

Probability review

Product rule: $P(X_{1:n}) = P(X_1) \prod_{i=2}^n P(X_i | X_{1:i-1})$.

Sum rule: $P(X,Y) = \sum_y P(X,Y=y)$.

Bayes rule: $P(X|Y) = \frac{P(Y|X)P(X)}{P(Y)}$.

Independence: $P_{XY} = P_X P_Y$.

Conditional independence: $P_{XY|Z} = P_{X|Z} P_{Y|Z}$.

Linearity of expectation: $\mathbb{E}_{x,y}[aX + bY] = a\mathbb{E}_x[X] + b\mathbb{E}_y[Y]$.

Expectation: $\mathbb{E}_p[f(X)] = \sum_{i=0}^n p(x)f(x)$ (don't forget $p(x)$).

Variance: $\text{Var}[X] = \mathbb{E}[(X - \mathbb{E}[X])^2] = \mathbb{E}[X^2] - \mathbb{E}[X]^2$.

Linearity of variance:

$\text{Var}[aX + bY + c] = a^2\text{Var}[X] + b^2\text{Var}[Y] + 2ab\text{Cov}(X,Y)$.

Covariance: $\text{Cov}(X,Y) = \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])]$.

Cum. dist. function: $\mathbb{P}(x \leq t) = F(t)$, where F is CDF.

Matrix inversion: $\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad-bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$.

Multivariate Gaussian:

$$\mathcal{N}(x; \mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^d |\Sigma|}} \exp\left(-\frac{1}{2}(x - \mu)^\top \Sigma^{-1}(x - \mu)\right).$$

A random vector is Gaussian if (1) the RVs are Gaussian, and (2) any linear combination of the RVs is Gaussian.

Properties:

$$\begin{aligned} X_A &\sim \mathcal{N}(\mu_A, \Sigma_{AA}) \\ X_A | X_B &\sim \mathcal{N}(\mu_{A|B}, \Sigma_{A|B}) \\ \mu_{A|B} &= \mu_A + \Sigma_{AB} \Sigma_{BB}^{-1} (x_B - \mu_B) \\ \Sigma_{A|B} &= \Sigma_{AA} - \Sigma_{AB} \Sigma_{BB}^{-1} \Sigma_{BA} \\ MX &\sim \mathcal{N}(M\mu, M\Sigma M^\top) \\ X + X' &\sim \mathcal{N}(\mu + \mu', \Sigma + \Sigma') \end{aligned}$$

If asked about **conditional distribution of linear functions of Gaussians**, notice that the functions can be jointly computed by a matrix operation, which results in a Gaussian, or use LoV.

Kalman filters: Motion model updates the state $X_{t+1} = FX_t + \epsilon_t$. Sensor model computes observation $Y_t = HX_t + \eta_t$. $\epsilon_t \sim \mathcal{N}(0, \Sigma_x)$, $\eta_t \sim \mathcal{N}(0, \Sigma_y)$. $\epsilon_t \uparrow \Rightarrow K_{t+1} \uparrow$, $\eta_t \uparrow \Rightarrow K_{t+1} \downarrow$, $\Sigma_t \uparrow \Rightarrow K_{t+1} \uparrow$. Update:

$$\begin{aligned} X_{t+1} | y_{1:t+1} &\sim \mathcal{N}(\mu_{t+1}, \Sigma_{t+1}) \\ \mu_{t+1} &= F\mu_t + K_{t+1}(y_{t+1} - HF\mu_t) \\ \Sigma_{t+1} &= (I - K_{t+1}H)(F\Sigma_t F^\top + \Sigma_x) \\ K_{t+1} &= (F\Sigma_t F^\top + \Sigma_x)H^\top (H(F\Sigma_t F^\top + \Sigma_x)H^\top + \Sigma_y)^{-1} \end{aligned}$$

Entropy: $H[p] = \mathbb{E}_p[-\log p(x)]$.

d-Gaussian: $H[\mathcal{N}(\mu, \Sigma)] = \frac{d}{2}(1 + \log(2\pi)) + \frac{1}{2}\log|\Sigma|$.

