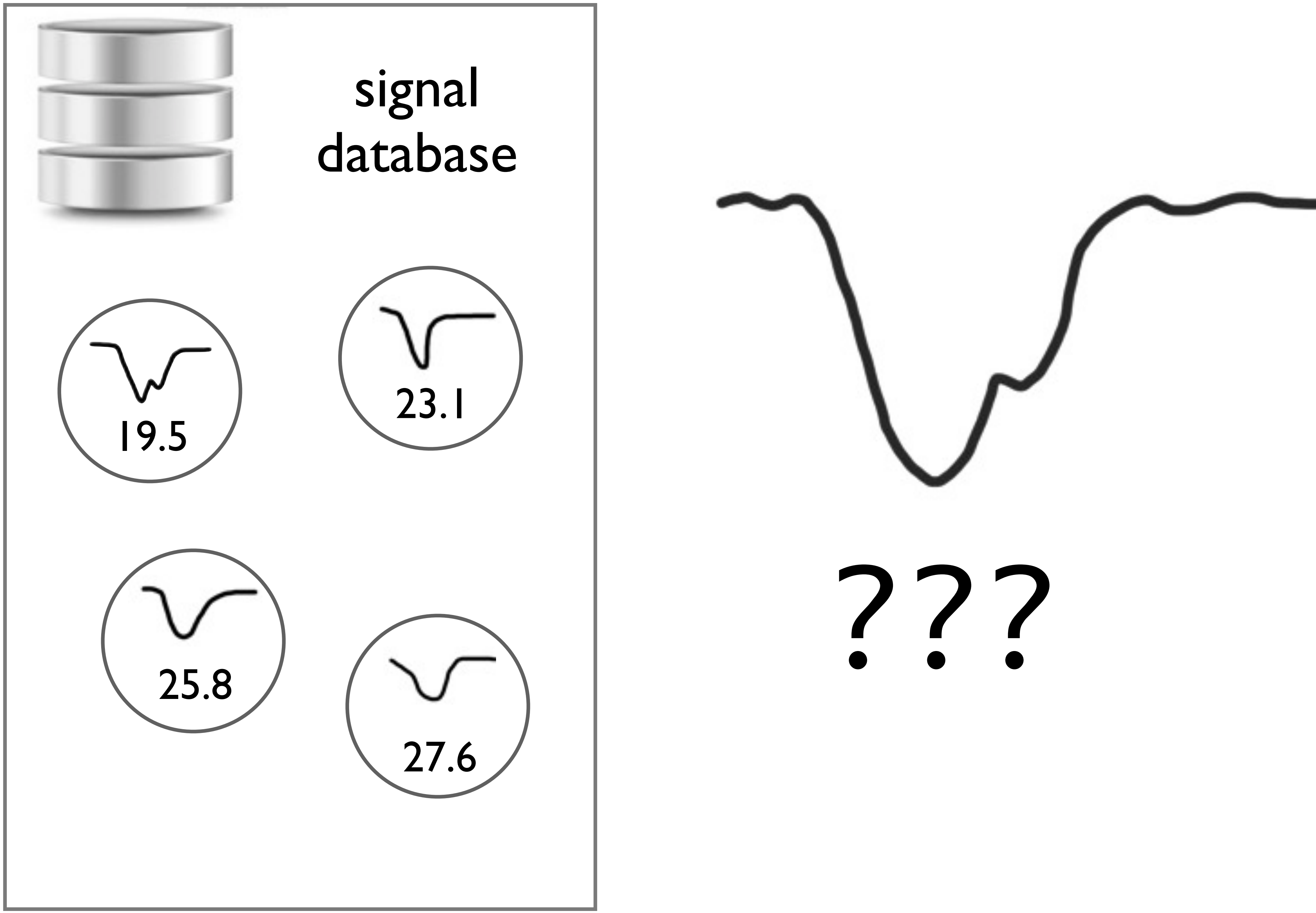
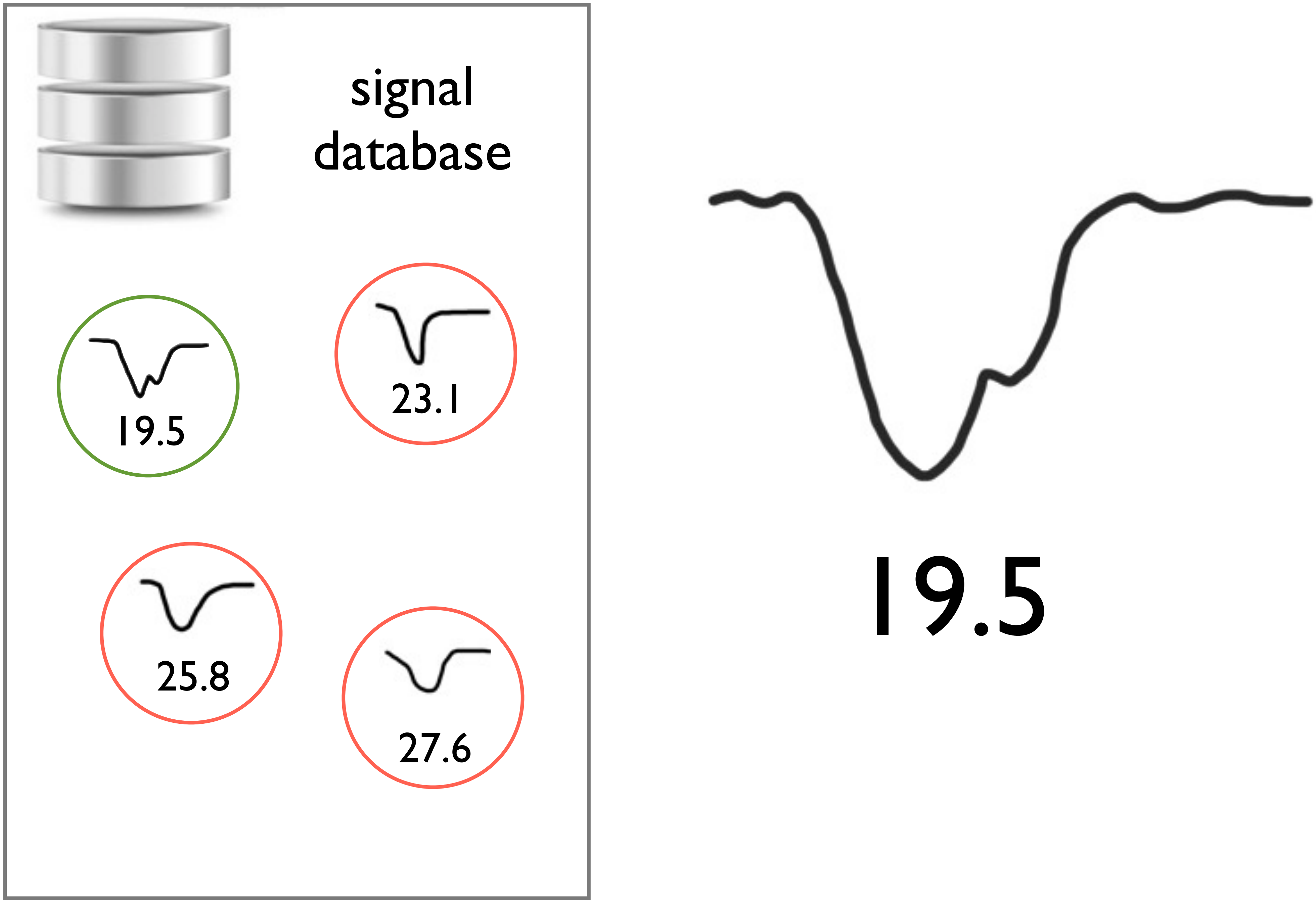
Hit position reconstruction using Hausdorff metric

Natalia Zoń


signal
database
19.5
23.1
25.8
27.6
???



signal
database
19.5
23.1
25.8
27.6
19.5

Assets:

signal database

- values of measured voltage at discrete time nodes
- the position of gamma quants hit on the scintillator

Tools:

C++ language, STL

QT + ROOT QTCreator (IDE)

GNU Scientific Library

Boost C++ Libraries

Program modules

- database services
- simulation of signals
- signal reconstruction
- signal comparison

Database services

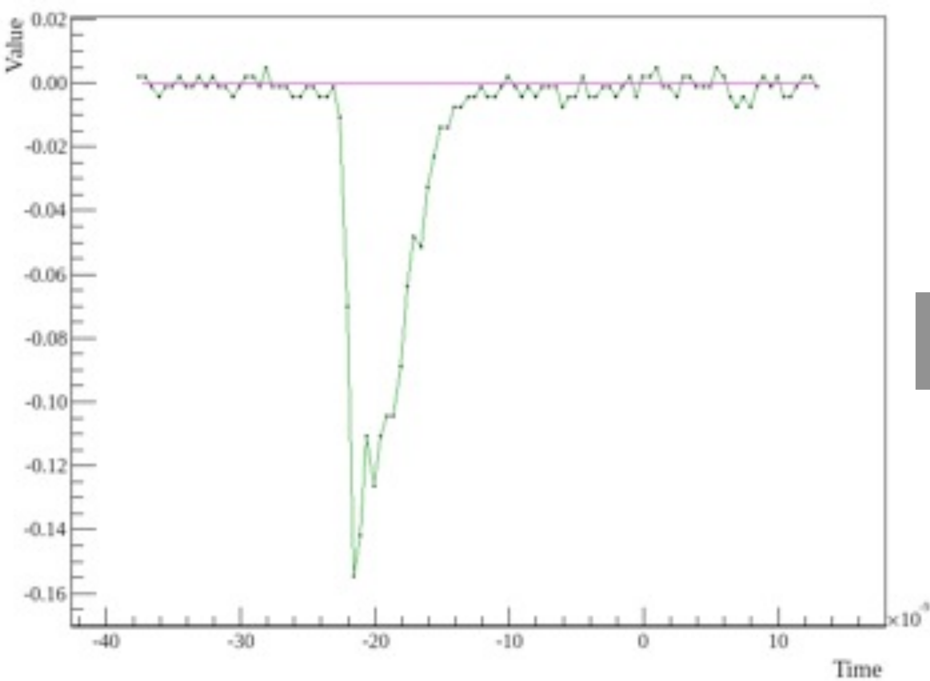
C++

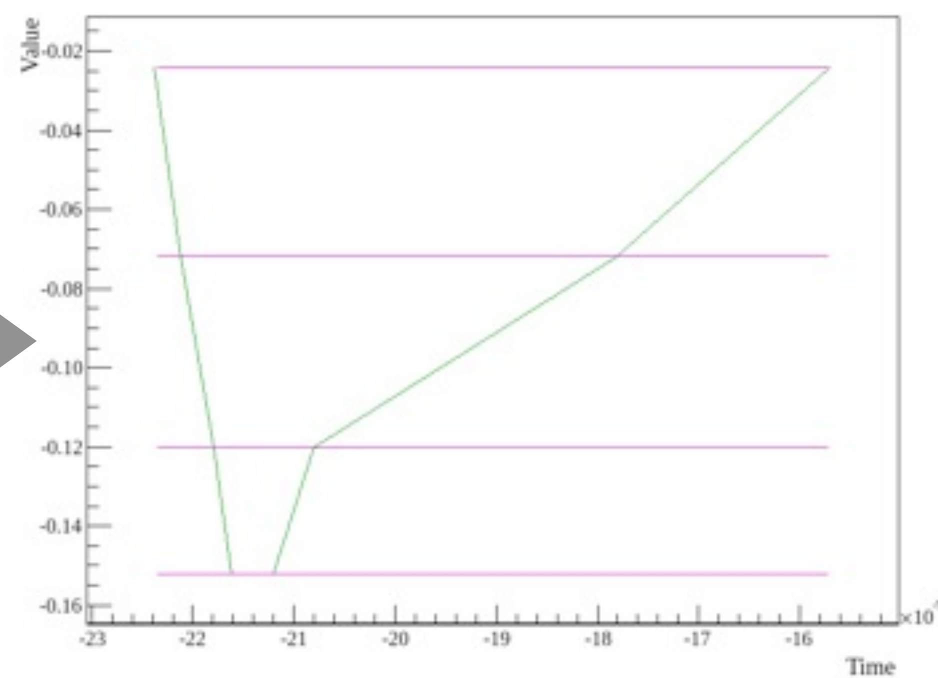
currently the implementation is based on C++ STL (Standard Template Library)

