1 Bayesian Regression

$w \sim N(0,\sigma_p^2 I)$, $\epsilon \sim N(0,\sigma_n^2 I)$, $y = Xw + \epsilon$

$y|w \sim N(Xw, \sigma_n^2 I)$

$w|y \sim N((X^TX+\lambda I)^{-1}X^Ty, (X^TX+\lambda I)^{-1}\sigma_n^2)$

2 Kalman Filter

$\begin{cases} X_{t+1} = FX_t + \epsilon_t & \epsilon_t \sim N(0,\Sigma_x) \\ Y_t = HX_t + \eta_t & \eta_t \sim N(0,\Sigma_y) \end{cases} X_1 \sim N(\mu_p, \Sigma_p)$

Then if X_0 is Gaussian then $X_t|Y_{1:t} \sim N(\mu_t, \sigma_t)$:

$\mu_{t+1} = F\mu_t + K_{t+1}(y_{t+1} - HF\mu_t)$

$\Sigma_{t+1} = (I - K_{t+1}H)(F\Sigma_t F^T + \Sigma_x)$

$K_{t+1} = (F\Sigma_t F^T + \Sigma_x)H^T(H(F\Sigma_t F^T + \Sigma_x)H^T + \Sigma_y)^{-1}$

3 Gaussian Processes

$f \sim GP(\mu, k) \Rightarrow \forall \{x_1, \dots, x_n\}\ \forall n < \infty$

$[f(x_1)\dots f(x_n)] \sim N([\mu(x_1)\dots\mu(x_n)], K)$

where $K_{ij} = k(x_i, x_j)$

3.1 Gaussian Process Regression

$f \sim GP(\mu, k)$ then: $f|y_{1:n}, x_{1:n} \sim GP(\tilde{\mu}, \tilde{k})$

$\tilde{\mu}(x) = \mu(x) + K_{A,x}^T(K_{AA} + \epsilon I_n)^{-1}(y_A - \mu_A)$

$\tilde{k}(x,x') = k(x,x') - K_{A,x}^T(K_{AA} + \epsilon I_n)^{-1}K_{A,x'}$

Where: $K_{A,x} = [k(x_1,x)\dots k(x_n,x)]^T$

$[K_{AA}]_{ij} = k(x_i, x_j)$ and $\mu_A = [\mu(x_1\dots x_n)]^T$

3.2 Kernels

$k(x,y)$ is a kernel if it's symmetric semidefinite positive:

$\forall\{x_1,\dots,x_n\}$ then for the Gram Matrix $[K]_{ij} = k(x_i,x_j)$ holds $c^TKc \geq 0 \forall c$

Some Kernels: (h is the bandwidth hyperp.)

Gaussian (rbf): $k(x,y) = \exp(-\frac{\|x-y\|^2}{h^2})$

Exponential: $k(x,y) = \exp(-\frac{\|x-y\|}{h})$

Linear kernel: $k(x,y) = x^Ty$ (here $K_{AA} = XX^T$)

3.3 Optimization of Kernel Parameters

Given a dataset A, a kernel function $k(x,y;\theta)$.

$y \sim N(0, K_y(\theta))$ where $K_y(\theta) = K_{AA}(\theta) + \sigma_n^2 I$

$\hat{\theta} = \arg\max_\theta \log p(y|X;\theta)$

In GP: $\hat{\theta} = \arg\min_\theta y^TK_y^{-1}(\theta)y + \log|K_y(\theta)|$

We can from here $\nabla\downarrow$:

$\nabla_\theta \log p(y|X;\theta) = \frac{1}{2}tr\left((\alpha\alpha^T - K^{-1})\frac{\partial K}{\partial\theta}\right), \alpha = K^{-1}y$

Or we could also be baysian about θ

3.4 Aproximation Techniques

Local method: $k(x_1,x_2) = 0$ if $\|x_1 - x_2\| \gg 1$

Random Fourier Features: if $k(x,y) = \kappa(x-y)$ $p(w) = \mathcal{F}\{\kappa(\cdot), w\}$. Then $p(w)$ can be normalized to be a density.

