Automated Feature Extraction with Machine Learning and Image Processing

PD Stefan Bosse

University of Siegen - Dept. Maschinenbau
University of Bremen - Dept. Mathematics and Computer Science

Data and Data Features

- Metrics and taxonomy of Data
- Features of Data
- Analysis of Data

Data

In general, data and their values can be divided into:

- **Scalar values**, such as temperature, age, etc.
- **Series** of scalar values, such as time series
- **Vector and matrix values** such as images
- **Composite data**, i.e. data structures (records)
- **Temporal-spatial** data, i.e. time-dependent spatial data series, $D=\{D(p,t)=\{d(p)_i\}\}$ with $i = \{1,2,3,..,t\}$, $p=\langle x,y,..\rangle$

Data have dimensionality $\mathbb{X}^N$

- The values of $\mathbb{X}$ are a dimension from the discrete number set $\mathbb{N}$, real number set $\mathbb{R}$, and the time scale $\mathbb{T}$ or any categorical value sets $\mathbb{S}$ (or subsets thereof), e.g., $\mathbb{X}=\mathbb{R} \times \mathbb{R} \times \mathbb{N}$.

Data Reduction

- The aim of data analysis is to reduce input data in terms of size and dimensionality:

$$P(X^N) : X^N \rightarrow Y^M$$
$$|Y| < |X|, M < N$$

Materials science, metrology, and construction engineering uses:

- Commonly metric input variables;
- Often metric or categorical output variables (incl. Boolean variables)

Data Reduction

```
function isRaining(temp,sunrad,moisture) = {
  if (temp < 0)                    FALSE
  else if (temp > 40)              FALSE
  else if ((sunrad-moisture) > 30) FALSE
  else                             TRUE
}
```

Ex. 1. A R example from measurement technology with a data reduction function $\mathbb{R}^3 \rightarrow \mathbb{B}$

Data classes

Numerical and Metric values

These are values that are countable and where you can meaningfully define relations (such as smaller or larger), i.e. for all real and integers.

- Examples: temperature, length, density, pore size, elongation, force, location, time

Categorical values

These are symbolic values for which either no (meaningful) order relation exists or where at least no differences can be formed.

- Examples: nationality, color names (red < yellow???), Damage type, characteristic feature (anomaly?)

```
m = 1
m = [1.0,1.5,2.5]
c = 'A'
c = ['A','B','A']
c = [TRUE,FALSE,TRUE]
c = factor(m,levels=[1,1.5,2,2.5],labels=['A','B','C','D'])
```

Ex. 2. R examples of numerical and categorical values and conversion (factorization)

Scaling of numerical values

Interval scaled

For this type of attributes, only differences (addition or subtraction) make sense. For example, the temperature measured in °C or °F is interval scaled. If it is 20 °C on one day and 10 °C on the following day, it makes sense to talk about a temperature drop of 10 °C, but it does not make sense to say that it is twice as cold as the day before (C(K)~K, but F(K)/~K!!).

Ratio scaled

Here you can calculate both differences and ratios between values. For example, for age, one can say that someone who is 20 years old is twice as old as someone who is 10 years old, and 20 is > 10.

Order relations

Nominal

The attribute values in the domain are unordered and therefore only equality comparisons make sense. That is, we can only check whether the value of the attribute is the same for two specific instances or not. For example, gender is a nominal attribute.

Ordinal

The attribute values are ordered and thus equality comparisons (is one value equal to another?) and relational comparisons (is one value smaller or larger than another?) are allowed, although it may not be possible to quantify the difference between the values!

Data Aggregations

1. Vectors (columns, one dimensional)
2. Lists (field record, one dimensional)
3. Matrices (two dimensional)
4. Arrays (multi dimensional)
5. Tables (data frames organized in rows and columns)

```
v = c(4)              v = [1.0,1.5,2.5]
v[1] = 1.2
l = list(a=1,b=2)     l = {a=1,b=2}     l={1.0,1.5,2.5}
l$a = 9
m = matrix(0,nrow=2,ncol=3)
m = [1,2,3;4,5,6]
a = array(0,dim=[3,2,4])
df = data.frame(a={1,2,3},b={3,4,5})
```

Data classes (longitudinal)

- Sensor and measurement data variables (both categorical and metric) can be further distinguished in:

Static

The variable s is not variable in time or is to be regarded as stationary (immutable) in a significant time interval $t \in [t_0, t_1]$.

Dynamic

The variable $s(t)$ is time-dependent and forms a data series (or time vector) $s(t)=\{s_0,s_1,..s_t\}$ in the case of discrete acquisition, i.e., we are talking about longitudinal data.

A digitized sensor signal is always discrete in time, but the physical variable that the sensor measures is continuous in time (note the sampling theorem)

Data

Data sets as matrices

- Data can be represented in **matrix form** as matrix ***D*** (analogy to table form) [1]:

$$\mathbf{D} = \left(\begin{array}{c|cccc} & X_1 & X_2 & \cdots & X_d \\ \hline \mathbf{x}_1 & x_{11} & x_{12} & \cdots & x_{1d} \\ \mathbf{x}_2 & x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{x}_n & x_{n1} & x_{n2} & \cdots & x_{nd} \end{array}\right)$$

- The vector $\boldsymbol{X}$ is the set of all variables X_i and represent the columns of the matrix $\boldsymbol{D}$:

$$\vec{X} = (X_1, X_2, .., X_d)$$

- Each row $\boldsymbol{x}_j$ is a record of the variable set $\boldsymbol{X}$=$\{X_i|i=1,d\}$ with values x and represent an individual example, instance, experiment, entitie, object, and feature vector as a d-digit tuple, depending on the application and objective:

$$\vec{d}_j = \vec{x}_j = (x_{j,1}, x_{j,2}, .., x_{j,d})$$

```
df = data.frame(
  X1={'x1,1','x1,2','...'},
  X2={'x2,1','x2,2','...'},
  X3={'x3,1','x3,2','...'}
)
print(df)
      X1      X2     X3 == X
1  "x1,1" "x2,1" "x3,1"
2  "x1,2" "x2,2" "x3,2"
3  "..."  "..."  "..."
```

