

Sparse Nystrom Approximation of Currents and Varifolds ^{*}

Allen Paul [†], Neill Campbell [‡], and Tony Shardlow [§]

Abstract. We derive an algorithm for compression of the currents and varifolds representations of shapes, using the Nystrom approximation in Reproducing Kernel Hilbert Spaces. Our method is faster than existing compression techniques, and comes with theoretical guarantees on the rate of convergence of the compressed approximation, as a function of the smoothness of the associated shape representation. The obtained compression are shown to be useful for down-line tasks such as nonlinear shape registration in the Large Deformation Metric Mapping (LDDMM) framework, even for very high compression ratios. The performance of our- algorithm is demonstrated on large-scale shape data from modern geometry processing datasets, and is shown to be fast and scalable with rapid error decay.

Key words. Currents and Varifolds, Compression, Sparsity, Convergence.

AMS subject classifications. 65D18, 65D15 , 68W20, 68W25

1. Introduction. Comparing and computing distances between geometric structures is a fundamental task in computer vision and geometric learning applications. Most commonly, such problems arise when building models of shape variation in a given application domain, for example, in computational anatomy where shapes are most commonly available as discrete curves and surfaces. When building such models, one requires a metric on shapes in order to best tune the parameters of the model to match observed variation in the data. Ideally, the available shape data for the given task is in parametric correspondence making the comparison trivial. However, in realistic shape learning tasks this is far from the case with shape data acquired by completely different methods, with inconsistent parametrizations and differing resolutions across a dataset.

In the case where shape data is available as sub-manifolds of \mathbb{R}^d , one can deal with the lack of parametric correspondences in a more principled manner, using techniques from geometric measure theory [9]. In particular, using the so-called currents [19], varifolds [10] and normal cycles [18] representation of shapes. These representations view shapes as objects that integrate differential forms on the underlying domain. In dimensions $d = 2, 3$ by choosing these differential forms to lie in a Hilbert space, these representations essentially embed the shapes into a dual space of differential forms. Using the dual metric on these representations allows one to compute the distance between shapes, in terms of their action of vector fields and not in terms of their parametrizations. The computation of the dual metric between submanifolds can be written down explicitly when the Hilbert space is a Reproducing Kernel Hilbert space, induced by a choice of positive definite kernel function $k : X \times X \rightarrow \mathbb{R}$. One

*

Funding: Allen Paul is supported by a scholarship from the EPSRC Centre for Doctoral Training in Statistical Applied Mathematics at Bath (SAMBa), under the project EP/S022945/1.

[†]University of Bath ap2746@bath.ac.uk.

[‡]University of Bath n.campbell@bath.ac.uk

[§]University of Bath t.shardlow@bath.ac.uk.

can also perform the same technique for discrete shape data in a way that is consistent as the resolution of the data tends to infinity. This framework has been used extensively in the LDDMM literature [9], for matching shapes with diffeomorphisms, using the functional shape metrics as the discrepancy term for matching.

However, while the aforementioned shape metrics account for the lack of parametric correspondence when comparing shapes, the practical computation of these metrics can be costly. For example, if one computes the currents metric between two triangulated surfaces with M and N triangles respectively, the metric computation has complexity $\mathcal{O}(MN)$. In modern geometry processing applications when M and N can exceed 10^4 , this becomes far too expensive both memory and computation wise. A similar issue holds for both the varifolds and normal cycles representations.

In order to deal with the cost of comparing shapes as geometric measures, there are many methods to approximately compute the metric. Approximate methods include fast multipole methods, gridding and grid based FFT methods, which are reviewed in [9]. However, these methods only approximate the metric computation itself, and do not provide approximants for the gradients of the metric, which are required for many shape registration algorithms using currents and varifolds. Furthermore, they are often restricted to smooth scalar radial kernels such as the Gaussian RBF kernel.

A recent development to allow large-scale exact metric computation (and therefore exact gradients), is using efficient GPU tiling schemes with CUDA and C++ as in the KeOps library [8]. This method allow scalable and fast metric computation up to a limit, typically 10^5 landmarks. Past this size the KeOps based method can become slow, especially for shape matching and groupwise registration due to repeated metric and gradient of metric computations.

1.1. Contribution. The main contribution of this work is a fast algorithm for compressing massively over-sampled currents/varifolds associated to shapes, using the Nystrom approximation and Reproducing Kernel Hilbert Space (RKHS) theory. The proposed algorithm is much faster than existing compression techniques, showing 10-100 times speed-ups in real-world experiments. We compress target currents and varifolds of effective complexity $N, M \geq 10^5$, by forming sparse approximations of effective size n and m such that $n \ll N, m \ll M$. Given two shapes of resolution M, N , post compression one can compute exact distances and gradients of distances between compressed shapes at cost $\mathcal{O}(mn)$, which gives significant savings when $m \ll M, n \ll N$. This can help massively speed up subsequent geometric learning algorithms, as well reduce memory issues with metric computations.

Though the algorithm introduces an approximation error, we provide theoretical guarantees on the error that allows us to make estimates of the desired sparsity of compression. Indeed, our main result [Theorem 5.1](#) and [Corollary 5.2](#) together, implies an error decay at an exponential rate in the Gaussian kernel setting

$$\|\mu - \hat{\mu}\|_{W^*}^2 \leq C \sum_{i=m+1}^{\infty} \lambda_i = \mathcal{O}(m \exp(-\alpha m^{\frac{1}{d}})),$$

for some $\alpha > 0$ and where $\hat{\mu}$ is an approximation of ‘size’ m to the original current/varifold μ in the dual of RKHS W . Such estimates hold both in expectation and high probability with

respect to random noisy samples of the original underlying shape.

1.2. Outline. In [section 2](#), we discuss the existing approaches to compression of measure-based shape representations, and compare to our work. Subsequently, we review the currents and varifolds representation of shape in [section 3](#). We introduce the Nystrom approximation and its RKHS variant in [section 4](#), as well as existing bounds on the error of this approximation. In [section 5](#), we present the proposed compression algorithm, theoretical guarantees and its application to compression of currents and varifolds, as well as LDDMM matching. In [section 6](#) we present proofs of the main results. Finally, we demonstrate the strengths and weaknesses of the proposed compression algorithm against existing work in [section 7](#), on large-scale shape data from modern geometry processing datasets. Finally, we discuss future work and possible extensions in [section 8](#).

2. Related Work.

2.1. Compression of currents and varifolds. The most closely related methods to this work are compression algorithms developed separately for currents and varifolds. In such methods, one builds a sparse greedy approximation to a given current/varifold based on a finite dictionary. To the best of our knowledge, there are three works [[14](#), [12](#), [16](#)] focusing on compression of measure-based representations of shapes. The first work [[14](#)] only applies to the case of compression of curve measures, which makes it restricted to specific applications with curve data such as sulcal lines and 2D curve data. Therefore, our work is best compared against the more general works of [[12](#)] and [[16](#)] which apply to compressing general submanifolds.

The work of [[12](#)] is the first work to consider compression of measure-based representation of shapes. In particular, the compression of large-scale currents. This algorithm is an iterative greedy approach to approximating a target current. At each step, one adds a Dirac delta measure that is most correlated with the current residual measure, which is updated after each step. This algorithm guarantees decrease of the approximation error, and comes with an exponential convergence rate to the target current. However in practice, the computation of the residual and maximization step tends to be very expensive when running this algorithm on large-scale currents. One can form approximation to this using grid-based projections and the Fast Fourier Transform. However, the iterative nature of the algorithm and the per-iteration cost means it can be very slow to compress a given large-scale current with $\sim 10^5$ landmarks. As a result, such an algorithm is not suitable when one is interested in computing compressions of multiple currents, or if one wishes to perform the compression routinely as part of an algorithm for shape analysis, in reasonable time scales. Furthermore, the grid-based nature of the algorithm that makes it well suited for currents is a drawback if one wishes to compress varifolds, as the cardinality of the grid representation explodes with the inclusion of the tangential component of varifolds.

The work of [[16](#)] applies to compression of varifold measures. This works by fixing a compression level m , and minimizing the discrepancy between the target varifold, and a discrete varifold of size m , using non-convex optimization techniques applied to the error in the varifolds metric. However, the optimization itself requires repeatedly computing the varifolds distance to the target measure, which is expensive. Furthermore, the non-convex nature of the associated minimization problem means there is no guarantee that one finds a good solution

without a careful initialization. This method also does not give any theoretical guarantees on the quality of approximation, which is undesirable.

In comparison to these existing works our method, based on the Nystrom approximation, can be used to compress both currents and varifolds in the same framework. Furthermore, our algorithm is not an iterative greedy approach, meaning we can compress measures in a fraction of the time of the existing methods, while still retaining strong approximation guarantees, via the theoretical bounds afforded to us by the Nystrom approximation. Finally, one is not required to compute the entire target measure in our method, as opposed to the previous existing works. Due to this property, our method scales well to currents and varifolds with $> 10^5$ landmarks without significant loss of speed and maintaining good approximation guarantees.

2.2. Kernel quadrature and interpolation. We also remark here on the connection to kernel compression methods that have been developed separately, in the context of large-scale kernel learning methods. In particular, the works of [2, 4, 5, 15]. These works all focus on the problem of constructing numerical quadrature rules for integration against probability measures. Such constructions yield discrete probability measures $\hat{\mu} \in W^*$ in the dual space of an RKHS W that best approximate a given target probability measure $\mu \in \mathcal{P}(\mathcal{X})$, in the dual norm on W^* .

The main technique we use for compression of the measure-based shape representations is an orthogonal projection onto a subspace generated by a randomized data-dependent distribution. In our case, the data-dependent distribution is defined by approximated leverage scores. In particular, we bound the compression error in terms of the interpolation error between two associated dual functions in the RKHS.

The idea of interpolation/quadrature approximation of RKHS functions via orthogonal projection onto randomized sub-spaces has been explored in [4] and [5], where the quadrature nodes (or control points as in this paper) are chosen via a repulsive point distribution. In [4], this distribution is a projection Determinantal Point Process (DPP), and in [5], this distribution is a continuous volume-sampling distribution, which is shown to be a mixture of projection DPPs.

However, these approximation schemes require access to an eigen-decomposition of the kernel of the RKHS, or in the discrete case an eigen-decomposition of the full kernel matrix. One proposed way to overcome this is via MCMC methods for DPP in [5], which eliminates the need for eigen-decomposition, but is still very slow for large-scale problems.

While we obtain similar interpolation bounds in the specific case of discrete currents and varifolds, we do so in a way that does not depend on using the DPP distribution. Instead, we derive a bound in terms of trace error of the Nystrom approximation by leveraging connections to sparse interpolation results of [20]. From here, we are free to apply any theoretically sound sampling technique for the Nystrom approximation, such as Ridge Leverage Score (RLS) sampling, DPP, Greedy initialisation etc. This step allows us to use the latest developments in RLS sampling such as [11], [17] in order to form fast low dimensional approximations, that are orders of magnitude faster than K -DPP sampling, while still retaining comparable theoretical bounds.

The recent work of [15] on kernel quadrature does not have the same restrictions as [4],[5],

and has competitive run-times for compression of probability measures, with similar error bounds. Furthermore, this method shows promise for large-scale compression, as evidenced by numerical examples on empirical measures on machine-learning datasets. However, this work applies to the case of probability measures and not to general linear functionals on an RKHS. Therefore, in the case of currents and normal-cycles, which yield non-positive linear functionals on vector-valued functions, this method is not directly applicable. However, it may be interesting to compare the performance of our method against that of [15] for varifold compression, as a varifold can straightforwardly be written as a probability measure, provided the kernel is sufficiently fast-decaying e.g. Gaussian kernel.

3. Currents and Varifolds. We now briefly review the currents and varifolds representation of shape, and their applications in computational geometry. In particular their application to computing a metric distance between shapes. We present the case for smooth surfaces $S \subset \mathbb{R}^d$, with $d = 3$, but the analogous statements hold for smooth curves C with $d = 2$.

