Some useful tools of ML for astronomy data analysis

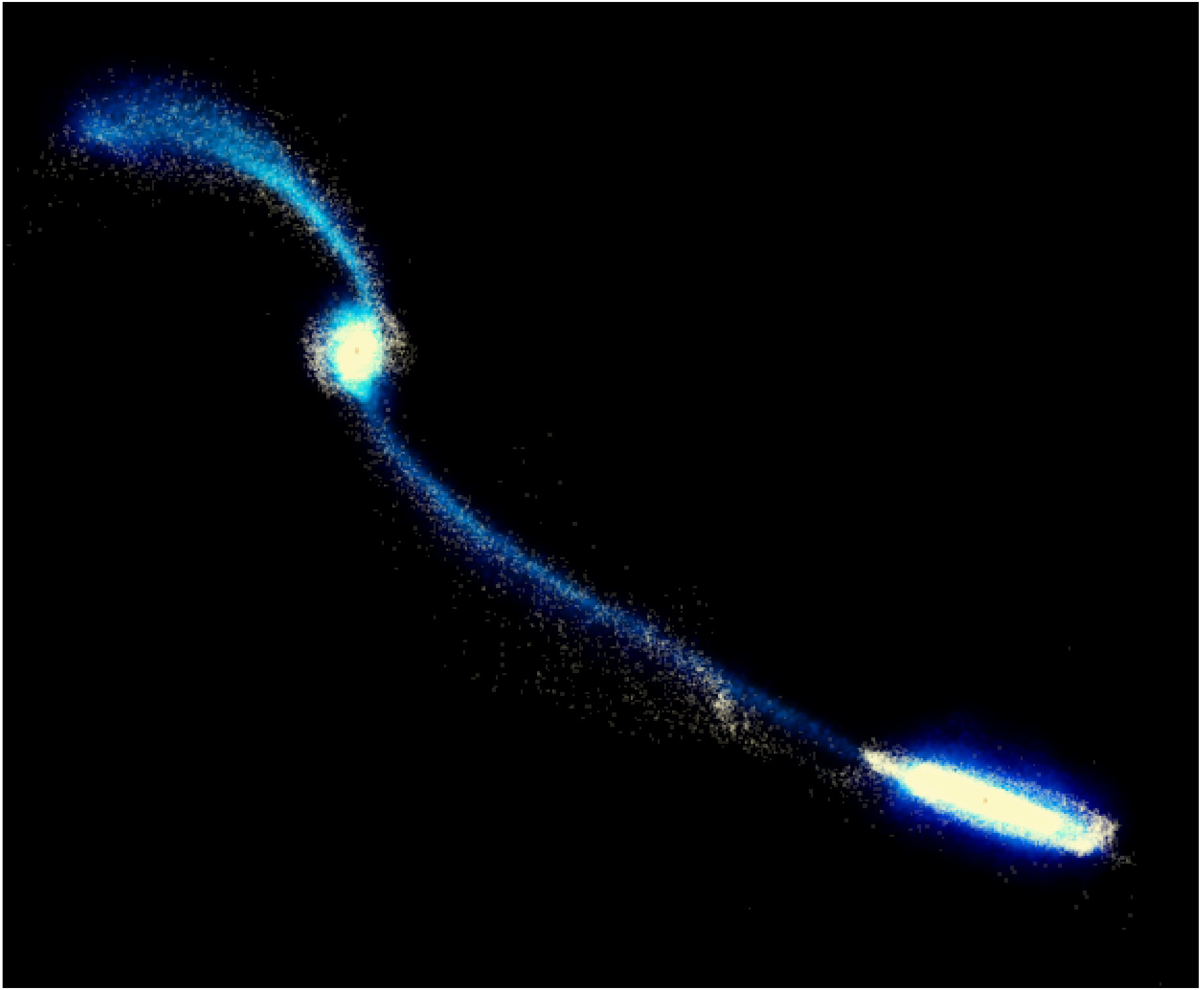
ISYA 41
Socorro
2018

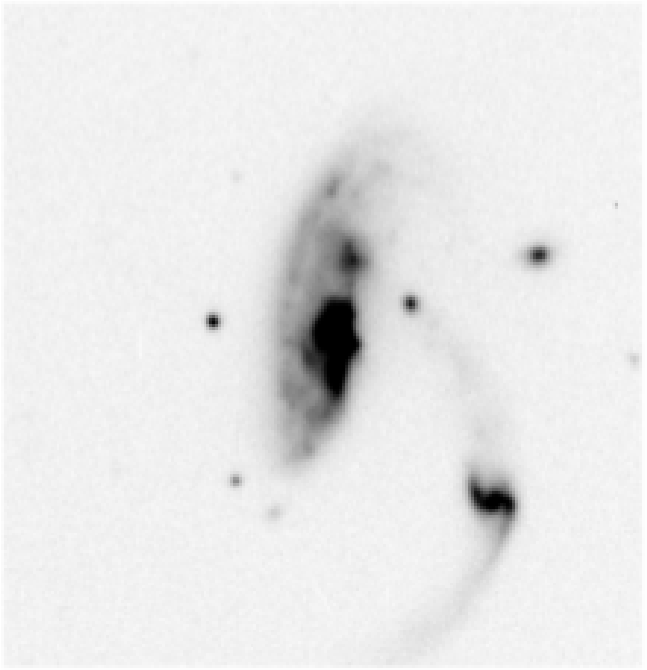
Juan Carlos Muñoz Cuartas
Instituto de Física
Universidad de Antioquia

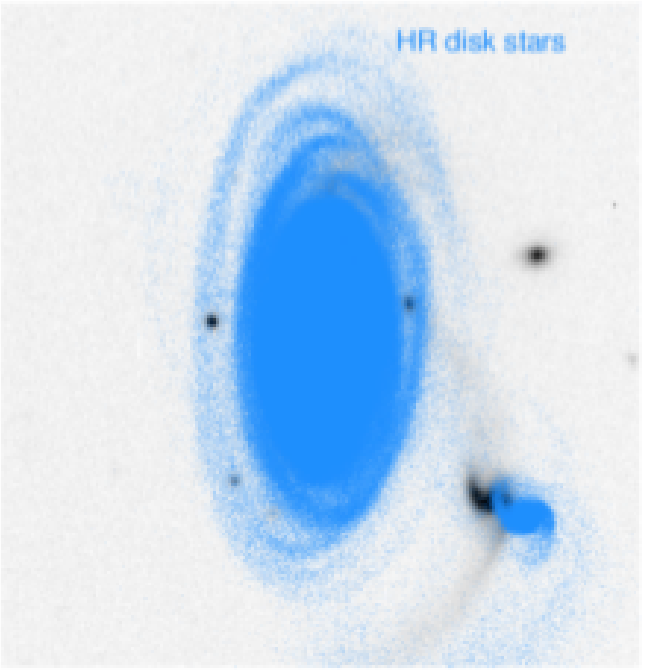
Juan Carlos Muñoz Cuartas

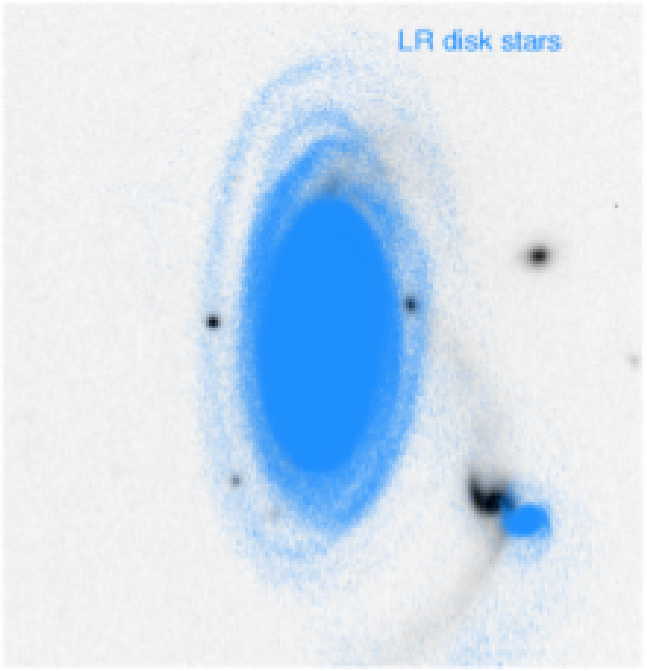
- Physicist, Msc. Universidad de Antioquia
- PhD Astrophysics Potsdam Universität, Leibniz-Institut Für Astrophyk Potsdam
- F.T. Professor at Universidad de Antioquia
- Coordinator (??) Research Group on Astrophysics

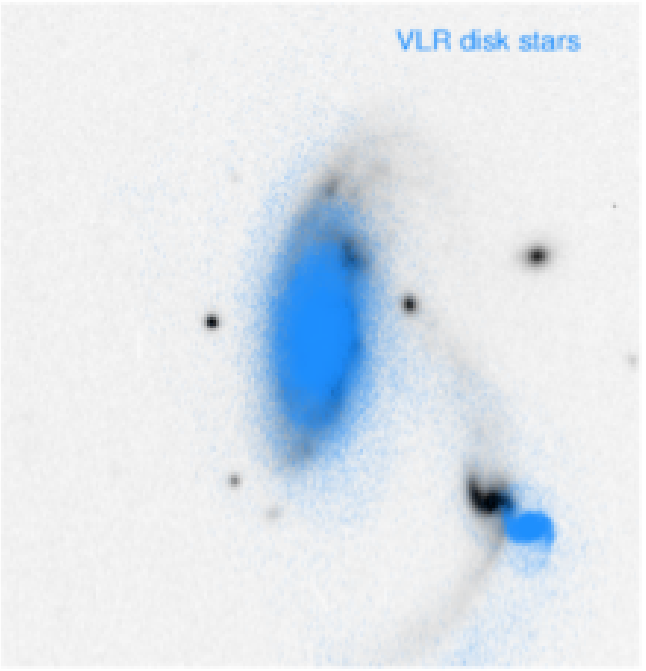
- **Galaxy formation and evolution.**
- **Semi-analytic models of galaxy formation**
- **Galaxy formation in hydrodynamical simulations**
- **Dynamics of interacting galaxies**
- **N-body simulations (Cosmological and non cosmological)**
- **Large scale structure of the universe**






