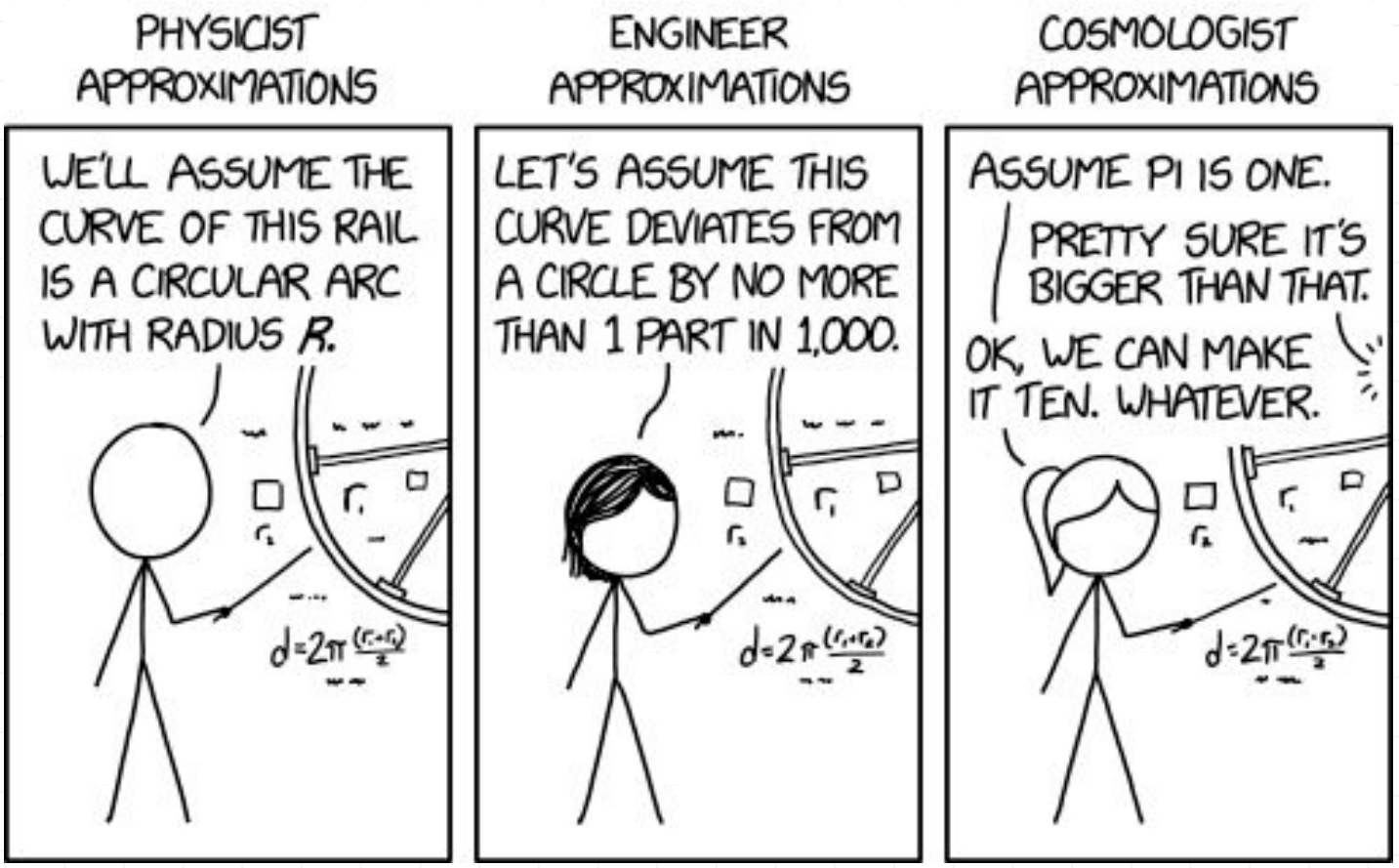
https://xkcd.com/2205/



Announcements

Released: Quiz 2, Assignment 2, video assignment

Next three lectures: graphical models

Week 10 Wed: guest lecture

Approximate Inference + GP Classification

Laplace approximation - in general

Laplace approximation - Bayesian logistic regression

GP classification

Laplace approximation - GP classification

Connection to neural networks

Bishop, Chap 4.4, 4.5
6.4 (6.4.5, part of 6.4.6, 6.4.7)

GP book chap 3
http://gaussianprocess.org/gpml/chapters/

Laplace approximation in general

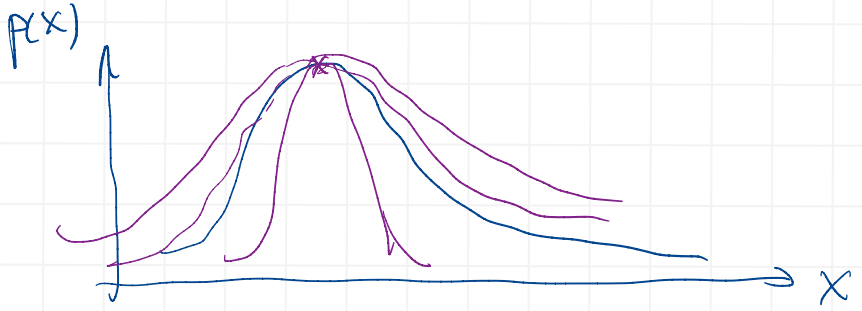
[Bishop 4.4]

Goal: find a Gaussian approximation to a probability density defined over a set of continuous variables.

