Data compression and regression based on local principal curves and manifolds

Jochen Einbeck

Department of Mathematical Sciences, Durham University

jochen.einbeck@durham.ac.uk

joint work with Ludger Evers (University of Glasgow),

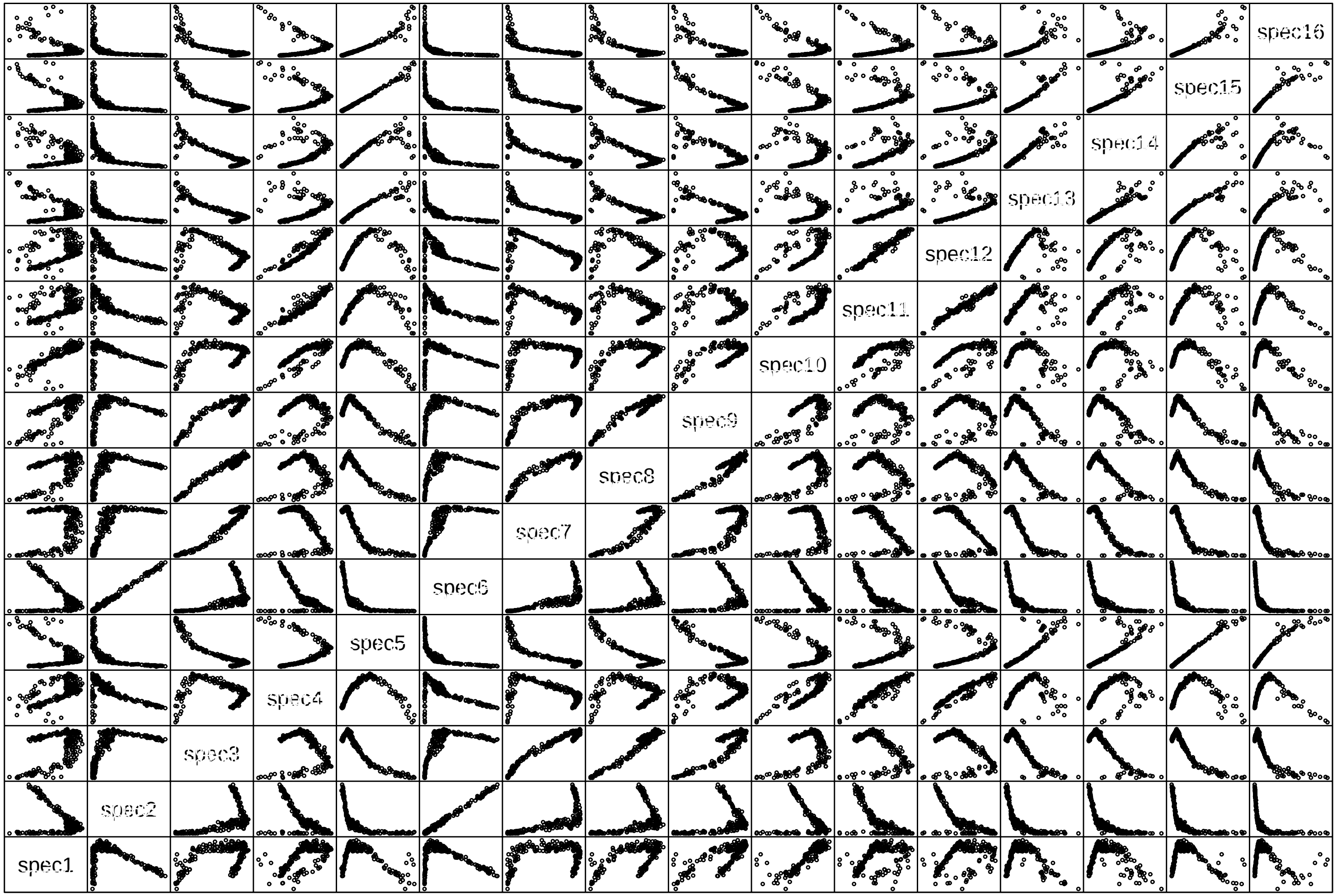
Durham, 14th of April 2010

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
  - Classify objects (star, galaxy, quasar,...)
  - Determine astrophysical parameters ("APs": temperature, metallicity, gravity) from spectroscopic data (photon counts at certain wavelengths).
- Group "Astrophysical parameters" at MPIA Heidelberg is in charge of developing the necessary statistical toolbox.
- 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

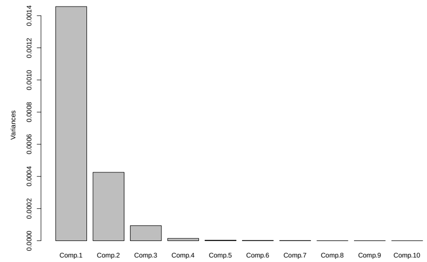
- Try linear model for the temperature, using training sample of size $n = 1000$:

```
> 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:



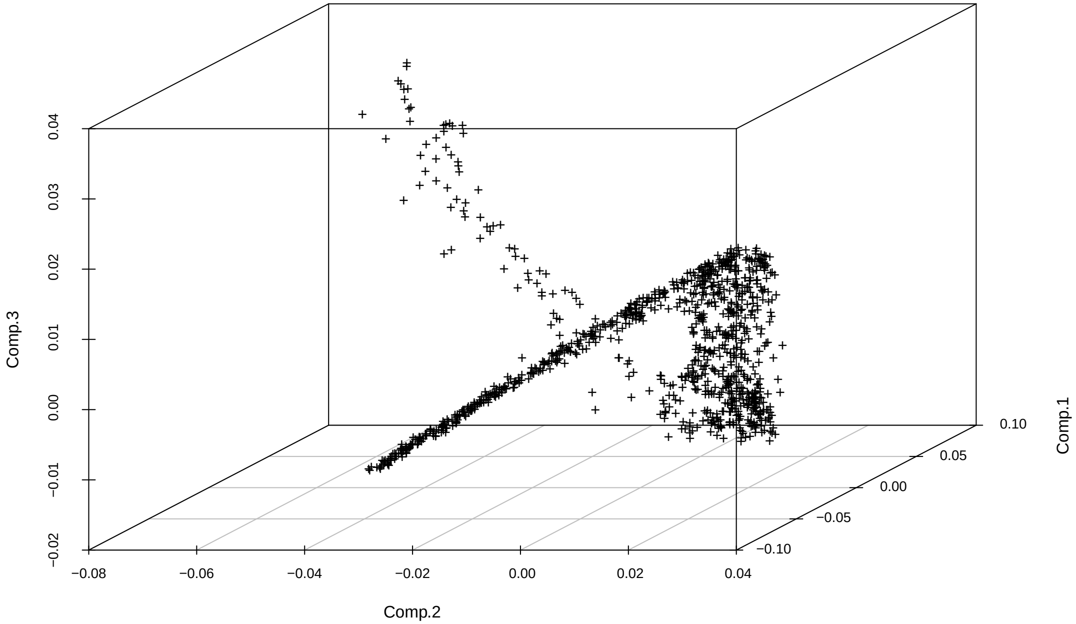
- Three principal components appear to be sufficient.

