldsymbol{x}^*, \boldsymbol{\theta})q_\lambda(\boldsymbol{\theta})d\boldsymbol{\theta}$ if we view it as an expectation over q_λ and approximate via Monte Carlo sampling $E_{\theta\sim q_\lambda}[p(y^*|\boldsymbol{x}^*, \boldsymbol{\theta})] \approx \frac{1}{m}\sum_{j=1}^m p(y^*|\boldsymbol{x}^*, \boldsymbol{\theta}_j)$, where $\boldsymbol{\theta}_j \sim^{iid} q_\lambda$.

IV. Information Theory

Surprise of an event with probability u: $S[u] = -\log u$.

The **entropy** of a distribution p is the average surprise of samples from p: $H[p] = E_{x\sim p}[S[p(x)]] = E_{x\sim p}[-\log p(x)]$, if p discrete $H[p] = -\sum p(x)\log_2 p(x)$, if p continuous $H[p] = -\int p(x)\log_2 p(x)\,dx$.

Gaussian: $H[N(\mu, \Sigma)] = \frac{1}{2}\log((2\pi e)^d \det\Sigma)$.

The **cross-entropy** of q relative to p is $H[p||q] = E_{x\sim p}[S[q(x)]] = E_{x\sim p}[-\log q(x)]$.

Kullback-Leibler (KL) divergence measures how different the distribution q is from p: $KL(p||q) = H[p||q] - H[p] = E_{\theta\sim p}\left[\log\frac{p(\theta)}{q(\theta)}\right]$

Properties of KL divergence: $KL(p||q) \ge 0$; $KL(p||q) = 0$ if and only if $p = q$ almost surely; there exists p, q such that $KL(p||q) \ne KL(q||q)$; $H[p||q] = H[p] + KL(p||q) \ge H[p]$.

Forward KL: $q_1^* = \mathrm{argmin}_{\mathrm{q}} KL(p||q)$.

Reverse KL: $q_2^* = \mathrm{argmin}_{\mathrm{q}} KL(q||p)$ greedily select mode and **underestimate** variance.

Useful KL divergences:

$KL(\mathrm{Bern}(p)||\mathrm{Bern}(q)) = p\log\frac{p}{q} + (1-p)\log\frac{1-p}{1-q}$

$KL(N(\mu_p, \Sigma_p)||N(\mu_q, \Sigma_q)) = \frac{1}{2}\left(tr(\Sigma_q^{-1}\Sigma_p) + (\mu_p - \mu_q)^T\Sigma_q^{-1}(\mu_p - \mu_q) - d + \log\frac{\det(\Sigma_q)}{\det(\Sigma_p)}\right)$

V. Evidence Lower Bound

$KL(q||p(\cdot|x_{1:n}, y_{1:n})) = \log p(y_{1:n}|x_{1:n}) - E_{\theta\sim q}[\log p(y_{1:n}, \theta|x_{1:n})] - H[q]$, where the latter two terms can be denoted as $-L(q, p; D_n)$, and (without the negative) it is called the evidence lower bound (ELBO) given the data $D_n = \{(x_i, y_i)\}_{i=1}^n$. Other forms:

$$\begin{aligned} L(q, p; \mathcal{D}_n) &= \mathbb{E}_{\theta\sim q}[\log p(y_{1:n}, \boldsymbol{\theta} \mid x_{1:n}) - \log q(\boldsymbol{\theta})] \\ &= \mathbb{E}_{\theta\sim q}[\log p(y_{1:n} \mid x_{1:n}, \boldsymbol{\theta}) + \log p(\boldsymbol{\theta}) - \log q(\boldsymbol{\theta})] \\ &= \underbrace{\mathbb{E}_{\theta\sim q}[\log p(y_{1:n} \mid x_{1:n}, \boldsymbol{\theta})]}_{\text{log-likelihood}} \underbrace{- \mathrm{KL}(q\|p(\cdot))}_{\text{proximity to prior}}. \end{aligned}$$

VI. Reparameterization Trick

Given a random variable $\epsilon \sim \phi$ (which is independent of λ) and given a differentiable and invertible function $g: R^d \to R^d$. We let $\theta = g(\epsilon; \lambda)$. Then $q_\lambda(\theta) = \phi(\epsilon) \cdot \left|\det\left(D_\epsilon g(\epsilon; \lambda)\right)\right|^{-1}$, $E_{\theta\sim q_\lambda}[f(\theta)] = E_{\epsilon\sim\phi}[f(g(\epsilon; \lambda))]$, nice function $f: R^d \to R^e$.