How: find Gaussian pdf q(z), centred on a mode of the distribution p(z)

Consider pdf

$$p(z) = \frac{1}{Z} f(z) \qquad (4.125) \qquad \text{where } Z = \int f(z)\,\mathrm{d}z$$



Gaussian pdf ⇔ log(q) quadratic

$$\mathcal{N}(x|\mu,\sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\} \tag{2.42}$$

$$p(z) = \frac{1}{Z} f(z) \tag{4.125}$$

where $Z = \int f(z)\,\mathrm{d}z$

Find mode z_0

$$\left.\frac{df(z)}{dz}\right|_{z=z_0} = 0. \tag{4.126}$$

Taylor expansion of *ln f(z)* at z_0

$$\ln f(z) \simeq \ln f(z_0) - \frac{1}{2}A(z-z_0)^2 \tag{4.127}$$

$$A = -\left.\frac{d^2}{dz^2} \ln f(z)\right|_{z=z_0}. \tag{4.128}$$

Take exp()

$$f(z) \simeq f(z_0) \exp\left\{-\frac{A}{2}(z-z_0)^2\right\}. \tag{4.129}$$

Normalise to obtain q(z)

$$q(z) = \left(\frac{A}{2\pi}\right)^{1/2} \exp\left\{-\frac{A}{2}(z-z_0)^2\right\}. \tag{4.130}$$

Assume A > 0

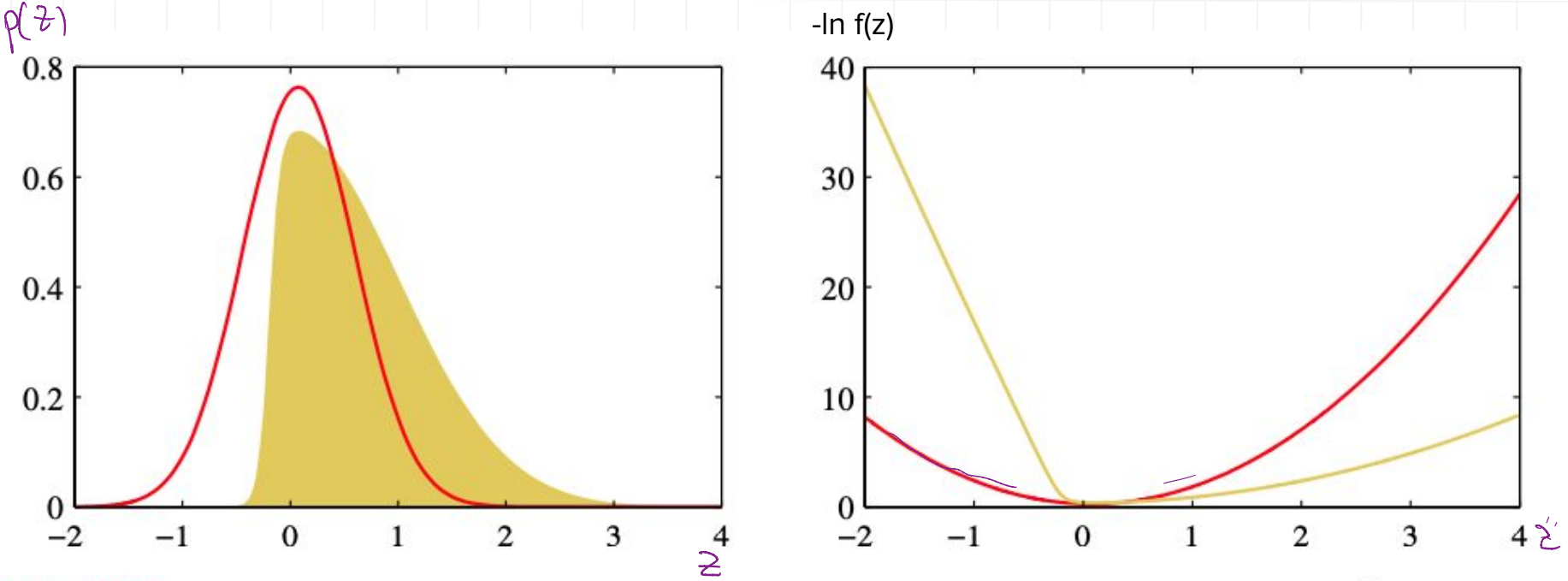


Figure 4.14 Illustration of the Laplace approximation applied to the distribution $p(z) \propto \exp(-z^2/2)\sigma(20z + 4)$ where $\sigma(z)$ is the logistic sigmoid function defined by $\sigma(z) = (1 + e^{-z})^{-1}$. The left plot shows the normalized distribution $p(z)$ in yellow, together with the Laplace approximation centred on the mode z_0 of $p(z)$ in red. The right plot shows the negative logarithms of the corresponding curves.

Laplace approximation in higher dimensions

$p(\vec{z}) = \frac{1}{Z} f(\vec{z})$

f(z), with z_0 a stationary point $\nabla f(z)|_{z=z_0} = 0$

Taylor expansion of ln f(**z**) around z_0

$$\ln f(\mathbf{z}) \simeq \ln f(\mathbf{z}_0) - \frac{1}{2}(\mathbf{z}-\mathbf{z}_0)^{\mathrm{T}}\mathbf{A}(\mathbf{z}-\mathbf{z}_0) \tag{4.131}$$

matrix

where the $M \times M$ Hessian matrix $\mathbf{A}$ is defined by

$$\mathbf{A} = -\nabla\nabla \ln f(\mathbf{z})|_{\mathbf{z}=\mathbf{z}_0} \tag{4.132}$$

and ∇ is the gradient operator. Taking the exponential of both sides we obtain

$$f(\mathbf{z}) \simeq f(\mathbf{z}_0)\exp\left\{-\frac{1}{2}(\mathbf{z}-\mathbf{z}_0)^{\mathrm{T}}\mathbf{A}(\mathbf{z}-\mathbf{z}_0)\right\}. \tag{4.133}$$

normalise

Laplace approximation in higher dimensions

$$f(\mathbf{z}) \simeq f(\mathbf{z}_0) \exp\left\{-\frac{1}{2}(\mathbf{z}-\mathbf{z}_0)^{\mathrm{T}}\mathbf{A}(\mathbf{z}-\mathbf{z}_0)\right\}. \qquad (4.133)$$

normalise, use (2.43) $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}}\frac{1}{|\boldsymbol{\Sigma}|^{1/2}}\exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}$

$$q(\mathbf{z}) = \frac{|\mathbf{A}|^{1/2}}{(2\pi)^{M/2}}\exp\left\{-\frac{1}{2}(\mathbf{z}-\mathbf{z}_0)^{\mathrm{T}}\mathbf{A}(\mathbf{z}-\mathbf{z}_0)\right\} = \mathcal{N}(\mathbf{z}|\mathbf{z}_0, \mathbf{A}^{-1}) \qquad (4.134)$$

q(**z**) is a valid multivariate Gaussian distribution iff A positive semi-definite

