PNAS
www.pnas.org

# Supplementary Information for

## Computing the Riemannian curvature of image patch and single-cell RNA sequencing data manifolds using extrinsic differential geometry

**Duluxan Sritharan, Shu Wang and Sahand Hormoz**

**Sahand Hormoz.**
**E-mail: Sahand_Hormoz@hms.harvard.edu**

**This PDF file includes:**

## Supporting Methods

**A. Differential Geometry of Theoretical Manifolds.** Here we briefly discuss how to compute the scalar curvature of, and sample from, theoretical manifolds given a parameterization.

For a $d$-dimensional manifold, $M$, with intrinsic coordinates $\{x_1, ..., x_d\}$ and embedding in $\mathbb{R}^n$ given by $f(x_1, ..., x_d)$, the metric is:

$$g_{ij} = \frac{\partial f^T}{\partial x^i} \frac{\partial f}{\partial x^j} \quad [1]$$

The scalar curvature of $M$ can then be derived analytically in intrinsic coordinates in terms of the metric as

$$S = g^{ij} \left( \Gamma^k_{ij,k} - \Gamma^k_{ik,j} + \Gamma^l_{ij}\Gamma^k_{kl} - \Gamma^l_{ik}\Gamma^k_{jl} \right) \quad [2]$$

where the $\Gamma^i_{jk}$s are Christoffel symbols given by

$$\Gamma^i_{jk} = \frac{g^{il}}{2} \left( \frac{\partial g_{lj}}{\partial x^k} + \frac{\partial g_{lk}}{\partial x^j} - \frac{\partial g_{jk}}{\partial x^l} \right) \quad [3]$$

and $\Gamma^i_{jk,l} = \frac{\partial \Gamma^i_{jk}}{\partial x^l}$.

To draw points from $M$ with $a_i \leq x_i \leq b_i$ so that the embedded manifold is uniformly sampled in $\mathbb{R}^n$, we use rejection sampling. For paired random variables $x \sim \text{Uniform}(a, b)$ and $y \sim \text{Uniform}(0, \max \sqrt{\det g})$, we retain $x$ as a sample point if $\sqrt{\det g}\big|_x \leq y$.

**B. Details of Intrinsic Approach to Curvature Estimation.** Here we explain how we used Equations 2-4 in the main text on the simplest of toy manifolds, the noise-free 2-dimensional hollow unit sphere, $\mathcal{S}^2$, to obtain an estimate of the average scalar curvature. The true scalar curvature is $S(p) = 2 \ \forall p \in M$. For the remainder of this section, we adopt the convention that symbols with overbars are estimates of the corresponding unaccented quantities.

***B.1. Approach for $\mathcal{S}^2$.*** Our approach mirrors the treatment in (1), in which heat-traces are fit over various intervals $[x_1, x_2]$ with $x_1 \geq 0$, to quadratic polynomials $\overline{p}_2(x) = \overline{c}_0 + \overline{c}_1 x + \overline{c}_2 x^2$ to estimate the geometric quantities in Equation 2 of the main text.

Here, we constrained the form of $\overline{p}_2(x)$ for fitting by assuming that (i) the manifold is boundary-less (so that $\overline{c}_1 = c_1 = 0$ and the second boundary term for $c_2$ vanishes), (ii) the volume is known (so that $\overline{c}_0 = c_0 = 4\pi$), and (iii) the scalar curvature is constant (so that $\overline{c}_2 = \frac{2\pi}{3}\overline{S}$), yielding $\overline{p}_2(x) = 4\pi + \frac{2\pi}{3}\overline{S}x^2$. These are strong assumptions that will not hold for an arbitrary manifold, which already precludes this as a generic procedure. Nonetheless, we proceeded for $\mathcal{S}^2$ to see if even with this privileged information, the scalar curvature could be estimated accurately. We declared an estimate to be accurate on the interval $[x_1, x_2]$ if $\overline{S}$ has error within $\pm 0.5$ i.e. $\overline{S} \in [1.5, 2.5]$. All quadratic fits were performed in MATLAB using the *lsqnonlin* function ('StepTolerance'=1e-3, 'FunctionTolerance'=1e-6).

First, we evaluated $z_m(x)$ using analytical eigenvalues for $\mathcal{S}^2$ given by $\lambda_{(\ell-1)^2+1}, ..., \lambda_{\ell^2} = \ell(\ell-1), \ell > 0$, and let $D_m$ be the collection of all intervals for which fits to $\overline{p}_2(x)$ yielded accurate $\overline{S}$. $D_m$ corresponds to intervals where Equation 4 of the main text is accurate to our desired tolerance when the eigenvalues are known exactly.

Next, we uniformly sampled $N = 10^4$ points from $\mathcal{S}^2$ (see Figure 1A; Methods Section D.1.1), estimated $\overline{\Delta}_M$ using the *random walk Graph Laplacian* with Gaussian kernel (see Equations 6-8 in Methods Section B.6), and computed empirical eigenvalues, $\overline{\lambda}_k$, from $\overline{\Delta}_M$. We selected $N = 10^4$ as it is the same order of magnitude as the sample size of current scRNAseq experiments, and is sufficient to identify $M$ as $\mathcal{S}^2$ by eye (see Figure 1A). We verified if estimates $\overline{z}_m(x)$, obtained by evaluating Equation 3 of the main text using $\overline{\lambda}_k$, when fit as described above to $\overline{p}_2(x)$ over intervals in $D_m$, recapitulated the accurate $\overline{S}$ obtained using $z_m(x)$. We restricted our attention to $D_m$ for calculations using empirical eigenvalues, since it is only over intervals in $D_m$ that it is even theoretically possible to compute scalar curvature to the desired accuracy.

Below, we report our findings for different $m$.

***B.2. Infinite Series.*** We first applied this approach to the ideal case in Equation 3 of the main text, where infinite analytical eigenvalues are available. We computed $z_\infty(x)$ (shown as a black line in Figure S1A) and obtained $\overline{S}$ by fitting $\overline{p}_2(x)$ over various intervals as described above. Figure S1B shows that $D_\infty$ is comprised of intervals with $0 \leq x_1 < x_2 \lesssim 1.15$. For $x_2 \gtrsim 1.15$, errors from neglecting higher-order terms $o(x^3)$ in Equation 4 of the main text dominate. Since $z_m(x)$ converges from $\infty$, $x_2 \lesssim 1.15$ necessarily holds for any interval in $D_m \ \forall m$.

***B.3. Truncated Series.*** We next considered $z_m(x)$ for $m < N$, since in practice, we will only have access to as many eigenvalues as datapoints ($N$). We computed $z_{1000}(x)$ using Equation 3 of the main text (shown as a solid blue line in Figure S1A), and obtained $\overline{S}$ by fitting $\overline{p}_2(x)$ (see Figure S1C). Intervals in $D_{1000}$ roughly satisfy $0.25 \lesssim x_1 < x_2 \lesssim 1.15$. However, we found that $\overline{z}_{1000}(x)$ (shown as a dashed blue line in Figure S1A) deviated markedly from $z_{1000}(x)$ in the rough interval $[0.1, 0.75]$, which has significant overlap with $D_{1000}$. Consequently, when we fit $\overline{p}_2(x)$ to $\overline{z}_{1000}(x)$ on $D_{1000}$, the resulting $\overline{S}$ was not accurate for any interval in $D_{1000}$ (see Figure S1D).

Note that this inaccuracy was not a consequence of not using all $N$ available eigenvalues. While picking $m = N$ would reduce the lower bound on valid intervals in $D_m$ (since $z_m(x)$ converges from $\infty$), it is exactly for small $x_1$ that $\overline{S}$ obtained from $\overline{z}_{1000}(x)$ is already over-estimated as shown in Figure S1D. Since $\overline{z}_{m_2}(x) > \overline{z}_{m_1}(x) \ \forall x, m_2 > m_1$, using a truncated series

with a larger $m$ would simply exaggerate the difference between $z_m(x)$ and $\overline{z}_m(x)$ for small $x$ and cause scalar curvatures estimated using the latter to be further over-estimated.

Following this line of thought, we reasoned that picking a fewer number of eigenvalues may ameliorate the issue. We selected $m = 49$ (instead of a round number like $m = 50$ so that all eigenvalues of a given multiplicity are included) and repeated this analysis for the same set of $N = 10^4$ points. $z_{49}(x)$ is shown as a solid red line in Figure S1A and the intervals over which fits to $\overline{p}_2(x)$ yield accurate $\overline{S}$, $D_{49}$, are shown in Figure S1E. While $\overline{z}_{49}(x)$ (shown as a dashed red line in Figure S1A) has a much smaller deviation from $z_{49}(x)$ than $\overline{z}_{1000}(x)$ did from $z_{1000}(x)$, no estimate of $\overline{S}$ obtained from fits of $\overline{z}_{49}(x)$ to $\overline{p}_2(x)$ on $D_{49}$ were sufficiently accurate once again (see Figure S1F).

***B.4. Eigenvalue Convergence.*** We refrained from reducing $m$ further to improve agreement between $\overline{z}_m(x)$ and $z_m(x)$ after noting that the size of the intervals in $D_m$ shrink with $m$. Though we may have a better chance of computing accurate $\overline{S}$ with $\overline{z}_m(x)$ on $D_m$ for smaller $m$, recall that in practice we will not have $D_m$ available to us since the analytical eigenvalues will be unknown. Therefore, we simply shift the problem to one of choosing an interval that will yield an accurate $\overline{S}$, from a shrinking pool of intervals that could even theoretically yield an accurate estimate.

Instead, we compared the estimated $\overline{\lambda}_k$s with their true values, $\lambda_k$, and observed that the former consistently under-estimate the latter (see Figure S1H). Furthermore, we found that the fractional error grows with $k$, and is $\approx 62\%$ for $k = 37, ..., 49$. Therefore, $\overline{z}_{49}(x)$ will only be accurate if $N$ is large enough to limit the fractional error.

To determine the required tolerance on the fractional error, we constructed a truncated series analogous to Equation 3 of the main text, but with eigenvalues interpolated between the analytical eigenvalues and the empirical eigenvalues determined for $N = 10000$, according to a parameter $f$:

$$\begin{aligned} \tilde{z}_m(x; f) &= (4\pi)^{\frac{d}{2}} x^d \sum_{k=1}^{m} e^{-\tilde{\lambda}_k(f)x^2} \\ \tilde{\lambda}_k(f) &= \lambda_k + f(\overline{\lambda}_k - \lambda_k) \end{aligned} \quad [4]$$

$f$ signifies that the fractional error of the interpolated eigenvalues is reduced by $1 - f$ relative to the empirical eigenvalues determined for $N = 10000$. We found that $f \leq 0.21$ is needed so that $\tilde{z}_{49}(x; f)$ (shown as a green line in Figure S1A) fit to $\overline{p}_2(x)$ yields accurate $\overline{S}$ on at least half the intervals in $D_{49}$ (see Figure S1G).

Given that the fractional error in estimating $\lambda_{37}, ..., \lambda_{49}$ by $\overline{\lambda}_{37}, ..., \overline{\lambda}_{49}$ is $\approx 62\%$ when $N = 10000$, how large does $N$ have to be to reduce this fractional error to $62\% \times 0.21 \approx 13\%$? A convergence rate for the fractional error is given in Theorem 1 of (2). For 2-dimensional manifolds:

$$\frac{\left|\overline{\lambda}_k - \lambda_k\right|}{\lambda_k} = O\left(\frac{(\log N)^{\frac{3}{8}}}{N^{\frac{1}{4}}}\right) \quad [5]$$

Assuming that the big-$O$ bound is sharp at $N = 10^4$ for $k = 37, ..., 49$ (i.e. the prefactor is given by $0.62\frac{10000^{\frac{1}{4}}}{\log(10000)^{\frac{3}{8}}} \approx 2.7$), we extrapolated that at least $N = 10^7$ datapoints are needed to reduce the fractional error to 13% (see Figure S1H). Equation 5 also applies to empirical eigenvalues of $\overline{\Delta}_M$ constructed from weighted kNN and $r$-neighborhood kernels instead of Gaussian kernels (see Methods Section B.6). However, the prefactor in Equation 5 is actually worse for these estimators since their empirical eigenvalues have larger fractional errors at $N = 10000$ (see Figure S1H), so that even larger $N$ would be required to attain the desired fractional error. Lastly, note that while we had analytical eigenvalues available with which to ascertain $m = 49$ as suitable, the naive approach of simply using all eigenvalues available ($m = N$), would require sample sizes that are even larger by several more orders of magnitude.

***B.5. Higher-Order Polynomial Fits.*** To verify that our inability to estimate scalar curvatures accurately was not an artifact of fitting the heat-traces to quadratic polynomials, we repeated our fitting procedure using cubic polynomials of the form, $\overline{p}_3(x) = 4\pi + \frac{2\pi}{3}\overline{S}x^2 + \overline{c}_3x^3$ (see Figure S1I-N) and quartic polynomials of the form $\overline{p}_4(x) = 4\pi + \frac{2\pi}{3}\overline{S}x^2 + \overline{c}_3x^3 + \overline{c}_4x^4$ (see Figure S1O-T), where $\overline{c}_3$ and $\overline{c}_4$ are free parameters. In order to improve the estimate $\overline{S}$, when determining $D_{1000}$ and $D_{49}$, we additionally discarded intervals for which fits to $z_{1000}(x)$ and $z_{49}(x)$ respectively, yielded non-monotonic polynomials. For both cubic and quartic polynomial fits, none of the estimates, $\overline{S}$, obtained from fitting $\overline{z}_{1000}(x)$ and $\overline{z}_{49}(x)$ to intervals in $D_{1000}$ and $D_{49}$ respectively, were accurate. Furthermore, when fitting $\tilde{z}_{49}(x; 0.21)$ using $\overline{p}_3(x)$ and $\overline{p}_4(x)$, only 26% and 13% respectively of the intervals on $D_{49}$ yielded accurate estimates. This suggests that sample sizes of $N > 10^7$ would also be required if higher-order polynomials are used.

***B.6. Estimating the Laplace-Beltrami Operator from Data.*** For $N$ points, $\{X_i\} \in \mathbb{R}^n$, sampled from $M$, we estimated $\Delta_M$ by normalizing the weight matrix $W$ (see below) using the random walk normalization (2, 3). $\overline{\Delta}_M$ constructed using this normalization converges to $\Delta_M$ when samples are drawn uniformly from the embedding of $M$ in $\mathbb{R}^n$, as was done in our analysis.

$$\begin{aligned} \overline{\Delta}_M &= \frac{4}{\epsilon}(I_N - D^{-1}W) \\ D &= \text{diag}\{W\mathbf{1}\} \end{aligned} \quad [6]$$

$I_N$ is the $N \times N$ identity matrix, $\mathbf{1} \in \mathbb{R}^N$ is a vector of ones and the kernel width, $\epsilon$, is set to match that used in Theorem 1 of (2):

$$\epsilon = \frac{(\log N)^{\frac{3}{8}}}{N^{\frac{1}{4}}} \quad [7]$$

Throughout our analysis, we used $W = W_g$, the weight matrix with entries given by a Gaussian kernel:

$$[W_g]_{i,j} = \exp\left(-\frac{\|X_i - X_j\|_2^2}{\epsilon}\right) - \delta_{i,j} \quad [8]$$

To check whether other estimators had more benign prefactors for eigenvalue convergence (see Figure S1H), we also considered the weighted kNN kernel, $W_{kNN}$, and the $r$-neighborhood kernel, $W_r$, with $r = \epsilon$ (4):

$$\begin{aligned}[W_{kNN}]_{i,j} &= [W_G]_{i,j}\left[\mathbb{1}_{kNN(j)}(i) \text{ OR } \mathbb{1}_{kNN(i)}(j)\right] \\ [W_r]_{i,j} &= \mathbb{1}_{B_{X_i}(r)}(X_j) - \delta_{i,j}\end{aligned} \quad [9]$$

$kNN(i)$ is the set of indices of the $k$-nearest neighbors of point $i$ in $\mathbb{R}^n$, $B_{X_i}(r)$ is the $n$-dimensional ball of radius $r$ centred at $X_i$, and $\mathbb{1}_A(x)$ is the indicator function for $x \in A$.