ELOB: $\nabla_\lambda E_{\theta\sim q_\lambda}[f(\theta)] = E_{\epsilon\sim\phi}[\nabla_\lambda f(g(\epsilon; \lambda))]$

Gaussian: $q_\lambda(\theta) = N(\theta; \mu, \Sigma)$, $\theta = g(\epsilon, \lambda) = \Sigma^{1/2}\epsilon + \mu$, $\epsilon = g^{-1}(\theta; \lambda) = \Sigma^{-1/2}(\theta - \mu)$, $\phi(\epsilon) = q_\lambda(\theta)|\det(\Sigma^{1/2})|$.

5. Markov Chain Monte Carlo Methods

I. Markov Chains

A Markov Chain over $S = \{0, \cdots, n-1\}$ is a sequence $(X_t)_{t\in N_0} \in S$, such that the Markov property $X_{t+1} \perp X_{0:t-1}|X_t$ is satisfied. The Markov property states that future behaviors is independent of past states given present state.

A **time-homogeneous** Markov Chain can be characterized by a transition function $p(x'|x) = P(X_{t+1} = x'|X_t = x)$, the transition matrix $\boldsymbol{P} = \begin{bmatrix} p(x_1|x_1) & \cdots & p(x_n, x_1) \\ \vdots & \ddots & \cdots \\ p(x_1|x_n) & \cdots & p(x_n|x_n) \end{bmatrix}$ where each row sums to 1 (stochastic). The states of a Markov Chain at t is a probability distribution $\boldsymbol{q}_t \in R^{1\times|S|}$, then we can have $\boldsymbol{q}_{t+k} = \boldsymbol{q}_t\boldsymbol{P}^k$.

II. Properties of Markov Chains

Stationary: A distribution π is stationary with respect to the transition function p if and only if $\pi(x) = \sum_{x'\in S} p(x|x')\pi(x')$, can be written as $\boldsymbol{\pi} = \boldsymbol{\pi}\boldsymbol{P}$ with the transition matrix $\boldsymbol{P}$.

Convergence: A Markov Chain converges to its *unique* stationary distribution if and only if we have $\lim_{t\to\infty} q_t = \pi$ regardless of initial q_0.

Ergodicity: A Markov Chain is ergodic if and only if there exists a $t \in N_0$ such that for any $x, x' \in S$ we have $p^{(t)}(x'|x) > 0$, where $p^{(t)}(x'|x)$ is the probability to reach x' from x in exactly t steps. Equivalently, for some $t \in N_0$ all entries of $\boldsymbol{P^t}$ are positive, or the Markov Chain is irreducible and aperiodic. To make a MC with $\boldsymbol{P}' = 1/2\ \boldsymbol{P} + 1/2\boldsymbol{I}$ ergodic, take $\boldsymbol{\pi}$ as a stationary distribution of $\boldsymbol{P}$, we have $\boldsymbol{\pi}\boldsymbol{P}' = 1/2\ \boldsymbol{\pi}\boldsymbol{P} + 1/2\ \boldsymbol{\pi}\boldsymbol{I} = \boldsymbol{\pi}$.

III. Detailed Balance Equation

If $\pi(x)p(x'|x) = \pi(x')p(x|x')$, a MC satisfies the *detailed balance equation* with respect to a distribution π, holds for any $x, x' \in S$. It is also called *reversible* to π.

* If a finite MX is reversible to π, then π is a stationary distribution.

IV. Ergodic Theorem

Given an ergodic MC $(X_t)_{t\in N_0}$ over a finite state space S with stationary distribution π and a function $f: S \to R$, $\frac{1}{n}\sum_{i=1}^n f(x_1) \to \sum_{x\in S}\pi(x)f(x) = E_{x\sim\pi}[f(x)]$ as $n \to \infty$, $x_i \sim X_i|x_{i-1}$. This shows why MCMC is possible.

V. Metropolis-Hastings Algorithm

```
initialize x ∈ R^n
for t = 1 to T do
    sample x' ~ r(x' | x)
    sample u ~ Unif([0,1])
    if u ≤ α(x' | x) then update x ← x'
    else update x ← x
```

$r(\boldsymbol{x}'|\boldsymbol{x})$ proposal distribution, given state x proposes x'. The acceptance distribution $Bern(\alpha(\boldsymbol{x}'|\boldsymbol{x}))$ is shown:

$\alpha(\boldsymbol{x}'|\boldsymbol{x}) = \min\left\{1, \frac{p(x')r(\boldsymbol{x}|\boldsymbol{x}')}{p(x)r(\boldsymbol{x}'|\boldsymbol{x})}\right\} = \min\left\{1, \frac{q(x')r(\boldsymbol{x}|\boldsymbol{x}')}{q(x)r(\boldsymbol{x}'|\boldsymbol{x})}\right\}$, with stationary distribution $p(\boldsymbol{x}) = \frac{1}{Z}q(\boldsymbol{x})$.