➜ z_0 is a local maximum, not local min or saddle point

What about f(z) with multiple modes? Different Laplace approximations for each mode

Pros + cons of Laplace approximation

$$q(\mathbf{z}) = \frac{|\mathbf{A}|^{1/2}}{(2\pi)^{M/2}} \exp\left\{-\frac{1}{2}(\mathbf{z}-\mathbf{z}_0)^{\mathrm{T}}\mathbf{A}(\mathbf{z}-\mathbf{z}_0)\right\} = \mathcal{N}(\mathbf{z}|\mathbf{z}_0, \mathbf{A}^{-1}) \qquad (4.134)$$

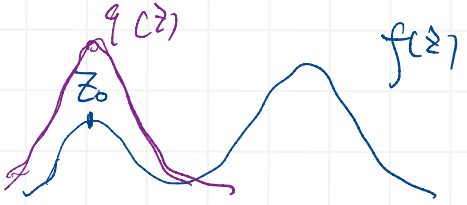
:) Normalisation constant Z for f(z) does not need to be known.

:) CLT → posterior increasingly better approximated by Gaussian as the number of observed data points grow.

if $z>0$ $z \longrightarrow \ln z \in \mathbb{R}$

:(Assumes the domain of z is R, or R^d, may need to transform the r.v.

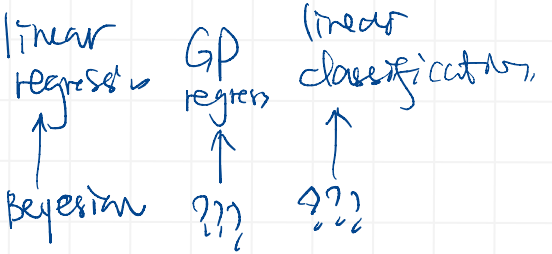
:(Is based purely on f(z) around z_0, no global info



Approximate Inference + GP Classification

Laplace approximation - in general

Laplace approximation - Bayesian logistic regression



GP classification

Laplace approximation - GP classification

Connection to neural networks

Recap: Logistic regression

$$\sigma(a) = \frac{1}{1 + \exp(-a)}.$$

$$p(\mathcal{C}_1|\boldsymbol{\phi}) = y(\boldsymbol{\phi}) = \sigma\left(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}\right) \tag{4.87}$$

data set $\{\boldsymbol{\phi}_n, t_n\}$, where $t_n \in \{0, 1\}$ and $\boldsymbol{\phi}_n = \boldsymbol{\phi}(\mathbf{x}_n)$.

$\sigma(w^T\phi)$ or $1-\sigma(w^T\phi)$

$$p(\mathbf{t}|\mathbf{w}) = \prod_{n=1}^{N} y_n^{t_n} \{1 - y_n\}^{1-t_n} \tag{4.89}$$

Negative log-likelihood, or cross-entropy error function

$$E(\mathbf{w}) = -\ln p(\mathbf{t}|\mathbf{w}) = -\sum_{n=1}^{N} \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\} \tag{4.90}$$

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N} (y_n - t_n)\boldsymbol{\phi}_n \tag{4.91}$$

Bayesian logistic regression

$$p(\mathcal{C}_1|\boldsymbol{\phi}) = y(\boldsymbol{\phi}) = \sigma\left(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}\right) \tag{4.87}$$

$$E(\mathbf{w}) = -\ln p(\mathbf{t}|\mathbf{w}) = -\sum_{n=1}^{N}\{t_n \ln y_n + (1-t_n)\ln(1-y_n)\} \tag{4.90}$$

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_0, \mathbf{S}_0) \tag{4.140}$$

$$p(\mathbf{w}|\mathbf{t}) \propto p(\mathbf{w})p(\mathbf{t}|\mathbf{w}) \tag{4.141}$$

← $p(\vec{t})$ const

$$\ln p(\mathbf{w}|\mathbf{t}) = -\frac{1}{2}(\mathbf{w}-\mathbf{m}_0)^{\mathrm{T}}\mathbf{S}_0^{-1}(\mathbf{w}-\mathbf{m}_0) + \sum_{n=1}^{N}\{t_n \ln y_n + (1-t_n)\ln(1-y_n)\} + \text{const} \tag{4.142}$$

quadratic part of $\mathcal{N}(w|m_0, S_0)$

(come from Ew)

normalizer of $\mathcal{N}(\dots)$ $p(\vec{t})$

Laplace approximation for Bayesian logistic regression

$$\ln p(\mathbf{w}|\mathbf{t}) = -\frac{1}{2}(\mathbf{w}-\mathbf{m}_0)^{\mathrm{T}}\mathbf{S}_0^{-1}(\mathbf{w}-\mathbf{m}_0) + \sum_{n=1}^{N}\{t_n \ln y_n + (1-t_n)\ln(1-y_n)\} + \text{const} \quad (4.142)$$

$\ln \sigma(w^T \phi_n)$

$\ln\left(\frac{1}{1+e^{w^T\phi}}\right)$

① compute local max z_0

$$q(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{w}_{\mathrm{MAP}}, \mathbf{S}_N). \quad (4.144)$$

