Tracks Recognition

Mikhail Hushchyn

MLHEP 2016, Lund

Tracks Recognition

The Course Introduction

Why

〉 This course is about how to find tracks of the particles among a set of hits.

OR

〉 This course is about how to find smooth curves among the number of points.

What

1. Problem formulation and key definitions
2. Global Methods of Tracks Recognition
3. Local Methods of Tracks Recognition
4. Tracks recognition in real experiments

How

Duration:

〉 2 lectures x 1.5 hours

〉 2 seminars x 1.5 hours

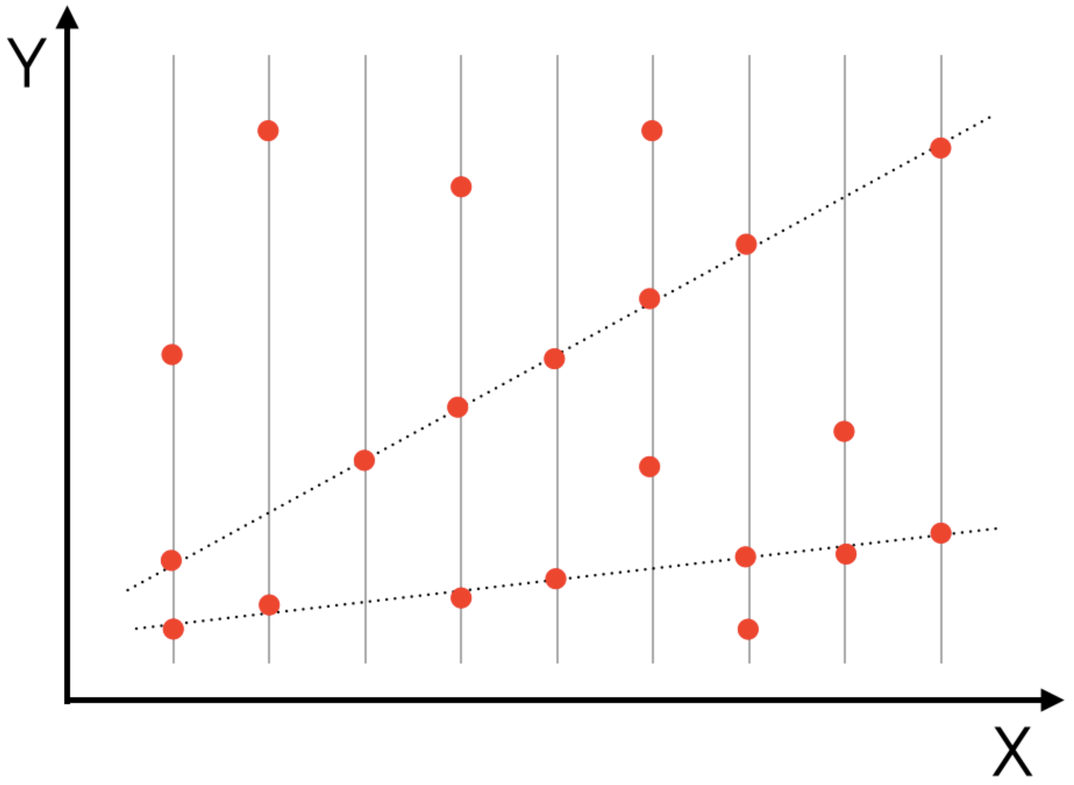
Ask your questions at any time!

