



Machine Learning Notes

All in the data.

Author: occupymars

Date: June. 23, 2025

Version: 0.1

Contents

Chapter 1	Survey Papers	1
1.1	Gaussian processes for dynamics learning in model predictive control (2025)	1
Chapter 2	Gaussian Processes for Machine Learning	8
2.1	Regression	8
Chapter 3	Other Gaussian Process Information	10
3.1	A unifying View of Sparse Approximate Gaussian Process Regression	10

Chapter 1 Survey Papers

Introduction

□ *Gaussian Process (2025)*

1.1 Gaussian processes for dynamics learning in model predictive control (2025)

1.1.1 Overview of static Gaussian process regression

GPR was introduced in the statistics community by *Curve Fitting and Optimal Design for Prediction*, and gained attention after *Bayesian Learning for Neural Networks* proved that they can be regarded as neural networks of infinite width.

Given two input data $Z = \{z_1, \dots, z_N\}$, $Z^* = \{z_1^*, \dots, z_N^*\}$, using the GP prior, we get:

$$\begin{bmatrix} g_Z \\ g_{Z^*} \end{bmatrix} \sim \mathcal{N} \left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \mathcal{K}_{Z,Z} & \mathcal{K}_{Z,Z^*} \\ \mathcal{K}_{Z^*,Z} & \mathcal{K}_{Z^*,Z^*} \end{bmatrix} \right)$$

Now given observation for Z as $Y = [y_1, \dots, y_N]$, the posterior can be written as:

$$p(g_Z, g_{Z^*} | Y) = \frac{p(Y | g_Z)p(g_Z, g_{Z^*})}{p(Y)}$$

The kernel is often taken as Gaussian one:

$$\mathcal{K}_{Z,Z^*} = \lambda \exp \left\{ -\frac{\|Z - Z^*\|^2}{2\eta} \right\}$$

If we are using the measurement model $y_i = g(z_i) + w_i$ and assume noise term is independent from prior g :

$$Y | g_Z \sim \mathcal{N}(g_Z, \sigma_w^2 I_N)$$

so the posterior is also Gaussian, we can get the posterior of interest:

$$p(g_{Z^*} | Y) = \int_Z \frac{p(Y | g_Z)p(g_Z, g_{Z^*})}{p(Y)} dg_Z$$

then compute the first and second order moments, we get $g_{Z^*} | Y \sim \mathcal{N}(\mu(Z^*), \Sigma(Z^*))$:

$$\begin{cases} \mu(Z^*) &= \mathcal{K}_{Z^*,Z}(\mathcal{K}_{Z,Z} + \sigma_w^2 I_N)^{-1} Y \\ \Sigma(Z^*) &= \mathcal{K}_{Z^*,Z^*} - \mathcal{K}_{Z^*,Z}(\mathcal{K}_{Z,Z} + \sigma_w^2 I_N)^{-1} \mathcal{K}_{Z,Z^*} \end{cases} \quad (1.1)$$

Remark 1, Alternative paradigms for uncertainty quantification, from RKHS to multi-arm bandits, frequency methods...

The hyperparameters are estimated from a subset of data (Z_h, Y_h) by optimizing the marginal likelihood:

$$\operatorname{argmax}_{\xi} p(Y_h | \xi) = \operatorname{argmax}_{\xi} \int_{\mathbb{R}^N} p(Y_h | g_{Z_h}, \xi) p(g_{Z_h} | \xi) dg_{Z_h} \quad (1.2)$$

if the measurement is i.i.d., then $Y_h | \xi \sim \mathcal{N}(0, \mathcal{K}_{Z_h, Z_h} + \sigma_w^2)$, so the above optimization problem can be written as a negative-log-likelihood minimization problem:

$$\operatorname{argmax}_{\xi} Y_h^\top (\mathcal{K}_{Z_h, Z_h} + \sigma_w^2)^{-1} Y_h + \log \det(\mathcal{K}_{Z_h, Z_h} + \sigma_w^2) \quad (1.3)$$

we can use a gradient based method to optimize this one, however this cost is not convex, so its result maybe not global minimum and thus unreliable. An alternative way is *Markov Chain Monte Carlo* approaches, perform numerical integration on 1.2.

1.1.2 Gaussian processes for dynamical systems

A first option to describe a dynamical system is the Nonlinear, Auto-Regressive with eXogenous input (*NARX*) model:

$$y_i = g_{NARX}(y_{i-1}, \dots, y_{\tau_y}, u_{i-1}, \dots, u_{\tau_u}) + w_i$$

We can write it as the state-space model:

$$\begin{cases} x_{i+1} &= f(x_i, u_i) + v_i \\ y_i &= g(x_i) + w_i \end{cases} \quad (1.4)$$

where f and g denotes transition and emission maps, typically g is known (even if it is not, we can augment it into transition maps). There are two challenges, learning two maps and state inference (from y get x), that is tackled by two different approaches in academic:

- Optimizing latent state variables: treat the state variables as optimization variables, jointly optimize it with model parameters to get maximum likelihood.
- Alternating function learning and state inference: this method try to extend Bayesian techniques such as *Extended Kalman Filter*, *Unscented Kalman Filter*, Assumed Density Filter, and Particle Filter to non-parametric models, the approximation in these studys are Taylor expansions, exact moment matching and particle representations. But when the state measurement are not available, we have to iteratively alternate between inferring the posterior and updating ξ to maximize the marginal likelihood, using algorithm like *Expectation Maximization*. The approximation to decrease the computational complexity are truncated orthogonal basis functions expansions (see [132, 133, 134]) and variational inference.

1.1.3 Problem formulation

The discrete model dynamic:

$$x_{i+1} = g_{nom}(x_i, u_i) + B_d g(x_i, u_i) + v_i \quad (1.5)$$

where $g_{nom} : \mathbb{R}^{n_x \times n_u} \rightarrow \mathbb{R}^{n_x}$, $g : \mathbb{R}^{n_x \times n_u} \rightarrow \mathbb{R}^{n_d}$, if we do not have nominal model, then $B_d = I_{n_x}$.

And we use $z_i = [x_i^\top \ u_i^\top]^\top$, we will train n_d GPs for each dimension separately.