1-Gaussian: $H[\mathcal{N}(\mu, \sigma^2)] = \frac{1}{2}\log(2\pi\sigma^2) + \frac{1}{2}$.

$$H[p,q] = H[p] + H[q|p]. \quad H[X|Y] = \mathbb{E}_p\left[\log \frac{p(x,y)}{p(x)}\right].$$

KL-divergence: $KL(q||p) = \mathbb{E}_q\left[\log \frac{q(x)}{p(x)}\right] = \mathbb{E}_q\left[-\log \frac{p(x)}{q(x)}\right]$. Non-negative (0 if $p=q$, ∞ if $p(x)=0$ for x with $q(x)>0$). Additional expected surprise when observing q samples while assuming p .

Mutual information: $I(X;Y) = H[X] - H[X|Y]$. Symmetric: $I(X;Y) = I(Y;X)$. Information never hurts: $I(X;Y) \geq 0$.

Jensen's inequality: If f convex: $f(\mathbb{E}[X]) \leq \mathbb{E}[f(X)]$.

MLE: $\hat{\theta} = \text{amax}_{\theta} p(y|X, \theta) = \text{amax}_{\theta} \sum_{i=1}^n \log p(y_i | x_i, \theta)$.

MAP: $\hat{\theta} = \text{amax}_{\theta} p(\theta|X, y) = \text{amax}_{\theta} \log p(\theta) + \sum_{i=1}^n \log p(y_i | x_i, \theta)$.

Bayesian learning: Prior: $p(\theta)$. Likelihood: $p(y|X, \theta) = \prod_{i=1}^n p(y_i | x_i, \theta)$. Posterior: $p(\theta|X, y) = \frac{1}{Z} p(\theta) \prod_{i=1}^n p(y_i | x_i, \theta)$. $Z = \int p(\theta) \prod_{i=1}^n p(y_i | x_i, \theta) d\theta$. **Prediction:** $p(y^* | x^*, X, y) = \int p(y^* | x^*, \theta) p(\theta|X, y) d\theta$. In general, intractable: GP, VI, and MCMC solve.

Aleatoric uncertainty: Uncertainty due to irreducible noise in data. **Epistemic uncertainty:** Uncertainty due to lack of data.

LoTV: $\text{Var}[y^* | x^*] = \mathbb{E}_{\theta}[\text{Var}_{y^*}[y^* | x^*, \theta]] + \text{Var}_{\theta}[\mathbb{E}_{y^*}[y^* | x^*, \theta]]$.

Bayesian Linear Regression $f^* = w^\top x^*, y^* = f^* + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma_n^2)$. Prior: $w \sim \mathcal{N}(0, \sigma_p^2 I)$. **Posterior:**

$$\begin{aligned} w | X, y &\sim \mathcal{N}(\bar{\mu}, \bar{\Sigma}), \quad \bar{\mu} = (X^\top X + \sigma_n^2 \sigma_p^{-2} I)^{-1} X^\top y = \sigma_n^{-2} \Sigma X^\top y, \\ \bar{\Sigma} &= (\sigma_n^{-2} X^\top X + \sigma_p^{-2} I)^{-1}. \end{aligned}$$

Inference: $y^* = x^{*\top} w + \epsilon \Rightarrow f^* | x^*, X, y \sim \mathcal{N}(x^{*\top} \bar{\mu}, x^{*\top} \bar{\Sigma} x^* + \sigma_n^2)$.

Logistic regression: Bernoulli likelihood $(\pi^y(1-\pi)^{1-y})$.

Recursive update: $p^{(t+1)}(\theta) = p(\theta | y_{1:t+1}) = \frac{1}{Z} p^{(t)}(\theta) p(y_{t+1} | \theta)$.

Online data recursion: $X^\top X = \sum_{i=1}^n x_i x_i^\top$, $X^\top y = \sum_{i=1}^n y_i x_i$.

Gaussian Processes **Problem:** BLR can only make linear predictions. **Solution:** GP describes distributions over (non-linear) functions. In function space, $f = Xw$, which can be sampled by $f \sim \mathcal{N}(0, X^\top X) \Rightarrow$ data points enter as inner products \Rightarrow use kernel function $f \sim \mathcal{N}(0, k(X, X))$. $\mathcal{GP}(\mu, k)$ is formally defined as an infinite collection of RVs, of which any finite number are jointly Gaussian.