3.1. Currents. Given a smooth surface $S \subset \mathbb{R}^d$, one can define a continuous linear functional μ_S on continuous bounded vector fields $v \in C_0(\mathbb{R}^d, \mathbb{R}^d)$ as

$$(3.1) \quad \mu_S(v) := \int_S \langle v(x), n(x) \rangle dS(x),$$

where $n(x)$ denotes the outward unit normal at $x \in S$, and $dS(x)$ denotes surface measure. The action (3.1) is simply the surface integral of v along S . This functional μ_S is the *current* [19] associated to the curve S . Introducing a test Reproducing Kernel Hilbert Space of continuous bounded vector fields V with matrix-valued kernel $k : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}^{d \times d}$, one embeds μ_S into the dual space V^* due to the embedding $V \hookrightarrow C_0(\mathbb{R}^d, \mathbb{R}^d)$. A natural choice of norm on the dual space V^* is the dual norm

$$\|\mu_S\|_{V^*} = \sup_{\substack{v \in V \\ \|v\| \leq 1}} |\mu(v)|.$$

Using the reproducing property of the kernel k on V , one can prove [19] that

$$\|\mu_S\|_{V^*}^2 = \int_S \int_S k(x, y) \langle n(x), n(y) \rangle dS(x) dS(y).$$

The dual norm induces the dual metric, which is simply the metric induced by the norm. Given two elements $\mu, \kappa \in V^*$, one can compute the dual metric as

$$d_{V^*}(\mu, \kappa) = \|\mu - \kappa\|_{V^*} = \sup_{\substack{v \in V \\ \|v\| \leq 1}} |\mu(v) - \kappa(v)|.$$

Given two smooth surfaces $S_1, S_2 \subset \mathbb{R}^d$, one can therefore compute distances between the shapes in the dual metric

$$d(S_1, S_2) := d_{V^*}(\mu_{S_1}, \mu_{S_2}) = \|\mu_{S_1} - \mu_{S_2}\|_{V^*} = \sup_{\substack{v \in V \\ \|v\| \leq 1}} |\mu_{S_1}(v) - \mu_{S_2}(v)|.$$

Intuitively, this distance measures how differently the shapes integrate the same vector fields $v \in V$ along their surface, ranging over the whole space V . Therefore, the global geometry of the shapes is distinguished by test vector fields that ‘probe’ the shapes via the surface integral. Of course, the regularity and frequency of the vector fields in V determines the characteristic scale at which shapes are distinguished in such metrics. For surfaces, the dual metric has the form

$$\|\mu_{S_1} - \mu_{S_2}\|_{V^*}^2 = \|\mu_{S_1}\|_{V^*}^2 - 2\langle \mu_{S_1}, \mu_{S_2} \rangle_{V^*} + \|\mu_{S_2}\|_{V^*}^2,$$

which can be computed using the explicit formulae

$$\langle \mu_{S_1}, \mu_{S_2} \rangle_{V^*} := \int_{S_1} \int_{S_2} k(x, y) \langle n_1(x), n_2(y) \rangle dS_1(x) dS_2(y), \quad \|\mu_{S_1}\|_{V^*}^2 = \langle \mu_{S_1}, \mu_{S_1} \rangle_{V^*},$$

where the former term can be interpreted as an inner product, see [9]. Notice that computation of this metric between S_1 and S_2 does not require *any* correspondence information, only access to the surfaces and normals. This makes the currents representation ideal for metric comparison of shape, as it does not require any further learning to extract dense parametric correspondences, and gives explicit formulae for the metric that gives a measure of the difference in global geometry.

Of course in practice, one only has access to a discretization of the smooth surfaces of interest $S_1, S_2 \subset \mathbb{R}^d$. Most commonly, surfaces may be available as a triangulation $\hat{S}_1 = \{T_i^1\}_{i=1}^N$, $\hat{S}_2 = \{T_i^2\}_{i=1}^M$. In order to make geometric comparisons between the discretized shapes, we can once again embed the observed data into the dual space V^* to obtain discrete sum of Dirac functionals

$$\hat{S}_1 \hookrightarrow \hat{\mu}_{S_1} = \sum_{i=1}^N \delta_{c_i^1} \nu_i^1 \in V^*, \quad \hat{S}_2 \hookrightarrow \hat{\mu}_{S_2} = \sum_{j=1}^M \delta_{c_j^2} \nu_j^2 \in V^*,$$

where $c_i^1, \nu_i^1 \in \mathbb{R}^d$ denotes the centre and *unnormalized* normal vector respectively of the i 'th triangle T_i of \hat{S}_1 . Both the centres and normals can be computed from the vertices and edges of the discrete triangles using averages and cross products. Each weighted Dirac functional $\delta_a b \in V^*$ in the above, has the action $(\delta_a b|v) = b^T v(a)$ for all $v \in V$. This gives a simple action for the discrete functionals. For example for \hat{S}_1 ,

$$\hat{\mu}_{S_1}(v) = \sum_{i=1}^N (\nu_i^1)^T v(c_i^1).$$

It is possible to show the discrete representation is consistent with the continuous case [9] in the sense that $\|\hat{\mu}_{S_1} - \mu_{S_1}\|_{V^*} \leq C\tau(N)$ such that $\tau(N) \rightarrow 0$ as $N \rightarrow \infty$. Therefore, for a sufficiently fine discretization, we do not lose much geometric information by working with $\hat{\mu}_{S_1}$.

Once again, we can compute the dual metric between $\hat{\mu}_{S_1}, \hat{\mu}_{S_2}$ in order to make geometric comparisons without correspondences. Furthermore in the discrete case, using the reproducing

property of the RKHS, one can show [19] the dual metric has a very simple expression that can be computed *exactly*, using kernel evaluations and normal vector inner products:

(3.2)

$$d_{V^*}(\hat{\mu}_{S_1}, \hat{\mu}_{S_2})^2 = \sum_{i,j=1}^N k(c_i^1, c_j^1) \langle \nu_i^1, \nu_j^1 \rangle - 2 \sum_{i=1}^N \sum_{j=1}^M k(c_i^1, c_j^2) \langle \nu_i^1, \nu_j^2 \rangle + \sum_{i,j=1}^M k(c_i^2, c_j^2) \langle \nu_i^2, \nu_j^2 \rangle.$$

This is easy to implement, and to compute gradients of for down-line tasks such as shape registration using the currents metric as a discrepancy term.

Remark 3.1. Note that the presentation of this section can easily be extended to continuous and discrete curve data in \mathbb{R}^d for $d \in \{2, 3\}$ as well. This is done by replacing the surface integrals with line integrals, and formulating the analogous discrete formulation with discrete tangent vectors and segment centres. See [13] for the specifics.

However, in modern geometry processing tasks where discretized shapes \hat{S}_1, \hat{S}_2 are available with resolutions $N, M \geq 10^4$, one has to be careful in the method used for computing the discrete currents metric presented in equation 3.2. The computational and memory cost of the discrete metrics scales as $\mathcal{O}(NM)$, which makes the naive implementation infeasible for the large N, M regime, without specialised hardware, or complex tiling schemes such as Keops.

3.2. Varifolds. As currents rely on the orientation of the shapes to be compared (directions of tangent vectors are important) and are linear as a function of the tangent vectors, the current metric often struggles when comparing shapes with high-frequency localised variations, such as spikes arising either from anatomical features or noise. In these cases such features cancel out in the current metric and are not ‘seen’. One solution to this problem, is the varifolds representation of a shape [10], which originates in geometric measure theory.

In the varifolds representation of shape [10], shapes are treated as linear functional on a space of real-valued functions lying in a RKHS V of the form:

$$f : \mathcal{X} \rightarrow \mathbb{R}, \quad \mathcal{X} = \mathbb{R}^d \times \mathbb{S}^{d-1},$$

induced by a real-valued kernel function $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$. Typically, this is a product kernel over the two components of \mathcal{X} , of the form

$$K((x, \nu), (y, \mu)) = K_p(x, y) K_s(\nu, \mu), \quad (x, \nu), (y, \mu) \in \mathcal{X},$$

that factorizes over spatial and spherical components, as K_p and K_s respectively. This allows different regularity of functions, in both spatial and spherical components.

As in section 3.1, smooth surfaces $S \subset \mathbb{R}^d$ are embedded into the dual space V^* via surface integral action

$$\mu_S(v) = \int_S v(x, n(x)) dS(x).$$

The functional μ_S is known as the varifold associated to S . Once again, given two smooth surfaces $S_1, S_2 \subset \mathbb{R}^d$ to be compared, one can compute geometric distances between them in the dual metric

$$d(S_1, S_2) := d_{V^*}(\mu_{S_1}, \mu_{S_2}) = \|\mu_{S_1} - \mu_{S_2}\|_{V^*}.$$

A similar intuition as section 3.1 holds, that metric comparison of shapes is made by comparing how differently shapes integrate the same functions in the RKHS V along their surface. However, one must note that the test functions now vary both in space and normal components. As we shall see in the metric expression, this means the varifolds representation of shapes gives a strictly richer representation of shape.

As in section 3.1, it is possible to show closed form expressions for the dual metric when applied to varifolds associated to shapes. In particular,

$$\|\mu_{S_1} - \mu_{S_2}\|_{V^*}^2 = \|\mu_{S_1}\|_{V^*}^2 - 2\langle \mu_{S_1}, \mu_{S_2} \rangle_{V^*} + \|\mu_{S_2}\|_{V^*}^2,$$

where the individual terms are explicitly computed as

$$\langle \mu_{S_1}, \mu_{S_2} \rangle_{V^*} := \int_{S_1} \int_{S_2} K_p(x, y) K_s(n_1(x), n_2(y)) dS_1(x) dS_2(y), \quad \|\mu_{S_1}\|_{V^*}^2 = \langle \mu_{S_1}, \mu_{S_1} \rangle.$$

In the case where the spherical kernel is linear, so that $K_s(u, v) = \langle u, v \rangle$, then the above metric reduces to the currents metric between shapes. However, in general one can choose *any* nonlinear valid kernel on the sphere, such as the Gaussian kernel. In this case, the expression of the inner-product term makes clear why the varifolds representation is often preferred over currents. In particular, the varifolds metric performs nonlinear kernel comparison of normal vectors along the surface, whereas the currents metric is restricted to linear comparison. The extra nonlinearity in the normal component means there are no cancellation effects for varifolds as opposed to currents, and allows a richer metric comparison of shapes.

One can perform a similar comparison for discrete shapes, using the same approach as in section 3.1. Given triangulated shapes $\hat{S}_1 = \{T_i^1\}_{i=1}^N$, $\hat{S}_2 = \{T_i^2\}_{i=1}^M$, one defines the associated discrete varifolds in V^* as

$$\mu_{\hat{S}_1} = \sum_{i=1}^N \delta_{(c_i^1, n_i^1)} \|\nu_i^1\|, \quad \mu_{\hat{S}_2} = \sum_{i=1}^M \delta_{(c_i^2, n_i^2)} \|\nu_i^2\|,$$

where the notation for c_i^k, ν_i^k are the same as section 3.1 and n_i^k denotes *unit* normal vector of triangle T_i . As for currents, the discrete representation is consistent with the continuous representation [10], in the sense that $\|\hat{\mu}_{S_1} - \mu_{S_1}\|_{V^*} \leq C\tau(N)$ such that $\tau(N) \rightarrow 0$ as $N \rightarrow \infty$, for smooth S .

Finally, the reproducing property of the RKHS kernel yields simple closed form formula for the varifolds metric between two discrete shapes

$$\begin{aligned} d_{V^*}(\hat{\mu}_{S_1}, \hat{\mu}_{S_2})^2 &= \sum_{i,j=1}^N K_p(c_i^1, c_j^1) K_s(n_i^1, n_j^1) \|\nu_i^1\| \|\nu_j^1\| - 2 \sum_{i=1}^N \sum_{j=1}^M K_p(c_i^1, c_j^2) K_s(n_i^1, n_j^2) \|\nu_i^1\| \|\nu_j^2\| \\ &\quad + \sum_{i,j=1}^M K_p(c_i^2, c_j^2) K_s(n_i^2, n_j^2) \|\nu_i^2\| \|\nu_j^2\|. \end{aligned}$$

The analogous statements of this section for varifolds associated with curves in \mathbb{R}^d for $d \in \{2, 3\}$, also hold, see [9]. Notice once more that in the large N, M regime, naive computation of the varifold metric become infeasible both computationally and memory wise, and requires specialised routines and GPU implementations such as in the Keops library.

4. Nystrom Approximation in RKHS. We now discuss the main tool we use to derive a compression algorithm; the Nystrom approximation.

Suppose we have a dataset $\{(x_1, y_1), \dots, (x_n, y_n)\} \subset \mathcal{X} \times \mathbb{R}$ on which we perform Kernel Ridge Regression (KRR) with positive definite kernel k . Kernel Ridge Regression is the following regularized minimization scheme over the RKHS H_k of real valued functions induced by the kernel k :

$$(4.1) \quad \hat{f} = \operatorname{argmin}_{f \in H_k} \sum_{i=1}^n (y_i - f(x_i))^2 + \lambda \|f\|_{H_k}^2.$$

There exists a unique minimizer to this problem [20], and takes the form

$$\hat{f} = \sum_{i=1}^n \alpha_i k(\cdot, x_i),$$

where the coefficient vector $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_n)^T \in \mathbb{R}^n$ is given by

$$\boldsymbol{\alpha} = (K_{XX} + \lambda I_n)^{-1} \mathbf{y}, \quad \mathbf{y} = (y_1, \dots, y_n) \in \mathbb{R}^n,$$

where $(K_{XX})_{ij} = k(x_i, x_j)$ is the kernel matrix evaluated on input points $\{x_i\}_{i=1}^n$.

The *Nystrom* Kernel Ridge Regression [20] is a related problem that fits the data in a restricted subspace of the RKHS, that is parametrized by a finite set of control points $\mathbf{c} = \{c_1, \dots, c_m\} \subset \mathcal{X}$ in the domain. Defining the subspace

$$M := \left\{ \sum_{j=1}^m \alpha_j k(\cdot, c_j) : \alpha_j \in \mathbb{R} \right\} = \operatorname{span}\{k(\cdot, c_1), \dots, k(\cdot, c_m)\},$$

the Nystrom KRR problem is to find

$$\bar{f} = \operatorname{argmin}_{f \in M} \sum_{i=1}^n (y_i - f(x_i))^2 + \lambda \|f\|_{H_k}^2.$$

Once again, there exists a unique solution to this problem of the form

$$f = \sum_{i=1}^m \beta_i k(\cdot, c_i),$$

where the coefficients $\boldsymbol{\beta} = (\beta_1, \dots, \beta_m)^T \in \mathbb{R}^m$ are given by

$$\boldsymbol{\beta} = (K_{CX}K_{XC} + \lambda K_{CC})^{-1} K_{CX} \mathbf{y}.$$

The following theorem of [20] provides a bound on the error between the full solution and the m -dimensional Nystrom approximation to the KRR problem.

