

Programming Data Visualization & Numerical Data Analysis

Luna Pianesi

Faculty of Technology, Bielefeld University

```
332
333
334     if extrapolate is None:
335         extrapolate = self.extrapolate
336     x = np.asarray(x)
337     x_shape, x_ndim = x.shape, x.ndim
338     x = np.ascontiguousarray(x.ravel(), dtype=np
339
340     # With periodic extrapolation we map x to the
341     # [self.t[k], self.t[n]].
342     if extrapolate == 'periodic':
343         n = self.t.size - self.k - 1
344         x = self.t[self.k] + (x - self.t[self.k]) *
345
346         extrapolate = False
347
348     out = np.empty((len(x), prod(self.c.shape[1:])),
349                   dtype=self.c.dtype)
350     self._ensure_c_contiguous()
351     self._evaluate(x, nu, extrapolate, out)
352     out = out.reshape(x_shape + self.c.shape[1:])
353
354     if self.axis != 0:
355         # transpose to move the calculated values to t
356         l = list(range(out.ndim))
357         l = l[x_ndim:x_ndim+self.axis] + l[:x_ndim] + l[x_ndim+self.axis:]
358         out = out.transpose(l)
359
360     return out
361
362 def _evaluate(self, xp, nu, extrapolate, out):
363     _bspl.evaluate_spline(self.t, self.c, reshape(self.c,
364
365     self.k, xp, nu, extrapolate, out)
366
367 def _ensure_c_contiguous(self):
368     """
369     C and t may be modified by the user. The Cython code
370     ensures that they are C contiguous.
371     """
372     self.c = np.ascontiguousarray(self.c)
373     self.t = np.ascontiguousarray(self.t)
```

Recap

File I/O and text mining

- ❖ File I/O: managing data in the form of files
- ❖ Reading, writing, and closing files
- ❖ File formats: plain text, XML, JSON, tables and matrices
- ❖ Text mining
- ❖ Information extraction with NLTK:
 - ❖ Lexical analysis: tokenization, stemming and lemmatization
 - ❖ Semantic analysis: parse trees
 - ❖ Domain analysis: anaphora resolution

***Data
Visualization***

***Numerical
Data Analysis
with NumPy***

***Modeling
Experimental
Data***

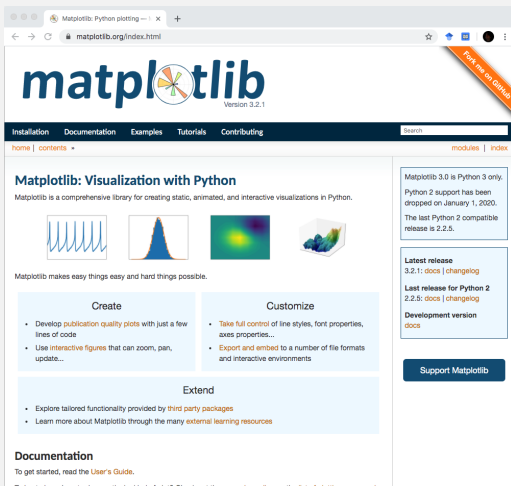
What is data visualization?

Also known as *dataviz*, data visualization is the practice of designing and creating **easy-to-communicate** and **easy-to-understand** images or other types of visual representations of complex data.

source: https://en.wikipedia.org/wiki/Data_and_information_visualization

Matplotlib: dataviz with Python

- ❖ de-facto standard library for scientific visualizations
- ❖ Many third party packages built on top of Matplotlib
- ❖ Comprehensive library for creating static, animated, and interactive visualizations

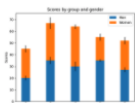


The screenshot shows the Matplotlib website homepage. At the top, the Matplotlib logo is displayed with the version number 3.2.1. Below the logo, there are navigation links for Installation, Documentation, Examples, Tutorials, and Contributing. A search bar is also present. The main content area features the heading "Matplotlib: Visualization with Python" and a sub-heading "Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python." Below this, there are four small images representing different types of plots: a line plot, a histogram, a heatmap, and a 3D surface plot. The text "Matplotlib makes easy things easy and hard things possible." is followed by three main sections: "Create", "Customize", and "Extend". Each section contains a list of bullet points describing the capabilities of the library. The "Create" section mentions developing publication quality plots and using interactive figures. The "Customize" section mentions taking full control of line styles and font properties, and exporting plots to various formats. The "Extend" section mentions exploring tailored functionality provided by third party packages and learning more about Matplotlib through external learning resources. On the right side, there are two boxes: one for the latest release (3.2.1) and one for the last release for Python 2 (2.2.5). A "Support Matplotlib" button is located at the bottom right.