$\kappa(x-y) = \mathbb{E}_{p(w)}\left[\exp\left\{iw^T(x-y)\right\}\right]$ antitransform

$\kappa(x-y) = \mathbb{E}_{b\sim\mathcal{U}([0,2\pi]), w\sim p(w)}[z_{w,b}(x)z_{w,b}(y)]$

where $z_{w,b}(x) = \sqrt{2}cos(w^Tx+b)$. I can MC extract features z. If # features is $\ll$ n then this is faster (X^TX vs XX^T)

Inducing points: We a vector of inducing variables u

$f_A|_u \sim N(K_{Au}K_u u^{-1}u, K_{AA} - K_{Au}K_u u^{-1}K_{uA})$

$f_*|_u \sim N(K_{*u}K_u u^{-1}u, K_{**} - K_{*u}K_u u^{-1}K_{u*})$

Subset of Regressors (SoR): ■ $\rightarrow 0$

FITC: ■ $\rightarrow$ its diagonal

4 Review of useful concepts and Introduction

4.1 Multivariate Gaussian

$f(x) = \frac{1}{2\pi\sqrt{|\Sigma|}}e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)}$

Suppose we have a Gaussian random vector

$\begin{bmatrix} X_A \\ X_B \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix}\mu_A \\ \mu_B\end{bmatrix}, \begin{bmatrix}\Sigma_{AA} & \Sigma_{AB} \\ \Sigma_{BA} & \Sigma_{BB}\end{bmatrix}\right) \Rightarrow X_A|X_B = x_B \sim$

$\mathcal{N}\left(\mu_A + \Sigma_{AB}\Sigma_{BB}^{-1}(x_B - \mu_B), \Sigma_{AA} - \Sigma_{AB}\Sigma_{BB}^{-1}\Sigma_{BA}\right)$

4.2 Convex / Jensen's inequality

g(x) is convex $\Leftrightarrow x_1, x_2 \in \mathbb{R}, \lambda \in [0,1] : g''(x) > 0$

$g(\lambda x_1 + (1-\lambda)x_2) \leq \lambda g(x_1) + (1-\lambda)g(x_2)$

$\varphi(E[X]) \leq E[\varphi(X)]$

4.3 Information Theory elements:

Entropy: $H(X) \doteq -\mathbb{E}_{x\sim p_X}[\log p_X(x)]$

$H(X|Y) \doteq -\mathbb{E}_{(x,y)\sim p_{(X,Y)}}\left[\log p_{Y|X}(y|x)\right]$

if $X \sim \mathcal{N}(\mu,\Sigma) \Rightarrow H(X) = \frac{1}{2}\log\left[(2\pi e)^d\det(\Sigma)\right]$

Chain Rule: $H(X,Y) = H(Y|X) + H(X)$

Mutual Info: $I(X,Y) \doteq KL(p_{(X,Y)}\|p_Xp_Y)$

$I(X,Y) = H(X) - H(X|Y)$

if $X \sim \mathcal{N}(\mu,\Sigma)$, $Y = X + \epsilon$, $\epsilon \sim \mathcal{N}(0,\sigma^2 I)$:

then $I(X,Y) = \frac{1}{2}\log\left[\det(I + \frac{1}{\sigma^2}\Sigma)\right]$

4.4 Kullback-Leiber divergence

$KL(p\|q) = \mathbb{E}_p\left[\log\frac{p(x)}{q(x)}\right]$

if $p_0 \sim \mathcal{N}(\mu_0,\Sigma_0)$, $p_1 \sim \mathcal{N}(\mu_1,\Sigma_1) \Rightarrow KL(p_0\|p_1)$

$= \frac{1}{2}\left(tr(\Sigma_1^{-1}\Sigma_0) + (\mu_1-\mu_0)^T\Sigma_1^{-1}(\mu_1-\mu_0) - k + \log\frac{|\Sigma_1|}{|\Sigma_0|}\right)$

$\hat{q} = \arg\min_q KL(p\|q) \Rightarrow$ overconservative

$\hat{q} = \arg\min_q KL(q\|p) \Rightarrow$ overconfident

5 Approximate inference

5.1 Laplace Approximation

$\hat{\theta} = \arg\max_\theta p(\theta|y)$

$\Lambda = -\nabla_\theta\nabla_\theta \log p(\theta|y)|_{\theta=\hat{\theta}}$

$p(\theta|y) \simeq q(\theta) = N(\hat{\theta}, \Lambda^{-1})$

5.2 Variationa Inverence

$\hat{q} = \arg\min_{q\in Q} KL(q\|p(\cdot|y))$

$\hat{q} = \arg\max_{q\in Q} ELBO$ Evidence Lower Bound

$ELBO \doteq \mathbb{E}_{\theta\sim q}[\log p(y|\theta)] - KL(q\|p(\cdot)) \leq \log p(y)$