```
> 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 acceptable...

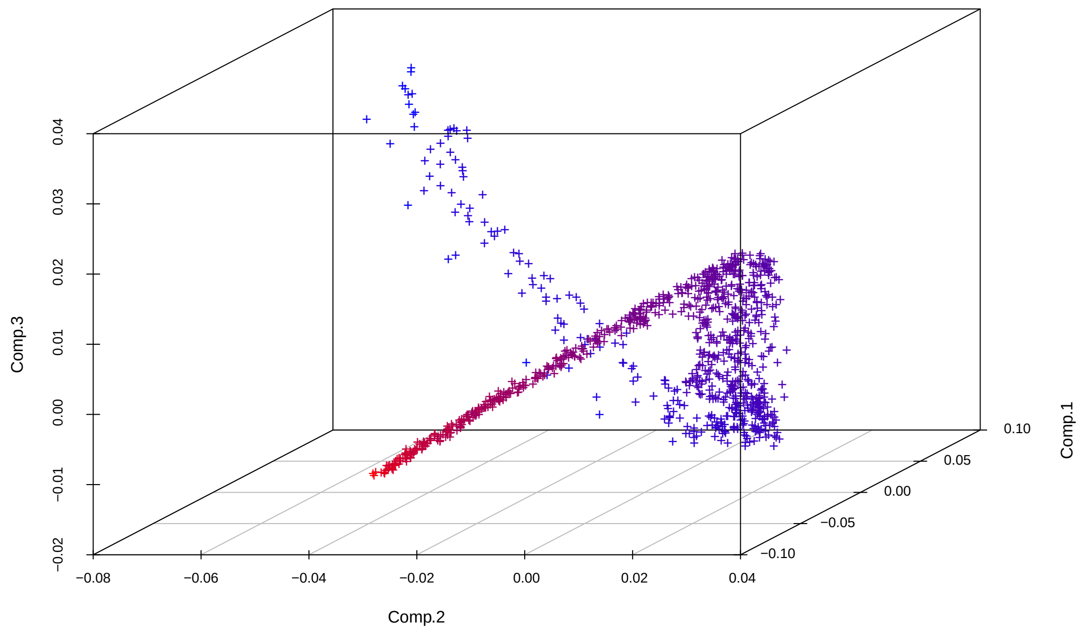
Principal component scores

- We plot the the first three principal component scores



Principal component scores

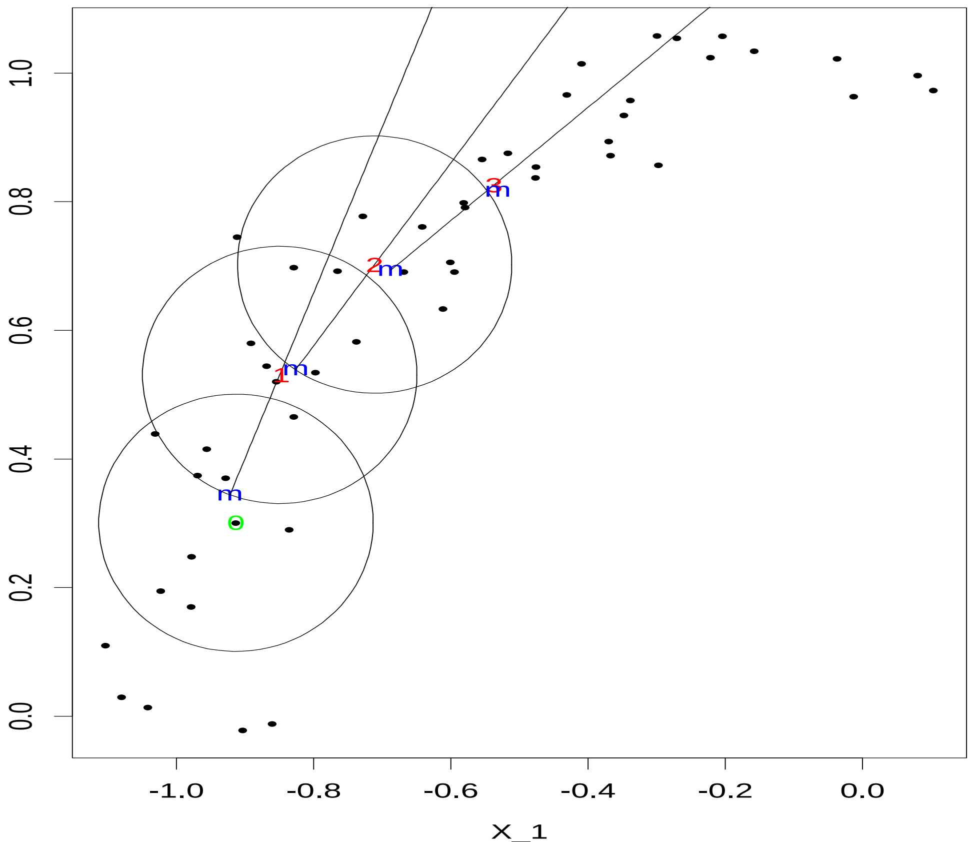
- We plot the the first three principal component scores and shade higher temperatures red.



