Data compression and regression based on local principal curves and surfaces

Jochen Einbeck

Department of Mathematical Sciences, Durham University

jochen.einbeck@durham.ac.uk

joint work with Ludger Evers (University of Glasgow),

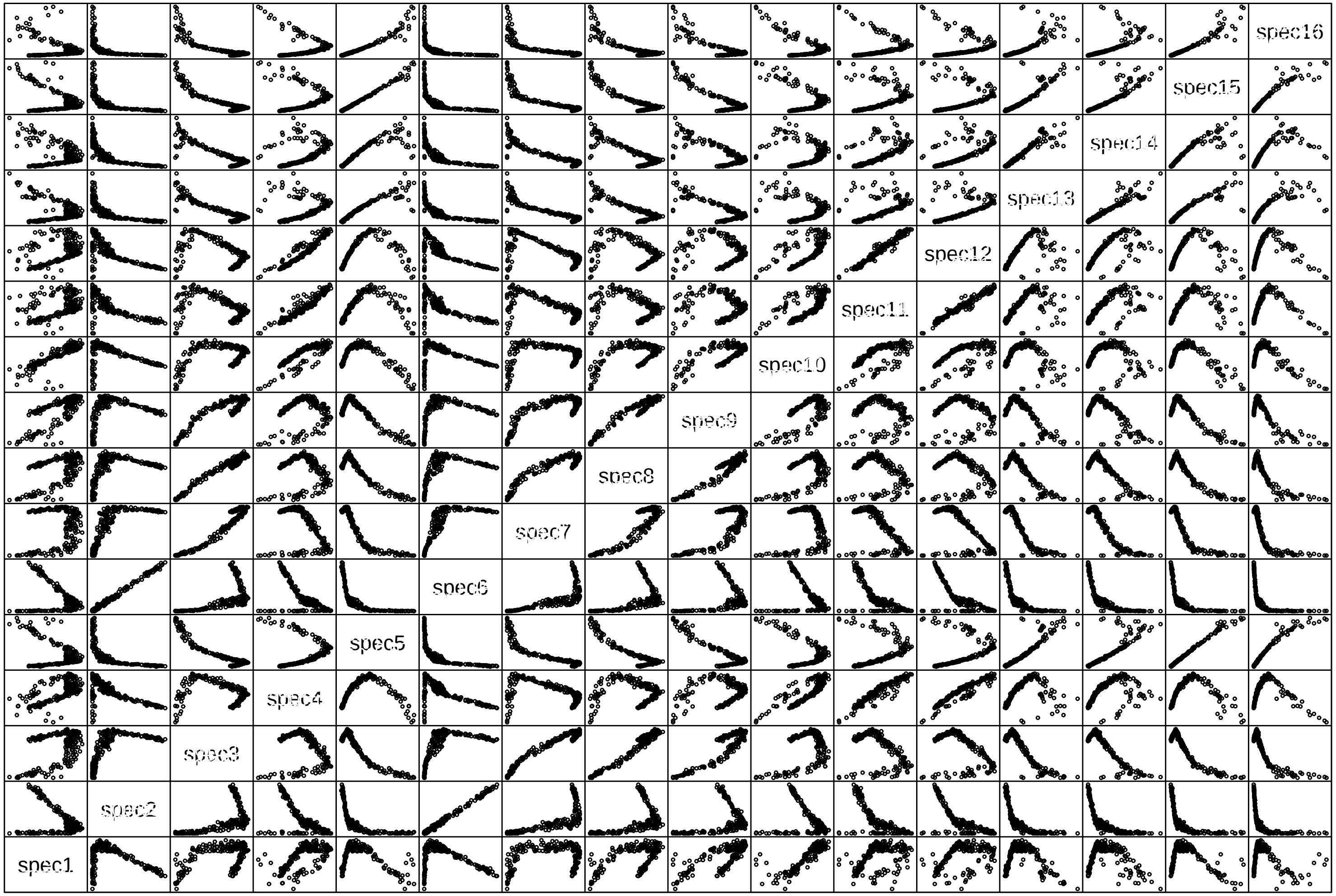
Galway, 3rd of December 2009

Motivation: GAIA data

- GAIA is an astrophysics mission of the European Space Agency (ESA) which will undertake a detailed survey of over 10^9 stars in our Galaxy and extragalactic objects.
- Satellite to be launched in 2012.
- Aims of the mission (among others)
  - Classify objects (star, galaxy, quasar,...)
  - Determine astrophysical parameters ("APs": temperature, metallicity, gravity) from spectroscopic data (photon counts at certain wavelengths).
- Work is led by the group "Astrophysical parameters" based at MPIA Heidelberg, being part of the DPAC (Data Processing and Analysis Consortium) which is responsible for the general handling of data from the GAIA mission.
- Yet, one has to work with simulated data generated through complex computer models.

GAIA data

- Photon counts $(n = 8286)$ simulated from APs:



GAIA data: Estimation of APs

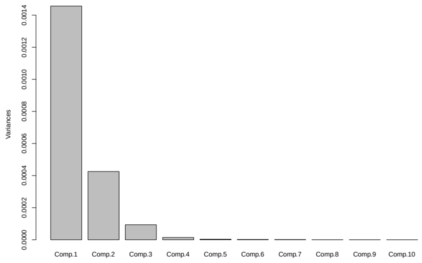
- Try linear model for the temperature, using training sample of size $n = 1000$:

```
> lm(temperature˜ spec1 +...+ spec16, data= gaia)
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -14033286   21104764  -0.665    0.506
spec1        14065842   21104812   0.666    0.505
spec2        14216977   21107526   0.674    0.501
.
.                 .          .       .        .
spec16       13886697   21106076   0.658    0.511
Residual standard error: 1978 on 983 degrees of freedom
```

- Multicollinearity!
- Does not seem to be a useful model for this data.

Dimension reduction

- Usual remedies:
  - Model/ variable selection procedures
  - Dimension reduction techniques
- Look at scree plot:



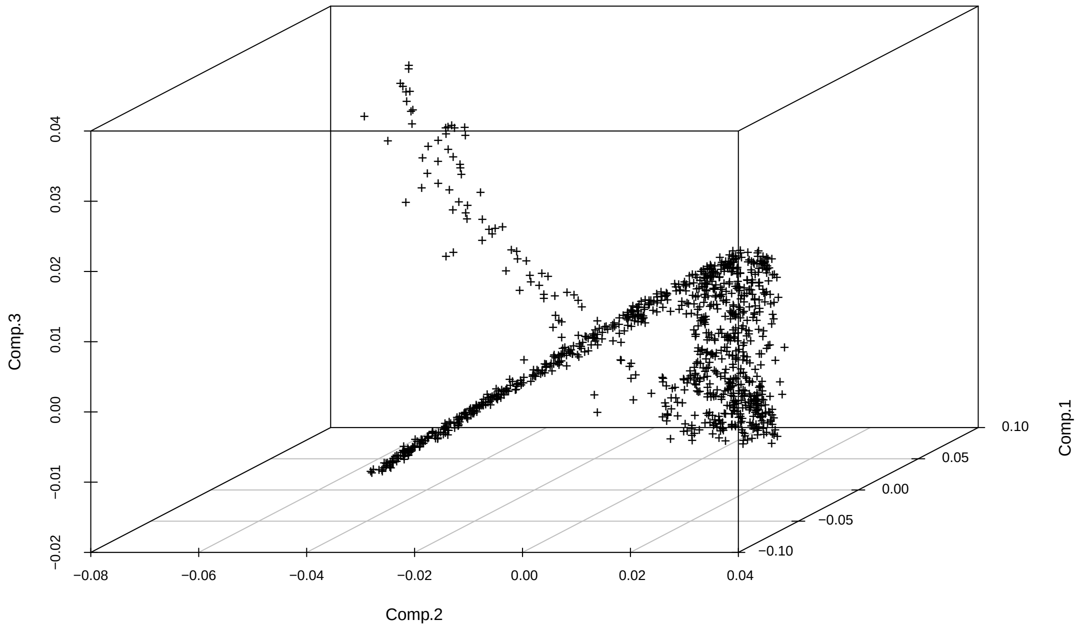
- Three principal components appear to be sufficient.

```
> lm(temperature ~ Comp1 + Comp2 + Comp3, data = gaiapc)
               Estimate Std. Error t value Pr(>|t|)
(Intercept)    10835.90      65.14  166.34   <2e-16 ***
Comp1        -187339.39    1706.85 -109.76   <2e-16 ***
Comp2        -173967.35    3157.61  -55.09   <2e-16 ***
Comp3        -155314.86    6726.19  -23.09   <2e-16 ***
Residual standard error: 2060 on 996 degrees of freedom
```

looks better than LM, but...