5.3 Markov Chain Monte Carlo

Idea: All we need is sampling from postirior

Ergodic Markov Chain:

$\exists t$ s.t. $\mathbb{P}(i \rightarrow j$ in t steps$) > 0\ \forall i,j \Rightarrow$

$\exists!\pi = \lim_{N\rightarrow\infty}\mathbb{P}(X_n = x)$ Limit distribution

Ergodic Theorem: if $(X_i)_{i\in\mathbb{N}}$ is ergodic:

$\lim_{N\rightarrow\infty}\frac{1}{n}\sum_{i=1}^N f(X_i) = \mathbb{E}_{x\sim\pi}[f(x)]$

Detailed Blanced Equation:

$P(x|x')$ is the transition model of a MC:

if $R(x)P(x'|x) = R(x')P(x|x')$ then R is the limit distribution of the MC

Metropolis Hastings Algo: Sample from a MC which has $P(x) = \frac{Q(x)}{Z}$ as limit dist.

Result: $\{X_i\}_{i\in\mathbb{N}}$ sampled from the MC
init: $R(x|x')$
/* Good R choice → fast convergence */
init: $X_0 = x_0$
for $t \leftarrow 1, 2, \dots$ **do**
  $x' \sim R(\cdot, x_{t-1})$
  $\alpha = \min\left\{1; \frac{Q(x')R(x_{t-1}|x')}{Q(x_{t-1})R(x'|x_{t-1})}\right\}$
  with *probability* α **do**
    $X_t = x'$;
  otherwise $X_t = x_{t-1}$;

Metropolis Adj. Langevin Algo (MALA):

Energy function: $P(x) = \frac{Q(x)}{Z} = \frac{1}{Z}\exp(-f(x))$

We chose: $R(x|x') = \mathcal{N}(x' - \tau\nabla f(x), 2\tau I)$

Stoch. Grad. Langevin Dynamics (SGLD):

We use SGD to Approximate ∇f. Converges also without acceptance step

Hamilton MC: SGD performance improoved by adding momentum (consider last step ∇f)

Gibbs sampling: Practical when $X \in \mathbb{R}^n$

Used when $P(X_{1:n})$ is hard but $P(X_i|X_{-i})$ is easy.

init: $x_0 \in \mathbb{R}^n$; $(x_0^{(B)} = x^{(B)})$ B is our data
for $t = 1, 2, \dots$ **do**
  $x_t = x_{t-1}$
  with $i \sim \mathcal{U}(\{1:n\}\setminus B)*$ **do**
    $x_{t-1}^{(i)} \sim P(x^{(i)}|x^{(-i)})$

$*$ if we do it $\forall i \notin B$ no DBE but more practical

5.4 Variable elimination for MPE (most probable explanation):

With loopy graphs, BP is often **overconfident/oscillates**.

6 Bayesian Neural Nets

Likelihood: $p(y|x;\theta) = \mathcal{N}(f_1(x,\theta), \exp(f_2(x,\theta)))$

Prior: $p(\theta) = \mathcal{N}(0, \sigma_p^2)$

$\theta_{MAP} = \arg\max\log(p(y,\theta))$

6.1 Variation inference:

Usually we use Q = Set of Gaussians

$\hat{q} = \arg\max ELBO$ Reparameterization trick

q approx. the posterior but how to predict?

$p(y^*|x^*,\mathcal{D}) \simeq \frac{1}{m}\sum_{j=1}^m p(y^*|x^*,\theta^{(i)})$, $\theta \sim \hat{q}(\theta)$

Gaussian Mixture distribution: $\mathbb{V}(y^*|x^*,\mathcal{D}) \simeq$

$\simeq \frac{1}{m}\sum_{j=1}^m \sigma^2(x^*,\theta^{(i)}) + \frac{1}{m}\sum_{j=1}^m\left(\mu(x^*,\theta^{(j)} - \overline{\mu}(x^*)\right)$