- Actually, we seem to need only *one* parameter if we were able to lay a smooth curve through the data cloud, and parametrize it.
- This is a task for principal curves, "smooth curves through the middle of a data cloud" (Hastie & Stuetzle, 1989).

Local principal curves (LPCs)

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

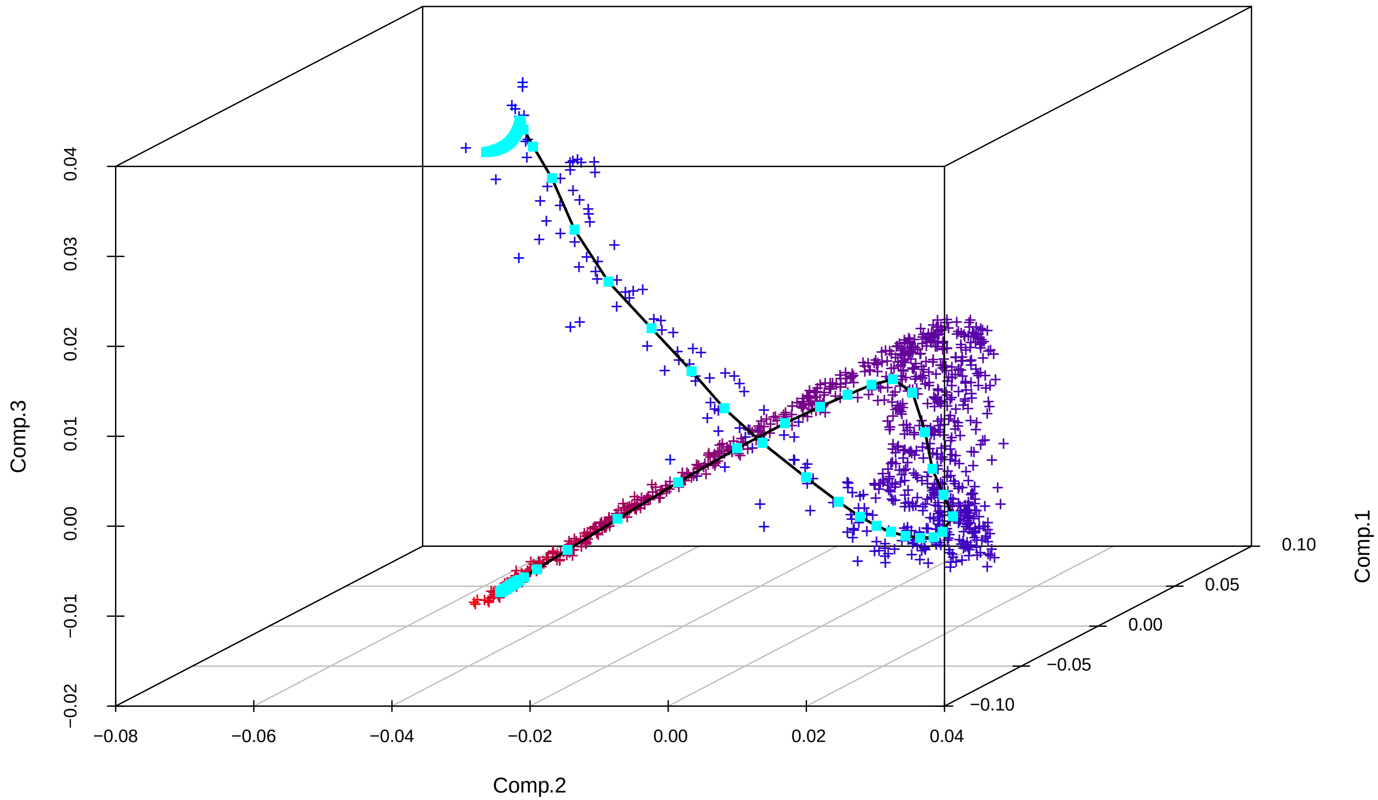


0: starting point,
$\boldsymbol{m}$: points of the LPC,
$1, 2, 3$: enumeration of steps.

Step 1: Fitting the LPC

- LPC through principal component scores of photon counts, with local centers of mass $\boldsymbol{m}$ (sky blue squares):