Kernel: Formally, $k(x, x') = \phi(x)^\top \phi(x')$ for some feature function $\phi \Rightarrow$ kernel function is more efficient. Intuitively, $k(x, x')$ describes how $f(x)$ and $f(x')$ are related. $k(X, X)$ is symmetric and positive semidefinite ($z^\top M z \geq 0$ for all $z \neq 0$). Must satisfy $k(x, x') \leq \sqrt{k(x, x)k(x', x')}$. **Stationary:** $k(x, x') = k(x - x')$. **Isotropic:** $k(x, x') = k(\|x - x'\|_2)$ (same as stat. in 1D). **Linear:** Line. **Gaussian (RBF):** Smooth, larger ℓ : smoother. **Laplace (exponential):** Non-smooth, larger ℓ : smoother. **Matérn:** $\nu = 1/2$: Laplace, $\nu \rightarrow \infty$: Gaussian. Addition, multiplication, scaling, polynomial function of kernel functions are also kernel functions.

Inference: $y^* | x^*, X, y \sim \mathcal{N}(\mu^*, k^* + \sigma_n^2)$ with data X_A :

$$\begin{aligned} \mu^* &= \mu(x^*) + k_{x^*, A}^\top (K_{AA} + \sigma_n^2 I)^{-1} (y - \mu_A) \\ k^* &= k(x^*, x^*) - k_{x^*, A}^\top (K_{AA} + \sigma_n^2 I)^{-1} k_{x^*, A}. \end{aligned}$$

GP posterior: $\mathcal{GP}(\mu', k')$ such that:

$$\begin{aligned} \mu'(x) &= \mu(x) + k_{x, A}^\top (K_{AA} + \sigma_n^2 I)^{-1} (y - \mu_A) \\ k'(x, x') &= k(x, x') - k_{x, A}^\top (K_{AA} + \sigma_n^2 I)^{-1} k_{x', A}. \end{aligned}$$

Forward sampling: Iter sample 1-d Gaussian with prod. rule $p(f_1, \dots, f_n)$. **Model selection:** Hyperparameters matter a lot \Rightarrow Can be learned by maximizing marginal likelihood,

$$\hat{\theta} = \text{amin}_{\theta} y^\top K_{y, \theta}^{-1} y + \log \det(K_{y, \theta}),$$

which balances the goodness of the fit (term 1) and model complexity (term 2).

Problem: To learn a GP, need to invert matrices, which take $\mathcal{O}(n^3)$ (BLR: $\mathcal{O}(dn^2)$). **Local methods:** Stationary kernels depend on distance, so only condition if $|k(x, x')| \geq \tau$. **Approximation:** Approximate stationary kernel with Random Fourier Transform. **Inducing point (FITC):** Throw away data where there is a lot (cubic in inducing points, linear in data points).

Variational Inference In some cases, not realistic to assume Gaussian \Rightarrow Approximate $p(\theta|y) = \frac{1}{Z} p(\theta, y) \approx q_\lambda(\theta) \Rightarrow$ Minimize $KL(q_\lambda || p) \Rightarrow q^* = \text{amax}_{\lambda} \mathbb{E}_{\theta \sim q_\lambda} [\log p(y|\theta)] - KL(q_\lambda || p_{\text{prior}})$. I.e., minimizing $KL(q_\lambda || p) \equiv$ maximizing expected likelihood, while remaining close to prior. **ELBO:** Lower bounds $\log p(y)$, so it is a good method of model selection.

Forward KL $KL(p || q_\lambda)$: covers full prob. density, but intractable.

Backward KL $KL(q_\lambda || p)$: greedily covers mode.

Laplace approximation: $q_\lambda(\theta) = \mathcal{N}(\hat{\theta}, \Lambda)$, where $\hat{\theta} = \text{amax}_{\theta} p(\theta|y)$ and $\Lambda = -H_{\theta} \log p(\theta|y)$. Matches shape of the true posterior around its mode \Rightarrow Extremely overconfident predictions, because it is greedy.