## C. Details of Extrinsic Approach to Curvature Estimation.

***C.1. Quadratic Regression on Local Neighborhoods of Data.*** Here we describe the regression model for computing the coefficients of the Second Fundamental Form, $h_{ij}^k$, at a particular point $p$. As described in the main text, after performing PCA on a neighborhood of $N_p$ points around $p$ in $\mathbb{R}^n$, each point in the neighborhood can be described in terms of $d$ tangent coordinates, $t_i$, and $n-d$ normal coordinates, $n_k$. We defer discussion of how the neighborhood is selected to Methods Section C.2.

The $n_k$s are treated as dependent variables that can be modelled as quadratic functions of the $t_i$s, which are taken to be independent variables. See Equation 10 below. Linear terms are excluded since they ought to have zero coefficients in the tangent basis. Constant terms, $C_k$, are included to account for affine shifts. Since $h_{ij}^k = h_{ji}^k$ according to Equation 5 of the main text, in practice we only consider $t_i t_j$ and $h_{ij}^k$ for $j \geq i$ so that $\mathbf{t}$ and $\mathbf{h}$ in Equation 10 have linearly independent columns, though we write the full form here for simplicity.

$$\begin{aligned}
\mathbf{n} &= \mathbf{t}\mathbf{h} + \mathcal{E} \\
\mathbf{n} &= \begin{bmatrix} n_1^{(1)} & \dots & n_{n-d}^{(1)} \\ \vdots & \ddots & \vdots \\ n_1^{(N_p)} & \dots & n_{n-d}^{(N_p)} \end{bmatrix} \\
\mathbf{t} &= \begin{bmatrix} 1 & t_1^{(1)}t_1^{(1)} & \dots & t_1^{(1)}t_d^{(1)} & t_2^{(1)}t_1^{(1)} & \dots & t_d^{(1)}t_d^{(1)} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 1 & t_1^{(N_p)}t_1^{(N_p)} & \dots & t_1^{(N_p)}t_d^{(N_p)} & t_2^{(N_p)}t_1^{(N_p)} & \dots & t_d^{(N_p)}t_d^{(N_p)} \end{bmatrix} \\
\mathbf{h} &= \begin{bmatrix} C_1 & h_{1,1}^1 & \dots & h_{1,d}^1 & h_{2,1}^1 & \dots & h_{d,d}^1 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ C_{n-d} & h_{1,1}^{n-d} & \dots & h_{1,d}^{n-d} & h_{2,1}^{n-d} & \dots & h_{d,d}^{n-d} \end{bmatrix}^T \\
\mathcal{E} &= \begin{bmatrix} \varepsilon_1^{(1)} & \dots & \varepsilon_{n-d}^{(1)} \\ \vdots & \ddots & \vdots \\ \varepsilon_1^{(N_p)} & \dots & \varepsilon_{n-d}^{(N_p)} \end{bmatrix} = \begin{bmatrix} {\varepsilon^{(1)}}^T \\ \vdots \\ {\varepsilon^{(N_p)}}^T \end{bmatrix}
\end{aligned} \quad [10]$$

Regression yields the following least-squares solution:

$$\begin{aligned}
\hat{\mathbf{h}} &= (\mathbf{t}^T\mathbf{t})^{-1}\mathbf{t}^T\mathbf{n} \\
\Sigma_\varepsilon &= \frac{(\mathbf{n} - \mathbf{t}\hat{\mathbf{h}})^T(\mathbf{n} - \mathbf{t}\hat{\mathbf{h}})}{N_p} \\
\Sigma_h &= \Sigma_\varepsilon \otimes (\mathbf{t}^T\mathbf{t})^{-1}
\end{aligned} \quad [11]$$

where $\hat{\mathbf{h}}$ is the matrix of estimates of the Second Fundamental Form, $\Sigma_\varepsilon$ is the estimated covariance structure of the residuals so that $\varepsilon^{(i)} \sim \mathcal{N}(0, \Sigma_\varepsilon)$, and $\Sigma_h$ is the covariance matrix for $\hat{\mathbf{h}}$. $\otimes$ denotes the Kronecker product. We used the *mvregress* function in MATLAB to perform this regression in our code.

When datapoints are sampled exactly from an analytical manifold, $\Sigma_\varepsilon$ measures the contribution of higher-order terms. In the limit of infinite sampling and infinitesimally small neighborhoods, $\Sigma_\varepsilon \to 0$. When observational noise is present (discussed in Methods Section D.2.2), $\Sigma_\varepsilon$ also depends on the magnitude of the noise ($\sigma$ in Equation 21).

***C.2. Selecting Local Neighborhoods for Regression.*** Here we describe the procedure for selecting a neighborhood around each point $p$ for computing the Second Fundamental Form. We adopt the simplest approach of selecting the neighborhood to be a ball of radius $r$ centred at $p$, $B_p(r)$.

If $r(p)$ is not specified, we set it according to statistical rather than geometric principles, since the geometry of the manifold may be non-trivial and unknown *a priori*. Specifically, we set $r(p)$ so that the elements in the covariance matrix, $\Sigma_h$, are upper-bounded by $\sigma_h^2$, the square of the specified target uncertainty. The largest elements in $\Sigma_h$ are the variance terms on the main diagonal, corresponding to the squares of the standard errors, $\sigma_{h_{ij}^k}$, for the coefficients $h_{ij}^k$. By inspection of Equation 11:

$$\sigma^2_{h_{ij}^k} = [\text{diag}\,\Sigma_\varepsilon]_k \left[\text{diag}\,(\mathbf{t}^T\mathbf{t})^{-1}\right]_{(ij)} \tag{12}$$

where $[\text{diag}\,\Sigma_\varepsilon]_k$ is the diagonal entry of $\Sigma_\varepsilon$ corresponding to the $k^{\text{th}}$ normal direction and $\left[\text{diag}\,(\mathbf{t}^T\mathbf{t})^{-1}\right]_{(ij)}$ is the diagonal entry in $(\mathbf{t}^T\mathbf{t})^{-1}$ for which the corresponding entry in $\mathbf{t}^T\mathbf{t}$ is $\sim \sum_l (t_i^{(l)} t_j^{(l)})^2$. Increasing $r(p)$ monotonically increases both $N_p(r)$, the number of points in $B_p(r)$, and the average magnitude of elements in $\mathbf{t}$, both of which reduce $\sigma_{h_{ij}^k}$.

To avoid sweeping $r(p)$ to find the minimum value such that $\max \sigma_{h_{ij}^k} < \sigma_h$, which is computationally expensive, for each point we instead model the dependence of $N_p(r)$ on $r$ as

$$N_p(r) \sim r^{d'} \tag{13}$$

so that

$$\sigma^2_{h_{ij}^k} \sim \frac{1}{r^{d'+4}} \tag{14}$$

To determine $d'$, $N_p(r)$ is counted at 10 log-spaced distances, $r_i$, and a line is fit to the $(\log r_i, \log N_p(r_i))$ pairs for $i \in \{2, ..., 8\}$. $r_1$ is set to be the distance from $p$ to the $\left(\frac{d(d+1)}{2} + 1\right)$-closest point to $p$ (the minimum number of points needed for regression). $r_{10}$ is set to be the distance from $p$ to the furthest point from $p$. To solve for $r$, we first guess $r_g = r_1$, perform regression on the set of points in $B_p(r_g)$ and assign $\sigma_g^2$ to be the largest diagonal entry in $\Sigma_h$. If $\left|\frac{\sigma_g}{\sigma_h} - 1\right|$ is within a desired tolerance, we set $r = r_g$, or else we update $r_g$ as shown and iterate to convergence.

$$r_g \leftarrow r_g \left(\frac{\sigma_g}{\sigma_h}\right)^{\frac{2}{d'+4}} \tag{15}$$

For large datasets, we speed up computation by only selecting $r$ in this manner for a subset of $N_{calib} \leq N$ randomly selected calibration points. All datapoints in the Voronoi cell of each calibration point are then assigned the same $r$ as the calibration point. Unless otherwise specified $N_{calib} = N$.

***C.3. Goodness-of-Fit Test for Quadratic Regression.*** For a fixed density of points, there is a fundamental trade-off between reducing uncertainty in the $h_{ij}^k$s and the validity of approximating local neighborhoods with quadratic fits. To reduce $\sigma_h$, more points must be included in the fit, but a larger neighborhood may not be well-modelled by only quadratic terms. Conversely, $\frac{d(d+1)}{2}+1$ points are sufficient to perform the regression, but there is then large uncertainty in the estimate of $h_{ij}^k$. Since our approach is to choose a neighborhood size to achieve a target $\sigma_h$, we include a companion goodness-of-fit (GOF) statistic measuring how well the neighborhood is fit by a quadratic. Namely, we use Mardia's test on the residuals from regression ($\varepsilon^{(i)}$ in Equation 10), which yields a p-value for the null hypothesis that the residuals are normally distributed (5).

When the p-values are small, the quadratic regression model is unlikely to be valid. In this case, curvatures computed using the resulting $h_{ij}^k$ may be suspect regardless of the tightness of the errorbars, and the user may want to consider increasing $\sigma_h$ to reduce the neighborhood size. However, the poor GOF may not be of concern if the length scale of interest is larger than the fluctuations in the manifold which give rise to the non-Gaussian residuals (see Methods Section C.5). Note that Mardia's test is relatively weak since it may yield false negatives for heteroskedastic residuals. This GOF measure is therefore only provided as a computationally cheap consistency check. Ideally, the density of sampled points is sufficiently high to (i) permit small $\sigma_h$ and (ii) produce GOF p-values that are uniformly distributed (consistent with the null model) and spatially uncorrelated.

***C.4. Standard Error and Bias of Scalar Curvature Estimate.*** Here we discuss how we compute the standard error, $\sigma_S$, of the estimate for $S$ and note sources of estimator bias. Since the Riemannian curvature tensor in Equation 6 of the main text is a bilinear form and the tensor contraction in Equation 7 of the main text is a straightforward sum, $\sigma_S$ can be computed using simple error propagation formulas in terms of the uncertainties from regression. Specifically, the standard error we report is the first-order approximation to the second moment of a function of random variables:

$$\sigma_S = \sqrt{J^T \Sigma_h J} \tag{16}$$

where $J = \left.\frac{\partial S}{\partial h_{ij}^k}\right|_{\hat{\mathbf{h}}}$.

It is important to note that our estimate for $S$ is biased and not normally distributed. First, the $h_{ij}^k$s are only normally distributed when the residuals ($\varepsilon^{(i)}$ in Equation 10) themselves are normally distributed. Second, even when the $h_{ij}^k$s are

normally distributed, our estimate of $S$ will not be due to its bilinear dependence on $h^k_{ij}$. Lastly, estimates for $S$ can be biased in a manifold-dependent and even position-dependent way. For instance, the analytical scalar curvature of $\mathcal{S}^2$ embedded in $\mathbb{R}^3$ is given by $S = 8(h^1_{11}h^1_{22} - h^1_{12}h^1_{21})$, with $h^1_{11} = h^1_{22} = \frac{1}{2}$ and $h^1_{12} = h^1_{21} = 0$. Numerically however, the symmetric off-diagonal terms will never be exactly 0 so $S$ will be systematically under-estimated. This is apparent in the left tail of the blue histogram in Figure 2I. In our experience, adding isotropic noise of small magnitude tends to remove the skew, presumably because then the residuals more closely match the regression assumptions (see for example Figure S2B, where the left tail disappears for $\sigma = 0.001$). Furthermore, in our examples, we observed that computed scalar curvatures were less biased when the ambient and/or manifold dimensions were large. We speculate that this is because the increased number of terms (with alternating signs) in Equations 6 and 7 of the main text leads to cancellation of errors, which is likely why the accuracy of computed scalar curvatures was higher for $\mathcal{S}^3$, $\mathcal{S}^5$ and $\mathcal{S}^7$ than $\mathcal{S}^2$, and the distribution of scalar curvatures less skewed (see Figure 2I).

***C.5. Note on Length Scales.*** Here we make three remarks regarding length scales relevant both for considering curvature theoretically and for applying our algorithm.

First, note that scalar curvature has units of inverse length squared. Therefore, scaling all the coordinates of the points on a manifold by a factor $L$, changes the scalar curvature at all points by $L^{-2}$. Thus, it is always important to contextualize the scalar curvature in terms of the global length scale associated with the manifold. For example, the scalar curvature of $\mathcal{S}^d$ with radius $R$ is $S_d(p) = \frac{d(d-1)}{R^2}$ $\forall p \in M$ (here $L = R$). In the case of the toy models shown in Figure 2, the global length scale is $L \approx 1$ (see Methods Section D.1). For the image patch dataset, a normalization is applied which places all patches on $\mathcal{S}^7$ (see Methods Section E.2), so that the global scale is again $L = 1$. For scRNAseq data, we computed scalar curvature on the datapoints after preprocessing (see Methods Section F.1), without imposing any additional scaling correction to achieve a standardized global length scale. Since other custom analyses also use these same boilerplate preprocessing steps, computing scalar curvatures in the context of the global length scale of the preprocessed data is sensible. For all the scRNAseq datasets, the global length scale happened to be $L \approx 10$ (see Methods Section F.6).

Second, since $h^k_{ij}$ is a dimension-ful quantity (which scales as $L^{-1}$), to keep the ratio of $\sigma_S$ to $S$ fixed when all coordinates are scaled by $L$, $\sigma_h$ needs to be scaled by $L^{-1}$.

Lastly, we note that our choice of $\sigma_h$ sets local length scales that are statistically rather than geometrically informed: neighborhoods are chosen to upper bound the uncertainty in estimates obtained from regression. This length scale can also be understood in terms of a bias-variance trade-off. Large length scales reduce variance but may introduce a bias if the resulting neighborhoods are larger than features on the manifold. This manifests as poor GOFs and can be corrected by finer sampling. However, for manifolds with features at different length scales (such as a golf ball, which can be treated as dimples superimposed on $\mathcal{S}^2$), neighborhoods chosen by this heuristic can also be much smaller than the feature of interest, so that fine-scale curvature fluctuations are detected (dimples) while coarser features are neglected ($\mathcal{S}^2$). Regardless, we default to this statistical approach because in general, the length scale of relevant features on a data manifold will not be uniform across the manifold or known *a priori*. However, we also provide the ability to manually set position-dependent $r(p)$ in the software to facilitate *ad hoc* computation of curvatures at any length scale of interest.

## D. Details of Toy Manifold Curvature Computations.

***D.1. Analytical Forms.*** Here we provide analytical forms for the toy manifolds shown in Figures 2 and S2.

**D.1.1. Hypersphere.** The $d$-dimensional unit hypersphere, $\mathcal{S}^d$, has intrinsic coordinates $\theta_1 \in [0, 2\pi]$, $\theta_2, ..., \theta_d \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$ and ambient coordinates in $\mathbb{R}^{d+1}$ given by:

$$x_i = \begin{cases} \prod_{j=1}^{d} \cos\theta_j, & i = 1 \\ \sin\theta_{i-1} \prod_{j=i}^{d} \cos\theta_j, & 1 < i \leq d+1 \end{cases} \quad [17]$$

Using the relations in Methods Section A, the scalar curvature is given by $S_d(p) = d(d-1)$ $\forall p \in M$. To draw samples uniformly from the embedding of $\mathcal{S}^d$ in $\mathbb{R}^{d+1}$, instead of applying rejection sampling on these intrinsic coordinates as described in Methods Section A, it is more straightforward to let $x_i \sim \mathcal{N}(0, 1)$ and scale the resulting vector $(x_1, ..., x_{d+1})$ to have unit norm.