② compute Hessian around z_0

$$\mathbf{S}_N = -\nabla\nabla \ln p(\mathbf{w}|\mathbf{t}) = \mathbf{S}_0^{-1} + \sum_{n=1}^{N} y_n(1-y_n)\boldsymbol{\phi}_n\boldsymbol{\phi}_n^{\mathrm{T}}. \quad (4.143)$$

$$y(\phi) = \sigma\left(\mathbf{w}^{\mathrm{T}}\phi\right)$$

$$\frac{d\sigma}{da} = \sigma(1-\sigma)$$

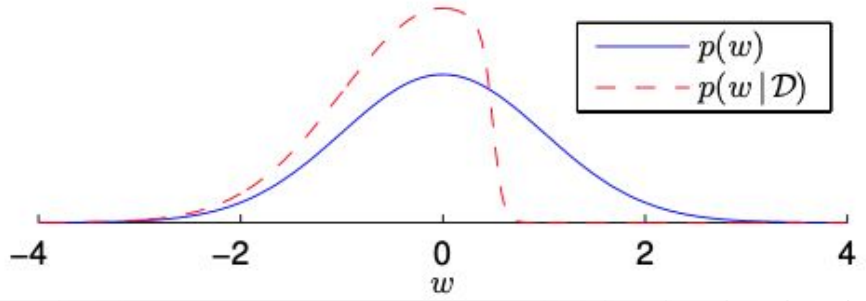
$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N}(y_n - t_n)\phi_n$$

③ $\nabla_w \; t_n \ln \sigma(w^T\phi_n)$

$= t_n \sigma(w^T\phi_n)(1-\sigma(w^T\phi_n)) \frac{1}{\sigma(w^T\phi_n)} \phi_n$

Example 1

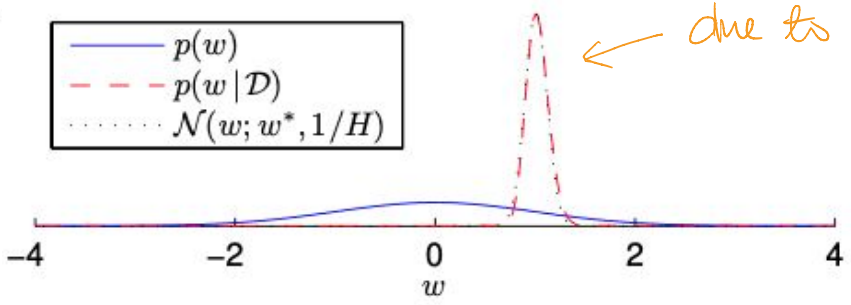
$$p(w) \propto \mathcal{N}(w; 0, 1)$$
$$p(w \mid \mathcal{D}) \propto \mathcal{N}(w; 0, 1)\, \sigma(10 - 20w).$$



Example 2

As another example, I generated $N = 500$ labels, $\{z^{(n)}\}$, from a logistic regression model with no bias and with $w = 1$ at $x^{(n)} \sim \mathcal{N}(0, 10^2)$. Then,

$$p(w) \propto \mathcal{N}(w; 0, 1)$$
$$p(w \mid \mathcal{D}) \propto \mathcal{N}(w; 0, 1) \prod_{n=1}^{500} \sigma(w x^{(n)} z^{(n)}), \qquad z^{(n)} \in \{\pm 1\}.$$



Example courtesy of Edinburgh MLPR course https://www.inf.ed.ac.uk/teaching/courses/mlpr/2016/notes/w8a_bayes_logistic_regression_laplace.pdf

What does the predictive distribution look like?

$$p(\mathcal{C}_1|\boldsymbol{\phi}, \mathbf{t}) = \int p(\mathcal{C}_1|\boldsymbol{\phi}, \mathbf{w}) p(\mathbf{w}|\mathbf{t})\, \mathrm{d}\mathbf{w} \simeq \int \sigma(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}) q(\mathbf{w})\, \mathrm{d}\mathbf{w} \qquad (4.145)$$

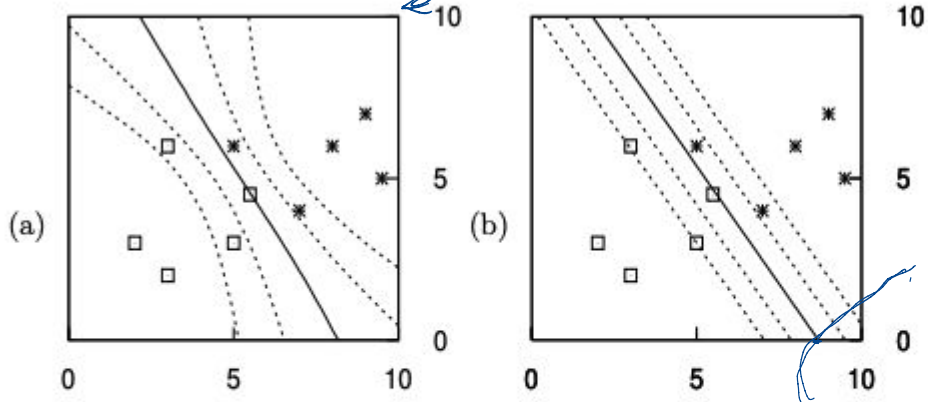
laplace approx.

$$= \int \sigma(a) \mathcal{N}(a|\mu_a, \sigma_a^2)\, \mathrm{d}a.$$

assumes $p(w|t) = \delta(w^*)$

$\sigma(w^{*T}\phi)$

over $\phi = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$



Again, the axes are the input features x_1 and x_2. The right hand figure shows $P(y=1 \mid \mathbf{x}, \mathbf{w}^*)$ for some fitted weights $\mathbf{w}^*$. No matter how these fitted weights are chosen, the contours have to be linear. The parallel contours mean that the uncertainty of predictions falls at the same rate when moving away from the decision boundary, no matter how far we are from the training inputs.