HR disk stars

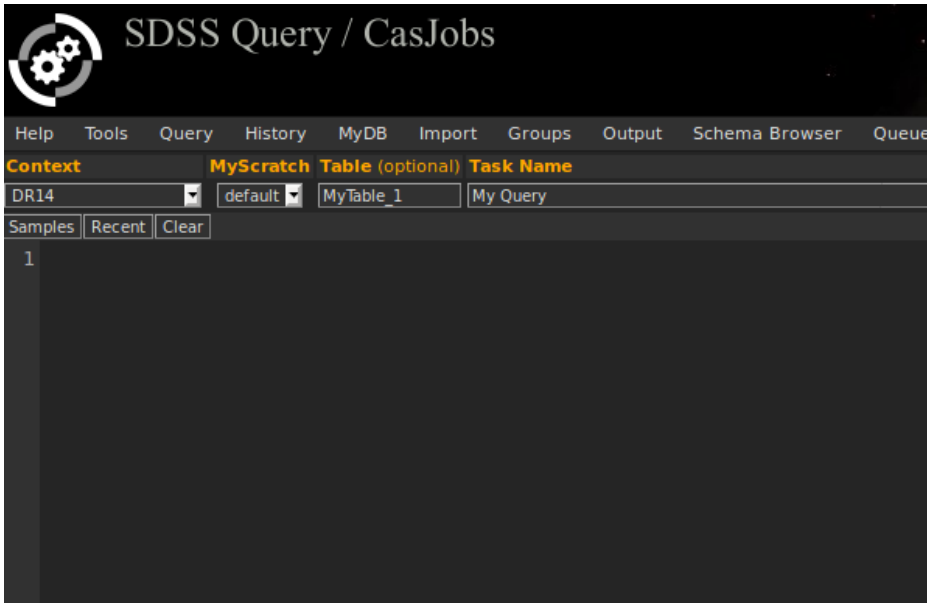

LR disk stars

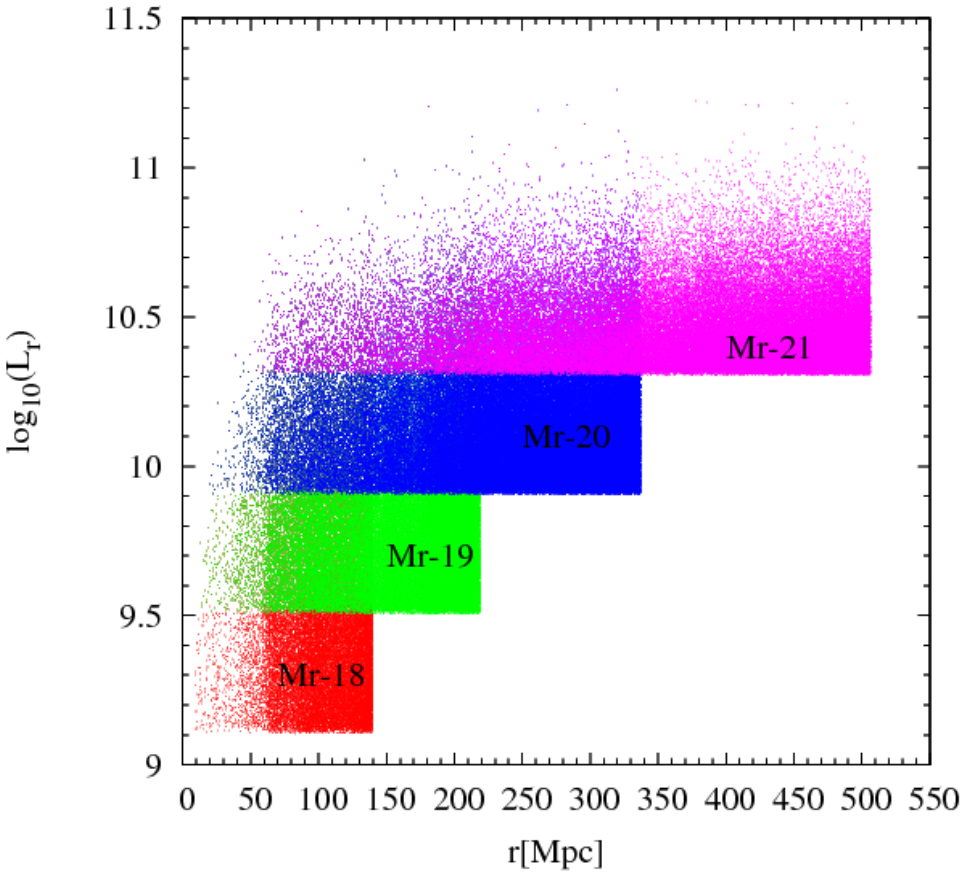

VLR disk stars

Taking data from SDSS

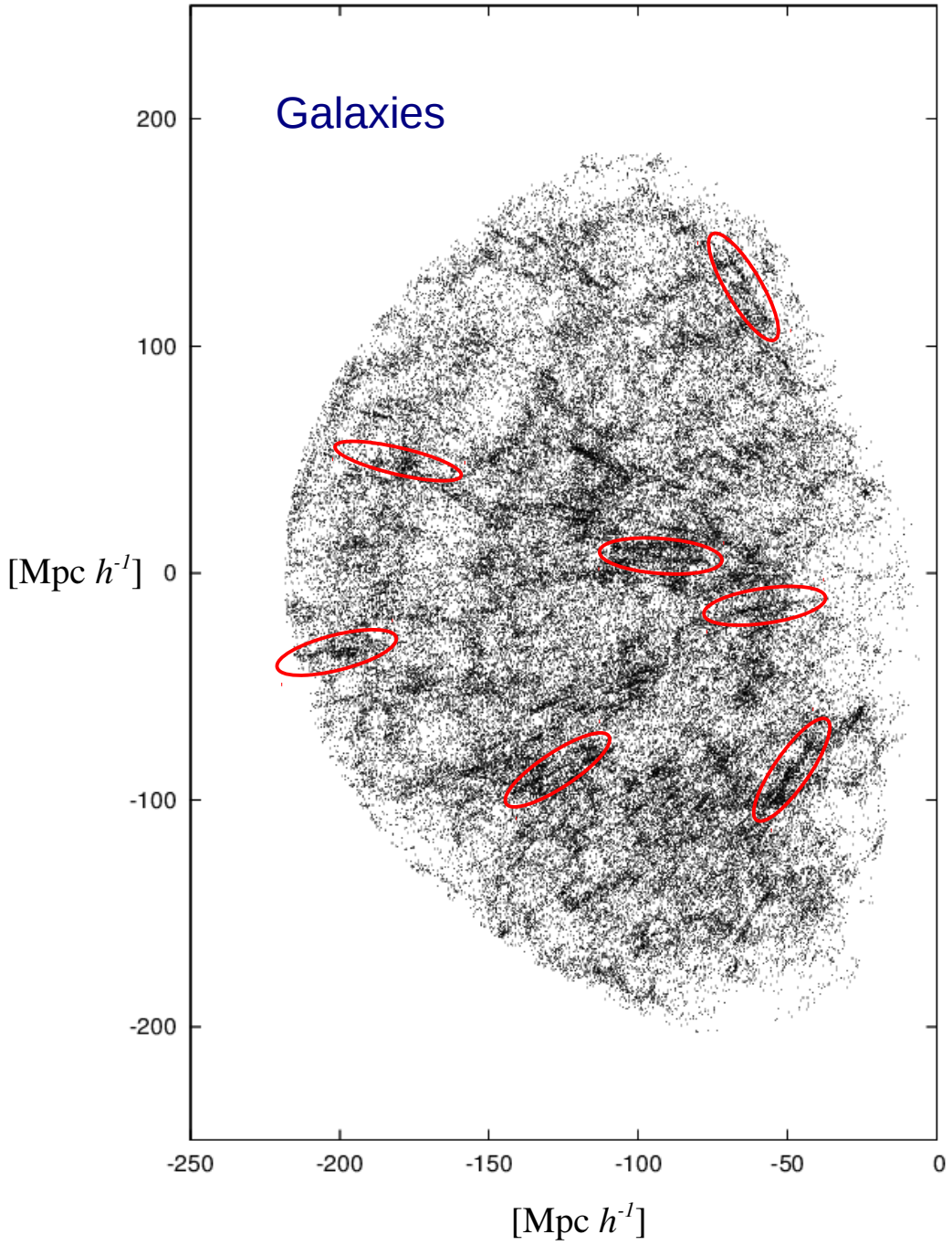
Four galaxy samples:

- Only galaxies in the MGS
- Only galaxies with good redshift determination
- K-corrections (Blanton et al. 2007)
- Evolution correction (Blanton et al. 2003)





Name	z_{min}	z_{max}	N_{gals}	$L_{eq}\ h^{-1}\,\mathrm{Mpc}$
Mr-18	0.002	0.047	50986	130
Mr-19	0.002	0.074	108546	200
Mr-20	0.002	0.115	155890	310
Mr-21	0.002	0.175	97064	465
All	0.002	0.2	412486	



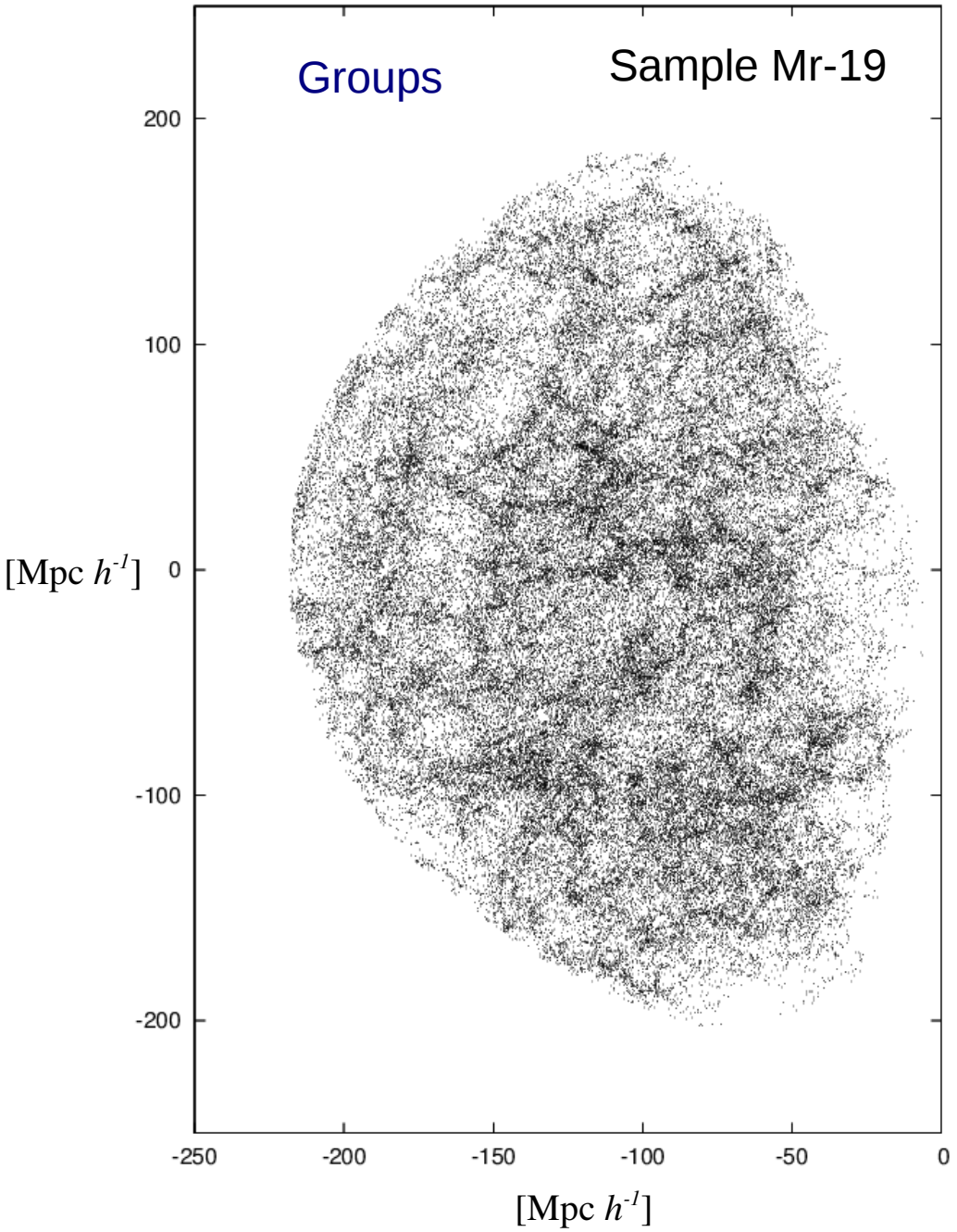
From galaxy redshift surveys we can measure:

+ Fluxes
+ Positions in redshift space

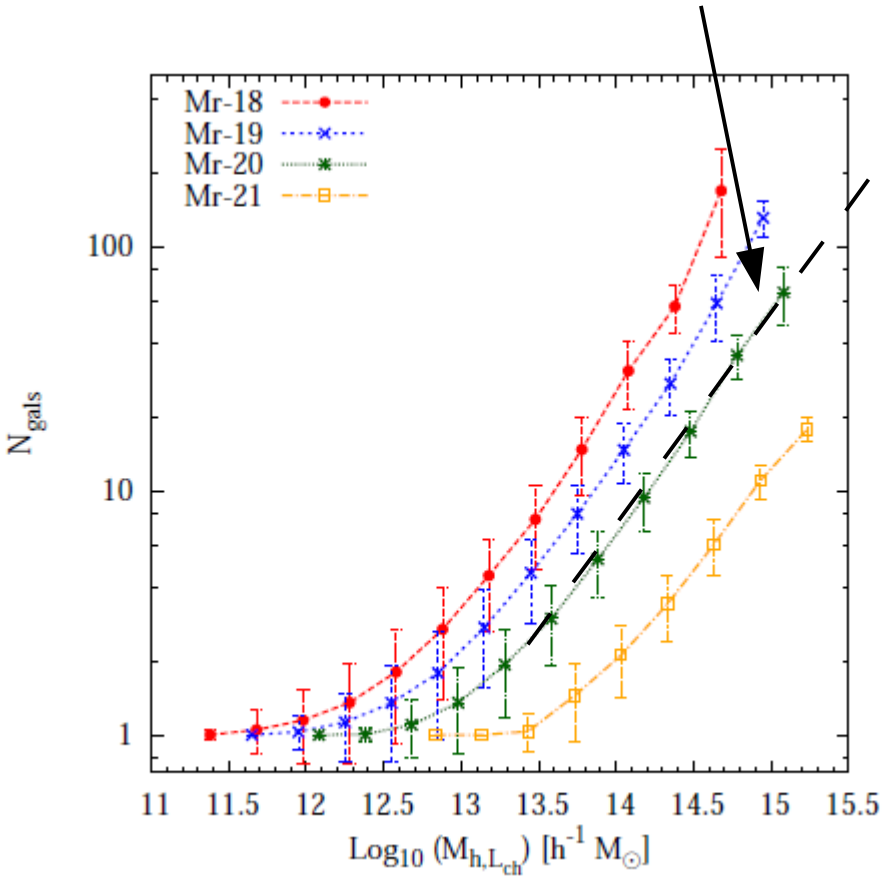
Redshift:
Distance to the galaxy + peculiar motion

Redshift space distortions

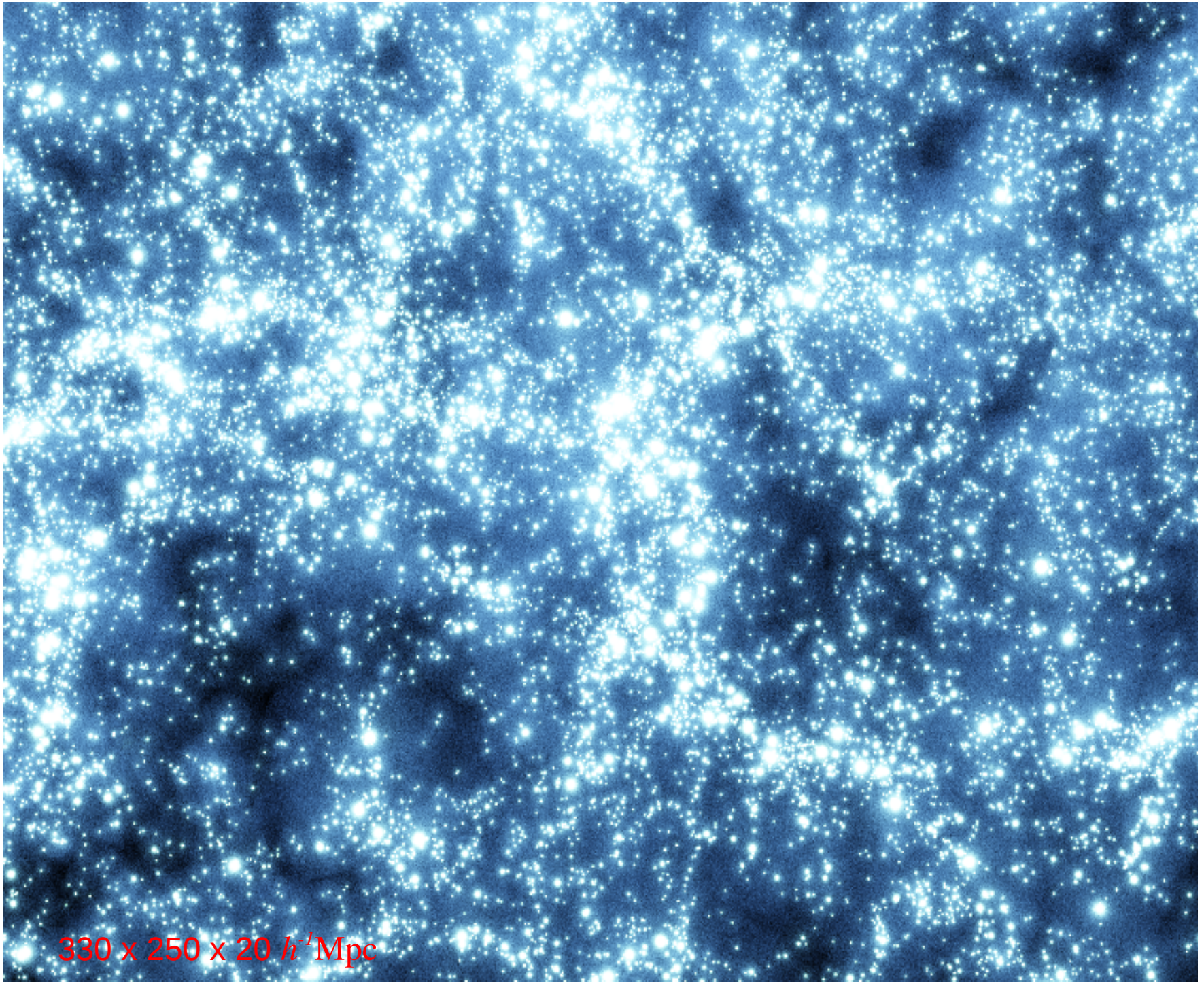
Main result: A catalog of galaxy groups: Positions, memberships, and estimated dark mater halo mass for the SDSS-DR7

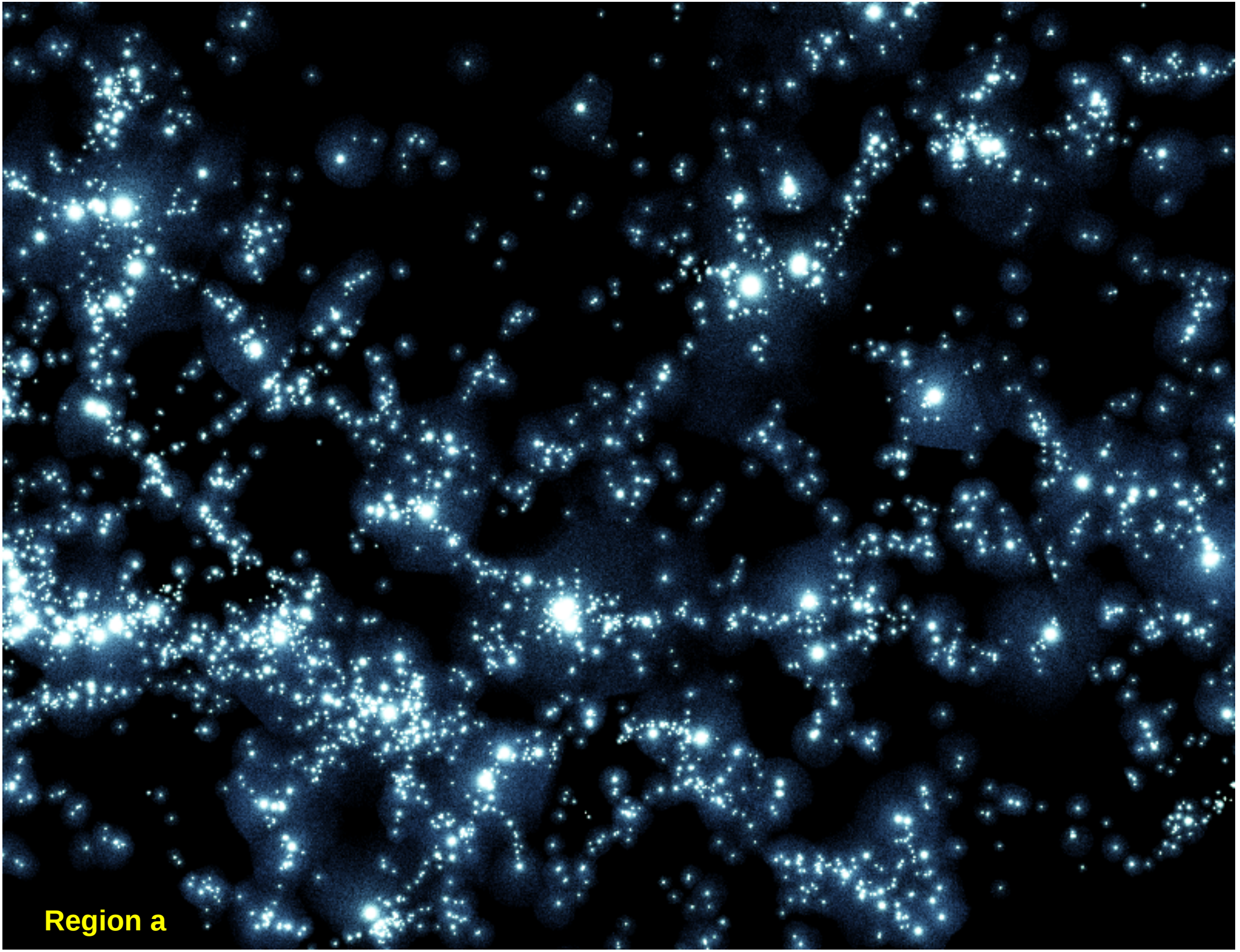


Power law on M (HOD)
(Van den Bosch et al. 2007)