**D.1.2. One-Sheet Hyperboloid.** The one-sheet hyperboloid, $H^2_2$, has intrinsic coordinates $\theta \in [0, 2\pi]$, $u \in \mathbb{R}$ and ambient coordinates in $\mathbb{R}^3$ given by:

$$\begin{aligned} x &= a\cos\theta\sqrt{u^2+1} \\ y &= b\sin\theta\sqrt{u^2+1} \\ z &= cu \end{aligned} \quad [18]$$

For Figure 2E,F, we used $a = b = 2$ and $c = 1$. Using the relations in Methods Section A, the scalar curvature is given by $S(z) = -\frac{2}{(5z^2+1)^2}$. To avoid edge effects in the $z$-direction, we constrained $u \in [-2, 2]$, and sampled points as described in Methods Section A until a subset of $N = 10^4$ points had $u \in [-1, 1]$. Scalar curvature was computed and visualized for these $N = 10^4$ points.

**D.1.3. Ring Torus.** The 2-dimensional ring torus, $T^2$, with $R > r$, has intrinsic coordinates $\theta, \phi \in [0, 2\pi]$ and ambient coordinates in $\mathbb{R}^3$ given by:

$$\begin{aligned} x &= (R + r\cos\theta)\cos\phi \\ y &= (R + r\cos\theta)\sin\phi \\ z &= r\sin\theta \end{aligned} \quad [19]$$

For Figure 2G,H, we used $R = 2.5$ and $r = 0.5$. Using the relations in Methods Section A, the scalar curvature is given by $S(\theta) = \frac{8\cos(\theta)}{5+\cos(\theta)}$.

**D.1.4. Hypercube.** The $m$-dimensional cube of side length $r$, $\mathcal{D}_r^m$, has intrinsic coordinates $z_1, ..., z_m \in \left[-\frac{r}{2}, \frac{r}{2}\right]$, and ambient coordinates in $\mathbb{R}^n$ for $n \geq m$ given by:

$$x_i = \begin{cases} z_i, & 1 \leq i \leq m \\ 0, & m < i \leq n \end{cases} \quad [20]$$

Using the relations in Methods Section A, the scalar curvature is given by $S(p) = 0\ \forall p \in M$.

***D.2. Practical Issues for Curvature Estimation on Real-World Datasets.*** For real-world data, small sample size is only one of the potential confounders for accurately estimating curvature. Here, we report how our algorithm fares when four other real-world confounders are applied to toy manifolds: non-uniform sampling, observational noise, large ambient dimension ($n$), and uncertainty in the manifold dimension ($d$).

**D.2.1. Non-Uniform Sampling.** We expect our approach to handle non-uniform sampling of the manifold gracefully: smaller (larger) neighborhoods will be used on densely (sparsely) sampled portions of the manifold to encapsulate the number of points needed to achieve $\sigma_h$. To computationally verify the robustness of our tool to non-uniform sampling, we constructed a toy model to roughly match the ($n$, $d$, $L$) parameters for the scRNAseq datasets explored in the paper, for which non-zero scalar curvatures seemed to appear at smaller length scales/higher densities. Specifically, we wanted to verify that non-zero scalar curvatures do not appear artifactually at specific length scales due to sharp changes in the local density of points sampled from a flat manifold. To this end, we formed a dataset with a sparse periphery and dense core by uniformly sampling $N_1 = 10^4$ points from $\mathcal{D}_{10}^3$ to establish a background density equal to 10 points per unit volume, and $N_2 = 10^3$ points from $\mathcal{D}_1^3$ to create a core density roughly equal to $10^3$ points per unit volume (see Methods Section D.1.4). We embedded these points in $\mathbb{R}^{11}$ by adding isotropic Gaussian noise with $\sigma = 0.01$ to the eight normal directions, for all datapoints.

We computed scalar curvature on this dataset for a fixed $\sigma_h$ (see Methods Section D.3) and found no significant deviation from the true value of zero in either the sparse or dense regions (see Figure S2A). We next computed scalar curvatures at three fixed length scales corresponding to the 5, 50, and 95%-ile $r$ values obtained using the specified $\sigma_h$ ($r = 0.54$, 0.90 and 1.22 respectively) and again saw no deviation from zero scalar curvature for points in either the sparse or dense region (see Figure S2A). We repeated this analysis for $N_2 = 10^4$ and again saw no deviation from zero scalar curvature, regardless of whether variable neighborhood sizes or fixed length scales ($r = 0.37$, 0.63 and 1.42 corresponding to the same percentiles) were used (see Figure S2A).

**D.2.2. Observational Noise.** Every ambient coordinate can be considered a measured observable with its own observational noise. Assuming each observable is distorted by independent, isotropic Gaussian noise with variance $\sigma^2$ (sometimes referred to as *convolutional* noise (6)), datapoints $X \in \mathbb{R}^n$ sampled from an embedded manifold $M$ are modelled by:

$$X = x + \mathcal{N}(0, \sigma I), \qquad x \in M \quad [21]$$

To study the sensitivity of our algorithm to noise, we uniformly sampled $N = 10^4$ datapoints from $\mathcal{S}^2 \in \mathbb{R}^3$, added convolutional noise with $\sigma$ ranging over several orders of magnitude, and estimated scalar curvatures using a fixed $\sigma_h$ (see Methods Section D.3). For small $\sigma$, the distribution of scalar curvatures was centred near the true value of 2, but once $\sigma$ became large ($\approx 10\%$ of $\mathcal{S}^2$'s radius), the estimated scalar curvatures approached 0 (see Figure S2B). Noise in the regression context does not change the expectation value of any estimated parameter. The apparent flattening that is observed therefore indicates that $X$ (obtained from convoluting $M$), has a geometry that is not trivially related to $M$. Certainly for $\sigma \approx 1$, $X$ does not even preserve the topology of $M$ as $\mathcal{S}^2$. From a practical perspective, it suffices to say that small convolutional noise can be handled by simple quadratic regression, while large convolutional noise obfuscates the original manifold.

These observations are consistent with literature defining a manifold's *reach* (7, 8), a noise scale beyond which noisy samples cannot be uniquely associated to a point on the noise-free manifold. When $\sigma$ exceeds the manifold's reach, the relationship between the empirical density of sampled points and the original manifold is non-trivial even for a relatively forgiving model of manifold-orthogonal noise. The *ridge* manifold (6, 9) of an empirical density has also been defined as an alternative to the unwieldy task of deconvoluting noisy samples to recover a noise-free manifold. This definition avoids the notion of a noise-free manifold altogether and instead defines manifolds as ridges, contours along which the empirical density of points is maximized.

**D.2.3. Large Ambient Dimension.** A high-dimensional dataset may have an ambient space comprised of tens of thousands of observables, i.e. $n$ is very large. Meanwhile, the underlying manifold dimension, $d$, may be small. Since convolutional noise occurs in $n$ dimensions, will a low-dimensional manifold still be discernable?

To explore this, we uniformly sampled $N = 10^4$ datapoints from $\mathcal{S}^2 \in \mathbb{R}^3$, embedded these points in $\mathbb{R}^n$ for a range of $n$ up to 100, and added convolutional noise of magnitude $\sigma = 0.01$, 0.03, and 0.05 in the $n$-dimensional ambient space. We computed curvatures for all combinations of $n$ and $\sigma$ using a fixed $\sigma_h$ (see Methods Section D.3). As $n$ or $\sigma$ increased, the algorithmically chosen neighborhood sizes, $r(p)$, expanded to include enough datapoints to maintain the desired $\sigma_h$. The distribution of estimated scalar curvatures (shown in Figure S2C) is centred on the true value of 2 for $n < 80$ and $\sigma \leq 0.05$.

However, we observed that $r$ was far less sensitive to changes in $n$ than changes in $\sigma$. For example, exploding $n$ from 3 to 100 at $\sigma = 0.01$ and tripling $\sigma$ from 0.01 to 0.03 at $n = 3$ required a comparable increase in $r$ (see Figure S2C). Therefore, consistent with the results of Methods Section D.2.2, as long as the noise scale $\sigma$ is small, a large ambient dimension $n$ is not a confounder. Practically however, to shorten computational overhead and avoid the large-$n$-and-$\sigma$ case, it is still helpful to reduce the ambient dimension by projecting datapoints to an affine subspace containing the manifold (e.g. by PCA). Such a transformation does not change the intrinsic curvature.

**D.2.4. Choice of Manifold Dimension.** The last practical consideration is accurate selection of the manifold dimension, $d$, which we have so far assumed to be known. There is no consensus on the definition of $d$ for a dataset, so various disciplines have devised different heuristics to determine $d$ in a data-driven fashion (10). From the regression perspective, any $d > 0$ corresponds to a well-defined regression problem. The choice of $d$ merely determines how local coordinates are partitioned into independent (tangent) and dependent (normal) variables. However, in our algorithm we noticed that some choices of $d$ result in excessively large $r(p)$ for a fixed $\sigma_h$. We explored this further using two toy manifolds and discovered a signature indicating that the specified manifold dimension may be incorrect.

The manifolds considered were $\mathcal{S}^3 \subset \mathbb{R}^5$ convoluted by isotropic Gaussian noise with $\sigma = 0.01$ and $\mathcal{S}^2 \times \mathcal{S}^2 \subset \mathbb{R}^6$, for which $d^*$, the true manifold dimension, is $d^* = 3$ and $d^* = 4$ respectively. We uniformly sampled $N = 10^4$ points from each manifold and estimated scalar curvatures by holding $\sigma_h$ fixed for different $d$ (see Methods Section D.3). For both manifolds, the average neighborhood size, $r$, was much larger for $d > d^*$ and $d < d^*$, than for $d = d^*$ (see Figure S2D). In the case of $\mathcal{S}^3$, for $d < d^*$, the average neighborhood size was even larger than the global length scale, $L$, of the manifold. Since neighborhood sizes are chosen to achieve a target $\sigma_h$, manually decreasing $r(p)$ is counter-productive and simply increases the uncertainty from regression above $\sigma_h$.

The large neighborhood sizes that emerged for both $d > d^*$ and $d < d^*$ can be understood in terms of the mis-assignment of normal vectors to the tangent space, or vice versa. According to Equation 12, $\sigma_{h_{ij}^k}$ increases with large variation in the normal direction ($[\operatorname{diag}\Sigma_\varepsilon]_k$), or with small variation in the tangent direction ($\left[\operatorname{diag}\left(\mathbf{t}^T\mathbf{t}\right)^{-1}\right]_{(ij)}$). When we choose $d > d^*$, we mis-attribute a normal direction with small variation $[\operatorname{diag}\Sigma_\varepsilon]_k$ as an independent variable, whereas variation along the true tangent space is $\gg [\operatorname{diag}\Sigma_\varepsilon]_k$. $r$ must therefore be increased to compensate for the lack of variation along this direction mis-classified as tangent. When $d < d^*$, we have spuriously assigned a tangent direction with large variation to be a normal direction. Since this spurious normal coordinate cannot be well-approximated as a function of tangent coordinates from which it is linearly independent, the perceived noise scale ($[\operatorname{diag}\Sigma_\varepsilon]_k$) is exaggerated so that a larger neighborhood is needed to attain $\sigma_h$.

This suggests a crude, operational definition of what constitutes an incorrect choice of $d$. When $\sigma_{h_{ij}^k}$ is large relative to the uncertainty in other coefficients, there is either too little variation along the $i^{\text{th}}$ and $j^{\text{th}}$ tangent directions, or too much variation along the $k^{\text{th}}$ normal direction. In the former case, the $i^{\text{th}}$ or $j^{\text{th}}$ tangent direction might be more appropriately classified as a normal direction ($d$ is too large and should be decreased), while in the latter case, the $k^{\text{th}}$ normal direction might be more appropriately classified as a tangent direction ($d$ is too small and should be increased). When this criterion is applied point-wise, there may be a different acceptable choice of $d$ for different parts of the manifold. When this criterion is generalized over the entire manifold, a $\sigma_h$ yielding a flat distribution of GOF p-values when the manifold dimension is specified to be $d$ will also yield a flat distribution for $d+1$ but not necessarily for $d-1$: if residuals in $n-d$ dimensions are well-modelled by a multivariate Gaussian, so too will residuals in $n-d-1$ dimensions, but not necessarily residuals in $n-d+1$ dimensions (see Figure S2D). Our observations are consistent with manifolds in literature with multiple possible manifold dimensions (like the helix manifold in (9)), and which could generally arise from non-isotropic noise or non-uniform sampling.

***D.3. Parameters for Curvature Estimation.*** For each manifold in Figure 2, we chose $\sigma_h$ so that the fraction of points with GOF p-value $\leq \alpha = 0.05$ most closely matched the null model of normally distributed residuals consistent with neighborhood sizes well-approximated by quadratic regression (see Section C.3). $\sigma_h = (0.017, 0.020, 0.028, 0.055, 0.022, 0.030)$ for $(\mathcal{S}^2, \mathcal{S}^3, \mathcal{S}^5, \mathcal{S}^7, H_2^2, T^2)$ resulted in $(7.4, 3.5, 1.4, 2.8, 7.4, 4.0)\%$ of points having GOF p-values $\leq \alpha = 0.05$. Theoretically, $\max|h_{ij}^k| = (0.5, 2, 2.5)$ for $(\mathcal{S}^d, H_2^2, T^2)$ so our choices for $\sigma_h$ result in small fractional errors in all cases.

For Figure S2A, we set $\sigma_h = (0.02, 0.01)$ for $N_2 = (10^3, 10^4)$ respectively which resulted in $(1.6, 2.5)\%$ of points having GOF p-values $\leq \alpha = 0.05$. For all other panels in Figure S2, where we were interested in ascertaining the sensitivity to different confounders, instead of minimizing uncertainty per se, we used a fixed value of $\sigma_h = 0.05$. This choice resulted in neighborhoods small enough to be well-approximated by quadratic regression, manifesting as a roughly uniform distribution of GOF p-values in all cases.

