Appendices

Table 4: Symbolic notation for EigenPro 3.0 in Algorithm 1. They satisfy $m < n$, and $q < s < n$.

Symbol	Purpose
n	Number of samples
m	Batch-size
p	Model size
s	Nyström approximation subsample size
q	Preconditioner level

A PROOFS OF INTERMEDIATE RESULTS

A.1 PROOF OF PROPOSITION 1

Proposition (Nyström extension)**.** For $1 \leq i \leq n$, let λ_i be an eigenvalue of $\mathcal{K}$, and ψ_i its unit $\boldsymbol{\mathcal{H}}$-norm eigenfunction, i.e., $\mathcal{K}\left\{\psi_i\right\} = \lambda_i \psi_i$. Then λ_i is also an eigenvalue of $K(X,X)$. Moreover if $\boldsymbol{e}_i$, is its unit-norm eigenvector, i.e., $K(X,X)\boldsymbol{e}_i = \lambda_i \boldsymbol{e}_i$, we have,

$$\psi_i = K(\cdot, X)\frac{\boldsymbol{e}_i}{\sqrt{\lambda_i}}. \tag{37}$$

Proof. Let $\psi \in \mathcal{H}$ be an eigenfunction of $\mathcal{K}$. Then by definition of $\mathcal{K}$ we have,

$$\lambda\psi = \mathcal{K}\left\{\psi\right\} = \sum_{i=1}^{n} K(\cdot, \boldsymbol{x}_i)\psi(\boldsymbol{x}_i). \tag{38}$$

As the result we can write ψ as below,

$$\psi = \sum_{i=1}^{n} \frac{\psi(\boldsymbol{x}_i)}{\lambda} K(\cdot, \boldsymbol{x}_i). \tag{39}$$

If we apply covariance operator to the both side of 39 we have,

$$\mathcal{K}\left\{\psi\right\} = \mathcal{K}\left\{\sum_{i=1}^{n} \frac{\psi(\boldsymbol{x}_i)}{\lambda} K(\cdot, \boldsymbol{x}_i)\right\} = \sum_{i,j=1}^{n} \frac{\psi(\boldsymbol{x}_i)}{\lambda} K(\boldsymbol{x}_i, \boldsymbol{x}_j) K(\cdot, \boldsymbol{x}_j) = \sum_{j=1}^{n} \psi(\boldsymbol{x}_j) K(\cdot, \boldsymbol{x}_j). \tag{40}$$

The last equation hold because of equation (38). If we define vector $\boldsymbol{\beta}$ such that $\boldsymbol{\beta}_i = \frac{\psi(x_i)}{\lambda}$, then 40 can be rewritten as,

$$\sum_{i=1}^{n}\sum_{j=1}^{n} \boldsymbol{\beta}_i K(\boldsymbol{x}_i, \boldsymbol{x}_j) K(\cdot, \boldsymbol{x}_i) = \lambda \sum_{i=1}^{n} \boldsymbol{\beta}_i K(\cdot, \boldsymbol{x}_i). \tag{41}$$

Compactly we can write 41 as below,

$$K(X,X)^2\boldsymbol{\beta} = \lambda K(X,X)\boldsymbol{\beta} \implies K(X,X)\boldsymbol{\beta} = \lambda\boldsymbol{\beta}.$$

The last implication holds because $K(X,X)$ is invertable. Thus $\boldsymbol{\beta}$ is an eigenvector of $K(X,X)$. It remains to determine the scale of $\boldsymbol{\beta}$.

Now, norm of ψ can be simplified as

$$\left\|\psi\right\|_{\boldsymbol{\mathcal{H}}}^2 = \left\langle \sum_{i=1}^{n} \beta_i K(\cdot, \boldsymbol{x}_i), \sum_{j=1}^{n} \beta_j K(\cdot, \boldsymbol{x}_j) \right\rangle_{\boldsymbol{\mathcal{H}}} \tag{42}$$

$$= \sum_{i,j=1}^{n} \beta_i \beta_j \left\langle K(\cdot, \boldsymbol{x}_i), K(\cdot, \boldsymbol{x}_j)\right\rangle_{\boldsymbol{\mathcal{H}}} = \boldsymbol{\beta}^\top K(X,X)\boldsymbol{\beta} = \lambda \left\|\boldsymbol{\beta}\right\|^2. \tag{43}$$