VI. Gibbs Sampling

initialize $\boldsymbol{x} = [x_1, \dots, x_n] \in \mathbb{R}^n$
for $t = 1$ **to** T **do**
  pick a variable i uniformly at random from $\{1, \dots, n\}$
  set $\boldsymbol{x}_{-i} \doteq [x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n]$
  update x_i by sampling according to the posterior distribution $p(x_i \mid \boldsymbol{x}_{-i})$

A Gibbs distribution is a continuous distribution $p(x) = \frac{1}{Z}\exp(-f(x))$, f is called an energy function, f convex, log-concave. If the posterior is a Gibbs distribution, then $\alpha(\boldsymbol{x}'|\boldsymbol{x}) = \min\left\{1, \frac{r(\boldsymbol{x}|\boldsymbol{x}')}{r(\boldsymbol{x}'|\boldsymbol{x})}\exp(f(\boldsymbol{x}) - f(\boldsymbol{x}'))\right\}$.

VII. MALA / LMC

$r(x'|x) = N(\boldsymbol{x}'; \boldsymbol{x} - \eta_t \nabla f(\boldsymbol{x}), 2\eta_t \boldsymbol{I})$ log-concave → convergence to stationary distribution in polynomial time, use gradient information for convergence.

VIII. SGLD

initialize $\boldsymbol{\theta}$
for $t = 1$ **to** T **do**
  sample $i_1, \dots, i_m \sim \mathrm{Unif}(\{1, \dots, n\})$ independently
  sample $\varepsilon \sim \mathcal{N}(\boldsymbol{0}, 2\eta_t \boldsymbol{I})$
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta_t\left(\nabla \log p(\boldsymbol{\theta}) + \frac{n}{m}\sum_{j=1}^{m} \nabla \log p(y_{i_j} \mid \boldsymbol{x}_{i_j}, \boldsymbol{\theta})\right) + \varepsilon$

Guaranteed to converges with $\eta_t = \Theta(t^{-1/3})$.

6. Deep Learning

For *Bayesian neural networks*, we have a Gaussian prior on weights $\theta \sim N(0, \sigma_p^2 I)$ and Gaussian likelihood $y|\boldsymbol{x}, \boldsymbol{\theta} \sim N(f(\boldsymbol{x}; \boldsymbol{\theta}), \sigma_n^2)$. We have $\widehat{\boldsymbol{\theta}}_{MAP} = \mathrm{argmin}_\theta \frac{1}{2\sigma_p^2}||\boldsymbol{\theta}||_2^2 + \frac{1}{2\sigma_n^2}\sum_{i=1}^n (y_i - f(\boldsymbol{x}_i; \boldsymbol{\theta}))^2$. Update $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}\left(1 - \frac{1}{\sigma_p^2}\eta_t\right) + \eta_t \sum_{i=1}^n \nabla \log p(y_i|\boldsymbol{x}_i, \boldsymbol{\theta})$.

$y|\boldsymbol{x}, \boldsymbol{\theta} \sim N(f_1(\boldsymbol{x}; \boldsymbol{\theta}), \exp(f_2(\boldsymbol{x}; \boldsymbol{\theta})))$ is the *heteroscedastic noise* when there are 2 outputs f_1, f_2 in a neural network. $\log p(y_i|\boldsymbol{x}_i, \boldsymbol{\theta}) = \log\frac{1}{\sqrt{2\pi}} - \frac{1}{2}\left[\log\sigma^2(\boldsymbol{x}_i; \boldsymbol{\theta}) + \frac{(y_i - \mu(\boldsymbol{x}_i; \boldsymbol{\theta}))^2}{\sigma^2(\boldsymbol{x}_i; \boldsymbol{\theta})}\right]$.

Calibration: $\mathrm{freq}(B_m) = \frac{1}{|B_m|}\sum_{i \in B_m} 1\{Y_i = 1\}$

$\mathrm{conf}(B_m) = \frac{1}{|B_m|}\sum_{i \in B_m} P(Y_i = 1|x_i)$.

l_{ECE} $\sum_{m=1}^{M} \frac{|B_m|}{n}|\mathrm{freq}(B_m) - \mathrm{conf}(B_m)|$

l_{MCE} $\max_{m \in [M]}|\mathrm{freq}(B_m) - \mathrm{conf}(B_m)|$

7. Active Learning

Idea: Use uncertainty to select training points.

I. Conditional & Joint Entropy

Conditional entropy $H[\boldsymbol{X}|\boldsymbol{Y} = y] = E_{\boldsymbol{x} \sim p(\boldsymbol{x}|\boldsymbol{y})}[-\log p(\boldsymbol{x}|\boldsymbol{y})]$; joint entropy $H[\boldsymbol{X}, \boldsymbol{Y}] = E_{(\boldsymbol{x},\boldsymbol{y}) \sim p(\boldsymbol{x},\boldsymbol{y})}[-\log p(\boldsymbol{x}, \boldsymbol{y})]$.