The optimal control problem is then:

$$\text{minimize}_{\{\pi_i\}} \quad \mathbb{E} \left[\bar{\mathcal{L}}_{\bar{T}}(x_{\bar{T}}) + \sum_{i=0}^{\bar{T}-1} \bar{\mathcal{L}}(x_i, u_i) \right] \quad (1.6)$$

$$\text{subject to} \quad x_{i+1} = g_{nom}(x_i, u_i) + B_d g(x_i, u_i) + v_i \quad (1.7)$$

$$u_i = \pi_i(x_i) \quad (1.8)$$

$$\mathbb{P}(h_j(x_i, u_i) \leq 0, \forall i \geq 0) \geq p_j \quad \forall j = 1, \dots, n_h \quad (1.9)$$

$$x_0 = \bar{x}_0 \quad (1.10)$$

This problem is hard to solve, so we transform it to a MPC problem at time step k :

$$\text{minimize}_{\{\pi_i|k\}} \quad \mathbb{E} \left[\mathcal{L}_T(x_{T|k}) + \sum_{i=0}^{T-1} \mathcal{L}(x_{i|k}, u_{i|k}) \right] \quad (1.11)$$

$$\text{subject to} \quad x_{i+1|k} = g_{nom}(x_{i|k}, u_{i|k}) + B_d g(x_{i|k}, u_{i|k}) + v_{i|k} \quad (1.12)$$

$$u_{i|k} = \pi_{i|k}(x_{i|k}) \quad (1.13)$$

$$\mathbb{P}(h_j(x_{i|k}, u_{i|k}) \leq 0, \forall i \geq 0) \geq p_j \quad \forall j = 1, \dots, n_h \quad (1.14)$$

$$x_{0|k} = x_k \quad (1.15)$$

1.1.4 Scalable methods for GPR

More detailed survey please refer to *When Gaussian Process Meets Big Data: A Review of Scalable GPs*.

Table 1.1: Computational Complexity of Gaussian Process Methods.

	GP Full	Subset of Data	Expert-based	FTC	SSGP	SKI	SVGP
Training	$\mathcal{O}(N^3)$	$\mathcal{O}(M^3)$	$\mathcal{O}(NM_e^2)$	$\mathcal{O}(NM^2)$	$\mathcal{O}(Np^2)$	$\mathcal{O}(N + M \log M)$	$\mathcal{O}(M^3)$
Inference	$\mathcal{O}(N^2)$	$\mathcal{O}(M^2)$	$\mathcal{O}(M_e^2)$	$\mathcal{O}(M^2)$	$\mathcal{O}(p^2)$	$\mathcal{O}(M \log M)$	$\mathcal{O}(M^2)$

1. Subset of Data, sample data using some criterion (refer to *Gaussian Process Models: PAC-Bayesian Generalisation Error Bounds and Sparse Approximations*, chapter 4) and clustering. This will overestimate uncertainty, but new study leveraging graphons complementss rigorous bounds for it.

We can also use multiple models for different regions for non-Stationarity or scalability. One of them is called **Mixture-of-Experts** (MoE), given N_{exp} GPs, denoting with $\{s_k(\cdot)\}_{k=1}^{N_{exp}}$ a set of gating functions, the overall likelihood is

$$p_{MoE}(y | g_z^1, \dots, g_z^{N_{exp}}) = \sum_{k=1}^{N_{exp}} s_k(g_z^k) p_k(y | g_z^k)$$

to scale well, we need to use infinite MoE or one of the approximation methods. We can also pre-allocate experts but this will lose connection between experts. See [166], [167] for online updates. And there is another method called "bagging".

Instead of resorting to a linear combination of GPs, we can use **Product-of-Experts** (PoE), where

$$p_{PoE}(y | g_z^1, \dots, g_z^{N_{exp}}) \propto \prod_{k=1}^{N_{exp}} p_k(y | g_z^k)$$

this will make weak expert plays which is not good, so we can use weighted product and Bayesian Committee Machine, combined we have [174]. MoE and PoE combine in *Deep Structured Mixtures of Gaussian Processes*. Analysis of theory in *An asymptotic analysis of distributed nonparametric methods*.

2. Inducing Variables, given inducing points (pseudo-inputs) \bar{Z} and $g_{\bar{Z}} \sim \mathcal{N}(0, \mathcal{K}_{\bar{Z}, \bar{Z}})$, g_Z and g_{Z^*} are conditionally independent, they can only communicate through $g_{\bar{Z}}$. There are two main groups, one is approximate prior $p(g_Z, g_{Z^*})$ and do exact inference (reviewed in *A Unifying View of Sparse Approximate Gaussian Process Regression*), one is from original prior and approximate $p(g_{Z^*} | Y)$ (reviewed in *A unifying framework for Gaussian process pseudo-point approximations using power expectation propagation*), they are compared in *Understanding Probabilistic Sparse Gaussian Process Approximations*.

First we talk about method approximating prior, with:

$$\begin{aligned} p(g_Z, g_{Z^*}) &= \int p(g_Z, g_{Z^*} | g_{\bar{Z}}) p(g_{\bar{Z}}) dg_{\bar{Z}} \\ &\approx \int q(g_Z | g_{\bar{Z}}) q(g_{Z^*} | g_{\bar{Z}}) p(g_{\bar{Z}}) dg_{\bar{Z}} \\ &= q(g_Z, g_{Z^*}) \end{aligned} \quad (1.16)$$

the choice of $q(g_Z | g_{\bar{Z}}) = \mathcal{N}(\mathcal{K}_{Z, \bar{Z}} \mathcal{K}_{\bar{Z}, \bar{Z}}^{-1} g_{\bar{Z}}, \tilde{Q}_{Z, Z})$ and $q(g_{Z^*} | g_{\bar{Z}}) = \mathcal{N}(\mathcal{K}_{Z^*, \bar{Z}} \mathcal{K}_{\bar{Z}, \bar{Z}}^{-1} g_{\bar{Z}}, \tilde{Q}_{Z^*, Z^*})$ will differ between different methods below. The conditional distribution is actually $q(g_Z | g_{\bar{Z}}) = \mathcal{N}(\mathcal{K}_{Z, \bar{Z}} \mathcal{K}_{\bar{Z}, \bar{Z}}^{-1} g_{\bar{Z}}, \mathcal{K}_{Z, Z} - \mathcal{K}_{Z, \bar{Z}} \mathcal{K}_{\bar{Z}, \bar{Z}}^{-1} \mathcal{K}_{\bar{Z}, Z})$, refer to **proof**, \tilde{Q} is a low rank matrix.

- **Subset of Regressors** (SoR), $\mathcal{K}_{Z, Z} \approx \mathcal{K}_{Z, \bar{Z}} \mathcal{K}_{\bar{Z}, \bar{Z}}^{-1} \mathcal{K}_{\bar{Z}, Z} = Q_{Z, Z}$, so covariance of $q(g_Z | g_{\bar{Z}})$ and $q(g_{Z^*} | g_{\bar{Z}})$ is $\tilde{Q}_{Z, Z} = Q_{Z, Z} - \mathcal{K}_{Z, \bar{Z}} \mathcal{K}_{\bar{Z}, \bar{Z}}^{-1} \mathcal{K}_{\bar{Z}, Z} = 0$, possibly leading to overconfident predictions. $g_{Z^*} = \mathcal{K}_{Z^*, \bar{Z}} W_{\bar{Z}}$, $W_{\bar{Z}} \sim \mathcal{N}(0, \mathcal{K}_{\bar{Z}, \bar{Z}}^{-1})$, $W_{\bar{Z}}$ can also be written as $\mathcal{K}_{\bar{Z}, \bar{Z}}^{-1} \bar{Z}$.
- **Deterministic Training Conditional** (DTC), same mean of SoR, covariance is more sensible but the result is an inconsistent GP (train with low-rank approximation, test with full variance).
- **Fully Independent Conditional** (FIC), assume g_Z and g_{Z^*} are independent of $g_{\bar{Z}}$, and **Fully Independent Training Conditional** (FITC) admits the factorization on the training conditional only, at the price of having again an inconsistent GP. **If the prediction is to be performed on a single point, this two method coincide.**