Tracks Recognition

Problem Formulation

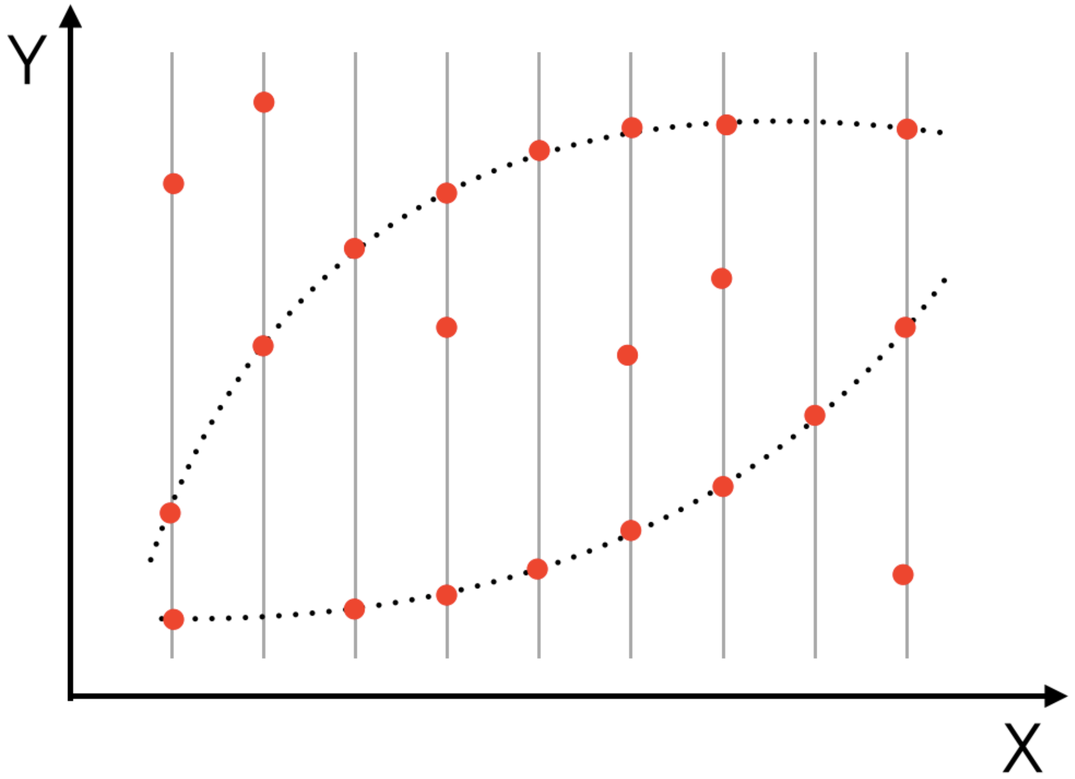
Problem Formulation

- For a given set of points find smooth curves such as straight lines, circles, etc.
- Estimate the lines parameters
- Estimate the point which belong to the lines



Problem Formulation

- Often we have assumptions about kind of the track: a straight line, a part of a circle…
- Some methods requires that assumptions
- Other ones just require that the track is not very steep



Tracks Recognition

Key definitions and quality metrics.

The Reference Set

〉 Reference set - is a set of tracks that an ideally performing algorithm should find.

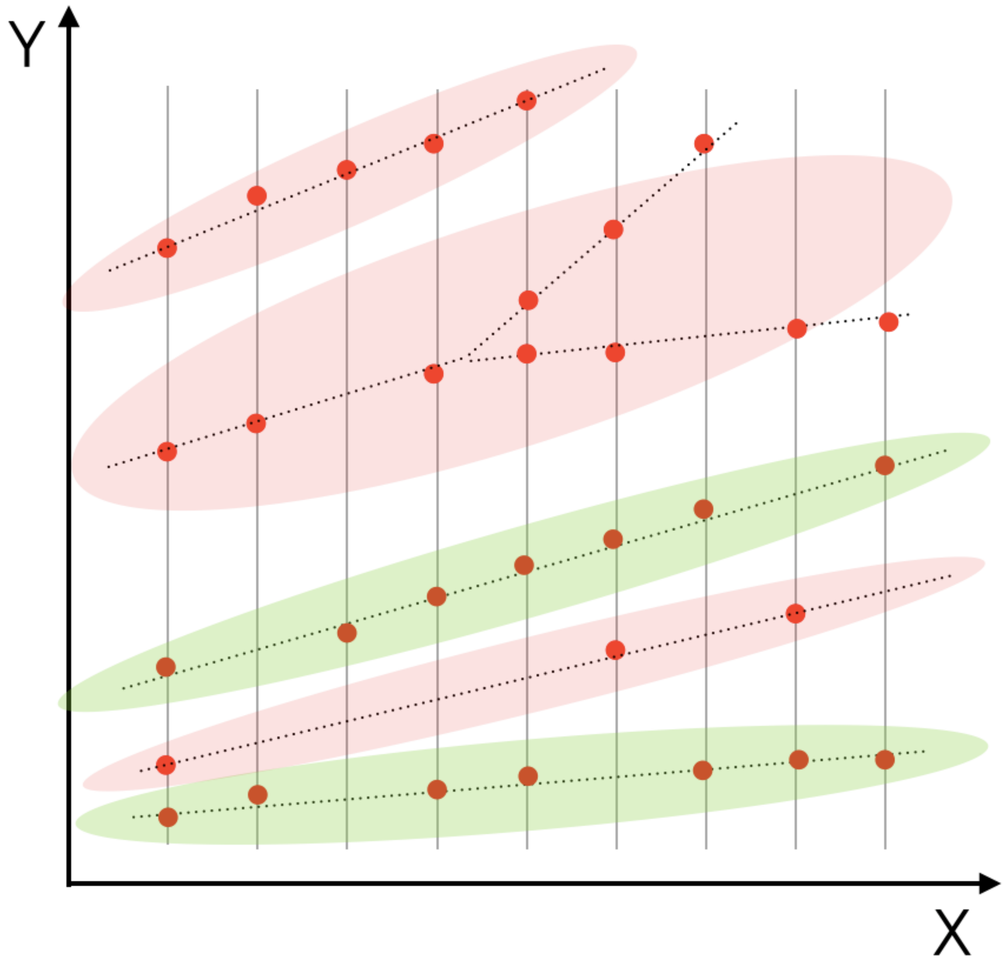
〉 Selection of reference tracks depends on the physics motivation of the experiment.