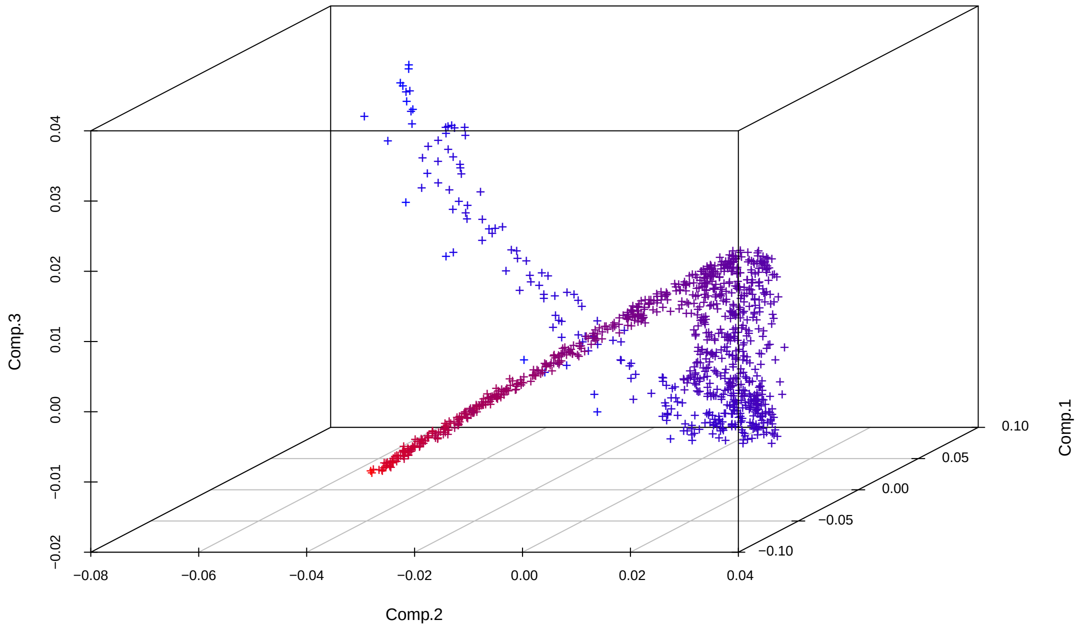
Principal component scores

- We plot the the first three principal component scores.



Principal component scores (cont.)

- We plot the the first three principal component scores and shade higher temperatures red.



- Actually, we seem to need only *one* parameter if we were able to lay a smooth curve through the data cloud.
- The parametrization along such a curve would be informative w.r.t. to the target variable, temperature.

GAIA data and principal curves

- Hence, the following is to do:
  (1) Estimate the smooth curve capturing the structure of the (3-dim/16-dim) predictor space.
  (2) Parametrize this curve.
  (3) Project all data points onto it.
  (4) Fit temperature (or other APs) against the (1-dim.) projections.
- Step (1) is a task for principal curves. There are a couple of principal curve algorithms available, but not all of them are suitable for tasks (2)-(4).

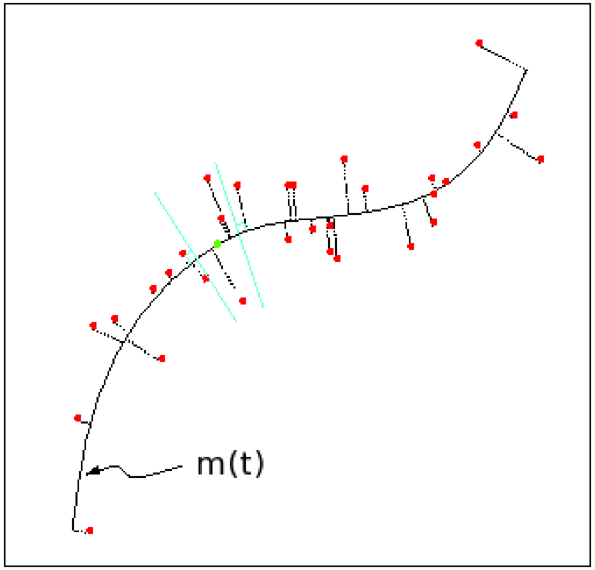
Principal curves: Original definition

Hastie & Stützle (HS, 1989) define each point on the principal curve m as the average of all points which project there ('self-consistency'), i.e.

$$m(t) = E(X|t_m(X) = t)$$

where $t_m(X)$ is the projection index of X onto the curve m.

- If the principal curve is linear, then it is a principal component.
- If a curve $m(t)$ is self-consistent, it is a critical point of the distance function
$$\triangle(m) = E\left(\inf_t ||X - m(t)||^2\right).$$
- However, it was later shown that the critical point is actually just a saddle point of $\triangle(m)$.
- If $X = g(T) + \epsilon$ with T uniform and $\epsilon \sim N(0, \sigma^2 I)$, then generally $m \neq g$!



(from: Hastie et al, 2001))

Types of principal curves

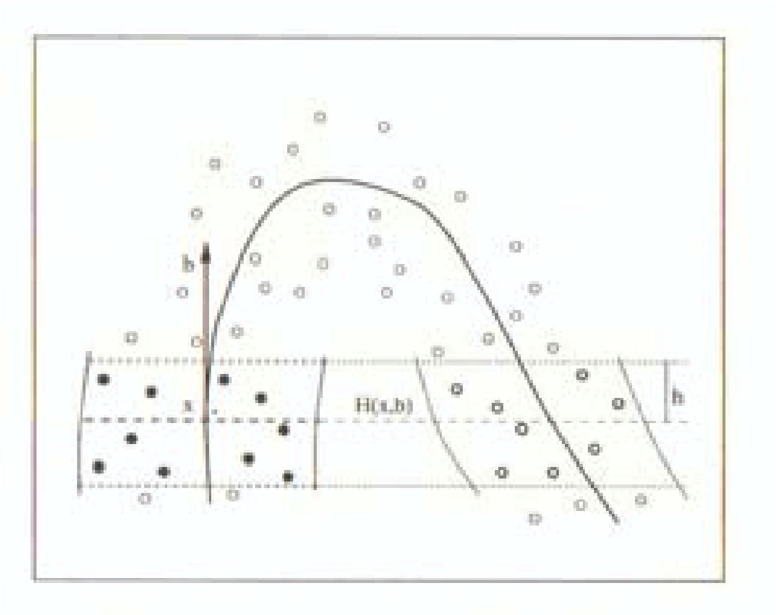
Today exist a variety of different notions of principal curves, which vary essentially in how the “middle” of the data cloud is defined/found:

- Global (‘top-down’) algorithms start with an initial line (usually the 1st PC) and bend this line or concatenate other lines to it until some convergence criterion is met (HS, Tibshirani 1992, Kégl et al 2002, . . .)
  - Allows theoretical analysis.
  - Goes wrong if initial oder of projection indices is not right.
  - Extension to branched or disconnected data clouds difficult.
- Local (‘bottom-up’) algorithms estimate the principal curve locally moving step by step through the data cloud (Delicado 2001, Einbeck et al 2005)
  - More flexible, but far more variable.
  - Extend straightforwardly to branched and disconnected data.
  - Theoretical investigations rather difficult.