Theorem 4.1. *Let $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}_{\geq 0}$ be a kernel function with associated RKHS H_k . Let \hat{f} , \bar{f} be the KRR and Nystrom KRR solutions respectively, for data $\{(x_1, y_1), \dots, (x_n, y_n)\} \subset \mathcal{X} \times \mathbb{R}$, control points $\mathbf{c} = \{c_1, \dots, c_m\}$ and $\lambda > 0$ fixed. Then, the following bound holds*

$$\|\hat{f} - \bar{f}\|_{H_k}^2 \leq \frac{2 \operatorname{tr}(K_{XX} - Q_{XX}) \|y\|^2}{\lambda^2}, \quad Q_{XX} := K_{XC} K_{CC}^{-1} K_{CX}.$$

The matrix Q_{XX} in the previous theorem is the well-known Nystrom approximation [3] to the kernel matrix K_{XX} . We now review this method in the following section.

4.0.1. Nystrom Approximation for Kernel Matrices. The Nystrom approximation method can be applied to general positive-semidefinite matrices, but are most frequently used in forming low-rank approximation to kernel matrices. Suppose that we are given n points $\mathbf{x} = \{x_1, \dots, x_n\}$ in a domain \mathcal{X} , and $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ is a positive semi-definite kernel function. Defining the matrix $(K_{XX})_{ij} = (k(x_i, x_j))$, how can we best approximate the kernel matrix $K_{XX} \in \mathbb{R}^{n \times n}$ with a low-rank matrix? It is well known from finite-dimensional linear algebra, that the best rank- k approximation to the kernel matrix is given by the SVD truncation, where optimality is measured by error in Frobenius norm. However, the computation of such an approximation requires computing an eigen-decomposition which scales as $\mathcal{O}(n^3)$. For very large n as frequently occurs in machine learning, this is an obstacle.

The Nystrom method for low-rank kernel matrix approximation uses a similar idea to the Nystrom method for solving integral eigenproblems (for the associated kernel integral operator). In the original Nystrom method [21], partial evaluations of the kernel at a fixed set of control points $\mathbf{c} = \{c_1, \dots, c_m\}$ is used to solve a finite-dimensional eigenproblem. The full eigenfunctions are approximated by extending the finite-dimensional solution in an appropriate way.

Similarly, in the Nystrom approximation for kernel matrices, one evaluates the kernel at a finite set of control points $\mathbf{c} = \{c_1, \dots, c_m\}$ drawn at random from $\mathbf{x} = \{x_1, \dots, x_n\}$, and forms a *Nystrom approximation* to the full matrix as,

$$K_{XX} \approx K_{XC}K_{CC}^{-1}K_{CX} =: Q_{XX}.$$

The matrix K_{CC} , is the matrix of cross evaluations of the kernel on the set \mathbf{c} . It is possible to obtain an error bound for this approximation [3], for a suitable sampling distribution of the control points. This is the content of the following theorem of [3].

Theorem 4.2. *Suppose that m control points $\mathbf{c} = \{c_1, \dots, c_m\}$, are drawn with probability $p(\mathbf{c}) \propto \det(K_{CC})$ that is proportional to the determinant of the cross evaluation matrix. Then, the following bound holds in expectation with respect to $p(\mathbf{c})$:*

$$\mathbb{E}_{\mathbf{c}} \|K_{XX} - Q_{XX}\|_2 \leq \mathbb{E}_{\mathbf{c}} [\text{tr}(K_{XX} - Q_{XX})] \leq (m+1) \sum_{i=m+1}^n \lambda_i(K_{XX}),$$

where $\lambda_i(K_{XX})$ denotes the i 'th largest eigenvalue of the full kernel matrix.

See [3] for a proof.

This theorem demonstrates that the rate of decay of the Nystrom matrix approximation error, is at least as fast as the eigenvalue decay of the target matrix. In fact, for smooth kernel functions - for example the Gaussian RBF kernel - the eigenvalues of such kernel matrices are known to decay rapidly. Furthermore, one can make the influence of the kernel frequency/regularity clear by averaging over different inputs. For example, [6] proves the following theorem.

Theorem 4.3. *Let p a probability density function over the domain \mathcal{X} . Denote $X = \{x_1, \dots, x_n\}$ a size n i.i.d sample, from this distribution. Taking expectation over such draws*

from this distribution, gives the following bound:

$$(4.2) \quad \mathbb{E}_{\mathbf{x}}[\mathbb{E}_{\mathbf{c}}[\text{tr}(K_{XX} - Q_{XX})]] \leq (m+1)\mathbb{E}_{\mathbf{x}}\left[\sum_{i=m+1}^n \lambda_i(K_{XX})\right] \leq n(m+1) \sum_{i=m+1}^{\infty} \lambda_m$$

where the eigenvalues λ_i in the right-hand side sum, are the eigenvalues of the kernel integral operator $\mathcal{K} : L^2(\mathcal{X}, p dx) \rightarrow L^2(\mathcal{X}, p dx)$ defined as follows,

$$(\mathcal{K}g)(x') = \int_{\mathcal{X}} g(x)k(x, x')p(x)dx,$$

where one defines

$$L^2(\mathcal{X}, p dx) := \left\{ f : \mathcal{X} \rightarrow \mathbb{R} : \int_{\mathcal{X}} f(x)^2 p(x) dx < +\infty \right\},$$

upto p almost everywhere equivalence.

Therefore, on average the quality of the Nystrom matrix approximation depends on the smoothness properties and eigendecay of the kernel. For common kernels such as the Gaussian RBF, one can analytically bound the eigendecay follows

$$(4.3) \quad \sum_{i=m+1}^{\infty} \lambda_m = \mathcal{O}(m \exp(-\alpha m^{\frac{1}{d}})),$$

for some $\alpha > 0$ that depends on the pair (k, p) as shown in [6].

The larger the lengthscale, the faster the eigendecay and therefore the better the approximation for small m . For kernels with smaller lengthscales and therefore slower eigendecay, a higher m will be required to make the average error smaller.

While sampling from the m -DPP gives a convergence bound, sampling naively from this distribution is computationally intractable for large n and moderate m . While there exists more sophisticated algorithms for sampling exactly from such distributions, they require computing an eigen-decomposition of the kernel matrix with cost $\mathcal{O}(n^3)$ which is also computationally infeasible. One way to overcome this issue while retaining the theoretical guarantees, is via MCMC sampling to sample from a k -DPP. The sampling scheme is given in [Appendix A.2](#).

4.1. Ridge Leverage Score (RLS) Sampling for Nystrom approximation. An alternate method to ensure good quality of Nystrom approximation is Ridge Leverage Score (RLS) sampling technique of control points. If one fixes a level of approximation λ , this technique samples each point $x_i \in X = \{x_1, \dots, x_n\}$ with probability

$$l_i(\lambda)(K) = (K(K + \lambda I)^{-1})_{i,i},$$

known as the ridge leverage score of point x_i , and constructs the Nystrom approximation on the sampled points $\mathbf{c} = \{c_1, \dots, c_m\}$. It can be shown this procedure gives the error bound

$$\|K - \tilde{K}\|_2 \leq \lambda, \quad \tilde{K} = K_{Xc}K_{cc}^{-1}K_{cX},$$

with probability $1 - \delta$, for randomized sample size satisfying $m = \mathcal{O}(d^\lambda \log(d^\lambda/\delta))$, where $d^\lambda := \text{tr}(K(K + \lambda I)^{-1})$. Note that we also have that the size m is randomized as a function of λ the error. If one sets $\lambda = \frac{1}{k} \sum_{k+1}^n \lambda_i(K)$, it may be shown [17] that $d^\lambda = \mathcal{O}(k)$, so that

$$(4.4) \quad \left\| K - \tilde{K} \right\|_2 \leq \frac{1}{k} \sum_{k+1}^n \lambda_i(K)$$

with probability $1 - \delta$ and randomized sample size $m = \mathcal{O}(k \log(k/\delta))$. Alternatively, in practice one may sample exactly $m = \lceil (k \log(k/\delta)) \rceil$ samples directly from the leverage score distribution without replacement, so that the above bounds hold approximately. This is often preferable in practice as the constants involved in the upper bound are large, and may result in oversampling relative to a fixed approximation level.

Of course, computing directly the RLS scores $\{l_i\}_{i=1}^n$ is computationally infeasible due to the prohibitively large kernel matrices and inverses involved. As a result, there are a wide array of algorithms that have been developed in order to approximately sample from this distribution while still maintaining the error bounds up to a factor. Such algorithms require significantly less memory and computational cost, making them the sampling algorithm of choice in practical situations for the Nystrom approximation. In this work, we consider two such sampling methods; the recursive method of [17] and more recently the divide and conquer method of [11].

For example, the sampling algorithm given as Theorem 3 of [17], gives the following theoretical guarantee:

Theorem 4.4. *Fix $\delta \in (0, \frac{1}{32})$, and $S \in \mathbb{N}$. There exists a constant $c > 0$ such that the Recursive RLS sampling algorithm of [17], with probability $1 - 3\delta$ returns $m \leq cS \log(S/\delta)$ such that*

$$(4.5) \quad \left\| K - \tilde{K} \right\|_2 \leq \frac{1}{S} \sum_{i=S+1}^n \lambda_i(K).$$

If one wants to construct an approximation with number of control points approximately m , it suffices to choose S such that $m \approx S \log(S/\delta)$, which would give the error bound on the right hand side with high probability. Sampling using this algorithm is significantly faster than using MCMC samplers, and in practice gives good quality samples that ensure the Nystrom approximation remains small with strong theoretical guarantee. In practice, one may sample exactly $m = \lceil (S \log(S/\delta)) \rceil$ samples directly from the approximate leverage score distribution without replacement, so that the above bounds hold approximately. This is implemented in ¹ made available by the authors of [17]. Again, we note that the constants in the upper bound of m are large, and in practice suffices to take $m \approx (S \log(S/\delta))$.

Another efficient RLS sampler is the Divide and Conquer (DAC) RLS approximation, which is algorithm 2 of [11]. This algorithm also yields the same bound (4.4) but with probability $1 - \delta$ and $m \leq c(S + n(1 - \alpha)) \log(\frac{S+n(1-\alpha)}{\delta})$, where $\alpha \in (0, 1)$. The theory bound on m here is looser than [17] with the additive factor of $n(1 - \alpha)$. While the theoretical guarantees of [11]

¹<https://github.com/cnmusco/recursive-nystrom>

algorithm are not quite as tight as that of [17], one observes that sampling $m \approx S \log(S/\delta)$ from the DAC leverage scores yields Nystrom approximations of similiar error to the method of [17], while being significantly faster in practice, especially in the case of $n \geq 10^5$. We also observe this in practice in section 7. Furthermore, it is very easy to parallelise the DAC algorithm, in contrast to the Recursive RLS scheme, thus gaining a further speedup. Therefore, even though we prove bounds in section 5 for the method of [17], the DAC method of [11] is usually our algorithm of choice for sampling control points for compression of very large currents and varifolds.

5. Main Results. We now propose a method to compress discrete linear functionals of the form

$$\mu = \sum_{i=1}^n \delta_{x_i} \alpha_i \in V^*, \quad (x_i, \alpha_i) \in \mathcal{X} \times \mathbb{R}^d,$$

where we take \mathcal{X} to be a non-empty set, and assume a fixed RKHS $V \subset C(\mathcal{X}, \mathbb{R}^d)$ induced by a scalar diagonal kernel $K(x, y) = k(x, y)I_d$. This choice of scalar diagonal kernel is by far the most popular in practical applications of currents and varifolds. In this section, we derive a method for a general $d \in \mathbb{N}$, and we shall apply this result to the special case of currents and varifolds in subsequent sections. The main techniques we employ are the Nystrom approximation and interpolation theory in RKHS, which allows us to obtain fast decaying error bounds for the compression, while also giving a computationally cheaper approximation method.

The compressed approximation to the full functional will take the form

$$(5.1) \quad \hat{\mu} = \sum_{i=1}^m \delta_{c_i} \alpha_i^c, \quad (c_i, \alpha_i^c) \in \mathcal{X} \times \mathbb{R}^d,$$

for appropriately chosen c_i, α_i^c and $m \ll n$ such that the error is bounded above by a small constant:

$$(5.2) \quad \|\mu_S - \hat{\mu}_S\|_{V^*} < \epsilon.$$

Note that we wish to approximate in the discrete delta form (5.1), as it will allow us to use the exact dual metric computation formula (3.2) on the compressed form, with $m \ll n$ terms. This reduces dual metric computation cost, from $\mathcal{O}(n^2)$ to $\mathcal{O}(m^2)$, which gives significant saving when $m \ll n$.

Henceforth, we denote by $L : V \rightarrow V^*$ the isometric duality mapping of the RKHS V . By the reproducing property of the kernel functions, it is a standard fact that the following equality holds:

$$(5.3) \quad L^{-1}(\delta_c w) = k(\cdot, c)w \in V, \quad (c, w) \in \mathcal{X} \times \mathbb{R}^d,$$

which is the dual vector field in V to the linear functional $\delta_c w$. Using the Riesz representation theorem and the explicit form 5.3 of the isometric duality map in Reproducing Kernel Hilbert

Spaces, the proposed approximation of discrete functionals is equivalent to approximating functions in V of the form:

$$v^\alpha(x) := \sum_{i=1}^n k(x, x_i) \alpha_i = L^{-1} \left(\sum_{i=1}^n \delta_{x_i} \alpha_i \right),$$

with compact approximations of the form,

$$v_m(x) := \sum_{i=1}^m k(x, c_i) \alpha_i^c = L^{-1} \left(\sum_{i=1}^m \delta_{c_i} \alpha_i^c \right),$$

with size $m \ll n$.