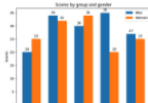
source: <https://matplotlib.org/>

What you can do with Matplotlib

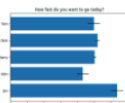
Lines, bars and markers



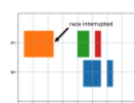
Stacked bar chart



Grouped bar chart
with labels



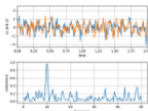
Horizontal bar chart



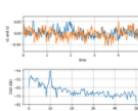
Broken Barh



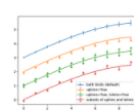
Plotting categorical
variables



Plotting the
coherence of two
signals



CSD Demo



Errorbar limit
selection

***Data
Visualization***

***Numerical
Data Analysis
with NumPy***

***Modeling
Experimental
Data***

What is numerical data analysis?

Now we know how to plot data, but what data are we plotting? Numerical analysis is the study of the methods that find ***approximate*** solutions to problems. Our numbers will be anything that can be measured, and our analyses will focus on algorithms that *approximately* make sense of that data.

source: https://en.wikipedia.org/wiki/Numerical_analysis

NumPy: numerical analysis with Python

- ❖ Fundamental package for scientific computing in Python
- ❖ The library provides:
 - ❖ A *multidimensional array object*
 - ❖ Fast and easy operations on *arrays*

The screenshot shows the NumPy website homepage. At the top, there are navigation links: Install, Documentation, Learn, Community, About Us, News, Contribute, and English. The main header features the NumPy logo and the tagline "The fundamental package for scientific computing with Python". A dark blue banner below the header announces "NumPy 2.1 released! 2024-08-18". Below this, there are six feature cards: "Powerful N-dimensional arrays", "Numerical computing tools", "Open source", "Interoperable", "Performant", and "Easy to use". At the bottom, there is a "Try NumPy" section with an interactive shell. The shell shows a terminal window with instructions: "the browser: put cell and press execute, and click on the toolbar" and "import NumPy". The shell output shows "A WebAssembly-powered Python kernel backed by Pyodide" and the code "[1]: import numpy as np".

source: <https://numpy.org>

N-dimensional array: `numpy.ndarray`

Array data structure

- ❖ ***immutable:*** can't be changed after creation
- ❖ ***n-dimensional:*** very useful for multidimensional data
- ❖ ***storage efficient:*** Python takes care of it
- ❖ can store only data of ***same type:*** can't mix `int` and `float`, for example