ROOT
An Object-Oriented Data Analysis Framework

plans for the future: to integrate a database solution based on the TTree structure from ROOT
(dr Marcin Zieliński)

Simulation of signals





an example of a full signal from the database

an example of a simulated signal

The steps of simulation:

The steps of simulation:

1. Sampling

The steps of simulation:

1. Sampling
    - a given signal from the DB is sampled for time at four chosen fractions of amplitude

The steps of simulation:

1. Sampling
   - a given signal from the DB is sampled for time at four chosen fractions of amplitude
   - if the desired time is in between nodes of the signal, the neighbouring area is interpolated

The steps of simulation:

1. Sampling
   - a given signal from the DB is sampled for time at four chosen fractions of amplitude
   - if the desired time is in between nodes of the signal, the neighbouring area is interpolated

2. Smearing

The steps of simulation:

1. Sampling
    - a given signal from the DB is sampled for time at four chosen fractions of amplitude
    - if the desired time is in between nodes of the signal, the neighbouring area is interpolated

2. Smearing
    - randomness is added to the times obtained in the process of sampling (Gaussian distribution)

The steps of simulation:

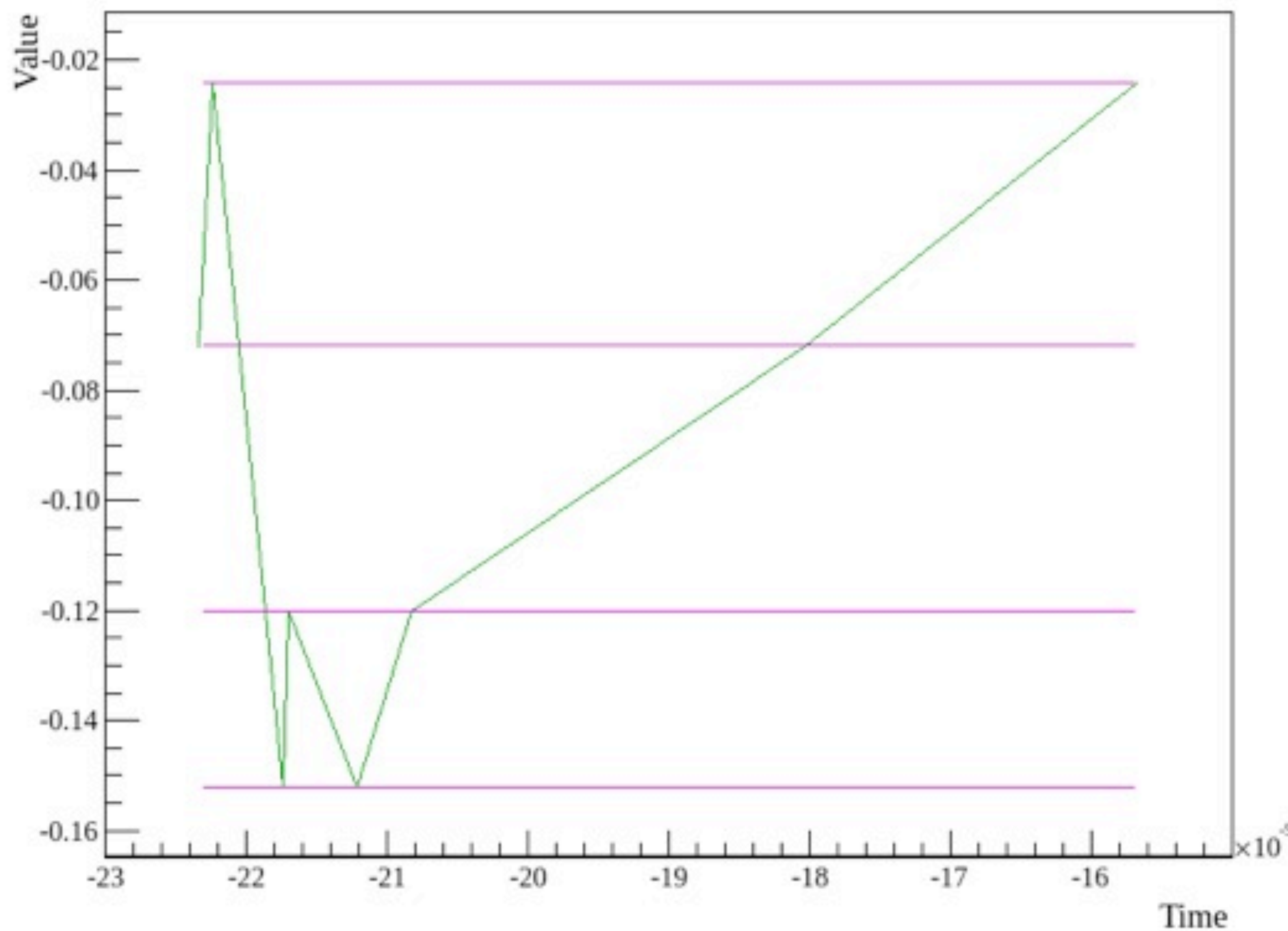
1. Sampling
   - a given signal from the DB is sampled for time at four chosen fractions of amplitude
   - if the desired time is in between nodes of the signal, the neighbouring area is interpolated

2. Smearing
   - randomness is added to the times obtained in the process of sampling (Gaussian distribution)
   - the implementation is based on `<tr1/random>` (STL)

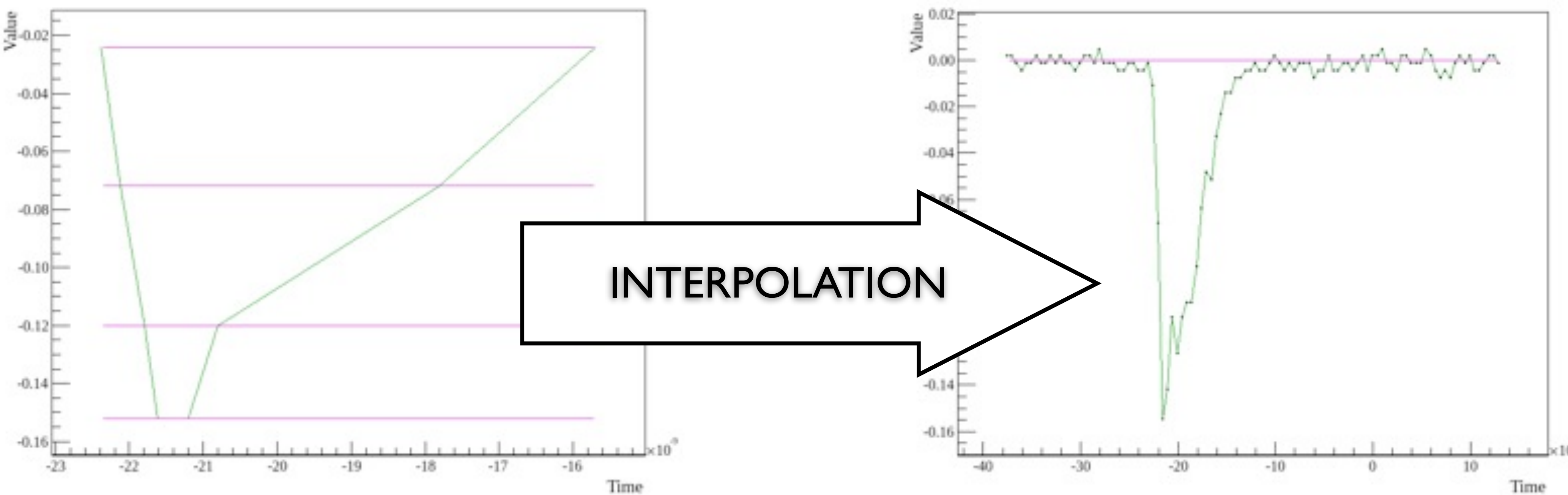
Limitations:

the standard deviation has to be less than:

$$\sigma \approx 10^{-11}$$



Signal reconstruction



Methods of interpolation used in the program:

- natural cubic spline
- periodic cubic spline
- Akima spline

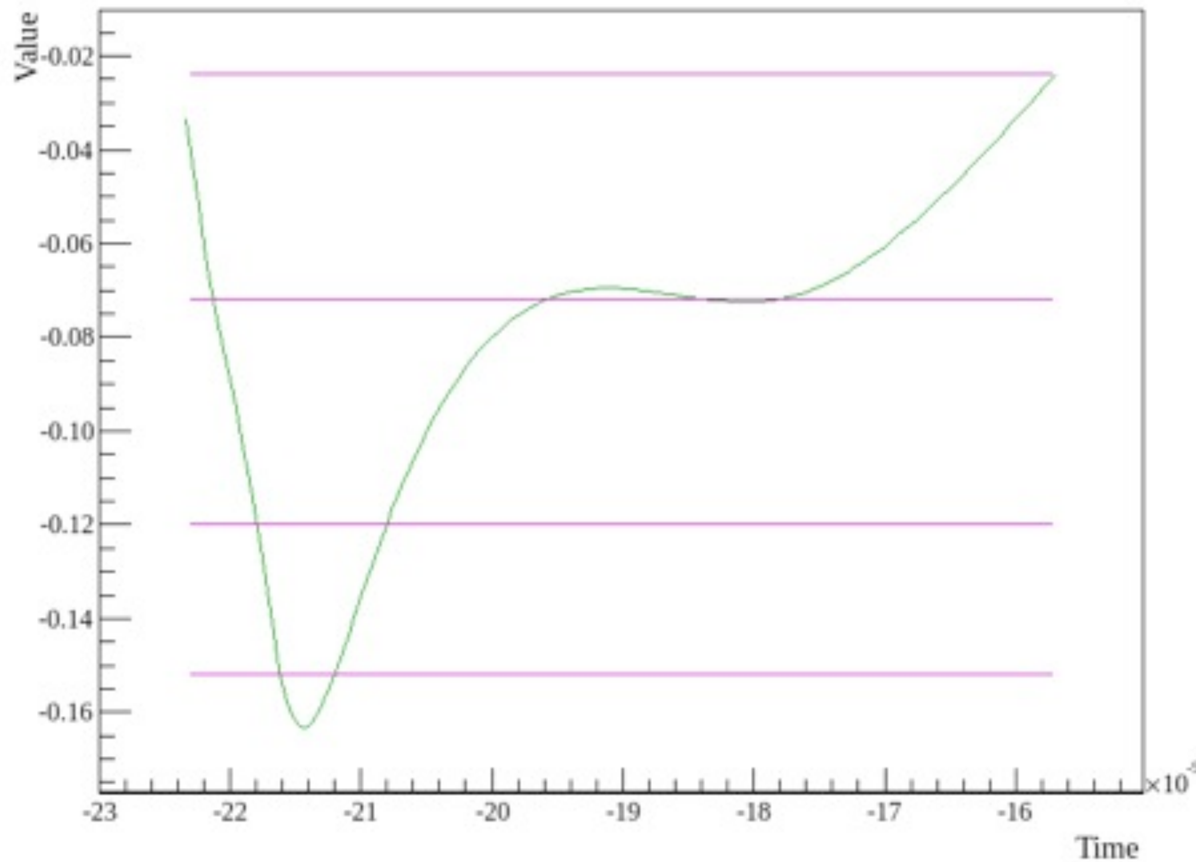
GNU Scientific Library (GSL):
<gsl_interp.h>, <gsl_spline.h>

