Supplementary Materials for: Product Kernel Interpolation for Scalable Gaussian Processes

Jacob R. Gardner[1], Geoff Pleiss[1], Ruihan Wu[1,2], Kilian Q. Weinberger[1], Andrew Gordon Wilson[1]
[1]Cornell University, [2]Tsinghua University

S1 Proof of Lemma 3.1

Letting $\mathbf{q}_i^{(1)}$ denote the i^{th} row of $Q^{(1)}$ and $\mathbf{q}_i^{(2)}$ denote the i^{th} row of $Q^{(2)}$, we can express the i^{th} entry $K\mathbf{v}$, $[K\mathbf{v}]_i$ as:

$$[K\mathbf{v}]_i = \mathbf{q}_i^{(1)} T^{(1)} Q^{(1)\top} \, D_{\mathbf{v}} \, Q^{(2)} T^{(2)} \mathbf{q}_i^{(2)\top}$$

To evaluate this for all i, we first once compute the $k \times k$ matrix:

$$M^{(1,2)} = T^{(1)} Q^{(1)\top} \, D_{\mathbf{v}} \, Q^{(2)} T^{(2)}.$$

This can be done in $O(nk^2)$ time. $T^{(1)} Q^{(1)\top}$ and $Q^{(2)} T^{(2)}$ can each be computed in $O(nk^2)$ time, as the Q matrices are $n \times k$ and the T matrices are $n \times k$. Multiplying one of the results by $D_{\mathbf{v}}$ takes $\mathcal{O}(nk)$ time as it is diagonal. Finally, multiplying the two resulting $n \times k$ matrices together takes $\mathcal{O}(nk^2)$ time.

After computing $M^{(1,2)}$, we can compute each element of the matrix-vector multiply as:

$$[K\mathbf{v}]_i = \mathbf{q}_i^{(1)} M^{(1,2)} \mathbf{q}_i^{(2)\top}.$$

Because $M^{(1,2)}$ is $k \times k$, each of these takes $\mathcal{O}(k)$ time to compute. Since there are n entries to evaluate in the MVM $K\mathbf{v}$ in total, the total time requirement after computing $M^{(1,2)}$ is $\mathcal{O}(kn)$ time. Thus, given low rank structure, we can compute $K\mathbf{v}$ in $\mathcal{O}(k^2 n)$ time total.

S2 Proof of Theorem 3.3

Given the Lanczos decompositions of $\tilde{K}^{(1)} = K_{XX}^{(1)} \circ \cdots \circ K_{XX}^{(a)}$ and $\tilde{K}^{(2)} = K_{XX}^{(a+1)} \circ \cdots \circ K_{XX}^{(d)}$, we can compute matrix-vector multiplies with $\tilde{K}^{(1)} \circ \tilde{K}^{(2)}$ in $\mathcal{O}(k^2 n)$ time each. This lets us compute the Lanczos decomposition of $\tilde{K}^{(1)} \circ \tilde{K}^{(2)}$ in $\mathcal{O}(k^3 n)$ time.

For clarity, suppose first that $d = 3$, i.e., $K = K_{XX}^{(1)} \circ K_{XX}^{(2)} \circ K_{XX}^{(3)}$. We first Lanczos decompose $K_{XX}^{(1)}$, $K_{XX}^{(2)}$ and $K_{XX}^{(3)}$. Assuming for simplicty that MVMs with each matrix take the same amount of time, This takes $\mathcal{O}(k\mu(K_{XX}^{(i)}))$ time total. We then use these Lanczos decompositions to compute matrix-vector multiples with $\tilde{K}_{XX}^{(1)}$ in $\mathcal{O}(k^2 n)$time each. This allows us to Lanczos decompose it in $\mathcal{O}(k^3 n)$ time total. We can then compute matrix-vector multiplications $K\mathbf{v}$ in $\mathcal{O}(k^2 n)$ time.

In the most general setting where $K = K_{XX}^{(1)} \circ \cdots \circ K_{XX}^{(d)}$, we first Lanczos decompose the d component matrices in $\mathcal{O}(dk\mu(K^{(i)}))$ and then perform $\mathcal{O}(\log d)$ merges as described above, each of which takes $\mathcal{O}(k^3 n)$ time. After computing all necessary Lanczos decompositions, matrix-vector multiplications with K can be performed in $\mathcal{O}(k^2 n)$ time.

As a result, a single matrix-vector multiply with K takes $\mathcal{O}(dk\mu(K^{(i)}) + k^3 n \log d + k^2 n)$ time. With the Lanczos decompositions precomputed, multiple MVMs in a row can be performed significantly faster. For example, running p iterations of conjugate gradients with K takes $\mathcal{O}(dk\mu(K^{(i)}) + k^3 n \log d + pk^2 n)$ time.