Figure courtesy of Edinburgh MLPR course https://www.inf.ed.ac.uk/teaching/courses/mlpr/2016/notes/w8a_bayes_logistic_regression_laplace.pdf

Approximate Inference + GP Classification

Laplace approximation - in general

Laplace approximation - Bayesian logistic regression

GP classification

Laplace approximation - GP classification

Connection to neural networks

GP for classification

mean fn

covariance fn

$\mathbf{a}(x) \sim GP(\mathbf{0}, \mathbf{K})$

Training set: input $\{\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_N\}$, target variable $\mathbf{t}=\{t_1, t_2, \ldots t_N\}^T$, with $t_i \in \{0, 1\}$

$$p(t|a) = \sigma(a)^t(1-\sigma(a))^{1-t}. \qquad (6.73)$$

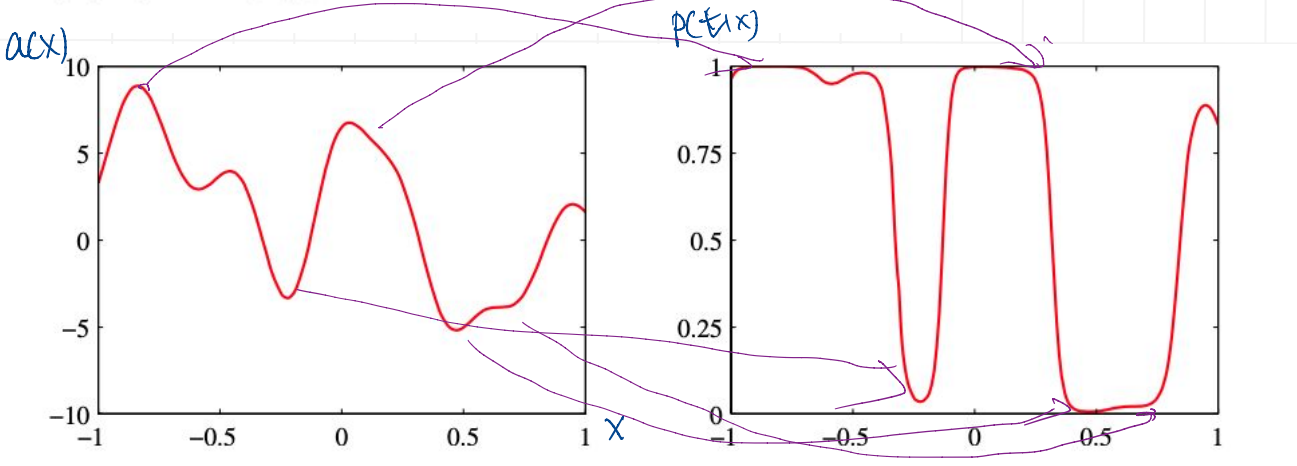
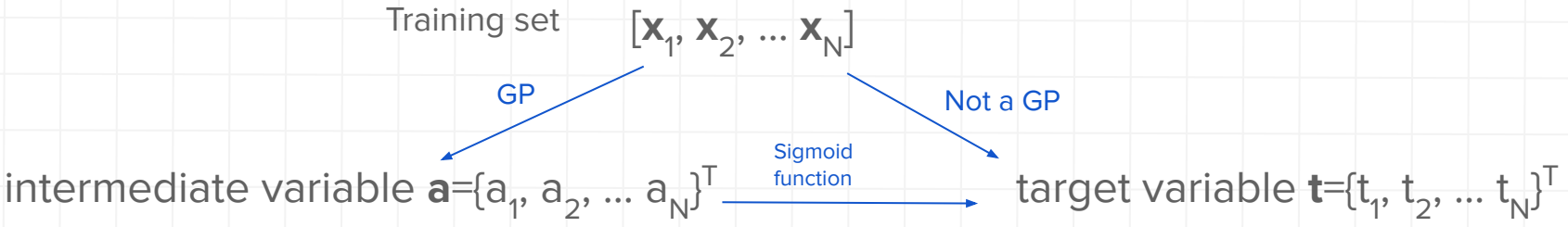


Figure 6.11 The left plot shows a sample from a Gaussian process prior over functions $a(\mathbf{x})$, and the right plot shows the result of transforming this sample using a logistic sigmoid function.

What we just said



For regression

$$p(\mathbf{t}_{N+1}) = \mathcal{N}(\mathbf{t}_{N+1}|\mathbf{0}, \mathbf{C}_{N+1}) \quad (6.64)$$

Rename the variable

$$p(\mathbf{a}_{N+1}) = \mathcal{N}(\mathbf{a}_{N+1}|\mathbf{0}, \mathbf{C}_{N+1}). \quad (6.74)$$

Covariance function of the GP **a**

last week

$C(x_n, x_m) = k(x_n, x_m)$

$+ \beta^{-1} \delta_{nm}$.

noise.

$$p(\mathbf{a}_{N+1}) = \mathcal{N}(\mathbf{a}_{N+1}|\mathbf{0}, \mathbf{C}_{N+1}). \tag{6.74}$$

- Assume it's noise-less
- BUT add diagonal term to ensure that it's positive semi-definite

$$C(\mathbf{x}_n, \mathbf{x}_m) = k(\mathbf{x}_n, \mathbf{x}_m) + \nu\delta_{nm} \tag{6.75}$$

Assume kernel function k(x, x' ; $\boldsymbol{\theta}$)

How to do prediction?

assume: x_{N+1} hidden.

Desired: $p(t_{N+1} = 1|\mathbf{t}_N)$

Have: $p(t_{N+1} = 1|a_{N+1}) = \sigma(a_{N+1})$

Plug in GP prediction

$$m(\mathbf{x}_{N+1}) = \mathbf{k}^{\mathrm{T}}\mathbf{C}_N^{-1}\mathbf{t} \quad (6.66)$$
$$\sigma^2(\mathbf{x}_{N+1}) = c - \mathbf{k}^{\mathrm{T}}\mathbf{C}_N^{-1}\mathbf{k}. \quad (6.67)$$

x is "hidden" inside kernels **k** and C

$$p(a_{N+1}|\mathbf{a}_N) = \mathcal{N}(a_{N+1}|\mathbf{k}^{\mathrm{T}}\mathbf{C}_N^{-1}\mathbf{a}_N, c - \mathbf{k}^{\mathrm{T}}\mathbf{C}_N^{-1}\mathbf{k}). \quad (6.78)$$

Use Bayes rule:

$$p(t_{N+1} = 1|\mathbf{t}_N) = \int p(t_{N+1} = 1|a_{N+1})p(a_{N+1}|\mathbf{t}_N)\,\mathrm{d}a_{N+1} \quad (6.76)$$

assumes $t_{N+1} \perp t_N$ given a_{N+1}

Computing posterior for GP classification

predictive.