Since ψ is unit norm, we have $\|\boldsymbol{\beta}\| = \frac{1}{\sqrt{\lambda}}$. This concludes the proof. $\square$

A.2 PROOF OF LEMMA 2

Lemma (Nyström preconditioning)**.** Let $\boldsymbol{a} \in \mathbb{R}^m$, then we have that,

$$\mathcal{P}_s\left\{K(\cdot, X_m)\boldsymbol{a}\right\} = K(\cdot, X_m)\boldsymbol{a} - K(\cdot, X_s)\boldsymbol{Q}_s K(X_s, X_m)\boldsymbol{a}. \tag{44}$$

Where $Q_s = E_{s,q}(\boldsymbol{I}_n - \lambda_{s,q+1}\Lambda_{s,q}^{-1})\Lambda_{s,q}^{-1}E_{s,q}^\top$.

Proof. Recall that $\mathcal{P}_s := \mathcal{I} - \sum_{i=1}^{q}\left(1 - \frac{\lambda_{q+1}}{\lambda_q}\right)\psi_i \otimes \psi_i$. By this definition we can write,

$$\begin{aligned}
\mathcal{P}_s\left(K(\cdot, X_M)\boldsymbol{\alpha}\right) &= K(\cdot, X_M)\boldsymbol{\alpha} - \sum_{i=1}^{s}(1 - \frac{\lambda_{q+1}^s}{\lambda_i^s})\left\langle \psi_i^s, K(\cdot, X_M)\boldsymbol{\alpha}\right\rangle_{\boldsymbol{\mathcal{H}}}\psi_i^s \\
&= K(\cdot, X_M)\boldsymbol{\alpha} - \sum_{i=1}^{q}\frac{1}{\lambda_i^s}(1 - \frac{\lambda_{q+1}^s}{\lambda_i^s})\left\langle K(\cdot, X_s)\boldsymbol{e}_i, K(\cdot, X_M)\boldsymbol{\alpha}\right\rangle_{\boldsymbol{\mathcal{H}}} K(\cdot, X_s)\boldsymbol{e}_i \\
&= K(\cdot, X_M)\boldsymbol{\alpha} - \sum_{i=1}^{q}\frac{1}{\lambda_i^s}(1 - \frac{\lambda_{q+1}^s}{\lambda_i})\left\langle K(\cdot, X_s)\boldsymbol{e}_i, K(\cdot, X_M)\boldsymbol{\alpha}\right\rangle_{\boldsymbol{\mathcal{H}}} K(\cdot, X_s)\boldsymbol{e}_i \\
&= K(\cdot, X_M)\boldsymbol{\alpha} - \sum_{i=1}^{q}(1 - \frac{\lambda_{q+1}^s}{\lambda_i^s})K(\cdot, X_s)\boldsymbol{e}_i\boldsymbol{e}_i^\top K(X_s, X_M)\boldsymbol{\alpha}.
\end{aligned}$$

Note that we used proposition 1 for ψ. Now we can compactly write the last expression as below,

$$\begin{aligned}
\mathcal{P}_s\left(K(\cdot, X_M)\boldsymbol{\alpha}\right) &= K(\cdot, X_M)\boldsymbol{\alpha} - K(\cdot, X_s)E_{s,q}(\boldsymbol{I}_n - \lambda_{s,q+1}\Lambda_{s,q}^{-1})\Lambda_{s,q}^{-1}E_{s,q}^\top K(X_s, X_M)\boldsymbol{\alpha} \\
&= K(\cdot, X_M)\boldsymbol{\alpha} - K(\cdot, X_s)Q_s K(X_s, X_M)\boldsymbol{\alpha}.
\end{aligned}$$

This concludes the proof.