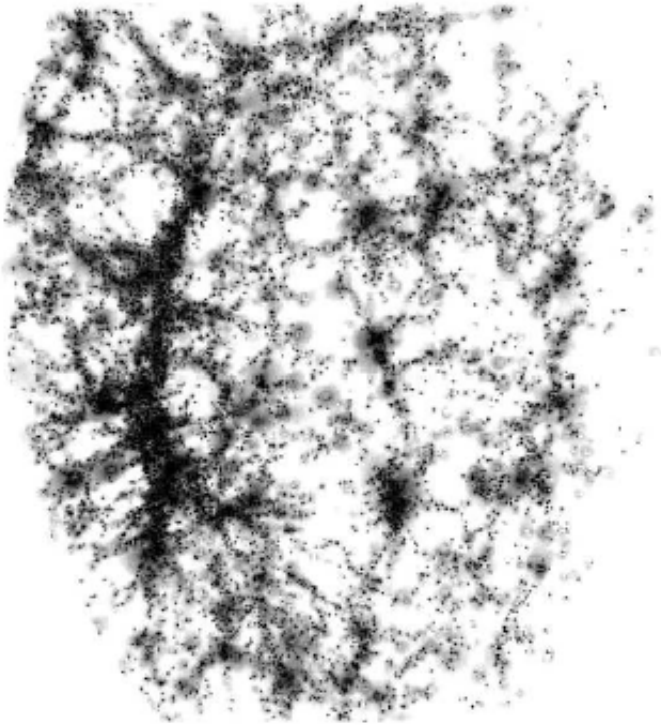


Main Result: Reconstruction of the density field SDSS



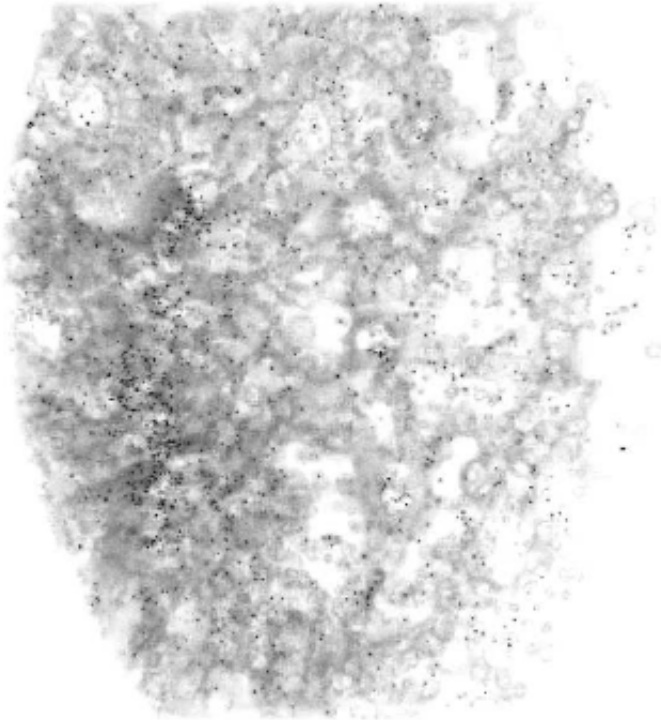

Region a

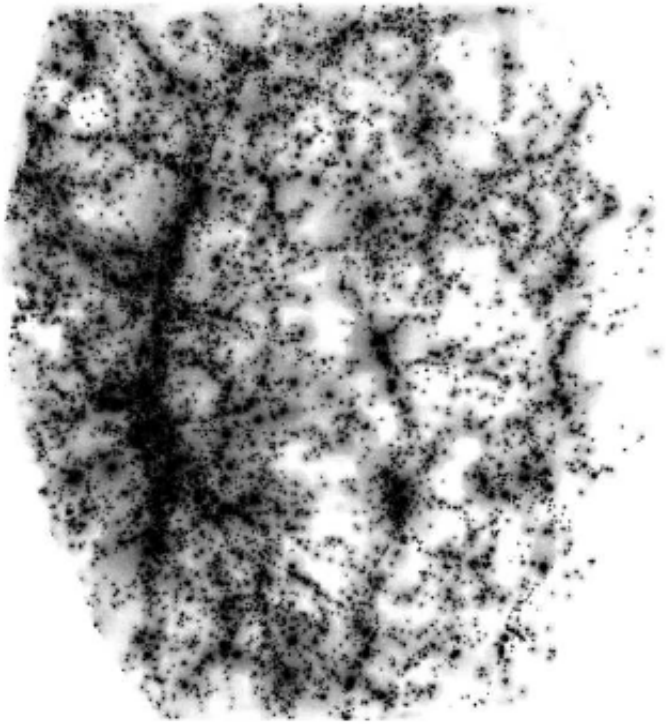
Filament



Peak

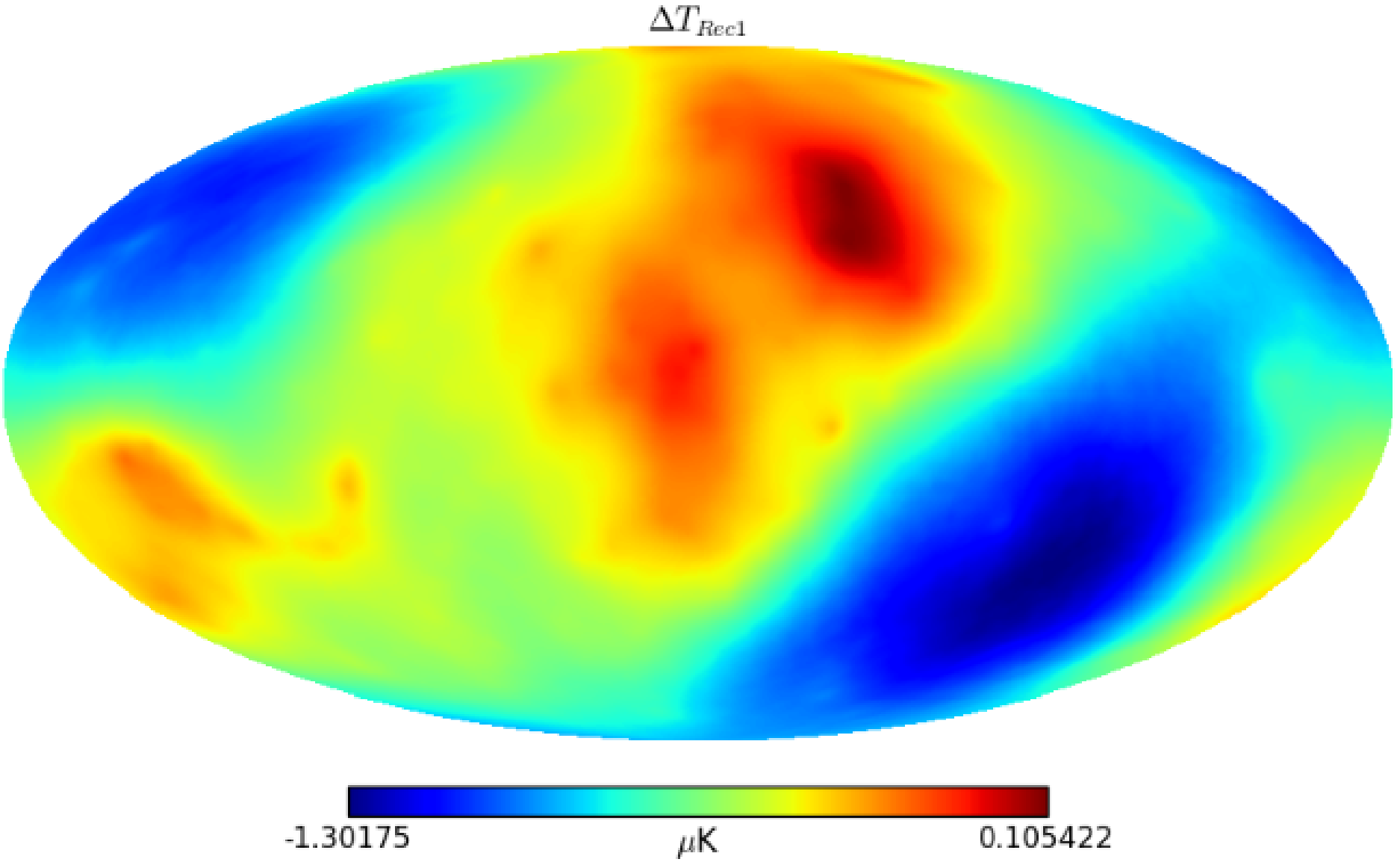
Sheets





All

$\lambda_{th}=0.2$


ΔT_{Rec1}
-1.30175
μK
0.105422

How do we deal with the data?

Principal component analysis

The goal: Identify relations among data variables

- *Not to find the relation*
- *Not to predict values of f(x) given x*
- *Very useful when dealing with multidimensional data*

The idea: Find a representation of the data vectors in such a way that it can be easily found the most relevant data relationships

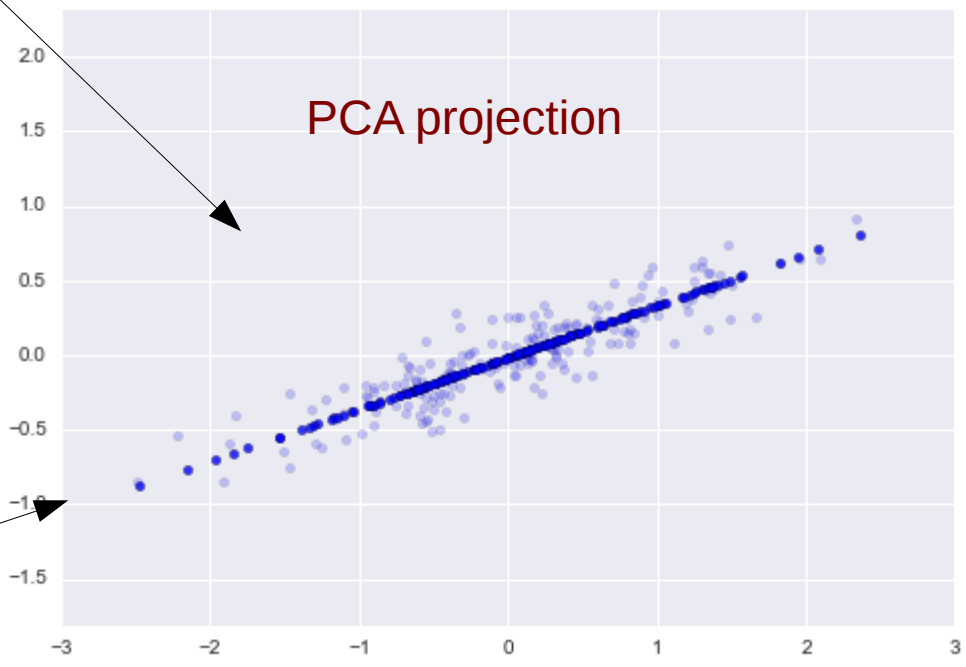
Examples are based on the text: Python data science handbook.

2D case: The simplest!

Data representation: The largest the spread along one axis, the larger the importance of the axis

Note that the scatter does not add any more information to the relation between x and y
(this comes from the correlation)

You can use the eigenvectors of the “important” directions to build new basis and fin a new representation
Project the data along the important direction



Are we missing information on the correlation between x an y?