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



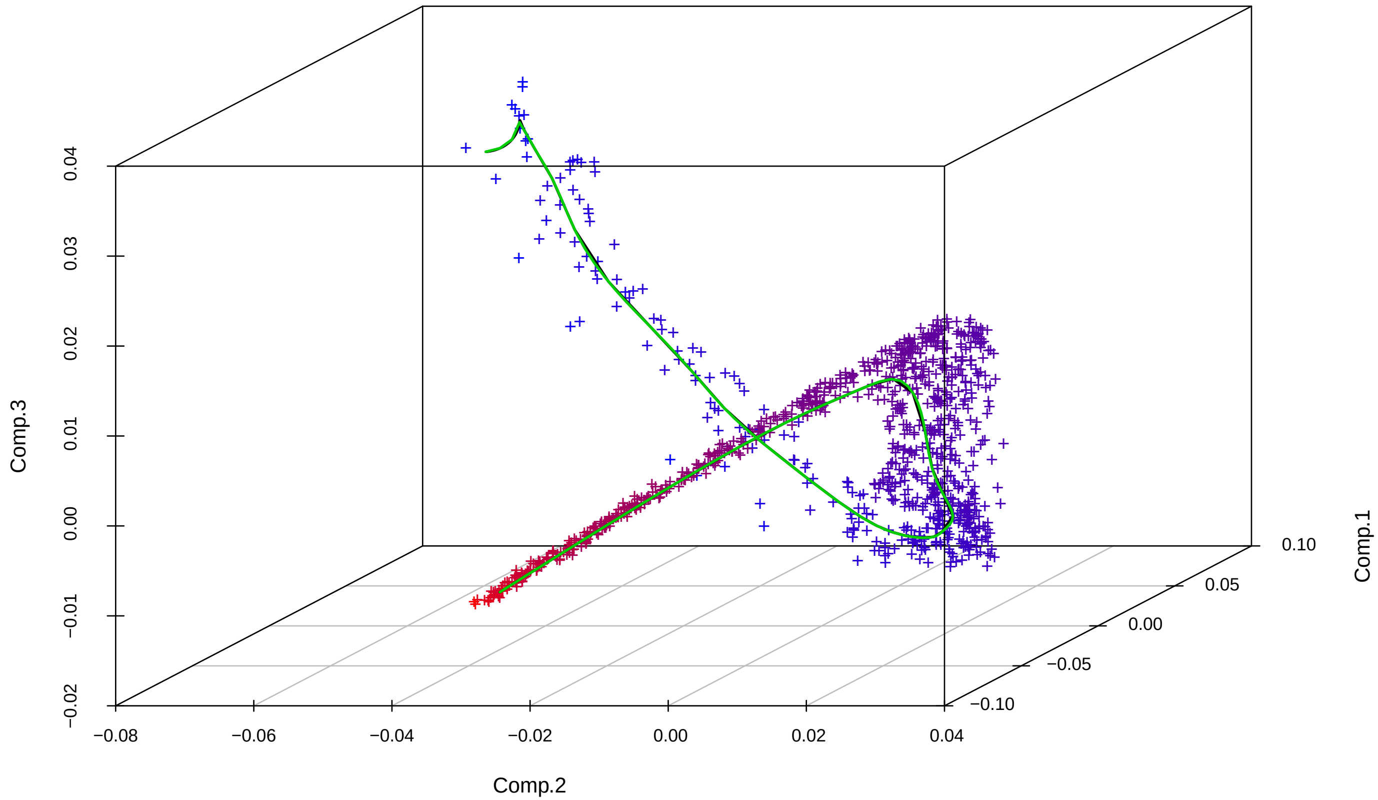
Step 2: Parametrization

- 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 $\boldsymbol{m} \in \mathbb{R}^d$.
- For each component m_j, $j = 1, \ldots, d$, use a natural cubic spline to construct functions $m_j(s)$, yielding together a function $(m_1, \ldots, m_d)(s)$ representing the LPC (no smoothing involved here!).
- Recalculate the parametrization along the curve through the arc length of the spline function,

$$t = \int_0^s \sqrt{(m_1'(u))^2 + \ldots + (m_p'(u))^2}\, du$$

Step 2: Parametrization (cont.)

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

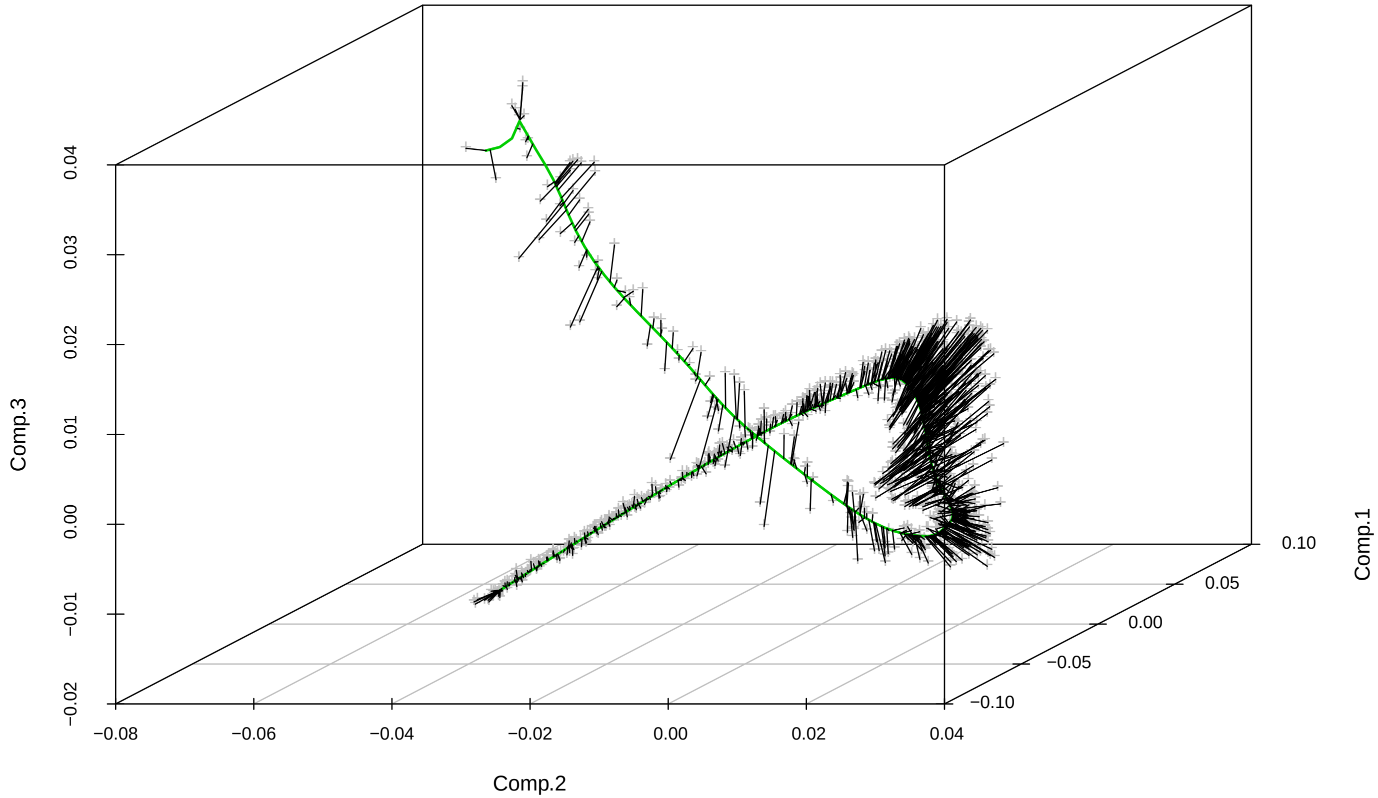


- The spline function (—) is almost indistinguishable from the original LPC (—).

