Class 4: Regression

In this class we will explore how to model an outcome variable in terms of input variable(s) using linear regression, principal component analysis and Gaussian processes

Class 4: Regression

At the end of this class you should be able to …

- … generate a least-squares regression line to a dataset
- … handle cases with errors in both co-ordinates
- … perform a principal component analysis on a set of variables
- … construct Gaussian Process models for interpolation

Regression

- **Regression** describes any statistical method which determines a relationship between a **dependent** (outcome) variable y and **independent** (predictor) variable(s) x

Image credit:
learningstatisticswithr.com

Regression

- In **linear regression** we suppose the relationship is a straight line; a standard method of determining that line is to **minimize the residuals** between it and the points:

Image credit:
learningstatisticswithr.com

Least-squares linear regression

- Specifically, the **least-squares linear regression line** is the linear fit to a dataset (x_i, y_i) that minimizes the sum of the squares of the y-residuals
- **With an intercept**, i.e. fitting the line $y = a\,x + b$:

$$a = \frac{\sum_{i=1}^{N} x_i y_i - N\bar{x}\bar{y}}{(N-1)\,{\sigma_x}^2} = r\frac{\sigma_y}{\sigma_x}$$

$$b = \mu_y - a\,\mu_x$$

- **Without an intercept**, i.e. fitting the line $y = a\,x$:

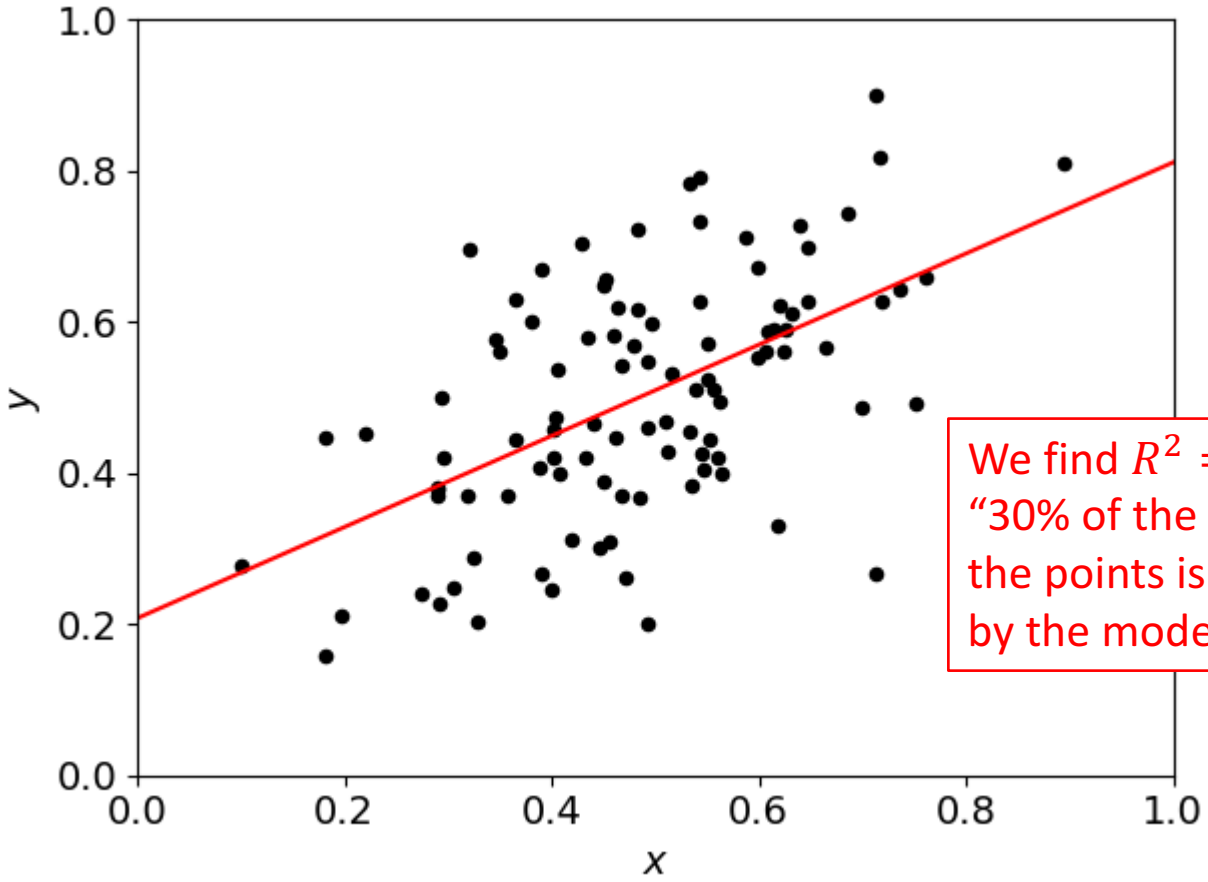
$$a = \frac{\sum_{i=1}^{N} x_i y_i}{\sum_{i=1}^{N} {x_i}^2}$$

Quantifying the regression fit

- As well as the best-fitting line, we also need to quantify the **accuracy of the model**
- Let's consider the **sum of the squared residuals** from the model, $SS_{\mathrm{res}} = \sum_i \left(y_i - y_{\mathrm{mod},i}\right)^2$
- We also consider the **total sum of squares** $SS_{\mathrm{tot}} = \sum_i (y_i - \bar{y})^2$ which is proportional to the **variance** of y_i
- We define the **coefficient of determination** $R^2 = 1 - \frac{SS_{\mathrm{res}}}{SS_{\mathrm{tot}}}$, which is the *"fraction of variance explained by the fit"*
- It's easy to use these formulae to show that *R is exactly the same as the correlation coefficient r* we met in Class 2

