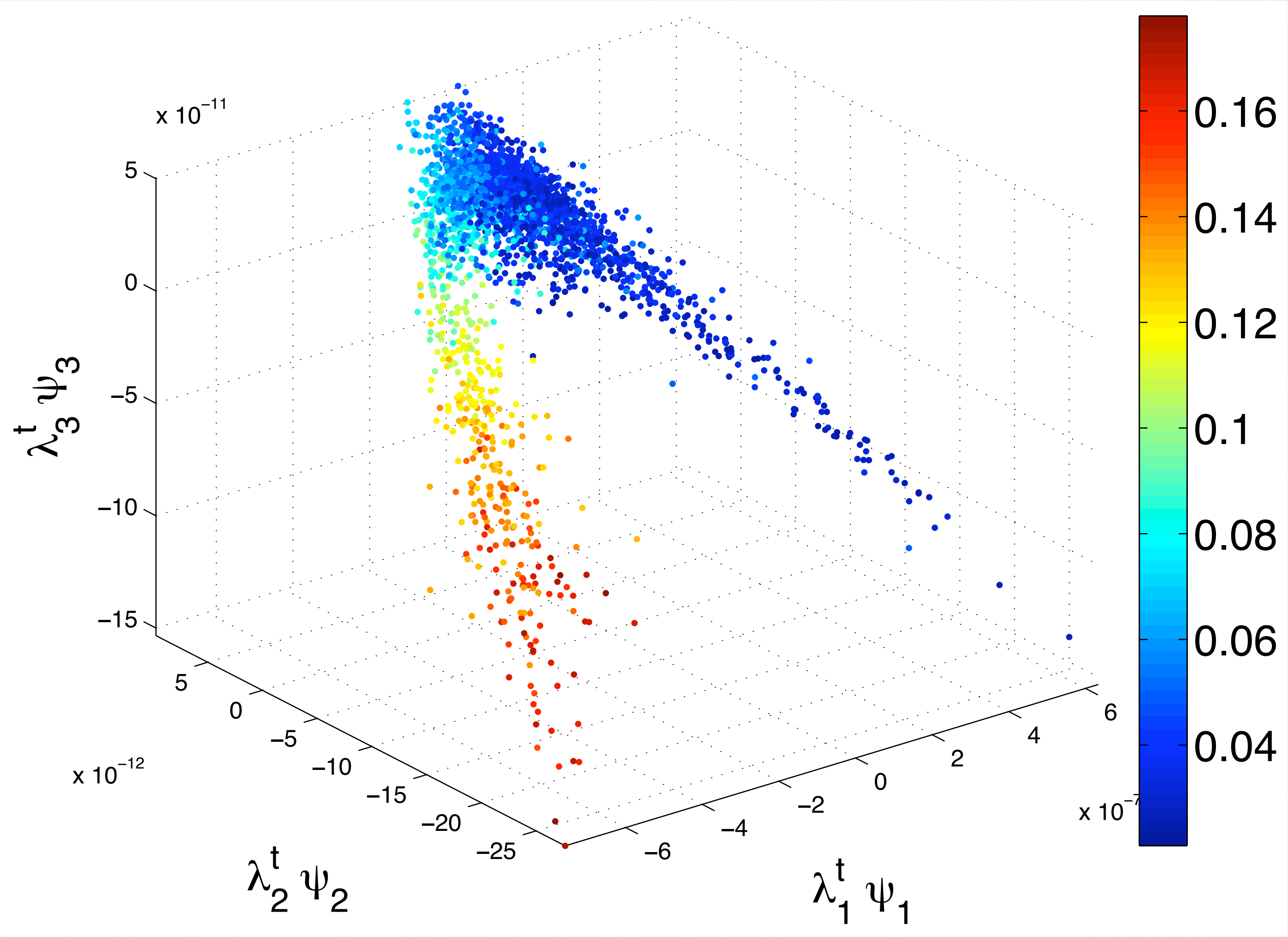
Nonlinear Data Transformation with Diffusion Map

Peter Freeman
Ann Lee
Joey Richards*
Chad Schafer

Department of Statistics
Carnegie Mellon University
www.incagroup.org

* now at U.C. Berkeley

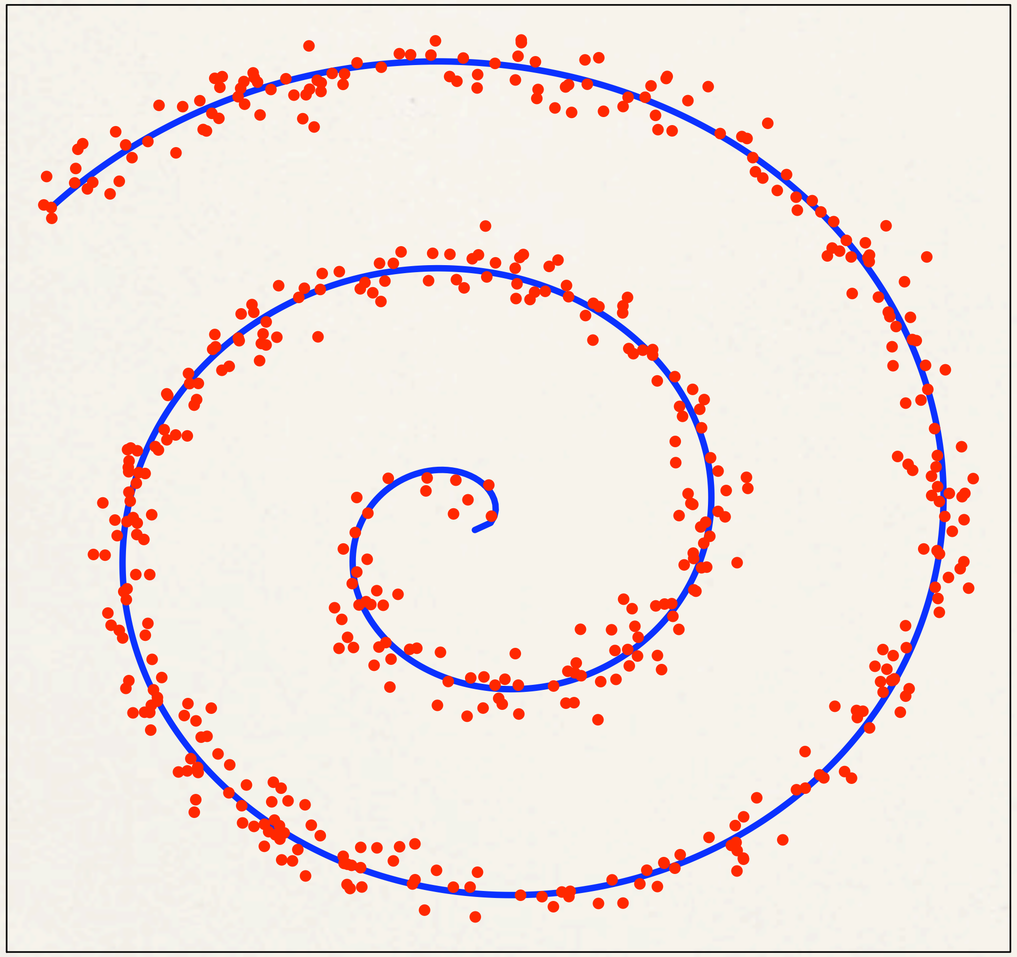


Richards et al. (2009; ApJ 691, 32)

Data Transformation: Why Do It?

- **The Problem:** Astronomical data that inhabit complex structures in (high-)dimensional spaces are difficult to analyze using standard statistical methods. For instance, we may want to:
  - Estimate photometric redshifts from galaxy colors
  - Estimate galaxy parameters (age, metallicity, etc.) from galaxy spectra
  - Classify supernovae using irregularly spaced photometric observations
- **The Solution:** If these data possess a simpler underlying geometry in the original data space, we transform the data so as to capture and exploit that geometry.
  - Usually (but not always), transforming the data affects dimensionality reduction, mitigating the "curse of dimensionality."
  - We seek to transform the data in such a way as to preserve relevant physical information whose variation is apparent in the original data.