## E. Details of Image Patch Dataset and Klein Bottle Manifolds.

***E.1. Notation and Preliminaries.*** First we introduce some notation needed to describe the image patch dataset. We refer readers to (11, 12) for a more detailed exposition. Let $P$ be the space of all bivariate polynomials $p : \mathbb{R}\times\mathbb{R} \to \mathbb{R}$ with $p \in P$, $h : P \to \mathbb{R}^9$ the vectorization operator given by $h(p) = [p(-1,1), p(-1,0), p(-1,-1), p(0,1), p(0,0), p(0,-1), p(1,1), p(1,0), p(1,-1)]^T$, $u : \mathbb{R}^m \to \mathcal{S}^{m-1}$ the normalization operator given by $u(v) = \frac{v}{\|v\|_2}$, and $c : \mathbb{R}^9 \to \mathbb{R}^8$ the projection operator given by $c(y) = \Lambda A^T y$, where $A = [\mathbf{e_1} \dots \mathbf{e_8}]$, $\Lambda = \text{diag}\{\frac{1}{\|\mathbf{e_1}\|_2^2}, ..., \frac{1}{\|\mathbf{e_8}\|_2^2}\}$, and $\{\mathbf{e_i}\}$ are vectorized basis vectors for the 2-dimensional discrete cosine transform (DCT) applied to 3x3-pixel patches:

$$\begin{aligned}
\mathbf{e_1} &= [1,0,-1,1,0,-1,1,0,-1]^T/\sqrt{6} \\
\mathbf{e_2} &= [1,1,1,0,0,0,-1,-1,-1]^T/\sqrt{6} \\
\mathbf{e_3} &= [1,-2,1,1,-2,1,1,-2,1]^T/\sqrt{54} \\
\mathbf{e_4} &= [1,1,1,-2,-2,-2,1,1,1]^T/\sqrt{54} \\
\mathbf{e_5} &= [1,0,-1,0,0,0,-1,0,1]^T/\sqrt{8} \\
\mathbf{e_6} &= [1,0,-1,-2,0,2,1,0,-1]^T/\sqrt{48} \\
\mathbf{e_7} &= [1,-2,1,0,0,0,-1,2,-1]^T/\sqrt{48} \\
\mathbf{e_8} &= [1,-2,1,-2,4,-2,1,-2,1]^T/\sqrt{216}
\end{aligned} \quad [22]$$

By inspection, $\mathbf{e_1}$ is the basis vector for patches with horizontal stripes and linear gradients, $\mathbf{e_2}$ for patches with vertical stripes and linear gradients, $\mathbf{e_3}$ for patches with horizontal stripes and quadratic gradients, $\mathbf{e_4}$ for patches with vertical stripes and quadratic gradients, and $\mathbf{e_5}$ for diagonally-oriented patches with quadratic gradients. All the patches produced by the embedding $k^0$ in Equation 24 below and visualized in Figure 3B can be written as a linear combination of these 5 basis vectors. Next, note that the components in each $\mathbf{e_i}$ sum to 0, so that the projection operator, $c$, additionally serves to remove the mean. Finally, observe that the vector norm formed under $D = A\Lambda^2 A^T$ (referred to hereafter as the D-norm following (12)) measures the contrast in a 3x3-pixel patch since

$$\|v\|_D = \sqrt{v^T D v} = \frac{1}{2}\sqrt{\sum_i \sum_{j\sim i} (v_i - v_j)^2} \quad [23]$$

where $j \sim i$ refers to all vertical and horizontal neighbors, $j$, of a pixel $i$ in the preimage of $v$ under $h$. The $\mathbf{e_i}$ are normalized so that $\|\mathbf{e_i}\|_D = 1$.

***E.2. Image Dataset.*** We used the same *van Hateren* IML dataset (13) consisting of 4167 greyscale images of size 1532x1020 pixels studied by Carlsson et al. in (11) and followed the same preprocessing steps used there. In short, we applied a *log1p* transformation to all pixel values and randomly sampled $5 \times 10^3$ (possibly overlapping) 3x3-pixel patches from each image. We indexed the pixels in each patch using standard Cartesian coordinates with the middle pixel as the origin, so that log-transformed pixel values are given by $p(x,y), x \in \{-1,0,1\}, y \in \{-1,0,1\}$. We then applied $h$ to vectorize each patch $p$, and retained the high-contrast patches comprising the top quintile of D-norms for each image, resulting in $N \approx 4.2\times10^6$ datapoints. Next, we normalized these high-contrast vectorized patches using the composition $u \circ c$, resulting in a set of datapoints on $\mathcal{S}^7 \subset \mathbb{R}^8$. We determined the density of these datapoints in $\mathbb{R}^8$ using the kNN density estimator with $k = 100$, and retained the densest decile, which yielded $N \approx 4.2 \times 10^5$ datapoints. This dense subset of high-contrast normalized patches was found using topological data analysis in (11) to be a Klein bottle, $K^2 \subset \mathcal{S}^7$, and is studied in Figures 3D,I and S3B.

To generate the augmented image patch dataset used in Figures 3J and S3E,F, we first considered all $N \approx 1.3 \times 10^9$ vectorized high-contrast patches in the *van Hateren* IML dataset using the same procedure described above (each of the 4167 images yields $1530 \times 1018$ patches, of which the top 20% by D-norm are retained per image). These were normalized by $u \circ c$ as before to place them on $\mathcal{S}^7 \subset \mathbb{R}^8$. We again wanted to retain the densest decile of points, since only these have the topology of a Klein bottle. Mirroring the approach in (11) where the $k$ used in the kNN estimator was scaled with sample size, $k = 10^2$ used for $N \approx 4.2 \times 10^6$ corresponds to $k = 10^2 \times \frac{1.3\times10^9}{4.2\times10^6} \approx 3 \times 10^4$ for $N \approx 1.3 \times 10^9$. Computing $k \approx 3 \times 10^4$ neighbors for all $N \approx 1.3 \times 10^9$ points is prohibitive however. To determine a reasonable smaller value of $k$, we randomly selected $2 \times 10^4$ points from the set of $N \approx 1.3 \times 10^9$ on which to compare estimators and found that 90% of points in the densest decile as computed with $k = 3 \times 10^4$ also appeared in the densest decile computed using $k = 6 \times 10^2$. We therefore used the latter value for density estimation and retained the $N \approx 1.3 \times 10^8$ datapoints comprising the densest decile.

***E.3. Parametric Family of Klein Bottle Embeddings.*** Let $\theta, \phi \in [0, 2\pi]$. Bivariate polynomials parameterized by $(\theta, \phi)$, $k_{\theta,\phi} \in K_{\theta,\phi} \subset P$, that satisfy $k_{\theta,\phi} = k_{\theta+\pi,2\pi-\phi}$ form a Klein bottle, $K^2$: the $(\theta,\phi) \sim (\theta+\pi, 2\pi-\phi)$ similarity relation results in edges being glued together in the manner definitional of a Klein bottle's topology (shown in Figure 3B). The candidate Klein bottle embedding supplied in (11) to model image patch data satisfies the similarity relation $\forall x, y$:

$$k^0 \equiv k^0_{\theta,\phi}(x,y) = \cos\phi\,[x\cos\theta + y\sin\theta]^2 + \sin\phi\,[x\cos\theta + y\sin\theta] \quad [24]$$

Note that any $k_{\theta,\phi} \in K_{\theta,\phi}$ can be decomposed as:

$$k_{\theta,\phi} = C + \kappa_\theta + \kappa_\phi + \kappa_{\theta,\phi} \quad [25]$$

where $\kappa_\theta = \kappa_{\theta+\pi}$, $\kappa_\phi = \kappa_{2\pi-\phi}$ and $\kappa_{\theta,\phi} = \kappa_{\theta+\pi,2\pi-\phi}$. The first three terms can be understood as constant, $\theta$-dependent and $\phi$-dependent phases respectively.

We sought an embedding of the Klein bottle for which the sum of Euclidean distances from each image patch to its closest point on the embedding is minimized. To accomplish this, we constructed a parametric family of models for each of the four terms in Equation 25. The first three of these are most conveniently expressed directly in the DCT basis.

$$\begin{aligned}
(c \circ h)(C) &= N_C \sum_{i=1}^{8} \mu_i \mathbf{e_i} \\
(c \circ h)(\kappa_\theta) &= \sum_{i=1}^{8} \left[ \sum_{\substack{j=2 \\ j \text{ even}}}^{N_\theta} \beta_{i,j} \cos(j\theta) + \gamma_{i,j} \sin(j\theta) \right] \mathbf{e_i} \\
(c \circ h)(\kappa_\phi) &= \sum_{i=1}^{8} \left[ \sum_{j=1}^{N_\phi} \zeta_{i,j} \cos(j\phi) \right] \mathbf{e_i}
\end{aligned} \quad [26]$$

$N_C$ is a Boolean variable, and $N_\theta$ and $N_\phi$ control the number of terms in the inner sum for $(c \circ h)(\kappa_\theta)$ and $(c \circ h)(\kappa_\phi)$ respectively. The expression for $(c \circ h)(\kappa_\theta)$ only includes even coefficients for $\theta$ so that the similarity relation $(\theta) \sim (\theta + \pi)$ is satisfied. The expression for $(c \circ h)(\kappa_\phi)$ only includes cosine terms so that the similarity relation $(\phi) \sim (2\pi - \phi)$ is satisfied.

For $\kappa_{\theta,\phi}$, we refrained from again writing a Fourier series-like expansion because we wanted to preserve the interpretation of $\theta$ and $\phi$ as parameters controlling the orientation and gradient respectively (11). Instead, we devised the following form, which we motivate further below:

$$\kappa_{\theta,\phi}(x,y) = \sum_{l=1}^{M_\phi} \cos^l(\phi) \left[ \sum_{\substack{0 \le s,t \le M_\theta \\ 1 < s+t \text{ even}}}^{s+t \le M_\theta} \eta_{l,s,t} (x\cos\theta)^s (y\sin\theta)^t \right] + \sum_{\substack{l=1 \\ l \text{ odd}}}^{M_\phi} \sin^l(\phi) \left[ \sum_{\substack{0 \le s,t \le M_\theta \\ s+t \text{ odd}}}^{s+t \le M_\theta} \vartheta_{l,s,t} (x\cos\theta)^s (y\sin\theta)^t \right] \quad [27]$$

The expression for $(c \circ h)(\kappa_{\theta,\phi})$ is unwieldy but we note the following identities for monomials of the form $q(x,y) = x^s y^t$, which can then be applied term-wise to $\kappa_{\theta,\phi}$.

$$(c \circ h)(q) = \begin{cases}
\sqrt{6}\mathbf{e_1}, & \text{if } s = 0 \text{ and } t \text{ odd} \\
-\sqrt{6}\mathbf{e_2}, & \text{if } t = 0 \text{ and } s \text{ odd} \\
\sqrt{6}\mathbf{e_3}, & \text{if } s = 0 \text{ and } t \text{ even} \\
\sqrt{6}\mathbf{e_4}, & \text{if } t = 0 \text{ and } s \text{ even} \\
-\sqrt{8}\mathbf{e_5}, & \text{if } s \text{ odd and } t \text{ odd} \\
\frac{2\sqrt{6}}{3}\mathbf{e_1} + \frac{4\sqrt{3}}{3}\mathbf{e_6}, & \text{if } s > 0 \text{ even and } t \text{ odd} \\
-\frac{2\sqrt{6}}{3}\mathbf{e_2} - \frac{4\sqrt{3}}{3}\mathbf{e_7}, & \text{if } t > 0 \text{ even and } s \text{ odd} \\
\frac{2\sqrt{6}}{3}(\mathbf{e_3} + \mathbf{e_4} + \mathbf{e_8}), & \text{if } s > 0 \text{ even and } t > 0 \text{ even}
\end{cases} \quad [28]$$

Note that the first inner sum in Equation 27 is a linear combination of basis vectors encoding purely quadratic gradients ($\mathbf{e_3}$, $\mathbf{e_4}$, $\mathbf{e_5}$ and $\mathbf{e_8}$), weighted by even trigonometric functions of $\theta$. The prefactors on this inner sum are functions that are even in $\phi$. This inner sum and its prefactor therefore jointly satisfy the similarity relation $(\theta, \phi) \sim (\theta + \pi, 2\pi - \phi)$ by independently satisfying $(\theta) \sim (\theta + \pi)$ and $(\phi) \sim (2\pi - \phi)$. Meanwhile, the second inner sum in Equation 27 is a linear combination of basis vectors containing linear gradients ($\mathbf{e_1}$, $\mathbf{e_2}$, $\mathbf{e_6}$ and $\mathbf{e_7}$), weighted by odd trigonometric functions of $\theta$. The prefactors on this inner sum are functions that are odd in $\phi$. This inner sum and its prefactor therefore jointly satisfy the similarity relation $(\theta, \phi) \sim (\theta + \pi, 2\pi - \phi)$, by independently satisfying $(\theta) \sim -(\theta + \pi)$ and $(\phi) \sim -(2\pi - \phi)$. Since the trigonometric functions of $\theta$ are coupled to $(x, y)$, $\theta$ controls the rotation of stripes in the image patches, just as in $k^0$. Similarly, since the prefactors on the inner sums are functions of $\phi$, $\phi$ controls the relative contribution of quadratic gradients ($\mathbf{e_3}$, $\mathbf{e_4}$, $\mathbf{e_5}$ and $\mathbf{e_8}$ in the first inner sum) and linear gradients ($\mathbf{e_1}$, $\mathbf{e_2}$, $\mathbf{e_6}$ and $\mathbf{e_7}$ in the second inner sum). Lastly, the boundary conditions for $\theta$ and $\phi$ in this parameterization of $\kappa_{\theta,\phi}$, yield patches with vertical (horizontal) stripes when $\theta = 0$ ($\theta = \frac{\pi}{2}$), and linear (quadratic) gradients when $\phi = \frac{\pi}{2}, \frac{3\pi}{2}$ ($\phi = 0, \pi$) just as in $k^0$.

A Klein bottle embedding belonging to this parametric family, $k^\alpha_{\theta,\phi} \in K_{\theta,\phi}$, can therefore be specified in terms of a vector $F = [N_C, N_\theta, N_\phi, M_\theta, M_\phi]$ defining its functional form, and a corresponding coefficient vector $\alpha = [\mu_i, ..., \beta_i, ..., \gamma_i, ..., \zeta_i, ..., \eta_i, ..., \vartheta_i]$. In this parametric family of Klein bottle embeddings, $k^0$ corresponds to $F = [0, 0, 0, 2, 1]$ with $\alpha = [\eta_{1,2,0}, \eta_{1,1,1}, \eta_{1,0,2}, \vartheta_{1,0,1}, \vartheta_{1,1,0}] = [1, 2, 1, 1, 1]$. Note that since scalar curvatures are only computed on the embedding after normalization, $\alpha$ is only meaningfully defined up to a multiplicative constant.

***E.4. Associating Image Patches to a Klein Bottle Embedding.*** For a given Klein bottle embedding, $k^{\alpha}_{\theta,\phi} \in K_{\theta,\phi}$, we associated each datapoint $v_i$ (already vectorized and normalized by $u \circ c \circ h$) to the closest point on $k^{\alpha}_{\theta,\phi}$ by minimizing the Euclidean distance in $\mathbb{R}^8$:

$$(\hat{\theta}_i, \hat{\phi}_i) = \text{argmin}_{\theta,\phi} \| (u \circ c \circ h) \left( k^{\alpha}_{\theta,\phi} \right) - v_i \|_2^2 \qquad [29]$$

We solved this minimization using the *lsqnonlin* function ('StepTolerance'=1e-3, 'FunctionTolerance'=1e-6) in MATLAB, supplying initial conditions corresponding to analytical values for a point on $k^0$:

$$\begin{aligned} \hat{\theta}_i &= \arctan \frac{\mathbf{e_1}^T v_i}{-\mathbf{e_2}^T v_i} \overset{v_i \in (u \circ c \circ h)(k^0)}{=} \arctan \frac{\sin \hat{\phi}_i \sin \hat{\theta}_i}{\sin \hat{\phi}_i \cos \hat{\theta}_i} \\ \hat{\phi}_i &= \arctan \sqrt{\frac{(\mathbf{e_1}^T v_i)^2 + (\mathbf{e_2}^T v_i)^2}{\mathbf{e_3}^T v_i + \mathbf{e_4}^T v_i}} \overset{v_i \in (u \circ c \circ h)(k^0)}{=} \arctan \sqrt{\frac{\sin^2 \hat{\phi}_i}{\cos^2 \hat{\phi}_i}} \end{aligned} \qquad [30]$$

We constrained solutions to $\hat{\theta}_i \in [0, \pi]$ and $\hat{\phi}_i \in [0, 2\pi]$ according to the $(\theta, \phi)$ similarity relation.