Least-squares linear regression

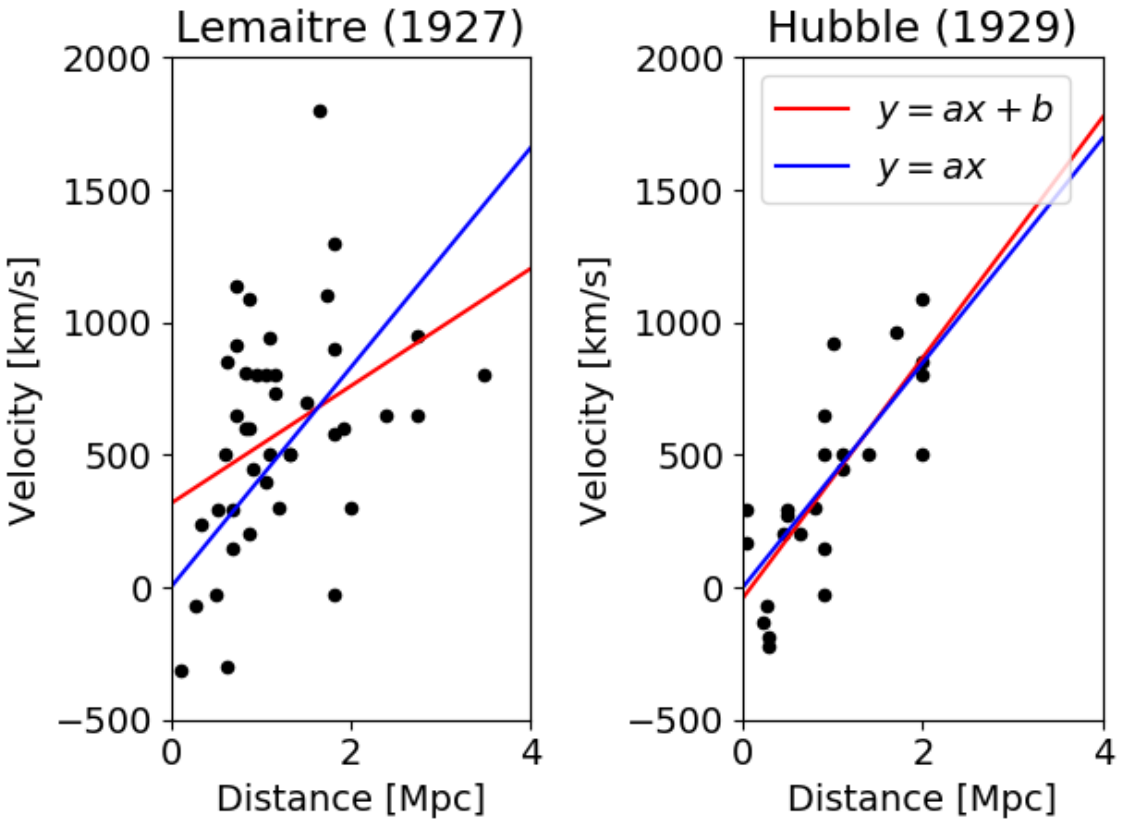
- *Determine the linear regression line for the test correlation dataset from Class 2:*



We find $R^2 = 0.30$, i.e. "30% of the variance of the points is explained by the model fit"

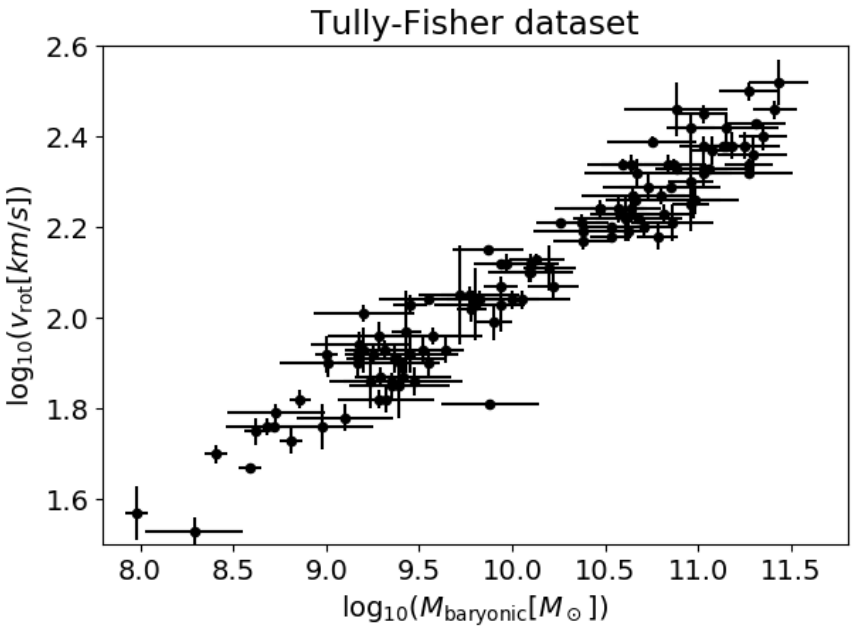
The Hubble parameter (continued)

- Returning to Hubble and Lemaitre's distance-velocity datasets, find the linear least-squares regression lines with and without an intercept, and the value of R^2



Weighted regression

- We can vary the **weights** w_i of each point when minimizing the model deviations (if for example, their errors σ_i vary)
- Note that linear regression with weights $w_i = 1/\sigma_i^2$ is equivalent to minimizing the χ^2 statistic in a model fit
- A more general case is with **errors in both co-ordinates**:



The case of errors in both co-ordinates

- One solution for cases with errors in both co-ordinates is to modify the function we are minimizing:

$$\chi^2(a,b) = \sum_{i=1}^{N} \frac{(y_i - ax_i - b)^2}{\sigma_{y,i}^2 + a^2\,\sigma_{x,i}^2}$$