Data Transformation: Example



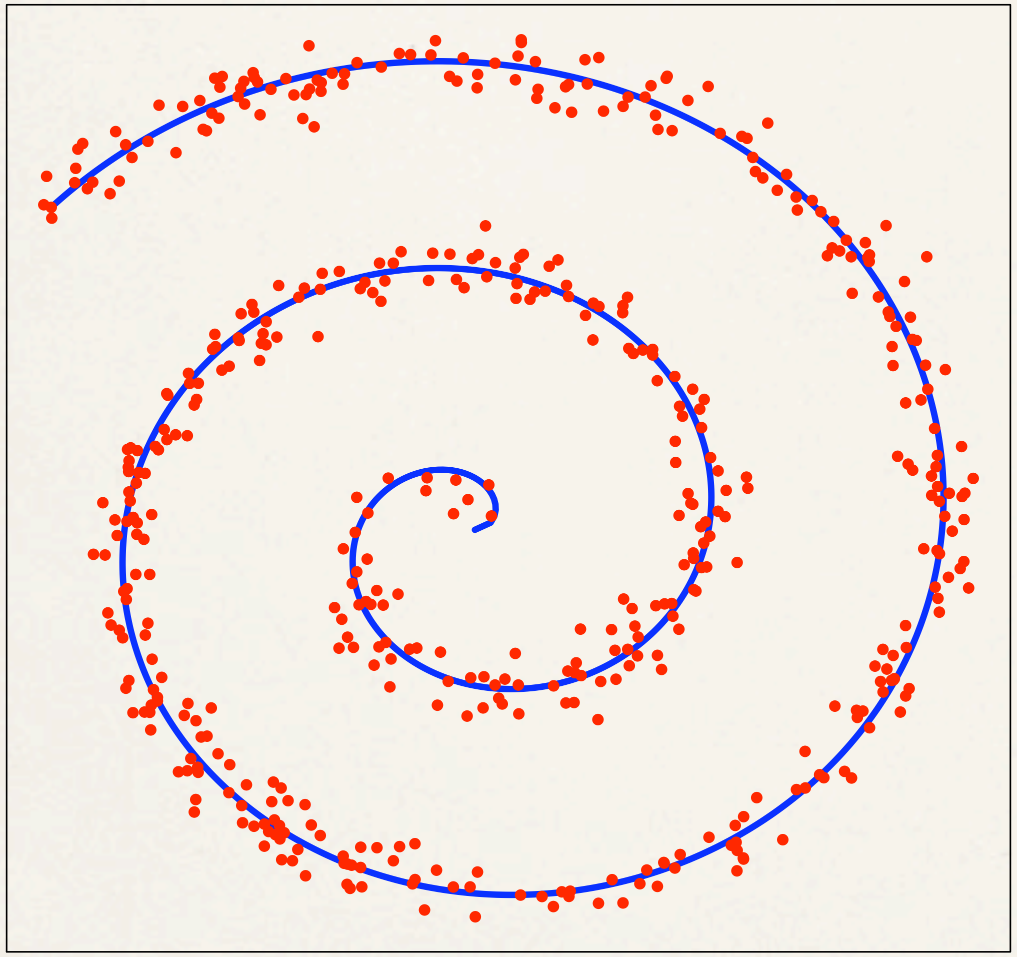
Note that these may be *non-standard* data (e.g., each data point may represent a vector of values, like a spectrum).

These data inhabit a one-dimensional manifold in a two-dimensional space.

Perhaps a physical parameter of interest (e.g., redshift) varies smoothly along the manifold.

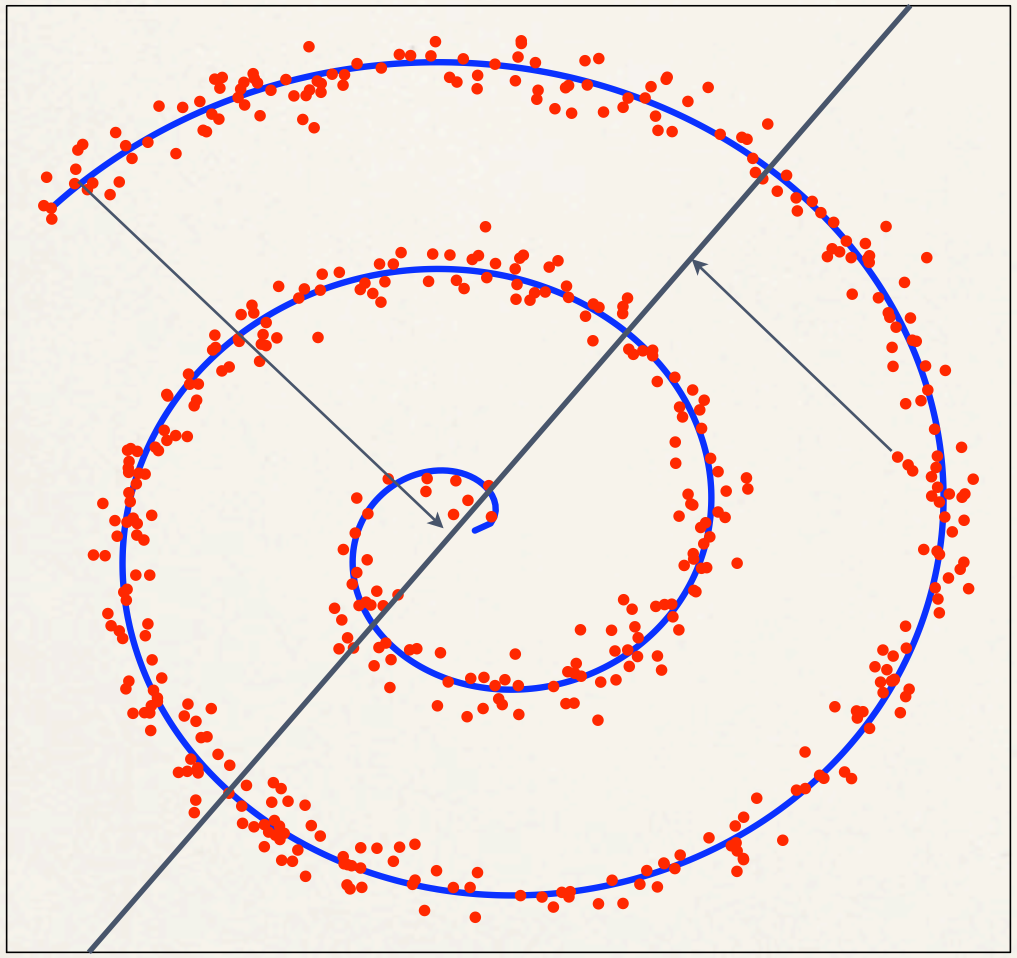
We want to transform the data in such a way that we can employ simple statistics (e.g., linear regression) to model the variation of that physical parameter. (Accurately.)

The Classic Choice: PCA



Principal components analysis will do a terrible job (at dimension reduction) in this instance because it is a *linear* transformer.

The Classic Choice: PCA



Principal components analysis will do a terrible job (at dimension reduction) in this instance because it is a *linear* transformer.

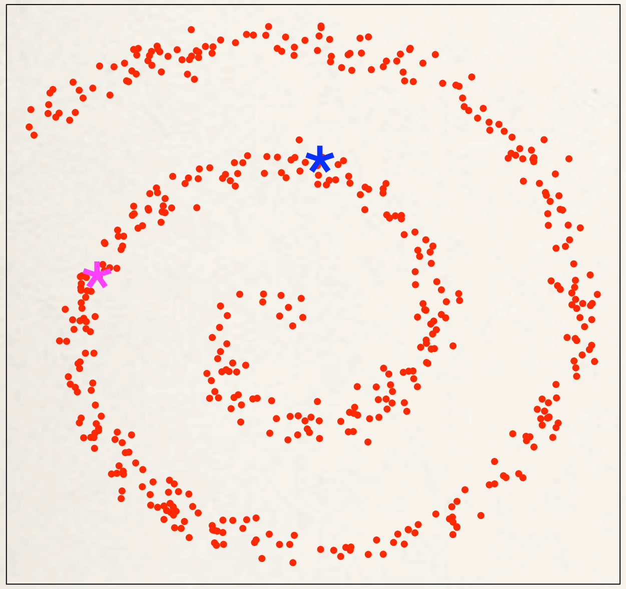
In PCA, high-dimensional data are projected onto hyperplanes. Physical information may not be well-preserved in the transformation.