$$\begin{aligned} \mu^{FITC}(Z^*) &= \mathcal{K}_{Z^*, \bar{Z}}, Q^{-1} \mathcal{K}_{\bar{Z}, Z} (\Lambda + \sigma_w^2 I_N)^{-1} Y \\ \Sigma^{FITC}(Z^*) &= \mathcal{K}_{Z^*, Z^*} - \mathcal{K}_{Z^*, \bar{Z}} (\mathcal{K}_{\bar{Z}, \bar{Z}}^{-1} - Q^{-1}) \mathcal{K}_{\bar{Z}, Z^*} + \sigma_w^2 \end{aligned} \quad (1.17)$$

where $Q = \mathcal{K}_{\bar{Z}, \bar{Z}} + \mathcal{K}_{\bar{Z}, Z} (\Lambda + \sigma_w^2 I_N)^{-1} \mathcal{K}_{Z, \bar{Z}}$ and $\Lambda_{a, a} = \mathcal{K}_{Z_a, Z_a} - \mathcal{K}_{Z_a, \bar{Z}} \mathcal{K}_{\bar{Z}, \bar{Z}}^{-1} \mathcal{K}_{\bar{Z}, Z_a}$, for $a = 1, \dots, N$.

- **Partially Independent (Training) Conditional** (PI(T)C) generalizes FI(T)C by introducing a block structure in the covariance. There maybe no significant improve with respect to FI(T)C. $\mathcal{K}_{PI(T)C} = Q_{Z, Z} - \text{block.diag}(\mathcal{K}_{Z, Z} - Q_{Z, Z})$.

These methods lead to an approximation marginal likelihood:

$$q(Y) = \mathcal{N}(0, \tilde{Q}_{Z, Z} + \mathcal{K}_{Z, \bar{Z}} \mathcal{K}_{\bar{Z}, \bar{Z}}^{-1} \mathcal{K}_{\bar{Z}, Z} + \sigma_w^2 I_N) \quad (1.18)$$

the choice of inducing points can be same as subset of data method, using information gain, online learning and greedy posterior maximization. Or we can treat them as hyperparameters, and maximized by 1.18, which is complicated and may

lead to local optimal and over-fitting. Other methods can be seen in MCMC schemes, which will take longer training times.

Second we talk about approximate the posterior. We can use so-called *Variational Free Energy* (VFE) to get:

$$\begin{aligned}
 p(g_{Z^*}|Y) &= \int \int p(g_{Z^*}|g_Z, g_{\bar{Z}})p(g_Z|g_{\bar{Z}}, Y)p(g_{\bar{Z}}|Y)dg_Z dg_{\bar{Z}} \\
 &\approx q(g_{Z^*}) \\
 &= \int \int p(g_{Z^*}|g_{\bar{Z}})p(g_Z|g_{\bar{Z}})p(g_{\bar{Z}}|Y)dg_Z dg_{\bar{Z}} \\
 &= \int p(g_{Z^*}|g_{\bar{Z}})p(g_{\bar{Z}}|Y)dg_{\bar{Z}} \\
 &\approx \int p(g_{Z^*}|g_Z)q(g_Z)dg_Z
 \end{aligned} \tag{1.19}$$

where $p(g_{Z^*}|g_Z, g_{\bar{Z}}, Y) = p(g_{Z^*}|g_Z, g_{\bar{Z}})$ because Y is just a noisy version of g_Z and $g_{\bar{Z}}$ is sufficient.

We then use variational inference to choose g_Z and \bar{Z} , by minimizing Kullback-Leibler (KL) divergence:

$$\mathcal{KL}(q(g_{Z^*}, g_Z) \| p(g_{Z^*}, g_Z|Y)) = \log p(Y) - \mathbb{E}_{q(g_Z, g_Z)} \left[\frac{p(Y, g_Z, g_Z)}{q(g_Z, g_Z)} \right]$$

I think here log is for both term, and I think left side Z^* should be \bar{Z} , or the $=$ should be \approx .

In [204] we get $g(g_{\bar{Z}}) = \mathcal{N}(\mu_q, \Sigma_q)$, where:

$$\begin{aligned}
 \mu_q &= \sigma_w^{-2} \mathcal{K}_{\bar{Z}, \bar{Z}} (\mathcal{K}_{\bar{Z}, \bar{Z}} + \sigma_w^{-2} \mathcal{K}_{\bar{Z}, Z} \mathcal{K}_{Z, \bar{Z}})^{-1} \mathcal{K}_{\bar{Z}, Z} Y \\
 \Sigma_q &= \mathcal{K}_{\bar{Z}, \bar{Z}} (\mathcal{K}_{\bar{Z}, \bar{Z}} + \sigma_w^{-2} \mathcal{K}_{\bar{Z}, Z} \mathcal{K}_{Z, \bar{Z}})^{-1} \mathcal{K}_{\bar{Z}, \bar{Z}}
 \end{aligned}$$

and the hyperparameters can be found by optimizing:

$$\log p(Y) - \log[\mathcal{N}(0, \sigma_w^2 + \mathcal{K}_{Z, \bar{Z}} \mathcal{K}_{\bar{Z}, \bar{Z}}^{-1} \mathcal{K}_{\bar{Z}, Z})] + \frac{1}{\sigma_w^2} \text{Tr}(\mathcal{K}_{Z, Z} - \mathcal{K}_{Z, \bar{Z}} \mathcal{K}_{\bar{Z}, \bar{Z}}^{-1} \mathcal{K}_{\bar{Z}, Z})$$

the predictive distribution is the same as the one obtained in the DTC approach, but the optimization problem yielding hyperparameters and pseudo-inputs differs by the last addendum, which plays the role of a regularizer and acts against over-fitting. And the relation with FITC in [220], [221] for mini-batch training. Pseudo-inputs can be set arbitrarily, or use *SKI* (structured kernel interpolation, kernel with structure good for efficient calculation), or from training inputs.

Sparse GP error estimation in [230], [231].

3. Finite-dimensional representations of the kernel operator is next method, as above sparse GP revolve around the concept of eigen-decomposition of Gram matrices $\mathcal{K}_{Z, Z}$, this one consider eigen-decomposition of the kernel operator $\mathcal{K} : Z \times Z \rightarrow \mathbb{R}$.