source: https://numpy.org/doc/stable/user/absolute_beginners.html

***Data
Visualization***

***Numerical
Data Analysis
with NumPy***

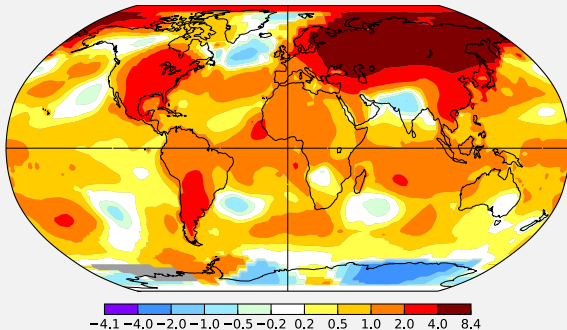
***Modeling
Experimental
Data***

NASA's GISS Surface Temperature Analysis

March 2020

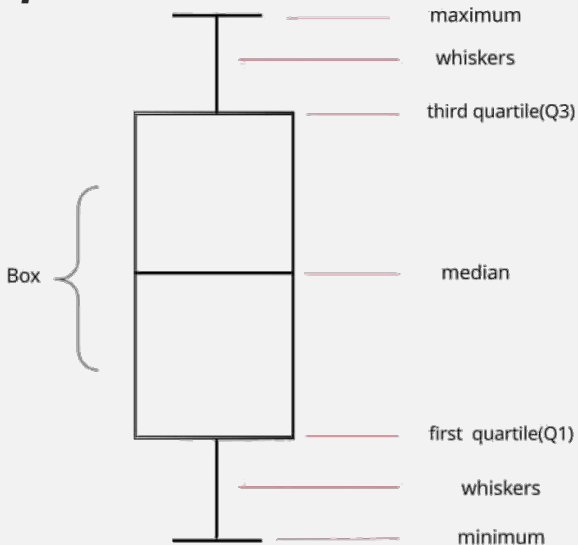
L-OTI(°C) Anomaly vs 1951-1980

1.18



- ❖ <https://data.giss.nasa.gov/gistemp>
- ❖ Collection of temperature data from thousands of meteorological stations
- ❖ Data represents *anomalies*, i.e., deviations from mean temperature measured in 1951-1980

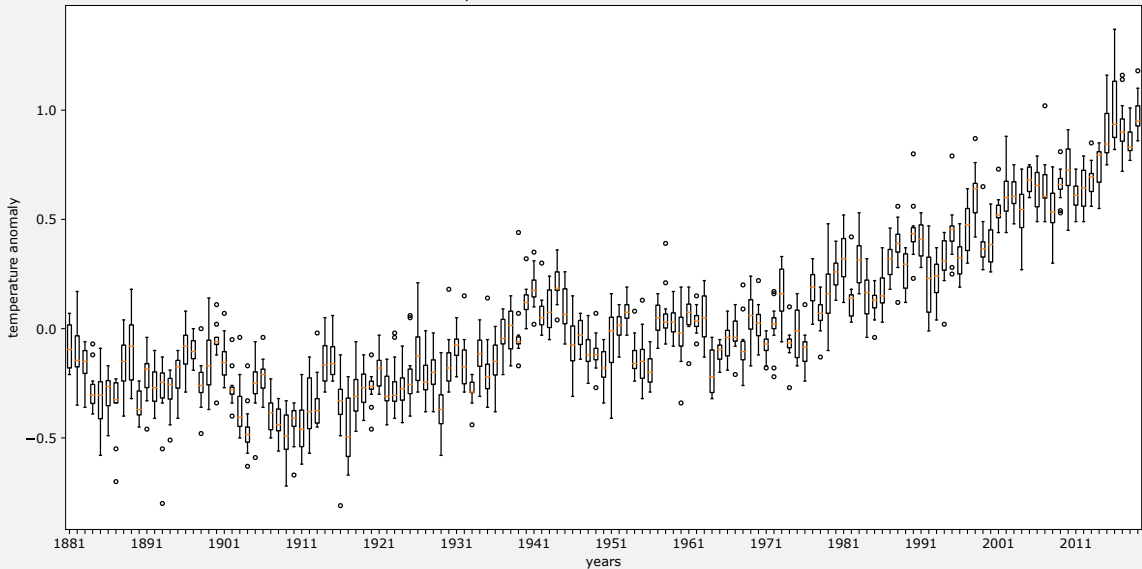
Box (whisker) plot



source: https://en.wikipedia.org/wiki/Box_plot

Whisker plot of GISS data

Temperature anomalies between 1881-2019



Further material on methods in Data Science

MIT Course 6.0002, Lectures on Understanding Experimental Data:

- <https://www.youtube.com/v/vIFKGF11Cn8>
- <https://www.youtube.com/v/fQvg-hh9dUw>

Linear regression

Linear regression is a *linear* approach for modelling a predictive relationship between some parameters and a given input:

$$X = \begin{pmatrix} X_0 \\ X_1 \\ \vdots \\ X_{N-1} \end{pmatrix}, Y = \begin{pmatrix} Y_0 \\ Y_1 \\ \vdots \\ Y_{N-1} \end{pmatrix} \rightarrow \alpha = \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_{N-1} \end{pmatrix}$$