Nonlinear Data Transformation

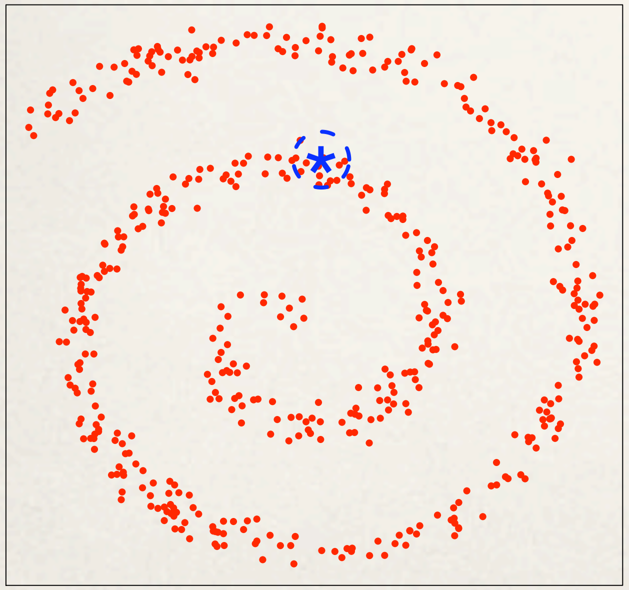
- There are many available methods for nonlinear data transformation which have yet to be widely applied to astronomical data:
  - Local linear embedding (LLE; see, e.g., Vanderplas & Connolly 2009)
  - Others: Laplacian eigenmaps, Hessian eigenmaps, LTSA
- We apply the **diffusion map** (Coifman & Lafon 2006, Lafon & Lee 2006; see **diffusionMap R** package).
  - **The Idea:** to estimate the "true" distance between two data points via a fictive diffusion (i.e., Markov random walk) process.
  - **The Advantage:** The Euclidean distance between points x and y in the space of transformed data is approximately the diffusion distance between those points in the original data space. Thus variations of physical parameters along the original manifold are approximately preserved in the new data space.

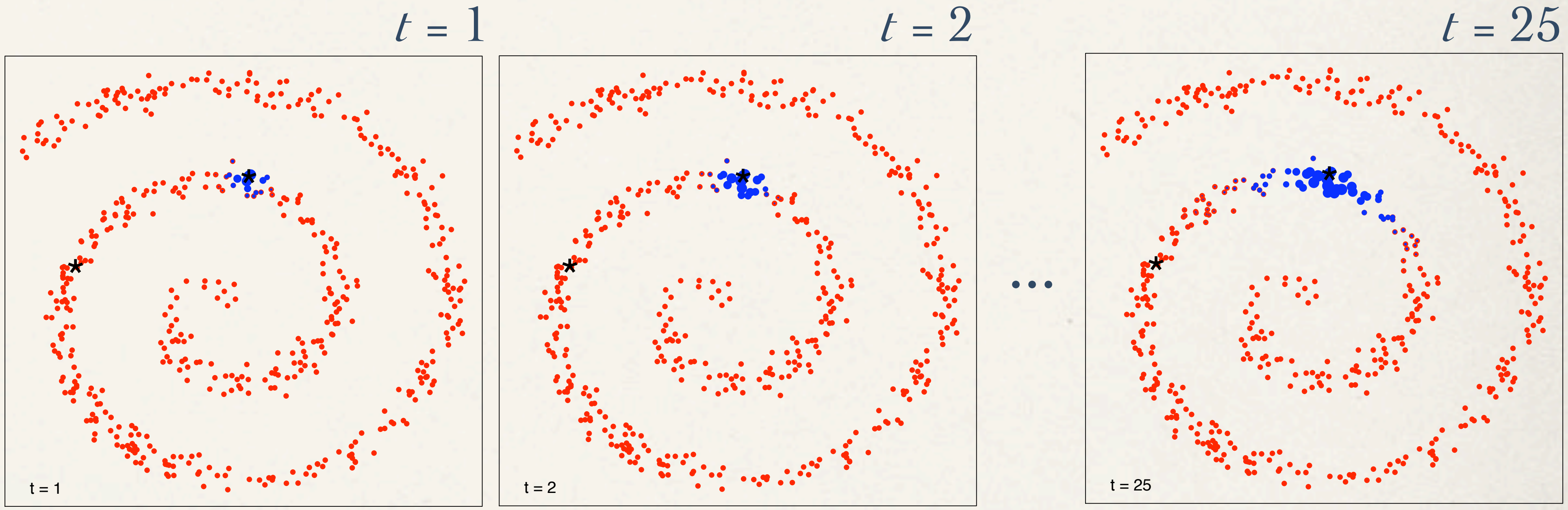
Diffusion Map: Intuition

Pick location...



...set up a kernel...




t = 1

t = 2

...

t = 25

...and map out the random walk.

Diffusion Map: The Math (Part I)

- Define similarity measure between two points *x* and *y*, e.g., the Euclidean distance:

$$s(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{\sum_{i=1}^{p} \left(c_{\boldsymbol{x},i} - c_{\boldsymbol{y},i}\right)^2}$$

- Construct a weighted graph:

$$w(\boldsymbol{x}, \boldsymbol{y}) \;=\; \exp\left(-\frac{s(\boldsymbol{x}, \boldsymbol{y})^2}{\epsilon}\right)$$

- Row-normalize to compute “one-step” probabilities:

$$p_1(\boldsymbol{x}, \boldsymbol{y}) \;=\; w(\boldsymbol{x}, \boldsymbol{y}) / \textstyle\sum_z w(\boldsymbol{x}, \boldsymbol{z})$$

- Use *$p_1(x,y)$* to populate *n* x *n* matrix *P* of one-step probabilities.

Diffusion Map: The Math (Part II)

- The probability of stepping from *x* to *y* in *t* steps is P^t.
- The diffusion distance between *x* and *y* at time *t* is

$$D_t^2(\boldsymbol{x}, \boldsymbol{y}) = \sum_{j=1}^{\infty} \lambda_j^{2t} (\boldsymbol{\psi}_j(\boldsymbol{x}) - \boldsymbol{\psi}_j(\boldsymbol{y}))^2$$