The first one is *Sparse Spectrum Gaussian processes* (SSGP), introduced in [234], using sum of Fourier features. It is suitable for stationary kernels:

$$\mathcal{K}(z_a, z_b) = \int \exp\{2\pi i s^\top (z_a - z_b)\} S(s) ds$$

where $S(s)$ is proportional to $p_S(s)$, for Gaussian kernel:

$$p_S(s) = \frac{1}{\lambda} \int \exp\{-2\pi i s^\top (z_a - z_b)\} \mathcal{K}(z_a, z_b) dz = \sqrt{2\pi\eta^{n_z}} \exp\{-2\pi^2 \eta \|s\|^2\}$$

which is a multivariate Gaussian. Using MC method, sample $\{s_r, -s_r\}_{r=1}^{M/2}$ to respect symmetry and exploiting trigonometric identities:

$$\mathcal{K}(z_a, z_b) \approx \frac{\lambda}{M} \sum_{r=1}^{M/2} (\cos(2\pi s_r^\top z_a) \cos(2\pi s_r^\top z_b) + \sin(2\pi s_r^\top z_a) \sin(2\pi s_r^\top z_b))$$

Then the kernel is approximate to $\mathcal{K}(z_a, z_b) = \langle \phi(z_a), \phi(z_b) \rangle$, where

$$\phi(z) = [\cos(2\pi s_1^\top z), \dots, \cos(2\pi s_{M/2}^\top z), \dots, \sin(2\pi s_{M/2}^\top z)]^\top$$

then $g(z) \approx \phi(z)^\top \alpha$.

Another one aims for a series expansion of the type (see Mercer's Theorem):

$$\mathcal{K}(z_a, z_b) = \sum_{k=1}^{+\infty} \gamma_k \varphi_k(z_a) \varphi_k(z_b)$$

such that $\int \mathcal{K}(z_a, z_b) \varphi_k(z_a) p_z(z_a) dz_a = \gamma_k \varphi_k(z_b)$, where $\{\varphi_k\}_{k=1}^{+\infty}$ is a family of orthonormal functions with respect to the measure induced by $p_z(\cdot)$, and $\{\gamma_k\}_{k=1}^{+\infty}$ is a set of decreasing, non-negative values.

4. Computational techniques is the last topic in this section.

First we talk about *Nystrom approximation*, $\mathcal{K}_{Z,Z} \approx \mathcal{K}_{Z,\bar{Z}} \mathcal{K}_{\bar{Z},\bar{Z}}^{-1} \mathcal{K}_{\bar{Z},Z}$, can be efficiently obtained from an incomplete (block) column-based Cholesky decomposition when the same (block) columns are selected.

Secondly, *Conjugate gradient* (CG) method is for optimization problem $\min_x \frac{1}{2} x^\top A x - x^\top b$, iteratively refines its solution estimate by performing the minimization successively in a growing subspace of orthogonal search directions. Computational complexity is $\mathcal{O}(PN^2)$, where P is the iteration, and CG can guarantee to when $P = N$, it will recover the exact solution. But we will make $P \ll N$ to get efficiency while get a good approximation. With efficient MVMs (matrix-vector multiplications) of SKI, CG can speed up sparse GP.

Thirdly, the *parallelization of computations*, for MoE, inducing point method (PITC has the block structure, using low-rank-cum-Markov approximations), Nystrom approximation (row-based Cholesky decomposition), CG (batch version).

5. online learning:

- *Active-data selection*, using a variety of methods to add or delete data points in dataset.
- Expert-based methods, compute the distance of new data and each local model, and update corresponding local model.
- Including point methods, [358] for FITC, [366] for VFE, a lot of papers mentioned.
- Finite-dimensional approximations of kernel operator, [245] for SSGP and other.
- Computational methods, leverage SKI [382]

1.1.5 Uncertainty Propagation

Directly predict n-step, [387-389]. Indirect method, iteratively do one-step prediction, neglect non-consecutive states relation, the uncertainty prediction can be deteriorate [391].

1. Independent one-step-ahead predictions, with random Z^* :

$$p(g_{Z^*} | Y) = \int p(g_{Z^*} | Z^*, Y) p(Z^*) dZ^* \quad (1.20)$$

the integral is the product of two Gaussian (if we assume $p(Z^*) = \mathcal{N}(\mu^*, \Sigma^*)$), it is not analytic, note that $p(g_{Z^*} | Z^*, Y) = \mathcal{N}(\mu(Z^*), \Sigma(Z^*))$. We actually only need first and second moments, which are given as:

$$\begin{aligned} \mathbb{E}_{g_{Z^*}}[g_{Z^*} | Y] &= \mathbb{E}_{Z^*}[\mathbb{E}_{g_{Z^*}}[g_{Z^*} | Z^*, Y]] \\ &\stackrel{1.1}{=} \mathbb{E}_{Z^*}[\mu(Z^*)], \\ \text{Var}_{g_{Z^*}}[g_{Z^*} | Y] &= \mathbb{E}_{Z^*}[\text{Var}_{g_{Z^*}}[g_{Z^*} | Z^*, Y]] + \text{Var}_{Z^*}[\mathbb{E}_{g_{Z^*}}[g_{Z^*} | Z^*, Y]] \\ &\stackrel{1.1}{=} \mathbb{E}_{Z^*}[\Sigma(Z^*)] + \text{Var}_{Z^*}[\mu(Z^*)]. \end{aligned} \quad (1.21)$$

We have three ways to approximate this. First is *Linearlization*, using Taylor expansion:

$$\begin{aligned} \mu(z^*) &\approx \mu(\mu^*) + \nabla_z \mu(z)|_{z=\mu^*}^\top (z^* - \mu^*) \\ \Sigma(z^*) &\approx \Sigma(\mu^*) + \nabla_z \Sigma(z)|_{z=\mu^*}^\top (z^* - \mu^*) + \frac{1}{2} (z^* - \mu^*)^\top \nabla_z^2 \Sigma(z)|_{z=\mu^*}^* (z^* - \mu^*) \end{aligned} \quad (1.22)$$

which then:

$$\begin{aligned} \mathbb{E}_{Z^*}[\mu(Z^*)] &\approx \mathbb{E}_{Z^*}[\mu(\mu^*) + \nabla_z \mu(z)|_{z=\mu^*}^\top (Z^* - \mu^*)] = \mu(\mu^*), \\ \mathbb{E}_{Z^*}[\Sigma(Z^*)] &\approx \mathbb{E}_{Z^*}[\Sigma(\mu^*) + \nabla_z \Sigma(z)|_{z=\mu^*}^\top (Z^* - \mu^*) + \frac{1}{2} (Z^* - \mu^*)^\top \nabla_z^2 \Sigma(z)|_{z=\mu^*}^* (Z^* - \mu^*)] \\ &= \Sigma(\mu^*) + \frac{1}{2} \text{Tr} \left\{ \nabla_z^2 \Sigma(z)|_{z=\mu^*}^* \Sigma^* \right\}, \\ \text{Var}_{Z^*}[\mu(Z^*)] &\approx \text{Var}_{Z^*}[\mu(\mu^*) + \nabla_z \mu(z)|_{z=\mu^*}^\top (Z^* - \mu^*)] = \nabla_z \mu(z)|_{z=\mu^*}^\top \Sigma^* \nabla_z \mu(z)|_{z=\mu^*}. \end{aligned} \quad (1.23)$$