Input and Output Variables

- The variable set is composed of input and output variables: $\boldsymbol{X}_{xy}=\boldsymbol{X} \cap \boldsymbol{Y}$
- Sensors are commonly input variables *X*
- Statements are output variables *Y*, i.e. results that can be derived from the input variables (by a function *F*):

$$\vec{X}_{xy} = (X_1, X_2, .., X_u, Y_1, Y_2, .., Y_v)$$
$$\vec{X} = (X_1, X_2, .., X_u)$$
$$\vec{Y} = (Y_1, Y_2, .., Y_v)$$
$$\vec{d}_j = (x_{j,1}, x_{j,2}, .., x_{j,u}, y_{j,1}, y_{j,2}, .., y_{j,v})$$
$$F(\vec{X}) : \vec{X} \rightarrow \vec{Y},$$

with $u+v=d$.

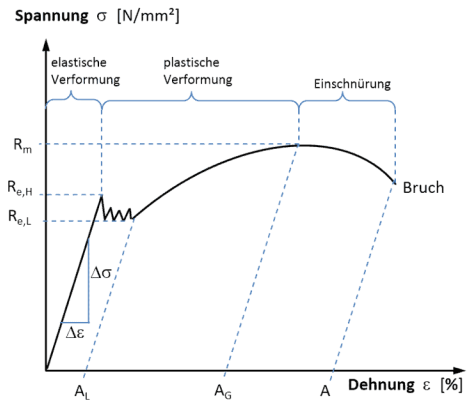
Example of a data matrix

- Botanical data set with geometric (numerical) properties of a plant and categorical classification:

	Sepal length X_1	**Sepal width** X_2	**Petal length** X_3	**Petal width** X_4	**Class** X_5
$\mathbf{x}_1$	5.9	3.0	4.2	1.5	Iris-versicolor
$\mathbf{x}_2$	6.9	3.1	4.9	1.5	Iris-versicolor
$\mathbf{x}_3$	6.6	2.9	4.6	1.3	Iris-versicolor
$\mathbf{x}_4$	4.6	3.2	1.4	0.2	Iris-setosa
$\mathbf{x}_5$	6.0	2.2	4.0	1.0	Iris-versicolor
$\mathbf{x}_6$	4.7	3.2	1.3	0.2	Iris-setosa
$\mathbf{x}_7$	6.5	3.0	5.8	2.2	Iris-virginica
$\mathbf{x}_8$	5.8	2.7	5.1	1.9	Iris-virginica
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$\mathbf{x}_{149}$	7.7	3.8	6.7	2.2	Iris-virginica
$\mathbf{x}_{150}$	5.1	3.4	1.5	0.2	Iris-setosa

- Measurement data set

Computed Strain-stress diagram



[www.precifast.de/elastizitaetsmodul-e-modul]

Measurement data from strain test

Strain [mm]	Force [kN]
0	0
0.1	0.2
0.2	0.7
0.3	1.5
0.4	1.7
0.5	1.9
0.6	2.0
0.7	0.2
0.8	-0.5

```
tt = data.frame(
  Strain = [0.0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8],
  Force  = [0.0,0.2,0.7,1.5,1.7,1.9,2.0,0.2,-0.5]
)
```

Ex. 4. Measure data stored in a R data.frame

Attributes

- The measured variables X_1 to X_4 are metric data variables, the variable $X_5=y$ is a categorical variable!
- The measured variables X_1 to X_4 (i.e. sensors) are called **attributes** because they are properties and descriptive variables of the target variable y.

High-dimensional Data

- Images $\boldsymbol{I}=\boldsymbol{I}(x,y[,z])$ are commonly two- or three-dimensional spatial data, organised in rows and columns (and levels)
- Spatiotemporal data $\boldsymbol{T}=\boldsymbol{T}(x,y[,z],t)$ is commonly three- or four dimensional and organised in rows, columns (levels), and discrete time points t.

Sensors

Which sensors and measurement data do you know:

Sensor

- Measurement
  - Physical quantities such as temperature, strain, stress, time, absorption
  - Merged survey variables (e.g. ensemble mean values, outliers, ..)
- When measuring with sensors, a distinction is made between:
  - Single or single measurements (single shot)
  - Repeated measurements of the same physical quantity (averaging..)
  - Series of measured values, especially time-resolved data series:
    $\boldsymbol{D} = \{d_1, d_2, .., d_n\}$, where commonly $\Delta t(d_i, d_{i+1})$ is constant

Sensor

- Socio-technical systems, surveys
  - Survey variables (answers to questions) are sensors of individual people
  - Merged survey variables (e.g. ensemble mean values) are sensors of groups of people
- Generally available data
  - Social networks and social media
  - Databases of authorities, etc.

Sensor model

- A sensor is a transducer (indicator for a property that is not directly measurable)
- A sensor therefore generally maps a physical quantity *x* to another quantity *y*:

$$S(x) : x \rightarrow y, K : correct(x \rightarrow y)$$

- There is usually a calibration function $K(f, x, y)$
- Examples are:
  - Pressure → Voltage, Radiation → current, etc.
  - Social networking → Numerical radius value, votes → Politics, i.e., **Assignment of numbers to objects or events according to established rules**

Sensor data

- Sensors *S* are data sources *d* of physical, sociological or other natural variables *x* that cannot be detected directly
- The data values (numeric) will be in a definable interval
- Knowledge of the value interval is important for later data processing, analysis, and machine learning!
- Categorical values are also defined by a set

$$S(x) : x \rightarrow d$$
$$d \in [a,b] \Rightarrow \{v_0, v_1, .., v_i\}$$

Measurement and sensory systems

The origin of data for analysis and machine learning!

A sensor rarely comes alone.