$H[X, Y] = H[Y] + H[X|Y] = H[X] + H[Y|X]$;
$H[X|Y] = H[Y|X] + H[X] - H[Y]$;
$H[X|Y] \le H[X]$.

II. Mutual Information

$I[\boldsymbol{X}; \boldsymbol{Y}] = H[\boldsymbol{X}] - H[\boldsymbol{X}|\boldsymbol{Y}] = H[\boldsymbol{X}] + H[\boldsymbol{Y}] - H[\boldsymbol{X}, \boldsymbol{Y}]$, $I[\boldsymbol{X}; \boldsymbol{Y}] = I[\boldsymbol{Y}; \boldsymbol{X}]$.Gaussian $\boldsymbol{Y} = \boldsymbol{X} + \boldsymbol{\epsilon}$, $\boldsymbol{X} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, $\boldsymbol{\epsilon} \sim N(0, \sigma_n^2 \boldsymbol{I})$, $I[\boldsymbol{X}; \boldsymbol{Y}] = \frac{1}{2}\log\det(\boldsymbol{I} + \sigma_n^{-2}\boldsymbol{\Sigma})$.

Conditional mutual information $I[\boldsymbol{X}; \boldsymbol{Y}|\boldsymbol{Z}] = I[\boldsymbol{X}; \boldsymbol{Y}, \boldsymbol{Z}] - I[\boldsymbol{X}; \boldsymbol{Z}]$, $I[\boldsymbol{X}; \boldsymbol{Y}|\boldsymbol{Z}] = I[\boldsymbol{Y}; \boldsymbol{X}|\boldsymbol{Z}]$.

III. Maximizing Mutual Information

Monotone submodular function (objective) $I[\boldsymbol{S}] = I[\boldsymbol{f}_S; \boldsymbol{y}_S] = H[\boldsymbol{f}_S] - H[\boldsymbol{f}_S|\boldsymbol{y}_S]$. Greedy approach picks the locations x_1 through x_n for the maximal mutual information, providing a $(1 - 1/e)$-approximation of the optimum.

IV. Uncertainty Sampling

$x_{t+1} = \mathrm{argmin}_x I(\boldsymbol{f}_x; y_x|\boldsymbol{Y}_{S_t})$, $S_t = \{x_1, \cdots, x_t\}$ does not distinguish aleatoric and epistemic.

V. Bayesian active learning by disagreement

Identifying points x where the models disagrees about the label y_x: $x_{t+1} = \mathrm{argmax}_x I(\theta; y_x|x_{1:t}, y_{1:t}) = \mathrm{argmax}_x H[y_x|x_{1:t}, y_{1:t}] - E_{\theta|x_{1:t}, y_{1:t}}[H[y_x|\theta]]$

8. Bayesian Optimization

I. Exploration & Exploitation

Exploration is choosing points that are informative with respect to the unknow function. **Exploitation** is choosing promising points where we expect the function to have high values.

II. Regret – Performance Metric

The regret for a time horizon T associated with choices $\{x_t\}_{t=1}^T$ is $R_T = \sum_{t=1}^T\left(\max_x f^*(x) - f^*(x_t)\right)$. $\lim_{T \to \infty}\frac{R_T}{T} = 0$ achieve sublinear regret.

III. Acquisition Functions – F

initialize $f \sim \mathcal{GP}(\mu_0, k_0)$
for $t = 1$ **to** T **do**
  choose $x_t = \arg\max_{x \in \mathcal{X}} F(x; \mu_{t-1}, k_{t-1})$
  observe $y_t = f(x_t) + \epsilon_t$
  perform a probabilistic update to obtain μ_t and k_t

To pick next point to sample.

IV. Upper Confidence Bound (UCB)

$x_{t+1} = \mathrm{argmax}_x \mu_t(x) + \beta_{t+1}\sigma_t(x)$, $\sigma_t(x) = \sqrt{k_t(x, x)}$, regulates how confident about the model. $\beta_t = 0$ leads to exploitation. $\beta_t \to \infty$ leads to uncertainty sampling (exploration). GP-UCB $R_T = O(\beta_T(\delta)\sqrt{\gamma_T T})$, $\gamma_T = \max I(\boldsymbol{f}_S; \boldsymbol{y}_S) = \max\frac{1}{2}\log\det(\boldsymbol{I} + \sigma_n^{-2}\boldsymbol{K}_{SS})$.