Secondly, we have *Exact Moment Matching*, if we using Gaussian kernel and assume Gaussian input, then the integral of 1.21 can be derived:

$$\mathbb{E}_{z^*}[\mu(z^*)] = \int \mu(z^*) p(z^*) dz^* = \beta^\top \mathbf{1}$$

where $\beta = (\mathcal{K}_{Z,Z} + \sigma_w^2 I_N)^{-1} Y$ and $\mathbf{1} = [l_1, \dots, l_N]$ is function of z_i, μ^*, Σ^* and hyperparameters.

$$\begin{aligned} \text{Var}_{g_{Z^*}}[g_{Z^*} | Y] &= \mathbb{E}_{Z^*}[\Sigma(Z^*)] + \mathbb{E}_{Z^*}[\mu^2(Z^*)] - (\mathbb{E}_{Z^*}[\mu(Z^*)])^2 \\ &= \beta^\top L \beta - \text{Tr} \left\{ (\mathcal{K}_{Z,Z} + \sigma_w^2 I_N)^{-1} L \right\} - (\beta^\top \mathbf{1})^2, \end{aligned}$$

Thirdly, we have *Sigma-point propagation*, we compute sigma-point for $z^* \sim (\mu^*, \Sigma^*)$, we denote by $\{\bar{z}_j^*\}_{j=0}^{2n_z}$,

where n_z is the dimensionality of z .

$$\begin{aligned}\bar{z}_0^* &= \mu^* \\ \bar{z}_j^* &= \mu^* + \sqrt{n_z + \lambda_{mm}} [\text{chol}(\Sigma^*)]_j, \quad \text{for } j = 1, \dots, n_z \\ \bar{z}_j^* &= \mu^* - \sqrt{n_z + \lambda_{mm}} [\text{chol}(\Sigma^*)]_j, \quad \text{for } j = n_z + 1, \dots, 2n_z\end{aligned}\tag{1.24}$$

where $[\text{chol}(\Sigma^*)]_j$ is the j -th column of the Cholesky factorization of matrix Σ^* , and λ_{mm} is a user-defined parameter representing how far the sigmapoints are spread from the mean. By viewing the g as $g = \mu + \tilde{w}$, where $\tilde{w} \sim \mathcal{N}(0, \Sigma)$ is treated as "process noise", we can approximate $\mathbb{E}_{g_{z^*}}[g_{z^*} | Y]$ by evaluating μ on the sigma-points:

$$\mathbb{E}_{g_{z^*}}[g_{z^*} | Y] \approx \mu_{\text{sp}} = \sum_{j=0}^{2n_z} W_j^m \mu(\bar{z}_j^*).$$

and the variance is:

$$\text{Var}_{g_{z^*}}[g_{z^*} | Y] \approx \sum_{j=0}^{2n_z} W_j^v (\mu(\bar{z}_j^*) - \mu_{\text{sp}})(\mu(\bar{z}_j^*) - \mu_{\text{sp}})^T + \Sigma(\mu^*).$$

the weights are choose to sum to 1, how to tune them can be found in [117]. This method can be compared to MC method, first one choose the points deterministically, second one randomly (but it does not need the assumption of GP prior).

Fourth is the numerical approximation, 1.20 can be computed using MC. [403] tackles the computational issue, where the sampling process is guided by a measure of correlation to the previous evaluations.

The above methods handle uncertainty in inference, does not exploit them in training. So here we talk about them, and they are connected to treat latent variables as optimization variables in the second section [108]. Or we can use Gaussian mixture models [397].

2. Robust uncertainty propagation, try to give a deterministic bound rather than stochastic one for the propagation. Model the $z \in \mathcal{E}(z_p, Q_p)$ which is an ellipsoidal confidence region, and it can maintain its shape through the $\tilde{g} = g_{\text{nom}}(\bar{z}) + \nabla_z g_{\text{nom}}(z)|_{z=\bar{z}}^T (z - \bar{z}) + \mu(\bar{z})$. This method maybe too conservative. More detail please refer to [407].

3. Overcoming the independence assumption, iteratively recondition the GP model, enabling a scenario-based [411] control design. while this may face the curse of dimensionality, other method try to approximate it with finite horizon [415], [391].

1.1.6 Closed-loop Safty Guarantees

This section we discuss how to make sure $\mathbb{P}(h_j(x_i, u_i) \leq 0, \forall i \geq 0, \forall \{j\}_{j=1}^{n_h}) \geq p$, which is a joint-in-time chance constraints, every time step must satisfy the constraints.

Table 1.2: Comparison of Closed-Loop Safety Guarantee Methods

Method	Joint-in-Time	Conservativeness	Complexity
6.1 Bounded-Support	X	Low–Medium	Low
6.2 Robust-in-Probability	Y	Medium–High	Medium–High
6.3 Sampling-Based	Y	Low	High
6.4 Other Methods	X (typically)	Varies	Low–Medium

- Bounded support assumption: **compact set** is bounded and the bound is inside it (for Euler space, other general case need to use Open coverage to define). In this assumption, the error is assume to be bounded with a compact set $C_\delta = \{\varepsilon : \mathbb{P}(\varepsilon \in C_\delta) \geq 1 - \delta\}$, where δ is a small probability, we can use χ^2 to get the compact set. The error can (is) be $|g - \hat{g}| = |g - \mu|$, where \hat{g} gets from the data, the model we trained. This method will make the joint-in-time constraints becomes point-wise-in-time chance constraints so it is not suitable for multi-step prediction.
- Robust-in-probability, construct a high confidence interval with a probability of p such that the probability of the model error falling within this set $\mathcal{G}(x, u)$ is p , and then perform robust analysis on all possible errors in $\mathcal{G}(x, u)$. Define $\Delta = \{g(x, u) \in \mathcal{G}(x, u) \text{ for all } (x, u) \in \mathcal{Z}\}$, this method guarantee that $\mathbb{P}(\Delta) \geq p$. Ensuring that

$$\mathbb{P}(h_j(x_i, u_i) \leq 0, \forall i \geq 0, \forall \{j\}_{j=1}^{n_h} | \Delta) = 1$$

we can then make sure:

$$\begin{aligned} & \mathbb{P}(h_j(x_i, u_i) \leq 0, \forall i \geq 0, \forall \{j\}_{j=1}^{n_h}) \\ & \geq \mathbb{P}(h_j(x_i, u_i) \leq 0, \forall i \geq 0, \forall \{j\}_{j=1}^{n_h} \mid \Delta) P(\Delta) \\ & \geq p \end{aligned}$$

a complete survey can be found in [64]. **It tries to get the error bound** $|g - \hat{g}|$, in order to guarantee safety, we have to make sure such a bound exist (and maybe small enough). One of the possible construction can be provided by RKHS method, $|g - \hat{g}| \leq \beta_n \sigma_n(x)$, where β_n is RKHS norm defined by $|g|_{\mathcal{H}_k}$, $\sigma_n(x)$ is the variance.