Natural cubic splines

Let K be a set of nodes:

$$K = \{x_0, x_1, ..., x_m\}$$

The function $s \in C^2[x_0, x_m]$ is a cubic spline if s is a cubic polynomial s_i in every interval $[x_i, x_{i+1}]$



natural cubic spline example

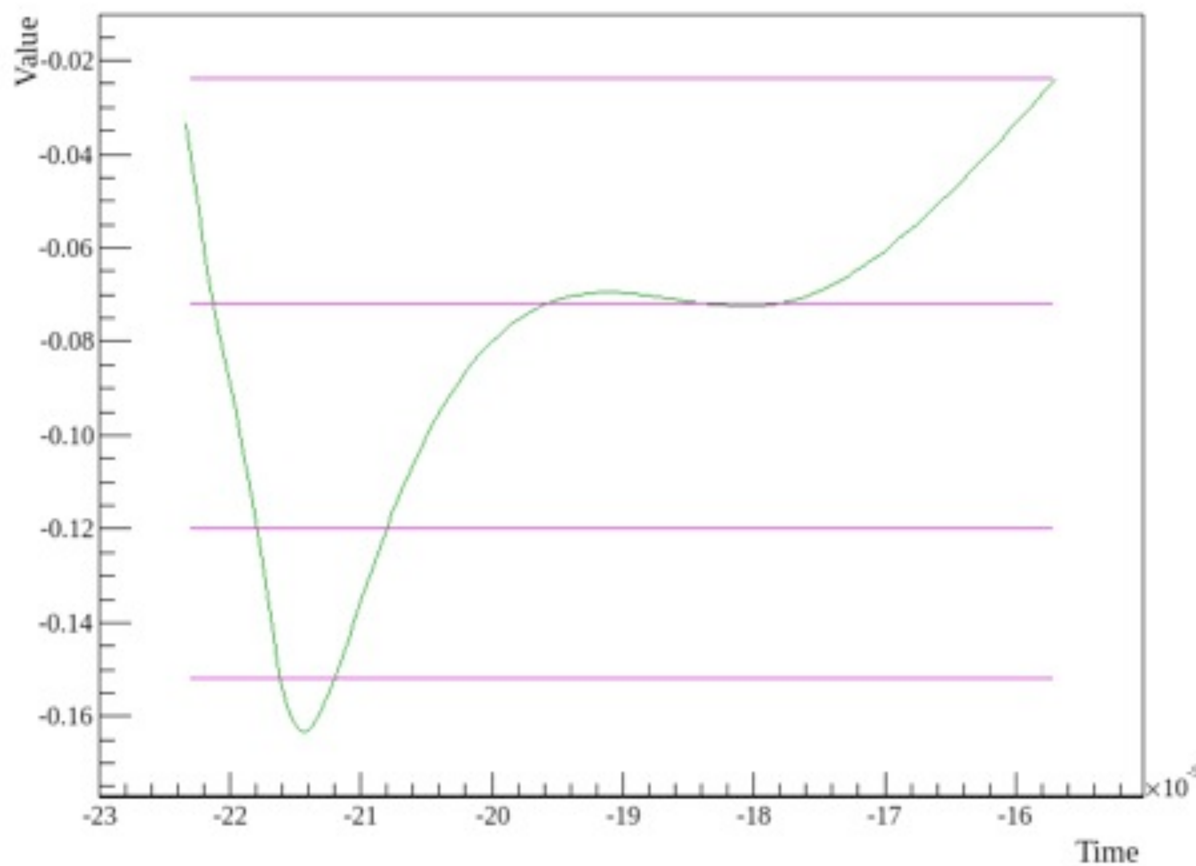
Natural cubic splines

Characteristics at nodes:

$$s_i(x_{i+1}) = s_{i+1}(x_{i+1})$$
$$s_i'(x_{i+1}) = s_{i+1}'(x_{i+1})$$
$$s_i''(x_{i+1}) = s_{i+1}''(x_{i+1})$$

Boundary condition:

$$s_0''(x_0) = 0$$
$$s_{m-1}''(x_m) = 0$$

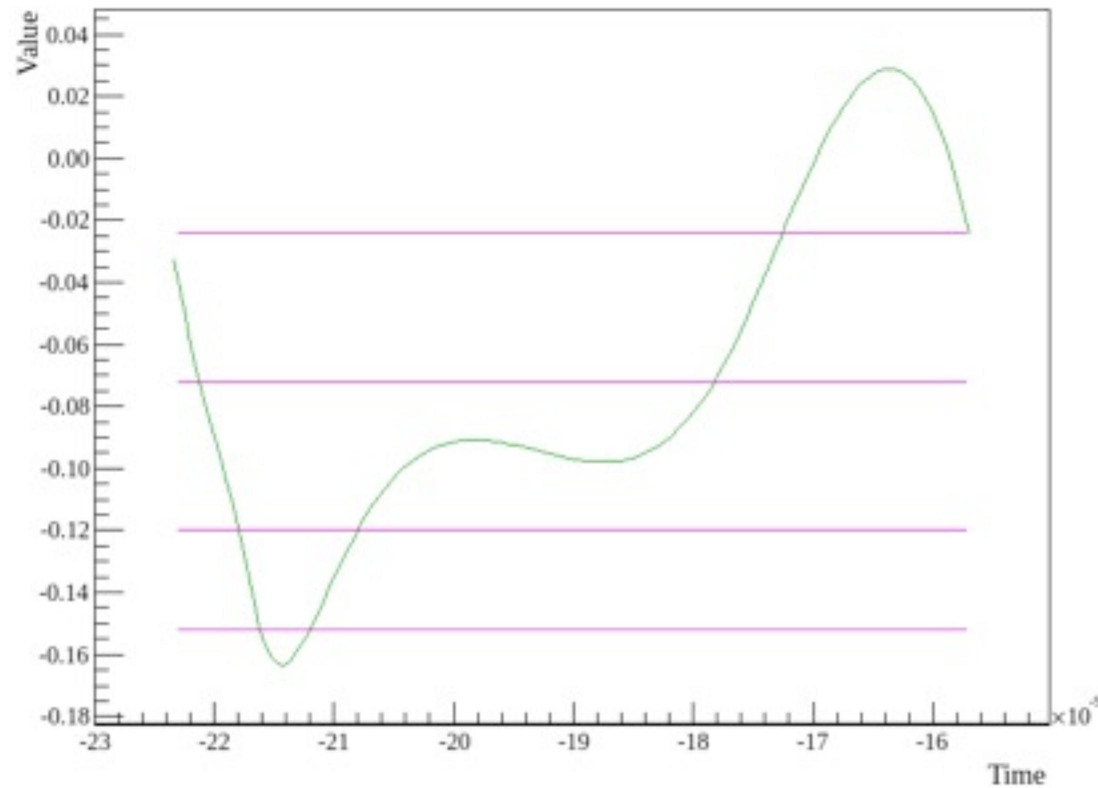


natural cubic spline example

Periodic cubic splines

Boundary condition:

$$s_0'(x_0) = s_{m-1}'(x_m)$$
$$s_0''(x_0) = s_{m-1}''(x_m)$$



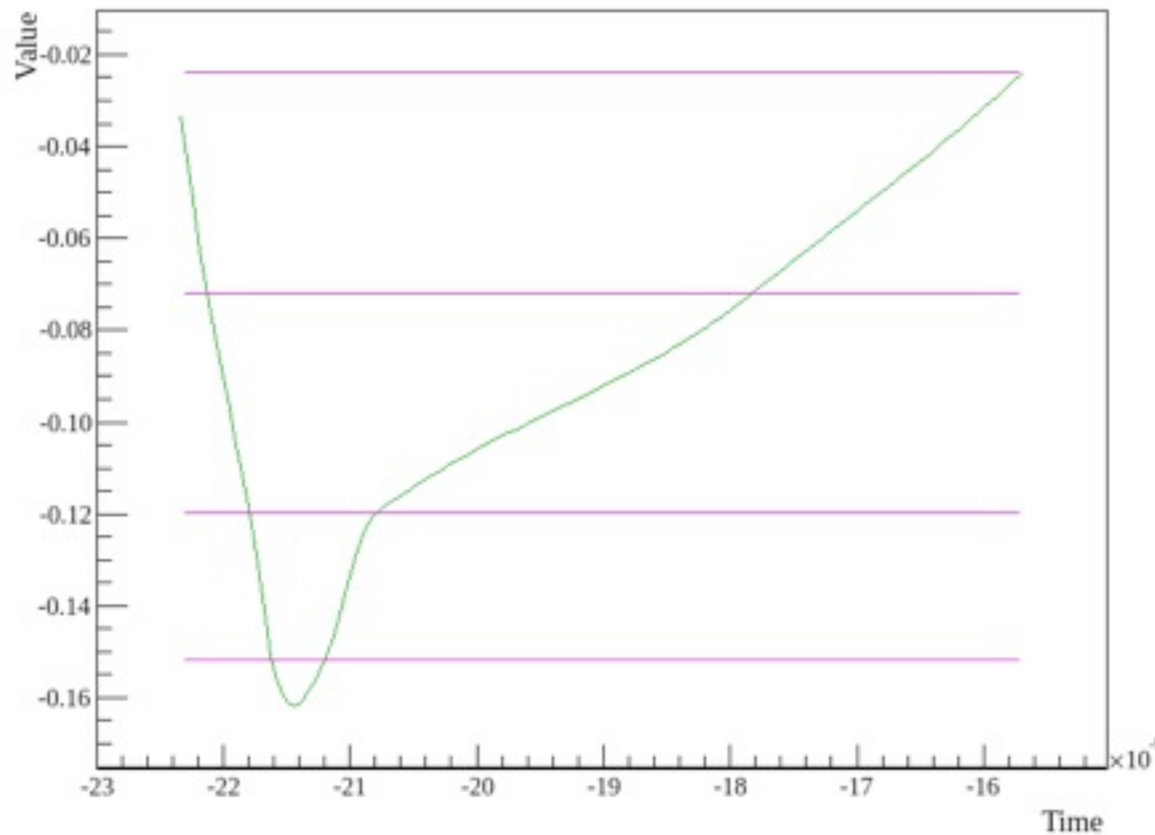
periodic cubic spline example

Akima splines

Akima spline will never produce oscillations.