$$p(t_{N+1} = 1|\mathbf{t}_N) = \int p(t_{N+1} = 1|a_{N+1}) p(a_{N+1}|\mathbf{t}_N)\, da_{N+1} \qquad (6.76)$$

$\sigma(a_{N+1})$

want to use $p(a_{N+1}|\mathbf{a}_N)$

$$\begin{aligned} p(a_{N+1}|\mathbf{t}_N) &= \int p(a_{N+1}, \mathbf{a}_N|\mathbf{t}_N)\, d\mathbf{a}_N \\ &= \frac{1}{p(\mathbf{t}_N)} \int p(a_{N+1}, \mathbf{a}_N) p(\mathbf{t}_N|a_{N+1}, \mathbf{a}_N)\, d\mathbf{a}_N \\ &= \frac{1}{p(\mathbf{t}_N)} \int p(a_{N+1}|\mathbf{a}_N) p(\mathbf{a}_N) p(\mathbf{t}_N|\mathbf{a}_N)\, d\mathbf{a}_N \\ &= \int p(a_{N+1}|\mathbf{a}_N) p(\mathbf{a}_N|\mathbf{t}_N)\, d\mathbf{a}_N \qquad (6.77) \end{aligned}$$

Bayes rule,

const

$p(\mathbf{t}_N|a_{N+1}, \mathbf{a}_N) = p(\mathbf{t}_N|\mathbf{a}_N)$

(x_n, t_n) known

$p(a_n)$ prior known.

Laplace approximation

$$p(a_{N+1}|\mathbf{a}_N) = \mathcal{N}(a_{N+1}|\mathbf{k}^{\mathrm{T}}\mathbf{C}_N^{-1}\mathbf{a}_N, c - \mathbf{k}^{\mathrm{T}}\mathbf{C}_N^{-1}\mathbf{k}). \qquad (6.78)$$

Want to approximate $p(\mathbf{a}_N|\mathbf{t}_N)$

$p(t_N|a_N)$ → logistic reg.

$$P(\mathbf{a}_N) = \mathcal{N}(\mathbf{a}_N \mid \mathbf{0}, C_N)$$

expand + simplify.

$$p(\mathbf{t}_N|\mathbf{a}_N) = \prod_{n=1}^{N} \sigma(a_n)^{t_n}(1-\sigma(a_n))^{1-t_n} = \prod_{n=1}^{N} e^{a_n t_n}\sigma(-a_n). \tag{6.79}$$

$1+e^{-a_n}$

Taylor expansion of $p(\mathbf{a}_N|\mathbf{t}_N)$

const

$-\ln p(\vec{t}_N)$

$$\begin{aligned}\Psi(\mathbf{a}_N) &= \ln p(\mathbf{a}_N) + \ln p(\mathbf{t}_N|\mathbf{a}_N) \\ &= -\frac{1}{2}\mathbf{a}_N^{\mathrm{T}}\mathbf{C}_N^{-1}\mathbf{a}_N - \frac{N}{2}\ln(2\pi) - \frac{1}{2}\ln|\mathbf{C}_N| + \mathbf{t}_N^{\mathrm{T}}\mathbf{a}_N \\ &\quad - \sum_{n=1}^{N}\ln(1+e^{a_n}) + \text{const.}\end{aligned} \tag{6.80}$$

Laplace approximation of $p(\mathbf{a}_N|\mathbf{t}_N)$

$$\begin{aligned}\Psi(\mathbf{a}_N) &= \ln p(\mathbf{a}_N) + \ln p(\mathbf{t}_N|\mathbf{a}_N) \\ &= -\frac{1}{2}\mathbf{a}_N^{\mathrm{T}}\mathbf{C}_N^{-1}\mathbf{a}_N - \frac{N}{2}\ln(2\pi) - \frac{1}{2}\ln|\mathbf{C}_N| + \mathbf{t}_N^{\mathrm{T}}\mathbf{a}_N \\ &\quad - \sum_{n=1}^{N}\ln(1+e^{a_n}) + \text{const.} \end{aligned} \tag{6.80}$$

$$\nabla\Psi(\mathbf{a}_N) = \mathbf{t}_N - \boldsymbol{\sigma}_N - \mathbf{C}_N^{-1}\mathbf{a}_N \tag{6.81}$$

$$\nabla\nabla\Psi(\mathbf{a}_N) = -\mathbf{W}_N - \mathbf{C}_N^{-1} \tag{6.82}$$

$\sigma_N = [\sigma(a_n)]$

$\mathbf{W}_N = diag[\sigma(a_n)(1-\sigma(a_n))]$

Claim: - $\nabla\nabla\Psi(\mathbf{a}_N)$: positive definite, but posterior is non-Gaussian (since $\mathbf{W}_N$ depend on $\mathbf{a}_N$)

$$q(\mathbf{a}_N) = \mathcal{N}(\mathbf{a}_N|\mathbf{a}_N^\star, \mathbf{H}^{-1}). \tag{6.86}$$

$$\mathbf{a}_N^\star = \mathbf{C}_N(\mathbf{t}_N - \boldsymbol{\sigma}_N). \tag{6.84}$$

$$\mathbf{H} = -\nabla\nabla\Psi(\mathbf{a}_N) = \mathbf{W}_N + \mathbf{C}_N^{-1} \tag{6.85}$$

valid Laplace approximation.