Problem: Want to compute gradient w.r.t. λ of an expectation w.r.t. λ . **Reparameterization trick:** Suppose $\epsilon \sim \phi$, $\theta = g(\epsilon, \lambda)$ (diff. and inv.), then $\mathbb{E}_{\theta \sim q_\lambda} [f(\theta)] = \mathbb{E}_{\epsilon \sim \phi} [f(g(\epsilon, \lambda))]$. (Can be used for MC obj, unbiased.) **Gaussian:** $\theta = g(\epsilon, \lambda) = \Sigma^{1/2} \epsilon + \mu \sim \mathcal{N}(\mu, \Sigma)$.

Inference: $p(y^* | x^*, y) \approx \int p(y^* | f^*) q_\lambda(f^* | x^*) df^*$ (intractable, but single dimension).

Markov Chain Monte Carlo **Markov chain:** Seq. of RVs s.t. $X_{t+1} \perp X_{1:t-1} | X_t$. **Stationary dist.:** $\pi(x) = \sum_{x'} p(x | x') \pi(x')$. (Solve by $\pi = P^\top \pi$, $\mathbf{1}^\top \pi = \mathbf{1}$.) **Ergodicity:** $\exists t [\forall x, x' [p^{(t)}(x' | x) > 0]]$,

where $p^{(t)}$ is prob. to reach x' from x in exactly t steps. (Terminal states \Rightarrow not ergodic.) (Ensure ergodicity by self-loops.) **Fundamental theorem of ergodic MCs:** Ergodic MC always converges to a unique pos. stat. dist. **Detailed balance equation:** For an unnormalized dist. q an MC satisfies DBE iff

$$q(x)p(x' | x) = q(x')p(x | x') \Rightarrow \text{stat. dist.} = \frac{1}{Z} q. \quad \textbf{Sampling:} \text{ Sample MC's stat. dist. by first doing a burn-in } t_0 \text{ to reach stat. dist.}$$

Idea: Approximate intractable p by drawing m samples from MC with stat. dist. $p(\theta|y) \Rightarrow p(y^* | x^*, y) = \mathbb{E}_{p(\cdot|y)} [p(y^* | x^*, \theta)] = \frac{1}{m} \sum_{i=1}^m p(y^* | x^*, \theta_i)$.

Hoeffding's inequality: Compute bound on error.
 $p(|\mathbb{E}_{p(\cdot|y)}[p(y^* | x^*, \theta)] - \frac{1}{m} \sum_{i=1}^m p(y^* | x^*, \theta_i)| > \epsilon) \leq 2 \exp(-2m\epsilon^2/C^2)$,
 where C is the upper bound of values (1 for prob. dist.). To get a prob. $\leq \delta$ of error $> \epsilon$, we need $m \geq \log 2 - \log \delta / 2\epsilon^2$ samples.

Metropolis-Hastings: Arbitrary proposal dist. $r(x' | x)$. Follow proposal with prob. $\alpha(x' | x) = \min\left\{1, \frac{q(x')r(x|x')}{q(x)r(x'|x)}\right\} \Rightarrow$ satisfies DBE to get stat. dist. $\frac{1}{Z}q(x)$. Arbitrary proposal influences how fast we converge to stat. dist. **Gaussian:** Prob. dist. has form $p(x) = \frac{1}{Z} \exp(-f(x)) \Rightarrow \alpha(x' | x) = \min\left\{q, \frac{r(x|x')}{r(x'|x)} \exp(f(x) - f(x'))\right\}$. If $r(x' | x) = \mathcal{N}(x'; x, \tau I) \Rightarrow \frac{r(x|x')}{r(x'|x)} = 1$ (symmetry). If r proposes low energy (high prob) region, acceptance is 1. **Problem:** Uninformed \Rightarrow Use gradient information (MALA requires full access to f).

Bayesian Deep Learning Non-linear dependencies.

Prior: $\theta \sim \mathcal{N}(0, \sigma_p^2, I)$. **Likelihood:** $y | x, \theta \sim \mathcal{N}(\mu_\theta(x), \sigma_\theta^2(x))$.

Homoscedastic: Same noise for all data points, $\sigma_\theta^2(x) = c$.

Heteroscedastic: Varying noise.