***E.5. Optimal Klein Bottle Embedding.*** Let $k^{\hat{\alpha}}_{\theta,\phi} \in K_{\theta,\phi}$ be the Klein bottle embedding that minimizes the sum of Euclidean distances in $\mathbb{R}^8$ between each image patch and the closest point on the embedding. To determine $k^{\hat{\alpha}}_{\theta,\phi}$ given a functional form $F$, we initialized the coefficient vector $\hat{\alpha}$ to have zero entries everywhere except for the values used in $k^0$. We then iterated between optimizing for $(\hat{\theta}_i, \hat{\phi}_i)$ according to Equation 29 and for $\hat{\alpha}$ as shown below using least-squares, until convergence:

$$\hat{\alpha} = \text{argmin}_{\alpha} \sum_i \| (u \circ c \circ h) \left( k^{\alpha}_{\hat{\theta}_i, \hat{\phi}_i} \right) - v_i \|_2^2 \qquad [31]$$

$k^1 \equiv k^{\hat{\alpha}}_{\theta,\phi}$ is the optimized Klein bottle embedding corresponding to $F = [1, 10, 10, 20, 10]$, for which results are shown in Figures 3H and S3D.

***E.6. Noisy Klein Bottle Embeddings.*** The set of $N \approx 4.2 \times 10^5$ image patches was associated to $k^0$ according to the procedure described in Methods Section E.4, yielding $(\hat{\theta}_i, \hat{\phi}_i)$ values. Isotropic Gaussian noise of magnitude $s\sigma$ was added element-wise in $\mathbb{R}^9$ (prior to normalization by $u \circ c$) to $h(k^0_{\hat{\theta}_i, \hat{\phi}_i})$, where $s = \text{median}_i\{\|h(k^0_{\hat{\theta}_i, \hat{\phi}_i})\|_2\} \approx 2.451$. Figures 3F,G and S3A correspond to noise with $\sigma = 0.007$, 0.01 and 0.03.

***E.7. Parameters for Curvature Estimation.*** For all scalar curvature computations on image patch datasets and Klein bottle embeddings, we set $d = 2$ and $N_{calib} = 10^4$. Unless the neighborhoods were manually specified, we used $\sigma_h = 0.1$, which yielded a flat distribution of GOF p-values (2.5% of points reported GOF p-values $\leq \alpha = 0.05$) for the set of $N \approx 4.2 \times 10^5$ points on $k^0$ closest to the image patches (shown in Figure 3E).

**F. Details of scRNAseq Datasets.** The PBMC dataset provided by 10x Genomics is comprised of $N = 10194$ PBMCs collected from a healthy donor (14). The mouse gastrulation dataset consists of $N = 116312$ cells collected at nine 6-hour intervals from embryonic day (E)6.5 to E8.5 (15). The mouse brain dataset is a benchmark from 10x Genomics consisting of $N = 1306127$ cells collected from the cortex, hippocampus and ventricular zone of two embryonic mice sacked at E18 (16). The mouse dentate gyrus dataset consists of the subset of $N = 18213$ cells considered in (17) sourced from Dataset C of (18).

***F.1. Preprocessing.*** For the PBMC dataset, we applied standard preprocessing steps using Seurat v3.1.2 (19) with default function arguments, to extract PC projections and UMAP coordinates ourselves. Specifically, we removed cells where the percentage of transcripts corresponding to mitochondrial genes exceeded 15%, or which had fewer than 500 transcripts. This reduced the number of cells from 10194 to 9385. On this filtered set, we normalized the data (NormalizeData(normalization.method='LogNormalize', scale.factor=10000)), retained the 2000 most variable genes (FindVariableFeatures(selection.method='vst', nfeatures=2000)), and z-scored the data (ScaleData). Next, we performed linear dimensionality reduction using PCA down to 50 dimensions (RunPCA(npcs=50)) and generated UMAP coordinates for visualization (RunUMAP(dims=1:30)). For the gastrulation (brain) dataset, we did not preprocess the data ourselves but instead directly used the 50 (20) PC projections and UMAP (t-SNE) visualization coordinates provided with the dataset. Please refer to (15, 16) for additional details. For the dentate gyrus dataset, we used the standard analysis recipe suggested in the dynamo v0.96.0 (20) software package. In short, we preprocessed and normalized the data (pp.recipe_monocle(n_top_genes=2000, shared_count=30)) and performed linear dimensionality reduction to 30 PCs (tl.reduceDimension).

***F.2. Cell Type Annotations.*** For the PBMC dataset, the AddModuleScore(ctrl=5) function was used to compute the per-cell average expression of marker genes corresponding to seven different cell types (21). To prepare Figure S4A, each cell was assigned the cell type for which its average marker gene expression was the highest. Cell type annotations for the gastrulation dataset (see Figure S5A) were sourced from Figure 1C of (15). Cell type annotations for the brain dataset (see Figure S6A) are predicted labels sourced from (22). Cell type annotations for the dentate gyrus dataset (see Figure S7A) were sourced from Figure 3A of (17).

***F.3. Statistical Tests.*** Here we describe the statistical tests applied to scalar curvatures computed for the scRNAseq datasets.

**F.3.1. Spatial Precision of Errorbars.** Let $m$ be the fraction of datapoints with 95% CIs containing the scalar curvatures reported by their respective kNNs. To check whether $m$ was significantly larger than chance, we used a permutation test. We randomly assigned the kNN of each datapoint to be one of the $N$ datapoints in the dataset and computed $m$. We repeated the procedure $T = 1000$ times to generate an empirical distribution of $m$ for the null model of random neighbors. The reported p-value for each $k$ is the fraction of the $T$ trials for which $m$ was greater than the value computed for data. See Figures S4D, S5D, S6D and S7D.

**F.3.2. Sensitivity to Cell Downsampling.** To check the sensitivity of the computed scalar curvatures to the average density of cells, we discarded $f$% of cells at random from the ambient space computed using the original set of $N$ datapoints, and recomputed scalar curvatures using the same ambient dimension, manifold dimension and neighborhood sizes as for the original dataset (see Methods Section F.6). Let $m$ be the fraction of downsampled datapoints with 95% CIs containing the scalar curvatures originally reported. Since the CIs grow as $f$ increases, we checked whether $m$ was significantly larger than chance by using a permutation test. We randomly paired each of the 95% CIs computed after downsampling, to one of the scalar curvatures reported by the downsampled points for the original dataset, and computed $m$. We repeated the procedure $T = 1000$ times to generate an empirical distribution of $m$ for the null model. The reported p-value for each $f$ is the fraction of the $T$ trials for which $m$ was greater than the value computed for data. See Figures S4I, S5I, S6I and S7I.

**F.3.3. Sensitivity to Transcript Downsampling.** To check the sensitivity of the computed scalar curvatures to the capture efficiency and sequencing depth of the data, we discarded $f$% of transcripts at random from the single-cell count matrix for the PBMC dataset, then performed the same preprocessing steps described in Methods Section F.1. We recomputed scalar curvatures using the same ambient dimension, manifold dimension and neighborhood sizes as for the original dataset (see Methods Section F.6). Let $m$ be the fraction of datapoints with 95% CIs containing the scalar curvatures originally reported. To check whether $m$ was significantly larger than chance, we used a permutation test. We randomly paired each of the 95% CIs computed after downsampling transcripts, to one of the scalar curvatures computed for the original dataset, and computed $m$. We repeated the procedure $T = 1000$ times to generate an empirical distribution of $m$ for the null model. The reported p-value for each $f$ is the fraction of the $T$ trials for which $m$ was greater than the value computed for data. See Figure S4J.

**F.3.4. Pairwise Comparison of Average Scalar Curvatures Across Cell Types.** To test whether the differences in average scalar curvature between cell types were statistically significant, we used Welch's t-test. However, since the scalar curvature for each point is computed using a neighborhood of nearby points, the scalar curvatures cannot be treated as independent observations. To avoid false positives arising from an over-estimation of the true degrees of freedom, we conservatively set the degrees of freedom for each cell type to be the number of cells of that type divided by the median neighborhood population for the dataset (see Figures S4F, S5F, S6F and S7F). We used this corrected value for the degrees of freedom both when computing the standard error from the standard deviation for each cell type, and when computing the cumulative density for the resulting t-statistic. Welch's t-test requires the degrees of freedom for each sample to be at least 5 (23), so we only computed p-values for pairs of cell types for which both satisfied this requirement, and performed multiple hypothesis testing on this restricted set using the Benjamini-Hochberg procedure with FDR=0.05.

***F.4. Correlating Scalar Curvature to Gene Expression.*** Since the scalar curvature is a multi-variate function of the expression levels of all genes, we fit the scalar curvature to a linear regression model of normalized gene expression data using the *fitlm* function in MATLAB, resulting in a fitted coefficient for each gene and a corresponding p-value. For the PBMC dataset, the model was fit to the z-scores of the 2000 highly variable genes selected during preprocessing (see Methods Section F.1). For the dentate gyrus dataset, the model was fit to the normalized expression values produced by pp.recipe_monocle (see Methods Section F.1). Normalized gene expression data was not publicly available for the gastrulation and brain datasets. Multiple hypothesis testing was performed on the p-values using the Benjamini-Hochberg procedure at FDR=0.05. Though the preprocessing steps select for highly-variable genes, we noted that some genes only appeared highly-variable due to the presence of a few outliers, so we disregarded these genes irrespective of their p-values.

***F.5. RNA Velocity Analysis.*** To infer the RNA velocity vector field, we used the standard analysis recipe suggested in the dynamo package. In short, after preprocessing as described in Methods Section F.1, we inferred the parameters needed for RNA velocity analysis (tl.dynamics(model='stochastic')), computed cell velocities in the basis of the 30 PCs (tl.cell_velocities) and learned the vector field (vf.VectorField). Lastly, we computed the speed (vf.speed) and divergence (vf.divergence). We visualized the flow lines of the vector field in Figure 5F using pl.streamline_plot.

***F.6. Parameters for Curvature Estimation.*** Let the variance explained by the $i^{\text{th}}$ PC be given by $\sigma_i^2$ and the cumulative fractional variance of the first $m$ PCs by $c_m = \frac{\sum_{i=1}^{m} \sigma_i^2}{\sum_i \sigma_i^2}$. For each dataset, we selected the ambient dimension as $n = \text{argmax}_m\{c_m | c_m \leq 0.8\}$, the manifold dimension as $d = \text{argmax}_m\{c_m | c_m \leq 0.64\}$, and considered the global length scale to be $L = 3\sigma_d$. $(n, d, L) = (8, 3, 18.3), (11, 3, 19.1), (9, 5, 24.9)$ and $(6, 2, 10.2)$ for the PBMC, gastrulation, brain and dentate gyrus datasets respectively. For the first three datasets, we computed scalar curvatures for manifold dimensions $d-1$, $d$ and $d+1$. Since it is not possible to compute curvature when $d-1 = 1$, for the dentate gyrus dataset, we computed scalar curvatures for manifold dimensions $d$, $d+1$ and $d+2$. It was not always possible to select $\sigma_h$ for each dataset and manifold dimension, so that the distribution of GOF p-values was flat, according to our usual heuristic. For consistency, we therefore picked $\sigma_h$ so that $\frac{1}{3}$ of points had GOF p-values $\leq \alpha = 0.05$. For manifold dimension $(d-1, d, d+1)$, $\sigma_h = (0.031, 0.041, 0.045), (0.036, 0.044, 0.053)$ and $(0.034, 0.050, 0.055)$ for

the PBMC, gastrulation and brain datasets respectively. For the dentate gyrus dataset, $\sigma_h = (0.053, 0.049, 0.061)$ for manifold dimension $(d, d+1, d+2)$.

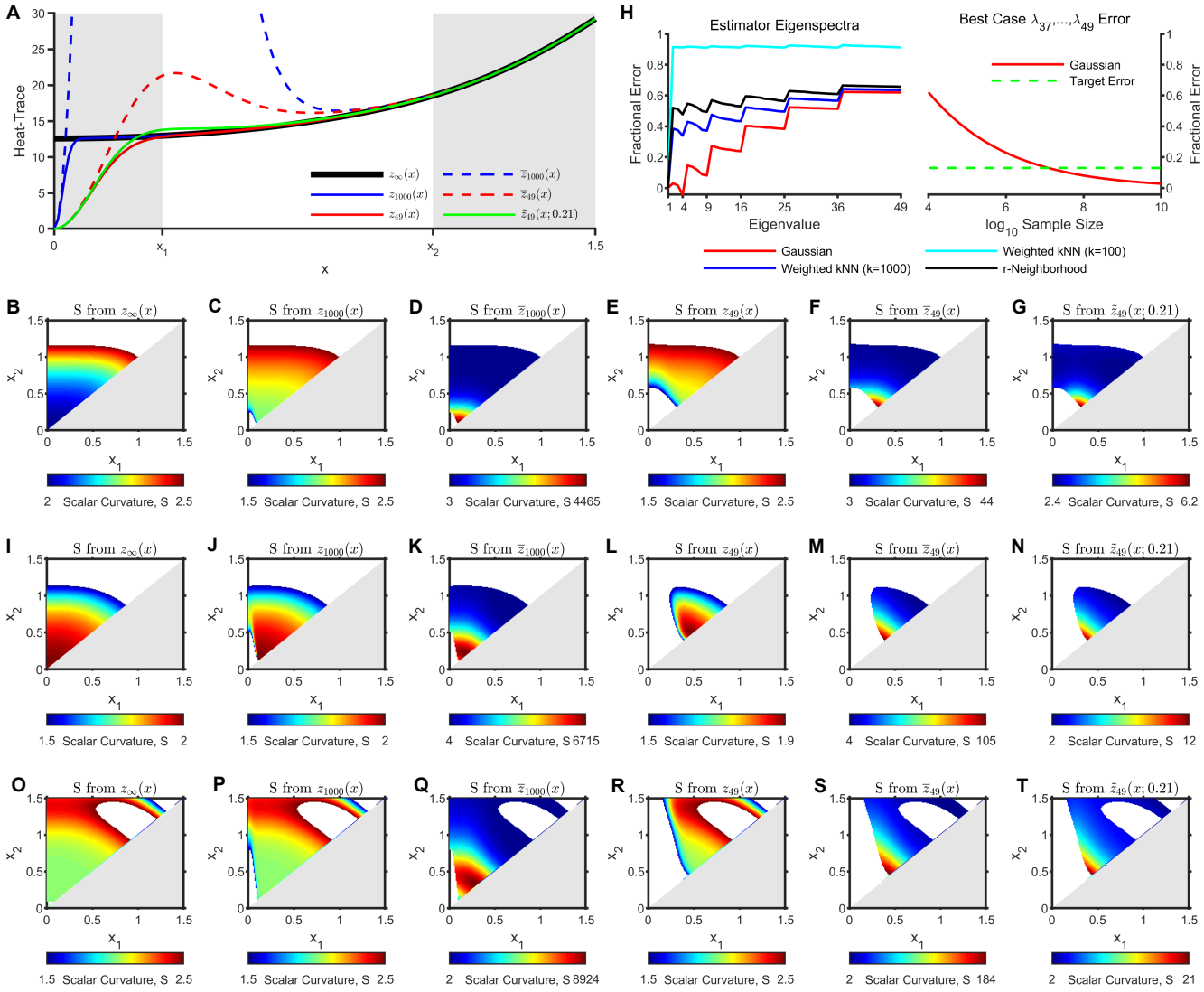
A
Heat-Trace
x
$z_\infty(x)$
$z_{1000}(x)$
$z_{49}(x)$
$\bar{z}_{1000}(x)$
$\bar{z}_{49}(x)$
$\tilde{z}_{49}(x;0.21)$
H
Estimator Eigenspectra
Fractional Error
Eigenvalue
Gaussian
Weighted kNN (k=1000)
Best Case $\lambda_{37},\ldots,\lambda_{49}$ Error
Gaussian
Target Error
$\log_{10}$ Sample Size
Weighted kNN (k=100)
r-Neighborhood
B S from $z_\infty(x)$ 2 Scalar Curvature, S 2.5
C S from $z_{1000}(x)$ 1.5 Scalar Curvature, S 2.5
D S from $\bar{z}_{1000}(x)$ 3 Scalar Curvature, S 4465
E S from $z_{49}(x)$ 1.5 Scalar Curvature, S 2.5
F S from $\bar{z}_{49}(x)$ 3 Scalar Curvature, S 44
G S from $\tilde{z}_{49}(x;0.21)$ 2.4 Scalar Curvature, S 6.2
I S from $z_\infty(x)$ 1.5 Scalar Curvature, S 2
J S from $z_{1000}(x)$ 1.5 Scalar Curvature, S 2
K S from $\bar{z}_{1000}(x)$ 4 Scalar Curvature, S 6715
L S from $z_{49}(x)$ 1.5 Scalar Curvature, S 1.9
M S from $\bar{z}_{49}(x)$ 4 Scalar Curvature, S 105
N S from $\tilde{z}_{49}(x;0.21)$ 2 Scalar Curvature, S 12
O S from $z_\infty(x)$ 1.5 Scalar Curvature, S 2.5
P S from $z_{1000}(x)$ 1.5 Scalar Curvature, S 2.5
Q S from $\bar{z}_{1000}(x)$ 2 Scalar Curvature, S 8924
R S from $z_{49}(x)$ 1.5 Scalar Curvature, S 2.5
S S from $\bar{z}_{49}(x)$ 2 Scalar Curvature, S 184
T S from $\tilde{z}_{49}(x;0.21)$ 2 Scalar Curvature, S 21