Measurement methods

A distinction is made between two different measurement methods:

Passive measuring method (P)

The sensory values are the result of an intrinsic property (e.g., density) or already existing external variables (temperature). The stimulus of the measurement is the component, the person, the environment.

Active measurement methods (A)

There is an active stimulus whose response signal is detected by the sensor. An example is the ultrasonic measurement method with guided waves. The sensor signal is always dependent on the stimulus. In sociology, for example, the stimulus is a catalog of questions in a survey, the answers are the sensor variables.

Acoustic Emission measuring technologies can belong to both classes,

Acoustic Emission measuring technologies can belong to both classes,

Guided Ultrasonic Waves belong to class A, and

Acoustic Emission measuring technologies can belong to both classes,

Guided Ultrasonic Waves belong to class A, and

X-ray imaging belongs commonly only to class *P*.

Signal Features

1. Statistical Features
2. Spatial Features (Images, geometric features)
3. Frequency and spectral Features /time and space)
4. Differences to reference signals
5. Transformed Signals

Statistical Features

Assumption: Data series

- But any image can be transformed into a pixel data series, too!
- Any column of a data table is a data series (but independent values and unordered!)

There is a data series ***d*** related to one variable *x* (from sensor *s*):

$$\vec{d} = \{d_1, d_2, \ldots, d_n\}, s : x \rightarrow d$$

Statistical Features

Feature	Formula
Sample Size	n
Extrema	$\min(x), \max(x)$
Sample Mean	$\overline{x} = \frac{\sum_{i=0}^{n} x_i}{n}$
Standard Deviation	$s = \sqrt{\frac{\sum_{i=0}^{n} \left(x_i - \overline{x}\right)^2}{n}}$
Sample Variance	$s^2 = \frac{\sum_{i=0}^{n} \left(x - \overline{x}_i\right)^2}{n}$

... and many more

Statistical Features

```
use math
Force  = [0.0,0.2,0.7,1.5,1.7,1.9,2.0,0.2,-0.5]
statsForce = fivenum(Force)
statsForce$std = sd(Force)
cprint(statsForce)
{min : -0.5 , q1 : 0.2 , median : 0.7 , mean : 0.855 ,
 q3 : 1.7 , max : 2, sd: 0.93}
```

Ex. 5. Statistical analysis of data series or vectors in R

Statistical Features

Feature	Formula
N-th moment about point *a*, e.g., $a = \bar{x}$	$\mu_n(a) = \sum (x-a)^n P(x)$
Gaussian Distribution	$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2 / 2\sigma^2}$
Fisher Skewness	$\gamma_1 = \frac{\mu_3}{\mu_2^{3/2}} = \frac{\mu_3}{\sigma^3}, \sigma = \sqrt{\mu}_2$

... and many more

Statistical Features

```
use math
Force  = [0.0,0.2,0.7,1.5,1.7,1.9,2.0,0.2,-0.5]
mn = moment(Force,order=2,central=TRUE)
print(mn)
```

Ex. 6. Higher order moment analysis of data series or vectors in R

Statistical Features

Moment ordinal	Moment			Cumulant	
	Raw	Central	Standardized	Raw	Normalized
1	Mean	0	0	Mean	—
2	–	Variance	1	Variance	1
3	–	–	Skewness	–	Skewness
4	–	–	(Non-excess or historical) kurtosis	–	Excess kurtosis
5	–	–	Hyperskewness	–	–
6	–	–	Hypertailedness	–	–
7+	–	–	–	–	–

Fig. 1. Meaning of higher order moments (Wikipedia)

Statistical Features

Statistical analysis is applied to the same static variable X with unordered values from repeated measurements of X under the same conditions

Statistical Features

Statistical analysis is applied to the same static variable X with unordered values from repeated measurements of X under the same conditions

Statistical measures for data series (e.g., time-dependent) of dynamic variables with values from measurements under different conditions are not valid ("non-sense"). But statistical measures can be still used as signal features posing a correlation between the input signal and the target features (e.g., damages), e.g., the mean value or higher order moments.

Statistical Features

An ordered data series $\{d_i\}$ can be considered as an ordered series of different variables $\{X_i\}$!

- Finally, all statistical features create a new input vector (for ML) $\boldsymbol{X}^f$ derived from the original input variables $\boldsymbol{X}$:

$$\mathrm{Stat}(X) : X \rightarrow X^f$$

$$X = (X_1, .., X_i), X^f = \left(X_1^f, .., X_j^f\right), i \gg j$$

Image Features

Low Level

1. Histogram $H(\boldsymbol{I})=\{h_1,..,h_k\}$, where each histogram variable represents the number of pixels within an intensity interval $[i,i+\Delta]$ (can be split into separate RGB histograms for colour images)
2. Average (mean) intensity $\boldsymbol{I}$, noise (intensity distribution statistics)
3. Extrema intensities $min(\boldsymbol{I})$, $max(\boldsymbol{I})$
4. Frequency spectrum $\boldsymbol{F}(\boldsymbol{I})=\{f_1,..,f_s\}$, where each frequency represents a wavenumber in the wave room
5. Intensity gradients and profiles along lines (axis)
6. Addition and subtraction of images (using, e.g., base-line reference images)

Image Features

High Level

1. Intensity gradients
2. Edges
3. Geometrical figures
4. Object clusters
5. Regions-of-interest (ROI), defined by bounding boxes or closed polygons
6. Labelled and classified ROIs
7. Feature point markings
8. Threshold Binarization (dimensionality reduction and feature amplification)

Transformations

Reduce Picture Dimension

A simple way to reduce the dimension of our feature vector is to decrease the size of the image with decimation (downsampling) by reducing the resolution of the image.

- If the color component is not relevant, we can also convert pictures to grayscale to divide the number dimension by three.
- Intensity homogenisation using transfer functions

A two-dimensional mathematical matrix is a grayscale image, a three-dimensional mathematical matrix is a color image.