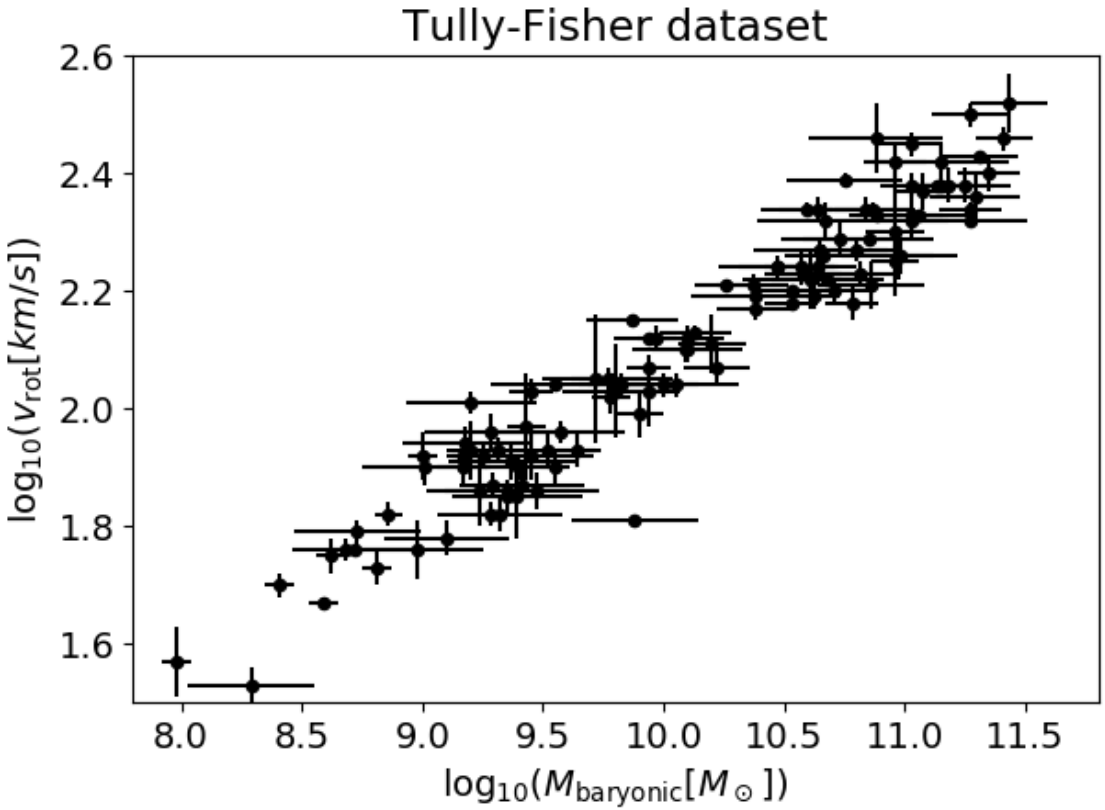
- The denominator propagates the variance in y from the data ($= \sigma_{y,i}^2$) and from the evaluation of the model at $y = ax_i + b$ ($= a^2\,\sigma_{x,i}^2$)

- [Small print: this expression is not symmetric in x and y]

The Tully-Fisher relation

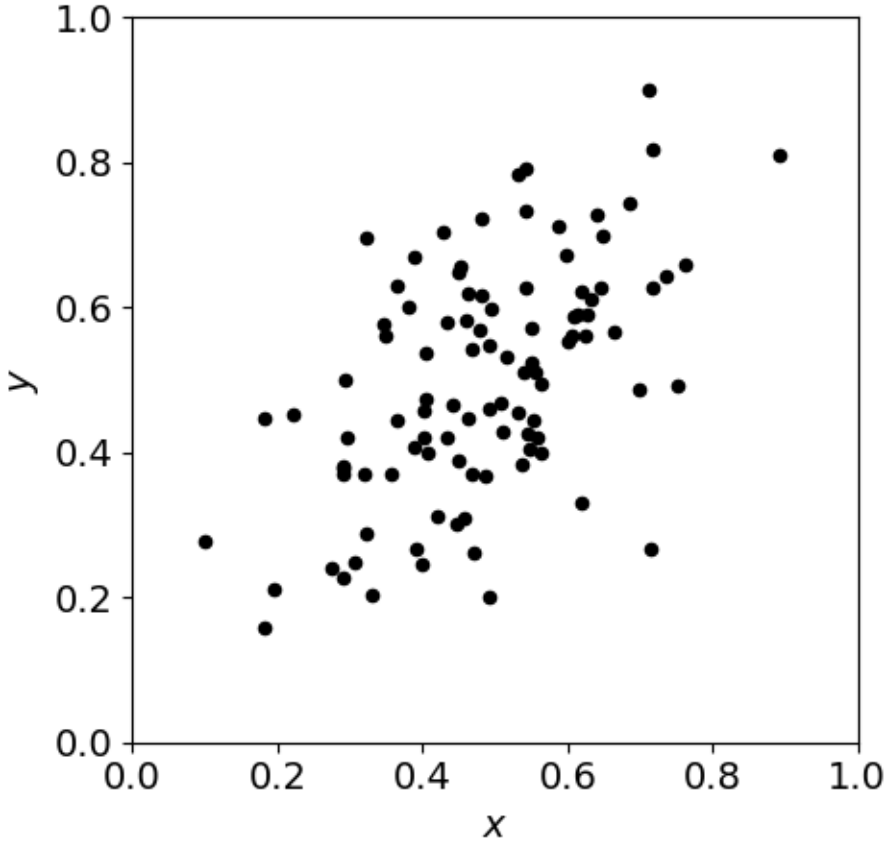
- For example, consider an example dataset containing the stellar masses and rotation velocities of galaxies:

- Find the best-fitting linear regression by minimizing the function on the previous slide using the errors in both co-ordinates



Principal component analysis

- Let's say we have a dataset which contains many variables for each object (e.g., magnitudes, sizes, types of galaxies)



(We'll just use 2 variables x and y to keep the illustration simple, but you can imagine that the "cloud" of points could extend into more variables)

Principal component analysis

- Let's say we have a dataset which contains many variables for each object (e.g., magnitudes, sizes, types of galaxies)
- **Principal component analysis** (PCA) is a procedure which uses the correlations between the variables to identify *which combinations of variables capture most information about the dataset*
- **Geometrically**, it identifies the directions in which the cloud of variables is most elongated
- **Mathematically**, it determines the ***eigenvectors*** of the covariance matrix and sorts them in importance according to their corresponding ***eigenvalues***

Principal component analysis

- Applying the mathematical steps to our (x, y) example:

- Find the **covariance matrix** of (x, y): $C = \begin{pmatrix} 0.021 & 0.013 \\ 0.013 & 0.026 \end{pmatrix}$

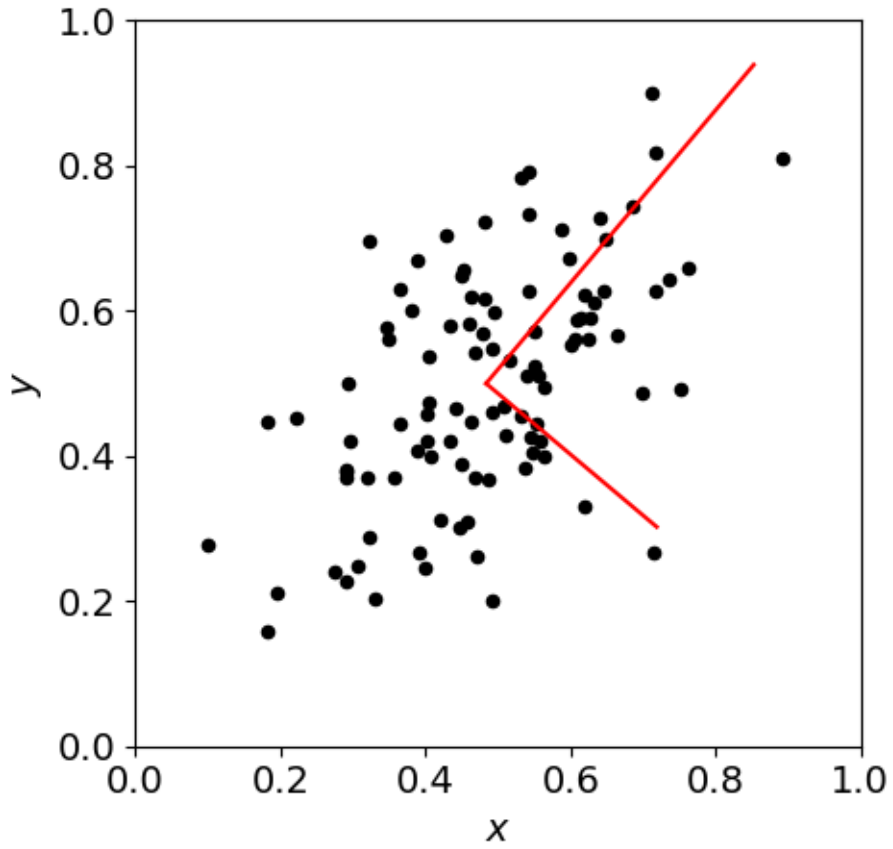
$$C = \begin{pmatrix} \mathrm{Cov}(x,x) & \mathrm{Cov}(x,y) \\ \mathrm{Cov}(y,x) & \mathrm{Cov}(y,y) \end{pmatrix} \qquad \mathrm{Cov}(x,y) = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})\,(y_i - \bar{y})$$

- Determine the **eigenvalues** and **eigenvectors** of C:
  eigenvalues are $\lambda_1 = 0.037$, $\lambda_2 = 0.011$ *with corresponding eigenvectors* $\vec{v}_1 = (0.64, 0.77)$ *and* $\vec{v}_2 = (0.77, -0.64)$

- Express the data points in the **basis of the eigenvectors** – new co-ordinates are (PC_1, PC_2) such that $\vec{x} = (x, y) = (\bar{x}, \bar{y}) + PC_1\,\vec{v}_1 + PC_2\,\vec{v}_2$