Q: what do we use for $\sigma(a_n)$

Bringing it back together

$$p(t_{N+1} = 1|\mathbf{t}_N) = \int p(t_{N+1} = 1|a_{N+1}) p(a_{N+1}|\mathbf{t}_N)\, \mathrm{d}a_{N+1} \qquad (6.76)$$

$$\int p(a_{N+1}|\mathbf{a}_N) p(\mathbf{a}_N|\mathbf{t}_N)\, \mathrm{d}\mathbf{a}_N \qquad (6.77)$$

$\sigma(a_{N+1})$

$$\mathcal{N}(a_{N+1}|\mathbf{k}^{\mathrm{T}}\mathbf{C}_N^{-1}\mathbf{a}_N, c - \mathbf{k}^{\mathrm{T}}\mathbf{C}_N^{-1}\mathbf{k})$$

$$q(\mathbf{a}_N) = \mathcal{N}(\mathbf{a}_N|\mathbf{a}_N^\star, \mathbf{H}^{-1}).$$

$$\mathbf{a}_N^\star = \mathbf{C}_N(\mathbf{t}_N - \boldsymbol{\sigma}_N).$$

$$\mathbf{H} = -\nabla\nabla\Psi(\mathbf{a}_N) = \mathbf{W}_N + \mathbf{C}_N^{-1}$$

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{y}|\mathbf{A}\boldsymbol{\mu} + \mathbf{b}, \mathbf{L}^{-1} + \mathbf{A}\boldsymbol{\Lambda}^{-1}\mathbf{A}^{\mathrm{T}}) \qquad (2.115)$$

$$\mathbb{E}[a_{N+1}|\mathbf{t}_N] = \mathbf{k}^{\mathrm{T}}(\mathbf{t}_N - \boldsymbol{\sigma}_N) \qquad (6.87)$$

$$\mathrm{var}[a_{N+1}|\mathbf{t}_N] = c - \mathbf{k}^{\mathrm{T}}(\mathbf{W}_N^{-1} + \mathbf{C}_N)^{-1}\mathbf{k}. \qquad (6.88)$$

Bringing it back together

$$p(t_{N+1}=1|\mathbf{t}_N)=\int p(t_{N+1}=1|a_{N+1})p(a_{N+1}|\mathbf{t}_N)\,\mathrm{d}a_{N+1} \tag{6.76}$$

$\sigma(a_{N+1})$

$$\begin{aligned}\mathbb{E}[a_{N+1}|\mathbf{t}_N] &= \mathbf{k}^{\mathrm{T}}(\mathbf{t}_N-\boldsymbol{\sigma}_N) &(6.87)\\ \mathrm{var}[a_{N+1}|\mathbf{t}_N] &= c-\mathbf{k}^{\mathrm{T}}(\mathbf{W}_N^{-1}+\mathbf{C}_N)^{-1}\mathbf{k}. &(6.88)\end{aligned}$$

$$\int \sigma(a)\mathcal{N}(a|\mu,\sigma^2)\,\mathrm{d}a \simeq \sigma\left(\kappa(\sigma^2)\mu\right) \tag{4.153}$$

$$\kappa(\sigma^2)=(1+\pi\sigma^2/8)^{-1/2}.$$

$\mathbf{k}^{\mathrm{T}}(\mathbf{t}_N-\boldsymbol{\sigma}_N)$

$c-\mathbf{k}^{\mathrm{T}}(\mathbf{W}_N^{-1}+\mathbf{C}_N)^{-1}\mathbf{k}$

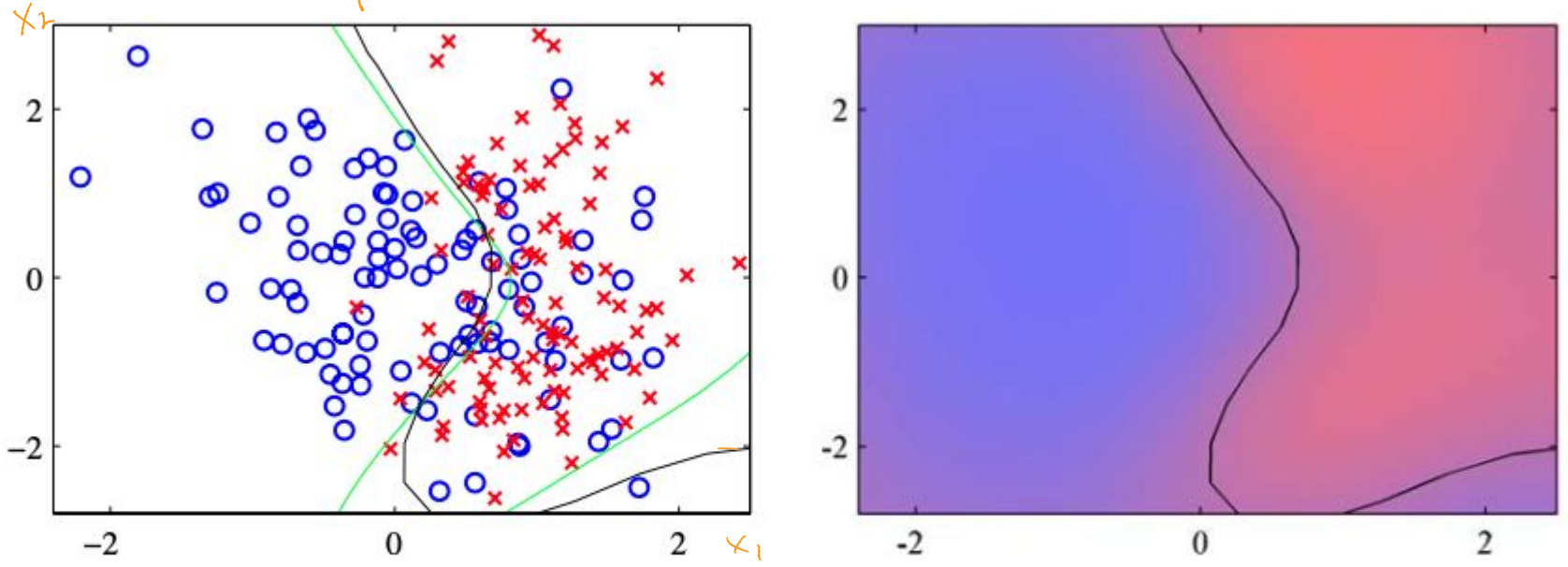
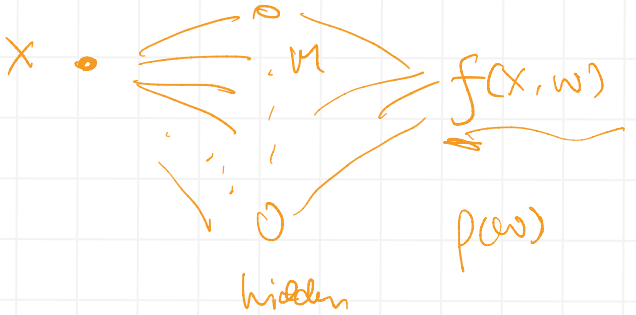


Figure 6.12 Illustration of the use of a Gaussian process for classification, showing the data on the left together with the optimal decision boundary from the true distribution in green, and the decision boundary from the Gaussian process classifier in black. On the right is the predicted posterior probability for the blue and red classes together with the Gaussian process decision boundary.