**Fig. S1**

**Fig. S1. The scalar curvature of $\mathcal{S}^2$ is poorly estimated using the Laplace-Beltrami operator.**
**(A)** The heat-trace with $m$ terms, ($z_m(x)$ in Equation 3 of the main text) is shown for $m = \infty$ (black), $m = 1000$ (solid blue) and $m = 49$ (solid red), when evaluated with analytical eigenvalues for $\mathcal{S}^2$. Empirical eigenvalues were obtained by uniformly sampling $N = 10^4$ points from $\mathcal{S}^2$ (see Figure 1A; Methods Section D.1.1) and estimating the Laplace-Beltrami (LB) operator using Equations 6-8. The heat-trace evaluated using these empirical eigenvalues, $\overline{z}_m$, is shown for $m = 1000$ (dashed blue) and $m = 49$ (dashed red). The heat-trace evaluated using eigenvalues obtained by interpolating between the analytical and empirical values ($\tilde{z}_m(x; f)$ in Equation 4) is shown for $m = 49$ and $f = 0.21$ (solid green). $f$ signifies that the fractional error of the interpolated eigenvalues is reduced by $1 - f$ relative to the empirical eigenvalues. $f = 0$ corresponds to the analytical eigenvalues while $f = 1$ corresponds to the empirical eigenvalues. The white region bounded by $[x_1, x_2]$ indicates a candidate interval over which to fit a heat-trace to a quadratic in order to extract an estimate for the scalar curvature (see Equations 2-4 in the main text; Methods Section B.1). On the one hand, since the knee of $z_m(x)$ shifts to the left as $m$ increases (i.e. $z_m(x)$ converges from $\infty$), larger $m$ results in more intervals for which $z_m(x)$ well-approximates $z_\infty(x)$ and will therefore yield accurate scalar curvature estimates. On the other hand, $\overline{z}_m(x)$ becomes a worse estimator for $z_m(x)$ as $m$ increases.
**(B)** Scalar curvatures estimated by fitting $z_\infty(x)$ to a quadratic over different intervals $[x_1, x_2]$ as defined in (A). Scalar curvatures are shown in color for intervals yielding accurate estimates ($S \in [1.5, 2.5]$). This colored region corresponds to $D_\infty$.
**(C)** As in (B) but with estimates obtained by fitting a quadratic to $z_{1000}(x)$. The colored region corresponds to $D_{1000}$. By inspection, $D_{1000} \subset D_\infty$.
**(D)** Scalar curvatures estimated by fitting $\overline{z}_{1000}(x)$ to a quadratic over each interval in $D_{1000}$. Though $D_{1000}$ was constructed using only intervals which yielded an accurate scalar curvature estimate when analytical eigenvalues were used in the heat-trace, no interval in $D_{1000}$ yields an accurate scalar curvature estimate when the same number of empirical eigenvalues are used in the heat-trace instead.
**(E)** As in (B) but with estimates obtained by fitting a quadratic to $z_{49}(x)$. The colored region corresponds to $D_{49}$. By inspection, $D_{49} \subset D_{1000}$.
**(F)** As in (D) but with estimates obtained by fitting $\overline{z}_{49}(x)$ to a quadratic over each interval in $D_{49}$. No estimate is accurate just as in (D).
**(G)** As in (F) but with estimates obtained by fitting $\tilde{z}_{49}(x; f = 0.21)$ to a quadratic over each interval in $D_{49}$. $f = 0.21$ was chosen so that half the intervals in $D_{49}$ yield an accurate scalar curvature estimate.
**(H)** (Left) The fractional error in the first 49 empirical eigenvalues of the LB estimator from (A) is shown in red. This operator was computed using the Gaussian kernel ($W_g$ in Equation 8). Eigenvalues 37-49 have a fractional error of 62%. The fractional error of the eigenvalues of LB estimators computed on the same $N = 10^4$ points but using the weighted kNN and $r$-neighborhood kernels ($W_{kNN}$ and $W_r$ respectively in Equation 9) is also plotted. Positive error indicates under-estimation. (Right) Projected fractional error for eigenvalues 37-49 of the LB estimator with Gaussian kernel computed using a larger sample size ($N$). The projection is based on the convergence rate given in Theorem 1 of (2), assuming that the big-$O$ bound is sharp at $N = 10^4$ for eigenvalues 37-49. The dashed green line corresponds to the 13% fractional error needed for scalar curvatures to be accurately estimated for half the intervals in $D_{49}$. This corresponds to $f = 0.21$ in (G) since $62\% \times f = 13\%$. For the LB estimator computed using the Gaussian kernel, achieving this fractional error requires $N \approx 10^7$. Since LB estimators computed using the other kernels have the same convergence rate but larger fractional error at $N = 10^4$, these estimators would require even larger $N$ to achieve the desired 13% fractional error. See Methods Section B.4.
**(I-N)** As in (B-G) respectively but using cubic polynomials for fitting. Only 26% of fits to $\tilde{z}_{49}(x; f = 0.21)$ yielded accurate scalar curvature estimates on $D_{49}$. See Methods Section B.5.
**(O-T)** As in (B-G) respectively but using quartic polynomials for fitting. Only 13% of fits to $\tilde{z}_{49}(x; f = 0.21)$ yielded accurate scalar curvature estimates on $D_{49}$. See Methods Section B.5.

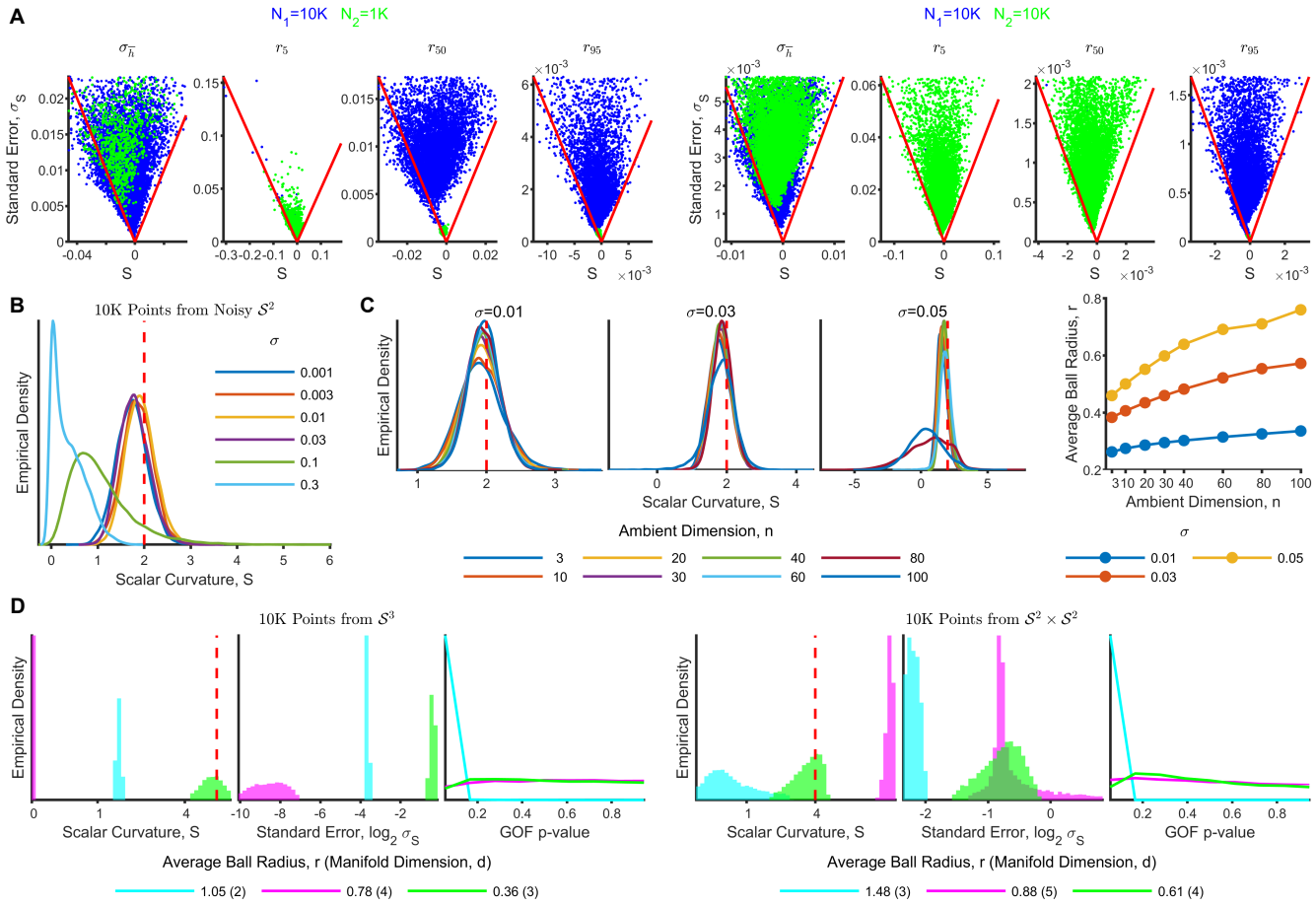


**Fig. S2. Sensitivity of algorithm to real-world confounders.**

**(A)** (Left) A dataset with a sparse periphery and a dense core was formed by uniformly sampling $N_1 = 10^4$ points from the 3-dimensional cube of side-length 10, $\mathcal{D}^3_{10}$, and $N_2 = 10^3$ points from the 3-dimensional cube of side-length 1, $\mathcal{D}^3_1$ (see Methods Section D.1.4). These points were embedded in $\mathbb{R}^{11}$ and padded with isotropic Gaussian noise of magnitude $\sigma = 0.01$ in the 8 normal directions. Scalar curvatures ($S$) were computed on this dataset of $N_1 + N_2$ points by setting $\sigma_h$ and are plotted against their standard errors ($\sigma_S$) in the leftmost panel. Curvature computations were also performed at fixed length scales corresponding to the 5, 50 and 95%-ile values for neighborhood size (left to right) used in the leftmost panel ($r = 0.54$, 0.90 and 1.22 respectively). Here, points for which the chosen $r$ led to neighborhoods with insufficient points for regression are not shown. For large length scales, all points in the dense region are able to report curvatures but are crowded into the apex of the plots. The $N_1$ ($N_2$) sparse (dense) points are shown in blue (green). Points enclosed by the red lines have 95% CIs including the true value of zero. The right four panels show analogous results when $N_2 = 10^4$. Here the the 5, 50 and 95%-ile values for neighborhood size are $r = 0.37$, 0.63 and 1.42 respectively. See Methods Section D.2.1.

**(B)** Distribution of scalar curvatures computed for $N = 10^4$ points uniformly sampled from $\mathcal{S}^2 \subset \mathbb{R}^3$ and convoluted with isotropic Gaussian noise of magnitude $\sigma$ in $\mathbb{R}^3$. Noise confounds accurate scalar curvature computation when $\sigma$ is roughly 10% of the sphere's radius. The deviation of the estimated scalar curvatures from the true value of 2 (shown as a dashed red line) for $\sigma \geq 0.1$ reflects the nontrivial geometry of a manifold convoluted by noise. See Methods Section D.2.2.

**(C)** (Left) $N = 10^4$ points were uniformly sampled from $\mathcal{S}^2$ and embedded in $\mathbb{R}^n$. Isotropic Gaussian noise of magnitude $\sigma$ was applied to each of the $n$ ambient dimensions. Scalar curvatures computed by keeping $\sigma_h$ fixed for all $n$ and $\sigma$, recapitulated the true value of 2 (shown as dashed red lines) for $n < 80$ and $\sigma \leq 0.05$. (Right) The neighborhood size ($r$) necessary to attain $\sigma_h$ is less sensitive to changes in $n$ than changes in $\sigma$. See Methods Section D.2.3.

**(D)** $N = 10^4$ points were uniformly sampled from (left) $\mathcal{S}^3 \subset \mathbb{R}^5$ convoluted with isotropic Gaussian noise in the ambient space with $\sigma = 0.01$ and (right) $\mathcal{S}^2 \times \mathcal{S}^2 \subset \mathbb{R}^6$. To investigate the effects of choosing the manifold dimension, $d$, differently than the true value, $d^*$, $\sigma_h$ was kept fixed, and scalar curvatures were computed for $d = d^* - 1$ (cyan), $d = d^* + 1$ (magenta) and $d = d^*$ (green). The panels show the distribution of (left to right) scalar curvatures ($S$), standard errors ($\sigma_S$) and GOF p-values. The true value of the scalar curvature (at $d = d^*$) is constant across both manifolds and shown as a dashed red line. The average neighborhood size ($r$ averaged over all points) is much larger for both $d = d^* - 1$ and $d = d^* + 1$ than for $d = d^*$ as shown in the legend. For the same $\sigma_h$, $d = d^* - 1$ also leads to a more skewed distribution of GOF p-values relative to $d = d^*$, while the distribution for $d = d^* + 1$ is still flat. See Methods Section D.2.4.