Locality:
values of function f
on the interval $[x_i, x_{i+1}]$

are dependant only
on its neighbourhood
$f_{i-2}, f_{i-1}, f_i, f_{i+1}, f_{i+2}, f_{i+3}$



Akima spline example

Signal comparison

Methods of signal comparision implemented so far:

- Least Squares based method
- Comparing times at thresholds
- Comparing integrals
- Hausdorff metric based method

Least squares based method

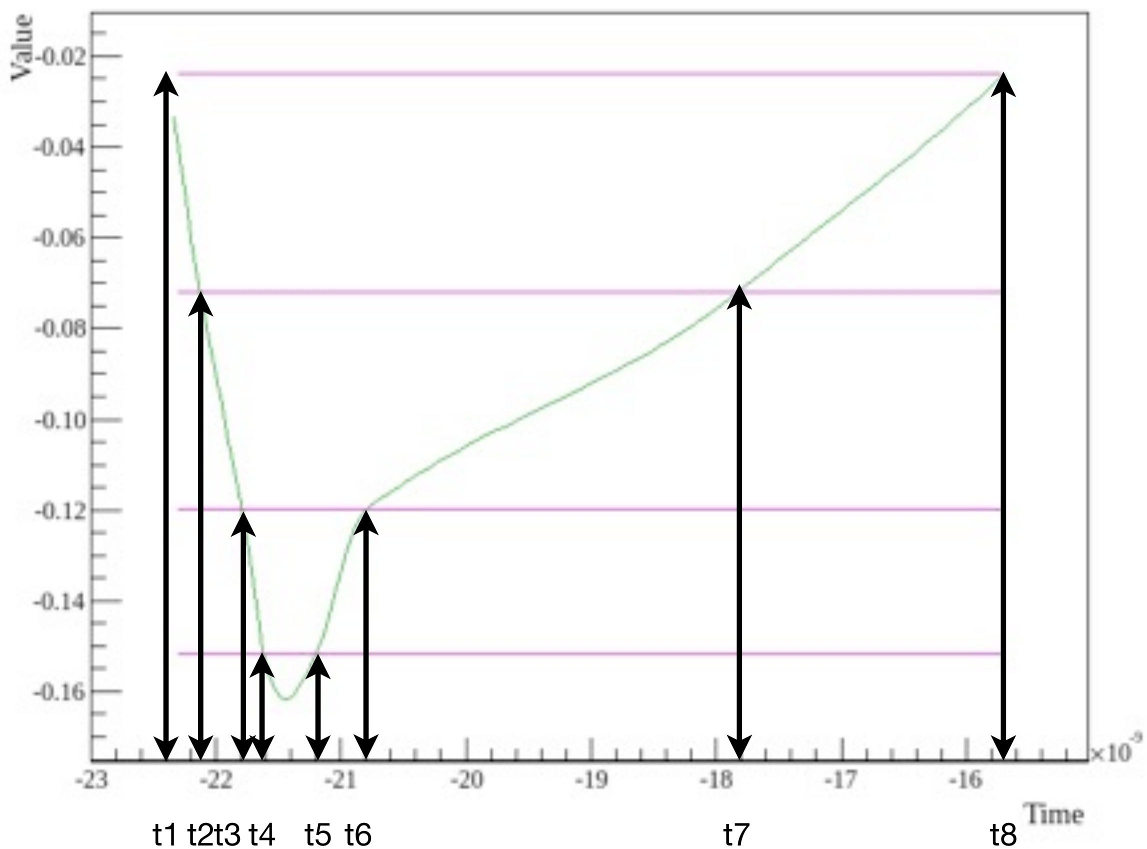
$$\chi^2 = \sum \frac{(y_i - Y(x_i))^2}{\sigma_i^2}$$

(x_i, y_i) a point from the first function

$Y(x)$ the value of the second function

σ^2 variance

Comparing times at thresholds



$$\sum_{i=1}^{8} |t_i - t'_i|$$

Comparing integrals

- uses the numerical integration algorithm from GSL: gsl_integration_qags
- this is an implementation of the Gauss-Kronrod Quadrature numerical integration method

- the comparition result is the absolute value of the difference between the values of integrals for both signals (on the same interval)

$$\left| \int_{t_0}^{t_n} Signal A - \int_{t_0}^{t_n} Signal B \right|$$

Comparing integrals - easy to optimize

- precompute and store the integrals for all of the signals in the database

- during runtime, compute only those integrals that hadn't been precomputed - that means computing only one integral for every test

- comparing signals is now as fast as subtracting two floating point numbers

Hausdorff metric

(first introduced by F. Hausdorff in 1914)

Given two finite point sets A and B:

$$A=\{a_{1,} \ldots, a_p\}$$
$$B=\{b_{1,} \ldots, b_p\}$$

the Hausdorff distance is defined as

$$H(A,B)=max(h(A,B),h(B,A))$$

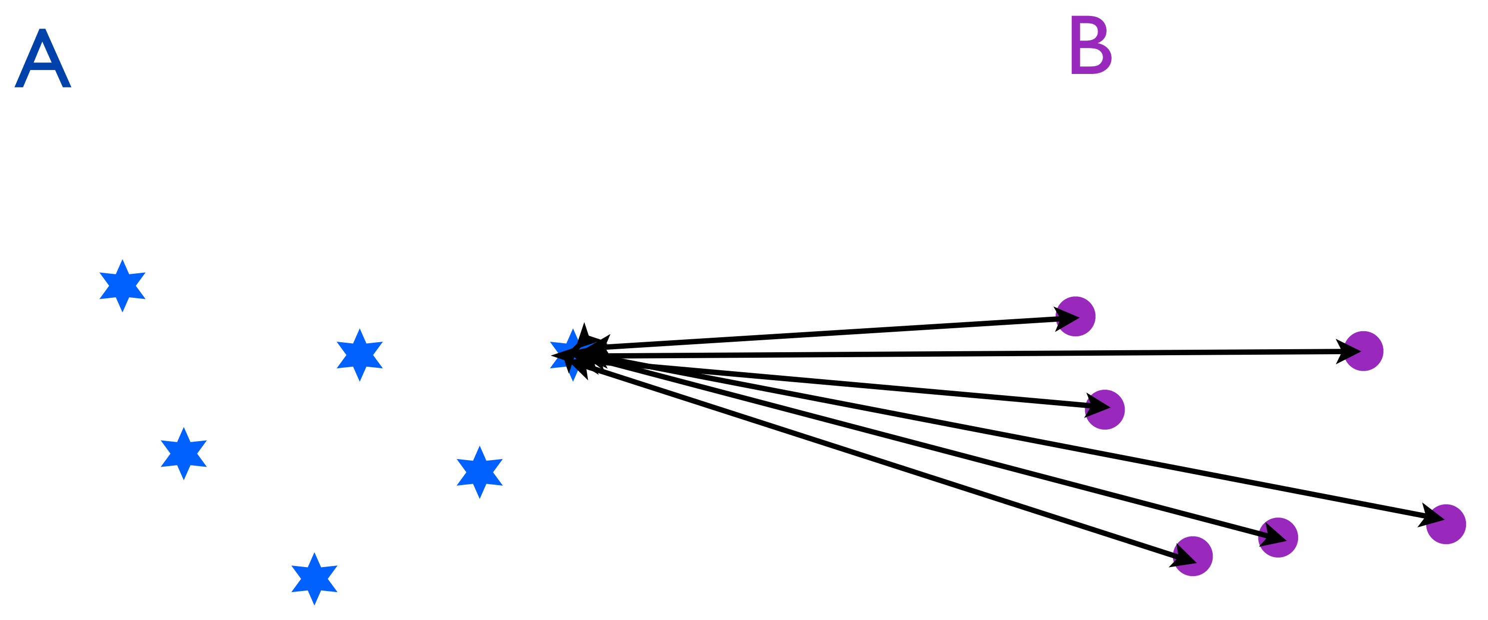
where h(A,B) is called the <u>directed</u> Hausdorff distance and defined as:

$$h(A,B)=\max_{a\in A}\min_{b\in B}||a-b||$$

and ||a - b|| is the Euclidean norm:

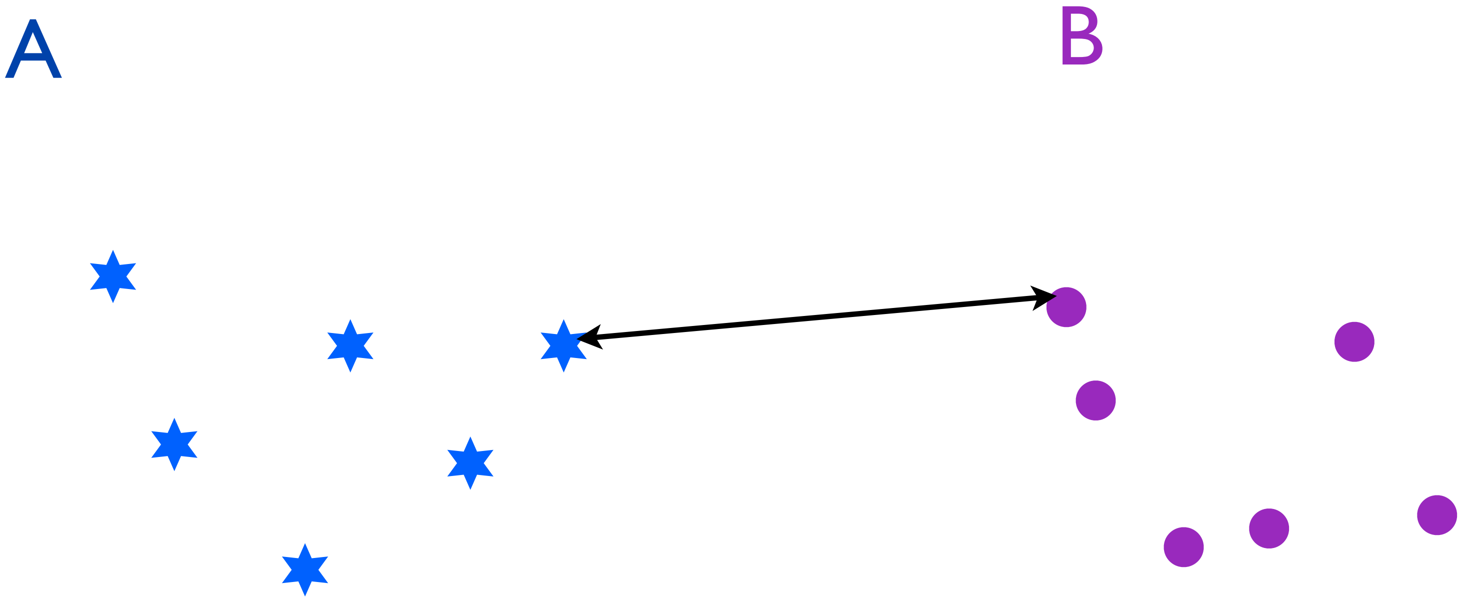
$$||a-b||=\sqrt{((a_x-b_x)^2+(a_y-b_y)^2)}$$

Hausdorff metric example



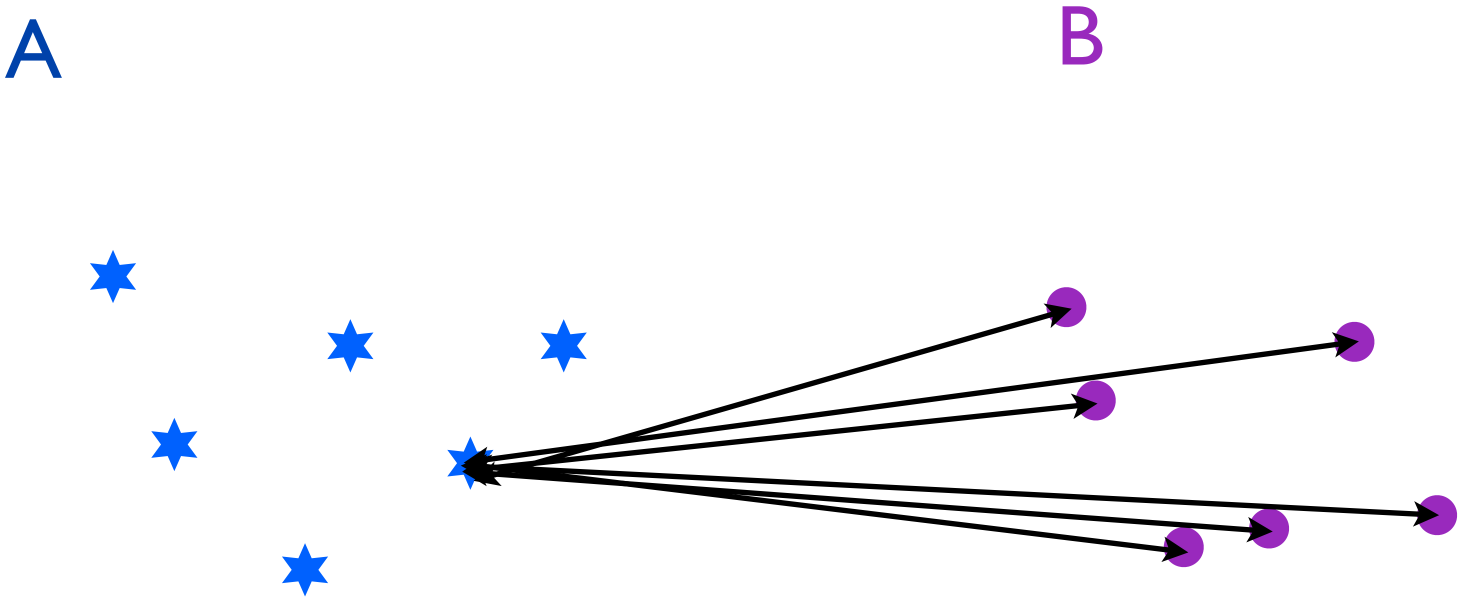
h(A,B) identifies the point a in A that is farthest from any point of B and measures the distance from a to its nearest neighbour in B