Delicado's PCOPs

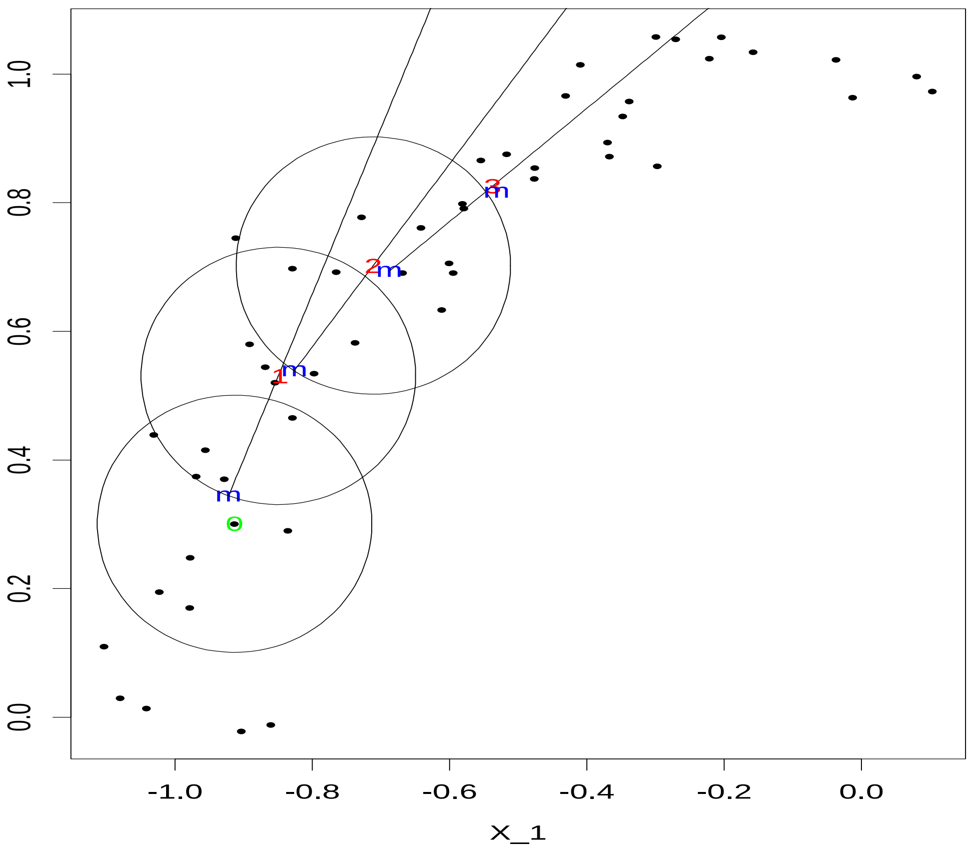
Delicado (2001) defines *principal curves of oriented points* (PCOPs) as a sequence of fixed points of the function $\mu^*(x) = E(X|X \in H)$, where H is the hyperplane through x minimizing locally the variance of the data points projected on it.

- Works fine for most (not too complex) data sets.
- Mathematically elegant
- However, quite complicated and computationally demanding.
- Requires a cluster analysis at every point of the principal curve.



Local principal curves (LPCs)

Einbeck, Tutz & Evers (2005) : Calculate alternately a local center of mass and a first local principal component.



0: starting point,
m: points of the LPC,
$1, 2, 3$: enumeration of steps.

Algorithm for LPCs

Given: A data cloud $X = (X_1, \ldots, X_n)$, where $X_i = (X_{i1}, \ldots, X_{id})$.

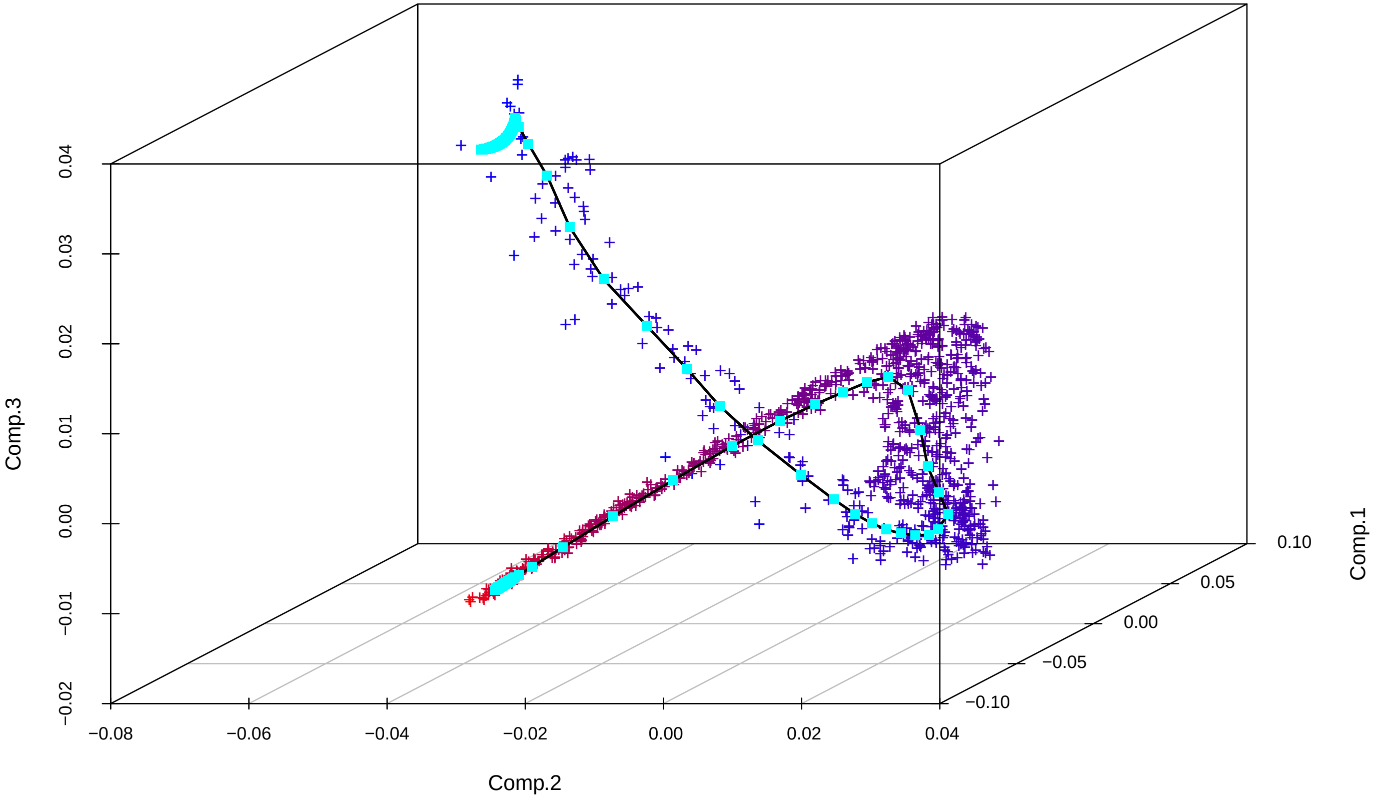
1. *Choose a starting point x_0. Set $x = x_0$.*
2. *At x, calculate the local center of mass $\mu^x = \sum_{i=1}^n w_i X_i$, where $w_i = K_H(X_i - x)X_i / \sum_{i=1}^n K_H(X_i - x)$.*
3. *Compute the 1^{st} local eigenvector γ^x of $\Sigma^x = (\sigma^x_{jk})_{(1 \leq j,k \leq d)}$, where*
$$\sigma^x_{jk} = \sum_{i=1}^n w_i (X_{ij} - \mu^x_j)(X_{ik} - \mu^x_k).$$
4. *Step from μ^x to $x := \mu^x + t_0 \gamma^x_1$.*
5. *Repeat steps 2. to 4. until the μ^x remain constant. Then set $x = x_0$, set $\gamma^x := -\gamma^x$ and continue with 4.*

The sequence of the local centers of mass μ^x makes up the local principal curve (LPC).

Step 1: Fitting the LPC (cont.)

- LPC through principal component scores of photon counts, with local centers of mass μ (sky blue squares):

```
> gaia.lpc <- lpc(gaia.pc$scores)
```