λ_j = j^{th} largest eigenvalue of *P*

ψ_j = j^{th} (right) eigenvector of *P*

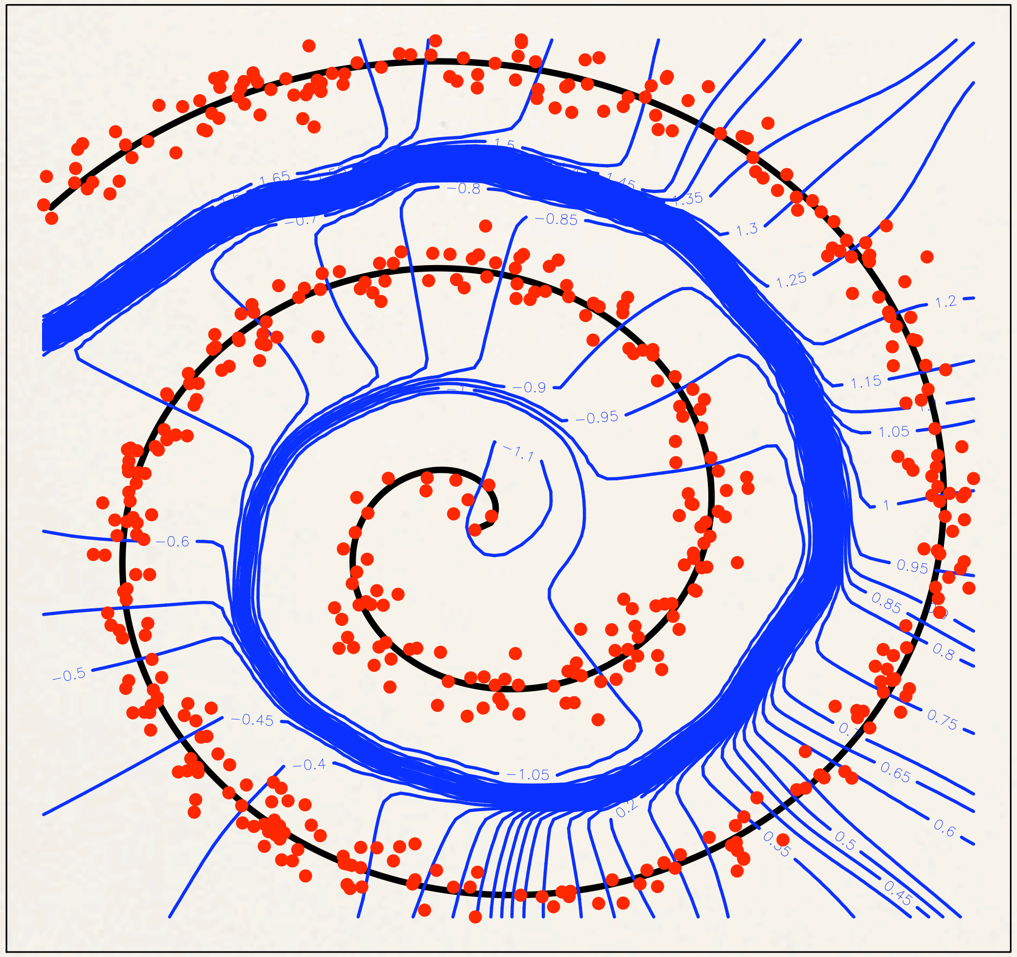
- Retain the “top” *m* eigenmodes to create diffusion map:

$$\boldsymbol{\Psi}_t : \boldsymbol{x} \mapsto [\lambda_1^t \boldsymbol{\psi}_1(\boldsymbol{x}), \lambda_2^t \boldsymbol{\psi}_2(\boldsymbol{x}), \ldots, \lambda_m^t \boldsymbol{\psi}_m(\boldsymbol{x})]$$

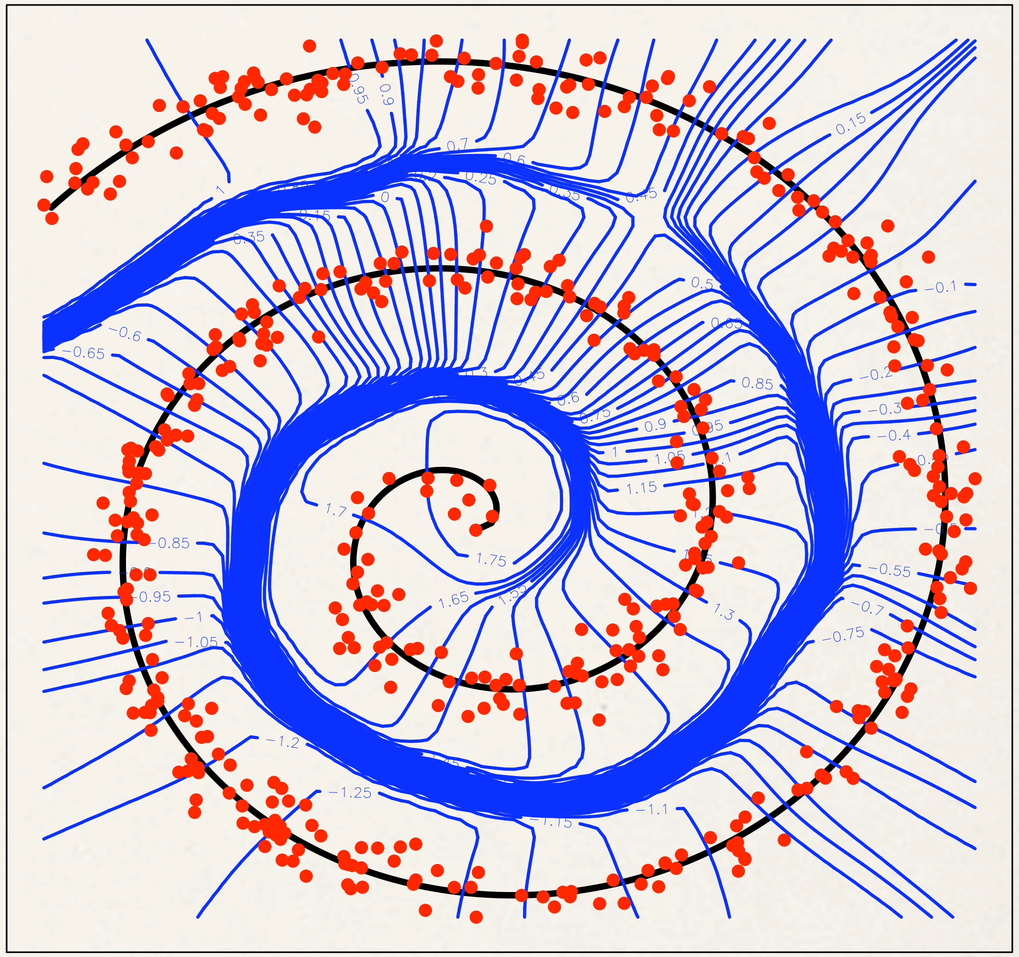
$$D_t^2(\boldsymbol{x}, \boldsymbol{y}) \simeq \sum_{j=1}^{m} \lambda_j^{2t} (\boldsymbol{\psi}_j(\boldsymbol{x}) - \boldsymbol{\psi}_j(\boldsymbol{y}))^2 \; = \; ||\boldsymbol{\Psi}_t(\boldsymbol{x}) - \boldsymbol{\Psi}_t(\boldsymbol{y})||^2$$

- The tuning parameters ε and *m* are determined by minimizing predictive risk (a topic I will skip over in the interests of time). The choice of *t* generally does not matter.