□

B DETAILS ON EigenPro 2.0

Lemma 3. *The iteration in* $\mathbb{R}^n$

$$\boldsymbol{\alpha}^{t+1} = \boldsymbol{\alpha}^{t+1} - \eta(\boldsymbol{I}_n - \boldsymbol{Q})(K(X, X)\boldsymbol{\alpha}^t - \boldsymbol{y}), \tag{45}$$

where $Q = \boldsymbol{E}(\boldsymbol{I}_n - \lambda_{q+1}\Lambda_q^{-1})\boldsymbol{E}^\top$*, emulates the following iteration in* $\boldsymbol{\mathcal{H}}$*.*

$$f^{t+1} = f^t - \eta\mathcal{P}\left\{\nabla_f L(f^t)\right\}. \tag{46}$$

Proof. Recall that $\nabla_f L(f^t) = K(\cdot, X)(f^t(X) - \boldsymbol{y})$ from equation (10), and $f^t(X) = K(X, X)\boldsymbol{\alpha}^t$. from equation (19). We define $\boldsymbol{g}^t := f^t(X) - \boldsymbol{y} = K(X, X)\boldsymbol{\alpha}^t - \boldsymbol{y}$. Following steps of the proof in Appendix A.2 we have

$$\begin{aligned}
\mathcal{P}\{\nabla_f L(f^t)\} &= K(\cdot, X)\boldsymbol{g}^t - \sum_{i=1}^{q}(1 - \frac{\lambda_{q+1}}{\lambda_i})K(\cdot, X)\boldsymbol{e}_i^\top\boldsymbol{e}_i K(X, X)\boldsymbol{g}^t \\
&= K(\cdot, X)\boldsymbol{g}^t - K(\cdot, X)\boldsymbol{E}(\boldsymbol{I}_n - \lambda_{q+1}\Lambda_q^{-1})\Lambda^{-1}\boldsymbol{E}^\top K(X, X)\boldsymbol{g}^t \\
&\overset{\text{(a)}}{=} K(\cdot, X)\boldsymbol{g}^t - K(\cdot, X)\boldsymbol{E}(\boldsymbol{I}_n - \lambda_{q+1}\Lambda_q^{-1})\Lambda^{-1}\boldsymbol{E}^\top\boldsymbol{E}\Lambda\boldsymbol{E}^\top\boldsymbol{g}^t \\
&= K(\cdot, X)\boldsymbol{g}^t - K(\cdot, X)\boldsymbol{E}(\boldsymbol{I}_n - \lambda_{q+1}\Lambda_q^{-1})\boldsymbol{E}^\top\boldsymbol{g}^t \\
&= K(\cdot, X)\boldsymbol{g}^t - K(\cdot, X)\boldsymbol{Q}\boldsymbol{g}^t \\
&= K(\cdot, X)(\boldsymbol{I}_n - \boldsymbol{Q})\boldsymbol{g}^t.
\end{aligned}$$