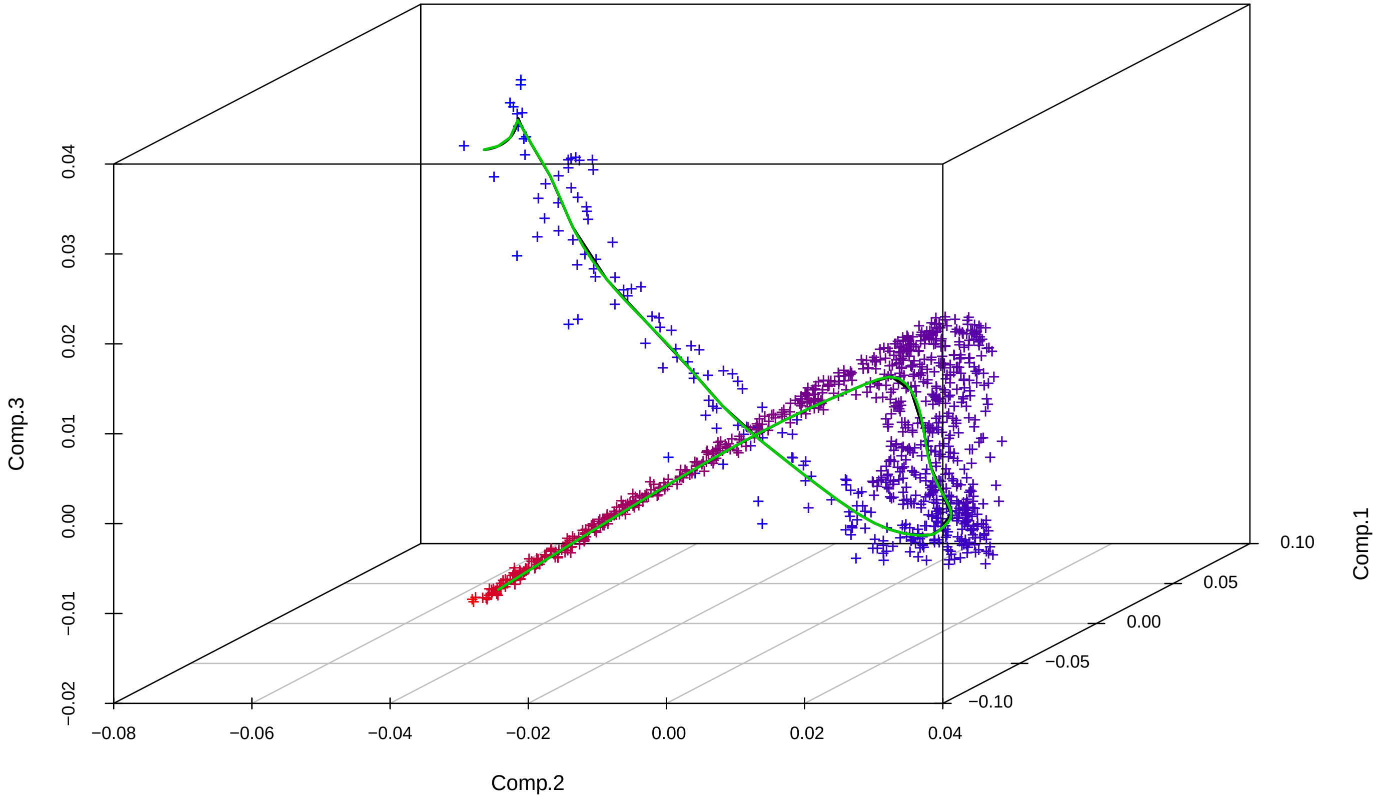
Step 2: Parametrization

- Unlike HS curves, LPCs do not have a natural parametrization, so it has to be computed retrospectively.
- Define a preliminary parametrization $s \in \mathbb{R}$ based on Euclidean distances between neighboring $\mu \in \mathbb{R}^d$.
- For each component μ_j, $j = 1, \ldots, d$, use a natural cubic spline to construct functions $\mu_j(s)$, yielding together a function $(\mu_1, \ldots, \mu_d)(s)$ representing the LPC (no smoothing involved here!).
- Recalculate the parametrization along the curve through the arc length of the spline function,

$$t = \int_0^s \sqrt{(\mu_1'(u))^2 + \ldots + (\mu_p'(u))^2}\, du$$

Step 2: Parametrization (cont.)

```
> lpc.spline(gaia.lpc)
```

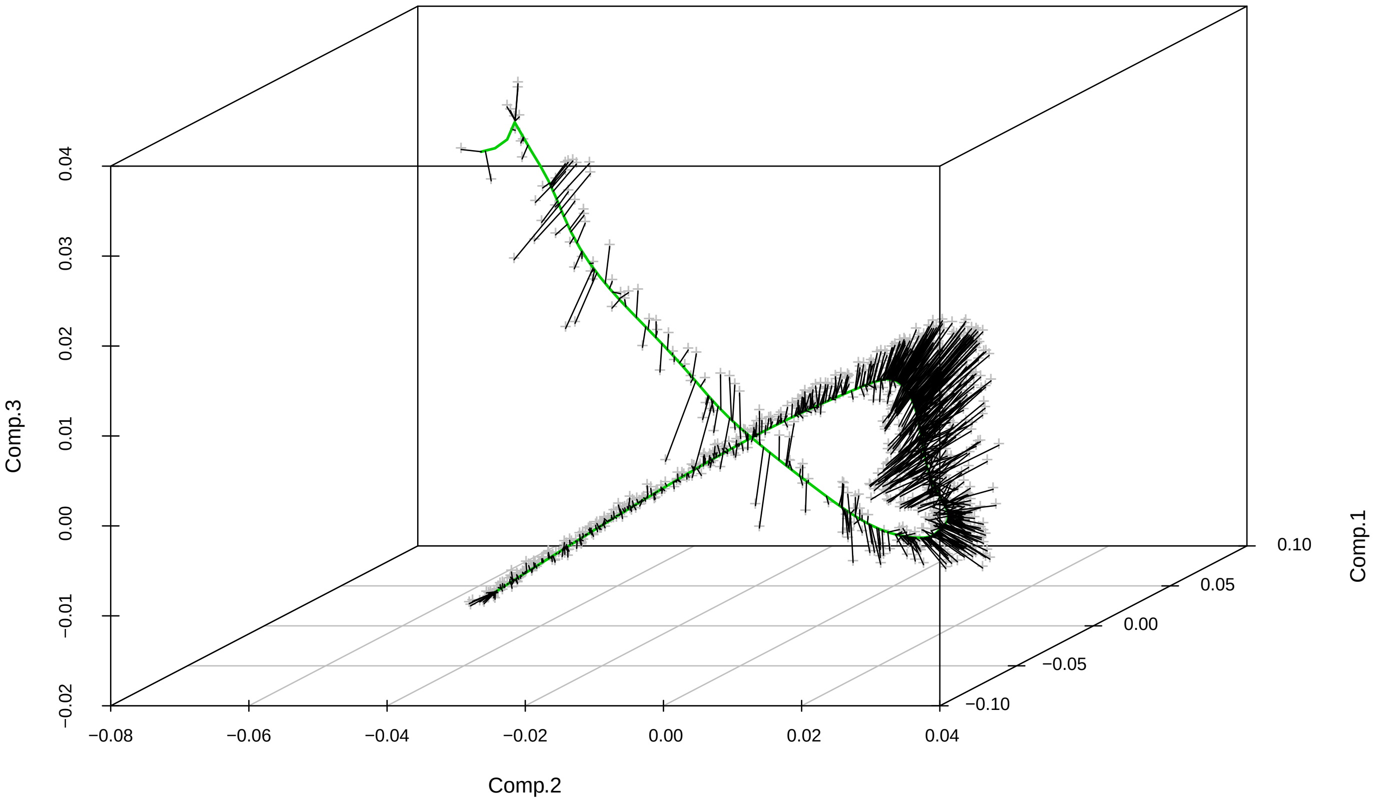


- The spline function (—) is almost indistinguishable from the original LPC (—).

Step 3: Projection

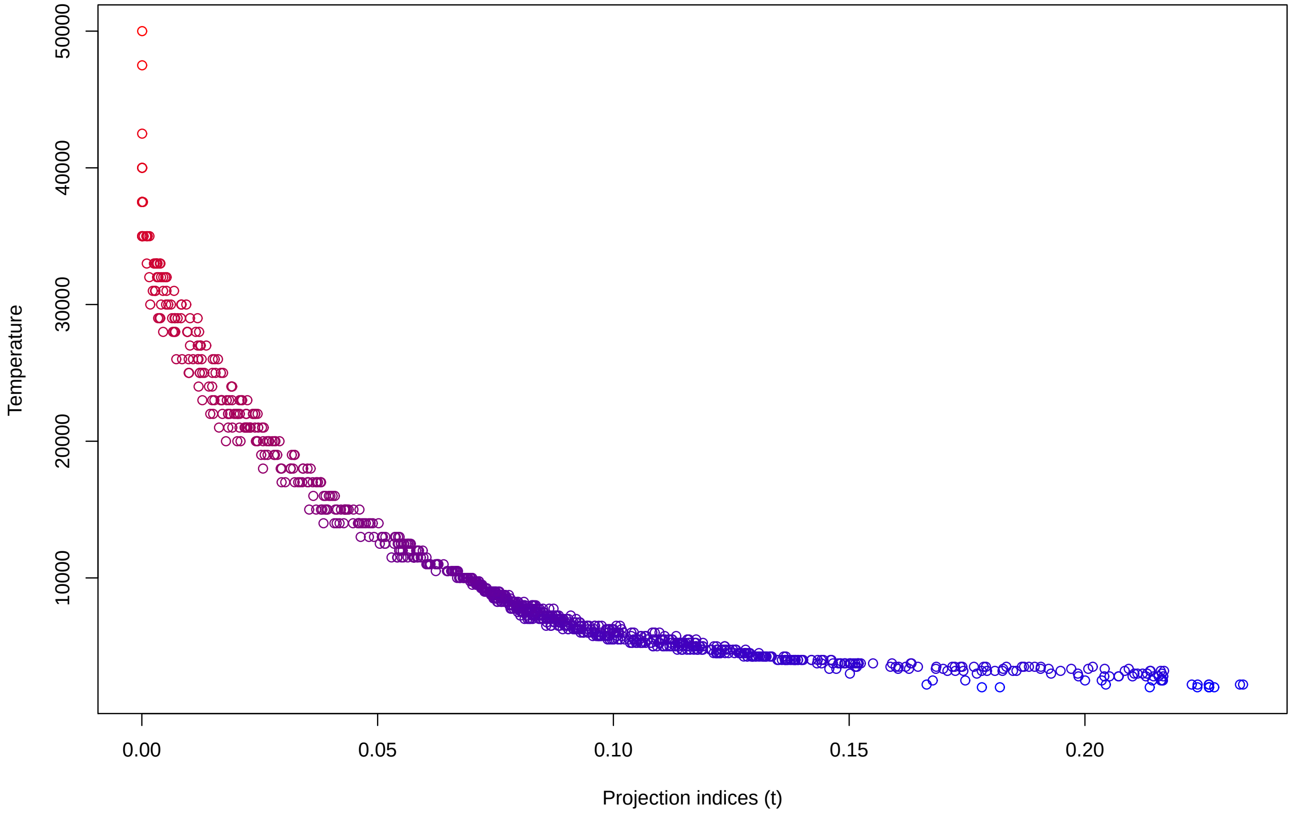
- Each point $x_i \in \mathbb{R}^d$ is projected on the point of the curve nearest to it, yielding the corresponding projection index t_i

```
> lpc.spline(gaia.lpc, project=TRUE)
```