The Spiral, Redux



The first diffusion coordinate



The second diffusion coordinate

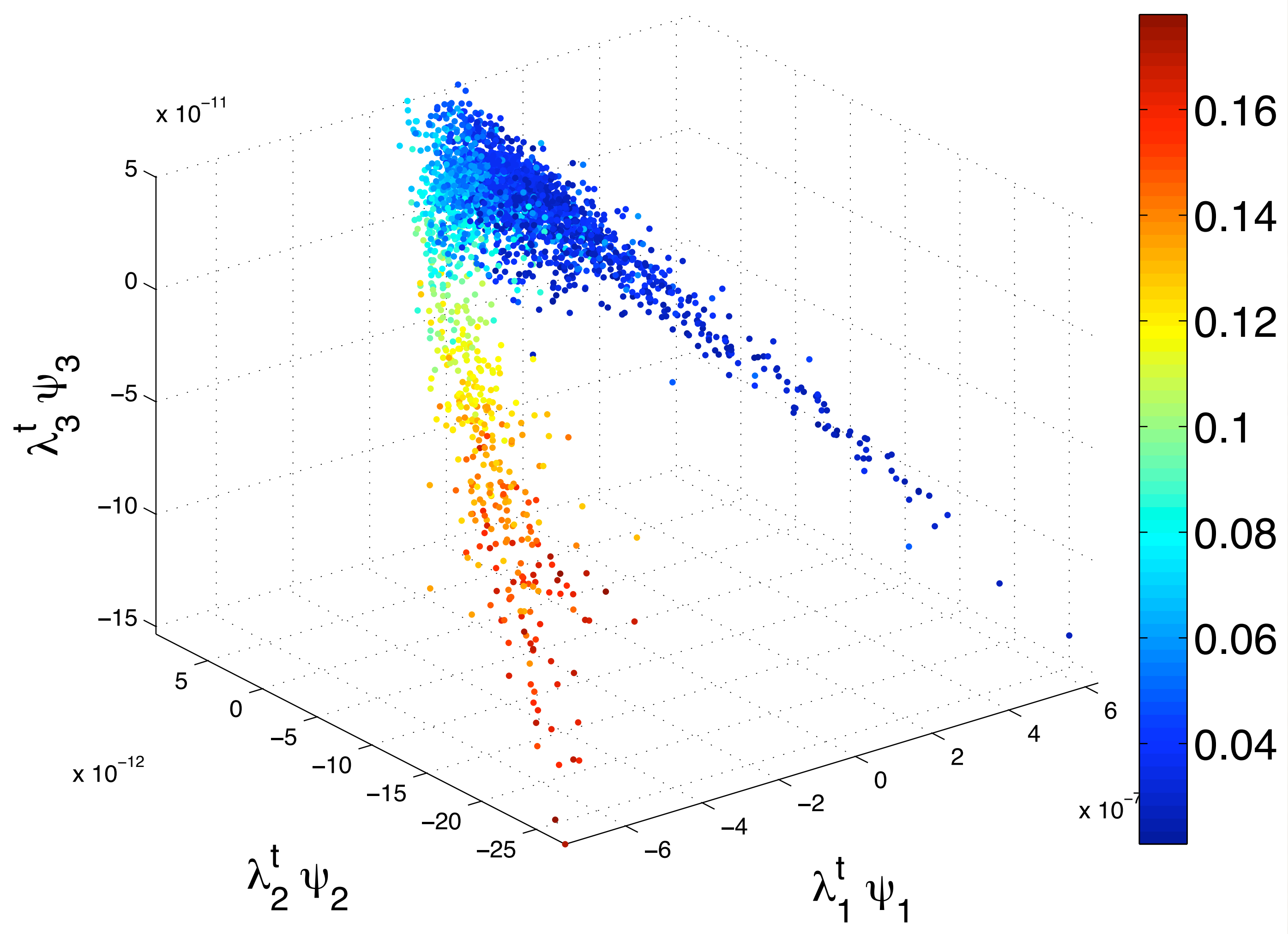
Application I

- Spectroscopic redshift estimation and outlier detection using SDSS galaxy spectra.
- Estimation via adaptive regression:

$$\widehat{r}(\boldsymbol{\Psi}_t) = \boldsymbol{\Psi}_t \widehat{\boldsymbol{\beta}} = \sum_{j=1}^{m} \widehat{\beta}_j \Psi_{t,j}(\boldsymbol{x})$$

$$= \sum_{j=1}^{m} \widehat{\beta}_j \lambda_j^t \psi_j(\boldsymbol{x}) = \sum_{j=1}^{m} \widehat{\beta}'_j \psi_j(\boldsymbol{x})$$

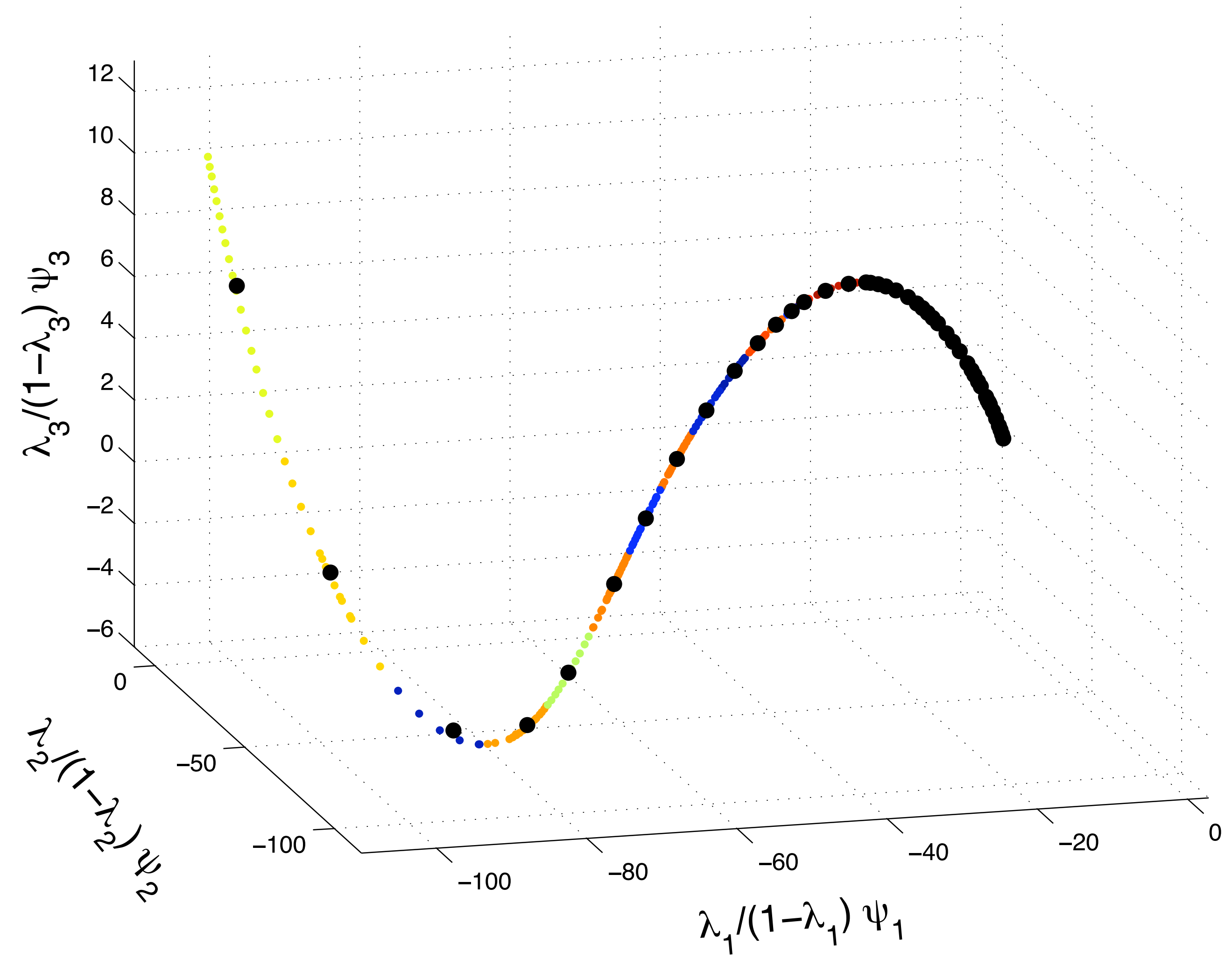
True Spectroscopic Redshift



Richards et al. (2009; ApJ 691, 32)

Application II

- Estimating properties of SDSS galaxies (age, metallicity, etc.) using a subset of the Bruzual & Charlot (2003) dictionary of theoretical galaxy spectra.
- Selection of prototype spectra made through diffusion K-means.