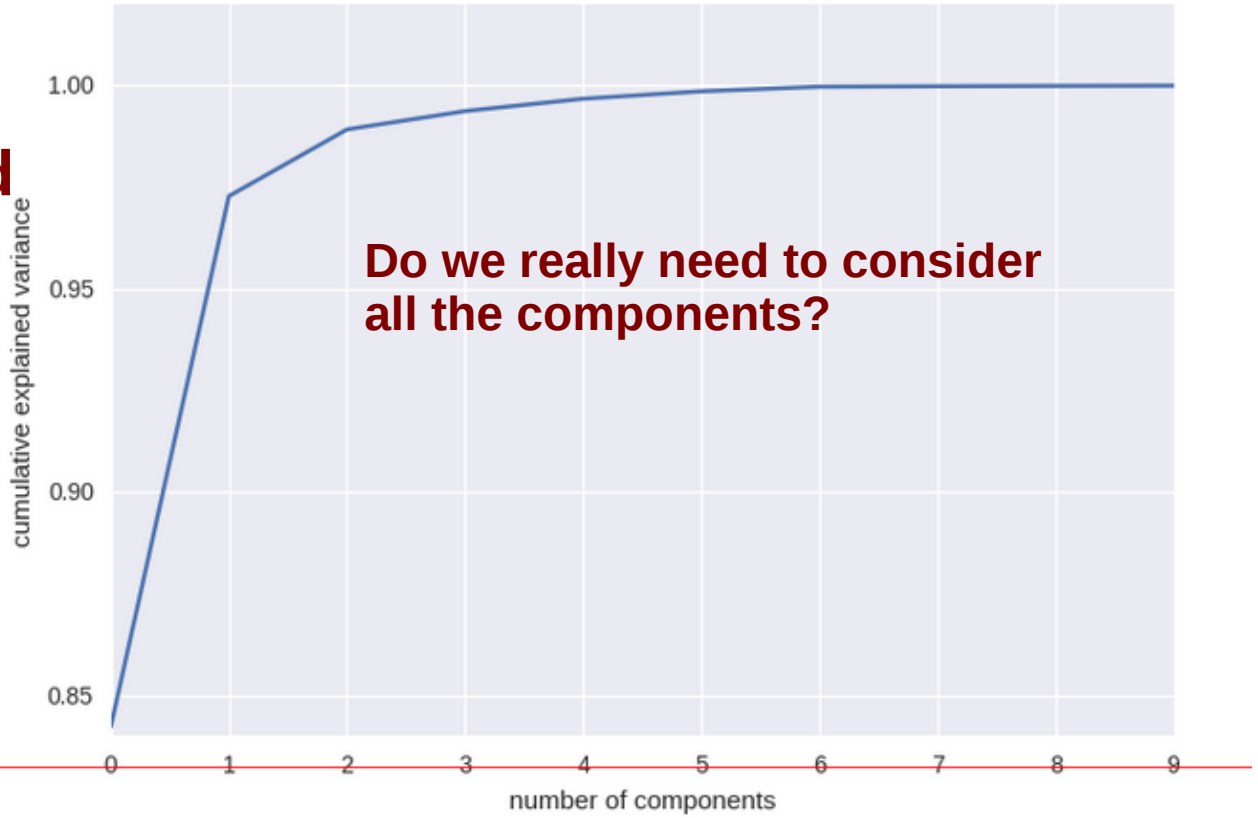
- Initially there are as many components as features in the data

For a set with 10 features (u, Δu, g, Δg, …) there are 10 components. You choose the amount of components you want to preserve in the new representation

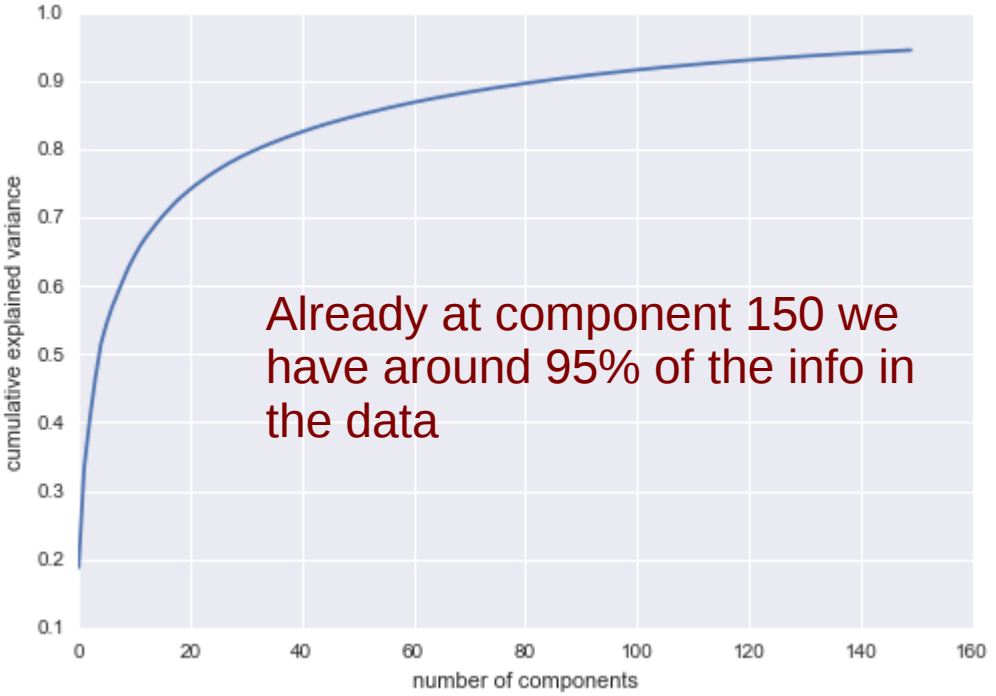
- Lowering the dimensionality of the problem focus your attention on real correlation

The **cumulative explained variance** quantifies the cumulative contribution of each component to the variance in data

Variance = importance



Some of the principal components of the initial 3000 features

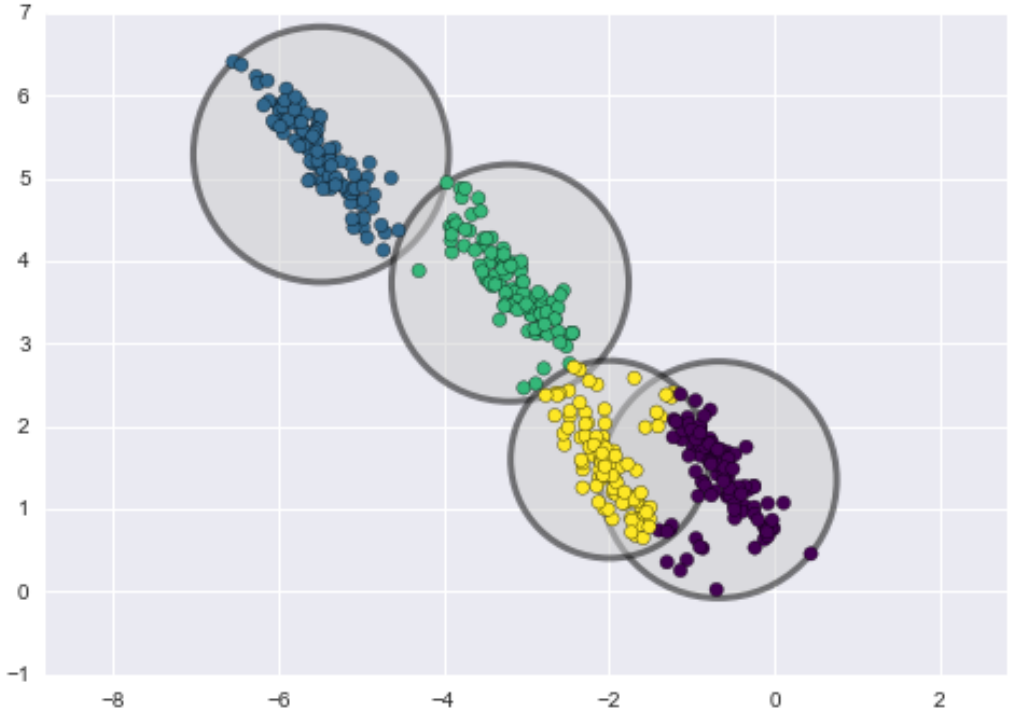


Already at component 150 we have around 95% of the info in the data

Reconstruction using 150 principal components

- Examples 0, 1, 2

Gaussian Mixture Models



Assume you have *N* data points in *M* dimensional space.

The goal: You want to know the properties of *K* multivariate Gaussians to describe the data population

Why is this an example of unsupervised ML?
There is no a priori control on which data point is going to fall in given Gaussian
After GMM you will get the probability of a point to belong to a given gaussian model

After GMM you will get the probability of a point to belong to a given gaussian model (responsibility matrix)

$$p_{nk} \equiv P(k|n) = \frac{N(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) P(k)}{P(\mathbf{x}_n)}$$

$0 < k < K$
$0 < n < N$

N x *K* matrix holding the probability that data point *n* belongs to model *k*

Here the problem is that knowing the data points and the number of gaussian models, we need to know the the properties of the gaussians

We are then going to use maximum likelihood to get the values of the parameters of the gaussians that best fit the data

$$\mathcal{L} = \prod_n P(\mathbf{x}_n) \qquad P(\mathbf{x}_n) = \sum_k N(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) P(k)$$

$$N(\mathbf{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{M/2} \det(\boldsymbol{\Sigma})^{1/2}} \exp[-\tfrac{1}{2}(\mathbf{x} - \boldsymbol{\mu}) \cdot \boldsymbol{\Sigma}^{-1} \cdot (\mathbf{x} - \boldsymbol{\mu})]$$

An that is all!

However, we need values of the mean and the covariance to compute the likelihood!

How do we do?

Iterate!

$$\hat{\boldsymbol{\mu}}_k = \sum_n p_{nk}\mathbf{x}_n \Big/ \sum_n p_{nk}$$

$$\hat{\boldsymbol{\Sigma}}_k = \sum_n p_{nk}(\mathbf{x}_n - \hat{\boldsymbol{\mu}}_k) \otimes (\mathbf{x}_n - \hat{\boldsymbol{\mu}}_k) \Big/ \sum_n p_{nk}$$

$$\hat{P}(k) = \frac{1}{N}\sum_n p_{nk}$$

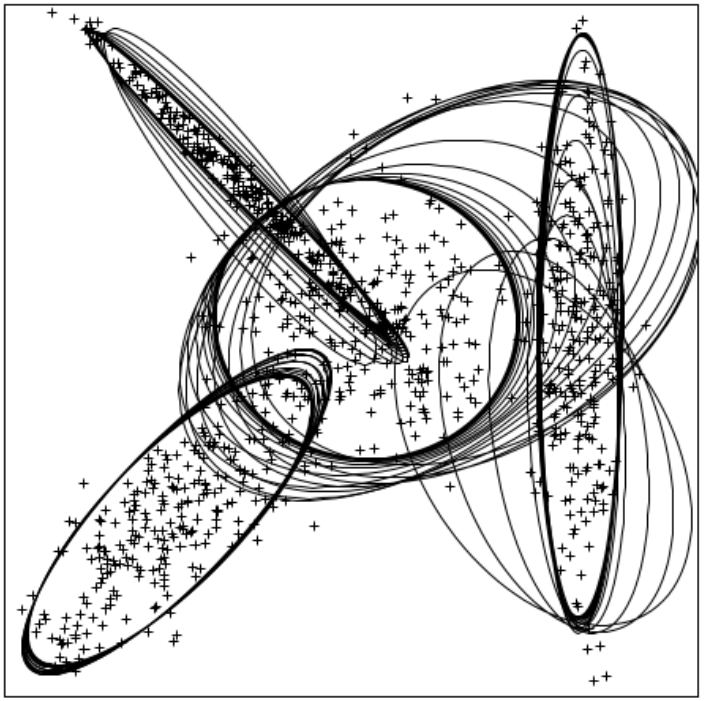
Maximum likelihood estimators of the parameters.