Step 4: Regression

- We want to predict stellar temperature from 16-d spectral data, using the projection indices of the spectra as predictors.

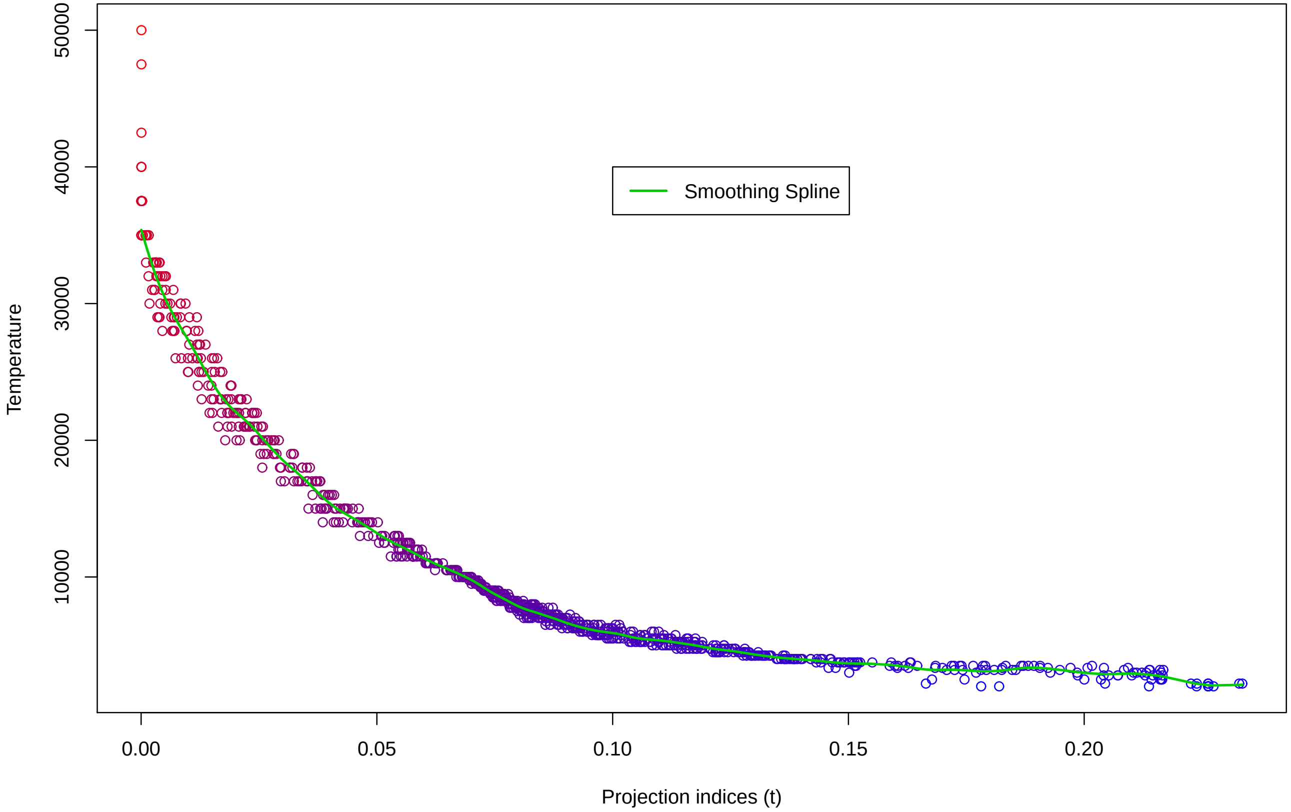


Step 4: Regression (cont.)

- This is now a simple one-dimensional regression problem.

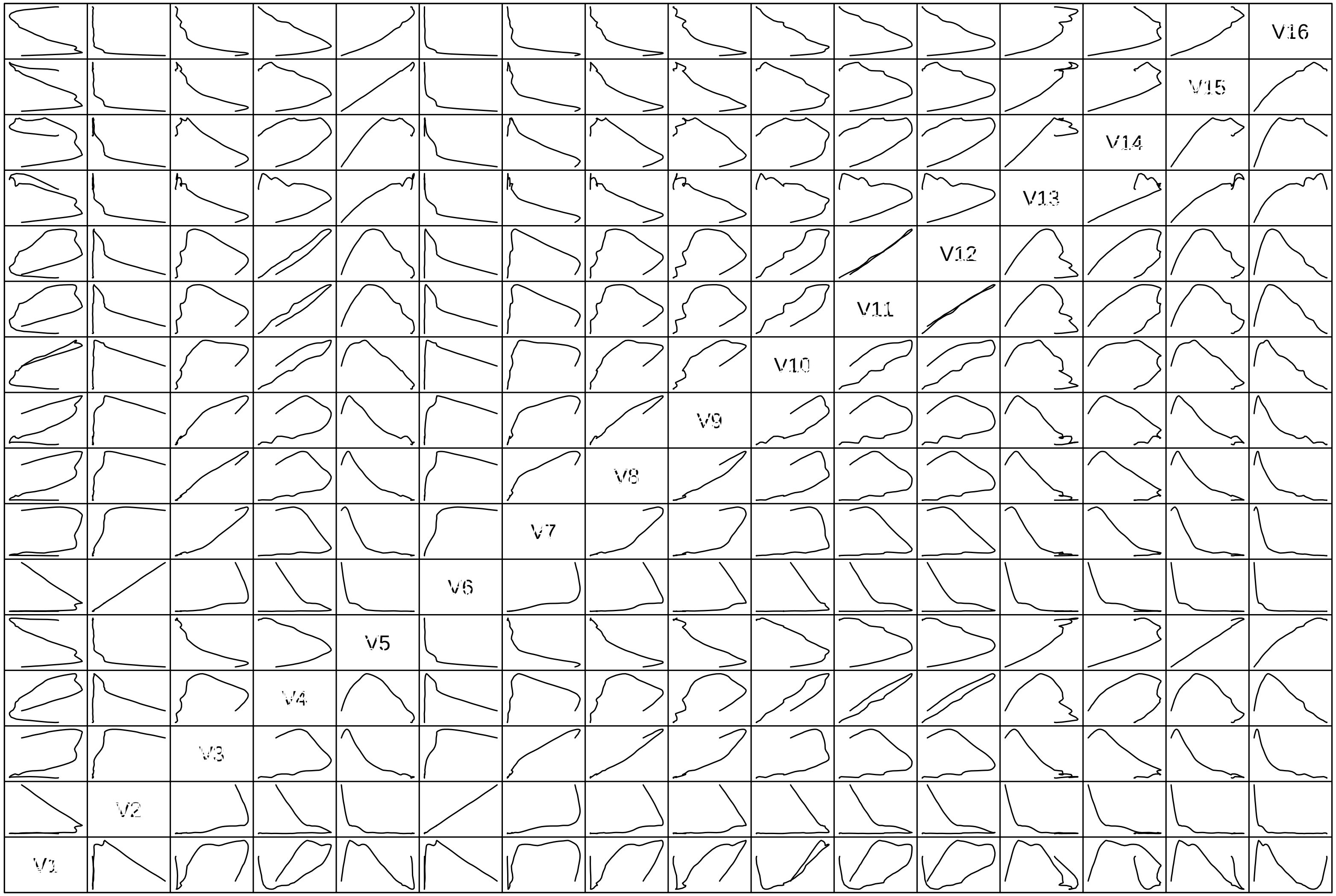
$$y_i = m(t_i) + \varepsilon_i$$

- Using penalized smoothing splines:



Shortcut

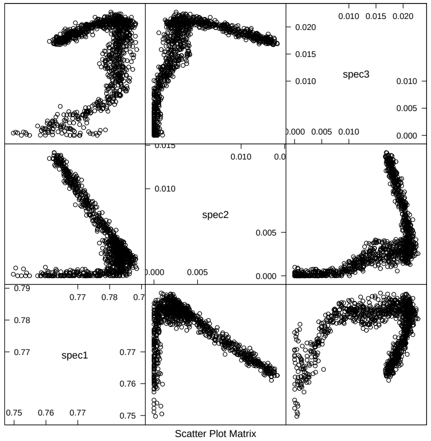
- LPC fitted *directly* through 16- dimensional space:



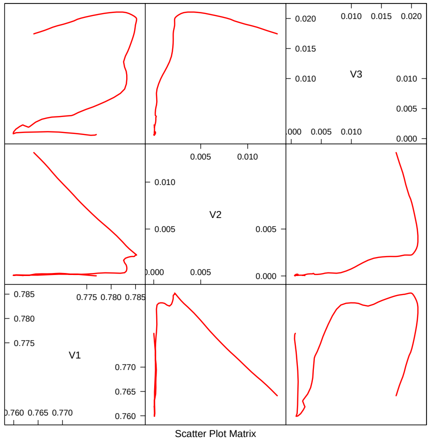
Shortcut (cont.)

- Zoom into the the first three dimensions:

Data



LPC



- Direct data compression with LPCs works in principle, but is potentially “dangerous” as data gets sparse in high dimensions and remote parts of the predictor space maye be missed.

Prediction

- For a new observation x_{new} (i.e., here, a new set of spectra), prediction proceeds as follows:
  - Project x_{new} onto the LPC, giving t_{new}.
  - Compute $\hat{y}_{new} = \hat{m}(t_{new})$ from the fitted regression model.
- Comparison: We sample $n' = 1000$ test data from the remaining $8286 - 1000$ observations and observe the prediction error:

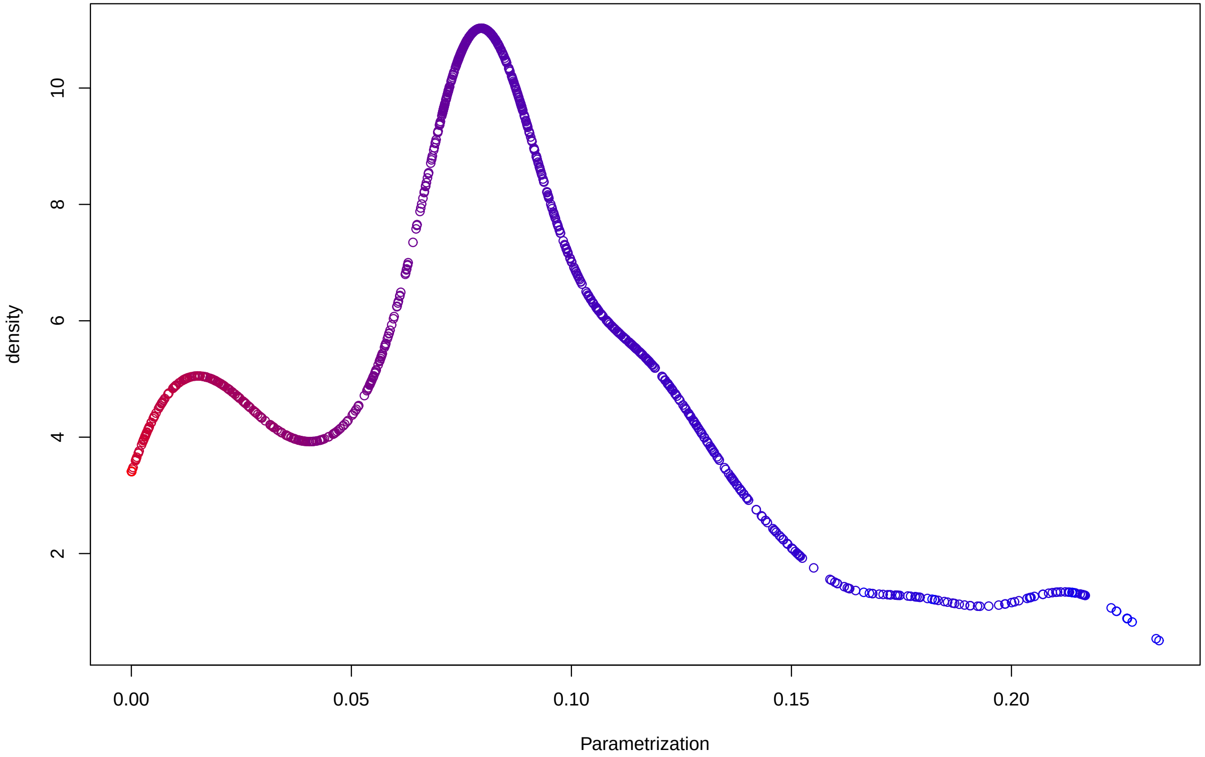
prediction error $/10^3$	LM	PC+LM	PC+AM	PC+LPC	LPC (2nd run)
average($\hat{\varepsilon}_i^2$)	4'593	4'967	1'732	1'430	1'044 (2'025)
median($\hat{\varepsilon}_i^2$)	1'049	1'124	104	52	69 (71)

where $\hat{\varepsilon}_i$ is the difference between true and predicted temperature.

Density estimation

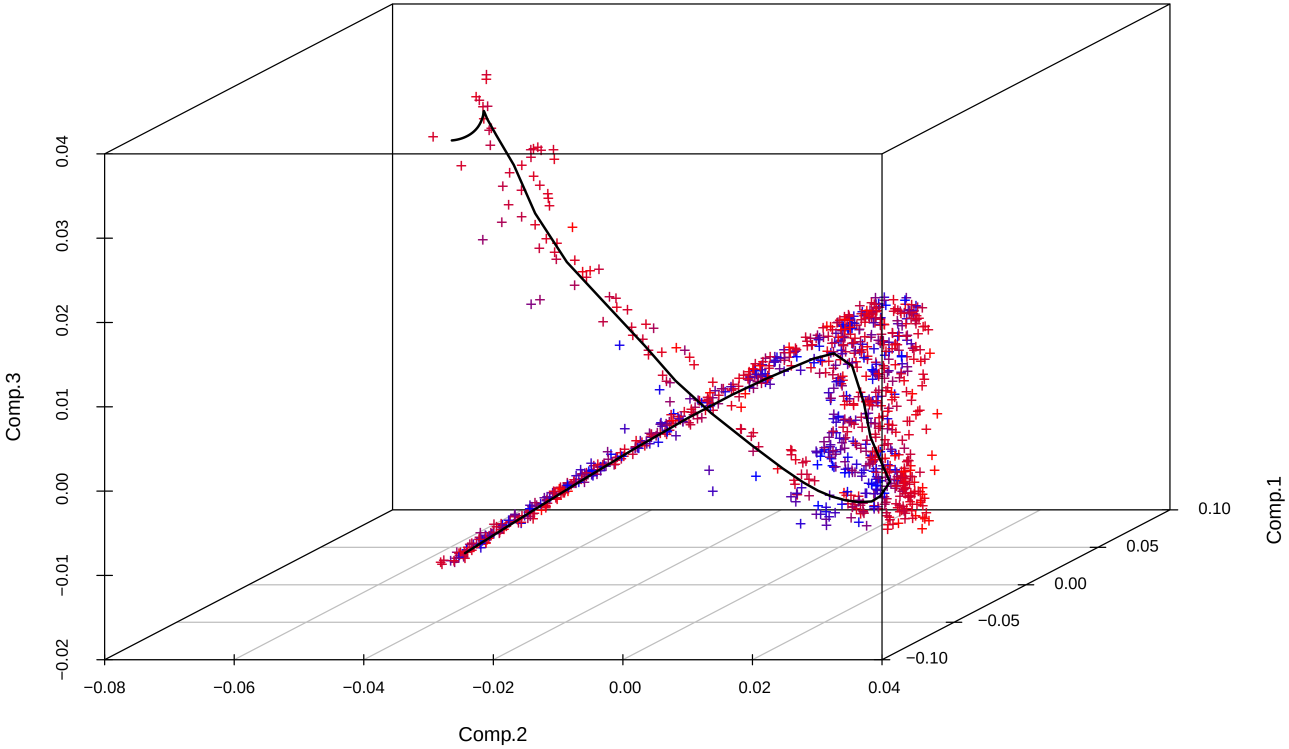
- Having now the projection indexes t_i, $i = 1, \ldots, n$, this can be easily used for other purposes such as "density estimation along the principal curve":

$$\hat{f}(t) = \frac{1}{nh} K\left(\frac{t - t_i}{h}\right)$$



Limits of one-dimensional data summaries

- Look at “metallicity”



- The relevant information seems to be orthogonal to the principal curve!