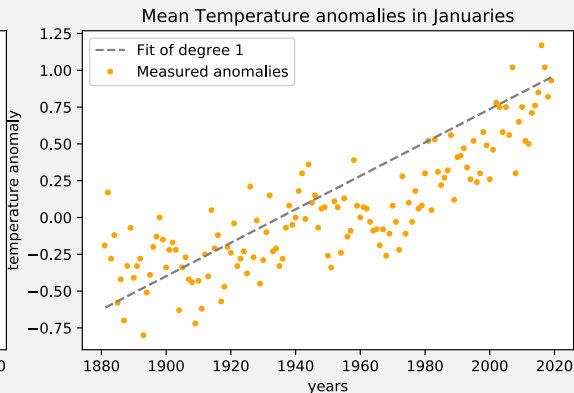
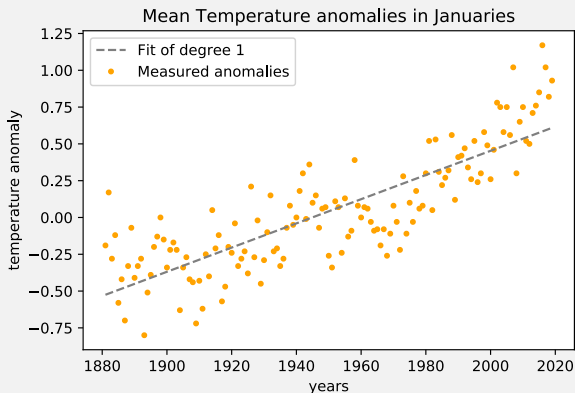
Estimator:

$$\hat{Y} = \alpha_0 + \alpha_1 X + \alpha_2 X^2 + \dots + \alpha_{N-1} X^{N-1}$$

Simple linear regression: Estimate line, i.e, estimate α_0, α_1 and set $\alpha_2 = \dots = \alpha_{N-1} = 0$

What criterion to optimize?

id est, *Which line is better?*

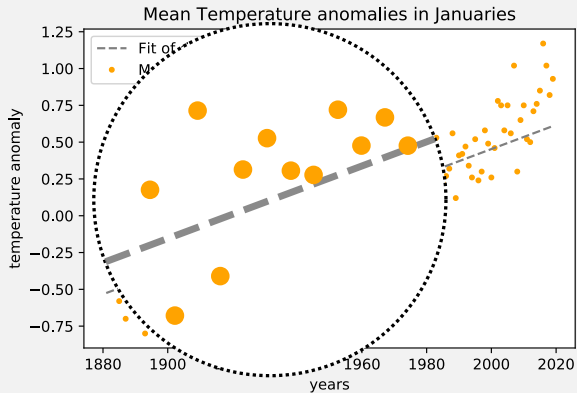


Optimization criteria

- ❖ Residual: difference predicted/observed $|Y_i - \hat{Y}_i|$
- ❖ Possible minimization criteria:
 - ❖ Sum of residuals
 - ❖ Maximum
 - ❖ Variance of residuals

$$\text{Var}_{res} := \frac{1}{N} \sum_i (Y_i - \hat{Y}_i)^2 = E[(Y - \hat{Y})^2]$$

- ❖ Minimize Var_{res} = ordinary least squares optimization

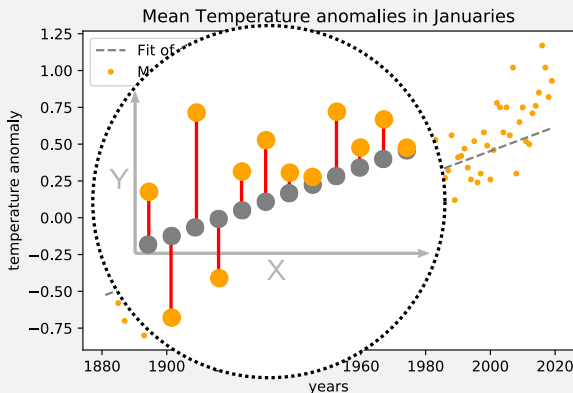


Optimization criteria

- ❖ Residual: difference predicted/observed $|Y_i - \hat{Y}_i|$
- ❖ Possible minimization criteria:
 - ❖ Sum of residuals
 - ❖ Maximum
 - ❖ Variance of residuals

$$Var_{res} := \frac{1}{N} \sum_i (Y_i - \hat{Y}_i)^2 = E[(Y - \hat{Y})^2]$$

- ❖ Minimize Var_{res} = ordinary least squares optimization



Why least squares?