■ $\rightarrow$Aletoric, ■ $\rightarrow$Epistemic

Dropouts Regularization: Random ignore nodes in SGD iteration: Equivalent to VI with $Q = \left\{q(\cdot|\lambda) = \prod_j q_j(\theta_j|\lambda),\ \lambda \in \mathbb{R}^d\right\}$

where $q_j(\theta_j|\lambda) = p\delta_0(\theta_j) + (1-p)\delta_{\lambda_j}(\theta_j)$

This allows to do Dropouts also in prediction

6.2 MCMC:

MCMC but cannot store all the $\theta^{(i)}$:

1) Subsampling: Only store a subset of the $\theta^{(i)}$

2) Gaussian Aproximation: We only keep:

$\mu_i = \frac{1}{T}\sum_{j=1}^T\theta_i^{(j)}$ and $\sigma_i = \frac{1}{T}\sum_{j=1}^T(\theta_i^{(j)} - \mu_i)^2$

And updete them online.

Predictive Esnable NNs:

Let $\mathcal{D} = \{(x_i,y_i)\}_{i=1:n}$ be our dataset.

Train θ_i^{MAP} on $\mathcal{D}_i$ with $i = 1,\dots,m$

$\mathcal{D}_i$ is a Bootstrap of $\mathcal{D}$ of same size

and $p(y^*|x^*,\mathcal{D}) \simeq \frac{1}{m}\sum_{j=1}^m p(y^*|x^*,\theta_i^{MAP})$

6.3 Model calibration

Train $\hat{q}$ on $\mathcal{D}_{train}$

Evaluate $\hat{q}$ on $\mathcal{D}_{val} = \{(y',x')\}_{i=1:m}$

Held-Out-Likelihood $\doteq \log p(y'_{1:m}|x'_{1:m},\mathcal{D}_{train})$

$\geq \mathbb{E}_{\theta\sim\hat{q}}\left[\sum_{i=1}^m \log p(y'_i|x'_i,\theta)\right]$ (Jensen)

$\simeq \frac{1}{k}\sum_{j=1}^k\sum_{i=1}^m \log p(y'_i|x'_i,\theta^{(j)})$, $\theta^{(j)} \sim \hat{q}$

Evaluate predicted accuracy: We divide $\mathcal{D}_{val}$ into bins according to predicted confidence values. In each bin we compare accuracy with confidence

7 Active Learning

Let $\mathcal{D}$ be the set of observable points.

We can observe $\mathcal{S} \subseteq \mathcal{D}, |\mathcal{S}| \leq R$

Information Gain: $\hat{\mathcal{S}} = \arg\max_{\mathcal{S}} F(\mathcal{S}) = I(f, y_{\mathcal{S}})$

For GPs: $F(\mathcal{S}) = \frac{1}{2}\log\left|I + \frac{1}{\sigma^2}K_{\mathcal{SS}}\right|$

This is NP Hard, $\Rightarrow$ Greedy Algo:

init: $\mathcal{S}^* = \emptyset$
for $t = 1:R$ **do**
  $x_t = \arg\max_{x\in\mathcal{D}} F(\mathcal{S}^* \cup \{x\})$
  $\left(x_t = \arg\max_{x\in\mathcal{D}}\sigma_x^2|\mathcal{S} \text{ for GPs}\right)$
  $\left(x_t = \arg\max_{x\in\mathcal{D}}\frac{\sigma_{f|\mathcal{S}}^2(x)}{\sigma_n^2(x)} \text{ for heter. GPs}\right)$
  $\mathcal{S}^* = \mathcal{S} \cup \{x_t\}$

F is **Submodular** if: $\forall x \in \mathcal{D}$, $\forall A \subseteq B \subseteq D$ holds that: $F(A \cup \{x\}) - F(A) \geq F(B \cup \{x\}) - F(B)$
F is Submodular $\Rightarrow F(\mathcal{S}^*) \geq \left(1 - \frac{1}{e}\right) F(\hat{\mathcal{S}})$

8 Bayesian Optimization

Like Active Learning but we only want to find the optima. We pick $x_1, x_2, \dots$ from $\mathcal{D}$ and observe $y_i = f(x_t) + \epsilon_t$.

Comulative regret:$R_T = \sum_{t=1}^{T} \left(\max_{x \in \mathcal{D} f(x) - f(x_t)} \right)$

Oss:$\frac{R_T}{T} \to 0 \Rightarrow \max_t f(x_t) \to \max_{x \in \mathcal{D}} f(x)$

8.1 Upper Confidence Sampling

With GP $x_t = \arg\max_{x \in \mathcal{D}} \mu_{t-1}(x) + \beta_t \sigma_{t-1}(x)$

Chosing the correct β_t we get: $\frac{R_T}{T} = \mathcal{O}\left(\sqrt{\frac{\gamma_T}{T}}\right)$.