MAP: $\text{amin}_\theta \frac{1}{2\sigma_p^2} \|\theta\|^2 - \sum_{i=1}^n \log \sigma_\theta^2(x_i) + \frac{(y_i - \mu_\theta(x_i))^2}{2\sigma_\theta^2(x_i)}$. Attenuate loss for certain data points by attributing error to large variance. Fails to model epistemic uncertainty \Rightarrow VI (Gaussian in expectation) and Monte Carlo or MCMC:

$\mathbb{E}[y^* | x^*, y] \approx \frac{1}{m} \sum_{j=1}^m \mu_{\theta_j}(x^*)$.

$\text{Var}[y^* | x^*, y] \approx \frac{1}{m} \sum_{j=1}^m \sigma_{\theta_j}^2(x^*) + \frac{1}{m-1} \sum_{j=1}^m (\mu_{\theta_j}(x^*) - \bar{\mu}(x^*))^2$

MCMC: Produce seq. $\theta_1, \dots, \theta_T$, then $p(y^* | x^*, y) \approx \frac{1}{T} \sum_{j=1}^T p(y^* | x^*, \theta_j)$. **Problem:** Cannot store T times params of network. **Solution:** Approx. with Gaussian and running mean/var. **MC dropout:** Dropout during inference \Leftrightarrow VI with Bernoulli. **Prob. ensembles:** Train networks on rand. subsets, average.

Calibration: Well-calibrated \Leftrightarrow confidence (assigned prob.) \approx frequency. **Reliability diagram:** Bin according to class pred. probs. (assume class 1). Above line: underconfident, below line: overconfident. (No samples \Rightarrow empty bin in diagram). $\text{freq}(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} \mathbb{1}\{Y_i = 1\}$,

$\text{conf}(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} p(Y_i = 1 | x_i)$. **ECE:** avg. deviation from perfect calibration: $\ell_{\text{ECE}} = \sum_{m=1}^M \frac{|B_m|}{n} |\text{freq}(B_m) - \text{conf}(B_m)|$.

Active Learning Decide which data to collect: \mathcal{NP} -hard.

Uncertainty sampling: Greedily pick points with maximal mutual information $\Rightarrow x_{t+1} = \text{amax}_x I(f_x; y_x | \mathcal{Y}_{S_t})$. If Gaussian: $x_{t+1} = \text{amax}_x \sigma_{x|S_t}^2 / \sigma_n^2(x)$. $\gamma_T = \max_{x_{1:T}} I(f(x_{1:T}); y_{1:T})$. Monotone submodular. Constant factor approx: $I(f(x_{1:T}); y_{1:T}) \geq (1 - 1/e) \gamma_T$ (near-optimal, $1 - 1/e \approx 0.63$).

BALD: $x_{t+1} = \text{amax}_x I(y_x; \theta | x_{1:t}, y_{1:t}) = \text{amax}_x H[y_x | x_{1:t}, y_{1:t}] - \mathbb{E}_{\theta | x_{1:t}, y_{1:t}}[H[y_x | \theta]]$. Want points where the post. is uncertain because all θ are certain about their differing pred. Approximate term 2 using VI and MC.

Bayesian Optimization Not only reduce uncertainty, but also maximize objective. $x_{t+1} = \text{amax}_x a(x)$.

Regret: $R_T = \sum_{t=1}^T (\max_x f(x) - f(x_t))$. Want algorithm with sublinear regret: $\lim_{T \rightarrow \infty} R_T/T = 0$. $f^* = \max_x f(x)$.

GP-UCB: Optimism in the face of uncertainty: pick point where we can hope for best outcome: $a_{\text{UCB}} = \mu_t(x) + \beta_t \sigma_t(x)$. μ, σ from GP. $R_T \in \mathcal{O}^*(\sqrt{\gamma_T/T})$. **GP bounds:** Linear: $\gamma_T \in \mathcal{O}(d \log T)$, Gaussian: $\mathcal{O}((\log T)^{d+1})$, Matérn: $\mathcal{O}(T^{d/2v+d} (\log T)^{2v/2v+d})$.

PI: $a_{\text{PI}}(x) = \Phi((\mu_t(x) - f^*)/\sigma_t(x))$ is prob. to improve f^* . Greedy. **EI:** $a_{\text{EI}}(x) = (\mu_t(x) - f^*) \Phi((\mu_t(x) - f^*)/\sigma_t(x)) + \sigma_t(x) \phi((\mu_t(x) - f^*)/\sigma_t(x))$ is expectation of improvement.