Hausdorff metric example



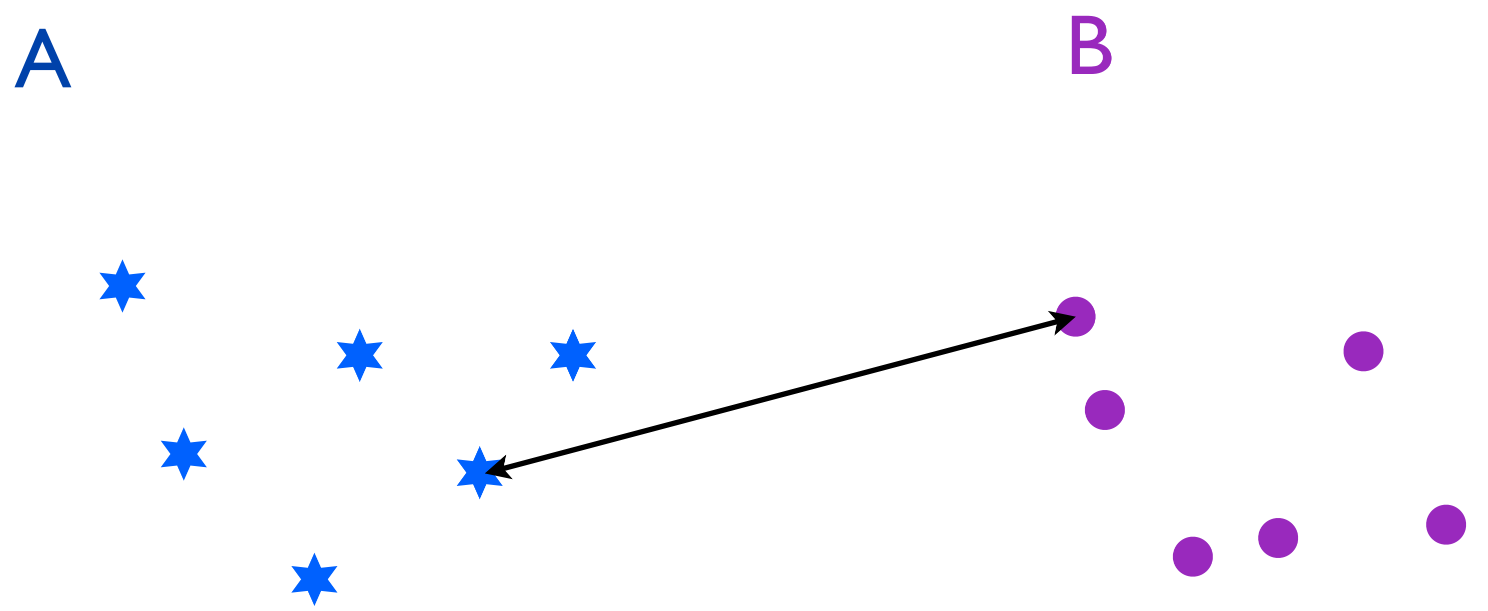
h(A,B) identifies the point a in A that is farthest from any point of B and measures the distance from a to its nearest neighbour in B

Hausdorff metric example



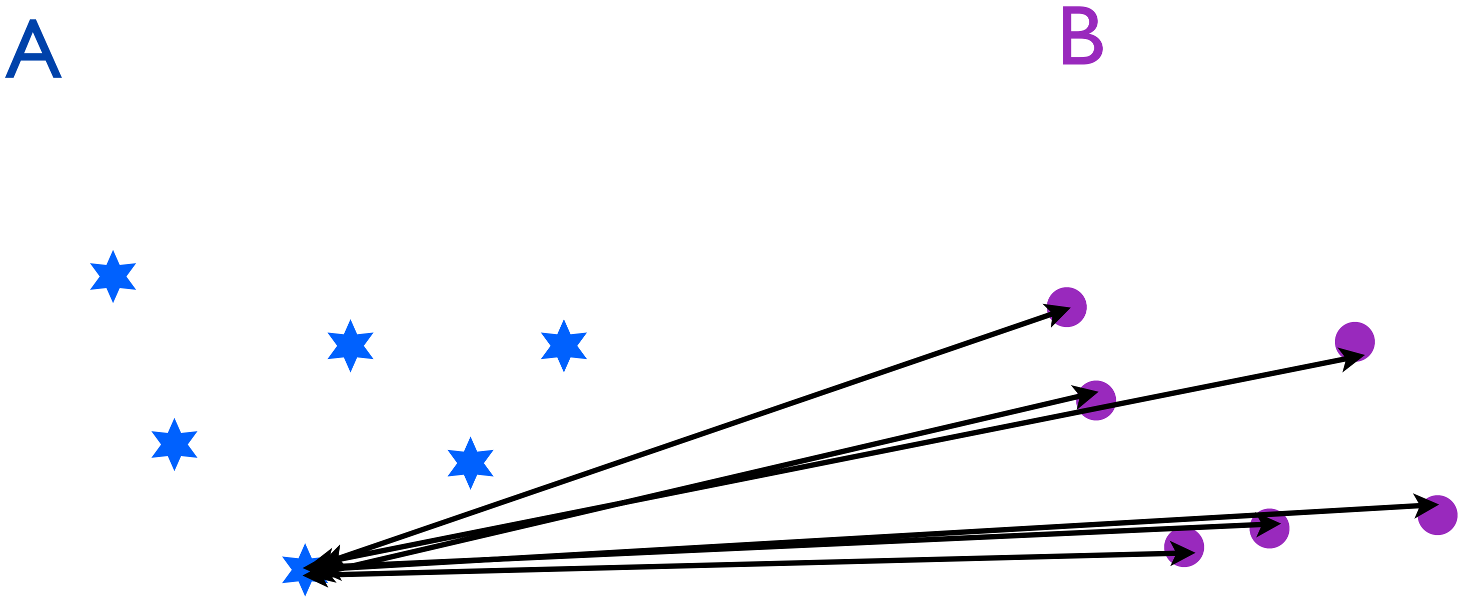
h(A,B) identifies the point a in A that is farthest from any point of B and measures the distance from a to its nearest neighbour in B

Hausdorff metric example



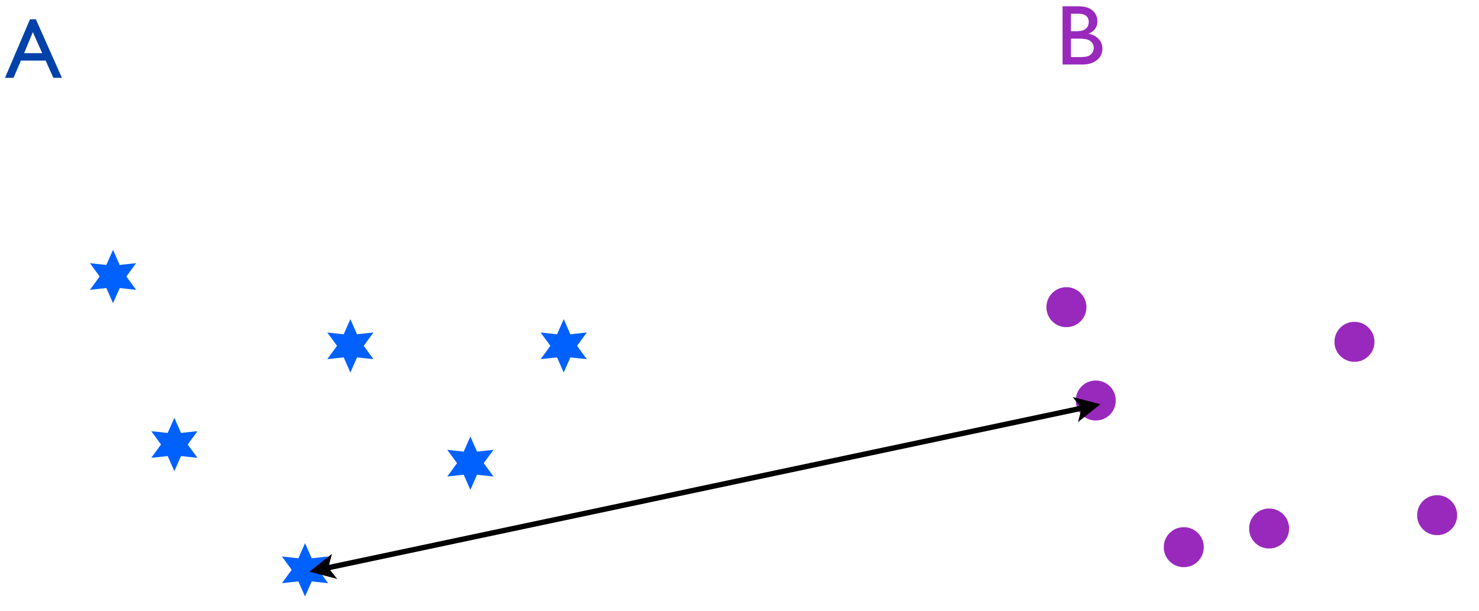
h(A,B) identifies the point a in A that is farthest from any point of B and measures the distance from a to its nearest neighbour in B

Hausdorff metric example



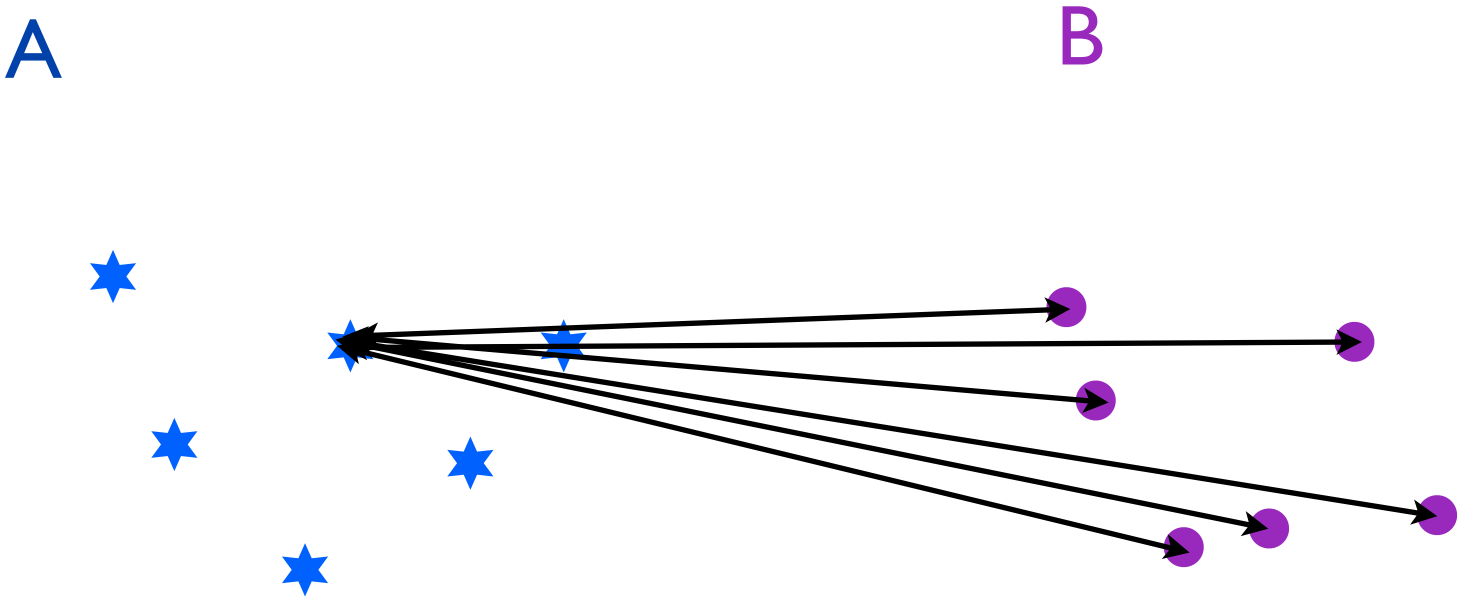
h(A,B) identifies the point a in A that is farthest from any point of B and measures the distance from a to its nearest neighbour in B