5.1. Algorithms and Theoretical guarantees. We summarise the resulting algorithm for compressing discrete functionals in RKHS in the following.

Algorithm 5.1 Discrete functional compression

- 1: Fix \mathcal{X} , $m \ll n$, RKHS kernel function $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$, a sampling algorithm for control points and target functional

$$\mu = \sum_{i=1}^n \delta_{x_i} \alpha_i, \quad (x_i, \alpha_i) \in \mathcal{X} \times \mathbb{R}^d.$$

- 2: Sample m control points $\{c_1, \dots, c_m\} \subset \{x_1, \dots, x_n\}$ from the chosen sampling algorithm for control points.
- 3: Form the dual vector valued function on the control points as

$$y_j = \sum_{i=1}^n k(c_j, x_i) \alpha_i, \quad j = 1, \dots, m.$$

- 4: Form the control point approximation via orthogonal projection

$$\hat{\mu} = \sum_{i=1}^m \delta_{c_i} \beta_i, \quad \beta = K_{CC}^{-1} y \in \mathbb{R}^{m \times d}.$$

Depending on the algorithm used for sampling control points, this algorithm yields different theoretical guarantees which we now describe.

Theorem 5.1. *Suppose we have a discrete target functional of the form*

$$\mu_S = \sum_{i=1}^n \delta_{x_i} \alpha_i \in V^*, \quad (x_i, \alpha_i) \in \mathcal{X} \times \mathbb{R}^d$$

with associated dual vector-valued function

$$v^\alpha(x) = \sum_{i=1}^n k(x, x_i) \alpha_i \in V.$$

Sample m control points $\mathbf{c} = \{c_1, \dots, c_m\} \subset \{x_1, \dots, x_n\}$ and define the matrix of values

$$Y_c = (v^\alpha(c_1), \dots, v^\alpha(c_m))^T \in \mathbb{R}^{m \times d},$$

which is the evaluation of the dual vector-valued function on the control point locations. Computing weights

$$\beta = [\beta_1, \dots, \beta_m]^T = K_{cc}^{-1} Y_c \in \mathbb{R}^{m \times d},$$

yields an approximation

$$\sum_{i=1}^m \delta_{c_i} \beta_i \approx \sum_{i=1}^n \delta_{x_i} \hat{\alpha}_i,$$

that satisfies

$$\left\| \sum_{i=1}^n \delta_{x_i} \hat{\alpha}_i - \sum_{i=1}^m \delta_{c_i} \beta_i \right\|_{V^*}^2 \leq C \text{tr}(K_{XX} - Q_{XX}).$$

1. Sampling control points from an m -DPP on the kernel matrix K_{XX} , yields the following in-expectation error bounds

$$\mathbb{E}_{\mathbf{c}} \left\| \sum_{i=1}^n \delta_{x_i} \hat{\alpha}_i - \sum_{i=1}^m \delta_{c_i} \beta_i \right\|_{V^*}^2 \leq C(m+1) \sum_{i=m+1}^n \lambda_i(K_{XX}).$$

2. By sampling control points using the Recursive RLS scheme instead, we obtain similar theoretical guarantees (upto a factor). Fixing $\delta \in (0, \frac{1}{32})$, $S \in \mathbb{N}$, with probability $1 - 3\delta$, we have $m \leq cS \log(S/\delta)$ and

$$\left\| \sum_{i=1}^n \delta_{x_i} \hat{\alpha}_i - \sum_{i=1}^m \delta_{c_i} \beta_i \right\|_{V^*}^2 \leq \frac{Cn}{S} \sum_{i=S+1}^n \lambda_i(K_{XX}).$$

We observe the randomized projection scheme yields a strong error-decay bound, at least as fast as the rate of decay of eigenvalues of the kernel matrix. We also remark here that almost identical bounds to those proven for the Recursive RLS scheme, also holds when sampling control points with the DAC RLS sampler of [11], with slightly looser bounds on m .

We may also relate the decay of error to the decay of eigenvalues of the kernel integral operator \mathcal{K}_p itself, where p denotes a sampling distribution for the delta centres $x_i \in \mathcal{X}$. This is summarised in the following corollary,

Corollary 5.2. *Suppose that the Dirac delta centres $\{x_i\}_{i=1}^n \in \mathcal{X}$ of the target $\mu \in W^*$ from a continuous distribution with density p . Then, the rate of decay of the of the m -DPP and RLS approximations in [Theorem 5.1](#) are bounded in expectation and with high probability respectively by,*

$$f(m) = \sum_{i=m+1}^{\infty} \lambda_i$$

In the special case where the input distribution is a Gaussian mixture and RKHS kernel is Gaussian on \mathbb{R}^d , this yields the following exponential decay rate

$$f(m) = \mathcal{O}(m \exp(-\alpha m^{\frac{1}{d}})), \quad \alpha > 0,$$

where α is a distribution dependent constant.

The Gaussian mixture distribution for delta centres in [Corollary 5.2](#), encompasses the case where shape data is sampled noisily from a continuous distribution centred on an underlying continuous shape. We now discuss the application of [Algorithm 5.1](#) and [Theorem 5.1](#) to compression of the currents and varifolds representations of shape.

5.2. Application to Compression of Currents and Varifolds. Given a discretized shape $\hat{S}_1 \subset \mathbb{R}^d$, we know that one can construct discrete currents and varifolds of the form

$$\mu_S = \sum_{i=1}^n \delta_{x_i} \alpha_i \in V^*, \quad (x, \alpha_i) \in \mathcal{X} \times \mathbb{R}^d,$$

in order to perform down-line tasks requiring metric shape comparison, without correspondences. As before, V^* is an RKHS induced by scalar diagonal kernel $K(x, y) = k(x, y)I_d$, with $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}_{\geq 0}$. We can apply the approximation methods of the previous section, in order to compress a given current or varifold in a sparse basis. It simply remains to explicit the forms of the underlying space \mathcal{X} , and the points and weights x_i, α_i for each representation.

1. For **currents**, we have that $\mathcal{X} = \mathbb{R}^d$ and for $i = 1, \dots, n_T$,

$$x_i := \frac{1}{3}(v_{i1} + v_{i2} + v_{i3}), \quad \alpha_i := \frac{1}{2}(v_{i3} - v_{i2}) \times (v_{i2} - v_{i1}),$$

so x_i, α_i are the i 'th triangle centre and (un-normalized) normal vector respectively.

2. For **varifolds** the base space is changed to $\mathcal{X} = \mathbb{R}^d \times \mathbb{S}^{d-1}$ and for $i = 1, \dots, n_T$,

$$x_i := \left(\frac{1}{3}(v_{i1} + v_{i2} + v_{i3}), \frac{\nu_i}{\|\nu_i\|} \right), \quad \alpha_i := \|\nu_i\|, \quad \nu_i := \frac{1}{2}(v_{i3} - v_{i2}) \times (v_{i2} - v_{i1})$$

where as before, x_i, α_i are the i 'th triangle centre and (un-normalized) normal vector respectively.

With these choices, applying the compression algorithm of the previous section, will allow us to compress currents and varifolds within the same framework, while retaining strong theoretical error bounds. Indeed, [Theorem 5.1](#), [Corollary 5.2](#) imply that for the common gaussian kernel setting, one can obtain compression of currents and varifolds associated to noisy shape samples with exponentially decaying compression error.

5.3. Choice of m . In the algorithms we have presented so far, the choice of the compression level m is a crucial one. In theory, one can set the eigenvalue bounds presented in lemma [5.1](#), to a pre-defined tolerance and choose m accordingly. However, in practice the cost of evaluating the upper bound is $\mathcal{O}(n^3)$ due to the eigen-decomposition cost, which is undesirable. Choice of m is also an issue that is unresolved in the works of [\[12\]](#) and [\[16\]](#), due to expensive to compute upper bound, and lack of theoretical error guarantees respectively.

As a cheaper practical alternative, we propose to use the preliminary trace bound of corollary 6.2 instead for this task. In particular, one can use the trace error of the Nystrom approximation as an indicator for the true compression error given m control points. For shift invariant kernels, computation of the trace term simplifies as,

$$\text{tr}(K_{XX} - Q_{XX}) = n - \text{tr}(K_{XC}K_{CC}^{-1}K_{CX}),$$

which has a computational cost of at most $m^3 + nm^2$, which is significantly less than n^3 of the eigen-decomposition.

After computing approximate leverage scores, one can choose m by setting a pre-defined tolerance $\epsilon > 0$ and increasing the number of samples m , until the change in the trace bound is below this tolerance. This is a sensible choice, as we observe in practice (see section 7) that the decay of the true compression error closely matches that of the trace bound, which makes it a suitable proxy for the error. This is given as algorithm 5.2. Note that this procedure, is similar to the use of trace bound for judging quality of inducing points in sparse Gaussian process methods [6].

Algorithm 5.2 Trace bound heuristic for choosing m

- 1: Fix error tolerance $\tau > 0$, $e := +\infty$ and $m = 0$.
 - 2: **while** $e > \tau$ **do**
 - 3: $m \leftarrow m + 1$
 - 4: Sample m control points $c = \{c_1, \dots, c_m\}$.
 - 5: Set $e = n - \text{tr}(K_{Xc}K_{cc}^{-1}K_{cX})$
 - 6: **end while**
-

5.4. Application to LDDMM matching. Thus far, we have derived a fast compression algorithm for large-scale currents and varifolds with theoretical guarantees on the quality of the approximation error. We now describe how the compressed measures can be used for the down-line task of nonlinear shape registration in the LDDMM framework, one of the areas in which these metrics are most commonly used.

A typical workflow in the the application of LDDMM to computational anatomy is to register two given shapes $T, S \subset \mathbb{R}^d$ via a minimal energy diffeomorphism φ . Such diffeomorphisms are typically induced as the flow of time dependent vector fields $v : [0, T] \times \Omega \rightarrow \mathbb{R}^d$ such that for each t , $v(t, \cdot) \in V$ where V is an RKHS of vector fields. In particular, one registers T to S in this framework, by minimizing an objective $E : L^2([0, T], V) \rightarrow \mathbb{R}_{\geq 0}$ of the form

$$(5.4) \quad E(v) = \int_0^T \|v\|_t^2 dt + \lambda \|\varphi_{0T}^v \cdot T - S\|_{W^*}^2, \quad \lambda > 0,$$

where φ_{0T}^v denotes the time T flow associated to v and W is the RKHS associated to the current/varifold used for shape comparison. The objective is a sum of two terms, the first term measuring the energy of the diffeomorphism induced by v , and the second a weighted correspondence-less matching term encouraging good quality of the registration as measured in $\|\cdot\|_{W^*}$. There are many efficient methods developed to optimize these objectives, using special properties of the solutions such as momentum conservation. However, for large-scale shapes

the discrepancy term is a challenge to compute and pass gradients through with cost scaling as $\mathcal{O}(|T||S|)$. As previously mentioned, the KeOps framework has been recently developed to deal with such issues. However, this method starts to become slow once the resolution of T, S exceeds 10^4 and requires a GPU with sufficient memory. This becomes especially noticeable in the group-wise setting when many shapes $\{S_1, \dots, S_n\}$ are being registered to T as part of learning a statistical model for shapes, and one is required to compute the metric multiple times per-iteration.

We now describe how to use the compression to reduce the cost of registration drastically, and scale up to extremely over-sampled shapes. Henceforth, we focus on the case of a single registration. At each iteration of the optimization procedure for the objective (5.4), one is required to compute the discrepancy term. Since S does not change from iteration to iteration, one can compress μ_S as a form of pre-processing for the metric computation. There are two options at this stage. The first, is to compute the compression of $\mu_{\varphi(T)}$ each iteration and make the metric comparison on the compressed measures. The second, is to compress the template itself and compute a push-forward of the compressed template with which we compute the metric against the compressed target. This push-forward acts naturally on geometric measures, with the attractive property $\varphi_{\#}(\mu_T) = \mu_{\varphi(T)}$. There are trade-offs in either case. In the former, **at each iteration** we have to compute the leverage score approximation and the orthogonal projection, both of which have a cost scaling as $\mathcal{O}(nm^2)$. In the latter, one only has to compute the compression **once** and only push forward m points on the template. However, computing push-forward of a general $\mu \in W^*$ requires computing the action of the Jacobian of φ , which requires solving another set of ODEs in parallel, which can be very expensive depending on the problem size.

Instead, we propose a more efficient variant that combines the best of both approaches. First, one compresses the initial template measure μ_T to obtain $\hat{\mu}_T$, supported on a set of control points $\mathbf{c} \subset \mathcal{X}$. Next, instead of computing the measure push-forward $\varphi_{\#}\mu_T$ explicitly, one can deform the full template T as in standard LDDMM, and subsequently perform the orthogonal projection of $\mu_{\varphi(T)}$ on to the subspace generated by $\varphi(\mathbf{c}(T))$ the transported control points at time T . This is a reasonable approximation to make in practice, as the push-forward of discrete currents/varifolds are discrete currents/varifolds centred at the transported control points, as shown in [12], [16] for currents and varifolds respectively. Note that in this variation, one is only required to perform the RLS approximation once, in order to compress the initial template. After this, each iteration only requires one additional orthogonal projection on the deformed template. The proposed compressed matching algorithm is given below, with $T = 1$.