V. Information Gain for Common Kernels

Linear $\gamma_T = O(d\log T)$; Gaussian $\gamma_T = O((\log T)^{d+1})$; Matern $(\nu > 0.5)$ $\gamma_T = O\left(T^{\frac{d}{2\nu + d}}(\log T)^{\frac{2\nu}{2\nu + d}}\right)$.

9. Markov Decision Processes (MDP)

I. Basics & Definition

A finite MDP is specified by a finite set of **states** $X = \{1, \cdots, n\}$, a set of **actions** $A = \{1, \cdots, m\}$, **transition probabilities** $p(x'|x, a) = P(X_{t+1} = x'|X_t = x, A_t = a)$, a **reward function** $r: X \times A \to R$ maps current state x and action a to some reward, induces a sequence of rewards $R_t = r(X_t, A_t)$. A **policy** is a function that maps each state $x \in X$ to a probability distribution over the actions. For $t > 0$, $\pi(a|x) = P(A_t = a|X_t = x)$, induces a MC $(X_t^\pi)_{t \in N_0}$ with transition probabilities $p^\pi(x'|x) = P(X_{t+1}^\pi = x'|X_t^\pi = x) = \sum_{a \in A}\pi(a|x)p(x'|x, a)$. The *discounted payoff* from time t is defined as $G_t = \sum_{m=0}^\infty \gamma^m R_{t+m}$, $\gamma \in [0,1)$ is the discount factor. **State value function** $v_t^\pi(x) = E_\pi[G_t|X_t = x]$ measures the average discounted payoff from time t, state $x \in X$. **State-action value function** (Q-function) $q_t^\pi(x, a) = E_\pi[G_t|X_t, A_t] = r(x, a) + \gamma\sum_{x' \in X} p(x'|x, a)\, v_{t+1}^\pi(x')$ measures average discounted payoff from time t, state $x \in X$, action $a \in A$. * $q_t^\pi(x, a)$, $v_t^\pi(x)$ are deterministic scalar-valued functions; the discounted payoff starting from a given state x will be independent of the start time t.

II. Bellman Expectation Equation

$v^\pi(x) = r(x, \pi(x)) + \gamma E_{x'|x,\pi(x)}[v^\pi(x')]$; for stochastic policies $v^\pi(x) = E_{a \sim \pi(x)}[q^\pi(x, a)]$, $q^\pi(x, a) = r(x, a) + \gamma E_{x'|x,a}E_{a' \sim \pi(x')}[q^\pi(x', a')]$; for deterministic policies $v^\pi(x) = q^\pi(x, \pi(x))$.

III. Policy Evaluation

$\boldsymbol{v}^\pi = \begin{bmatrix} v^\pi(1) \\ \vdots \\ v^\pi(n) \end{bmatrix}$, $\boldsymbol{r}^\pi = \begin{bmatrix} r(1, \pi(1)) \\ \vdots \\ r(n, \pi(n)) \end{bmatrix}$, $\boldsymbol{p}^\pi = \begin{bmatrix} p(1|1, \pi(1)) & \cdots & p(n|1, \pi(1)) \\ \vdots & \ddots & \vdots \\ p(1|n, \pi(n)) & \cdots & p(n|n, \pi(n)) \end{bmatrix}$

$\Rightarrow \boldsymbol{v}^\pi = (\boldsymbol{I} - \gamma\boldsymbol{P}^\pi)^{-1}\boldsymbol{r}^\pi$ takes $O(n^3)$ to solve. Fixed-point iteration is faster, with $\boldsymbol{B}^\pi$ affine, $\boldsymbol{v}^\pi \leftarrow \boldsymbol{B}^\pi\boldsymbol{v}^\pi = \boldsymbol{r}^\pi + \gamma\boldsymbol{P}^\pi\boldsymbol{v}^\pi$ from $t = 1$ to T.

IV. Policy Optimization

Goal: $\pi^* = \mathrm{argmax}_\pi E_\pi[G_0]$, $v^*(x)$, $q^*(x, a)$.
Greedy: $\pi_q(x) = \mathrm{argmax}_a q(x, a)$, $\pi_v(x) = \mathrm{argmax}_a r(x, a) + \gamma\sum_{x'} p(x'|x, a)\, v(x')$.
* v, q can be used interchangeably in this case.
Bellman: π^* is optimal if and only if $\pi^*(x)$ is a distribution over the set $\mathrm{argmax}_a q^*(x, a)$. For every state there is a *unique* action that maximizes the state-action value function, the policy $\pi^*$is *deterministic* and *unique*. The optimal value functions v^* and $q^*$are a fixed-point of Bellman update $v^* = \max_{a \in A} q^*(x, a) = \max_{a \in A} r(x, a) + \gamma E_{x'|x,a}[v^*(x')]$.