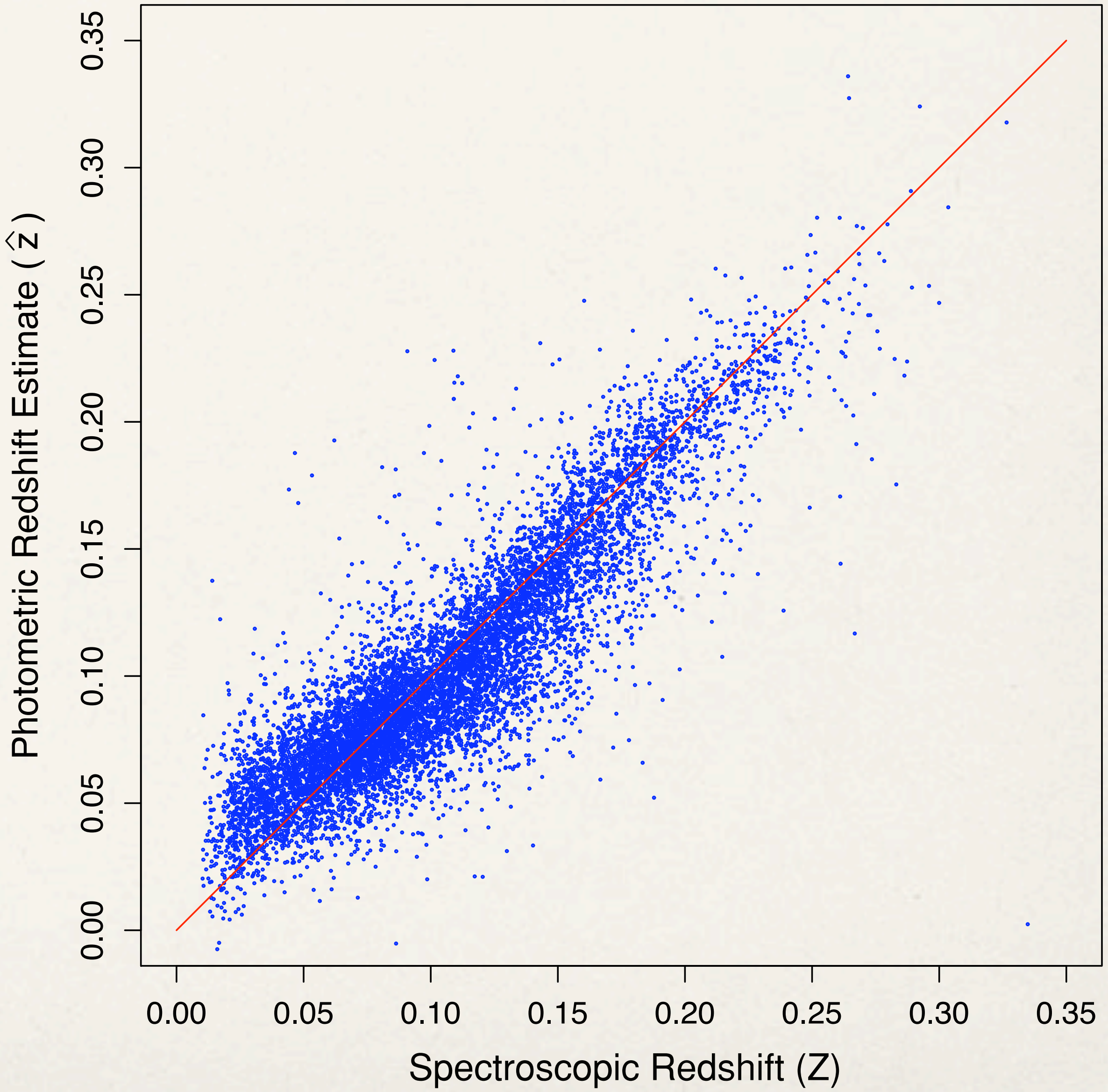


Richards et al. (2009; MNRAS 399, 1044)

Application III

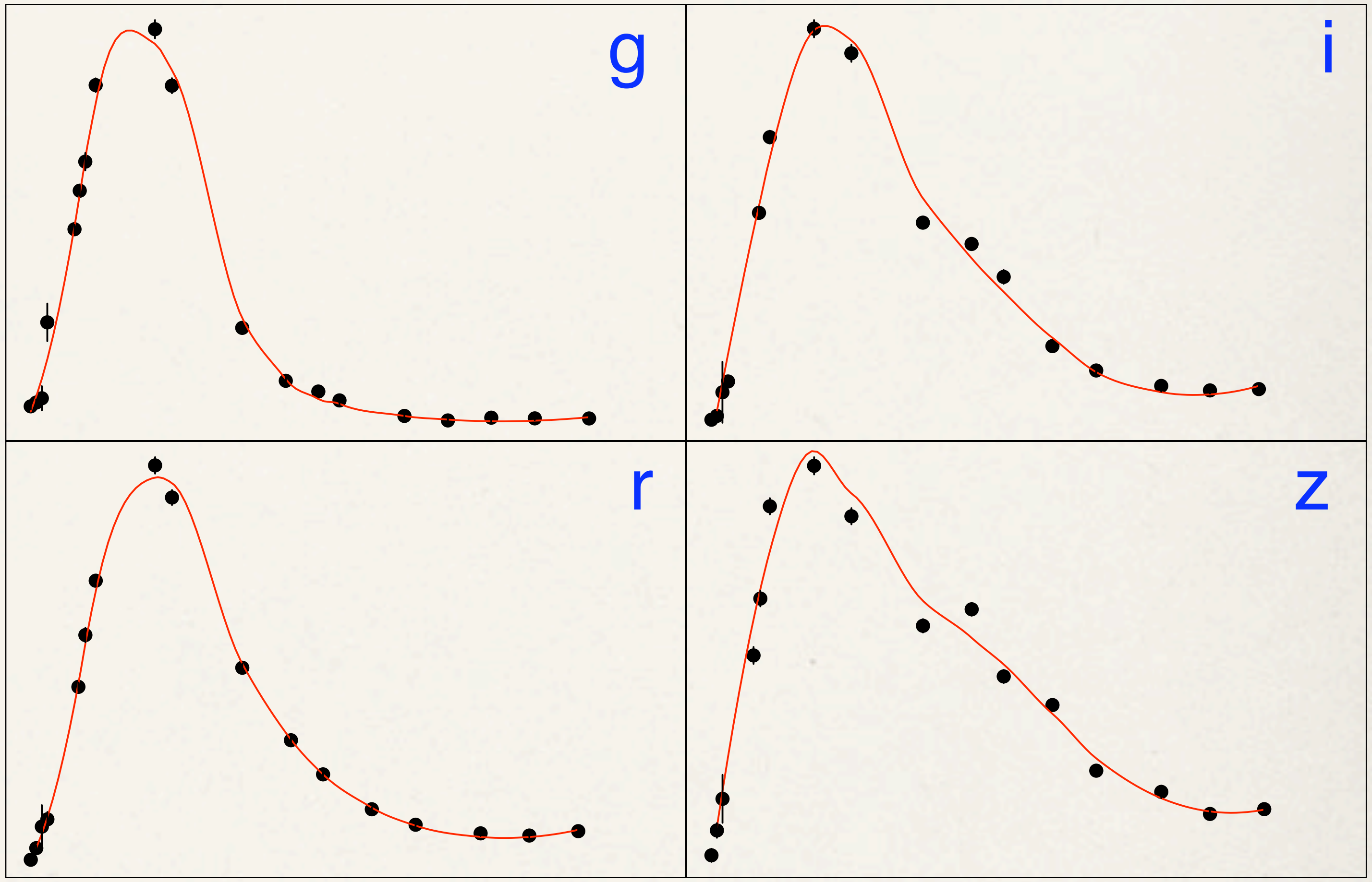
- Photometric redshift estimation for SDSS Main Sample Galaxies.
- Uses Nyström Extension for quickly predicting photometric redshifts of test set data, given the diffusion coordinates of training set data.
- Displays effect of flux measurement error upon predictions: attenuation bias.