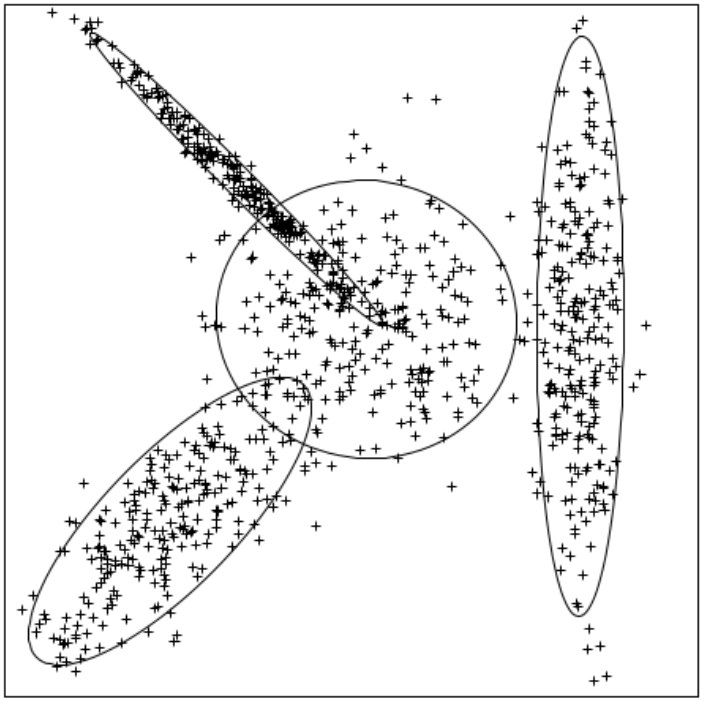
1) *Estimate (from an initial guess) the params.*

2) *go back to previous slide, compute expectations.*

3) *Come back, maximize, go back…*

Stop when Likelihood does not change any more.





Examples 3 and 4?

Kernel density estimation with GMM

Sometimes all we want is to be able to characterize the properties of the sample distribution. We want to be able to sample the “density distribution” from where the data was drawn during observations

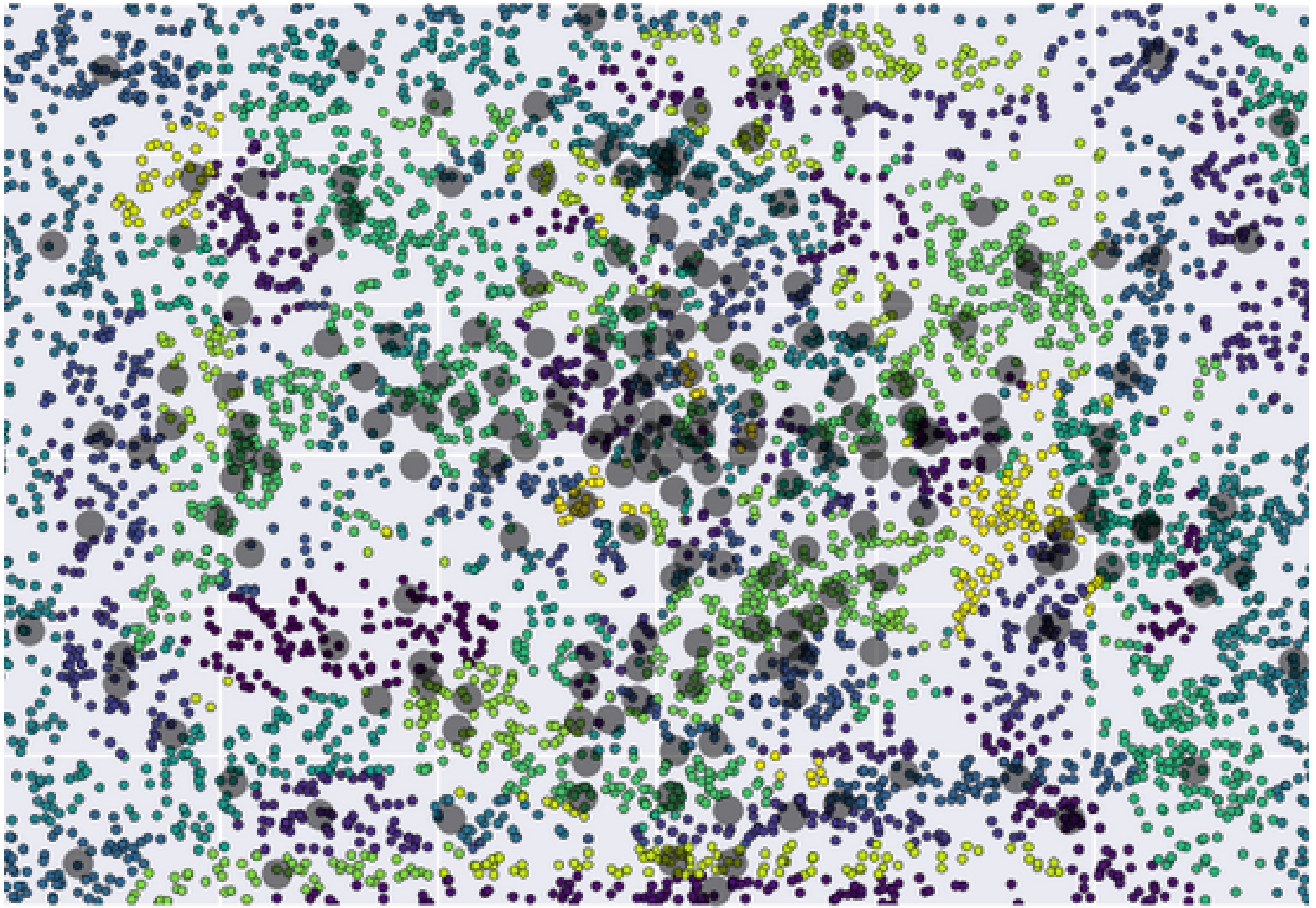
If we have a finite set of sampling points, a kernel that softens (weights) the reconstruction may be useful!

$$\widehat{f}_N(x) = \frac{1}{Nh^D} \sum_{i=1}^{N} K\left(\frac{d(x, x_i)}{h}\right),$$

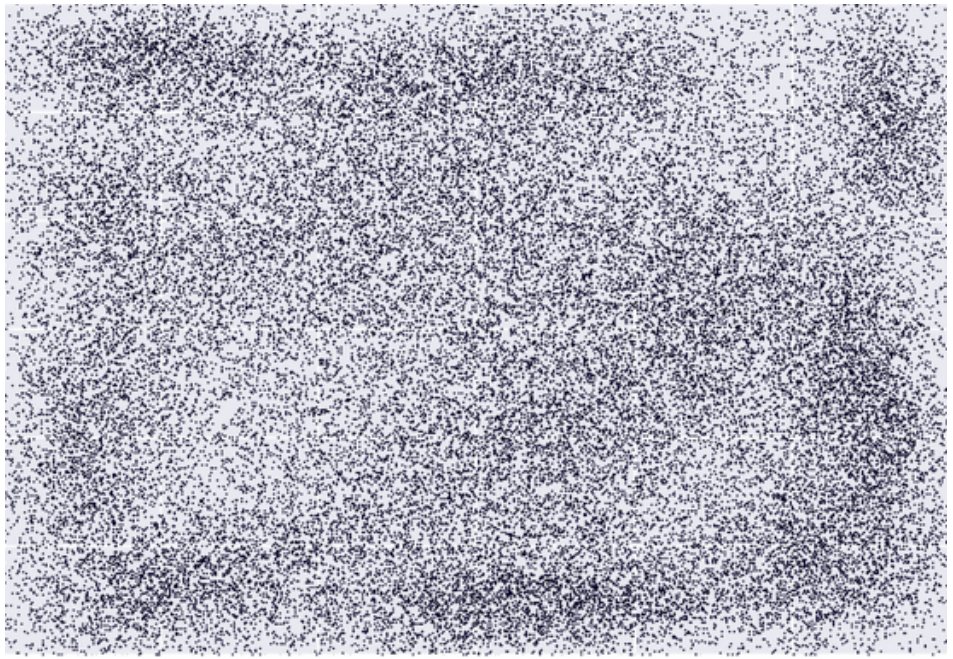
Probability distribution for the data set!

Kernel density that softens the field

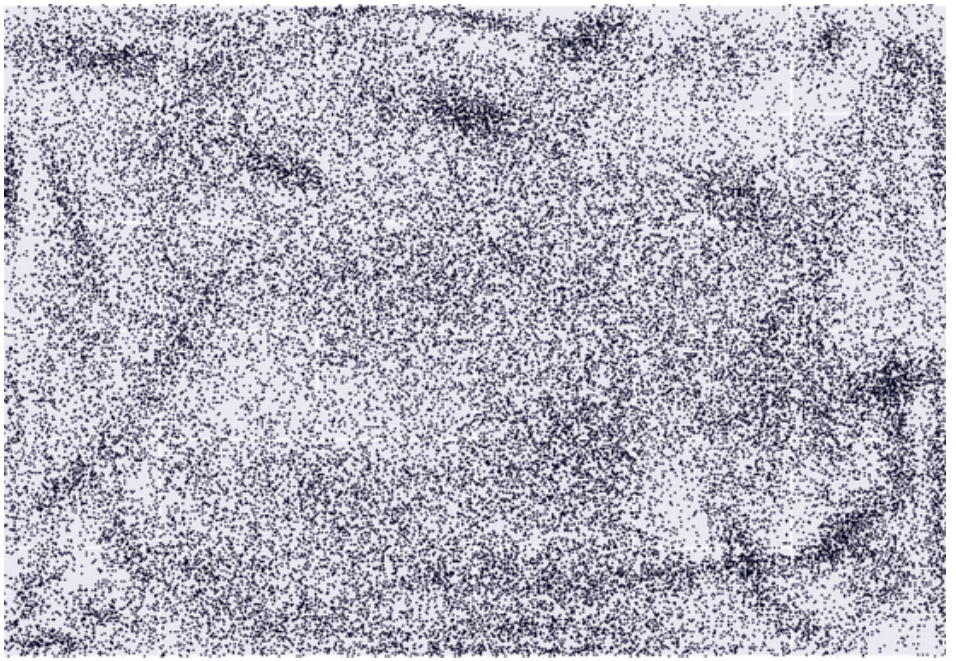
A multivariate gaussian function is a good kernel for multidimensional density estimation