Hausdorff metric example



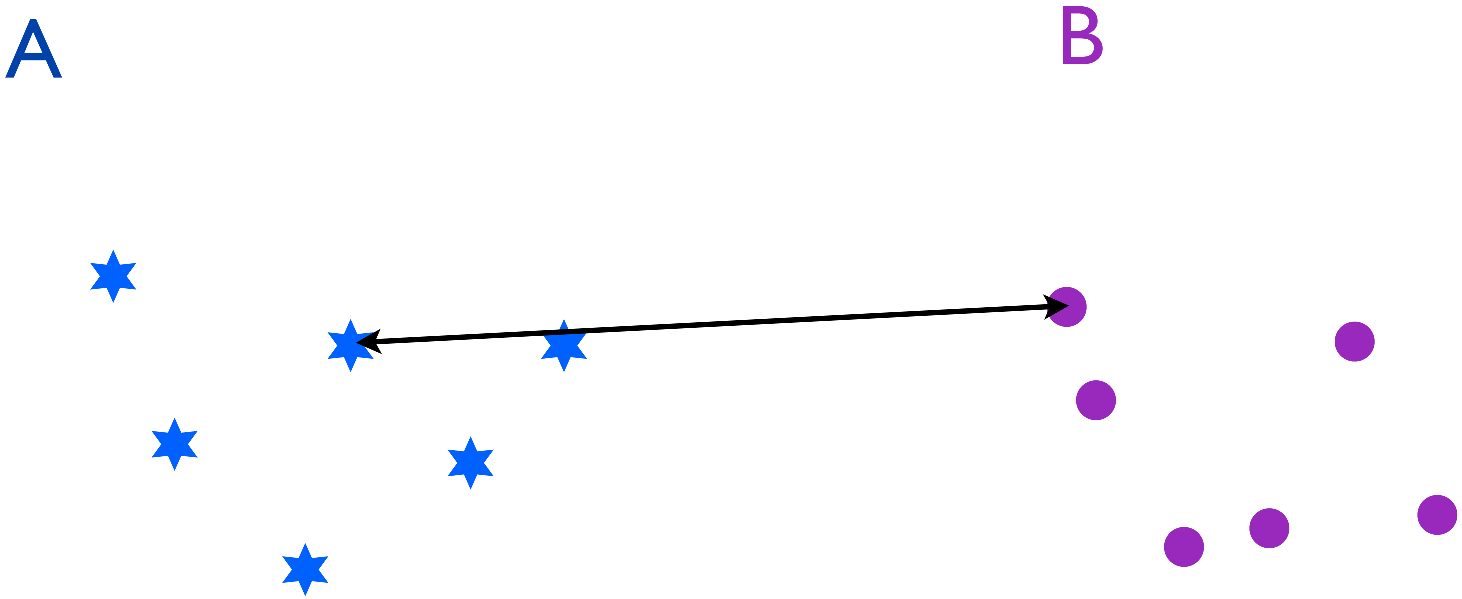
h(A,B) identifies the point a in A that is farthest from any point of B and measures the distance from a to its nearest neighbour in B

Hausdorff metric example



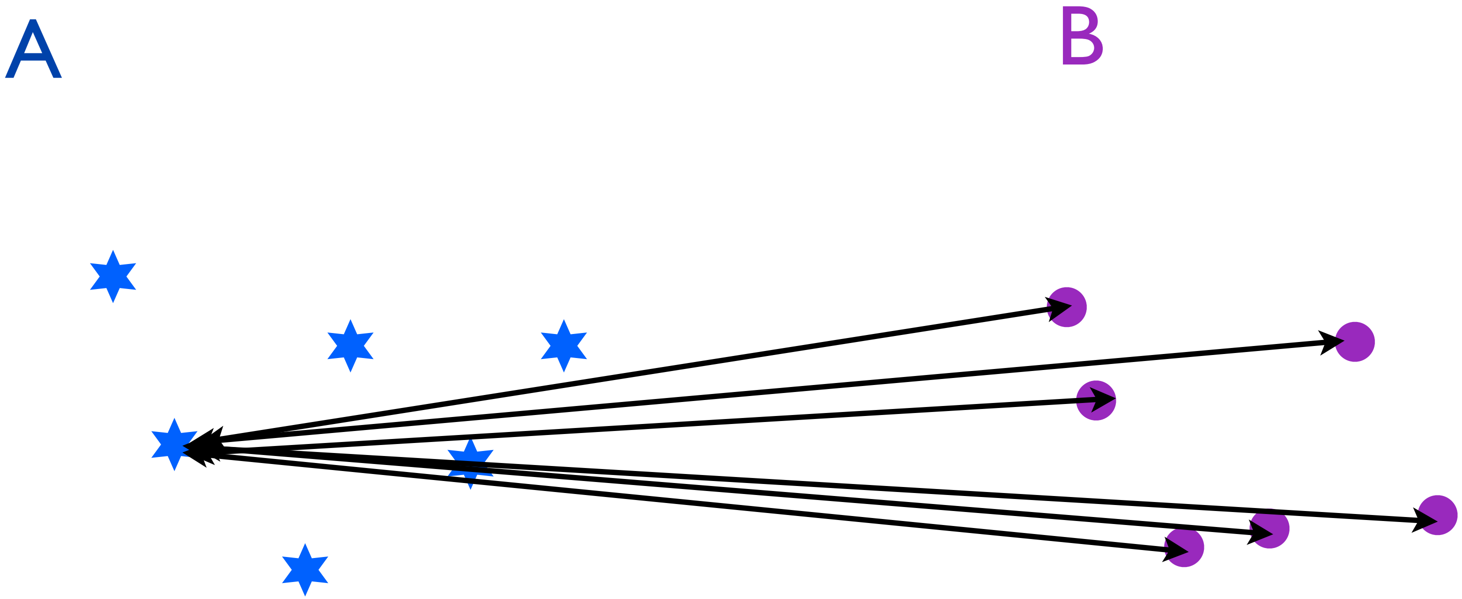
h(A,B) identifies the point a in A that is farthest from any point of B and measures the distance from a to its nearest neighbour in B

Hausdorff metric example



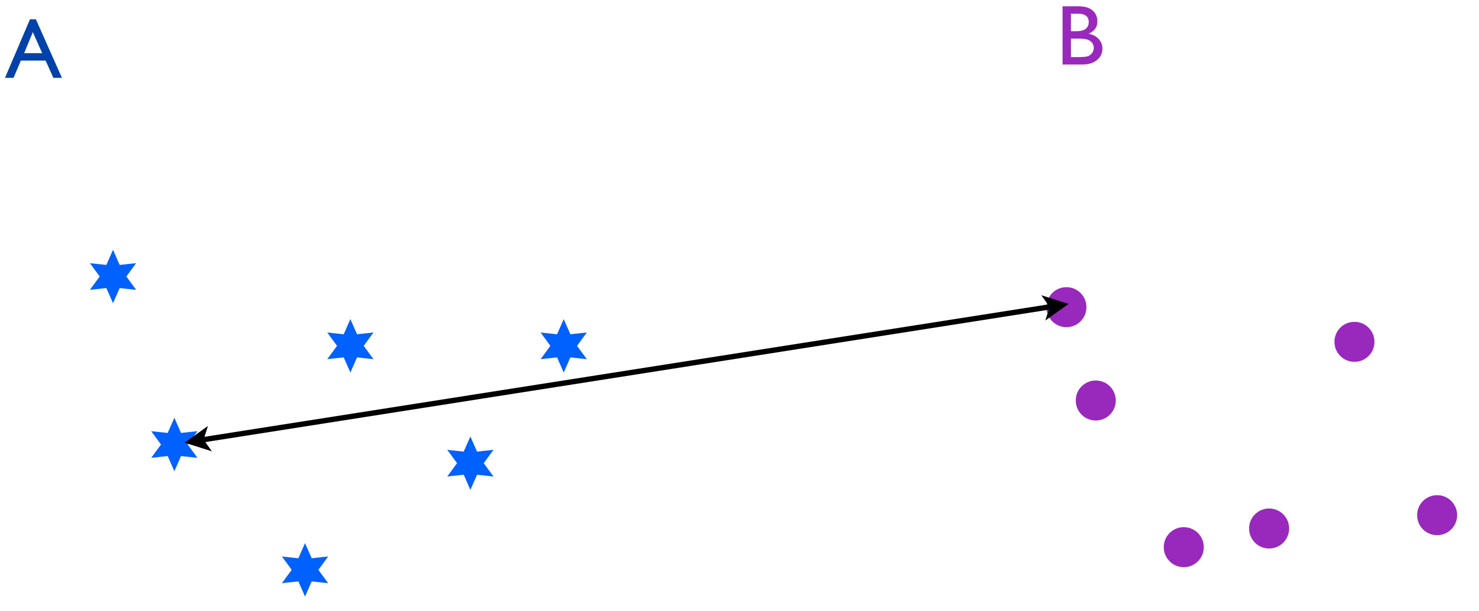
h(A,B) identifies the point a in A that is farthest from any point of B and measures the distance from a to its nearest neighbour in B

Hausdorff metric example



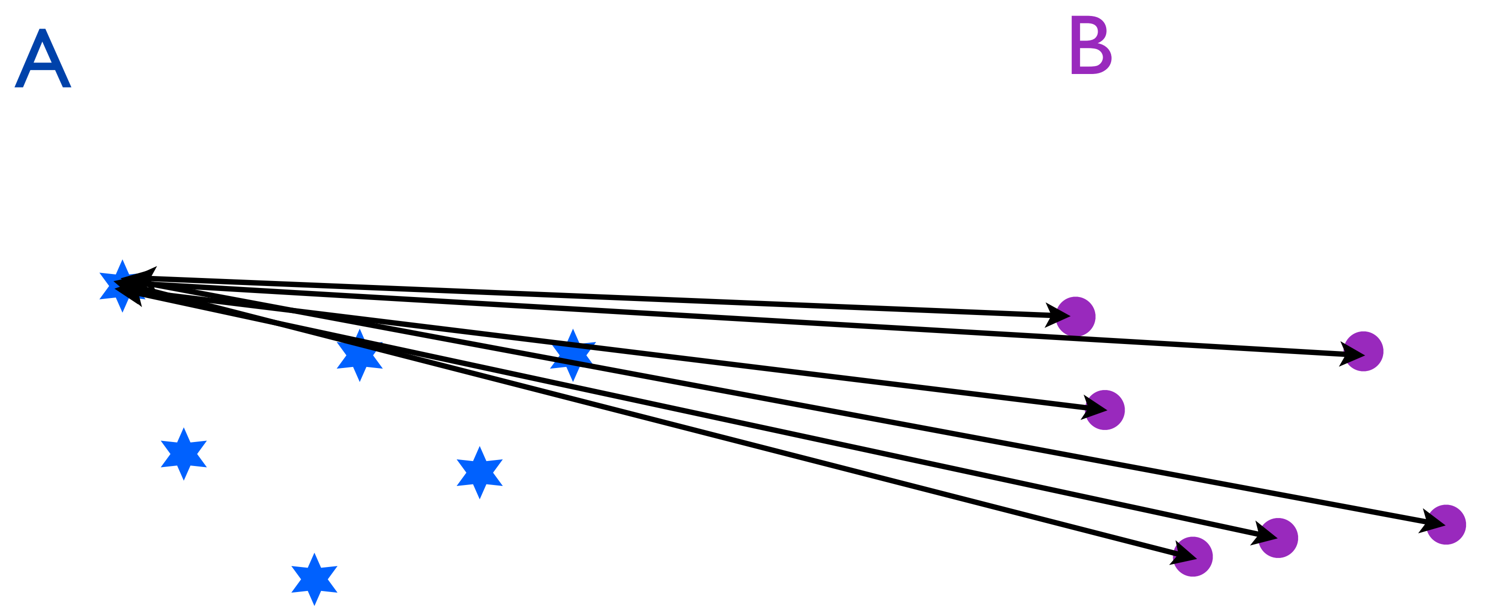
h(A,B) identifies the point a in A that is farthest from any point of B and measures the distance from a to its nearest neighbour in B