Freeman et al. (2009; MNRAS 398, 2012)



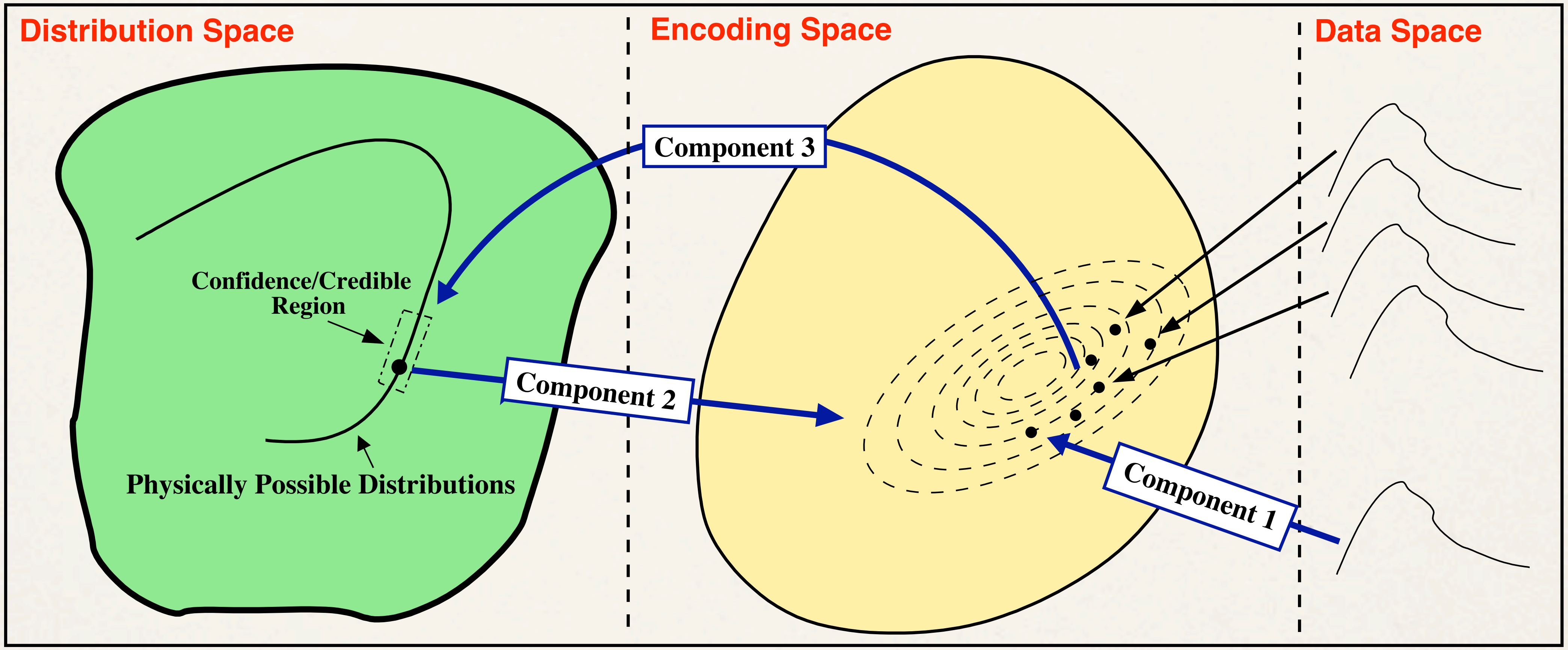
Application IV

- Classifying SNe in the Supernova Photometric Classification Challenge (Kessler et al. arXiv: 1001.5210)
- See talk by Joey Richards for more details!



Richards et al. (2010; in preparation)

Future Application



Transform observed light curves and theoretical light curves to a low-dimensional encoding space, where they may be compared using nonparametric density estimation.

Supernova light curves

Diffusion Map: Challenges

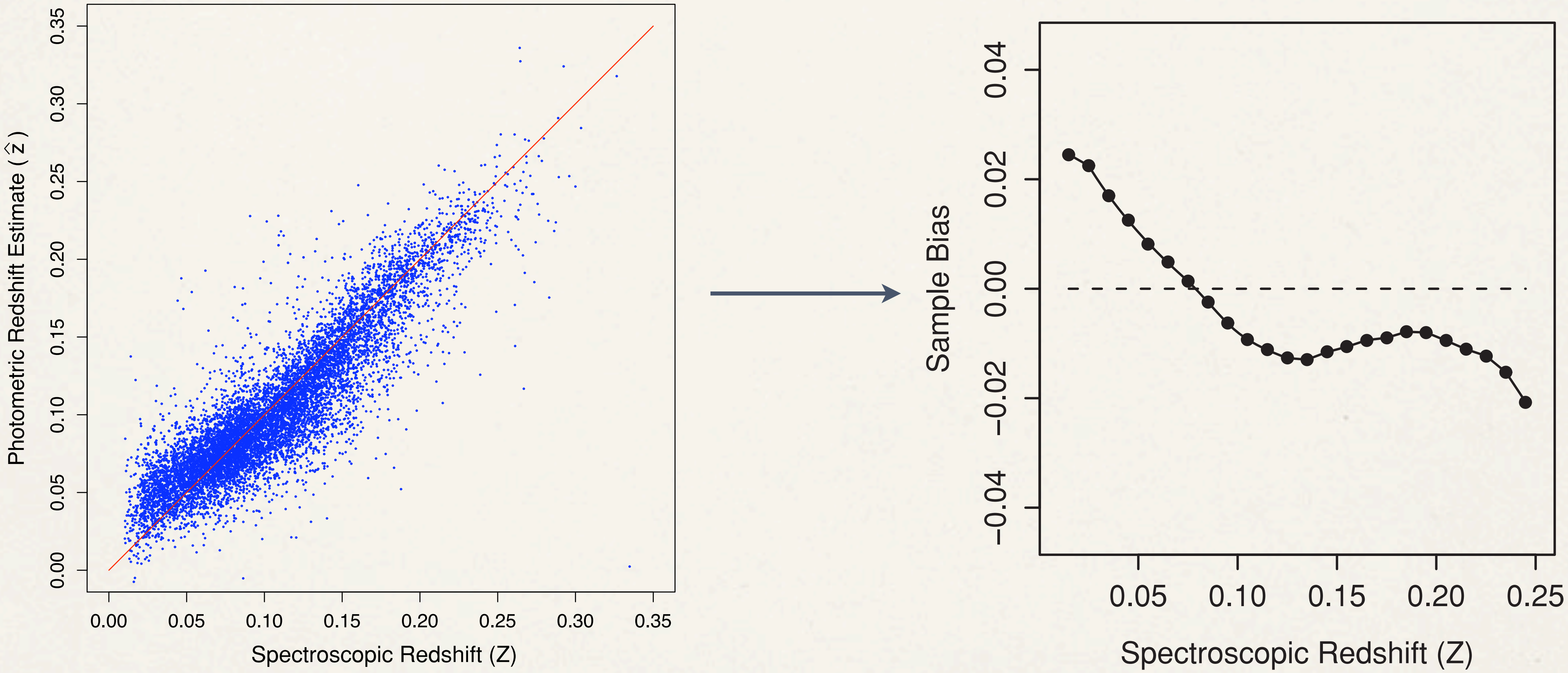
- Computational Challenge I: efficient construction of weighted graph w.
  - Distance computation slow for high-dimensional data.
  - Graph may be sparse: can we short-circuit the distance computation?
- Computational Challenge II: execution time and memory requirements for eigen-decomposition of the one-step probability matrix P.
  - SVD limited to approximately 10,000 x 10,000 matrices on typical desktop computers.
  - Slow: we only need the top n% of eigenvalues and eigenvectors, but typical SVD implementations compute *all* of them.
  - P may be sparse: efficient sparse SVD algorithms?
  - Would algorithm of Budavári et al. (2009; MNRAS 394, 1496) help?

Diffusion Map: Challenges

- Computational Challenge III: efficient implementation of the Nyström Extension to apply training set results to far larger test sets.
    - Predictions for 350,000 SDSS MSGs computed in 10 CPU hours...is this too slow in the era of LSST?