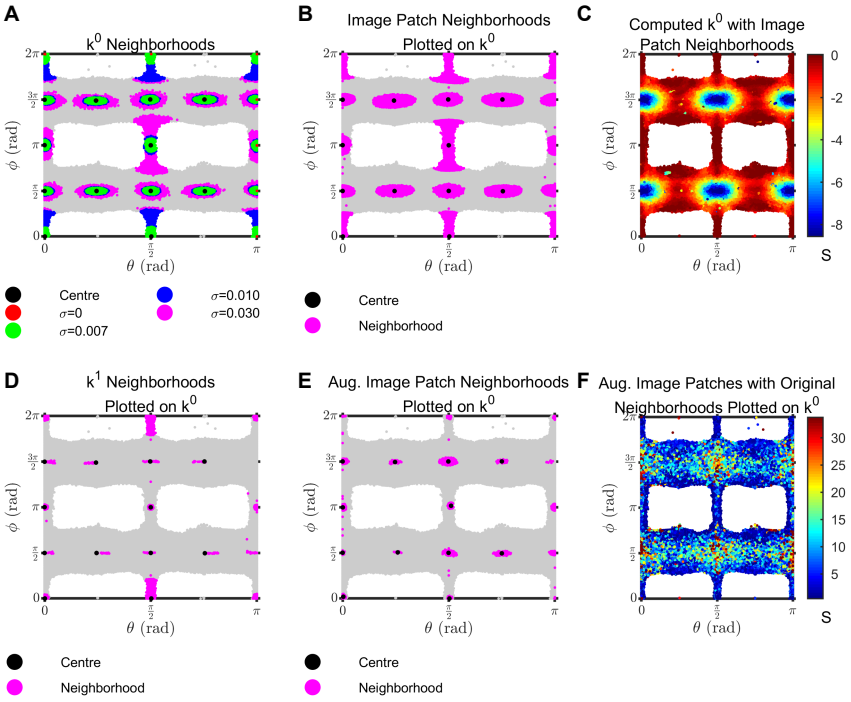


**Fig. S3. Additional details of the image patch dataset and Klein bottle embeddings (related to Figure 3).**
**(A)** To compute scalar curvatures for Figure 3E, each image patch was associated to the $(\theta_0, \phi_0)$ coordinates of the closest point on $k^0$. Here we select a handful of these associated points on $k^0$ (shown in black) and visualize how neighborhoods chosen in $\mathbb{R}^8$ to compute scalar curvatures for Figure 3E appear in $(\theta_0, \phi_0)$ coordinates (shown in red). When noise of increasing magnitude, $\sigma$, is added to the set of closest points on $k^0$ (see Figure 3F; Methods Section E.6), the neighborhood size at each point grows until $\sigma_h$ is attained.
**(B)** As in (A), but showing neighborhoods used in computing the scalar curvatures in Figure 3D for the image patch dataset. Note the close correspondence in neighborhood size with $\sigma = 0.03$ in (A).
**(C)** Scalar curvatures computed for the set of closest points $(\theta_0, \phi_0)$ on $k^0$ as in Figure 3E, but using the same neighborhood sizes determined for the image patch dataset shown in Figure 3D, some of which are visualized in (B).
**(D)** As in (A) but showing neighborhoods used in computing the scalar curvatures in Figure 3H for the set of closest points on $k^1$. Neighborhoods are visualized on $(\theta_0, \phi_0)$ coordinates instead of $(\theta_1, \phi_1)$ coordinates for ease of comparison.
**(E)** As in (B) but showing neighborhoods used in computing the scalar curvatures for the augmented image patch dataset with $N \approx 1.3 \times 10^8$ points in Figure 3J.
**(F)** Scalar curvatures computed for the augmented image patch dataset with $N \approx 1.3 \times 10^8$ points as in Figure 3J, but using the same neighborhood sizes determined for the original image patch dataset with $N \approx 4.2 \times 10^5$ shown in Figure 3D and (B). Note the close correspondence with Figure 3D.

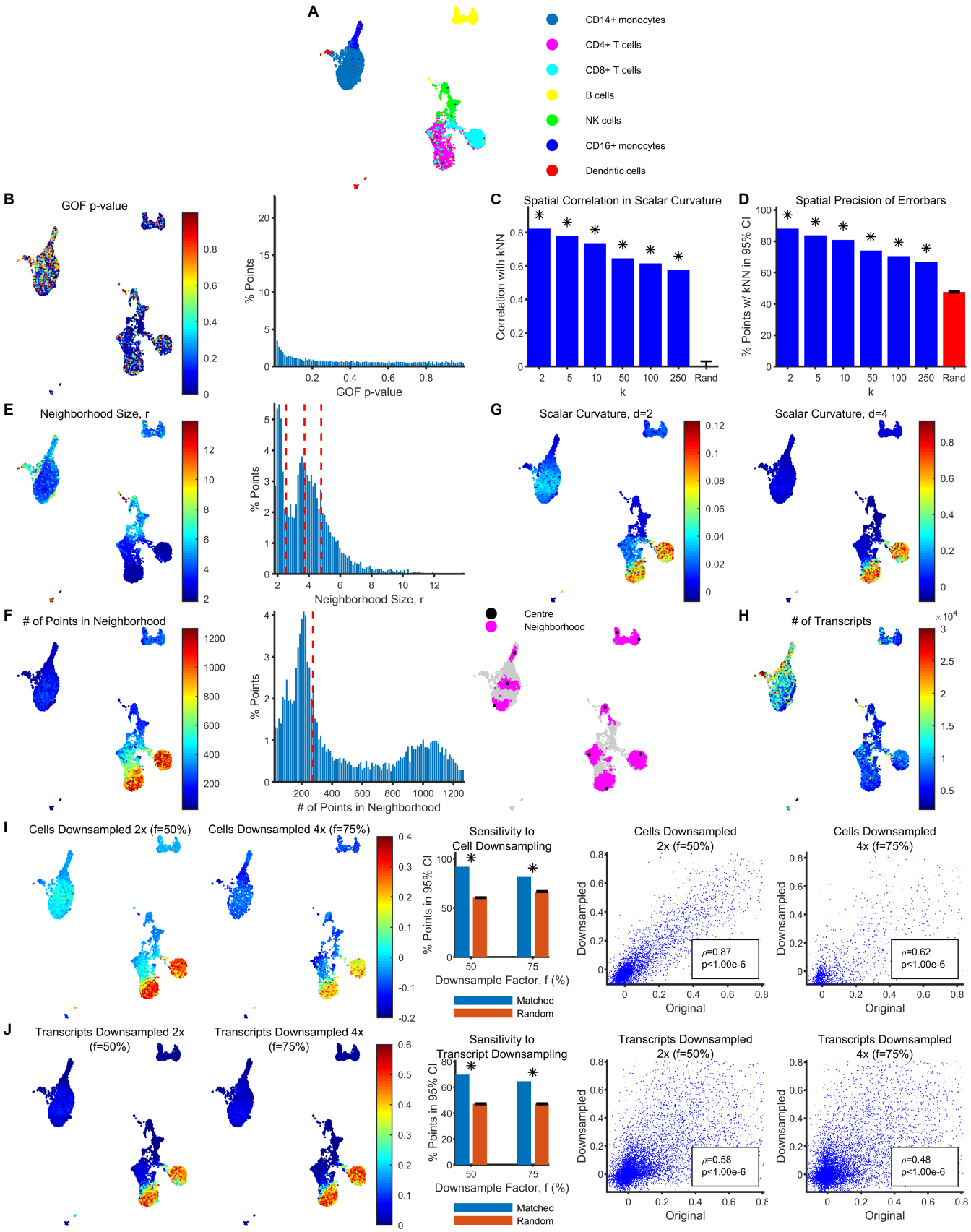
A
CD14+ monocytes
CD4+ T cells
CD8+ T cells
B cells
NK cells
CD16+ monocytes
Dendritic cells
B
GOF p-value
% Points
GOF p-value
C
Spatial Correlation in Scalar Curvature
Correlation with kNN
k
D
Spatial Precision of Errorbars
% Points w/ kNN in 95% CI
k
E
Neighborhood Size, r
% Points
Neighborhood Size, r
G
Scalar Curvature, d=2
Scalar Curvature, d=4
F
# of Points in Neighborhood
% Points
# of Points in Neighborhood
Centre
Neighborhood
H
# of Transcripts
I
Cells Downsampled 2x (f=50%)
Cells Downsampled 4x (f=75%)
Sensitivity to Cell Downsampling
% Points in 95% CI
Downsample Factor, f (%)
Matched
Random
Cells Downsampled 2x (f=50%)
Cells Downsampled 4x (f=75%)
Downsampled
Original
ρ=0.87
p<1.00e-6
ρ=0.62
p<1.00e-6
J
Transcripts Downsampled 2x (f=50%)
Transcripts Downsampled 4x (f=75%)
Sensitivity to Transcript Downsampling
% Points in 95% CI
Downsample Factor, f (%)
Matched
Random
Transcripts Downsampled 2x (f=50%)
Transcripts Downsampled 4x (f=75%)
Downsampled
Original
ρ=0.58
p<1.00e-6
ρ=0.48
p<1.00e-6

**Fig. S4**

**Fig. S4. Additional details of the PBMC scRNAseq dataset (related to Figure 4).**
**(A)** Cell types overlaid onto UMAP coordinates and sorted in decreasing order of abundance in the legend. Cells were annotated as described in Methods Section F.2.
**(B)** A goodness-of-fit p-value was computed for each point by applying Mardia's test to the residuals obtained from fitting the neighborhood around the point to a quadratic function (see Methods Section C.3). These p-values are visualized on UMAP coordinates corresponding to each point (left) and their empirical distribution is shown using a histogram (right). Small p-values suggest that the residuals are non-normal so that approximating local neighborhoods as quadratic may not be valid.
**(C)** Pearson correlation between the scalar curvature reported by each point and its $k^{\text{th}}$-nearest neighbor (kNN) for different $k$ (shown in blue). The red bar shows the mean and standard deviation of the Pearson correlation when neighbors are chosen randomly over 1000 trials ($^*p < 10^{-6}$).
**(D)** The percentage of points with 95% CIs containing the scalar curvatures reported by their respective kNNs (shown in blue). The red bar shows the mean and standard deviation of this percentage when neighbors are chosen randomly over 1000 trials ($^*p < 0.001$; see Methods Section F.3.1).
**(E)** The neighborhood size ($r$) used for computing scalar curvature at each point, overlaid onto UMAP coordinates (left) and a corresponding histogram of the empirical distribution (right). The dashed red lines correspond to the 25, 50, and 75%-ile values of $r(p)$ used for computing scalar curvatures at fixed neighborhood sizes for Figure 4C. See Methods Section C.2.
**(F)** The number of points in each neighborhood (corresponding to the neighborhood sizes in (E)) overlaid onto UMAP coordinates (left) and a corresponding histogram of the empirical distribution (middle). The dashed red line corresponds to the median neighborhood size. (Right) The set of neighbors used for computing scalar curvature (purple) is visualized on UMAP coordinates for a handful of points (black).
**(G)** Scalar curvatures were computed for manifold dimension $d - 1$ (left) and $d + 1$ (right). They are plotted here on UMAP coordinates after smoothing over the same set of $k = 250$ neighbors used in Figure 4A. See Methods Section F.6.
**(H)** The total number of transcripts observed in each cell overlaid onto UMAP coordinates.
**(I)** (Left) Scalar curvatures were computed after downsampling the number of cells in the ambient space by a factor of 2 and 4, using the same ambient dimension, manifold dimension and neighborhood sizes determined for the original dataset. They are plotted here on UMAP coordinates after smoothing over the same set of neighbors (which survive downsampling) used in Figure 4A. (Middle) The percentage of points in the downsampled datasets with a 95% CI containing the originally reported scalar curvature (blue), and likewise for a negative control obtained by randomly pairing 95% CIs and originally reported scalar curvatures for points in the downsampled dataset (red). Errorbars for the negative control are the standard deviation of this percentage over 1000 trials with different random pairings ($^*p < 0.001$). (Right) Scatter plot of the (unsmoothed) scalar curvatures computed using the full dataset shown in Figure 4B, against the (unsmoothed) scalar curvatures computed after downsampling the number of cells in the ambient space by a factor of 2 and 4. Pearson correlations are reported in the text boxes. See Methods Section F.3.2.
**(J)** (Left) Scalar curvatures were computed after downsampling the number of transcripts by a factor of 2 and 4, using the same ambient dimension, manifold dimension and neighborhood sizes determined for the original dataset. They are plotted here on UMAP coordinates after smoothing over the same set of $k = 250$ neighbors used in Figure 4A. (Middle) The percentage of points in the downsampled datasets with a 95% CI containing the originally reported scalar curvature (blue), and likewise for a negative control obtained by randomly pairing 95% CIs and originally reported scalar curvatures for points in the downsampled dataset (red). Errorbars for the negative control are the standard deviation of this percentage over 1000 trials with different random pairings ($^*p < 0.001$). (Right) Scatter plot of the (unsmoothed) scalar curvatures computed using the full dataset shown in Figure 4B, against the (unsmoothed) scalar curvatures computed after downsampling the number of transcripts by a factor of 2 and 4. Pearson correlations are reported in the text boxes. See Methods Section F.3.3.