Where $\gamma_t = \max_{|S| < T} I(f; y_S)$. On d dims:

Linear: $\gamma_T = \mathcal{O}(d \log T)$ RBF: $\gamma_T = \mathcal{O}((log T)^{d+1})$

Optimal $\beta_t = \mathcal{O}(\|f\|_K^2 + \gamma_t \log^3 T)$

Oss: $\beta \uparrow$=more exploration

8.2 Thompson Samling

$x_t = \arg\max_{x \in \mathcal{D}} \tilde{f}(x), \quad \tilde{f} \sim p(f | x_{1:n}, y_{1:n})$

9 Markov Decision Process (MDP)

9.1 Definitions

$\mathcal{X} = \{1, \dots, n\}$ states; $\mathcal{A} = \{1, \dots, m\}$ actions;
$p(x'|x, a)$ transition probability;
$r(x, a)$ reward; $\pi : \mathcal{X} \to \mathcal{A}$ policy;
$T^\pi \in \mathbb{R}^{n \times n}$, $T_{ij}^\pi = p(j|i, \pi(i))$ Transition Matrix:
$J(\pi) = \mathbb{E}\left[\sum_{i=0}^\infty \gamma^i r(X_i, \pi(X_i))\right]$ Expected value:
$V^\pi : \mathcal{X} \to \mathbb{R}$, $x \mapsto J(\pi | X_0 = x)$ Value function;
$Q^V(x, a) = r(x, a) + \gamma \sum_{x \in \mathcal{X}} p(x'|x, a) V(x)$ Q func;
$\pi_{\mathcal{G}}^V(x) = \arg\max_a Q^V(x, a)$ greedy policy w.r.t. V;

9.2 Value function Theorem

$V^\pi(x) = r(x, \pi(x)) + \gamma \sum_{x' \in \mathcal{X}} p(x'|x, \pi(x)) V^\pi(x')$
Matrix formulation: $(I - \gamma T^\pi) V^\pi = r^\pi$

9.3 Bellman Theorem

1) π^*, V^* are optimal policy and it's value func.
2) $\pi^* = \pi_{\mathcal{G}}^{V^*}$
3) $V^*(x) = max_a [r(x, a) + \gamma \sum_{x \in \mathcal{X}} p(x'|x, a) V^*(x)]$
1) $\Leftrightarrow$ 2) $\Leftrightarrow$ 3)

9.4 Algorithms

Policy iteration:

while *no more changes* **do**
  $\pi \leftarrow \pi_{\mathcal{G}}^V$ (Update the Policy)
  $V \leftarrow (I - \gamma T^\pi)^{-1} r^\pi)$ (Update the value)

Value iteration:

while $\|V_t - V_{t-1}\| \leq \epsilon$ **do**
  foreach $x \in \mathcal{X}$, $a \in \mathcal{A}$ **do**
    $Q_t(x, a) \leftarrow r(x, a) + \gamma \sum_{x' \in \mathcal{X}} p(x'|x, a) V_{t-1}(x)$
  foreach $x \in \mathcal{X}$ **do**
    $V_t(x) \leftarrow \max_a Q_t(x, a)$

$\hat{\pi} = \pi_{\mathcal{G}}^{V_T}$; where V_T last found Value

9.5 Partialy Observable MDP (POMDP)

POMDP can be seen as MDP where:
1) $\mathcal{X}_{POMDP}$ are prob. distribution over $\mathcal{X}_{MDP}$
2) the actions are the same
3) $r_{POMDP}(b, a) = \mathbb{E}_{x \sim b}[r_{MDP}(x, a)]$
4) Trans. model: $b_{t+1}(x) = \mathbb{P}(X_{t+1} = x | y_{1:t+1}, a_t)$
$b_{t+1}(x) = \frac{1}{Z} p(y_{t+1} | X_{t+1} = x) \sum_{x \in \mathcal{X}_M DP} p(x|x', a_t) b_t(x')$
How to solve? Discretize $\mathcal{X}_{POMDP}$ and treat it as a MDP or Policy gradient techniques

10 Reinforcement Learning

It is an MDP with unknown $p(x'|x, a)$ and $r(x, a)$

10.1 Model-based RL

10.1.1 ϵ greedy

With probability ϵ, pick random action. With prob $(1 - \epsilon)$, pick best action. If sequence ϵ satisfies Robbins Monro criteria $\to$ convergence to optimal policy with prob 1.