Local principal surfaces

- To handle this and more complex data, the extension to *local principal surfaces* and *manifolds* should be considered.
- To this end, firstly observe that, from the two components of the LPC algorithm, namely

  (1) local center of mass (mean shift)

  (2) localized first principal component

  the more important is (rather surprisingly) (1).
- Instead of (2), any other movement "roughly in the direction of the data cloud" can be made, and step (1) will shift it back to the data cloud.
- We exploit this observation for the extension to local principal surfaces.

Local principal surfaces (cont.)

- Instead of points x, we work with the “building block” triangles Δ.
- Local PCA is only used to determine the initial triangle, say Δ_0.
- Then, the algorithm iterates

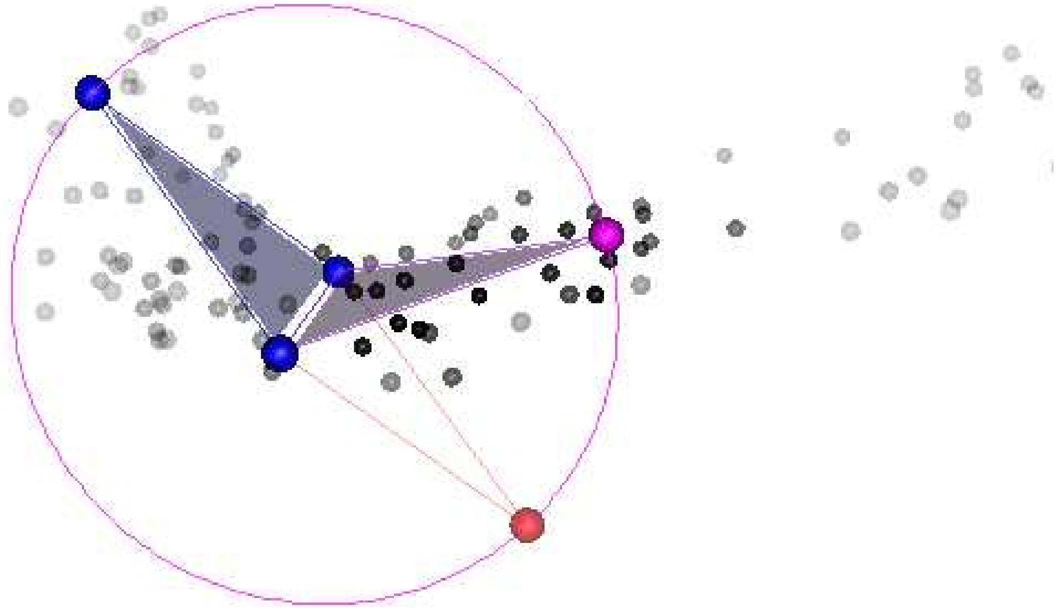
  (1) For a given triangle Δ, we glue further triangles at each of its sides $j = 1, 2, 3$.

  (2) For $j = 1, 2, 3$, adjust the free triangle vertex via the mean shift. We dismiss the new triangle if
    - the new vertex falls into a region of small density, or
    - the new vertex is too close to an existing one (Delaunay triangulation).

  until all sides of all triangles (including the new ones) have been considered.

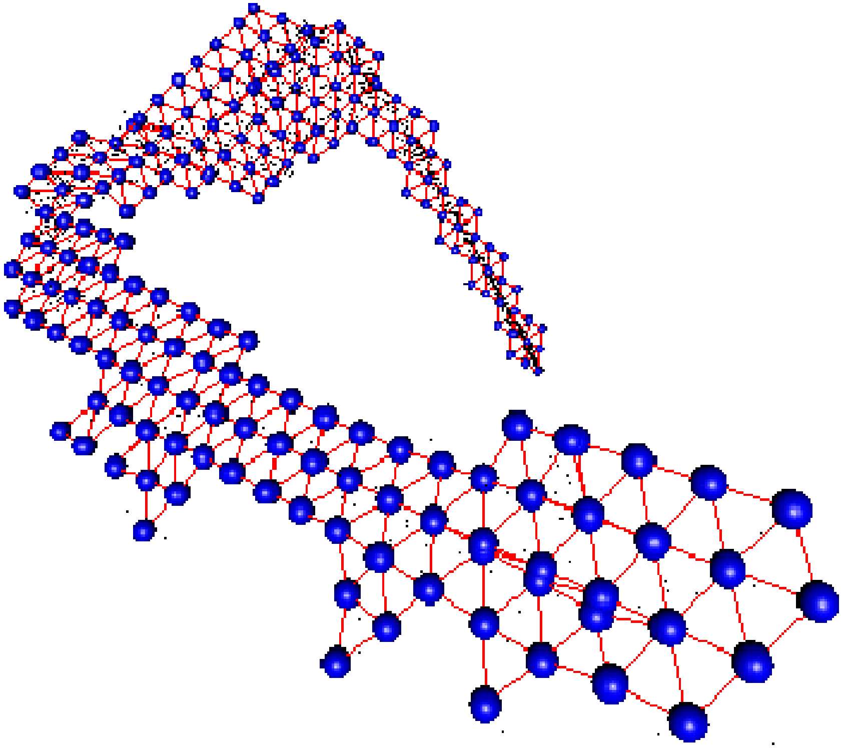
Local principal surfaces (cont.)

- Illustration: Constrained mean shift on a circle (enforcing equiliteral triangles):



Local principal surface for GAIA data

- Local principal surface (LPS) for PC scores based on training data set with $n = 1000$:



Regression on the surface

- Then, how to use this surface for regression?
- It seems hard to define a meaningful 2-dim. parametrization on the surface.
- Instead, we do some sort of kernel smoothing: For each triangle, we can count the distance d to all other triangles through the smallest number of triangle borders that have to be crossed to walk from one to the other.
- Local weights are assigned through a discrete distance-based kernel

$$\kappa(d) = e^{-d/\lambda}$$

The smoothing parameter $\lambda \in [0, \infty)$ steers the degree of smoothing on the manifold: the higher λ, the smoother it is.

Regression on the surface (cont.)

The entire fitting process is summarized as follows:

(I) Fit a LPS as explained above, leading to surface with, say, R triangles.

(II) Assign each data point $X_i, i = 1, \ldots, n$ to their nearest triangle.

(III) For each triangle $r = 1, \ldots, R$, compute the mean $\bar{y}_r$ over the response values of all data points assigned to it.

(IV) Compute all pairwise distances $d_{r,s}$ between all triangles on the surface.

(V) Use the discrete kernel $\kappa(\cdot)$ to smooth over the manifold. The smoothed response value m_r on triangle r is given by

$$m_r = \frac{\sum_s \kappa(d_{r,s})\bar{y}_s}{\sum_s \kappa(d_{r,s})}$$

which is at the same time the fitted value of all data points assigned to triangle r.

Simulation study

Prediction errors for $n' = 1000$ test data. The LPS is fitted with $\lambda = 1$.