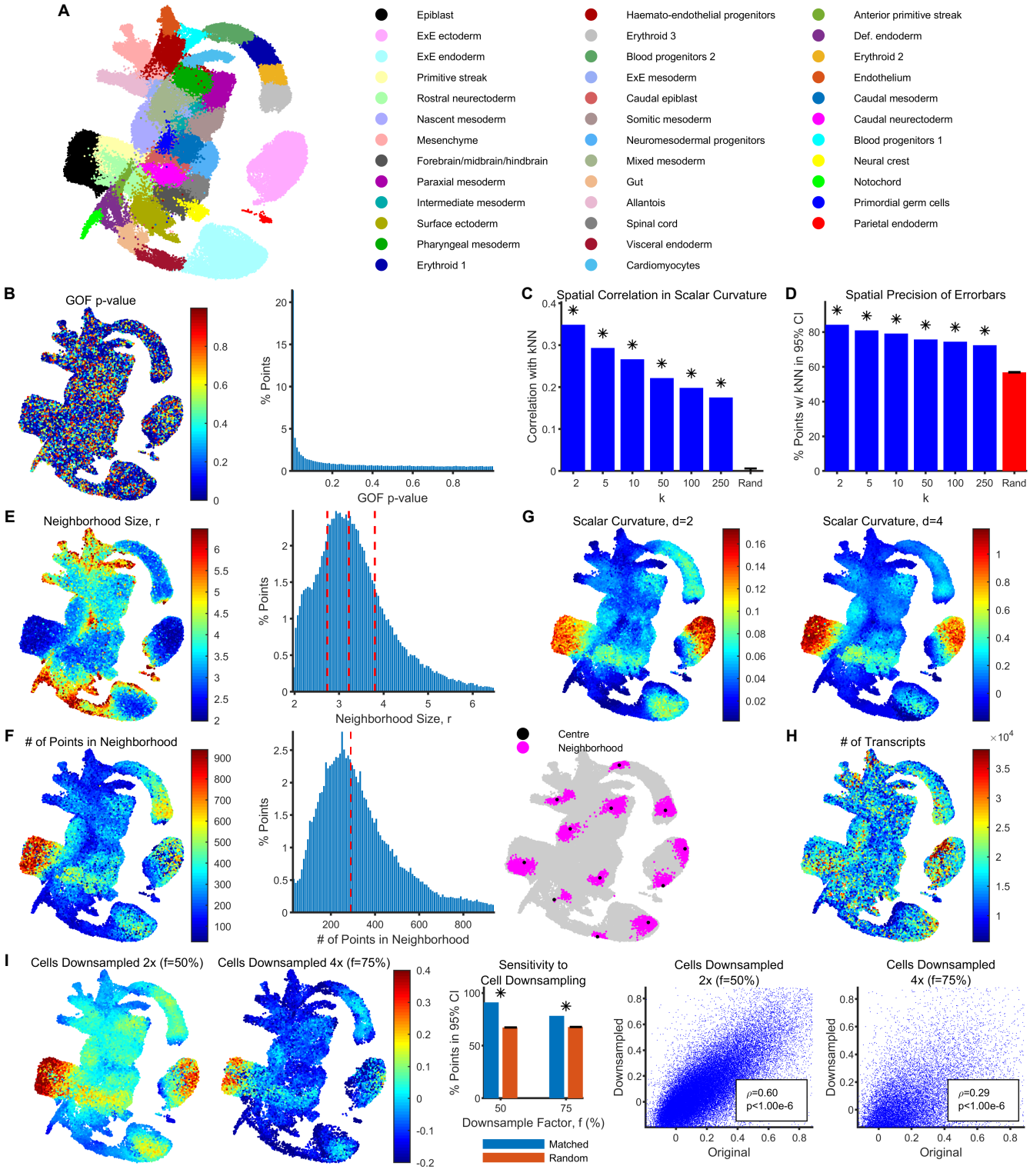


**Fig. S5. Additional details of the gastrulation scRNAseq dataset (related to Figure 4).** Panels as in Figure S4.

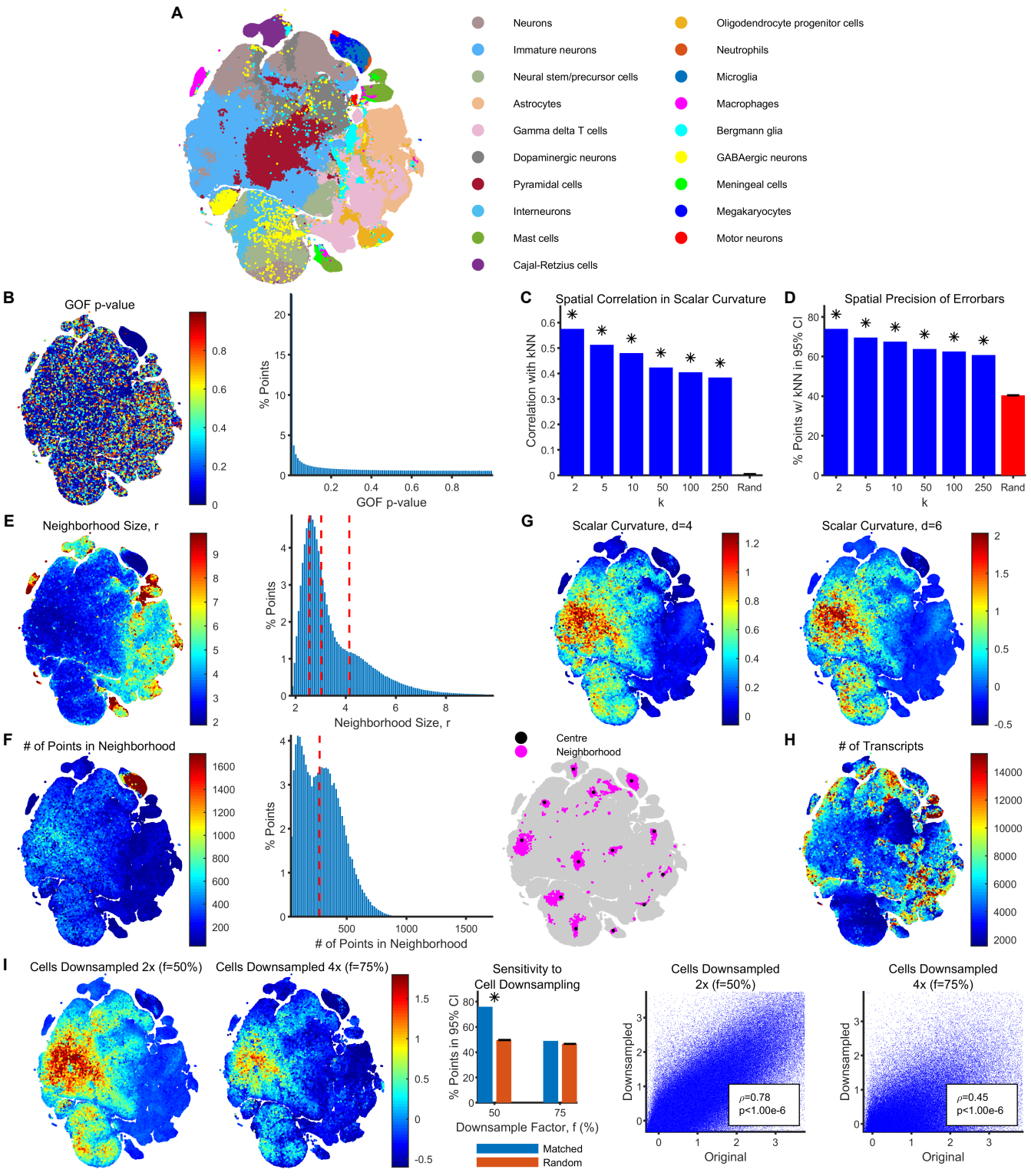


**Fig. S6. Additional details of the brain scRNAseq dataset (related to Figure 4).** Panels as in Figure S4 but with t-SNE instead of UMAP plots.

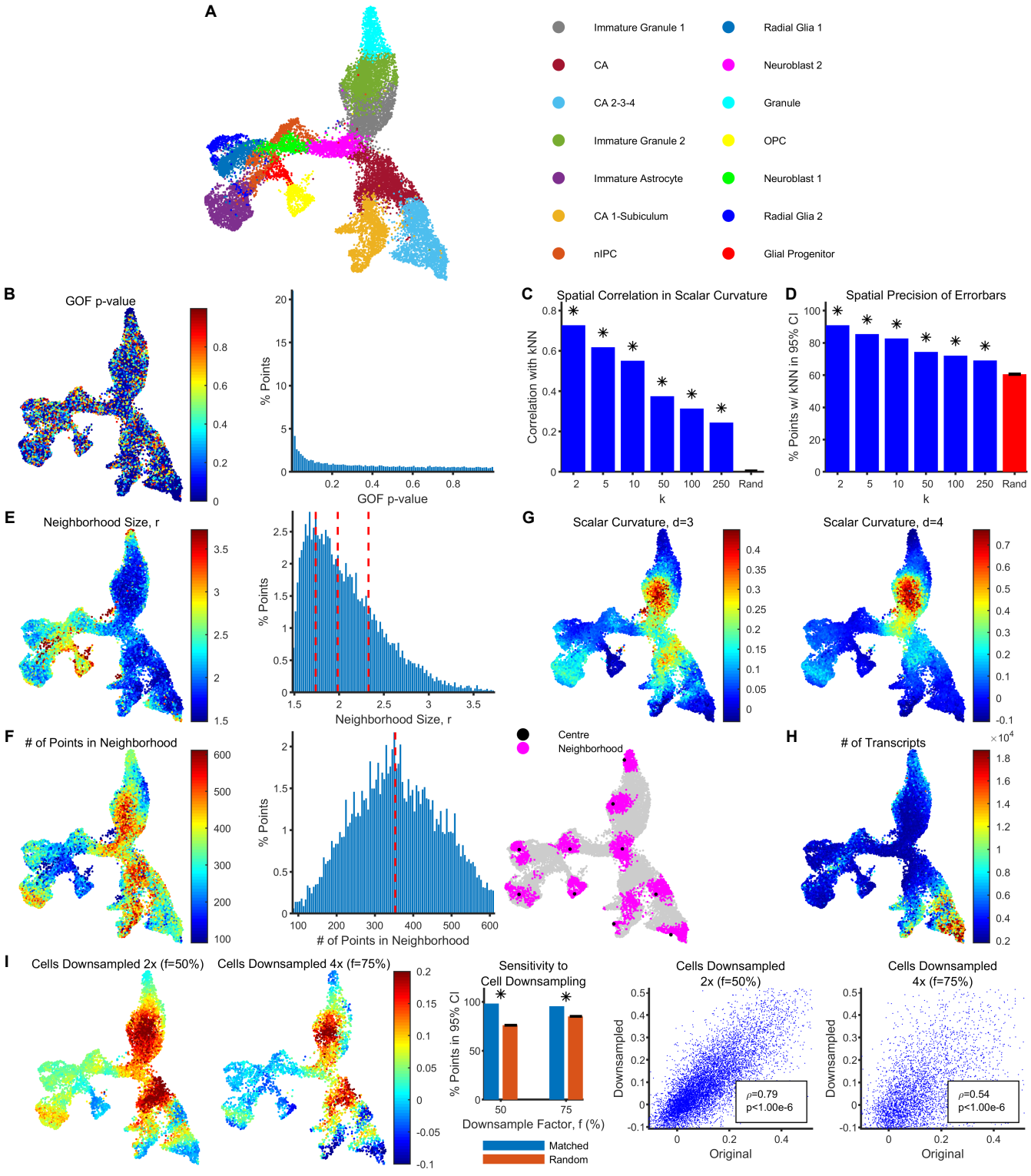


**Fig. S7. Additional details of the dentate gyrus scRNAseq dataset (related to Figure 5D-E).** Panels as in Figure S4.

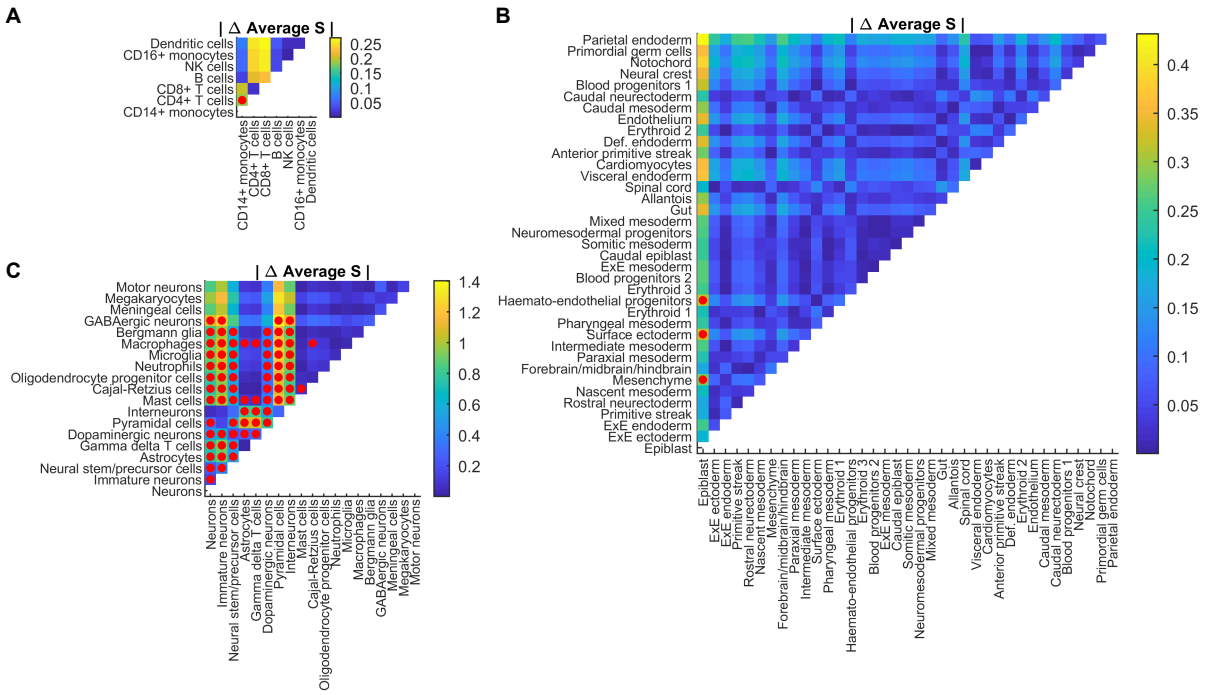


**Fig. S8. Pairwise comparisons of average scalar curvature across cell types (related to Figure 5A-C).**
**(A)** The absolute difference in average scalar curvature between cell types in the PBMC dataset, corresponding to the boxplot in Figure 5A. Entries marked with a red dot are statistically significant at FDR=0.05 (see Methods Section F.3.4).
**(B)** As in (A) but for the gastrulation dataset.
**(C)** As in (A) but for the brain dataset.

## References


1. M Reuter, FE Wolter, N Peinecke, Laplace–Beltrami spectra as 'Shape-DNA' of surfaces and solids. *Comput. Des.* **38**, 342–366 (2006).
2. NG Trillos, M Gerlach, M Hein, D Slepčev, Error estimates for spectral convergence of the graph Laplacian on random geometric graphs toward the Laplace–Beltrami operator. *Foundations Comput. Math.* **20**, 827–887 (2020).
3. M Hein, JY Audibert, U von Luxburg, Graph Laplacians and their convergence on random neighborhood graphs. *J. Mach. Learn. Res.* **8**, 1325–1370 (2007).
4. D Ting, L Huang, M Jordan, An analysis of the convergence of graph Laplacians. *arXiv* (2011).
5. KV Mardia, Measures of multivariate skewness and kurtosis with applications. *Biometrika* **57**, 519–530 (1970).
6. CR Genovese, M Perone-Pacifico, I Verdinelli, L Wasserman, Nonparametric ridge estimation. *The Annals Stat.* **42**, 1511–1545 (2014).
7. H Federer, Curvature measures. *Transactions Am. Math. Soc.* **93**, 418–418 (1959).
8. P Niyogi, S Smale, S Weinberger, Finding the homology of submanifolds with high confidence from random samples. *Discret. & Comput. Geom.* **39**, 419–441 (2008).
9. U Ozertem, D Erdogmus, Locally defined principal curves and surfaces. *J. Mach. Learn. Res.* **12**, 1249–1286 (2011).
10. P Campadelli, E Casiraghi, C Ceruti, A Rozza, Intrinsic dimension estimation: Relevant techniques and a benchmark framework. *Math. Probl. Eng.* **2015**, 1–21 (2015).
11. G Carlsson, T Ishkhanov, V De Silva, A Zomorodian, On the local behavior of spaces of natural images. *Int. J. Comput. Vis.* **76**, 1–12 (2008).
12. AB Lee, KS Pedersen, D Mumford, The nonlinear statistics of high-contrast patches in natural images. *Int. J. Comput. Vis.* **54**, 83–103 (2003).
13. JH van Hateren, A van der Schaaf, Independent component filters of natural images compared with simple cells in primary visual cortex. *Proceedings: Biol. Sci.* **265**, 359–366 (1998).
14. 10x Genomics, PBMCs from a Healthy Donor: Whole Transcriptome Analysis (https://support.10xgenomics.com/single-cell-gene-expression/datasets/4.0.0/Parent_NGSC3_DI_PBMC) (2020).
15. B Pijuan-Sala, et al., A single-cell molecular map of mouse gastrulation and early organogenesis. *Nature* **566**, 490–495 (2019).
16. 10x Genomics, 1.3 Million Brain Cells from E18 Mice (https://support.10xgenomics.com/single-cell-gene-expression/datasets/1.3.0/1M_neurons) (2017).
17. G La Manno, et al., RNA velocity of single cells. *Nature* **560**, 494–498 (2018).
18. H Hochgerner, A Zeisel, P Lönnerberg, S Linnarsson, Conserved properties of dentate gyrus neurogenesis across postnatal development revealed by single-cell RNA sequencing. *Nat. Neurosci.* **21**, 290–299 (2018).
19. A Butler, P Hoffman, P Smibert, E Papalexi, R Satija, Integrating single-cell transcriptomic data across different conditions, technologies, and species. *Nat. Biotechnol.* **36**, 411–420 (2018).
20. X Qiu, et al., Mapping transcriptomic vector fields of single cells. *bioRxiv* (2021).
21. Y Hu, et al., Single-cell transcriptome mapping identifies common and cell-type specific genes affected by acute delta9-tetrahydrocannabinol in humans. *Sci. Reports* **10**, 1–14 (2020).
22. K Xie, Y Huang, F Zeng, Z Liu, T Chen, scAIDE: clustering of large-scale single-cell RNA-seq data reveals putative and rare cell types. *NAR Genomics Bioinforma.* **2** (2020).
23. DS Starnes, DS Moore, D Yates, *The Practice of Statistics.* (W.H. Freeman and Company, London), 3rd edition, p. 792 (2008).