Thompson sampling: Draw sample from GP and select max.

Markov Decision Processes Env. that makes Markov ass. (states \mathcal{X} , actions \mathcal{A} , transitions $p(x' | x, a)$, rewards $r(x, a)$). **Policy:** π maps states to actions (induces MC with $p(x' | x) = \sum_a \pi(a | x) p(x' | x, a)$), want to find π that max. long-term rewards. Horizon T reward: $\mathbb{E}_\pi[\sum_{t=0}^T r(x_t, a_t)]$. Horizon ∞ reward: $\mathbb{E}_\pi[\sum_{t=0}^\infty \gamma^t r(x_t, a_t)]$. **Geo series:** $\sum_{t=0}^\infty \gamma^t = 1/(1-\gamma)$.

$V^\pi(x) = \mathbb{E}_x[\sum_{t=0}^\infty \gamma^t r(X_t, \pi(X_t)) | X_0 = x]$
 $= r(x, \pi(x)) + \gamma \sum_{x'} p(x' | x, \pi(x)) V^\pi(x')$ (Bellman eq).

$Q(x, a) = r(x, a) + \gamma \sum_{x'} p(x' | x, a) V^\pi(x')$. $V(x) = \max_a Q(x, a)$.

Bellman theorem: π is optimal \Leftrightarrow greedy w.r.t. V^π .

Policy iteration: $\pi \Rightarrow V^\pi, V \Rightarrow \pi_V$ (alternate). $\pi_V(x) = \text{amax}_a r(x, a) + \gamma \sum_{x'} p(x' | x, a) V(x')$ (greedy). Converges monotonically, guaranteed to converge in $\mathcal{O}(|\mathcal{X}|^2 |\mathcal{A}| / (1-\gamma))$ iterations, expensive (computed efficiently by solving single LSoE).

Value iteration: Dynamic programming:
 $V_t(x) = \max_a r(x, a) + \gamma \sum_{x'} p(x' | x, a) V_{t-1}(x')$. Iterates until $\|v_t - v_{t-1}\|_\infty \leq \epsilon$: **ϵ -optimal convergence**, in polynomial in iterations, per iteration: $\mathcal{O}(|\mathcal{X}|^2 |\mathcal{A}|)$, inexpensive. Then, pick greedy policy.

Reinforcement Learning Learn within unknown MDP. **On-policy:** Learn from own data, **Off-policy:** Learn from other policy data, **Model-based:** Learn MDP and solve, **Model-free:** Learn value function directly. Data points: $\langle x, a, r, x' \rangle$.

Robbins-Monro: $\sum_{t=0}^\infty x_t = \infty, \sum_{t=0}^\infty x_t^2 < \infty$. E.g. $1/t$. **ϵ -greedy (based):** Pick random action with prob. ϵ_t , or best action according to MDP with prob. $1 - \epsilon_t$. **Guaranteed to converge to optimal policy if ϵ_t satisfies RM.** Problem: does not quickly eliminate suboptimal actions.

R_{\max} (based): Solves problem. Add fairy tale $p(x^* | x^*, a) = 1, r(x^*, a) = R_{\max}$, assume unexplored go there.

TD-learning (on, free): $V^\pi(x) \leftarrow (1 - \alpha_t) V^\pi(x) + \alpha_t (r + \gamma V^\pi(x'))$. **Guaranteed to converge if α_t satisfies RM and all states are visited infinitely often.** Space: $\mathcal{O}(|\mathcal{X}|)$.

Q-learning (off, free): $Q(x, a) \leftarrow (1 - \alpha_t) Q(x, a) + \alpha_t (r + \gamma \max_{a'} Q(x', a'))$. **Guaranteed to converge if α_t satisfies RM and all state-action pairs are visited infinitely often.** Space: $\mathcal{O}(|\mathcal{X}| |\mathcal{A}|)$.