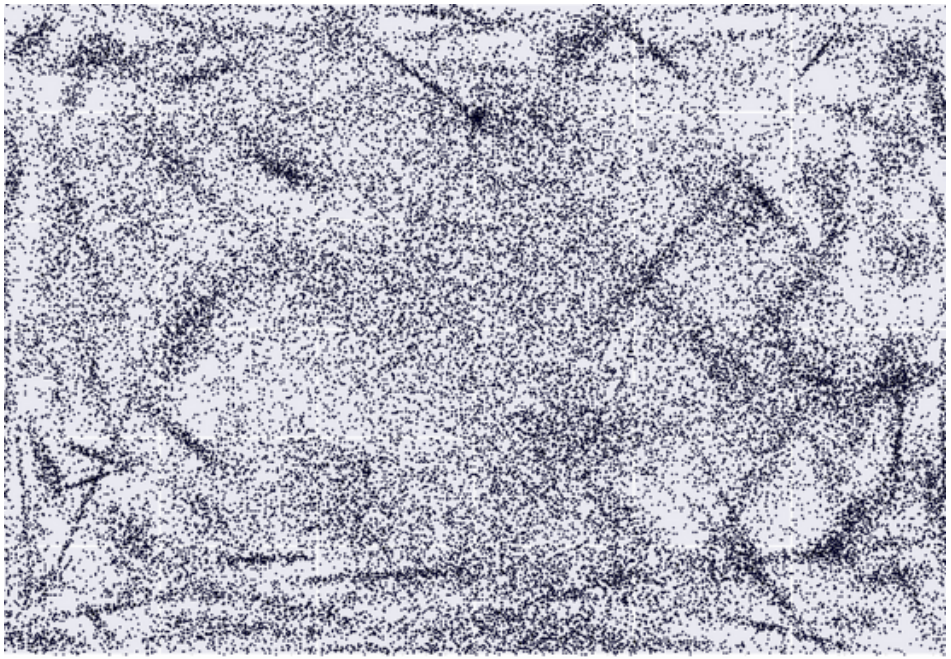


20 Gaussians



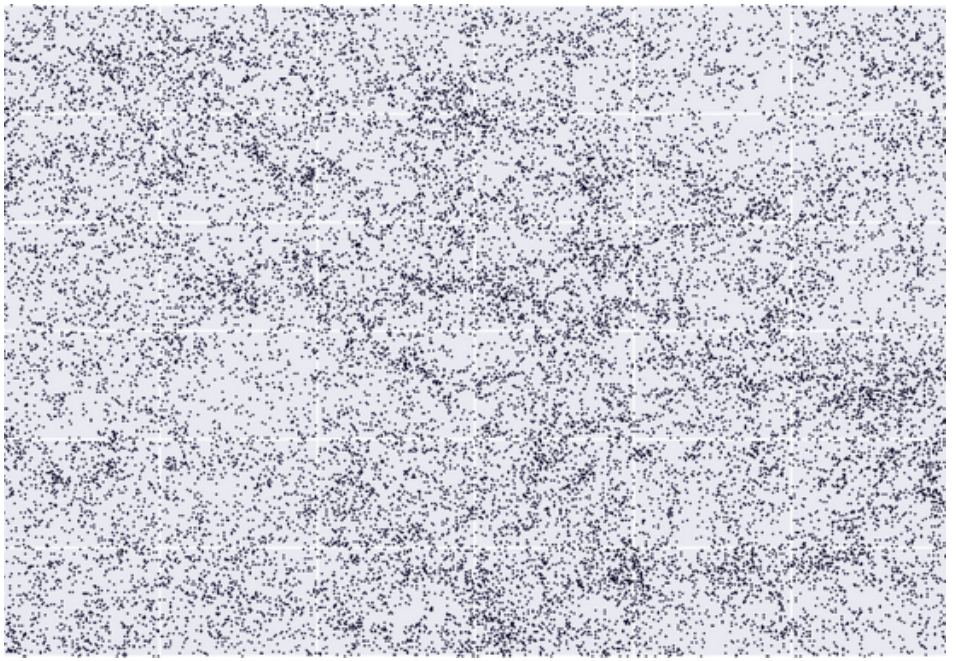
500 Gaussians





1000 Gaussians

Original Data



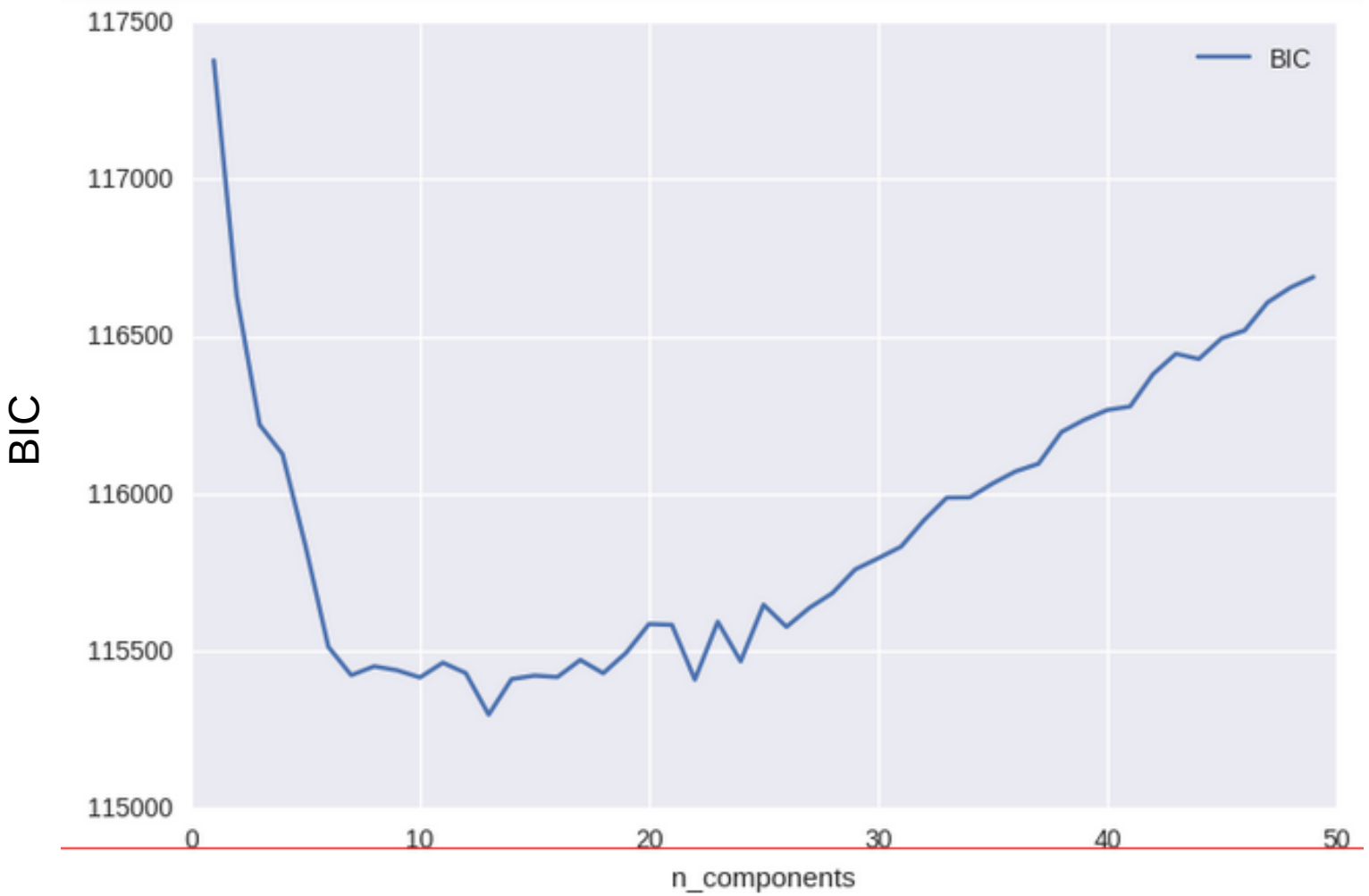
Bayesian Information Criterion

$$\mathrm{BIC} = \ln(n)k - 2\ln(\hat{L})$$

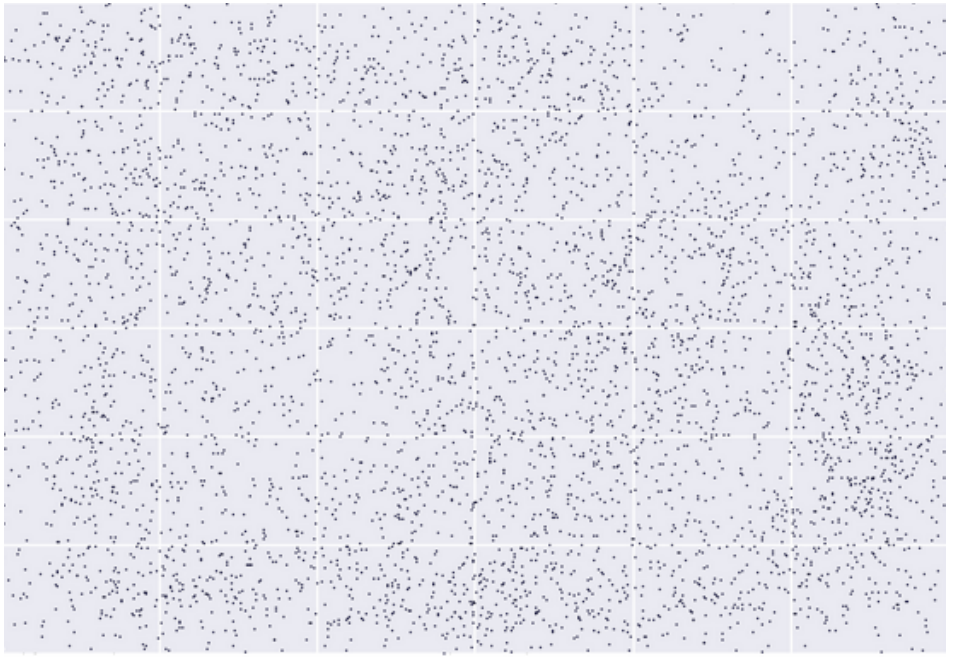
Likelihood

# of data points

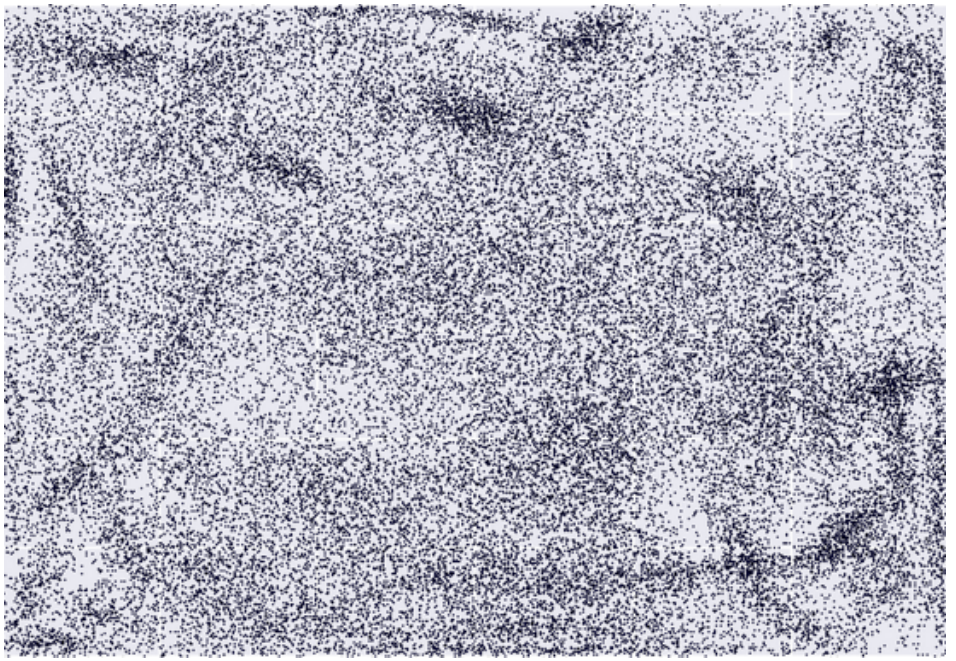
# of params

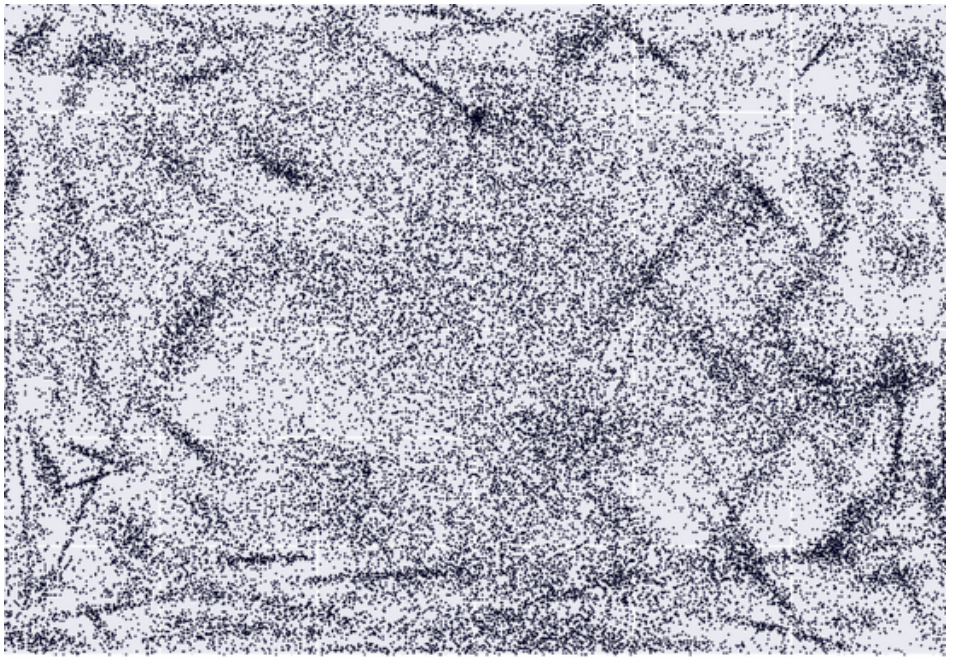


20 Gaussians



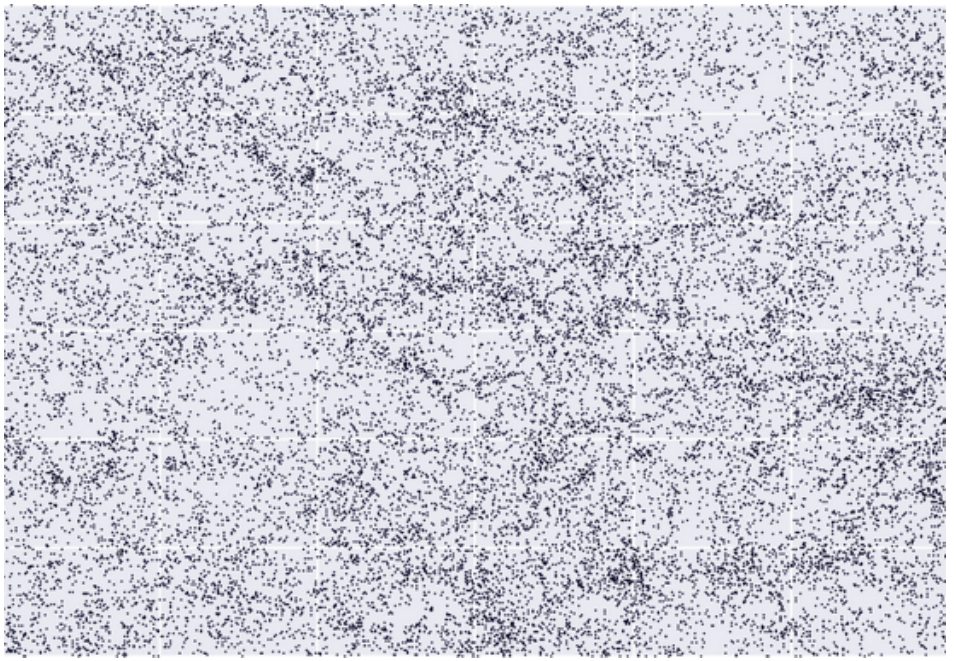
500 Gaussians





1000 Gaussians

Original Data



K-means clustering

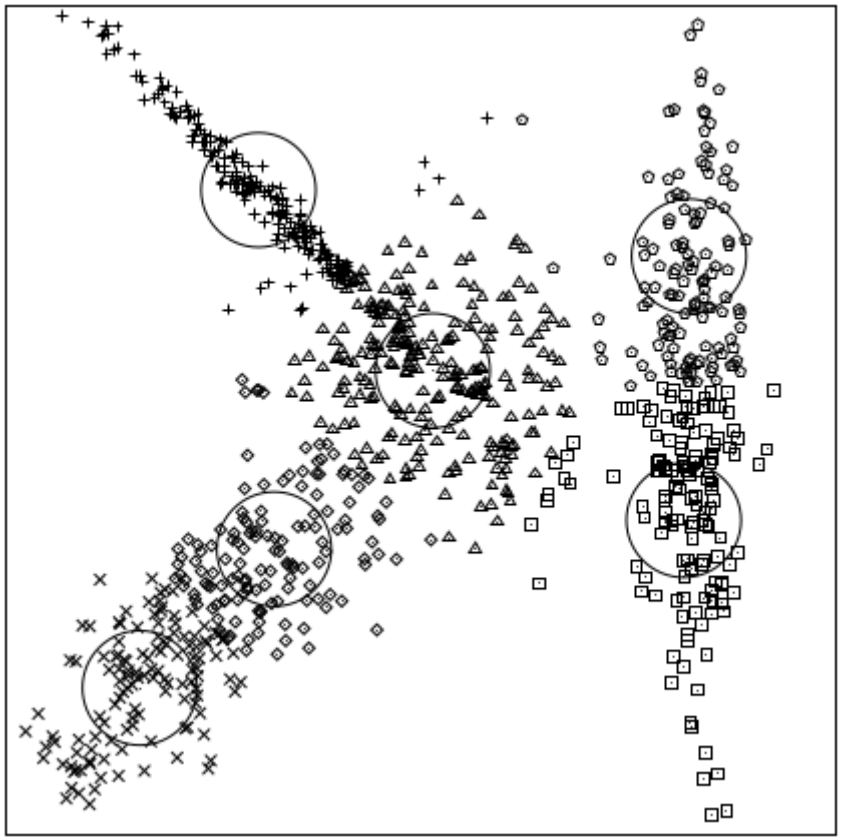
Lets assume a simplistic situation where *we want to find, for a given number of group centers, the set of data points that are closest to a given center than to any other!*

1) *Estimate (from an initial guess) the mean position of the center.*

2) *Estimate new memberships.*

3) *Iterate...*

Stop when cluster sets do not change any more.



- Kmeans is actually limit case of GMM, with diagonal and almost Identity covariance.

- Each data point will belong to one and only one cluster

What have we done (learn?) in this lectures?

- How to deal with the use of the architecture of computer
- Do not write like crazy in HD… use it wisely
- Optimize when developing your code...
- Do not forget you are telling the computer what to do, and how… do it right:
  - *Example of people dividing 2/3 and getting nonsense...*
  - *Example of people doing funny stuff with *,/ with no care about the use of (...) and getting wrong results*
  - *People running stuff veeeeeeeery slowly...*

- We have seen the basic ideas of parallel computing
- **I hope** we understood the basic ideas of parallelization with MPI
- Collective communication, point-to-point communication