Three advantages:

1. Penalizes large deviations from the observed data very strongly and sums over all data points;
2. Finding the polynomial that minimizes the variance can be done efficiently via least squares optimization methods;
3. Minimizing the variance guarantees that there is one and only one solution.

Coefficient of determination R^2

How to measure quality of fit?

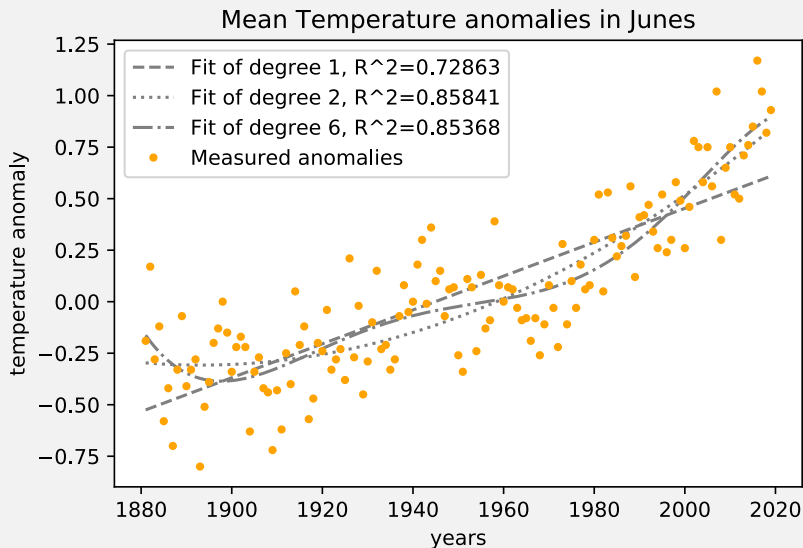
Recall: Ordinary Least squares optimization minimizes Var_{res}

R^2 is a normalized measure thereof:

$$R^2 := 1 - \frac{Var_{res}}{Var(Y)}$$

i.e., $R^2 \rightarrow 0$ bad fit, $R^2 \rightarrow 1$ good fit

Separated training from testing



Quiz

❖ True or false?

- ❖ The residual is the distance between an observed and its predicted data point
- ❖ Linear regression always minimizes the variance of residuals
- ❖ Linear regression is the task of fitting a line to a set of data points
- ❖ Ordinary least squares always minimizes the variance of residuals

❖ How does linear regression measure the distance between an observed and its predicted data point?



(a)



(b)



(c)

Quiz

True or false?

- ❖ The residual is the distance between an observed and its predicted data point true
 - ❖ Linear regression always minimizes the variance of residuals false
 - ❖ Linear regression is the task of fitting a line to a set of data points false
 - ❖ Ordinary least squares always minimizes the variance of residuals true
- ❖ How does linear regression measure the distance between an observed and its predicted data point? (a)



(a)



(b)



(c)

Recap

Summary

- ❖ Plots with `matplotlib`:
 - ❖ Line- and scatter plot
 - ❖ Histogram
 - ❖ Whisker (box) plot
- ❖ Numpy
 - ❖ `ndarray` data type
 - ❖ Vectorized operations, broadcasting
 - ❖ Curve fitting: `polyfit()`
- ❖ Realistic data analysis: climate trends

What comes next?

- ❖ Draw your first plots with `matplotlib`
- ❖ Further reading about NumPy: Chapter 2 of the “Python Data Science Handbook”:
<https://jakevdp.github.io/PythonDataScienceHandbook/>
- ❖ Due date for this week’s exercises is **Wednesday, December 11, 2pm, 2024.**

Next lecture: Pandas! ...