DQN (off, free, cont. states): GD on $\frac{1}{2} (Q(x, a; \theta) - (r + \gamma \max_{a'} Q(x', a'; \theta^{\text{old}})))^2$ (Bellman error). **Slow to converge:** maintain constant target network. **DDQN:** Maximization bias makes DQN overestimate. Solution: 2 Q networks where we take minimum to be value.

Policy search (on, free, cont. actions): Parametrize $\pi(x; \theta)$. Maximize expected trajectory reward. $\nabla_\theta \mathbb{E}_{\tau \sim \pi_\theta} [r(\tau)] = \mathbb{E}_{t \sim \pi_\theta} [r(\tau) \nabla_\theta \log \pi_\theta(\tau)] = \mathbb{E}_{t \sim \pi_\theta} [r(\tau) \sum_{i=0}^T \nabla_\theta \log \pi_\theta(a_t | x_t)] \Rightarrow$ Do not need to know MDP to compute gradient. **Baselines:** Large variance \Rightarrow introduce baseline: $\mathbb{E}_{\tau \sim \pi_\theta} [r(\tau) \nabla_\theta \log \pi_\theta] = \mathbb{E}_{\tau \sim \pi_\theta} [(r(\tau) - b(\tau)) \nabla_\theta \log \pi_\theta(\tau)]$.

REINFORCE (on, free): $\theta \leftarrow \theta + \eta \gamma^t G_t \nabla_\theta \log \pi_\theta(a_t | x_t)$, where $G_t = r(\tau) - b_t = \sum_{t'=t}^T \gamma^{t'-t} r_{t'}$.

Actor-critic (on, free): REINFORCE gradient = $\mathbb{E}_{(x,a) \sim \pi_\theta} [Q(x, a) \nabla_\theta \log \pi_\theta(a | x)]$. So: parametrize actor π_θ and critic Q_θ . Use in each others' update equations:
 $\theta_\pi \leftarrow \theta_\pi + \eta_t Q(x, a | \theta_Q) \nabla_\theta \log \pi_\theta(a | x)$.

$\theta_Q \leftarrow \theta_Q - \eta_t (Q(x, a; \theta_Q) - r - \gamma Q(x', \pi(x'; \theta_\pi); \theta_Q)) \nabla_\theta Q(x, a; \theta)$. **A2C (on, free):** Add value network V_θ for the baseline: $A(x, a) = Q(x, a) - V(x, a)$ (advantage function). This centers the Q-values.

DDPG (off, free): Replace $\max_{a'} Q(x', a'; \theta^{\text{old}})$ in DQN by $\pi(x'; \theta_\pi)$, where π should follow the greedy policy w.r.t. Q . **Key idea:** If we use a rich enough parameterization of policies, selecting the greedy policy w.r.t. Q is equivalent to $\theta_\pi^* = \text{amax}_{\theta_\pi} \mathbb{E}_{x \sim \mu} [Q(x, \pi(x; \theta_\pi); \theta_Q)]$. $\mu(x) > 0$ is an exploration distribution with full support. This needs det. π , thus we inject noise for exploration (akin ϵ -greedy).

TD3 (off, free): Add second critic network to address max. bias. **SAC (off, free):** Add entropy regularization to loss $\lambda H(\pi_\theta)$. **Planning:** Det. transition function $x_{t+1} = f(x_t, a_t)$ and reward function $r(x_t, a_t)$. Cannot plan over infinite horizon \Rightarrow **Key idea:** plan over finite horizon H , carry out first action, repeat. Optimize $J_H(a_{t:t+H-1}) = \sum_{t'=t}^{t+H-1} \gamma^{t'-t} r(x_{t'}, a_{t'})$. Local minima, vanishing/exploding gradient \Rightarrow heuristics \Rightarrow Random shooting (m samples, pick best). Will not work if sparse rewards \Rightarrow Get access to value function to look further: $J_H(a_{t:t+H-1}) = \sum_{t'=t}^{t+H-1} \gamma^{t'-t} r_t + \gamma^H V(x_{t+H})$. **Model-based ML:** Estimate f and r off-policy with supervised learning ($(x_t, a_t) \mapsto (r_t, x_{t+1})$, regression). **Benefit:** Dramatically decreases sample complexity (need less data). Use MAP estimate \Rightarrow exploited by planning algos \Rightarrow Uncertainty (GP/BNN).