Color Spaces

1. RGB: Three channels per pixel for each color R(ed), G(reen), B(lue) providing the color intensity
2. RGBA: RGB with an additional alpha (tranparency) channel
3. Grayscale: One channel per pixel providing the intensity (average or luminescence)

Conversion from color to grayscale uses a specific color model transformation. Be careful.

Color Spaces

1. Average RGB ⇒ Grayscale transformation

$$I(x,y) = \frac{R(x,y) + G(x,y) + B(x,y)}{3}$$

2. More natural color weighted luma RGB ⇒ Grayscale transformation

$$I(x,y) = 0.299R(x,y) + 0.587G(x,y) + 0.114B(x,y)$$

3. RGBA ⇒ Grayscale transformation

$$I(x,y) = \frac{f(R(x,y)) + f(G(x,y)) + f(B(x,y))}{3}$$

$$f(i,a) = (1-a)_k + a_i$$

$$a = \frac{A(x,y)}{k}$$

Look-up Tables

Intesnity distributions can be transformed with continous functions e.g., an exponential gamma correction, or by using a look-up table.

- A look-up table can be considered as a discrete mapping function $f(x)$: $x \rightarrow y$, whereby the index, i.e,, a specific row, is given by the (discrete) x value, and y is the value in the specific row.
- Only meaningful for small and discrete intensity value ranges, e.g., 8 Bit [0,255]
- Only rough approximation of an intensity transfer function with continous value distributions, but fast method!

Look-up Tables

```
use plot,math,imager
vals = [1,3,5,6,7.5,8,8.5,9,9.5,10]
mylut = lut(vals,range=[0,9])
img = matrix(runif(100)*10,10,10)
img.isca = mylut(img)
plot(img,auto.scale=TRUE)
hist(img,breaks=20)
plot(img.isca,auto.scale=TRUE)
hist(img.isca,breaks=20)
```

Ex. 7. LUT function in R(+) applied to a random matrix

Histogram of Oriented Gradient

The HOG feature descriptor is a popular technique used in computer vision and image processing for detecting objects in digital images.

> The HOG descriptor is a type of feature descriptor that encodes the shape and appearance of an object by computing the distribution of intensity gradients in an image.

Histogram of an Image

```
use math,plot
img = matrix(runif(100),10,10)
plot(img,auto.scale=TRUE)
hist(img,ylim=[0,1])
img[img>0.5]=1
plot(img,auto.scale=TRUE)
hist(img,ylim=[0,1])
```

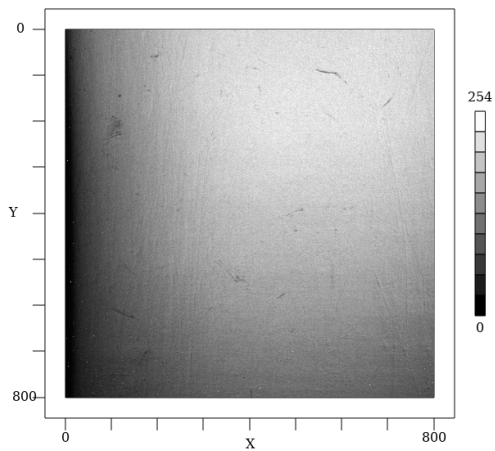
Ex. 8. Histogram of a uniformly distributed random image and image binarization

Intensity Homogenization

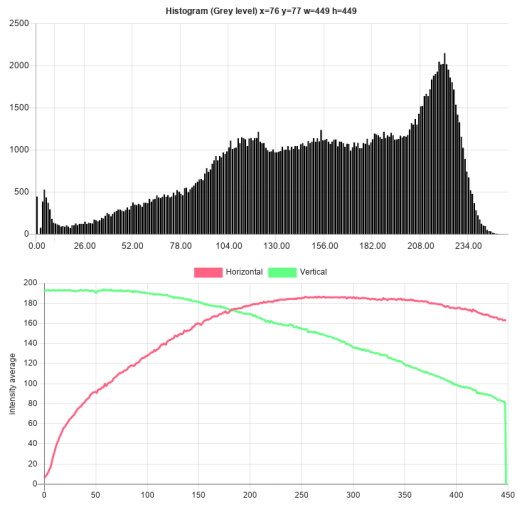
The intensity of an image can vary significantly across the spatial x-y plane, e.g., as a result of the measuring method and conditions.

- Image processing and transformation algorithms can be sensitive to intensity inhomogeneity.
- Algorithms:
  - Histogram Equalization (HE), Brightness Preserving Bi-Histogram Equalization (BBHE)
  - Geometrical Image Intensity Equalization
  - Model-based (physical model of illumination)

Microcracks Image



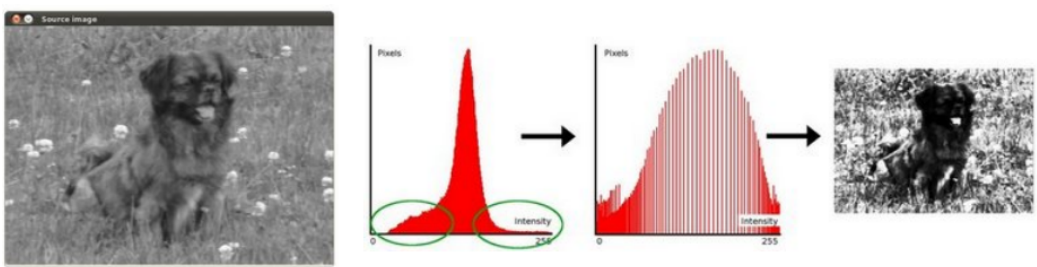
Intensity Profiles



Histogram Equalization

[https://docs.opencv.org/3.4/d4/d1b/tutorial_histogram_equalization.html]

- It is a method that improves the contrast in an image, in order to stretch out the intensity range.
- From the image below, you can see that the pixels seem clustered around the middle of the available range of intensities.