- Sampling-based approach, [413], [414], [427] samples GP to get tightened constraints. This method is not conservative, but it is computational heavy.
- other results, generally consider point-wise-in-time chance constraints, can be seen in [432]...

1.1.7 Discussion

1. MPC with Scalable GP, compare of FITC and VFE can be seen in *Understanding Probabilistic Sparse Gaussian Process Approximations, A Framework for Evaluating Approximation Methods for Gaussian Process Regression* compare FITC with SoD, MoE and CG.

And for online learning, [407] based on RKHSs type bound, feedback linearization in [437], [438]. *Online learning-based Model Predictive Control with Gaussian Process Models and Stability Guarantees* recursively update Cholesky factor and give a stability prove.

2. Real-time GPMPC, the formulation is:

$$\begin{aligned} \underset{\{u_{i|k}, \mu_{i|k}^x, \Sigma_{i|k}^x\}}{\text{minimize}} \quad & \ell_T(\mu_{T|k}^x, \Sigma_{T|k}^x) + \sum_{i=0}^{T-1} \ell_i(\mu_{i|k}^x, u_{i|k}, \Sigma_{i|k}^x) \\ \text{subject to} \quad & \mu_{i+1|k}^x = \rho(u_{i|k}, \mu_{i|k}^x, \Sigma_{i|k}^x), \quad i = 0, \dots, T-1, \\ & \Sigma_{i+1|k}^x = \Phi(u_{i|k}, \mu_{i|k}^x, \Sigma_{i|k}^x), \quad i = 0, \dots, T-1, \\ & h_j(\mu_{i|k}^x, u_{i|k}) + \nu_j(u_{i|k}, \mu_{i|k}^x, \Sigma_{i|k}^x) \leq 0, \quad i = 0, \dots, T-1, \quad j = 1, \dots, n_h, \\ & \mu_{0|k}^x = x_k, \\ & \Sigma_{0|k}^x = 0_{n_x \times n_x}. \end{aligned} \tag{1.25}$$

the problem is the optimization variables are too many, the solution can be condense solution in [444] (Unmanned Quadrotor), non-condense one in [445, 446]. But still the computations are very heavy.

Another problem is the gradient of the mean and variance, one solution is to neglect the variance, compute it solely on the mean dynamic [448, 442, 449, 450]. Or we can use the last time step solution for the variance, or the last solver iteration for the quick update.

3. Alternative models for dynamic, Nonlinear Output Error (NOE) structures, consider the simulation error rather than one step ahead prediction error in NARX, so it is a direct method that predict multiple steps. Or use latent variables optimization, leverage the approximate training and variational inference to get variance into training stage.

We also have so-called **Sapio Temporal GP**, consider $g(z_i; t_k)$, first attempts mainly in geostatistics community. Another way is reformulating the STGP as Kalman filtering problem, which is reviewed in *The SPDE approach for Gaussian and non-Gaussian fields: 10 years and still running*.

Then we can also consider multi-output GP in geostatistics and multi-task learning, by linear combination of latent functions, or by vector-valued GP and multi-output RKHS.

4. Alternative uncertainty quantification, try heuristics β in 1.1.6 second category but no theoretical guarantees. Or directly use the GP prior [505], use student-t, kernel embedding. In noise free setting we can use distributional kernel embeddings and sample method, for bounded noise we can use multi-step predictors, for additive unbounded noise in linear system (tightening) and nonlinear (conformal prediction).

Chapter 2 Gaussian Processes for Machine Learning

2.1 Regression

2.1.1 Weight-space View

Given a training dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i) \mid i = 1, \dots, n\} = (X, \mathbf{y}) \in \mathbb{R}^{D \times n}$, we will review the Bayesian analysis of the standard linear regression model with Gaussian noise.

$$f(\mathbf{x}) = \mathbf{x}^\top \mathbf{w}, \quad y = f(\mathbf{x}) + \varepsilon, \quad (2.1)$$

where $\varepsilon \sim \mathcal{N}(0, \sigma_n^2)$ is the noise.

Definition 2.1 (Likelihood)

Likelihood is the probability density of the observations given the parameters

$$\begin{aligned} p(\mathbf{y}|X, \mathbf{w}) &= \prod_{i=1}^n p(y_i|\mathbf{x}_i, \mathbf{w}) \\ &= \prod_{i=1}^n \frac{1}{\sqrt{2\pi}\sigma_n} \exp\left(-\frac{(y_i - \mathbf{x}_i^\top \mathbf{w})^2}{2\sigma_n^2}\right) \\ &= \frac{1}{(2\pi\sigma_n^2)^{n/2}} \exp\left(-\frac{1}{2\sigma_n^2}|\mathbf{y} - X^\top \mathbf{w}|^2\right) \\ &= \mathcal{N}(X^\top \mathbf{w}, \sigma_n^2 \mathcal{I}) \end{aligned} \quad (2.2)$$

where $|z|$ denotes the Euclidean length of vector z .

In the Bayesian formalism we need to specify a prior over the parameters, expressing our beliefs about the prior parameters before we look at the observations. We put a zero mean Gaussian prior with covariance matrix Σ_p on the weights

$$\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \Sigma_p) \quad (2.3)$$

Definition 2.2 (Posterior)

Based on Bayes' rule, we have

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{marginal likelihood}}, \quad p(\mathbf{w}|\mathbf{y}, X) = \frac{p(\mathbf{y}|X, \mathbf{w})p(\mathbf{w})}{p(\mathbf{y}|X)} \quad (2.4)$$

where the normalizing constant, also known as the **marginal likelihood**, is independent of the weights and given by

$$p(\mathbf{y}|X) = \int p(\mathbf{y}|X, \mathbf{w})p(\mathbf{w})d\mathbf{w} \quad (2.5)$$

Then we can obtain

$$\begin{aligned} p(\mathbf{w}|X, \mathbf{y}) &\propto \exp\left(-\frac{1}{2\sigma_n^2}(\mathbf{y} - X^\top \mathbf{w})^\top (\mathbf{y} - X^\top \mathbf{w})\right) \exp\left(-\frac{1}{2}\mathbf{w}^\top \Sigma_p^{-1} \mathbf{w}\right) \\ &\propto \exp\left(-\frac{1}{2}(\mathbf{w} - \bar{\mathbf{w}})^\top \left(\frac{1}{\sigma_n^2} X X^\top + \Sigma_p^{-1}\right) (\mathbf{w} - \bar{\mathbf{w}})\right) \end{aligned} \quad (2.6)$$

where $\bar{\mathbf{w}} = \sigma_n^{-2}(\sigma_n^{-2} X X^\top + \Sigma_p^{-1})^{-1} X \mathbf{y}$, we can write it concisely as

$$p(\mathbf{w}|X, \mathbf{y}) \sim \mathcal{N}(\bar{\mathbf{w}} = \frac{1}{\sigma_n^2} A^{-1} X \mathbf{y}, A^{-1}) \quad (2.7)$$

Notice that for this model (and indeed for any Gaussian posterior) the mean of the posterior distribution $p(\mathbf{w}|X, \mathbf{y})$ is also its mode, which is also called the **maximum a posteriori** (MAP) estimate of \mathbf{w} . In a non-Bayesian setting the negative log prior is sometimes thought of as a penalty term, and the MAP point is known as the penalized maximum likelihood estimate of the weights. Note, however, that in the Bayesian setting the MAP estimate plays no special role. In

this case, due to symmetries in the model and posterior, it happens that the mean of the predictive distribution is the same as the prediction at the mean of the posterior. However, this is not the case in general. The penalized maximum likelihood procedure is known in this case as ridge regression because of the effect of the quadratic penalty term $\frac{1}{2}\mathbf{w}^T \Sigma_p^{-1} \mathbf{w}$ from the log prior.