Algorithm 5.3 Compressed LDDMM iteration

- 1: Fix $\epsilon > 0$, $r := +\infty$, template T , vector field $v_\theta(t, x)$ and compressed target measure $\hat{\mu}_S$.
- 2: Compress μ_T with RLS sampling, and control points $\mathbf{c} = \{c_1, \dots, c_m\}$.
- 3: **while** $r > \epsilon$ **do**
- 4: Push forward template under current estimated diffeomorphism,

$$\varphi_\theta := \varphi_{01}^{v_\theta}.$$

- 5: Project $\mu_{\varphi(T)}$ onto subspace spanned by $\mathbf{c}(T)$, to obtain $\mathcal{P}_{\mathbf{c}}(\mu_{\varphi(T)})$.
- 6: Compute compressed objective,

$$(5.5) \quad E(v_\theta) = \int_0^1 \|v_\theta(t, \cdot)\|^2 dt + \lambda \|\hat{\mu}_S - \mathcal{P}_{\mathbf{c}}(\mu_{\varphi(T)})\|_{W^*}^2$$

at significantly reduced cost.

- 7: Compute gradients of objective (5.5) with respect to parameters θ of diffeomorphism.
- 8: Update θ , via gradient based optimizer.
- 9: **end while**

With this modification, at each iteration we only compute a single orthogonal projection of the deformed template at cost $\mathcal{O}(|T|m^2)$, in addition to the metric computation between compressed measures which is now significantly cheaper at $\mathcal{O}(mn)$.

6. Proofs of Main Results. We present in this section the proof of Theorem 5.1. We begin with a sparse control point approximation of discrete functionals using the Nystrom approximation in RKHS.

Lemma 6.1. *Given $v^{\hat{\alpha}}$ of the form*

$$v^\alpha(x) = \sum_{i=1}^n k(x, x_i) \alpha_i, \quad (x_i, \alpha_i) \in \mathcal{X} \times \mathbb{R}^d$$

m distinct control points $\mathbf{c} = \{c_1, \dots, c_m\} \subset \{x_1, \dots, x_n\}$, and $\mu > 0$, there is a control point field of the form:

$$v^{\mathbf{c}, \mu} = \sum_{i=1}^m \beta_i^{\mathbf{c}, \mu} k(\cdot, c_i), \quad \beta_i^{\mathbf{c}, \mu} \in \mathbb{R}^d,$$

and a constant C_1 such that the following holds

$$(6.1) \quad \left\| v^{\hat{\alpha}} - v^{\mathbf{c}, \mu} \right\|_V^2 \leq C_1 \text{tr}(K_{XX} - Q_{XX}), \quad Q_{XX} := K_{XC} K_{CC}^{-1} K_{CX}.$$

Proof.

The idea is to rewrite each component $v_j^{\hat{\alpha}}$ as a solution to a KRR problem of the form 4.1, and form its Nystrom approximation in the control point subspace.

Recall that given a field of the form $v^{\hat{\alpha}}$, the j 'th component can be written

$$v_j^{\hat{\alpha}}(x) = \sum_{i=1}^n k(x, x_i) \hat{\alpha}_{ij}, \quad \hat{\alpha}_j = (\hat{\alpha}_{1j}, \dots, \hat{\alpha}_{nj}), \quad \hat{\alpha}_j = K_{XX}^{-1} y_j.$$

In order to apply [Theorem 4.1](#), we rewrite for arbitrary $\mu > 0$

$$\hat{\alpha}_j = (K_{XX} + \mu I_n)^{-1} (K_{XX} + \mu I_n) \hat{\alpha}_j = (K_{XX} + \mu I_n)^{-1} [(K_{XX} + \mu I_n) K_{XX}^{-1} y_j],$$

so that $v_j^{\hat{\alpha}}$ is written as a kernel ridge regression solution with data

$$\tilde{y}_j = (K_{XX} + \mu I_n) K_{XX}^{-1} y_j.$$

One can now form the Nystrom KRR solution approximating $v_j^{\hat{\alpha}}$ as:

$$v_j^{c,\mu}(x) = \sum_{i=1}^m \beta_{ij} k(x, c_i), \quad \beta_j = (K_{CX} K_{XC} + \mu K_{CC})^{-1} K_{CX} \tilde{y}_j,$$

which gives the resulting trace bound by [Theorem 4.1](#)

$$(6.2) \quad \left\| v_j^{\hat{\alpha}} - v_j^{c,\mu} \right\|_{V_k}^2 \leq \frac{2 \text{tr}(K_{XX} - Q_{XX}) \|\tilde{y}_j\|^2}{\mu^2}.$$

By applying this bound component-wise, and applying the norm splitting lemma [A.1](#), this gives us the existence of a function of the form

$$v^{c,\mu}(x) = \sum_{i=1}^m \beta_i k(x, c_i), \quad \beta_i = (\beta_{i1}, \dots, \beta_{id}), \quad v^{c,\mu} = (v_1^{c,\mu}, \dots, v_d^{c,\mu}),$$

such that

$$\|v^\alpha - v^{c,\mu}\|_V^2 = \sum_{l=1}^d \left\| v_l^{\hat{\alpha}} - v_l^{c,\mu} \right\|_{V_k}^2 \leq C \text{tr}(K_{XX} - Q_{XX}), \quad C = \frac{2d}{\mu^2} \|\tilde{y}\|^2,$$

so the error is controlled by trace error of the Nystrom approximation. ■

Applying the isometric duality mapping and the above result, this gives the following corollary

Corollary 6.2. *Suppose we have a discrete functional of the form*

$$\sum_{i=1}^n \delta_{x_i} \alpha_i \in V^*, \quad (x_i, \alpha_i) \in \mathcal{X} \times \mathbb{R}^d,$$

which we wish to compress. There exists a discrete functional of the form

$$\sum_{i=1}^m \delta_{c_i} \alpha_i^{c,\mu} \in V^*,$$

with $m \ll n$, such that the following bounds hold:

$$\left\| \sum_{i=1}^n \delta_{x_i} \hat{\alpha}_i - \sum_{i=1}^m \delta_{c_i} \alpha_i^{c,\mu} \right\|_{V^*}^2 \leq C \text{tr}(K_{XX} - Q_{XX}).$$

Proof. By the isometric property of the duality mapping L , we have

$$\begin{aligned} \|v^\alpha - v^{c,\mu}\|_V^2 &= \|L(v^\alpha - v^{c,\mu})\|_{V^*}^2 = \|L(v^\alpha) - L(v^{c,\mu})\|_{V^*}^2 \\ &= \left\| L\left(\sum_{i=1}^n k(x, x_i)\alpha_i\right) - L\left(\sum_{i=1}^m \alpha_i^{c,\mu} k(\cdot, c_i)\right) \right\|_{V^*}^2 \\ &= \left\| \sum_{i=1}^n L(k(x, x_i)\alpha_i) - \sum_{i=1}^m L(\alpha_i^{c,\mu} k(\cdot, c_i)) \right\|_{V^*}^2 \\ &= \left\| \sum_{i=1}^n \delta_{x_i} \hat{\alpha}_i - \sum_{i=1}^m \delta_{c_i} \alpha_i^{c,\mu} \right\|_{V^*}^2. \end{aligned}$$

Applying [Lemma 6.1](#), yields

$$\left\| \sum_{i=1}^n \delta_{x_i} \hat{\alpha}_i - \sum_{i=1}^m \delta_{c_i} \alpha_i^{c,\mu} \right\|_{V^*}^2 \leq C \text{tr}(K_{XX} - Q_{XX}). \quad \blacksquare$$

So far, we have approximated a given discrete input functional sparsely, based on a subset of control points chosen from the original basis. We have shown the error of this approximation is bounded by the trace error of the Nystrom approximation to the kernel matrix K , based on the chosen control points. Of course the goodness of this approximation is dependent on a good choice of control point locations. The following corollary shows, that it suffices to randomly sample control points from a data dependent input distribution, in order to give strong theoretical error bounds on the approximation procedure.

Corollary 6.3. *Suppose we sample control points $\mathbf{c} = \{c_1, \dots, c_m\}$ from a m -DPP as in [theorem 4.2](#). Then, the approximation of [Corollary 6.2](#) satisfies the following in-expectation bound with respect to this distribution:*

$$\mathbb{E}_{\mathbf{c}} \left\| \sum_{i=1}^n \delta_{x_i} \hat{\alpha}_i - \sum_{i=1}^m \delta_{c_i} \alpha_i^{c,\mu} \right\|_{V^*}^2 \leq C(m+1) \sum_{i=m+1}^n \lambda_i(K_{XX}),$$

where $\lambda_i(K_{XX})$ are eigenvalues of the kernel-matrix evaluated on the discrete input points x_i .

Proof. Taking expectation with respect to an m -DPP distribution of control points, one obtains

$$\mathbb{E}_{\mathbf{c}} \left\| \sum_{i=1}^n \delta_{x_i} \hat{\alpha}_i - \sum_{i=1}^m \delta_{c_i} \alpha_i^{c,\mu} \right\|_{V^*}^2 \leq C \mathbb{E}_{\mathbf{c}}[\text{tr}(K_{XX} - Q_{XX})].$$

Applying [lemma 4.2](#) yields,

$$\mathbb{E}_{\mathbf{c}} \left\| \sum_{i=1}^n \delta_{x_i} \hat{\alpha}_i - \sum_{i=1}^m \delta_{c_i} \alpha_i^{c,\mu} \right\|_{V^*}^2 \leq C \mathbb{E}_{\mathbf{c}}[\text{tr}(K_{XX} - Q_{XX})] \leq C(m+1) \sum_{i=m+1}^n \lambda_i(K_{XX}). \quad \blacksquare$$

In summary, in order to form this approximation one:

1. First samples control points $\mathbf{c} = \{c_1, \dots, c_m\}$ from an m -DPP distribution.
2. For each $j = 1, \dots, d$ compute

$$\tilde{y}_j = (K_{XX} + \mu I_n) K_{XX}^{-1} y_j = (K_{XX} + \mu I_n) \hat{\alpha}_j,$$

and subsequently the weights,

$$\beta^j = (K_{CX} K_{XC} + \mu K_{CC})^{-1} K_{CX} \tilde{y}_j, \quad \beta_j \in \mathbb{R}^m.$$

3. Form the approximation

$$\sum_{i=1}^m \delta_{c_i} \alpha_i^{c, \mu} \approx \sum_{i=1}^n \delta_{x_i} \hat{\alpha}_i \quad (\alpha_i^{c, \mu})_j = \beta_i^j, \quad j = 1, \dots, d.$$

Computing the weights β for the approximation however can be expensive, as it requires the one off cost of computation of a large-scale kernel matrix vector product $K_{XX} \hat{\alpha}_j$ with cost $\mathcal{O}(n^2)$, which is comparable to the original metric computation cost for currents and varifolds. For compressing currents as a form of pre-processing for down-line tasks such as matching and computing statistics, this is fine, as one can treat it as a one-off cost, before performing the task of interest. However, this can be tedious if one wishes to compress many currents or varifolds, or perform multiple compressions at different scales.

One can make this compression algorithm more practical, while retaining the theoretical guarantees by using the orthogonal projection onto the random subspace. This will be significantly cheaper to compute for a given target functional compared to the Nystrom approximation weights, while still having the same error bound in expectation. One may further make the approximation cheaper, by using RLS sampling schemes for choosing control point positions. This is the content of [Theorem 5.1](#) which we now prove.

Proof of Theorem 5.1. We denote by \mathcal{P}_c the orthogonal projection into the subspace

$$M(\mathbf{c}) = \text{span}\{k(\cdot, c_i) e_j : e_j \in \mathbb{R}^d, \quad i = 1, \dots, m \quad j = 1, \dots, d\},$$

spanned by the m control points. We denote the orthogonal projection of $v^{\hat{\alpha}}$ onto $M(\mathbf{c})$ as

$$\mathcal{P}_c(v^{\hat{\alpha}}) = \mathcal{P}_c\left(\sum_{i=1}^n k(x, x_i) \alpha_i\right) = \sum_{i=1}^m k(x, c_i) \beta_i.$$

By the orthogonal projection conditions applied to the RKHS V , one can show the explicit relation

$$\beta = [\beta_1, \dots, \beta_m]^T = K_{CC}^{-1} Y_C,$$

for the projection weights [\[20\]](#). Here we recall the notation,

$$Y_C := (v^\alpha(c_1), \dots, v^\alpha(c_m))^T \in \mathbb{R}^{m \times d},$$

a matrix of values, which is the evaluation of the dual vector-valued function on the control point locations.

