Tyler Gordon
September 17, 2018

1 Input coordinates in second dimension are same as data coordinates

In the 1D case, the predictive distribution of a GP at coordinates $\boldsymbol{y^*} = (t_1^*, t_2^* ... t_p^*)^T$ has a mean given by

$$\mu_p^* = \mu_\theta(t_p^*) + K(t_p^*, \boldsymbol{t})\boldsymbol{z} \tag{1}$$

where $\boldsymbol{z} = K(\boldsymbol{t}, \boldsymbol{t})^{-1}[\boldsymbol{y} - \boldsymbol{\mu_\theta}(\boldsymbol{t})]$. In the 2D case for a GP with covariance in the second dimension defined the covariance matrix $Q(\boldsymbol{u}, \boldsymbol{u})$, the predictive distribution is given by

$$\boldsymbol{\mu_p^*} = \boldsymbol{\mu_\theta}(t_p^*) + K(t_p^*, \boldsymbol{t})X \tag{2}$$

where $X_p = Q(\boldsymbol{u}, \boldsymbol{u})\boldsymbol{z}_{pM:(p+1)M}$ is a vector of length M and $\boldsymbol{z} = K_{2D}^{-1}[\boldsymbol{y} - \boldsymbol{\mu_\theta}(\boldsymbol{t})]$ where K_{2D} is the full 2D covariance matrix and $\boldsymbol{y} = [y(t_1, u_1), y(t_1, u_2), ...y(t_1, u_M), y(t_2, u_1), ...y(t_N, u_M)]^T$ is the dataset on which the GP is conditioned. The algorithm for computing the predictive distribution is given by equations B26-B30 in FM17 with the substitution of X for $\boldsymbol{z}$. The result is a matrix $\mu^* \in \mathbb{Z}^{NxM}$ where $\mu_{n,m}$ is the predicted mean at coordinates (t_n, u_q).

2 Input coordinates in second dimension are different from data coordinates

When the input coordinates for the second dimension do not coincide with the data coordinates, the matrix X becomes

$$X_p = Q(\boldsymbol{u}^*, \boldsymbol{u})\boldsymbol{z}_{pM:(p+1)M} \tag{3}$$

From which we see that $\boldsymbol{X}_p$ is just the mean of the predictive distribution for a 1D process with kernel Q at input coordinates $\boldsymbol{u}^*$. If the covariance in the second dimension is also defined by a celerite kernel, the cost of computing each $\boldsymbol{X}_p$ is $\mathcal{O}(mM + rR)$ where M is the number of data coordinates in the second dimension, R is the number of input coordinates, and m and r are some constants. Therefore the total cost of computing X is $\mathcal{O}(mMP + rRP)$, and the total cost of the 2D prediction algorithm is $\mathcal{O}(mMP + rRP + pP + nN) \sim \mathcal{O}((mM + rR + p)P + nN)$