[https://github.com/YuAo/Accelerated-CLAHE]

- Histogram equalization (HE) is a method in image processing of contrast adjustment using the image's histogram.
- This method usually increases the global contrast of many images, especially when the usable data of the image is represented by close contrast values.
- Through this adjustment, the intensities can be better distributed on the histogram.

This allows for areas of lower local contrast to gain a higher contrast and **attention in visual inspection**.

- Histogram Equalization stretch out this range.
- Equalization implies mapping one distribution (the given histogram) to another distribution (a wider and more uniform distribution of intensity values) so the intensity values are spread over the whole range.
- To accomplish the equalization effect, the remapping should be the cumulative distribution function (cdf). For the histogram $\boldsymbol{H}(i)$, its cumulative distribution $\boldsymbol{H}_{\text{cd}}(i)$ is (N: Number of pixels):

$$H_{cd}(i) = \frac{\sum_{0 \leq j < i} H(j)}{N}$$

- Finally, we use a simple remapping procedure to obtain the intensity values of the equalized image:

$$I_{eq}(x,y) = H_{cd}(I(x,y))$$

Cummulative Distribution Function (CDF)

```
use math,plot
m=matrix(runif(100),10,10)
h=hist(m,ylim=[0,1],breaks=20,plot=FALSE)
print(h$density)
cdf=vector('numeric',length(h$density))
for (i in 1:length(h$density)) {
  cdf[i]=sum(h$density[1:i])
}
plot(cdf,auto.scale=TRUE,main='CDF')
```

Ex. 9. Higher order moment analysis of data series or vectors in R

Spatial Image Intensity Equalization

This simple Histogram Equalization is not sensitive to spatial intensity inhomogeneities and variations! Spatial uniform intensity distributions are assumed!

- Intensity variations can be a result of a statistical process or due to the measuring technology and conditions
  - Variation can be considered as an overlay (addition) to the "real" measuring signal $s(x,y)v(x,y)+n(x,y)$, and noise n
- Methods based on a spatial filtering of the images use the assumption that the bias field (intensity inhomogeneity) consists of a low spatial frequency intensity variation ⇒ Applying a High-pass filter in the wavenumber space!?
- Low-pass filtering methods can be used to extract non-uniformity

Trivial Approach

- Assumption:
  1. There is only one axis in the image with low-frequency intensity variations due to inhomogeneous illumination
  2. The image content has statistically averaged homogeneous, i.e., equally distributed (small) features like cracks
- The mean image intensity $I_{\text{mean}}(p)$ can be computed along a line $l(p)$ (parametric equation, orientation by visual inspection along the strongest intensity variation/gradient) by using the average intensity along the perpendicular line at each point p:

$$x_l = x_0 + ap$$
$$y_l = y_0 + bp$$
$$l(p) : p \rightarrow (x, y)$$
$$l_\perp(p, q) : q \rightarrow (x, y)$$

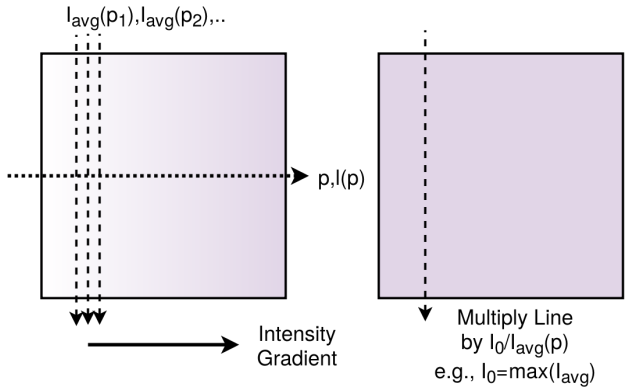


Fig. 2. (Left) Computing the average intensity $I_{avg}(p)$ perpendicular to a line along the intensity gradient (Right) Correct all pixels perpendicular to the correction line with a equalization factor

Contrast Limited Adaptive Histogram Equalization

CLAHE (Contrast Limited Adaptive Histogram Equalization) is an algorithm for enhancing local contrast in images, and is frequently used in application areas like underwater photography, traffic control, astronomy, and medical imaging. [https://github.com/YuAo/Accelerated-CLAHE]

CLAHE can also be used in the tone mapping operation of displaying a HDR (High Dynamic Range) image.

- Adaptive histogram equalization (AHE) differs from ordinary histogram equalization in the respect that the adaptive method computes several histograms, each corresponding to a distinct section of the image, and uses them to redistribute the lightness values of the image.
- It is therefore suitable for improving the local contrast and enhancing the definitions of edges in each region of an image.
- AHE has a tendency to overamplify noise in relatively homogeneous regions of an image.
    - A variant of adaptive histogram equalization called contrast limited adaptive histogram equalization (CLAHE) prevents this by limiting the amplification.

1. Compute the neighborhood histogram for each pixel in the image.
2. Clip each histogram at a predefined value and redistribute the clipped histogram equally among all the histogram bins.
3. Compute the CDF (Cumulative Distribution Function) and transformation function for each pixel using the clipped histogram.
4. Apply the transformation function to each pixel to get the equalized image.

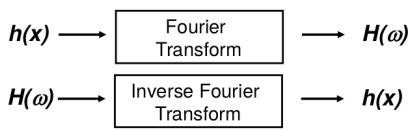
Alg. 1. The basic CLAHE algorithm

Frequency Transformation

- Time-dependent signal $s(t)$ can be transformed in the frequency space $S(\omega)$ by using a frequency transformation, e.g., Discrete Fourier Transformation (DFT):

$$|DFT(s)| : s(t) \rightarrow S(\omega)$$
$$DFT(\{x_n\}) : \{x_n\} \rightarrow \{X_k\}$$
$$X_k = \sum_{0 \leq n < N} x_n e^{\frac{-2i\pi}{N}k}$$
$$X_k = \sum_{0 \leq n < N} x_n \left(\cos\left(\frac{2\pi}{N}kn\right) - i\sin\left(\frac{2\pi}{N}kn\right)\right)$$