Connection to neural nets



2-layer neural network, with M hiddent units

Bayesian neural network function f(x, **w**), with prior over **w**

(Neal 1996) for a broad class of prior distributions over w, the distribution of functions generated by a neural network will tend to a Gaussian process when M tends to infinity.

e.g. $k(x'-x) = f(\|x-x'\|)$ $\exp(-\frac{\theta}{2}\|x-x'\|^2)$

non-stationary $k(x, x') = x^T x'$

Covariance function k(x, x') non-stationary for neural nets with probit and Gaussian activations.

Weight prior determine the lengths scales of the neural net function.

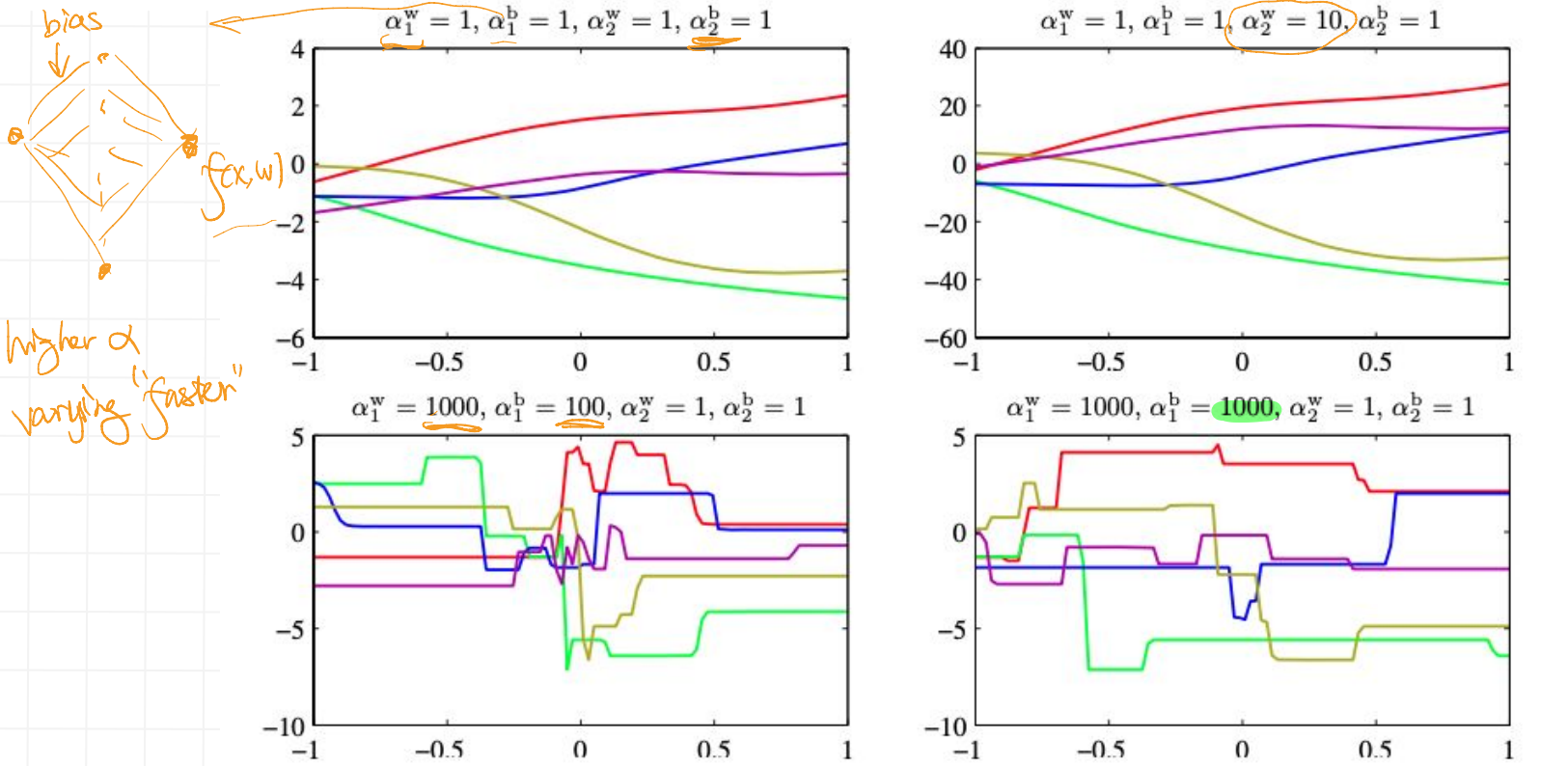


Figure 5.11 Illustration of the effect of the hyperparameters governing the prior distribution over weights and biases in a two-layer network having a single input, a single linear output, and 12 hidden units having 'tanh' activation functions. The priors are governed by four hyperparameters α_1^b, α_1^w, α_2^b, and α_2^w, which represent the precisions of the Gaussian distributions of the first-layer biases, first-layer weights, second-layer biases, and second-layer weights, respectively. We see that the parameter α_2^w governs the vertical scale of functions (note the different vertical axis ranges on the top two diagrams), α_1^w governs the horizontal scale of variations in the function values, and α_1^b governs the horizontal range over which variations occur. The parameter α_2^b, whose effect is not illustrated here, governs the range of vertical offsets of the functions.

Gaussian Processes 2

GP classification

Laplace approximation - in general

Laplace approximation - Bayesian logistic regression

Laplace approximation - GP classification