- We saw the basic functions of SQL queries, with examples in SDSS
- Finally we saw how can we use some tools from scikit to get properties from data using tools of ML

Some useful references

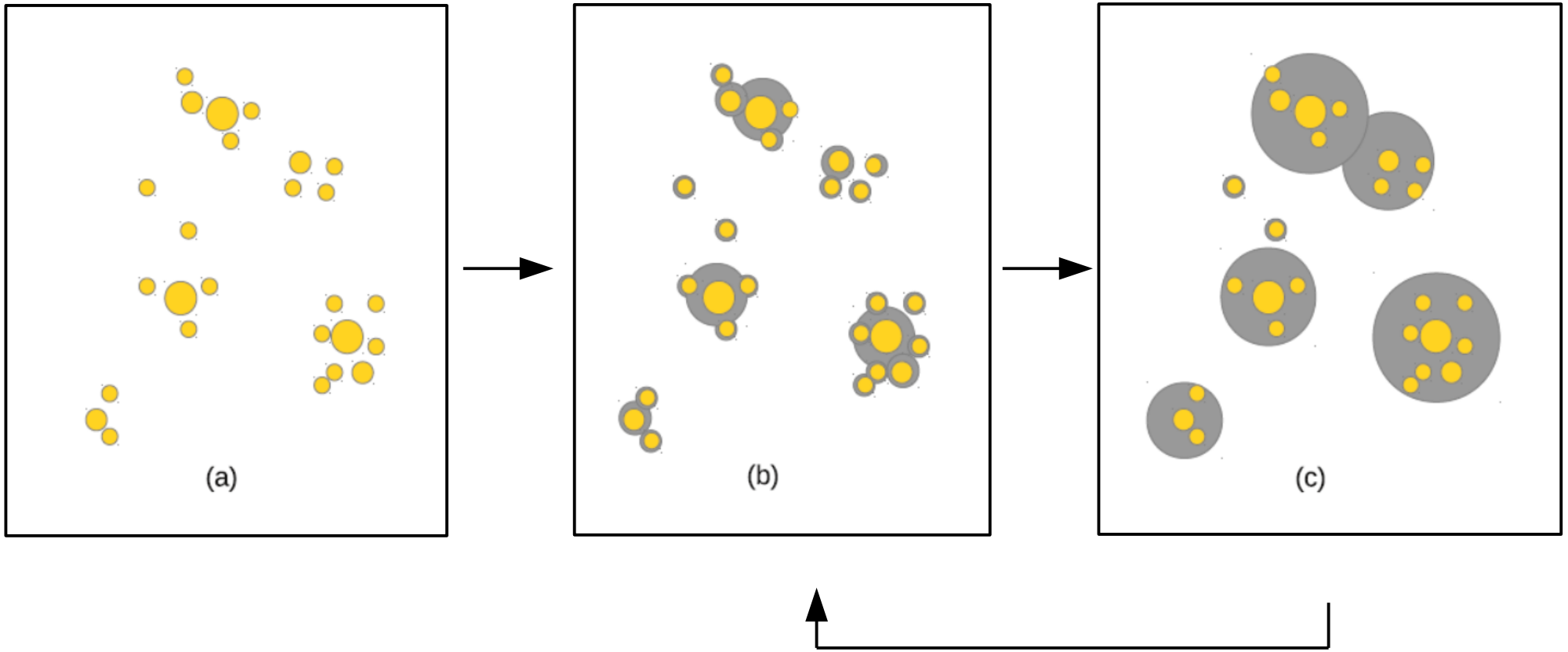
- Introduction to High Performance Computing for Scientists and Engineers, Hagger & Wellein, 2011
- Software Design for Engineers and Scientists, Allen Robinson, 2004
- Introduction to High Performance Scientific Computing, Eijkhout, 2014
- Using MPI: Portable Parallel Programming with the Message-Passing Interface, Gropp , Lusk,Skjellum, 2014
- Python Data Science Handbook, VanderPlas, 2017
- Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Worked Examples, and Case Studies, Kelleher, Namee, D'Arcy, 2015
- Statistics, Data Mining, and Machine Learning in Astronomy: A Practical Python Guide for the Analysis of Survey Data, Ivezić, Connolly, VanderPlas, Gray, 2014

- Exer 1:
- Reproduce the example we did in class with GMM, but now look for a way to make it work with Bayesian GMM.
- What is the difference between BGMM and GMM?

- Exer 2:
- Go to SDSS, download data of objects classified as galaxies.
- You will download g and r data to build a CMD digram (using not corrected absolute magnitudes provided by the data base). (zwarnong=0, z>0.03)
- Use the GMM technique to separate the data you get in two components.

- Exer 3:
- Take the data of halos we used for the kmeans example, use a GMM.
- Can you get a better clustering pattern?

The Method (Summarized)



Initialize galaxy properties:
x, y, z (Redshift space)
L, etc.

Compute group properties:
Stellar mass
Group luminosity
Halo mass
Virial radius R_{vir}
Ellipsoidal R_{zs}

Merge:
Merge “overlapping” halos
Intersecting the ellipsoid
$f(R_{vir},R_{zs})$
Inherit galaxies
Recompute center