Step 3: Projection

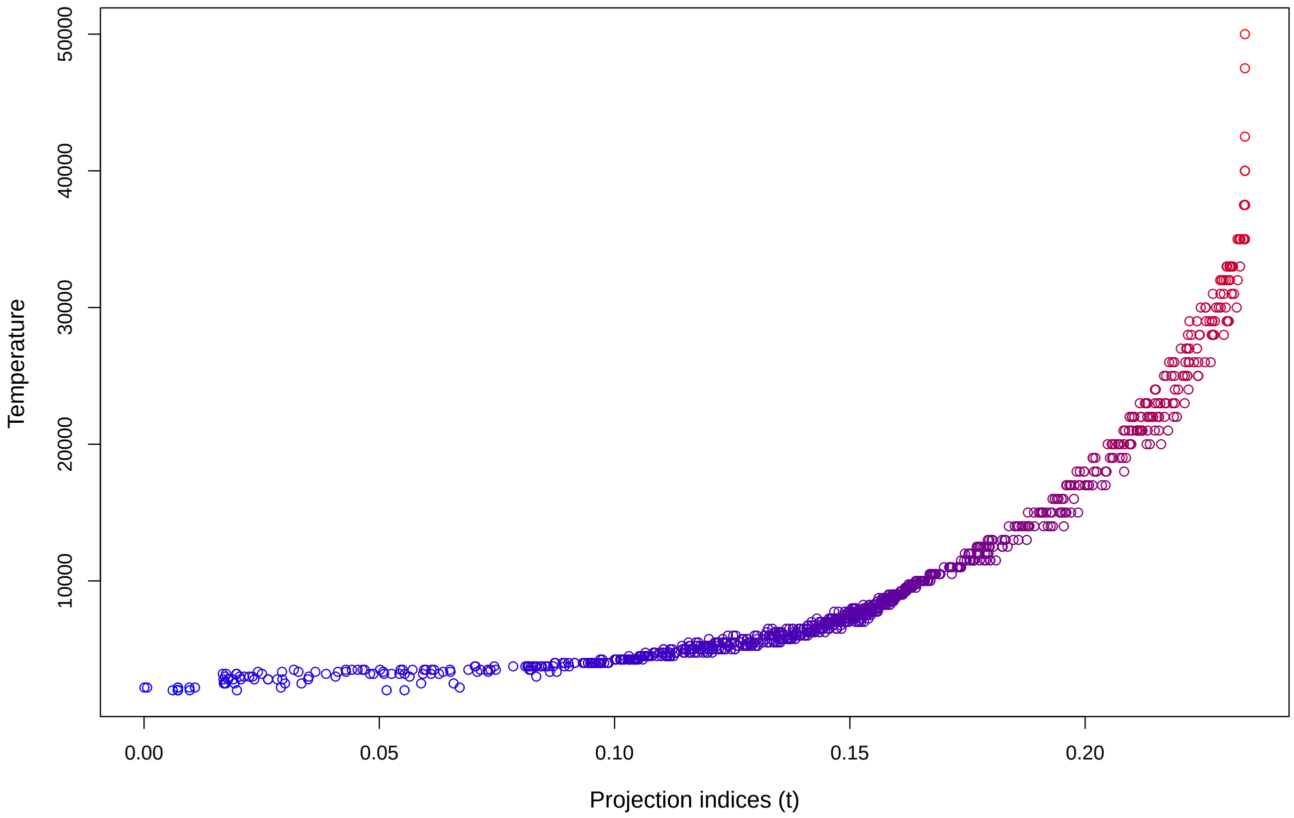
- Each point $\boldsymbol{x}_i \in \mathbb{R}^d$ is projected on the point of the curve nearest to it, yielding the corresponding projection index t_i

```
> 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.

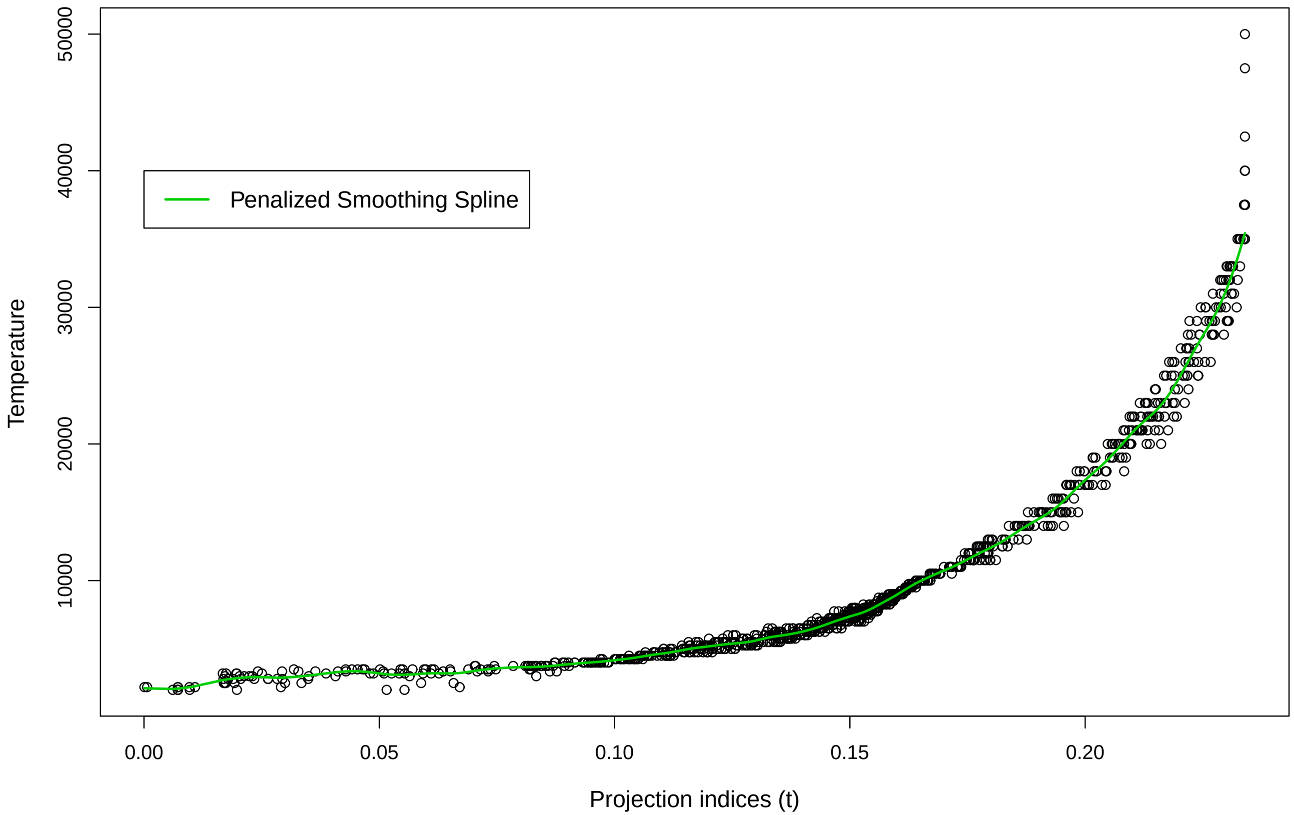


- This is now a simple one-dimensional regression problem,

$$y_i = g(t_i) + \varepsilon_i.$$

Step 4: Regression

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



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

$$y_i = g(t_i) + \varepsilon_i.$$

Prediction

- For a new observation $\boldsymbol{x}_{new} \in \mathbb{R}^d$, prediction proceeds as follows:
  - Project $\boldsymbol{x}_{new}$ onto the LPC, giving t_{new}.
  - Compute $\hat{y}_{new} = \hat{g}(t_{new})$ from the fitted regression model.
- Comparison: We sample $n' = 1000$ test data from the remaining $8286 - 1000$ observations and observe the prediction error:

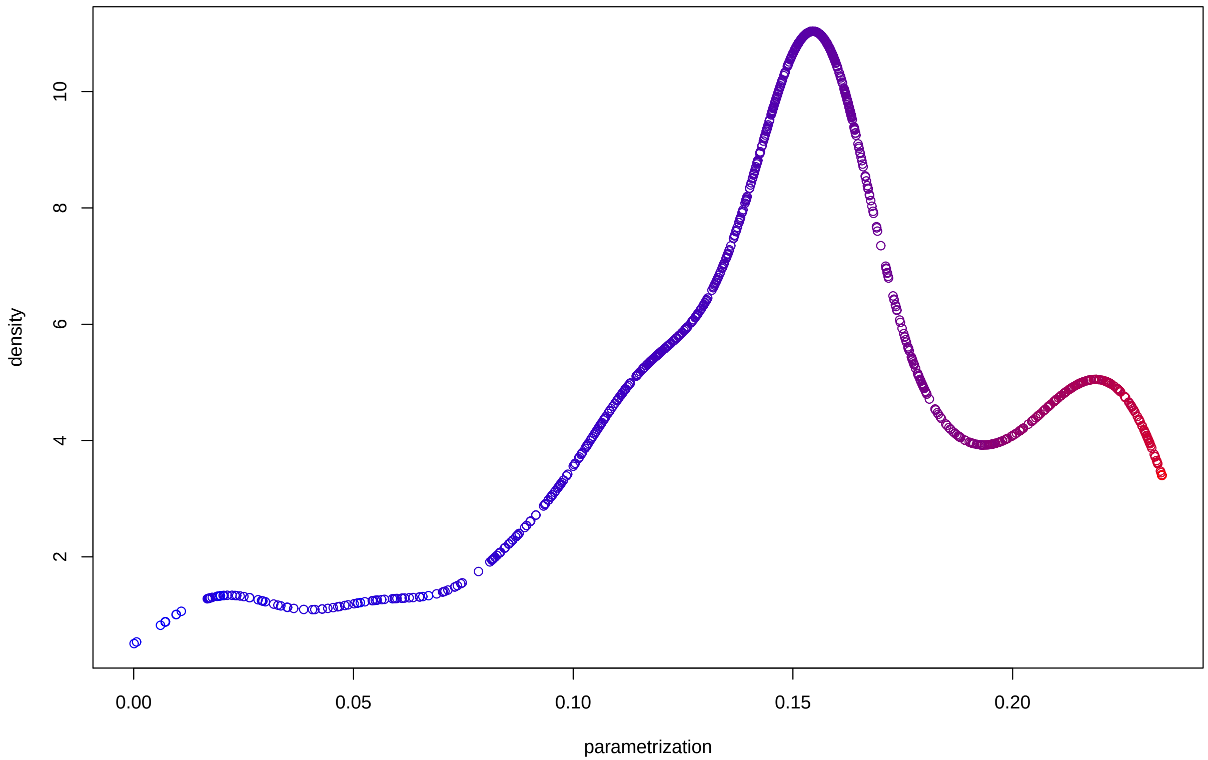
prediction error $/10^3$	LM	PC+LM	PC+AM	PC+LPC