V. Policy & Value Iteration

initialize π (arbitrarily)
repeat
  compute v^π
  compute π_{v^π}
  $\pi \leftarrow \pi_{v^\pi}$
until converged

Initialize $V_0(x) = r(x, a), t = 0$
Repeat (until convergence
  ► $Q_t(x, a) = r(x, a) + \gamma\Sigma_{x'}P(x'|x, a)V_{t-1}(x')\ \forall x, a$
  ► $V_t(x) = \max_a Q_t(x, a)$
  ► $t = t + 1$
Convergence determined by l_∞ norm

Policy iteration (shown on the left) *guarantees* to converge $O(n^2 m/(1 - \gamma))$ to an optimal policy for *finite* MDPs. **Value iteration** (below) only get ϵ-optimal policy, converges $O(P)$ as v^*, q^* are *fixed-points* of $\boldsymbol{B}^*$. * Latter is faster *in general*.

VI. Partial Observability (POMDP)

POMDP as MDP with enlarged state space. Powerful but *intractable*. States with belief $B = \Delta(\{1, \cdots, n\}) = \{b: \{1, \cdots, n\} \to [0,1], \sum_x b(x) = 1\}$, actions same as MDP, transition model (stochastic) $P(Y_{t+1} = y|b_t, a_t) = \sum_{x,x'} b_t(x)P(x'|x, a_t)P(y|x')$, state update $b_{t+1}(x') = \frac{1}{z}\sum_x b_t(x)P(X_{t+1}|X_t, a_t)P(y_{t+1}|x')$, reward function $r(b_t, a_t) = \sum_x b_t(x)r(x, a_t)$.

* Most belief never reached. Policy gradients!

10. Tabular Reinforcement Learning

I. Overview

A trajectory $\tau = (\tau_0, \tau_1, \cdots)$ with $\tau_i = (x_i, a_i, r_i, x_{i+1})$. Agent can choose any policy (**on-policy**) or not using policy (**off-policy**), learn the underlying MDP (**model-based**) or learn the value function directly (**model-free**).

II. R_{max} (ONP & MB)

add the fairy-tale state x^* to the Markov decision process
set $\hat{r}(x, a) = R_{\max}$ for all $x \in X$ and $a \in A$
set $\hat{p}(x^* \mid x, a) = 1$ for all $x \in X$ and $a \in A$
compute the optimal policy $\hat{\pi}$ for $\hat{r}$ and $\hat{p}$
for $t = 0$ **to** ∞ **do**
  execute policy $\hat{\pi}$ (for some number of steps)
  for each visited state-action pair (x, a), update $\hat{r}(x, a)$
  estimate transition probabilities $\hat{p}(x' \mid x, a)$
  after observing "enough" transitions and rewards, recompute the optimal policy $\hat{\pi}$ according the current model $\hat{p}$ and $\hat{r}$.

With probability at least $1 - \delta$, R_{max} reaches an ϵ-optimal policy in a number of steps that is polynomial in $|X|, |A|, T, 1/\epsilon, 1/\delta$.
* For R_{max}, $O(n^2 m)$ space and $O(nm)$ time.

III. TD Learning (ONP & MF)

initialize V^π arbitrarily (e.g., as 0)
for $t = 0$ **to** ∞ **do**
  follow policy π to obtain the transition (x, a, r, x')
  $V^\pi(x) \leftarrow (1 - \alpha_t)V^\pi(x) + \alpha_t(r + \gamma V^\pi(x'))$

Update with bootstrap. If learning rate $(\alpha_t)_{t \in N_0}$ satisfies RM and all state-action pairs are chosen infinitely often, then V^π converges to v^π with prob. = 1.
* RM: $\alpha_t \ge 0, \sum_{t=0}^\infty \alpha_t = \infty, \sum_{t=0}^\infty \alpha_t^2 < \infty$.
We can also use Q-function, then the update is $Q^\pi(x, a) \leftarrow (1 - \alpha_t)Q^\pi(x, a) + \alpha_t(r + \gamma Q^\pi(x', a'))$.
* $v^\pi(x) \approx r + \gamma v^\pi(x')$, $q^\pi(x, a) \approx r + \gamma q^\pi(x', a')$.

IV. Q-learning (OFP & MF)

initialize $Q^*(x, a)$ arbitrarily (e.g., as **0**)
for $t = 0$ **to** ∞ **do**
  observe the transition (x, a, r, x')
  $Q^*(x, a) \leftarrow (1 - \alpha_t)Q^*(x, a) + \alpha_t(r + \gamma\max_{a' \in A} Q^*(x', a'))$

With probability at least $1 - \delta$, Q-learning converges ϵ-optimal policy in number of steps polynomial in $\log|X|, \log|A|, 1/\epsilon, \log 1/\delta$.