- The DFT transforms a series of complex numbers $\{x_n\}$ into a sequence of complex numbers $\{X_k\}$.
  - The transformation is reversible (as long as complex numbers, i.e., magnitude and phase, is preserved).
- Low-, High-, and Bandpassfiltering can be performed by applying a mask function to the frequency distribution $\{X_k\}$ and transforming back into time-space (blending in frequency space)



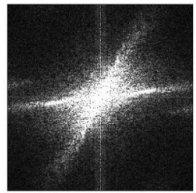
[TU Graz, IVU_frequency_2017]

2D DFT

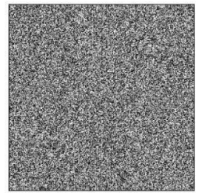
- Images can be transformed into the frequency space, too, called wavenumber space
- A two-dimensional (2D) DFT is used (output is a matrix, too)

$$IF(k,l) = \sum_{0 \le m < N} \sum_{0 \le n < N} I(m,n) e^{-2i\pi\left(k\frac{m}{N} + l\frac{n}{N}\right)}$$

Source image $f[x,y]$



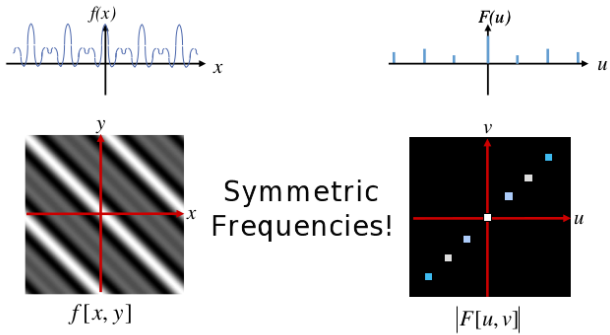
Fourier spectrum $|F[u,v]|$



Fourier phase $\phi[u,v]$

[TU Graz, IVU_frequency_2017]

The signal frequency distribution is symmetric!



[TU Graz, IVU_frequency_2017]

Wavelet Decomposition

Disadvantage of Fourier transformations is the lost of the time or spatial information.

- A solution can be the application of a moving window of size $m \ll n$, with n as the sample size (time signal: number of time samples, image: width and height).
  - But: The Fourier transformation delivers $m/2$ frequencies
  - If the window size is lowered, the time or spatial resolution increases, but the frequency resolution decreases!

Wavelet decomposition is a way of breaking down a signal in both space and frequency. In the case of pictures, this means breaking down the image into its horizontal, vertical, and diagonal components.

Wavelet Decomposition

[Parida et al.,2017]

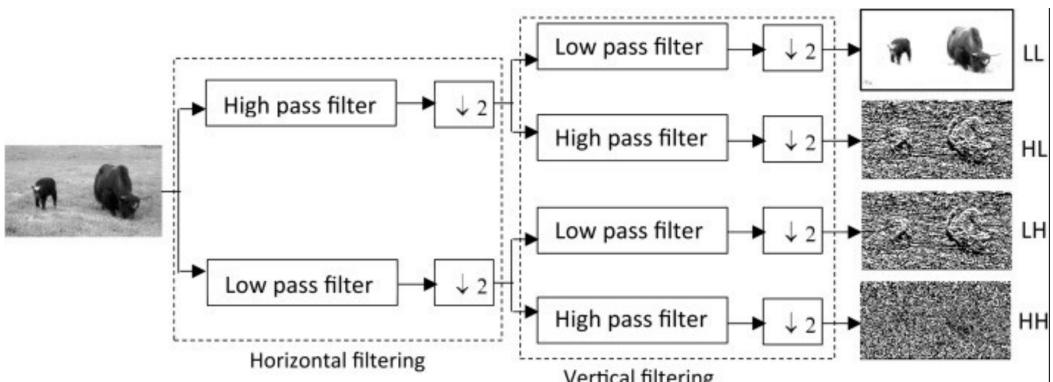


Fig. 3. Decomposition of an image 2-D discrete wavelet transform with filter banks (2-D DWT)

Wavelet Decomposition

[Bosse et al., doi:10.3390/computers10030034]

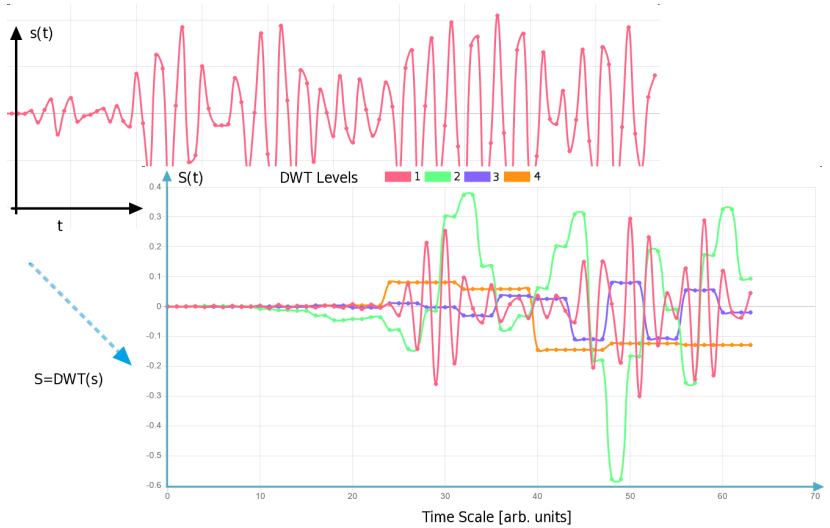


Fig. 4. Example of a DWT signal decomposition of a US time-dependent signal

An image wavelet is a two-dimensional function $\Phi(x,y)$, and we need two.dimensional convolution operations. Time consuming!