average($\hat{\varepsilon}_i^2$)	4593	4967	1732	1430
median($\hat{\varepsilon}_i^2$)	1049	1124	104	52

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

Density estimation

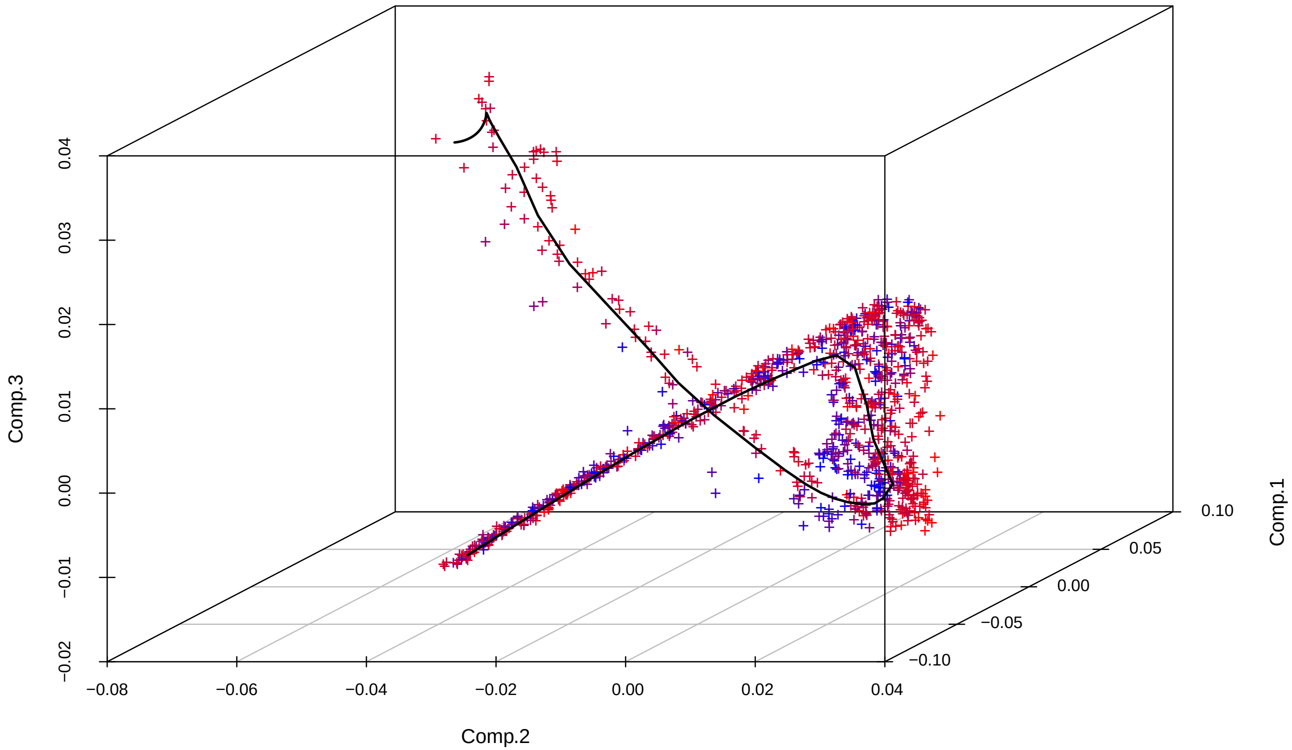
- Having now the projection indexes t_i, $i = 1, \ldots, n$, this can be easily used for other purposes such as "density estimation along the principal curve":

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



Limits of one-dimensional data summaries

- Look at "metallicity"



- The relevant information seems to be orthogonal to the principal curve!
- Would a *principal surface* help?

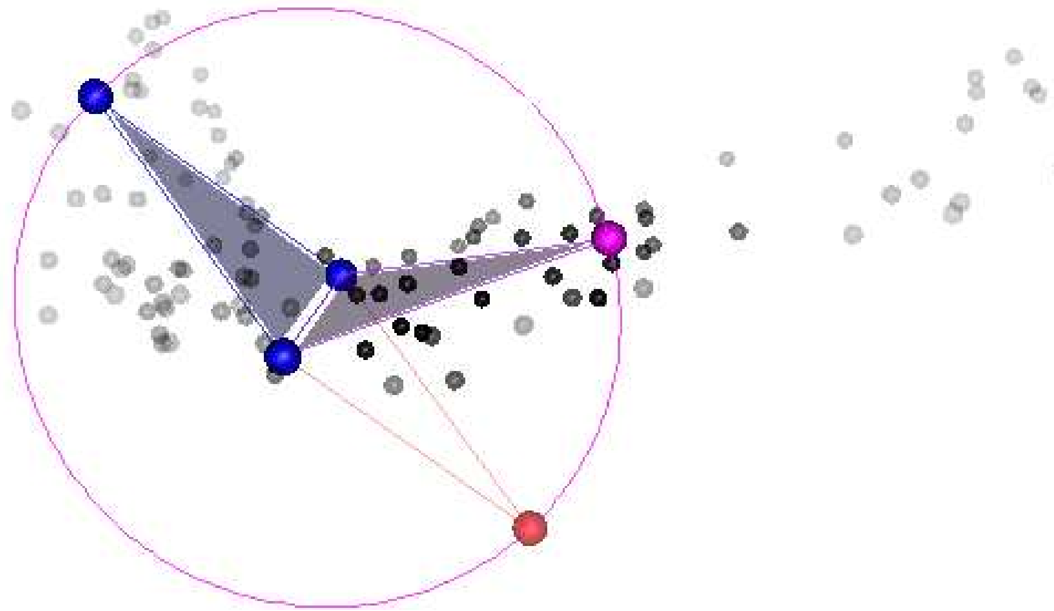
Local principal surfaces

- 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.

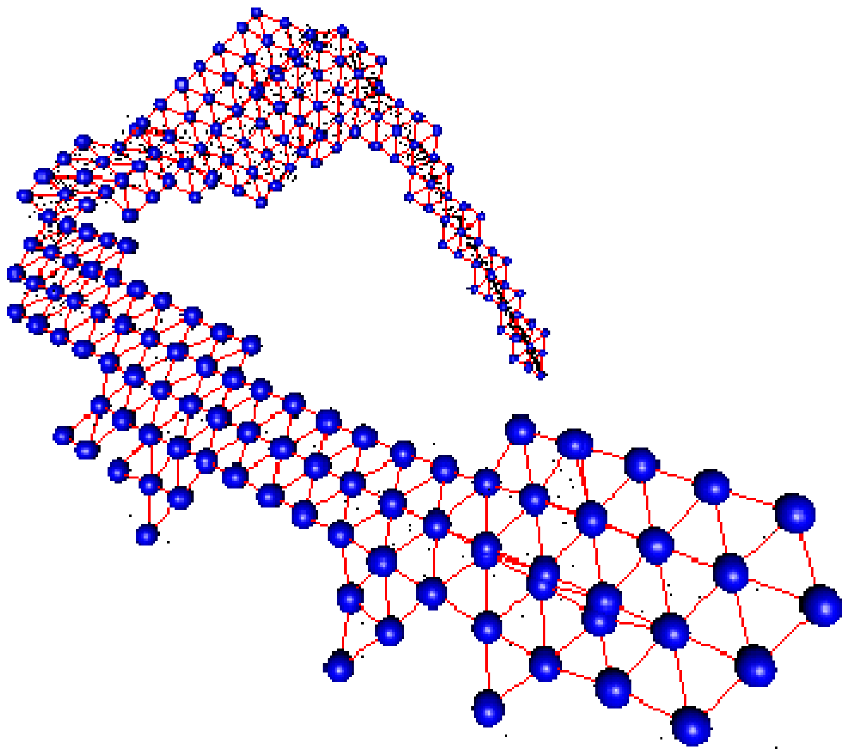
Local principal surfaces (cont.)

- 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.
- However, we may use *distances* instead: 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.
- Assign local weights via discrete distance-based kernel