Indeed, the orthogonal projection conditions yield

$$\langle v^{\hat{\alpha}} - \mathcal{P}_c(v^{\hat{\alpha}}), k(\cdot, c_k)e_j \rangle_V = 0, \quad k = 1, \dots, m, \quad j = 1, \dots, d.$$

Expanding this out gives

$$\begin{aligned} \left\langle \sum_{i=1}^n k(x, x_i)\alpha_i - \sum_{i=1}^m k(x, c_i)\beta_i, k(\cdot, c_k)e_j \right\rangle_V &= \sum_{i=1}^n k(c_k, x_i)\alpha_i^T e_j - \sum_{i=1}^m k(c_k, c_i)\beta_i^T e_j \\ &= \left(\sum_{i=1}^n k(c_k, x_i)\alpha_i^T \right) e_j - \left(\sum_{i=1}^m k(c_k, c_i)\beta_i^T \right) e_j \\ &= Y_k^T e_j - \left(\sum_{i=1}^m k(c_k, c_i)\beta_i^T \right) e_j = 0, \end{aligned}$$

so that

$$Y_{kj} = \left(\sum_{i=1}^m k(c_k, c_i)\beta_{ij} \right) \implies Y_C = K_{CC}\beta \implies \beta = K_{CC}^{-1}Y_C.$$

We now prove the DPP sampling bound of [Theorem 5.1](#). Using the optimality of the orthogonal projection in inner product spaces and [theorem 6.1](#), we have the following bound

$$\begin{aligned} \left\| \sum_{i=1}^n k(x, x_i)\alpha_i - \sum_{i=1}^m k(x, c_i)\beta_i \right\|_V^2 &= \left\| \sum_{i=1}^n k(x, x_i)\alpha_i - \mathcal{P}_c \left(\sum_{i=1}^n k(x, x_i)\alpha_i \right) \right\|_V^2 \\ &\leq \left\| v^{\hat{\alpha}} - v^{c, \mu} \right\|_V^2 \leq C \text{tr}(K_{XX} - Q_{XX}), \end{aligned}$$

and therefore by taking expectations and using [Theorem 4.2](#) gives,

$$\mathbb{E}_c \left\| \sum_{i=1}^n k(x, x_i)\alpha_i - \sum_{i=1}^m k(x, c_i)\beta_i \right\|_V^2 \leq C \mathbb{E}_c \text{tr}(K_{XX} - Q_{XX}) \leq C(m+1) \sum_{i=m+1}^n \lambda_i(K_{XX}).$$

Applying the isometry property of the Riesz map ones again yields,

$$\mathbb{E}_c \left\| \sum_{i=1}^n \delta_{x_i} \hat{\alpha}_i - \sum_{i=1}^m \delta_{c_i} \beta_i \right\|_{V^*}^2 = \mathbb{E}_c \left\| \sum_{i=1}^n k(x, x_i)\alpha_i - \sum_{i=1}^m k(x, c_i)\beta_i \right\|_V^2 \leq C(m+1) \sum_{i=m+1}^n \lambda_i(K_{XX}). \blacksquare$$

We now prove the RLS sampling bound. For this proof, we note that the following holds [\[20\]](#) for symmetric positive-definite matrices $A \in \mathbb{R}^{n \times n}$:

$$(6.3) \quad \text{tr}(A) \leq n \|A\|_2,$$

where the norm on the right hand side is the spectral norm.

Suppose we fix $\delta \in (0, \frac{1}{32})$, $S \in \mathbb{N}$ and sample m control points $\mathbf{c} = \{c_1, \dots, c_m\}$ from the Recursive RLS sampler of [17]. From the proof of [Theorem 5.1](#), we recall that the orthogonal projection approximation based on given control point set, satisfies

$$\left\| \sum_{i=1}^n \delta_{x_i} \hat{\alpha}_i - \sum_{i=1}^m \delta_{c_i} \beta_i \right\|_{V^*}^2 \leq C \text{tr}(K_{XX} - Q_{XX}).$$

The inequality (6.3) implies that

$$(6.4) \quad \text{tr}(K_{XX} - Q_{XX}) \leq n \|K_{XX} - Q_{XX}\|_2,$$

and therefore,

$$\left\| \sum_{i=1}^n \delta_{x_i} \hat{\alpha}_i - \sum_{i=1}^m \delta_{c_i} \beta_i \right\|_{V^*}^2 \leq Cn \|K_{XX} - Q_{XX}\|_2.$$

By applying the inequality of (4.5) for Recursive RLS sampling to the right-hand side, we know with probability $1 - 3\delta$ that $m < cS \log(S/\delta)$ and

$$\left\| \sum_{i=1}^n \delta_{x_i} \hat{\alpha}_i - \sum_{i=1}^m \delta_{c_i} \beta_i \right\|_{V^*}^2 \leq \frac{Cn}{S} \sum_{i=S+1}^n \lambda_i(K_{XX}).$$

This concludes the proof. ■

Finally, we give proof of [Corollary 5.2](#) now follows by the eigenvalue bounds presented in section 4.0.1, and taking expectation/union bounds.

Proof of Corollary 5.2. Both the m -DPP and RLS sampling case, are upper bounded upto a multiplicative constant, by the term

$$\sum_{i=m+1}^n \lambda_i(K_{XX}),$$

as a function of m . By the identity (4.2), we have

$$\mathbb{E}_X \left[\sum_{i=m+1}^n \lambda_i(K_{XX}) \right] \leq n \sum_{i=m+1}^{\infty} \lambda_i.$$

Markov's inequality may also be applied to the above to yield probability $1 - \delta$ bound of,

$$\sum_{i=m+1}^n \lambda_i(K_{XX}) \leq \frac{n}{\delta} \sum_{i=m+1}^{\infty} \lambda_i.$$

Taking expectation/union bound of and inserting the above bounds into the m -DPP and RLS cases of [Theorem 5.1](#) respectively gives the first part of [Corollary 5.2](#).

For the proof of the second part of [Corollary 5.2](#), it is known by lemma 11 of [7], that the eigenvalue decay of a mixture of Gaussians can be upper bounded by that of a single Gaussian p with sufficiently large variance. The rate of decay of eigenvalues of \mathcal{K}_p for a single Gaussian p is already known to be (4.3), which yields the desired exponential decays, by inserting the estimates into the first part of [Corollary 5.2](#). ■

7. Numerical Experiments. We now illustrate the performance of our algorithm on compression of real-world shapes taken from modern geometry processing datasets. In particular, we explore the following main strengths of our algorithm for compression of geometric measures: **Fast compression times**, **Rapid error decay** supported by theoretical bounds, and applicability to both currents and varifolds. We illustrate both the true error and theoretical rates when computationally feasible. In all subsequent experiments in this section, we use the algorithm of [11] for RLS approximation as opposed to the recursive algorithm of [17], for which we prove the second bound of [Theorem 5.1](#). Both methods yield similar theoretical upper bounds on the Nystrom error, with the [17] being tighter. However, despite the tighter theoretical bounds of [17], in practice the algorithm of [11] gives similar error decay while being significantly faster in approximating RLS scores. Therefore, this is our preferred choice of RLS approximation to use in [Algorithm 5.1](#). We make the code used to generate the results of this section publicly available ².

7.1. Error decay. In this section, we study rate of decay of the compression error as a function of m , the number of samples. We use the following surface data for our experiments in this section.



(a) **Left:** Cat (14410 triangles) **Middle:** Head (31620 triangles) **Right:** Flamingo (52895 triangles)

The data are centred and scaled, so that the cat surface lies in a box of size $1.3 \times 3.3 \times 7.1$, the head surface lies in a box of size $3.8 \times 5.3 \times 4.0$ and the flamingo surface lies in a box of size $1.5 \times 5.3 \times 3.8$.

We run the RLS currents compression algorithm, [Algorithm 5.1](#), on our test surfaces which are centred and normalized, prior to our experiments. For each test surface, we calculate and plot the true squared error $E = \|\mu - \hat{\mu}_m\|_{W^*}^2$ of the compression using RLS as a function of m , where μ is the target, and $\hat{\mu}_m$ denotes an approximation formed with m delta-centres. We also plot for comparison, the error curve for uniform sampling, where control points are simply subsampled from the discrete uniform distribution. Finally, the trace bound on the squared error derived in [Corollary 6.2](#) is also plotted, with rescaled numerically tighter constants. Note that we do not plot the curves for eigenvalue bounds, due to the prohibitive cost of computing the eigendecomposition as $m \rightarrow n$.

²<https://github.com/allenpaul0/GeometricMeasureCompression>

For the currents experiments, we choose the Gaussian kernel $k(x, y) = \exp(-\frac{\|x-y\|^2}{2\sigma^2})$ for K_p , and set the spatial kernel equal to $\sigma = 0.5$ for all three test cases. For the varifolds experiments, we choose the Gaussian kernel $k(x, y) = \exp(-\frac{\|x-y\|^2}{2\sigma^2})$ for K_p , and the spherical Gaussian kernel $k(s, r) = \exp(-\frac{(2-2\langle s, r \rangle)}{2\sigma_s^2})$ for K_s . Here, we choose $\sigma_s = 0.5$ and $\sigma_p \in \{0.3, 0.5, 0.25\}$ respectively, for the cat, head and flamingo test cases.

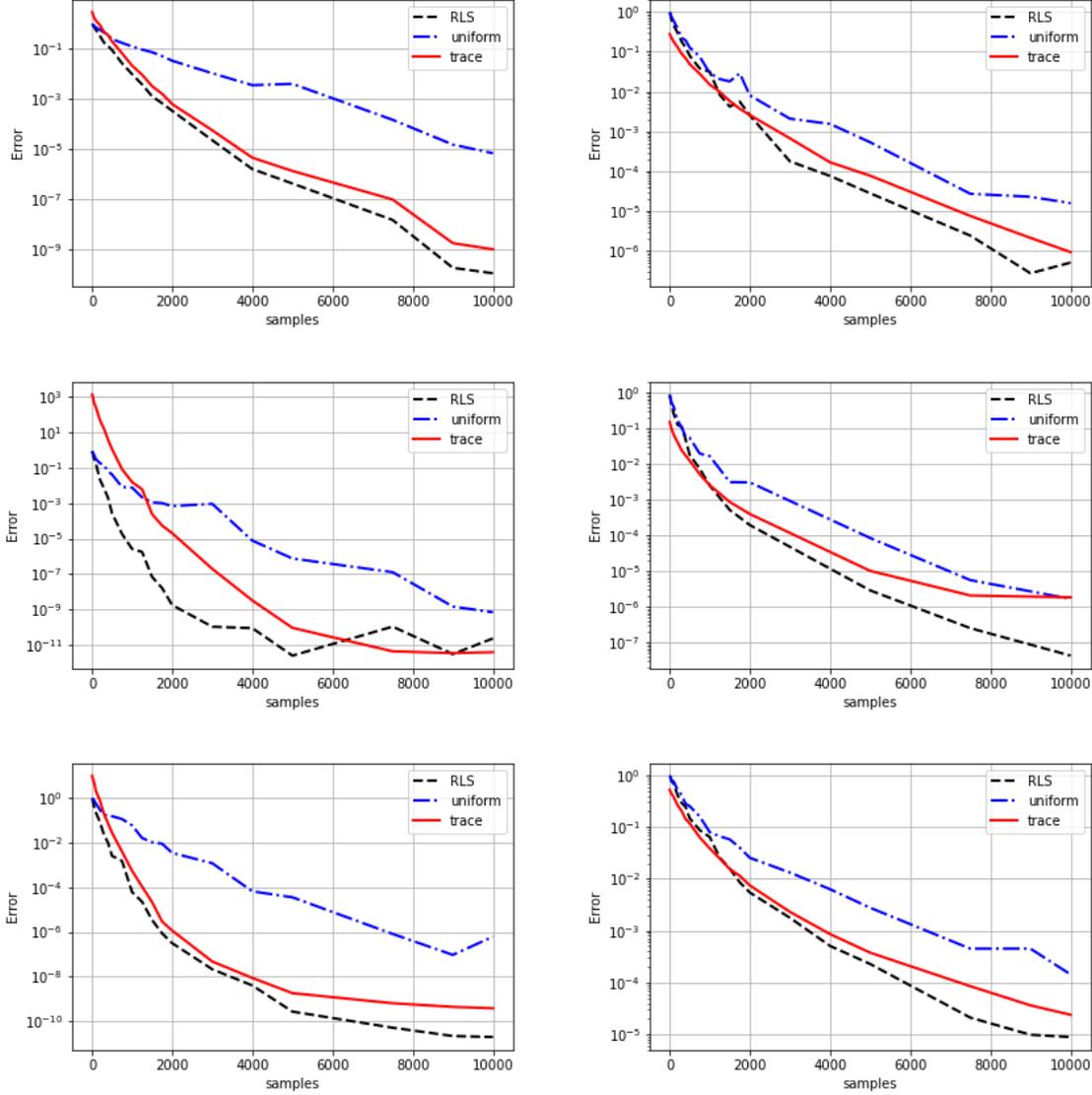


Figure 2: Numerical curves comparing error decay of RLS compression (black) to theory bound (red) and uniformly sampled compression (blue), on cat (top), head (middle) and flamingo (bottom) surfaces. **Left:** Error curves for currents. **Right:** Error curves for varifolds.

We observe in all cases, that the decay of error of the compressed approximation is rapid,

and one can take $m \ll n$ for a good quality of approximation, across all the example cases. One also observes, that the trace bounds (in red), have decay rate that generally matches that of the true squared error. Finally, we observe that RLS sampling consistently outperforms Uniform sampling in all cases.

We also note that in practice, the RLS sampling tends to produce better *quality* samples (in terms of squared error) than uniform sampling. In general, uniform sampling will tend to have more points sampled in regions which are oversampled to begin with, thus only representing the underlying surface well only in densely sampled regions. On the other hand, RLS sampling measures the local ‘importance’ of triangles through the ridge leverage scores and samples them accordingly. Thus, RLS sampling tends to produce more ‘diverse’ samples [17] and will produce samples that are well spread out independent of the initial sampling density.

Finally, we note that in many geometry processing applications, one often does not require an extremely small error in order to perform down-line tasks with the compressed representation. In many situations, the data itself is often acquired in a noisy way, and can contain many local variations that are not relevant in describing the global geometry. As such, the above curves suggest one can practically choose $m \ll n$ and obtain an acceptable error for the tasks which we perform with the compressions. For example, [12] suggests a heuristic of $\tau = 5\%$ relative error cut-off for compression of currents in the orthogonal matching pursuit algorithm.

7.2. Matching quality. We now illustrate the effectiveness of the compressed measures for nonlinear shape registration in the LDDMM framework. In particular, we demonstrate that one can obtain comparable quality of registration to the full (uncompressed) matching problem, when using the compressed representation with $m \ll n$, even when only 1 – 2% of the underlying triangles are used in the compression. We shall also demonstrate that using the compressed representation also gives a massive computational savings for the registration algorithm, in terms of memory and run-time.