Principal component analysis

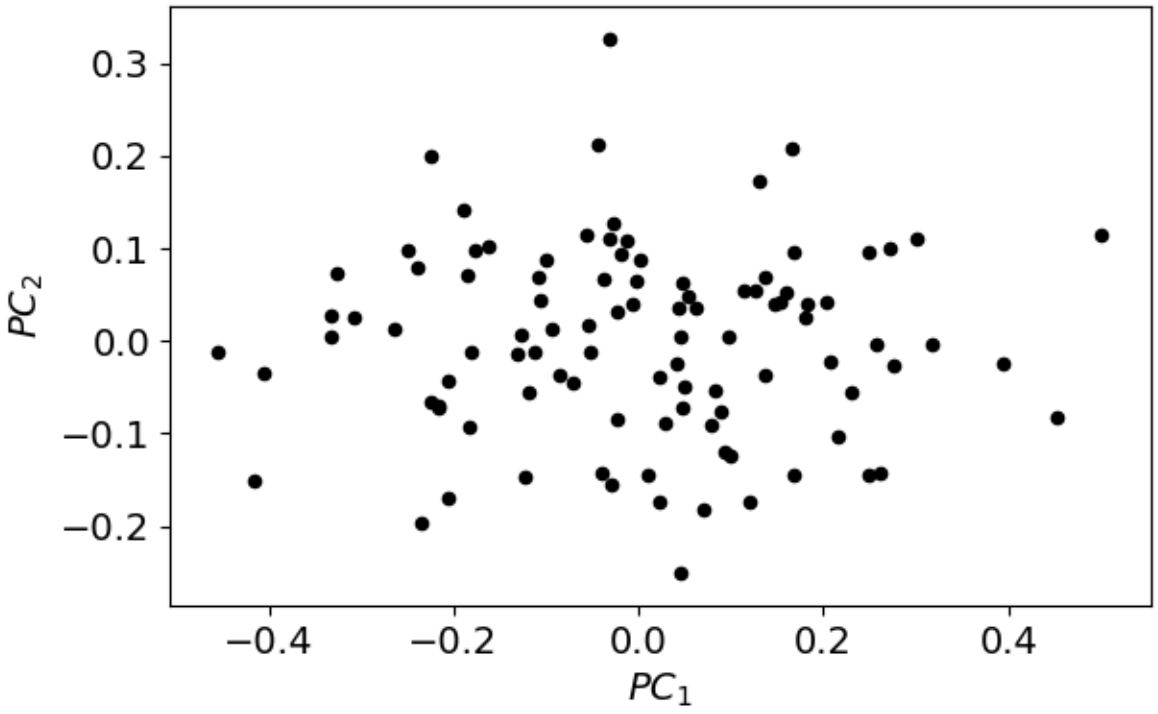
- *Here are the eigenvectors overplotted on the data, with lengths proportional to the square root of the eigenvalues:*



- The eigenvectors define the directions of the “principal axes” of the cloud of points
- The size of the eigenvalues corresponds to the variance (spread) of data along each principal axis

Principal component analysis

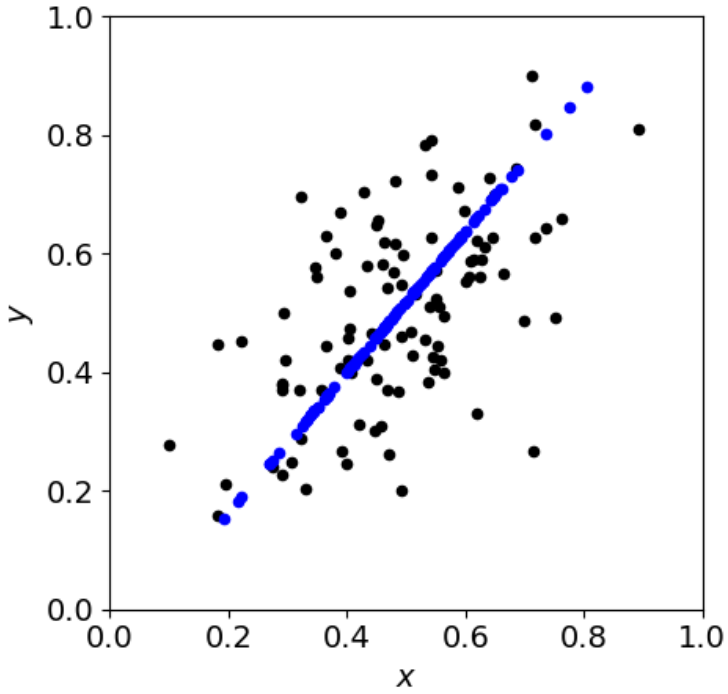
- *Here are the principal component values of each data point:*



- The cloud of points has been rotated such that its principal axes line up with the co-ordinate system
- PCA is analogous to a rotation: $C = \Lambda D \Lambda^T$, where D is a diagonal matrix and Λ is a matrix whose columns are the eigenvectors

Principal component analysis

- PCA is commonly used for **dimensionality reduction**, i.e. *approximating a dataset with a fewer number of variables*
- We can illustrate this by reconstructing our previous dataset using only 1 principal component, $(x, y) = (\bar{x}, \bar{y}) + PC_1 \, \vec{v}_1$



- The blue points are an approximation of the original black points
- The amount of variance retained is determined by the size of λ_1 compared to $\sum_i \lambda_i$

Principal component analysis

- Perform a Principal Component Analysis on the provided dataset of SDSS quasar magnitudes. How many principal components are needed to explain 90% of the variance?