Algorithm 3 EigenPro 2.0$(X, \boldsymbol{y})$. Solves the linear system $K(X,X)\boldsymbol{\theta} = \boldsymbol{y}$

```
Require: Data $(X, \boldsymbol{y})$, Nyström size $s$, preconditioner level $q$
  $\boldsymbol{\alpha} \leftarrow \mathbf{0} \in \mathbb{R}^n$                                  ▷ initialization
  $X_s, (\boldsymbol{E}, \boldsymbol{D}), \lambda_{q+1}, m \leftarrow$ EigenPro 2.0_setup$(X, s, q)$
  Set batchsize $m \leftarrow \frac{1}{\lambda_{q+1}}$
  while Stopping criterion not reached do
      $\boldsymbol{\alpha} \leftarrow$ EigenPro 2.0_iteration$(X, \boldsymbol{y}, X_s, \boldsymbol{E}, \boldsymbol{D}, \boldsymbol{\alpha}, m, \eta)$
  end while
  return $\boldsymbol{\alpha}$

EigenPro2_setup$(X, s, q)$
Require: Data $X$, Nyström size $s$, preconditioner size $q$
  Fetch a subsample $X_s \subseteq X$ of size $s$
  $(\boldsymbol{E}, \Lambda) \leftarrow$ top-$q$ eigensystem of $K(X_s, X_s)$        ▷ $\boldsymbol{E} \in \mathbb{R}^{q \times s}, \Lambda = \mathrm{diag}(\lambda_i) \in \mathbb{R}^{q \times q}$
  $\boldsymbol{D}_{ii} = \frac{1}{s\lambda_i}\left(1 - \frac{\lambda_{q+1}}{\lambda_i}\right)$
  $\beta \leftarrow \max_i K(\boldsymbol{x}_i, \boldsymbol{x}_i) \in S$
  $m \leftarrow \min\left(\frac{\beta}{\lambda_{q+1}}, \mathrm{bs}_{\mathrm{gpu}}\right)$                        ▷ batch size[2]
  $\eta \leftarrow \begin{cases} \frac{\beta}{2m} & m < \frac{\beta}{\lambda_{q+1}} \\ \frac{0.99m}{\beta + (m-1)\lambda_{q+1}} & \text{otherwise} \end{cases}$        ▷ learning rate
  return $X_s, (\boldsymbol{E}, \boldsymbol{D}), \eta, m$

EigenPro2_iteration$(X, \boldsymbol{y}, X_s, \boldsymbol{E}, \boldsymbol{D}, \boldsymbol{\alpha}, m, \eta)$
Require: Data $(X, \boldsymbol{y})$, Nyström subset $X_s$, preconditioner $(\boldsymbol{E}, \boldsymbol{D})$, current estimate $\boldsymbol{\alpha}$, batchsize $m$
  Fetch minibatch $(X_m, \boldsymbol{y}_m)$ of size $m$
  $\boldsymbol{g}_m \leftarrow K(X_m, X)\boldsymbol{\alpha} - \boldsymbol{y}_m$                  ▷ stochastic gradient
  $\boldsymbol{\alpha}_m \leftarrow \boldsymbol{\alpha}_m - \frac{\eta}{m}\boldsymbol{g}_m$                       ▷ gradient step
  $\boldsymbol{\alpha}_s \leftarrow \boldsymbol{\alpha}_s + \boldsymbol{E}\boldsymbol{D}\boldsymbol{E}^\top K(X_s, X_m)\boldsymbol{g}_m$      ▷ gradient correction
  return Updated estimte $\boldsymbol{\alpha}$
```

Where (a) follows from $K(X,X) = \boldsymbol{E}\Lambda\boldsymbol{E}^\top$. Now since $f^t = K(\cdot, X)\boldsymbol{\alpha}^t$, equation (46) can be rewritten,

$$\begin{aligned} f^{t+1} &= K(\cdot, X)\boldsymbol{\alpha}^{t+1} - \eta K(\cdot, X)(\boldsymbol{I}_n - Q)\boldsymbol{g}^t \\ &= K(\cdot, X)(\boldsymbol{\alpha}^{t+1} - \eta(\boldsymbol{I}_n - Q)\boldsymbol{g}^t). \end{aligned}$$

Replacing $g^t = K(X,X)\boldsymbol{\alpha}^t - y$ leads to final update rule below,

$$f^{t+1} = K(\cdot, X)(\boldsymbol{\alpha}^{t+1} - \eta(\boldsymbol{I}_n - Q)(K(X,X)\boldsymbol{\alpha}^t - y)).$$

This concludes the proof. □

Thus each update constitutes a *stochastic gradient step* which consists updating m weights corresponding to a minibatch size m, followed by a *gradient correction* which consists of updating all n weights.

A higher preconditioner level q also allows for a higher optimal batch size m and hence better GPU utilization, see Ma et al. (2018) for details.

With this approximation, the gradient correction simplifies drastically, and only s weights need to be updated.

---

[2] bs$_{\mathrm{gpu}}$ is the maximum batch-size that the GPU allows.