10.1.2 R_{max} algorithm

Input: starting x_0, discount factor γ.
Initially: add fairy tale state x^* to MDP
- Set $r(x, a) = R_{max}$ for all states x and actions a
- Set $P(x^*|x, a) = 1$ for all states x and actions a
- Choose the optimal policy for r and P

Repeat: 1. Execute policy π and, for each visited state/action pair, update $r(x, a)$
2. Estimate transition probabilities $P(x'|x, a)$
3. If observed 'enough' transitions/rewards, recompute policy π, according to current model P and r.
Ënough"? See Hoeffding's inequality. To reduce error ϵ, need more samples N.
Theorem: With probability $1 - \delta$, R_{max} will reach an ϵ-optimal policy in a number of steps that is polynomial in $|X|, |A|, T, 1/\epsilon$ and $log(1/\delta)$. Memory $O(|X^2||A|)$.

10.2 Model-free RL: estimate V*(x) directly

10.2.1 Q-learning

$Q(x, a) \leftarrow (1 - \alpha_t) Q(x, a) + \alpha_t (r + \gamma \max_{a'} Q(x', a'))$
Theorem: If learning rate α_t satisfies: $\sum_t \alpha_t = \infty$ and $\sum_t \alpha_t^2 < \infty$ (Robbins-Monro), and actions are chosen at random, then Q learning converges to optimal Q^* with probability 1.
Optimistic Q learning:
Initialize: $Q(x, a) = \frac{R_{max}}{1-\gamma} \prod_{t=1}^{T_{init}} (1 - \alpha_t)^{-1}$
Same convergence time as with R_{max}. Memory $O(|X||A|)$. Comp: $O(|A|)$.
Parametric Q-function approximation: $Q(x, a; \theta) = \theta^T \phi(x, a)$ to scale to large state spaces. (You can use Deep NN here!)
SGD for ANNs: initialize weights. For t = 1,2..., pick a data point (x,y) uniformly at random. Take step in negative gradient direction. (In practise, mini-batches).
Deep Q Networks: use CNN to approx Q function. $L(\theta) = \sum_{(x,a,r,x') \in D} (r + \gamma \max_{a'} Q(x', a'; \theta^{old}) - Q(x, a; \theta))^2$ **Double DQN:** current network for evaluating argmax (too optimistic, and you remove θ^{old} and put θ).

10.3 Gaussian processes

A GP is an (infinite) set of random variables (RV), indexed by some set X, i.e., for each x in X, there is a RV Y_x where there exists functions $\mu : X \to \mathbb{R}$ and $K : X \times X \to \mathbb{R}$ such that for all: $A \in X$, $A = x_1, \dots x_k$, it holds that $Y_A = [Y_{x_1}, \dots, Y_{x_k}] \sim N(\mu_a, \Sigma_{AA})$, where: $\Sigma_{AA} =$ matrix with all combinations of $K(x_i, x_j)$.
K is called kernel (covariance) function (must be symmetric and pd) and μ is called mean function. **Making prediction with GPs:** Suppose $P(f) = GP(f; \mu, K)$ and we observe $y_i = f(\vec{x_i}) + \epsilon_i$, $A = \{\vec{x_1} : \vec{x_k}\}$ $P(f(x) | \vec{x_1} : \vec{x_k}, y_{1:k}) = GP(f; \mu', K')$. In particular, $P(f(x) | \vec{x_1} : \vec{x_k}, y_{1:k}) = N()f(x); \mu_{x|A}, \sigma_{x|a}^2$, where $\mu_{x|a} = \mu(\vec{x}) + \Sigma_{x,A}(\Sigma_{AA} + \sigma^2 I)^{-1} \Sigma_{x,A}^T (\vec{y_A} - \mu_A)$ and $\sigma_{x|a}^2 = K(\vec{x}, \vec{x}) - \Sigma_{x,A}(\Sigma_{AA} + \sigma^2 I)^{-1} \Sigma_{x,A}^T$.
Closed form formulas for prediction!