- Temperature

prediction error $/10^3$	LM	PC+LM	PC+AM	PC+LPC	PC+ LPS
average($\hat{\varepsilon}_i^2$)	4'593	4'967	1'732	1'430	1'252
median($\hat{\varepsilon}_i^2$)	1'049	1'124	104	52	49

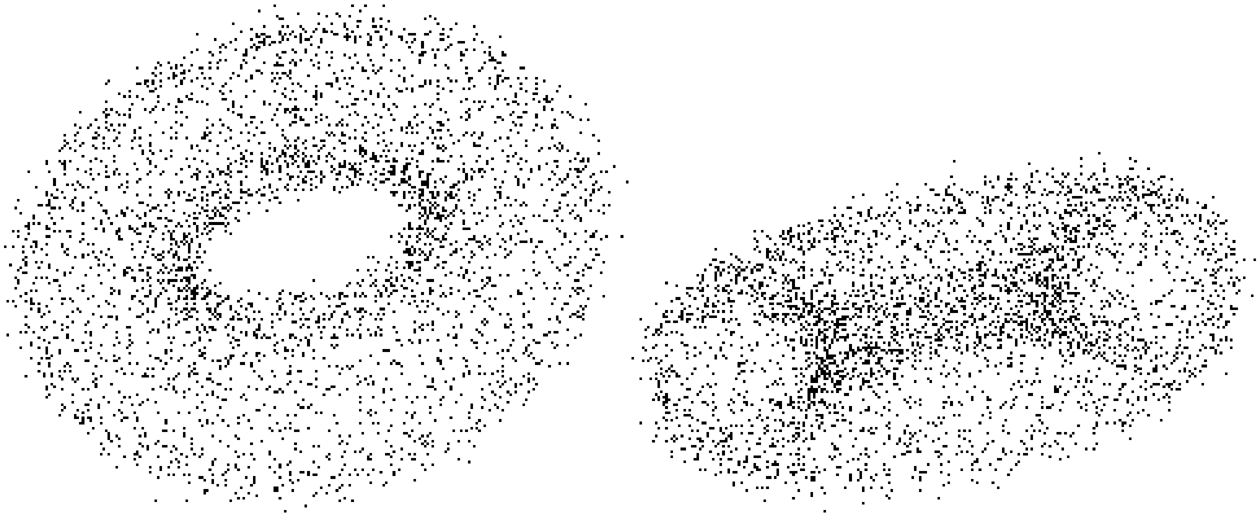
- Metallicity

prediction error	LM	PC+LM	PC+AM	PC+LPC	PC+ LPS
average($\hat{\varepsilon}_i^2$)	2.601	3.084	2.849	3.070	3.067
median($\hat{\varepsilon}_i^2$)	1.287	1.821	1.671	1.859	1.323

The torus

- Simulate a torus:

```
> t <- 0:60/30
> t <- cbind(rep(t,each=length(t)),rep(t,length(t)))
> t <- t + 0.01 * rnorm(length(t))
> data <- cbind(sin(pi*t[,1])*(1-0.4*cos(pi*t[,2])),
               cos(pi*t[,1])*(1-0.4*cos(pi*t[,2])),0.4*sin(pi*t[,2]))
> data <- data + 0.05*rnorm(length(data))
```

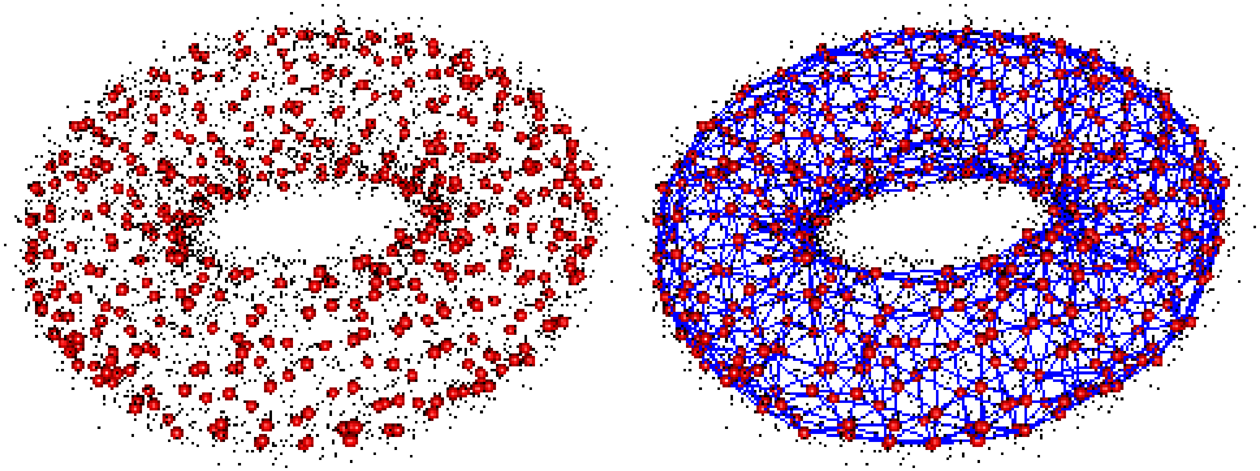


The torus (cont.)

- Fit the LPS:

```
> lpm(data, h=25)
```

- LPS vertices (left), and triangle mesh (right):

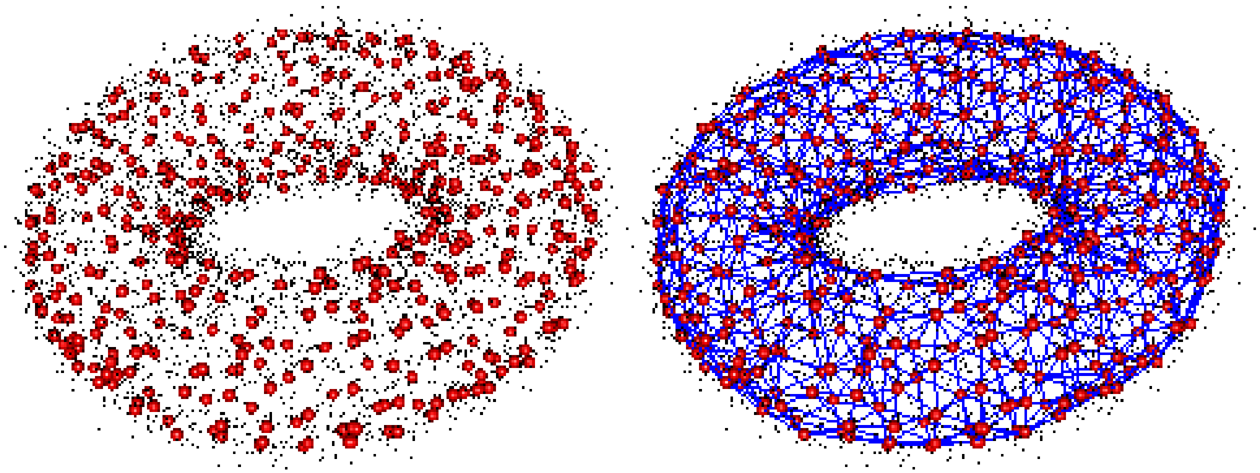


The torus (cont.)

- Fit the LPS:

```
> lpm(data, h=25)
```

- LPS vertices (left), and triangle mesh (right):



- any applications beyond coffee breaks?

Conclusion

- After parametrization through cubic splines, LPCs can be used for dimension reduction provided that
  - the intrinsic (topological) dimensionality of the data cloud is close to 1, or, at least,
  - the projections on the curve are informative for the target variable.
- Extension of LPCs to LPSs works by considering the building block "triangles".
- Regression on surfaces is (yet) done via a discrete kernel approach (due to a lack of parametrization).
- Extendable to local principal manifolds (LPMs) of arbitrary dimension > 2 by replacing "triangles" with suitable "tetrahedrons".
- R package **LPCM** in development, available on request from authors.

References

Hastie & Stuetzle (1989): Principal Curves. *JASA* **84**, 502–516.

Hastie, Tibshirani, & Friedman (2001): The Elements of Statistical Learning. Springer.

Tibshirani (1992): Principal Curves Revisited. *Statistics and Computing* **2**, 183–190.

Kégl, Krzyzak, Linder,& Zeger (2000): Learning and Design of Principal Curves. *IEEE Transactions Patt. Anal. Mach. Intell.* **24**, 59–74.

Delicado (2001): Another Look at Principal Curves and Surfaces, *Journal of Multivariate Analysis* **77**, 84–116.

References (cont.)

Einbeck, Tutz & Evers (2005): Local principal curves. *Statistics and Computing* **15**, 301–313.

Einbeck, Evers & Bailer-Jones (2008): Representing complex data using localized principal components with application to astronomical data. In Gorban et al. (Eds): Principal Manifolds for Data Visualization and Dimension Reduction; *Lecture Notes in Computational Science and Engineering* **58**, 180–204.

Einbeck, Evers & Hinchliff (2009): Data compression and regression based on local principal curves. In Fink et al. (Eds): Advances in Data Analysis, Data Handling, and Business Intelligence, Heidelberg, pp. 701–712, Springer.

Einbeck, Evers & Powell (2009): Data compression and regression through local principal curves and surfaces. *Technical report, submitted to IJNS.*

LPCM: Local principal curves and manifolds. R package version 0.36-2, available on request from authors.