We shall compare the quality of the obtained registrations Hausdorff metric, rather than in the varifolds metric as it is independent of the optimization objective and gives a measure alignment of both surfaces without correspondence. Recall that the Hausdorff metric between two sets $A, B \subset \mathbb{R}^d$ is defined as follows,

$$d_H(A, B); = \max\left(\sup_{a \in A} d(a, B), \sup_{b \in B} d(b, A)\right), \quad d(x, A) := \inf_{a \in A} \|x - a\|_2.$$

Quality of registration is compared by computing the Hausdorff metric between target, and deformed template.

We demonstrate on two extremely densely sampled shapes, taken from modern geometry processing datasets. The first, a super high-resolution version of the Stanford bunny with 259,898 triangles, and the second a high-resolution brain surface taken from the Thing10K dataset with 350,328 triangles. The bunny is centred and scaled to lie in a box of size $4.0 \times 4.1 \times 3.4$, and the brain surface to a box of size $3.5 \times 3.5 \times 3.6$.

In both cases, we consider an initial momentum LDDMM matching problem 5.4, using the varifolds kernel with Gaussian spatial and spherical part with parameters $\sigma_p, \sigma_s > 0$ respectively, and a spherical template. We consider varifolds as our choice of discrepancy, as they often perform better than currents, at matching complex surfaces. For the spatial kernel

K_V parametrizing vector fields in the LDDMM framework, we fix a sum of 4 Gaussian kernels of decreasing length-scales $\sigma_i \in \{1.0, 0.5, 0.2, 0.1\}$. For the varifolds kernels we choose K_p to be Gaussian with length-scale $\sigma_p = 0.2$, and K_s to be spherical Gaussian of length-scale $\sigma_s = 0.3$.

With this problem configuration, in both cases we compare the matching quality and matching time taken for the uncompressed, and compressed matching (using algorithm 5.3) problems. For all experiments, the diffeomorphism and push-forward are computed via a forward Euler scheme with 10 time-steps. The kernel and gradient computations for the diffeomorphisms, are performed using Keops and automatic differentiation. Optimization is performed via an LBFGS routine, with strong Wolfe line search run for 500 iterations. All experiments are performed on a Tesla T4 GPU with 16gb of RAM.

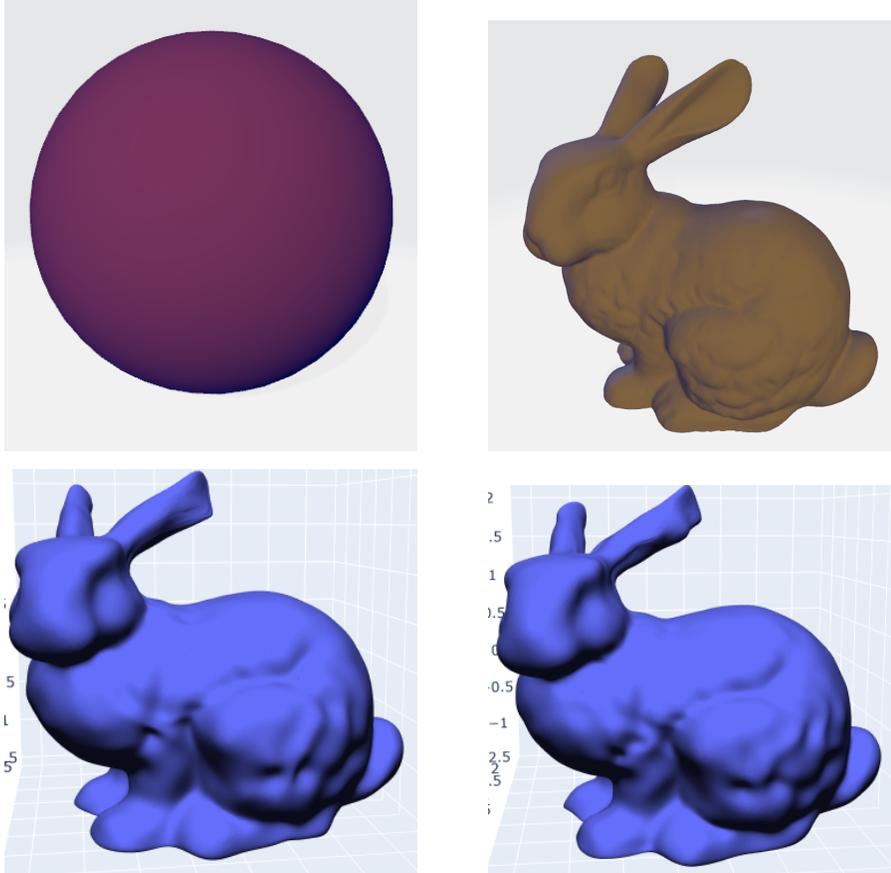


Figure 3: **Top left:** spherical template. **Top right:** target mesh. **Bottom left:** Matching with full metrics taking 1 hour and 41 minutes, and Hausdorff metric error $d_H = 0.026$. **Bottom right:** Matching with 97% compression of template and target taking only 14 minutes, and Hausdorff metric error $d_H = 0.030$.

In the above, we compress the target and template down to 7500 triangles each, which is a compression ratio of 97%, giving a significant memory saving of the resulting matching algorithm. We observe in the figure above, that the matching quality is almost identical in d_H between compressed and uncompressed and in some regions better than the full matching. As

one expects, the compressed matching algorithm yields a significant speedup of 6 – 7 times over the uncompressed version, reducing overall matching time from 5918s to 860s.

In the second example, we consider the analogous matching problem, for a high resolution brain surface taken from the Thingi10K dataset with 350,328 triangles. The data and results are shown below.

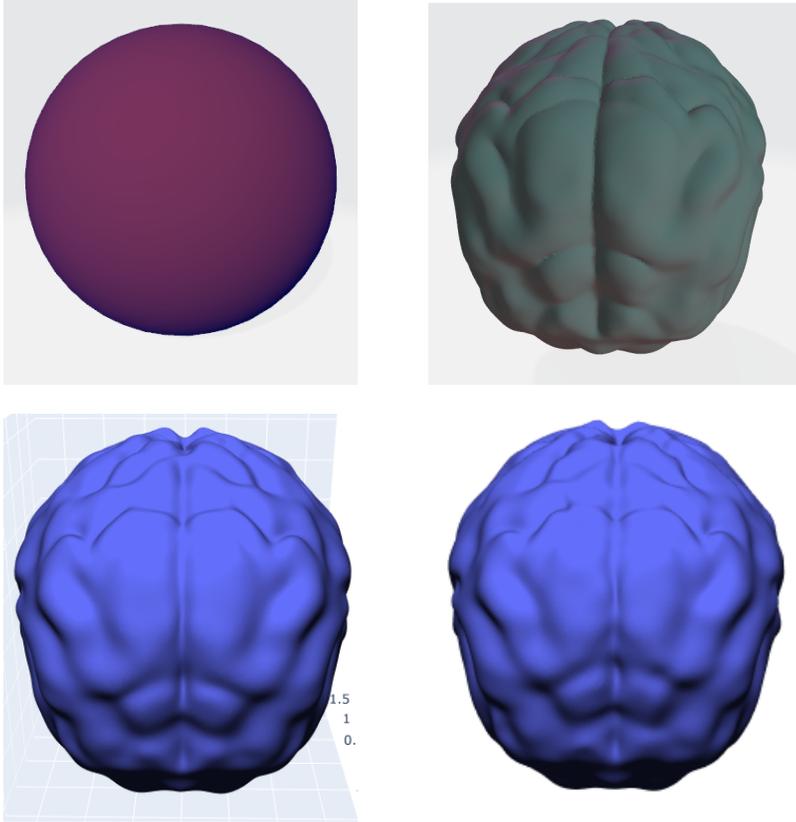


Figure 4: **Left:** an example of LDDMM matching with Varifolds, from spherical template to brain without compression, taking 1 hour and 49 minutes, and Hausdorff metric error $d_H = 0.005$. **Right:** the same example matching but with 99% compression taking only 11 minutes, and Hausdorff metric error $d_H = 0.007$.

Once again, we compress the target and template down to 5000 triangles each, which is yields a compression ratio of 99%. This yields huge memory savings and almost a factor of 10 times speed-up over the uncompressed matching problem. Indeed, overall matching time is reduced from 6540s to 660s, on 500 iterations while still maintaining a similar d_H error.

7.3. Compression speed. Finally, we compare the runtime of our compression algorithm, to existing compression methods for currents [12] and varifolds [16]. We demonstrate in this section that one can compress currents and varifolds to a fixed size using our algorithm, in a fraction of the run-time of the existing algorithms, while retaining fast decay of the compression error in dual metric as a function of m . This makes the RLS compression particularly suited to real-time/per-iteration compression where one wishes to compress shapes routinely as part of a

general shape based learning algorithm; e.g. the LDDMM matching framework. Furthermore, in situation where multiple compressed measures need to be computed, such as group-wise shape modelling, our algorithm has a distinct advantage due to rapid compression times, and error decay. All experiments are performed on a Tesla T4 GPU with 16gb of ram.

First we illustrate, how the run-time (in seconds) and compression error of the RLS scheme compares to the existing state of the art greedy method [12] for **currents** compression. We demonstrate this on the cat and head example shapes from section 7.1. For both examples, we set the spatial kernel K_p to be Gaussian with length-scale $\sigma = 0.5$.

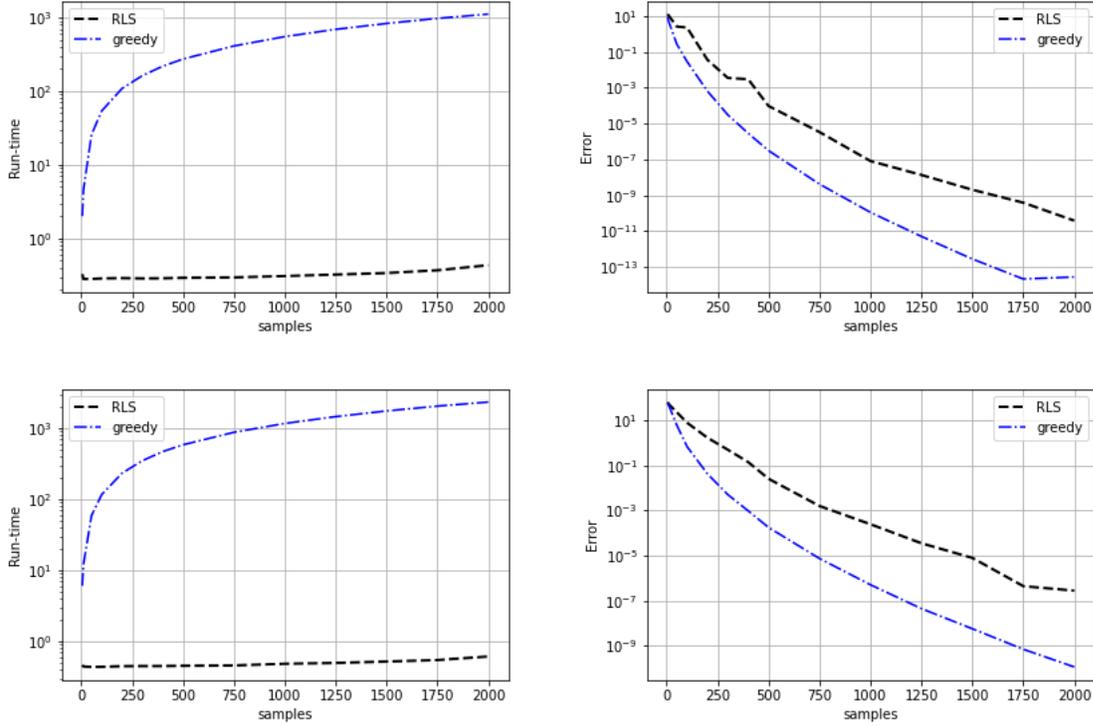


Figure 5: Run-time in seconds (left column) and error comparison (right column) of RLS method vs Greedy method [12] for compression of **currents** on example surface ‘cat’ (top row) and ‘head’ (bottom row).

In figure 5, we observe that our proposed compression algorithm is up to 1000× faster than the existing method for current compression for compressing to a fixed size. The run-time of [12] grows rapidly as a function of sample size m , whereas our method grows sub-linearly. In terms of error, we observe that the rate of decay is similarly fast for both methods, although the greedy method tends to require slightly fewer samples to achieve a similar error.

Next, we illustrate how the run-time and compression error of the RLS scheme compares to the existing state [16] of the art for **varifolds** compression. These comparisons are made on the bunny and brain examples of section 7.2. We are able to make the comparison here on much larger examples, as the RLS sampling and varifold quantization [16] both have run-times

that are much faster than [12] so we are able to generate the following curves in a reasonable time. For both examples, we set K_p to be the Gaussian kernel with length-scale $\sigma = 0.5$, and K_s to be spherical gaussian with $\sigma_s = 0.5$.