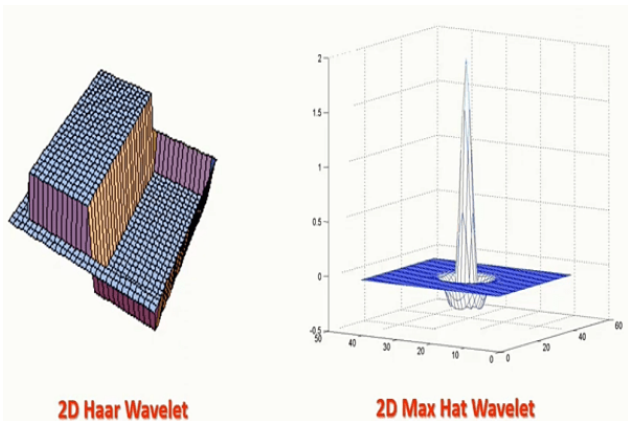


Fig. 5. Examples of 2D wavelets (Left) Haar (Right) Max Hat

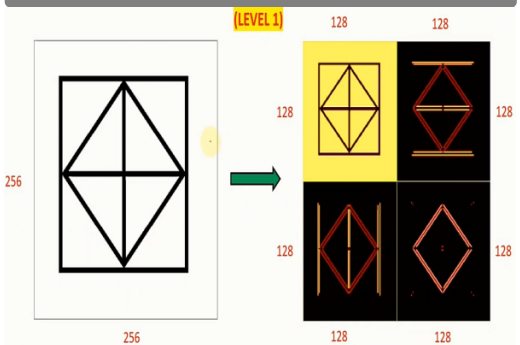
[https://www.section.io/engineering-education/wavelet-transform-analysis-of-images-using-waveletanalyzer-toolbox-in-matlab/]

Wavelet Decomposition

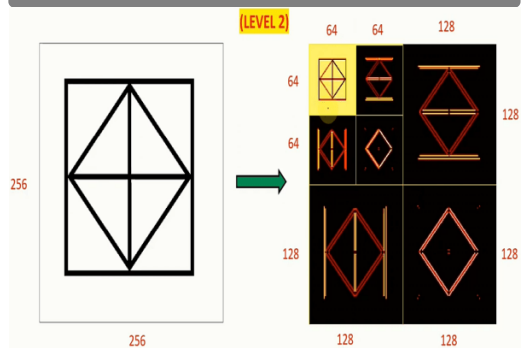
- Instead performing a 2-D wavelet convolution, we can apply the 1-D transformation to the rows and columns of images as separable 2-D transformations.
- In most applications where wavelets are used for image processing, this approach is more practical due to the low computational complexity of separable transformations.
- Each decomposition reduces the image size by a factor 2 in each dimension: DWT: $M \times M \rightarrow M/2 \times M/2$;
- The DWT decomposition can be repeated by using the ouput of the previous level

Wavelet Decomposition

Wavelet 1st Level



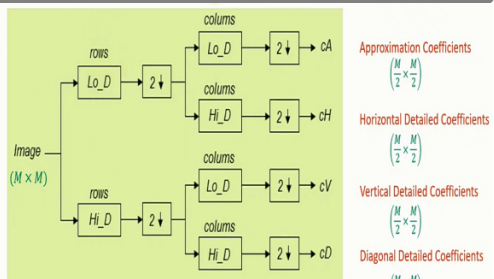
Wavelet 2nd Level



[https://www.section.io/engineering-education/wavelet-transform-analysis-of-images-using-waveletanalyzer-toolbox-in-matlab/]

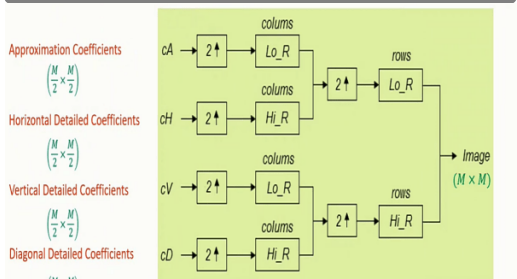
Wavelet Decomposition and Reconstruction

Wavelet Image Decomposition



Lo_D: Low Pass Wavelet Decomposition Filter
Hi_D: High Pass Wavelet Decomposition Filter
2 ↓: Down Sampled by 2.

Wavelet Image Reconstruction



Lo_R: Low Pass Wavelet Re-construction Filter
Hi_R: High Pass Wavelet Re-construction Filter
2 ↑: Up Sampled by 2.

Image Gradient

The (intensity) gradient of an image is the vector $\nabla \boldsymbol{I}(x,y)$. It is characterized by a magnitude *m* and a direction φ in the image:

$$\nabla I(x,y) = (\frac{\partial I(x,y)}{\partial x}, \frac{\partial I(x,y)}{\partial y})^t.$$

$$m = \sqrt{(\frac{\partial I(x,y)}{\partial x}^2 + \frac{\partial I(x,y)}{\partial y}^2)},$$

$$\phi = \arctan(\frac{\partial I(x,y)}{\partial y} / \frac{\partial I(x,y)}{\partial x}).$$

Image Laplacian

Another important image transformation is the Laplacian of an image with intensity $\boldsymbol{I}(x,y)$ that is defined by:

$$\nabla^2 I(x,y) = \frac{\partial^2 I(x,y)}{\partial x^2} + \frac{\partial^2 I(x,y)}{\partial y^2}.$$

- Invariant to image rotations.
- The laplacian is often used in image enhancement to increase contour effects

$$I'(x,y) = I(x,y) - c\nabla^2 I(x,y).$$

- Higher sensitivity to noise than the gradient.

Edge Detection

Two main strategies:

1. Gradient strategy: detection of the local extrema in the gradient direction.
2. Laplacian strategy: detection of zero-crossing.