Hausdorff metric example



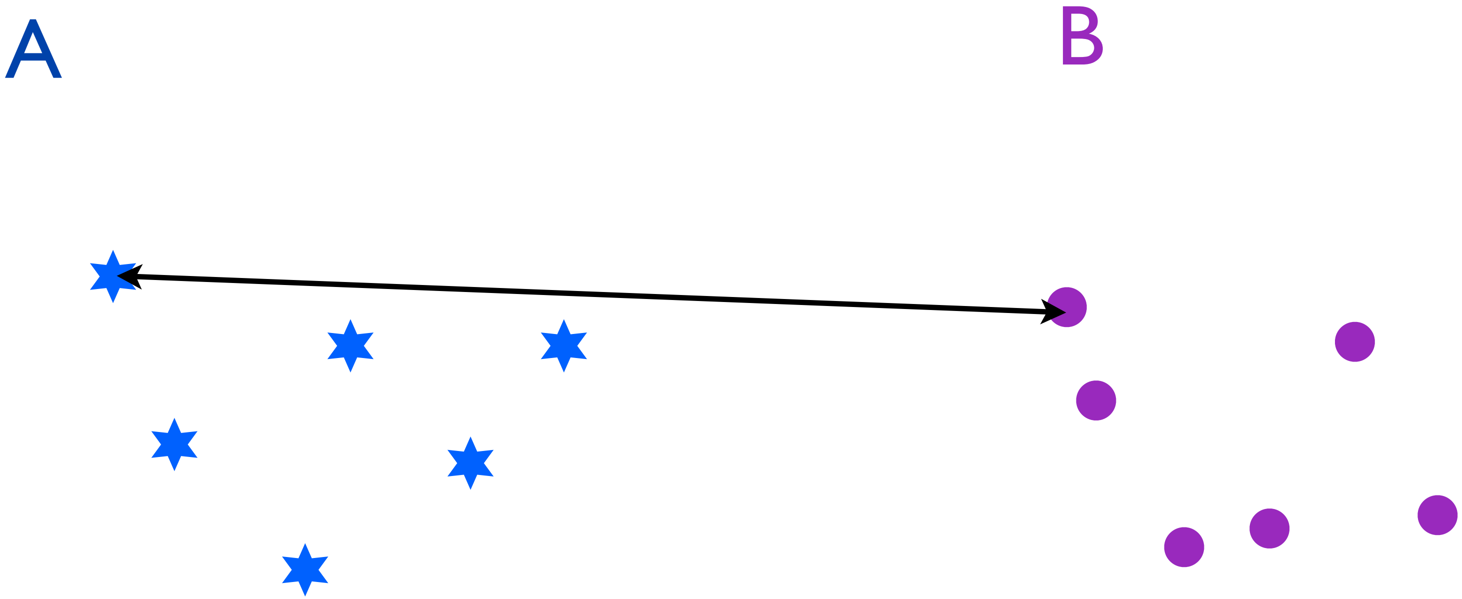
h(A,B) identifies the point a in A that is farthest from any point of B and measures the distance from a to its nearest neighbour in B

Hausdorff metric example



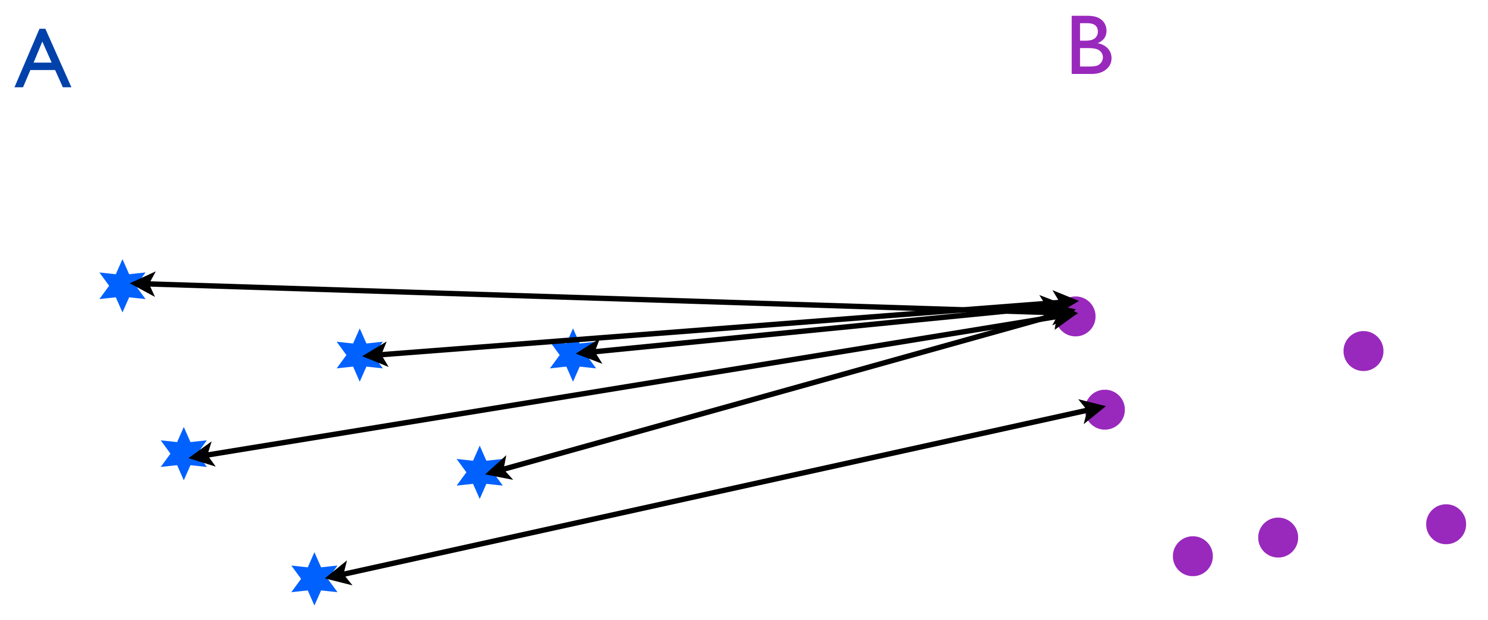
h(A,B) identifies the point a in A that is farthest from any point of B and measures the distance from a to its nearest neighbour in B

Hausdorff metric example



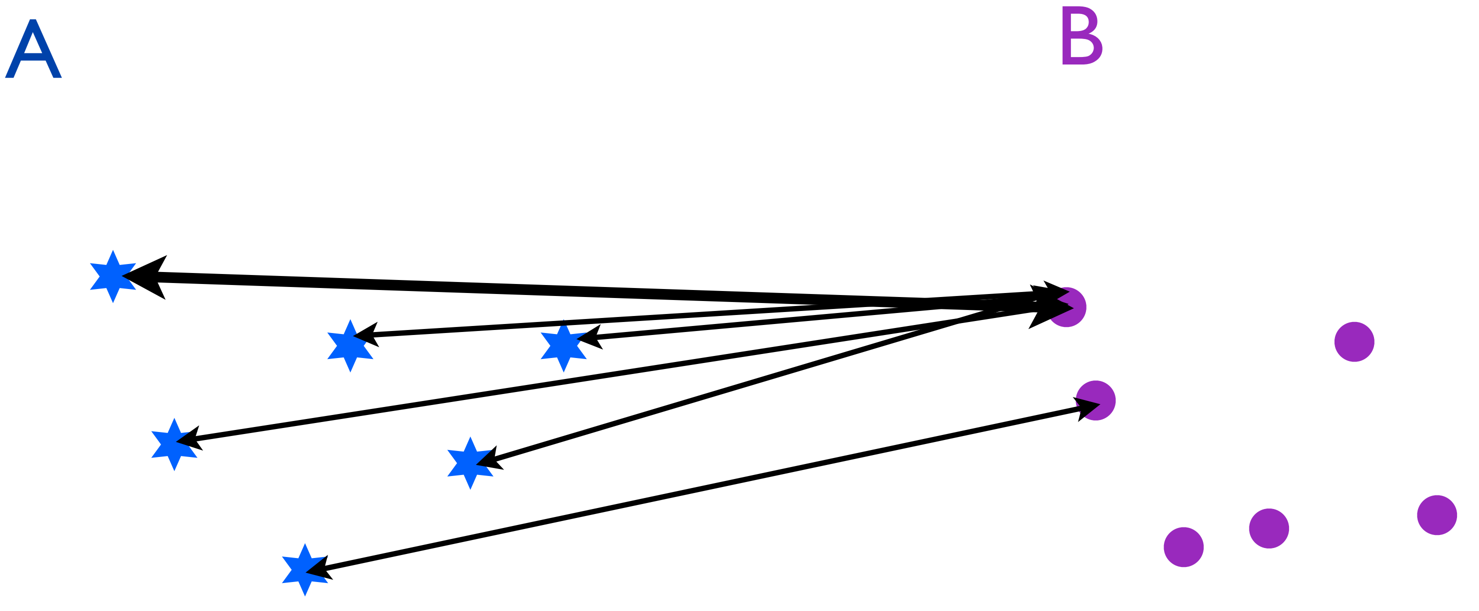
h(A,B) identifies the point a in A that is farthest from any point of B and measures the distance from a to its nearest neighbour in B

Hausdorff metric example



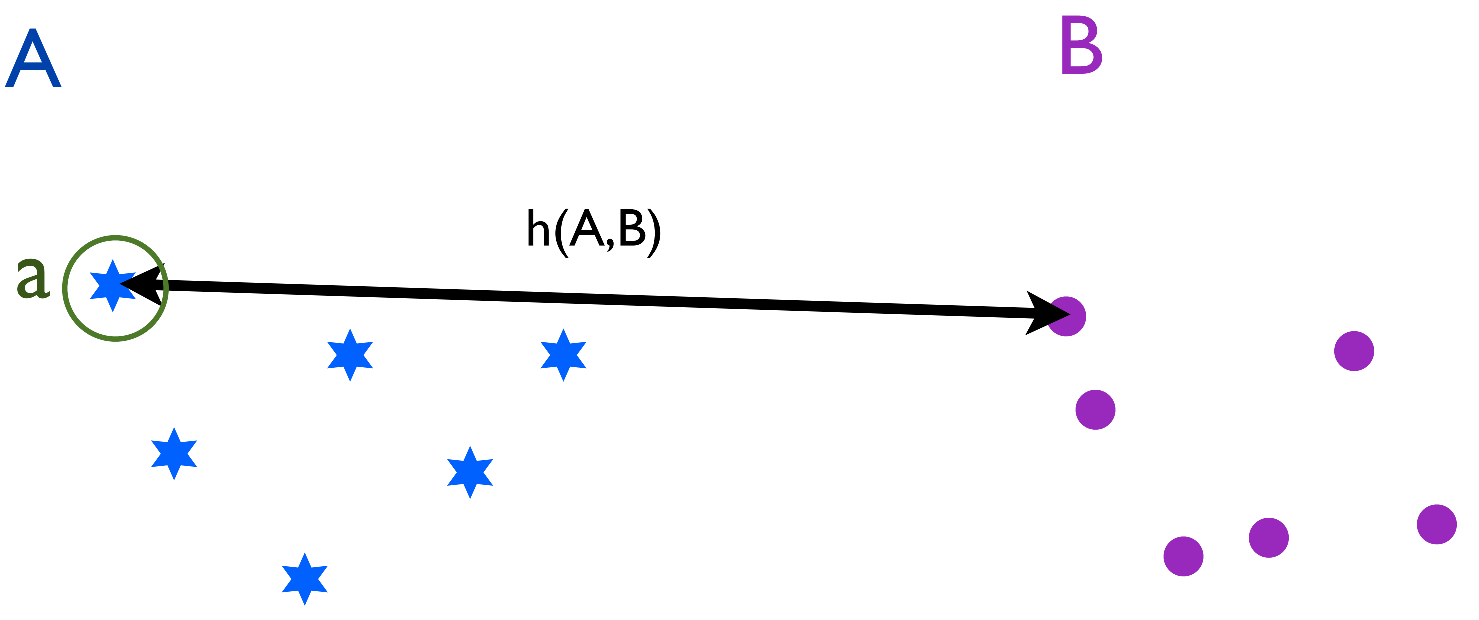
h(A,B) identifies the point a in A that is farthest from any point of B and measures the distance from a to its nearest neighbour in B