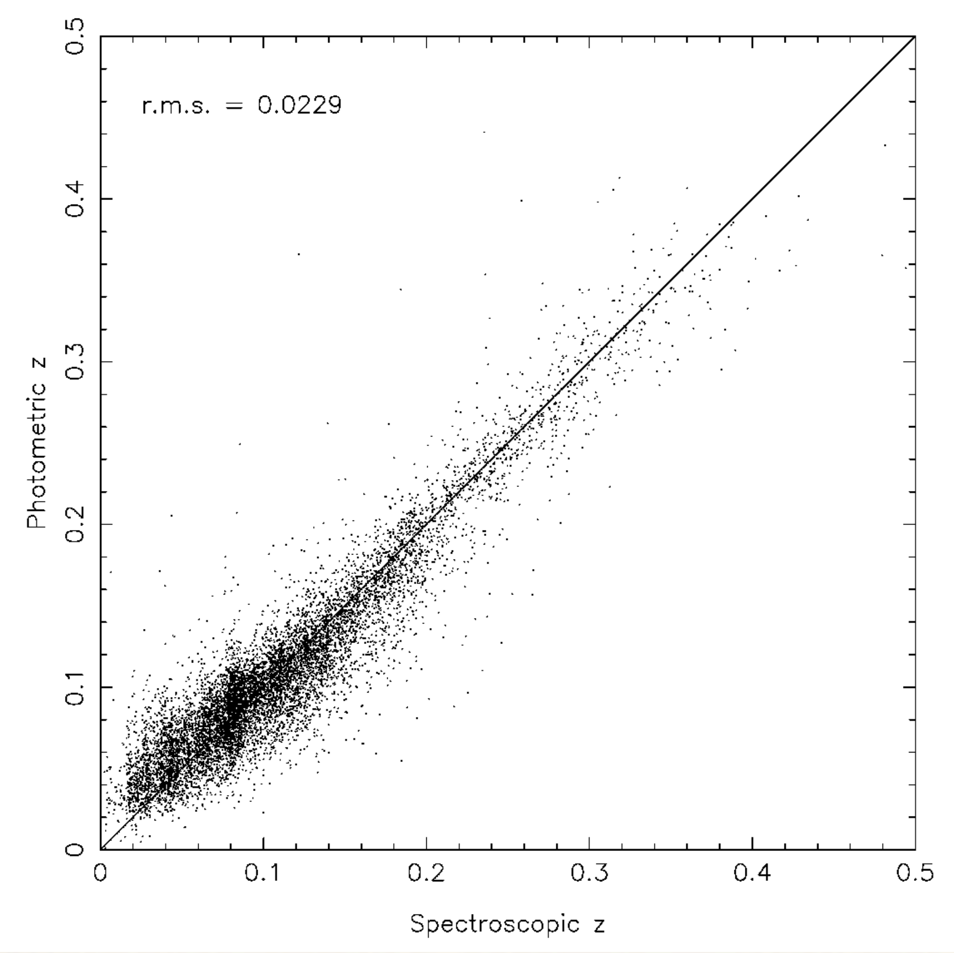
And One Statistical Challenge...

- Flux measurement error causes attenuation bias:

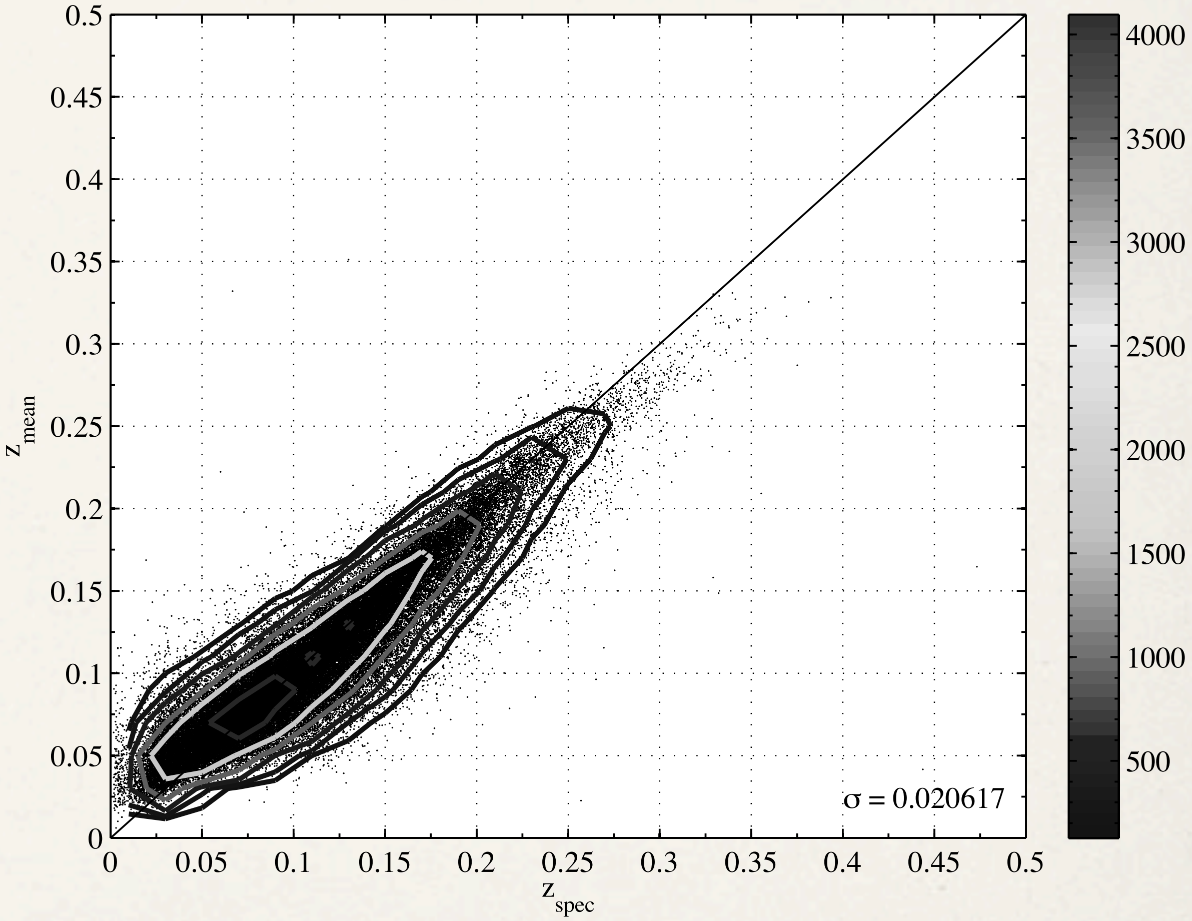


- Can attenuation bias be effectively mitigated? TBD.
- This is *not* diffusion map specific...

And One Statistical Challenge...



ANNz (Collister & Lahav 2004; PASP 116, 345)



kNN (Ball et al. 2008; ApJ 683, 12)

Summary

- Methods of nonlinear data transformation such as diffusion map can help make statistical analyses of complex (and perhaps high-dimensional) data tractable.
- Analyses with diffusion map generally outperform (i.e., result in a lower predictive risk) similar analyses with PCA, a linear technique.
- Nonlinear techniques have great promise in the era of LSST, so long as certain computational challenges are overcome. We seek
  - Optimal construction of weighted graphs
  - Optimal implementations of SVD (memory, execution time, sparsity)
  - Optimal implementation of the Nyström Extension
- Regardless of whether the challenges are overcome, the accuracy of our results may be limited by measurement error.

Predictive Risk: an Algorithm

- Pick tuning parameter values ε and *m*.
- Transform the data into diffusion space.
- Perform *k*-fold cross-validation on the transformed data:
    - Assign each datum to one of *k* groups.
    - Fit model (e.g., linear regression) to the data in *k*-1 groups (i.e., leave the data of the k^{th} group out of the fit).
    - Given best-fit model, compute estimate $\hat{y}_i$ for all data in the k^{th} group.
    - Repeat process until all *k* groups have been held out.
- Assuming the L_2 (squared-error) loss function, our estimate of the predictive risk is generally

$$\widehat{R}(\epsilon, m) = \frac{1}{n}\sum_{j=1}^{n}[\widehat{y}_j(\epsilon, m) - Y_j]^2$$

- We vary ε and *m* until the predictive risk estimate is minimized.

Nyström Extension

- The basic idea: compute the similarity of a test set datum to the training set data, and use that similarity to determine the diffusion coordinate for that datum via interpolation, with no eigen-decomposition.
- Mathematically:

$$\Psi' = \mathbf{W}\Psi\Lambda$$

- W is the matrix of similarities between the test set data and the training set data, while Λ is a diagonal matrix with entries $1/\lambda_i$.