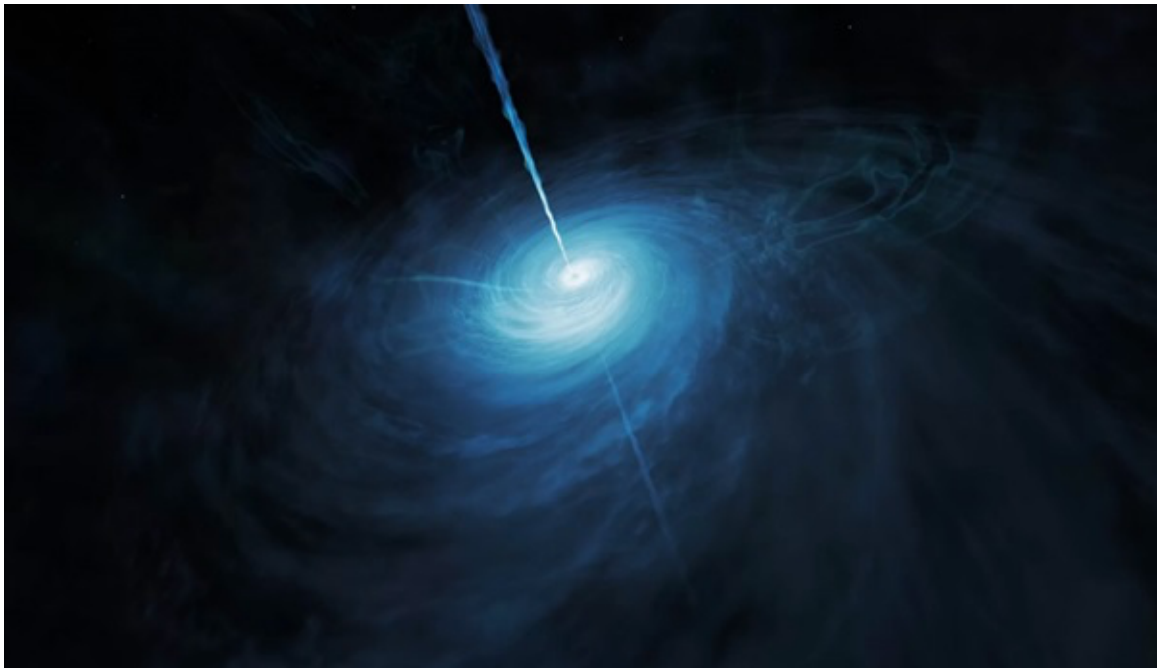
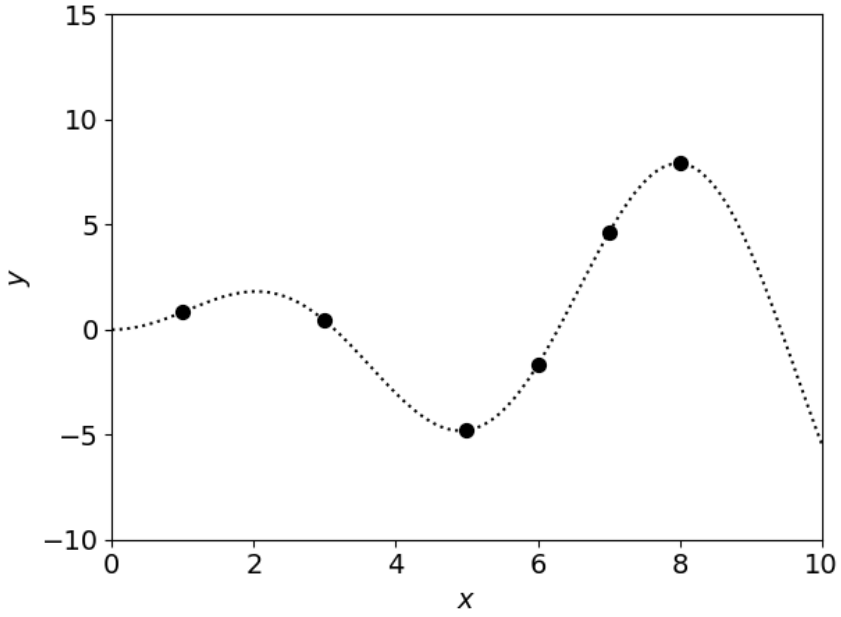


Image credit: astronomy.com

Interpolation

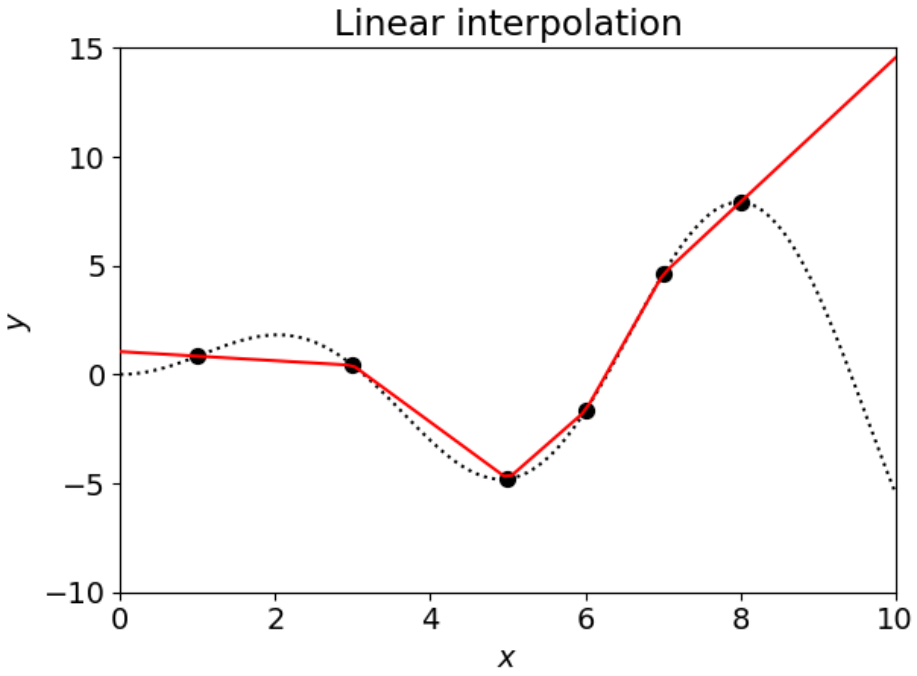
- We may wish to use our model to predict outcome values in between the positions of our data points ("**interpolation**")
- There are various possible approaches to this, depending on what assumptions we want to make about the properties of the interpolating function