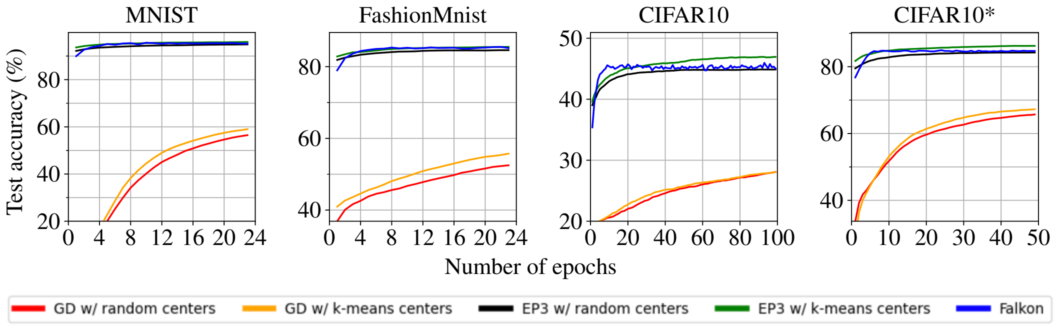

Figure 3: **Comparison with gradient descent and Falkon:** Figure 3 shows the slow convergence of gradient descent given in (50) compared to our algorithm and FALKON from Rudi et al. (2017). Note that FALKON involves a matrix inverse for a projection operation and hence converges faster.

C DETAILS ON EXPERIMENTS AND IMPLEMENTATION OF ALGORITHM 1

C.1 COMPUTATIONAL RESOURCES USED

This work used the Extreme Science and Engineering Discovery Environment (XSEDE) (Towns et al., 2014). We used machines with 2x NVIDIA-V100 GPUs, each with a memory of 32GB, and 4x cores of Intel(R) Xeon(R) Gold 6248 CPU @ 2.50GHz with a RAM of 100 GB.

C.2 CHOICE OF HYPERPARAMETERS

We choose hyperparameters to minimize computation and maximize GPU utilization. The only hyperparameters that we need to set are s, q for outer gradient step, and σ, ξ for projection sub-problem. For σ, ξ, we used the same criteria as Ma & Belkin (2019) to optimally use GPU utilization. For s, q, we prefer larger q because as it is explained in Ma et al. (2018), larger q allows for larger learning rate and better condition number. However, in our algorithm we need to approximate the top q eigensystem of Nyström sub-samples matrix. We used Scipy Virtanen et al. (2020) library to approximate these eigensystem. The stability and precision of these approximations depends on how large is the ratio of $\frac{s}{q}$. Empirically we need this ratio to be larger than 10. On the other hand increasing s will increase setup cost, computation cost and memory cost. We take steps below to choose q and s,

1. We first choose s as big as our GPU memory allow
2. We choose $q \approx \frac{s}{10}$
3. We set batch size and learning rate automatically using the *new* top eigenvalue as it is explained in Ma & Belkin (2019) and Ma et al. (2018).

D CLASSICAL APPROACH TO LEARNING KERNEL NETWORKS WITH GD

If you plug in the form of (4) into (1), we get

$$\underset{\boldsymbol{\alpha}}{\text{minimize}}\ L(\boldsymbol{\alpha}) = \sum_{i=1}^{n} L\Big(\sum_{j=1}^{p} K(\boldsymbol{x}_i, \mathbf{z}_j)\alpha_j, y_i\Big) + \lambda \left\langle \sum_{j=1}^{p} K(\cdot, \mathbf{z}_j), \sum_{j=1}^{p} K(\cdot, \mathbf{z}_j) \right\rangle_{\mathcal{H}} \tag{47}$$

$$= \sum_{i=1}^{n} L(\boldsymbol{I}_n^{(i)} K(X, Z)\boldsymbol{\alpha}, y_i) + \lambda \boldsymbol{\alpha}^\top K(Z, Z)\boldsymbol{\alpha}, \tag{48}$$

where $\boldsymbol{I}_n^{(i)}$ is the i^{th} row of identity $\boldsymbol{I}_n$. For the square loss this is

$$\underset{\boldsymbol{\alpha}}{\text{minimize}}\ \|K(X, Z)\boldsymbol{\alpha} - \boldsymbol{y}\|^2 + \lambda \boldsymbol{\alpha}^\top K(Z, Z)\boldsymbol{\alpha}. \tag{49}$$

Gradient descent on this problem for the square loss yields the update equation,

$$\boldsymbol{\alpha}^{t+1} = \boldsymbol{\alpha}^{t} - \eta K(Z, X)((K(X, Z)\boldsymbol{\alpha}^{t} - \boldsymbol{y}) - \eta\lambda K(Z, Z)\boldsymbol{\alpha}. \tag{50}$$