To make predictions for a test case we average over all possible parameter predictive distribution values, weighted by their posterior probability. This is in contrast to non-Bayesian schemes, where a single parameter is typically chosen by some criterion.

$$\begin{aligned} p(f_* | \mathbf{x}_*, X, \mathbf{y}) &= \int p(f_* | \mathbf{x}_*, \mathbf{w}) p(\mathbf{w} | X, \mathbf{y}) d\mathbf{w} \\ &= \mathcal{N}\left(\frac{1}{\sigma_n^2} \mathbf{x}_*^T A^{-1} X \mathbf{y}, \mathbf{x}_*^T A^{-1} \mathbf{x}_*\right) \end{aligned} \quad (2.8)$$

we can proof it with $\mathbb{E}(f_*) = \mathbb{E}(\mathbf{x}_*^T w) = \mathbf{x}_*^T \mu_w$, $\text{Var}(f_*) = \mathbf{x}_*^T \Sigma_w \mathbf{x}_*$. Or in a complicate way (my proof)

$$\begin{aligned} p(f_* | \mathbf{x}_*, X, \mathbf{y}) &= \int \delta(f_* - \mathbf{x}_*^T w) \cdot \frac{1}{\sqrt{2\pi \Sigma_w}} \exp\left[-\frac{(w - \mu_w)^T (w - \mu_w)}{2 \Sigma_w}\right] dw \\ &= \frac{1}{\sqrt{2\pi \Sigma_w \mathbf{x}_*^T}} \exp\left[-\frac{(w - \mu_w)^T \mathbf{x}_* \mathbf{x}_*^T (w - \mu_w)}{2 \mathbf{x}_*^T \Sigma_w \mathbf{x}_*}\right] |_{f_* = \mathbf{x}_*^T w} \\ &= \frac{1}{\sqrt{2\pi \Sigma_w \mathbf{x}_*^T}} \exp\left[-\frac{(f_* - \mathbf{x}_*^T \mu_w)^T (f_* - \mathbf{x}_*^T \mu_w)}{2 \mathbf{x}_*^T \Sigma_w \mathbf{x}_*}\right] \end{aligned}$$

Chapter 3 Other Gaussian Process Information

3.1 A unifying View of Sparse Approximate Gaussian Process Regression

3.1.1 Gaussian Processes for Regression

Probabilistic regression is usually formulated as follows, given a training dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i) \mid i = 1, \dots, n\} = (X, \mathbf{y}) \in \mathbb{R}^{D \times n}$


$$y_i = f(\mathbf{x}_i) + \varepsilon_i, \quad \text{where } \varepsilon_i \sim \mathcal{N}(0, \sigma_{\text{noise}}^2) \quad (3.1)$$

Gaussian process (GP) regression is a Bayesian approach which assumes a GP prior over functions, assumes a priori that function values behave according to

$$p(\mathbf{f} | \mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n) = \mathcal{N}(0, K) \quad (3.2)$$

where $\mathbf{f} = [\mathbf{f}_1, \mathbf{f}_2, \dots, \mathbf{f}_n]^\top$ is a vector of latent function values $f_i = f(\mathbf{x}_i)$ and K is a covariance matrix, whose entries are given by the covariance function $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. Note that the GP treats the latent function values f_i as **random variables**, indexed by the corresponding input. The GP model is concerned only with the conditional of the outputs given the inputs; we do not model anything about the inputs themselves.

Definition 3.1 (Degenerate GP)

A Gaussian process is called degenerate if the covariance function has a finite number of non-zero eigenvalues. Degenerate GPs (such as e.g. with polynomial covariance function) correspond to finite linear (-in-the-parameters) models, whereas non-degenerate GPs (such as e.g. with squared exponential or RBF covariance function) do not. 

Using joint GP prior $p(\mathbf{f}, \mathbf{f}_*)$ and the likelihood $p(\mathbf{y} | \mathbf{f})$ we get the joint posterior

$$p(\mathbf{f}, \mathbf{f}_* | \mathbf{y}) = \frac{p(\mathbf{f}, \mathbf{f}_*)p(\mathbf{y} | \mathbf{f})}{p(\mathbf{y})} \quad (3.3)$$

where $p(\mathbf{y} | \mathbf{f}) = p(\mathbf{y} | \mathbf{f}, \mathbf{f}_*)$ since the likelihood is conditionally independent of everything else given \mathbf{f} .

The final step needed to produce the desired posterior predictive distribution is to marginalize out the unwanted training set latent variables

$$p(\mathbf{f}_* | \mathbf{y}) = \int p(\mathbf{f}, \mathbf{f}_* | \mathbf{y}) d\mathbf{f} = \frac{1}{p(\mathbf{y})} \int p(\mathbf{y} | \mathbf{f}) p(\mathbf{f}, \mathbf{f}_*) d\mathbf{f}, \quad (3.4)$$

The joint GP prior and the independent likelihood are both Gaussian

$$p(\mathbf{f}, \mathbf{f}_*) = \mathcal{N}\left(0, \begin{bmatrix} K_{\mathbf{f}, \mathbf{f}} & K_{*, \mathbf{f}} \\ K_{\mathbf{f}, *} & K_{*, *} \end{bmatrix}\right), \quad p(\mathbf{y} | \mathbf{f}) = \mathcal{N}(\mathbf{f}, \sigma_{\text{noise}}^2 I) \quad (3.5)$$

then

$$p(\mathbf{f}_* | \mathbf{y}) = \mathcal{N}(K_{*, \mathbf{f}}(K_{\mathbf{f}, \mathbf{f}} + \sigma_{\text{noise}}^2 I)^{-1} \mathbf{y}, K_{*, *} - K_{*, \mathbf{f}}(K_{\mathbf{f}, \mathbf{f}} + \sigma_{\text{noise}}^2 I)^{-1} K_{\mathbf{f}, *}) \quad (3.6)$$

3.1.2 A New Unifying View

Consider using **inducing variables** $\mathbf{u} = [u_1, u_2, \dots, u_m]^\top$ rewrite the joint prior, corresponding to a set of input location $X_{\mathbf{u}}$ which we called **inducing inputs**. Due to the consistency of Gaussian processes, we know

$$p(\mathbf{f}_*, \mathbf{f}) = \int p(\mathbf{f}_*, \mathbf{f}, \mathbf{u}) d\mathbf{u} = \int p(\mathbf{f}_*, \mathbf{f} | \mathbf{u}) p(\mathbf{u}) d\mathbf{u}, \quad \text{where } p(\mathbf{u}) = \mathcal{N}(0, K_{\mathbf{u}, \mathbf{u}}) \quad (3.7)$$