V. Optimistic Q-learning (OFP & MF)

Assume $0 \le r(x, a) \le R_{max}$, $V_{max} = \frac{R_{max}}{1 - \gamma} \ge \max_{x \in X, a \in A} q^*(x, a)$, we have the following
initialize $Q^*(x, a) = V_{\max}\prod_{t=1}^{T_{\mathrm{init}}}(1 - \alpha_t)^{-1}$
for $t = 0$ **to** ∞ **do**
  pick action $a_t = \arg\max_{a \in A} Q^*(x, a)$ and observe the transition (x, a_t, r, x')
  $Q^*(x, a_t) \leftarrow (1 - \alpha_t)Q^*(x, a_t) + \alpha_t(r + \gamma\max_{a' \in A} Q^*(x', a'))$

If T_{init} is large enough, converges quickly.
* $O(nm)$ space, $O(Tm)$ time, better than R_{max}

11. Model-free Reinforcement Learning

$O(|X|)$ space for value function store in a table, $O(|X| \cdot |A|)$ for Q-function.
Approximation is the solution.

I. Value Function Approximation

Have a linear function $\boldsymbol{\theta}$ with well-designed feature map $\boldsymbol{\phi}$, $Q(\boldsymbol{x}, \boldsymbol{a}; \boldsymbol{\theta}) = \boldsymbol{\theta}^T\boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{a})$. $\delta_B = r + \gamma\max_{a' \in A} Q^*(\boldsymbol{x}', \boldsymbol{a}'; \boldsymbol{\theta}^{old}) - Q^*(\boldsymbol{x}, \boldsymbol{a}; \boldsymbol{\theta})$ is the difference between approximation and the optimization target. Let $l(\boldsymbol{\theta}; \boldsymbol{x}, \boldsymbol{a}, r, \boldsymbol{x}') = \frac{1}{2}\delta_B^2$.

Gradient $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha_t\nabla_\theta l(.) \Leftrightarrow \boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha_t\delta_B\boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{a})$. * This demo is for Q-learning.

II. DQN & DDQN (from Q-learning)

$l_{DQN}(\boldsymbol{\theta}; D) = \frac{1}{2}\sum\left(r + \gamma\max_{a' \in A} Q^*(\boldsymbol{x}', \boldsymbol{a}'; \boldsymbol{\theta}^{old}) - Q^*(\boldsymbol{x}, \boldsymbol{a}; \boldsymbol{\theta})\right)^2$

Experience replay data D, cloned network to keep constant NN across episode.
$l_{DDQN}(\boldsymbol{\theta}; D) = \frac{1}{2}\sum(r + \gamma Q^*(\boldsymbol{x}', \boldsymbol{a}^*(\boldsymbol{x}'; \boldsymbol{\theta}); \boldsymbol{\theta}^{old}) - Q^*(\boldsymbol{x}, \boldsymbol{a}; \boldsymbol{\theta}))^2$, $\boldsymbol{a}^*(\boldsymbol{x}'; \boldsymbol{\theta}) = \mathrm{argmax}_{a' \in A} Q^*(\boldsymbol{x}', \boldsymbol{a}'; \boldsymbol{\theta})$. * Two Q's DDQN; DQN overestimates Q; large X small a.

III. Policy Gradient

$\nabla J(\theta) = \nabla E_{\tau \sim \pi_\theta} r(\tau) = E_{\tau \sim \pi_\theta}[r(\tau)\nabla\log\pi_\theta(\tau)]$ unbiased, high variance, reduce with baseline.

IV. REINFORCE (ONP & MF) local minima

initialize policy weights $\boldsymbol{\varphi}$
repeat
  generate an episode (i.e., rollout) to obtain trajectory τ
  for $t = 0$ **to** $T - 1$ **do**
    set $g_{t:T}$ to the downstream return from time t
    $\boldsymbol{\varphi} \leftarrow \boldsymbol{\varphi} + \eta\gamma^t g_{t:T}\nabla_{\boldsymbol{\varphi}}\log\pi_{\boldsymbol{\varphi}}(\boldsymbol{a}_t \mid \boldsymbol{x}_t)$ // (12.41)
until converged

V. Actor Critic

initialize parameters $\boldsymbol{\varphi}$ and $\boldsymbol{\theta}$
repeat
  use $\pi_{\boldsymbol{\varphi}}$ to obtain transition (x, a, r, x')
  $\delta = r + \gamma Q(x', \pi_{\boldsymbol{\varphi}}(x'); \boldsymbol{\theta}) - Q(x, a; \boldsymbol{\theta})$
  // actor update
  $\boldsymbol{\varphi} \leftarrow \boldsymbol{\varphi} + \eta\gamma^t Q(x, a; \boldsymbol{\theta})\nabla_{\boldsymbol{\varphi}}\log\pi_{\boldsymbol{\varphi}}(\boldsymbol{a} \mid x)$
  // critic update
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta\delta\nabla_{\boldsymbol{\theta}} Q(x, a; \boldsymbol{\theta})$
until converged

Actor-critic algorithms use a baseline to reduce **variance**. **Improve** under compatibility conditions.