- These strategies rely on the fact that edges correspond to 0-order discontinuities of the intensity function.
- The derivative computation requires a pre-filtering of the images.
    - For instance: linear filtering for zero mean noises (e.g. white Gaussian noise and Gaussian filter) and non-linear filtering for impulse noise (median filter).
- Since all edge detection results are easily affected by the noise in the image, it is essential to filter out the noise to prevent false detection caused by it. To smooth the image, a Gaussian filter

Edge Detection: Sobel Derivative Filter

The Sobel filter is a x- and y-sensitive gradient filter by using a convolution operation with two 3×3 kernels.. The x- and y-gradients are merged finally in one image.

```
use math,imager,plot
img.sobel <- sobelEdges(img,blur=2,gradient=TRUE)
print(summary(img.sobel))
plot(img.sobel,auto.scale=TRUE)
```

Ex. 10. Sobel edge filter. The gaussian blurring is essential to reduce noise.

Edge Detection: Canny Filter

The canny edge filter is a multi-stage algorithm. After denoising, intensity gradients of the image are computed ofr x- and y-direction, then a non-maximum suppression is applied, finally applying a hysteris threhold filtering.

```
use math,imager,plot
img.canny <- cannyEdges(img,t1=0,t2=50,blur=4)
print(summary(img.canny))
plot(img.canny,auto.scale=TRUE)
```

Ex. 11. Canny edge filter. The gaussian blurring is essential to reduce noise. The edge detection thresholds t_1 and t_2 relate to the intensity gradient and must be set carefully.

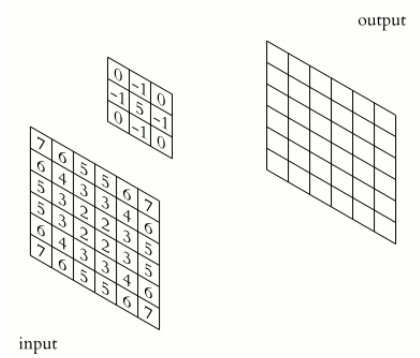
[https://docs.opencv.org/4.x/da/d22/tutorial_py_canny.html]

Kernel-based Convolution Algorithms

Convolution is using a kernel matrix to extract certain features from images.

- A kernel is a matrix, which is shifted across the image and multiplied with the input pixels covered by the kernel matrix such that the output is transformed in a certain desirable manner. Watch this in action below.

[https://towardsdatascience.com/types-of-convolution-kernels-simplified-f040cb307c37]



Geometric Transformations

Simple geometrical operations of entire image or parts of the image are:

1. Translation;
2. Rotation around a specific position;
3. Scaling.

Advanced geometrical operations of entire image:

1. Linear affine transformations (including combinations of simple operations from above)
2. Image warping (using affine transformations)
3. Non-linear transformations for the correction of geometric distortions like Barrel and Pincushin ⇒ **Fisheye Correction**
4. Perspective transformations (perspective warping)

Geometric Distortions

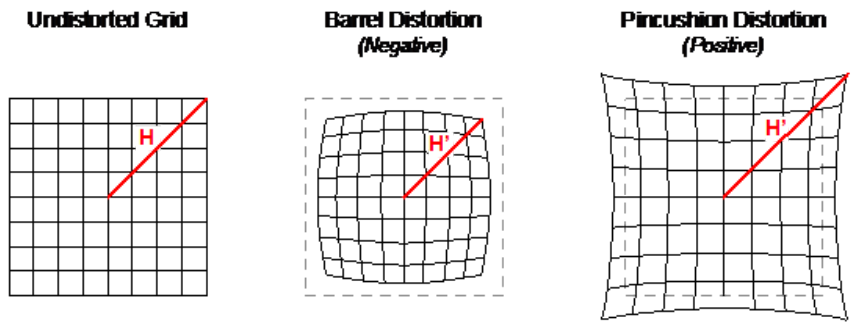


Fig. 6. Local geometric distortions caused by optical imaging (lense distortion)

[https://www.image-engineering.de/library/image-quality/factors/1062-distortion]

Measurement error and confidence

Systematic deviation (systematic error)

- Deviation is caused by the sensor, environment, and sometimes physical processes
- E.g.: incorrect calibration, constantly existing faults such as friction
- Can only be eliminated by carefully examining the source of the error

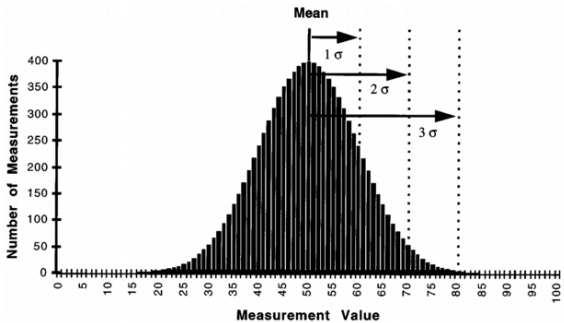
Random deviation (Random or statistical error)

- Deviation is caused by unavoidable, irregular disturbances
- with repeated measurement, individual results differ from each other
- Individual results vary by an average value

Measurement error and confidence

Random error scattering

- Random errors affect the accuracy of a measurement (noise).
- Noise affects input and target feature computation (ML output)!
- If one repeats a measurement of a quantity X which is falsified by pure random errors, the frequency distribution of the measured values is $S = \{s_1, s_2,...,s_n\}$ by a mean value $\bar{S}$ given by a Gaussian distribution (the number of measurements N must be large).



[9]

Fig. 7. Frequency distribution according to Gauss of measured values centered around an average value

Examples: Statistical Analysis

WorkBook Live

CLEAR LOAD + - Plot Analysis

Summary

- Data can be classified into:
  - Categorical variables and values
  - Metric variables and values
  - Temporal static variables
  - Temporal dynamic variables (time series)
- All sensor variables are subject to measurement errors:
  - Noise
  - Distortion
  - Displacement (bias)
  - Problem of reproducibility and systematic errors (environment!)
- A (statistical) data analysis is often the first step in the ML workflow

Summary

- There are different levels of sensor data features
  - Aggregates like statstical measures
  - Time- and freqency domain features
  - Spatial features like edges in images or geometric properties
  - Region-of-Interest Markinh
  - Semantic features, i.e., classified features like damages

The signal feature selection and extraction is the first step to compute and detect target features like damages using data-driven models.