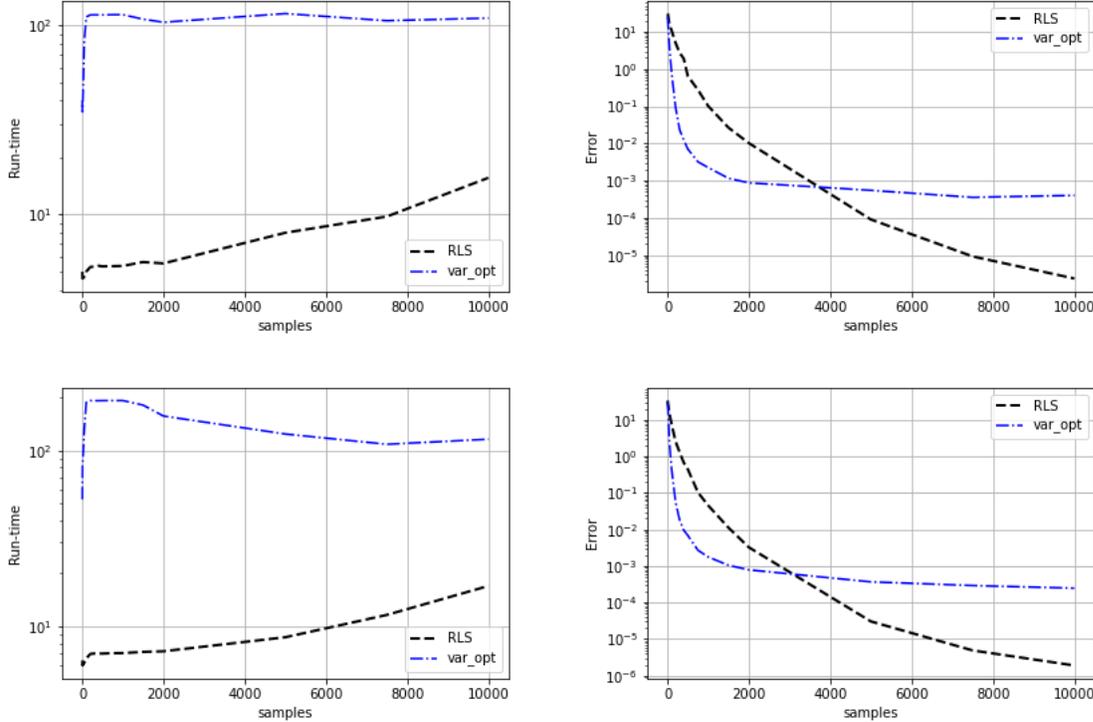


Figure 6: Run-time in seconds (left column) and error comparison (right column) of RLS method vs optimization method [16] for compression of **varifolds**, on example surface ‘bunny’ (top row) and ‘brain’ (bottom row).

For varifolds, we observe that the RLS method outperforms the optimization based approach in run-time, and has similar rate of decay of the compression error. Indeed, the optimization based approach while having fast initial decay, can become stuck in local minima as the sample size increases, while the RLS based approach continues to decrease the error as m increases. Furthermore, the run-time comparison on both examples demonstrates how the RLS method tends to be on average 10 – 100 \times faster than the optimization based approach.

8. Conclusion. In this work, we have derived an algorithm for compression of large-scale currents and varifolds using the Nystrom approximation in Reproducing Kernel Hilbert Spaces, and approximate Ridge Leverage Score sampling methods. We have derived convergence bounds and rates of convergence on this compression algorithm, as a function of the chosen kernels and associated smoothness parameters. The rapidly decaying error bounds, scalability and fast run-times of the algorithm, make it well-suited to routine use as well as pre-processing of measure based shape representations. Numerous practical examples highlight the strengths of this framework, especially in the large-scale setting. We have also demonstrated the benefits

of our method over existing compression techniques, that rely on greedy iterative schemes and non-convex optimization procedures both of which become slow in the large-scale setting.

We leave as future work the task of extending/modifying our algorithm to compression of higher-order geometric measure representations such as normal-cycles [18]; a generalization of currents, which is provably sensitive to curvature information and boundaries, in discrete and continuous shapes. Such properties make it an attractive choice of shape metric, especially for nonlinear shape registration frameworks, such as LDDMM. However, the normal-cycles representation comes at a significantly increased computational cost, making it unattractive in large-scale applications. A compression algorithm adapted to the normal-cycle representation would finally allow the routine use of such higher-order measures, on large-scale geometry processing applications.

Appendix A. Appendix.

A.1. Norm Splitting. We now show that computation of the squared V norm for certain types of vector fields we are interested in, splits into a sum of squared V_k norms over each dimension.

Lemma A.1. *Suppose, that we are given an RKHS of vector fields V induced by a scalar diagonal reproducing kernel of the form $K(x, y) = k(x, y)Id$. Given $v \in V$ of the form:*

$$v(x) = \sum_{i=1}^n K(x, x_i) \alpha_i,$$

the V norm (induced by K) has the following form:

$$(A.1) \quad \|v\|_V^2 = \sum_{i,j=1}^n \alpha_j^T K(x_i, x_j) \alpha_i,$$

which for scalar diagonal kernels reduces to,

$$(A.2) \quad \|v\|_V^2 = \sum_{l=1}^d \|v_l\|_{V_k}^2,$$

where v_l denotes the dimension l component of v and V_k is the RKHS of real valued functions induced by k .

Proof. The identity (A.1) is standard and follows from the following identities in RKHS

$$\begin{aligned} \|v\|_V^2 &= (Lv, v) \\ L(K(\cdot, x)\alpha) &= \alpha^T \delta_x. \end{aligned}$$

For the second claim, it is an easy computation that

$$\begin{aligned} \|v\|_V^2 &= \sum_{i,j=1}^n k(x_i, x_j) \alpha_j^T \alpha_i = \sum_{i,j=1}^n k(x_i, x_j) \sum_{l=1}^d \alpha_{jl} \alpha_{il} = \sum_{i,j=1}^n \sum_{l=1}^d k(x_i, x_j) \alpha_{jl} \alpha_{il} \\ &= \sum_{l=1}^d \sum_{i,j=1}^n k(x_i, x_j) \alpha_{jl} \alpha_{il} = \sum_{l=1}^d \|v_l\|_{V_k}^2, \end{aligned}$$

where v_l denotes the dimension l component of v so that

$$v_l = \sum_{i=1}^n k(\cdot, x_i) \alpha_{il},$$

with weights $\alpha_l = (\alpha_{il})_{i=1}^n$. ■

A.2. MCMC k -DPP. The most popular way to sample cheaply from a k -dPP is to run an MCMC chain that converges to the target k -DPP. There is a large literature on deriving fast mixing MCMC algorithms for k -DPP sampling. We give one example here from [1]

Algorithm A.1 MCMC for sampling a k -DPP

- 1: Initialise number of points m , kernel function k , number of MCMC chain iterations R to sample from $\mathbf{X} = \{x_1, \dots, x_n\}$, and uniformly sample an index set S_0 of size m .
 - 2: **while** $r < R$ **do**
 - 3: Sample i uniformly from S_r and j uniformly from $\mathbf{X} - S_r$. Define the set $T = (S - i) \cup j$.
 - 4: Compute transition probabilities $p_{ij} = \frac{1}{2} \min 1, \det(K_T) / \det(K_{S_r})$.
 - 5: Sample from the transition probability so that with probability p_{ij} we have $S_{r+1} = T$, otherwise $S_{r+1} = S$.
 - 6: **end while**
 - 7: Return sample from k -DPP S_R
-

Using a smart way to compute the inner loop, one can show the per-iteration cost is $\mathcal{O}(m^2)$. This chain is observed to converge to the target m -DPP much faster in practice than the theoretical bounds, as evidenced in [1]. Furthermore, one obtains the theoretical guarantees of sections 6, asymptotically as the chain converges.

REFERENCES

- [1] N. ANARI, S. OVEIS GHARAN, AND A. REZAEI, *Monte carlo markov chain algorithms for sampling strongly rayleigh distributions and determinantal point processes*, in 29th Annual Conference on Learning Theory, V. Feldman, A. Rakhlin, and O. Shamir, eds., vol. 49 of Proceedings of Machine Learning Research, Columbia University, New York, New York, USA, 23–26 Jun 2016, PMLR, pp. 103–115, <https://proceedings.mlr.press/v49/anari16.html>.
- [2] F. BACH, *On the equivalence between kernel quadrature rules and random feature expansions*, Journal of machine learning research, 18 (2017), pp. 1–38.
- [3] M.-A. BELABBAS AND P. J. WOLFE, *Spectral methods in machine learning and new strategies for very large datasets*, Proceedings of the National Academy of Sciences - PNAS, 106 (2009), pp. 369–374.
- [4] A. BELHADJI, R. BARDENET, AND P. CHAINAIS, *Kernel quadrature with dpps*, in Advances in Neural Information Processing Systems, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds., vol. 32, Curran Associates, Inc., 2019, https://proceedings.neurips.cc/paper_files/paper/2019/file/7012ef0335aa2adbab58bd6d0702ba41-Paper.pdf.
- [5] A. BELHADJI, R. BARDENET, AND P. CHAINAIS, *Kernel interpolation with continuous volume sampling*, in Proceedings of the 37th International Conference on Machine Learning, H. D. III and A. Singh, eds., vol. 119 of Proceedings of Machine Learning Research, PMLR, 13–18 Jul 2020, pp. 725–735, <https://proceedings.mlr.press/v119/belhadji20a.html>.
- [6] D. BURT, C. RASMUSSEN, AND M. VAN DER WILK, *Rates of convergence for sparse variational gaussian process regression*, PMLR, 2019.
- [7] D. R. BURT, C. E. RASMUSSEN, AND M. VAN DER WILK, *Convergence of sparse variational inference in gaussian processes regression*, Journal of Machine Learning Research, 21 (2020), pp. 1–63, <http://jmlr.org/papers/v21/19-1015.html>.
- [8] B. CHARLIER, J. FEYDY, J. A. GLAUNÈS, F.-D. COLLIN, AND G. DURIF, *Kernel operations on the gpu, with autodiff, without memory overflows*, Journal of Machine Learning Research, 22 (2021), pp. 1–6, <http://jmlr.org/papers/v22/20-275.html>.
- [9] N. CHARON, B. CHARLIER, J. GLAUNÈS, P. GORI, AND P. ROUSSILLON, *12 - fidelity metrics between curves and surfaces: currents, varifolds, and normal cycles*, in Riemannian Geometric Statistics in Medical Image Analysis, X. Pennec, S. Sommer, and T. Fletcher, eds., Academic Press, 2020, pp. 441–

- 477, <https://doi.org/https://doi.org/10.1016/B978-0-12-814725-2.00021-2>, <https://www.sciencedirect.com/science/article/pii/B9780128147252000212>.
- [10] N. CHARON AND A. TROUVÉ, *The varifold representation of nonoriented shapes for diffeomorphic registration*, SIAM journal on imaging sciences, 6 (2013), pp. 2547–2580.
 - [11] F. CHERFAOUI, H. KADRI, AND L. RALAIVOLA, *Scalable ridge leverage score sampling for the nystrom method*, in ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2022, pp. 4163–4167, <https://doi.org/10.1109/ICASSP43922.2022.9747039>.
 - [12] S. DURRLEMAN, X. PENNEC, A. TROUVÉ, AND N. AYACHE, *Statistical models of sets of curves and surfaces based on currents*, Medical Image Analysis, 13 (2009), pp. 793–808, <https://doi.org/https://doi.org/10.1016/j.media.2009.07.007>, <https://www.sciencedirect.com/science/article/pii/S1361841509000620>. Includes Special Section on the 12th International Conference on Medical Imaging and Computer Assisted Intervention.
 - [13] J. GLAUNÈS, A. QIU, M. I. MILLER, AND L. YOUNES, *Large deformation diffeomorphic metric curve mapping*, International journal of computer vision, 80 (2008), pp. 317–336.
 - [14] P. GORI, *Parsimonious approximation of streamline trajectories in white matter fiber bundles.*, IEEE Transactions on Medical Imaging, 35 (2016), pp. 2609–2620.
 - [15] S. HAYAKAWA, H. OBERHAUSER, AND T. LYONS, *Positively weighted kernel quadrature via subsampling*, in Advances in Neural Information Processing Systems, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, eds., vol. 35, Curran Associates, Inc., 2022, pp. 6886–6900, https://proceedings.neurips.cc/paper_files/paper/2022/file/2dae7d1ccf1edf76f8ce7c282bdf4730-Paper-Conference.pdf.
 - [16] H.-W. HSIEH AND N. CHARON, *Metrics, quantization and registration in varifold spaces*, Foundations of computational mathematics, 21 (2021), pp. 1317–1361.
 - [17] C. MUSCO AND C. MUSCO, *Recursive sampling for the nystrom method*, in Advances in Neural Information Processing Systems, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds., vol. 30, Curran Associates, Inc., 2017, https://proceedings.neurips.cc/paper_files/paper/2017/file/a03fa30821986dff10fc66647c84c9c3-Paper.pdf.
 - [18] P. ROUSSILLON AND J. A. GLAUNÈS, *Representation of surfaces with normal cycles and application to surface registration*, Journal of mathematical imaging and vision, 61 (2019), pp. 1069–1095.
 - [19] M. VAILLANT AND J. GLAUNÈS, *Surface matching via currents*, INFORMATION PROCESSING IN MEDICAL IMAGING, PROCEEDINGS, 3565 (2005), pp. 381–392.
 - [20] V. WILD, M. KANAGAWA, AND D. SEJDINOVIC, *Connections and equivalences between the nystrom method and sparse variational gaussian processes*, arXiv.org, (2021).
 - [21] C. WILLIAMS AND M. SEEGER, *Using the nystrom method to speed up kernel machines*, in Advances in Neural Information Processing Systems, T. Leen, T. Dietterich, and V. Tresp, eds., vol. 13, MIT Press, 2000, https://proceedings.neurips.cc/paper_files/paper/2000/file/19de10adbaa1b2ee13f77f679fa1483a-Paper.pdf.