Hausdorff metric example



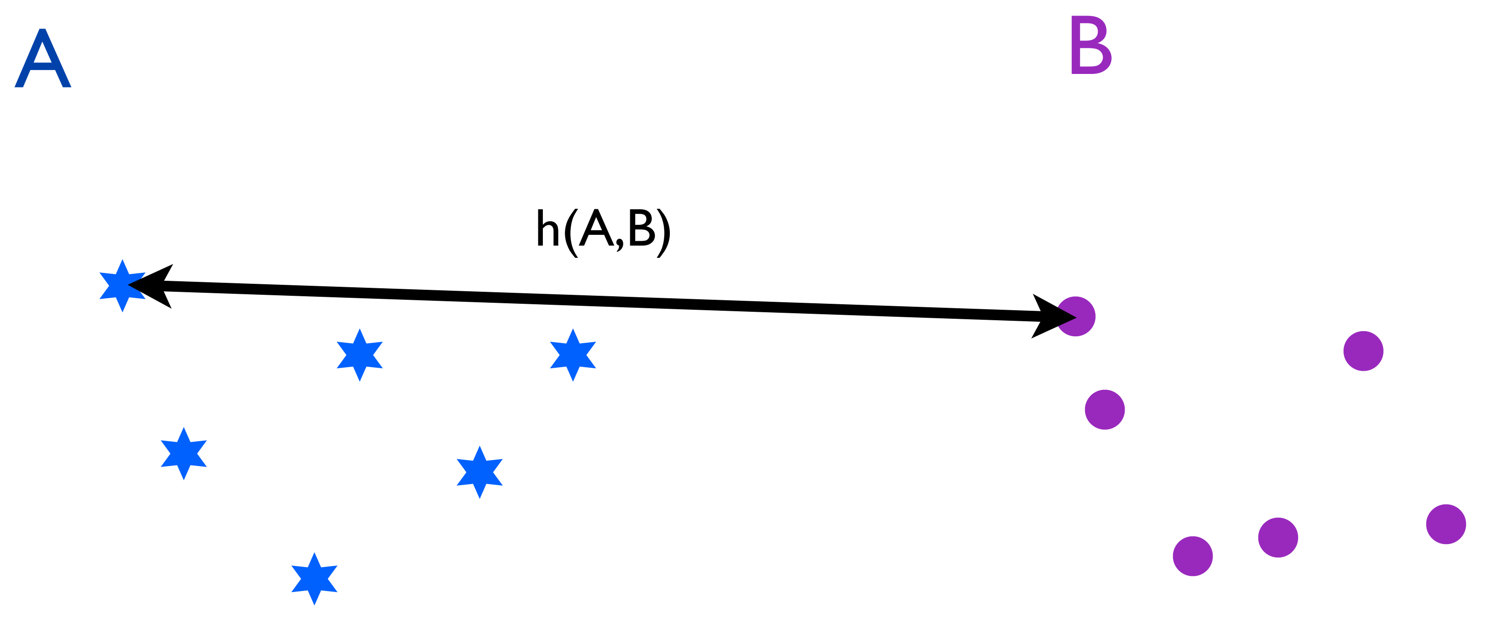
h(A,B) identifies the point a in A that is farthest from any point of B and measures the distance from a to its nearest neighbour in B

Hausdorff metric example



h(A,B) identifies the point a in A that is farthest from any point of B and measures the distance from a to its nearest neighbour in B

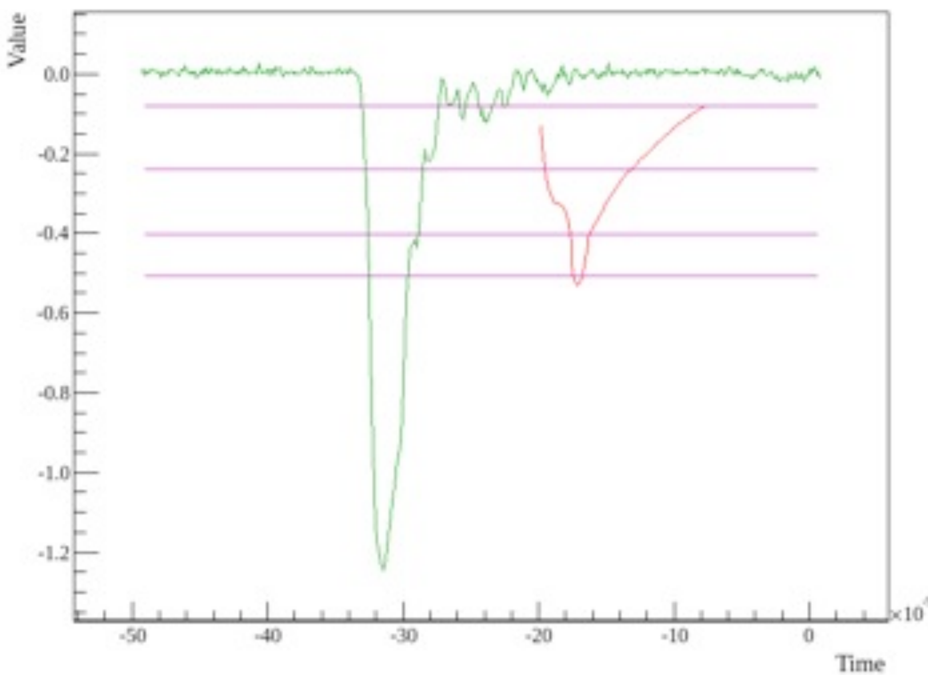
Hausdorff metric example

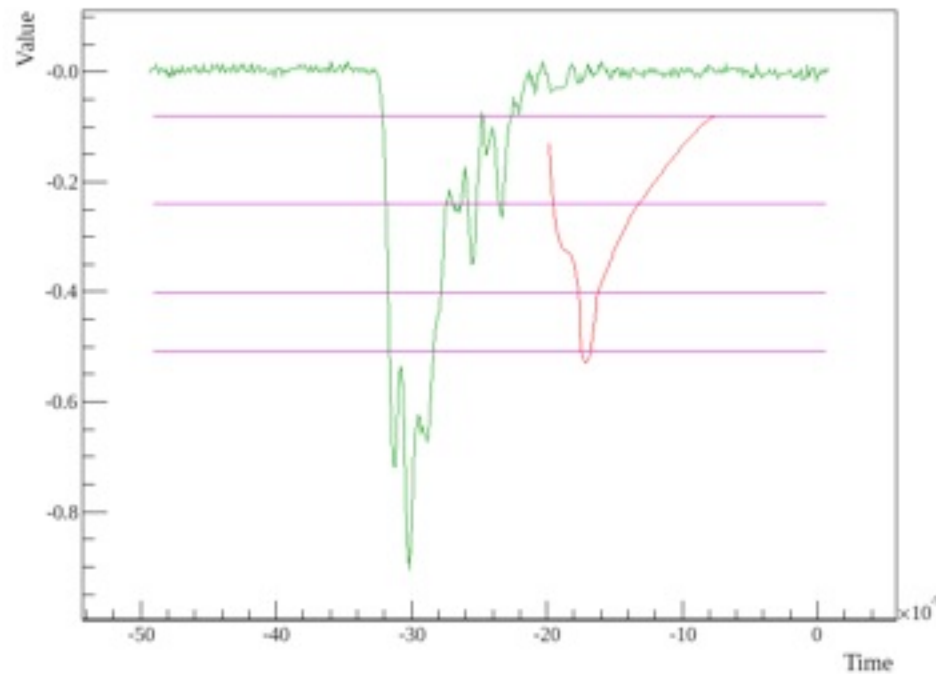


h(A,B) ≠ h(B,A), thus:

$$H(A,B)=max(h(A,B),h(B,A))$$

Hausdorff metric example





Hd = 0.715624

Hd = 0.374432

Testing method:

Testing method:

1. A signal is randomly chosen from the DB

Testing method:

1. A signal is randomly chosen from the DB
2. The signal is then used to generate a simulated signal (consisting of 8 nodes which are smeared)

Testing method:

1. A signal is randomly chosen from the DB
2. The signal is then used to generate a simulated signal (consisting of 8 nodes, which are smeared)
3. The simulated signal is compared with each signal from the DB

Testing method:

1. A signal is randomly chosen from the DB
2. The signal is then used to generate a simulated signal (consisting of 8 nodes which are smeared)
3. The simulated signal is compared with each signal from the DB
4. A best matching signal is returned

Testing method:

1. A signal is randomly chosen from the DB
2. The signal is then used to generate a simulated signal (consisting of 8 nodes which are smeared)
3. The simulated signal is compared with each signal from the DB
4. A best matching signal is returned

Evaluation:

Testing method:

1. A signal is randomly chosen from the DB
2. The signal is then used to generate a simulated signal (consisting of 8 nodes which are smeared)
3. The simulated signal is compared with each signal from the DB
4. A best matching signal is returned

Evaluation:

1. Is the best matching signal the original one from which the simulated signal was produced? (a perfect match)

Testing method:

1. A signal is randomly chosen from the DB
2. The signal is then used to generate a simulated signal (consisting of 8 nodes which are smeared)
3. The simulated signal is compared with each signal from the DB
4. A best matching signal is returned

Evaluation:

1. Is the best matching signal the original one from which the simulated signal was produced? (a perfect match)
2. Is the position of gamma quants hit the same for both signals? (a hit position match)

What's been tested:

Perfect matches - can the comparison method find the exact signal from which a simulated signal was generated?

Hit position matches - does the signal returned by the comparing method correspond to the same hit position as the simulated signal?

Average same best result rate - for how many best-matching signals was the comparison result identical?

CPU clock ticks - a way to measure complexity

Database:
11 possible hit points
500 signals in for each point

Simulation:
std deviation = 2.5×10^{-10}
interpolation: Akima,
node density: 5

Results using new (better quality) data:

	Comparing integrals	Times at thresholds	Least Squares	Hausdorff Distance
'Perfect' Matches	0.12%	1.96%	5.67%	0%
Hit position matches	100%	100%	100%	100%
Average same best result rate	1	1.022	410.974	46.1357
CPU clock ticks	2.5×10^{7}	1.86×10^{10}	5.0×10^{10}	2.56×10^{10}

Results using 'old' data:
(for 4000 independant tests)

Perfect matches:

Least squares:	194 (4.85%)
Times at thresholds:	74 (1.85%)
Integrals:	5 (0.125%)
Hausdorff distance:	6 (0.15%)

Hit position matches:

Least squares:	250 (6.25%)
Times at thresholds:	78 (1.85%)
Integrals:	95 (2.375%)
Hausdorff distance:	178 (4.45%)

The End