12. Model-based Reinforcement Learning

Algorithm 13.1: Model-based reinforcement learning (outline)
start with an initial policy π and no (or some) initial data $\mathcal{D}$
for several episodes do
  roll out policy π to collect data
  learn a model of the dynamics f and rewards r from data
  plan a new policy π based on the estimated models

Algorithm 13.2: Model predictive control, MPC
for $t = 0$ **to** ∞ **do**
  observe x_t
  plan over a finite horizon H,
  $$\max_{a_{t:t+H-1}} \sum_{\tau=t}^{t+H-1} \gamma^{\tau-t} r(x_\tau, a_\tau) \quad \text{such that } x_{\tau+1} = f(x_\tau, a_\tau) \quad (13.3)$$
  carry out action a_t

Planning example

Learn *unknown* dynamics: conditional Gaussian, MAP (poor), Bayesian Learning. Exploration – Exploitation dilemma, add noise, Thomson sampling, optimistic *exploration*.

Algorithm 13.11: Thompson sampling
start with (possibly empty) data $\mathcal{D} = \emptyset$ and a prior $p(f) = p(f \mid \mathcal{D})$
for several episodes do
  sample a model $f \sim p(\cdot \mid \mathcal{D})$
  plan a new policy π to (approximately) maximize,
  $$\max_\pi J_H(\pi; f) \quad (13.21)$$
  roll out policy π to collect more data
  update posterior $p(f \mid \mathcal{D})$

13. Kalman Filter

start with a prior over initial states $p(x_0)$
for $t = 1$ **to** ∞ **do**
  assume we have $p(x_t \mid \boldsymbol{y}_{1:t-1})$
  compute $p(x_t \mid \boldsymbol{y}_{1:t})$ using the new observation y_t (conditioning)
  compute $p(x_{t+1} \mid \boldsymbol{y}_{1:t})$ (prediction)

Bayesian Filter

$\boldsymbol{X}_0 \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$; $\boldsymbol{X}_{t+1} = \boldsymbol{F}\boldsymbol{X}_t + \boldsymbol{\epsilon}_t$; $\boldsymbol{Y}_t = \boldsymbol{H}\boldsymbol{X}_t + \boldsymbol{\eta}_t$; $\boldsymbol{F} \in R^{d \times d}$; $\boldsymbol{H} \in R^{m \times d}$; $\boldsymbol{\epsilon}_t \sim N(0, \boldsymbol{\Sigma}_x)$; $\boldsymbol{\eta}_t \sim N(0, \boldsymbol{\Sigma}_y)$.
Update: $\boldsymbol{X}_{t+1}|\boldsymbol{y}_{1:t+1} \sim N(\boldsymbol{\mu}_{t+1}, \boldsymbol{\Sigma}_{t+1})$, $\boldsymbol{\mu}_{t+1} = \boldsymbol{F}\boldsymbol{\mu}_t + \boldsymbol{K}_{t+1}(\boldsymbol{y}_{t+1} - \boldsymbol{H}\boldsymbol{F}\boldsymbol{\mu}_t)$, $\boldsymbol{\Sigma}_{t+1} = (\boldsymbol{I} - \boldsymbol{K}_{t+1}\boldsymbol{H})(\boldsymbol{F}\boldsymbol{\Sigma}_t\boldsymbol{F}^T + \boldsymbol{\Sigma}_x)$, $\boldsymbol{K}_{t+1} = (\boldsymbol{F}\boldsymbol{\Sigma}_t\boldsymbol{F}^T + \boldsymbol{\Sigma}_x)\boldsymbol{H}^T(\boldsymbol{H}(\boldsymbol{F}\boldsymbol{\Sigma}_t\boldsymbol{F}^T + \boldsymbol{\Sigma}_x)\boldsymbol{H}^T + \boldsymbol{\Sigma}_y)^{-1}$ is Kalman gain.
Prediction: $\boldsymbol{X}_{t+1}|\boldsymbol{y}_{1:t} \sim N(\widehat{\boldsymbol{\mu}}_{t+1}, \widehat{\boldsymbol{\Sigma}}_{t+1})$, $\widehat{\boldsymbol{\mu}}_{t+1} = \boldsymbol{F}\boldsymbol{\mu}_t$, $\widehat{\boldsymbol{\Sigma}}_{t+1} = \boldsymbol{F}\boldsymbol{\Sigma}_t\boldsymbol{F}^T + \boldsymbol{\Sigma}_x$.