This is an exact expression. Now, we introduce the fundamental approximation which gives rise to almost all sparse approximations. We approximate the joint prior by assuming that \mathbf{f}_* and \mathbf{f} are conditionally independent given \mathbf{u} .

$$p(\mathbf{f}_*, \mathbf{f}) \simeq q(\mathbf{f}_*, \mathbf{f}) = \int q(\mathbf{f}_* | \mathbf{u}) q(\mathbf{f} | \mathbf{u}) p(\mathbf{u}) d\mathbf{u} \quad (3.8)$$

Let us specify two conditionals (using the **proof**)

$$\begin{aligned} \text{training conditional : } q(\mathbf{f}|\mathbf{u}) &= \mathcal{N}(K_{\mathbf{f},\mathbf{u}}K_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, K_{\mathbf{f},\mathbf{f}} - Q_{\mathbf{f},\mathbf{f}}) \\ \text{test conditional : } q(\mathbf{f}_*|\mathbf{u}) &= \mathcal{N}(K_{*,\mathbf{u}}K_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, K_{*,*} - Q_{*,*}) \end{aligned} \quad (3.9)$$

where $Q_{a,b} = K_{a,\mathbf{u}}K_{\mathbf{u},\mathbf{u}}^{-1}K_{\mathbf{u},b}$. We emphasize that all the sparse methods discussed in the paper correspond simply to different approximations to the conditionals in 3.9.

3.1.3 The Subset of Data (SoD) Approximation

Computation complexity is $\mathcal{O}(m^3)$, where $m < n$, only mentioned as a baseline method.

3.1.4 The Subset of Regressors (SoR) Approximation

SoR models are finite linear-in-the-parameters models with a particular prior on the weights. For any input \mathbf{x}_* , the corresponding function value \mathbf{f}_* is given by (this is the prior):

$$\mathbf{f}_* = K_{*,\mathbf{u}}\mathbf{w}_{\mathbf{u}}, \quad \text{with } p(\mathbf{w}_{\mathbf{u}}) = \mathcal{N}(0, K_{\mathbf{u},\mathbf{u}}^{-1}) \quad (3.10)$$

where $\mathbf{w}_{\mathbf{u}} = K_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}$, because (the left part, the right part is the prior on the inducing point)

$$\mathbf{u} = K_{\mathbf{u},\mathbf{u}}\mathbf{w}_{\mathbf{u}} \Rightarrow \langle \mathbf{u}\mathbf{u}^\top \rangle = K_{\mathbf{u},\mathbf{u}}\langle \mathbf{w}_{\mathbf{u}}\mathbf{w}_{\mathbf{u}}^\top \rangle K_{\mathbf{u},\mathbf{u}} = K_{\mathbf{u},\mathbf{u}} \quad (3.11)$$

then it is clear

$$\mathbf{f}_* = K_{*,\mathbf{u}}K_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, \quad \text{with } \mathbf{u} \sim \mathcal{N}(0, K_{\mathbf{u},\mathbf{u}}) \quad (3.12)$$

Then the conditionals are

$$q_{\text{SoR}}(\mathbf{f}|\mathbf{u}) = \mathcal{N}(K_{\mathbf{f},\mathbf{u}}K_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, 0), \quad q_{\text{SoR}}(\mathbf{f}_*|\mathbf{u}) = \mathcal{N}(K_{*,\mathbf{u}}K_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, 0) \quad (3.13)$$

We can then get the prior

$$q_{\text{SoR}}(\mathbf{f}, \mathbf{f}_*) = \mathcal{N}\left(0, \begin{bmatrix} Q_{\mathbf{f},\mathbf{f}} & Q_{\mathbf{f},*} \\ Q_{*,\mathbf{f}} & Q_{*,*} \end{bmatrix}\right) \quad (3.14)$$

Proof Let

$$\mathbf{y} = \begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} = \begin{bmatrix} K_{\mathbf{f},\mathbf{u}} \\ K_{*,\mathbf{u}} \end{bmatrix} K_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u} = AK_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}$$

then using the **linear transform rule** we get

$$\mathbf{y} \sim \mathcal{N}(0, AK_{\mathbf{u},\mathbf{u}}^{-1}A^\top)$$

A more descriptive name for this method, would be the **Deterministic Inducing Conditional** (DIC) approximation. We see that this approximate prior is degenerate, zero variance on the conditionals will lead to a over confident prediction.

Finally we get to write the predictive distribution

$$\begin{aligned} q_{\text{SoR}}(\mathbf{f}_*|\mathbf{y}) &= \mathcal{N}(Q_{*,\mathbf{f}}(Q_{\mathbf{f},\mathbf{f}} + \sigma_{\text{noise}}^2 I)^{-1}\mathbf{y}, Q_{*,*} - Q_{*,\mathbf{f}}(Q_{\mathbf{f},\mathbf{f}} + \sigma_{\text{noise}}^2 I)^{-1}Q_{\mathbf{f},*}) \\ &= \mathcal{N}(\sigma^{-2}K_{*,\mathbf{u}}\Sigma K_{\mathbf{u},\mathbf{f}}\mathbf{y}, K_{*,\mathbf{u}}\Sigma K_{\mathbf{u},*}) \end{aligned} \quad (3.15)$$

where we have defined $\Sigma = (\sigma^{-2}K_{\mathbf{u},\mathbf{f}}K_{\mathbf{f},\mathbf{u}} + K_{\mathbf{u},\mathbf{u}})^{-1}$. The second formulation is computationally cheaper!

The new covariance function has rank (at most) m , thus

Remark The SoR approximation is equivalent to exact inference in the degenerate Gaussian process with covariance function $k_{\text{SoR}}(\mathbf{x}_i, \mathbf{x}_j) = k(\mathbf{x}_i, \mathbf{u})K_{\mathbf{u},\mathbf{u}}^{-1}k(\mathbf{u}, \mathbf{x}_j)$.

Compared to the Subset of Data, the SoR has the complexity of $\mathcal{O}(nm^2)$, and $\mathcal{O}(m)$ and $\mathcal{O}(m^2)$ for predictive mean and variance.

3.1.5 The Deterministic Training Conditional (DTC) Approximation

Seeger et al. (2003), who called the method Projected Latent Variables (PLV), presented the method as relying on a likelihood approximation, based on the projection $\mathbf{f} = K_{\mathbf{f},\mathbf{u}} K_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{u}$

$$p(\mathbf{y}|\mathbf{f}) \simeq q(\mathbf{y}|\mathbf{u}) = \mathcal{N} \left(K_{\mathbf{f},\mathbf{u}} K_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{u}, \sigma^2 I \right) \quad (3.16)$$