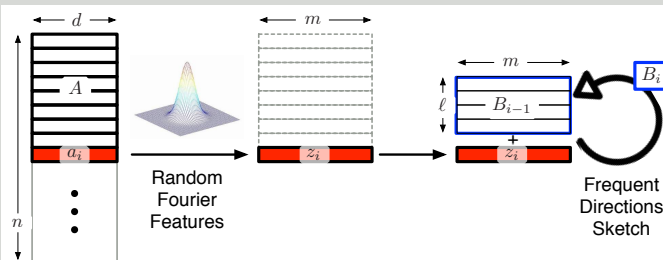


Objectives for a streaming KPCA

- ▶ Small space requirement
- ▶ Small training time (process training data)
- ▶ Small testing time (evaluate unseen test data)
- ▶ Bound on potential error



Algorithm: SKPCA

Input: Data $A \in \mathbb{R}^{n \times d}$, kernel K , $\ell, m \in \mathbb{Z}^+$

Output: RFF maps $[f_1, \dots, f_m]$, subspace W

$[f_1, \dots, f_m] = \text{RFF}(K, m)$

$B \leftarrow 0^{\ell \times m}$

for $i \in [n]$ do

$z_i = \sqrt{\frac{2}{m}} [f_1(a_i), \dots, f_m(a_i)]$ } RFF projection

$B \leftarrow z_i$

if B has no zero valued rows then

$[Y, \Sigma, W] \leftarrow \text{svd}(B)$

$B \leftarrow \sqrt{\max\{0, \Sigma^2 - \Sigma_{\ell/2, \ell/2}^2\}} \cdot W^T$ } Frequent Directions

end if

end for

Return $[f_1, \dots, f_m]$ and W

Theorem 1: Spectral error bound

Let $G = \Phi\Phi^T$ be the exact kernel matrix over n points. Let $\tilde{G} = ZW^TWZ^T$ be the result of Z from $m = O((1/\epsilon^2) \log(n/\delta))$ RFF and W from running Algorithm SKPCA with $\ell = 4/\epsilon$. Then with probability at least $1 - \delta$, we have $\|G - \tilde{G}\|_2 \leq \epsilon n$.

Theorem 2: Frobenius error bound

Given that $\|G - G'\|_2 \leq \epsilon n$ we can bound $\|G - G'_k\|_F \leq \|G - G_k\|_F + \epsilon\sqrt{kn}$.

Runtime bounds to obtain $\|G' - G\|_2 \leq \epsilon n$

	TRAIN TIME	TEST TIME
KPCA	$O(n^2(n+d))$	$O(n(d+n^2))$
NYSTRÖM	$O(nd + n/\epsilon^2 + 1/\epsilon^4)$	$O(d/\epsilon^2 + 1/\epsilon^4)$
RNCA	$O(n((d/\epsilon^2) \log n + (1/\epsilon^4) \log^2 n))$	$O((1/\epsilon^2)(d+n) \log n)$
SKPCA	$O(n \log n(d/\epsilon^2 + 1/\epsilon^3))$	$O((d+1/\epsilon)/\epsilon^2 \log n)$

Space bounds to obtain $\|G' - G\|_2 \leq \epsilon n$

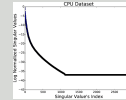
	SPACE
KPCA	$O(n^2 + nd)$
NYSTRÖM	$O(d/\epsilon^2 + 1/\epsilon^4)$
RNCA	$O((d/\epsilon^2)n \log n)$
SKPCA	$O(((d+1/\epsilon)/\epsilon^2) \log n)$

Previous work

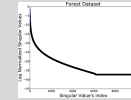
- ▶ Existing approaches to streaming/online KPCA either provide no error bound, require substantial space during training time, or have an expensive matrix inverse at test time.
- ▶ *Incremental KPCA techniques* update the eigenspace of kernel PCA without storing training data, but suffer from unbounded compound error in intermediate approximations of the eigenspace on adversarial data sets.
- ▶ *Nystrom approximation* methods approximate the kernel (Gram) matrix $G = CW_k^T C^T$, by sampling columns of G in a non-streaming setting, but require a costly matrix inverse at test time.
- ▶ *Randomized Nonlinear Component Analysis (RNCA)* uses a *Random Fourier Feature (RFF)* approximation to G via randomized feature maps by directly approximating the lifting function, but use an exact (costly) covariance computation.
- ▶ We propose *Streaming KPCA (SKPCA)*, combining the computational benefits of Random Fourier Features (RFF) and approximation bounds of Frequent Directions (FD) to achieve the stated goals.

Datasets

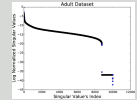
- ▶ Methods were compared on real and synthetic datasets, including three real datasets below from the UCI machine learning repository.
- ▶ The kernel matrix was found using an RBF kernel (or RFF equivalent) with the bandwidth set to the average inter-point distance – the spectra and input data sizes from the three datasets are shown below.



CPU
7373 × 21

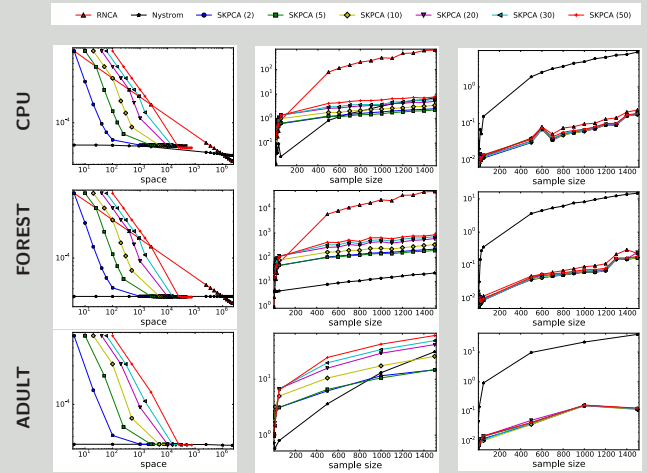


FOREST
523910 × 54



ADULT
33561 × 123

Results



- ▶ **Frobenius error** is the approximation error, $\|G' - G\|_F$.
- ▶ **Training time (sec)** and **Testing time (sec)**
- ▶ The size of the sketch, ℓ , is a parameter and we compare several choices, $\ell = \{2, 5, 10, 20, 30, 50\}$, indicated by parenthesis in the legend.

Discussion

- ▶ *Nystrom*: fast training time (random sampling), considerably slower testing time due to sample Gram matrix inversion.
- ▶ *RNCA*: fast testing time (matrix multiplication), training slower because complete covariance accumulation as data are observed
- ▶ *SKPCA*: obtains a more balanced runtime where both training and testing are competitive
 - ▶ Error is competitive with previous methods (all methods less than 10^{-3} in error)
 - ▶ Improved error vs space for RFF based methods