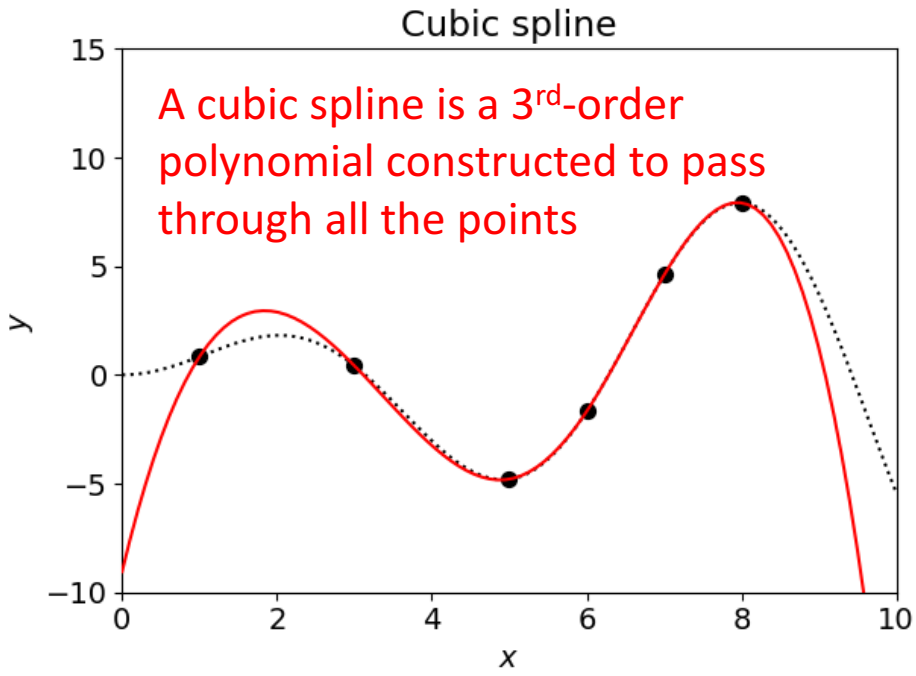


Let's consider the example of the function $y = x \sin x$ sampled at $x = (1,3,5,6,7,8)$ [credit: scikit-learn documentation]

Interpolation

- Two general approaches are to use **linear interpolation** or a **cubic spline**:

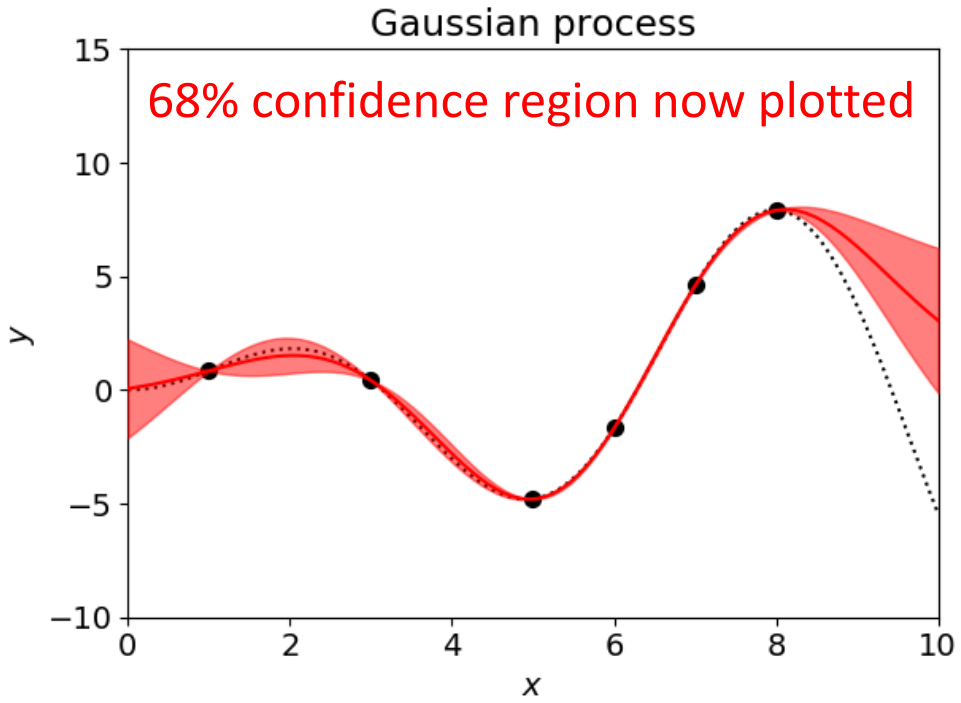




- *These approaches don't provide an error in the interpolation*

Interpolation

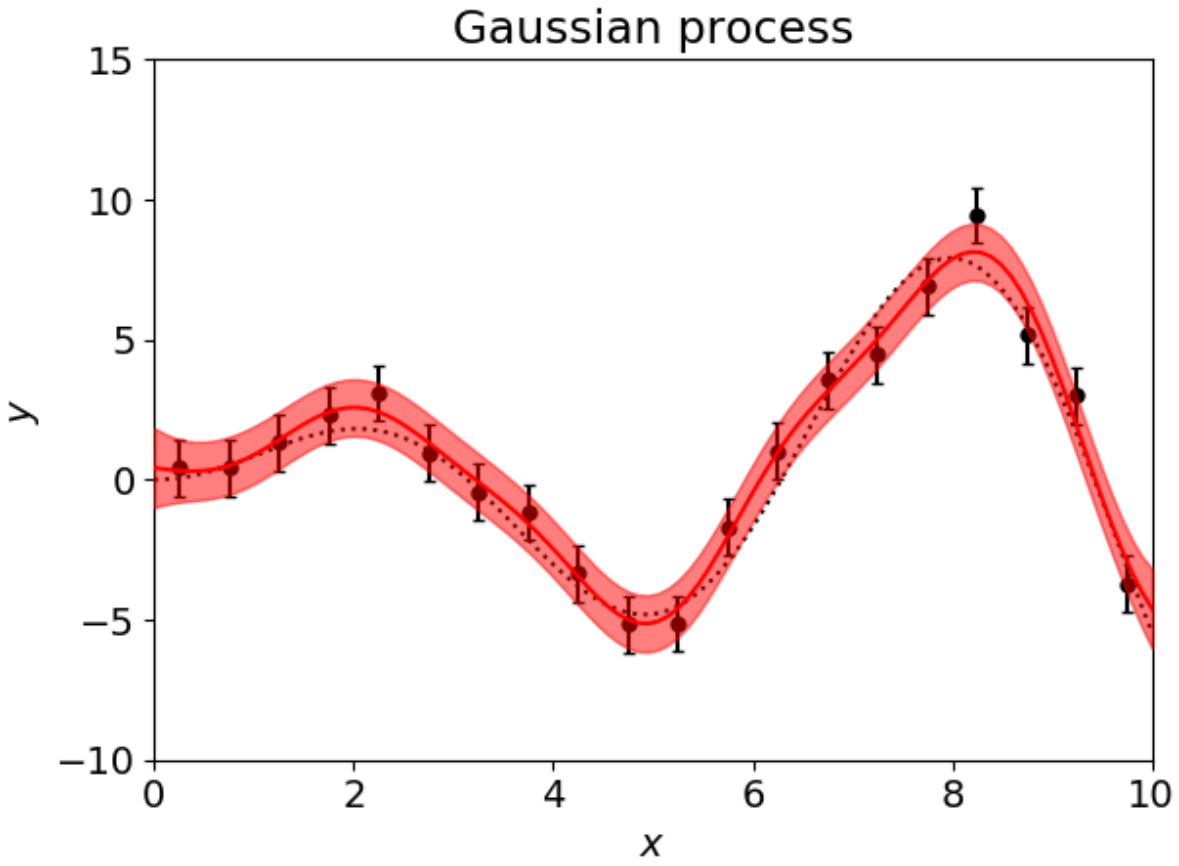
- Another approach is to model the function using a **Gaussian process** (which is also known as **kriging** in some fields)
- *In so doing, we're imposing a statistical model for the correlations in the function (a "smoothness prior")*



- The Gaussian Process requires us to specify a "kernel" which describes the degree of correlation which is allowed in the function
- Here we have assumed $K(x_1, x_2) = e^{-\frac{1}{2}\left(\frac{x_1 - x_2}{L}\right)^2}$ where $L = 1$; this is a length scale of allowed variation

Interpolation

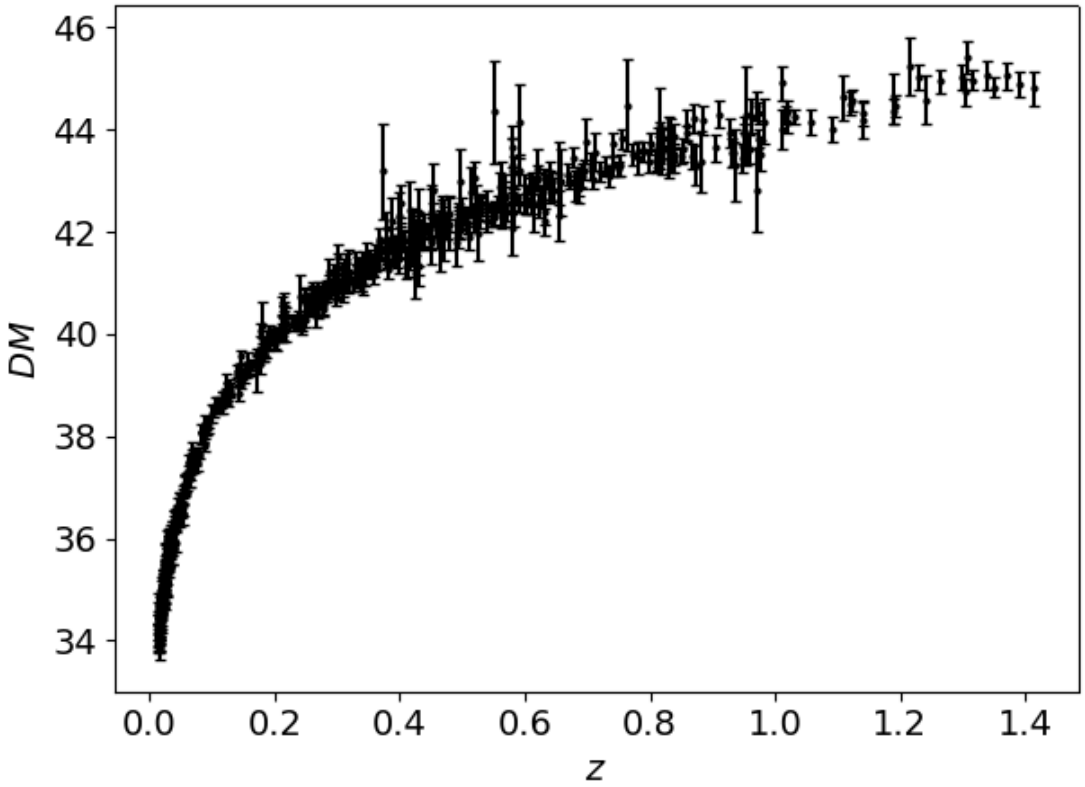
- A Gaussian process can also propagate **noise** in the data into the **error in the prediction**:



- Here we changed the kernel to include a term modelling the noise

Supernova cosmology (continued)

- Let's return to the **supernova distance-redshift dataset** from Class 3. *Fit a Gaussian process model to this dataset to predict the distance modulus and its error at any redshift.*



Summary

At the end of this class you should be able to …

- … generate a least-squares regression line to a dataset
- … handle cases with errors in both co-ordinates
- … perform a principal component analysis on a set of variables
- … construct Gaussian Process models for interpolation