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

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

Regression on the surface (cont.)

The entire fitting process is summarized as follows:

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

(II) Assign each data point $\boldsymbol{x}_i, i = 1, \ldots, n$ 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 g_r on triangle r is given by

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

Simulation study

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

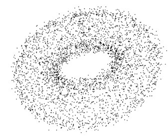
- Temperature

prediction error $/10^3$	LM	PC+LM	PC+AM	PC+LPC	PC+LPS
average($\hat{\varepsilon}_i^2$)	4593	4967	1732	1430	1252
median($\hat{\varepsilon}_i^2$)	1049	1124	104	52	49

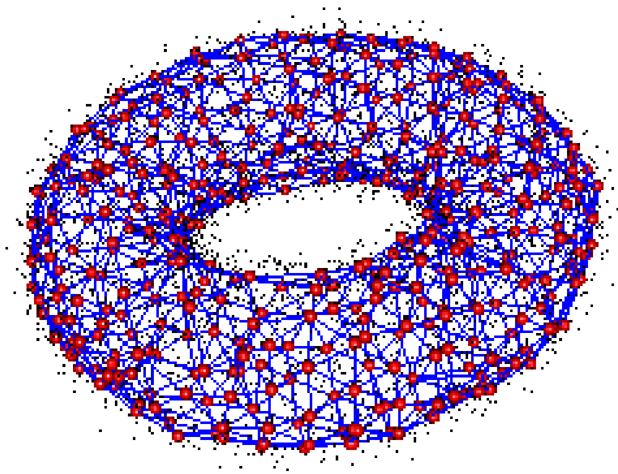
- Metallicity

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

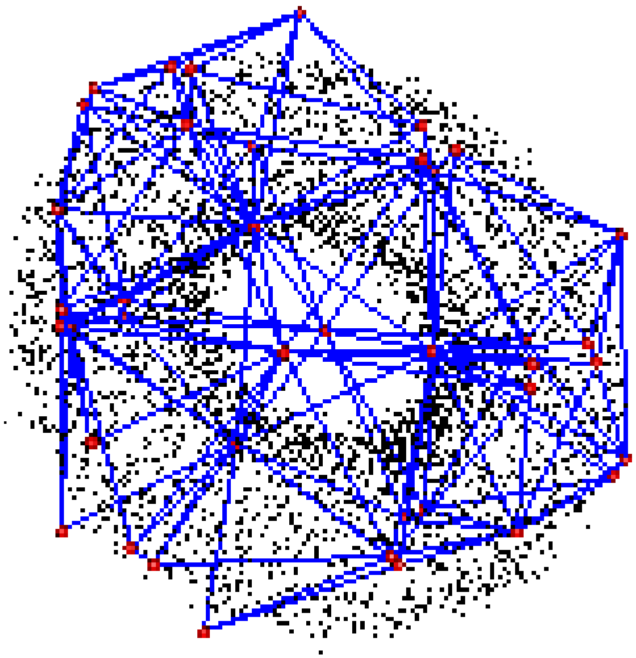
Manifolds of higher dimension?

- The techniques extend to local principal manifolds (LPMs) of higher dimensions by using tetrahedrons instead of triangles.
- Visualization of course tricky....
- Slightly contrived example: 3d-Torus , with:

2d surface



3d manifold



Conclusion

- Principal curves or surfaces can be used for dimension reduction provided that
  - the intrinsic (topological) dimensionality of the data cloud is close to 1 or 2, respectively,
  - or, at least, the projections are informative for the target variable.
- Regression on surfaces is (yet) done via a discrete kernel approach (due to a lack of parametrization).
- *Direct* LPC/ LPS regression (without preliminary PCA step) in principle possible.
- Extendable to local principal manifolds (LPMs) of arbitrary dimension > 2 by replacing "triangles" with suitable "tetrahedrons" or "simplices".

References

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

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 (2010): 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.

LPCM: Local principal curves and manifolds. R package version 0.36-3, available on request from authors.