Criteria:

〉 A track should lay inside of the geometrical acceptance

〉 A track should has certain minimal number of hits

〉 A track should has a particular shape

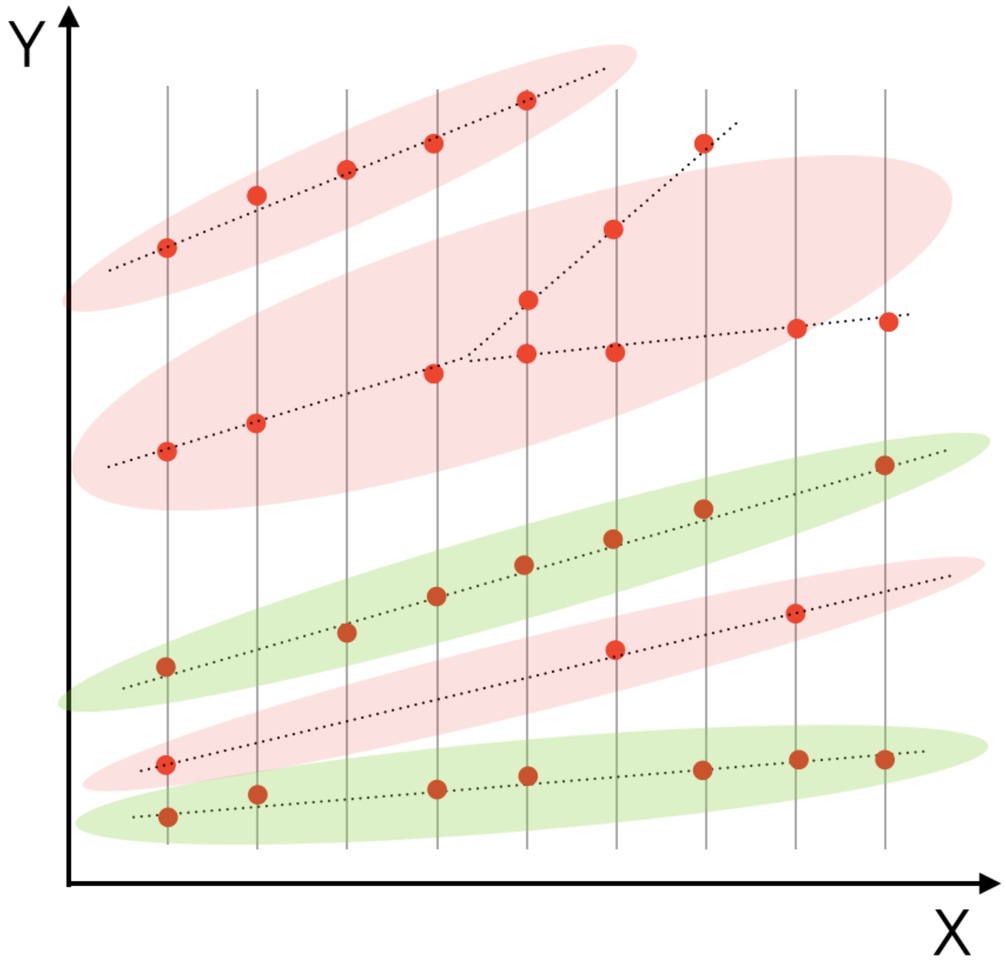


The Reference Set

The definition of the reference set can be regarded as a definition of effective geometrical acceptance:

$$\epsilon_{geo} = \frac{N_{ref}}{N_{total}}$$

with N denoting the number of particles of interest in the reference set and in total.



Track Finding Efficiency. Hit Matching.

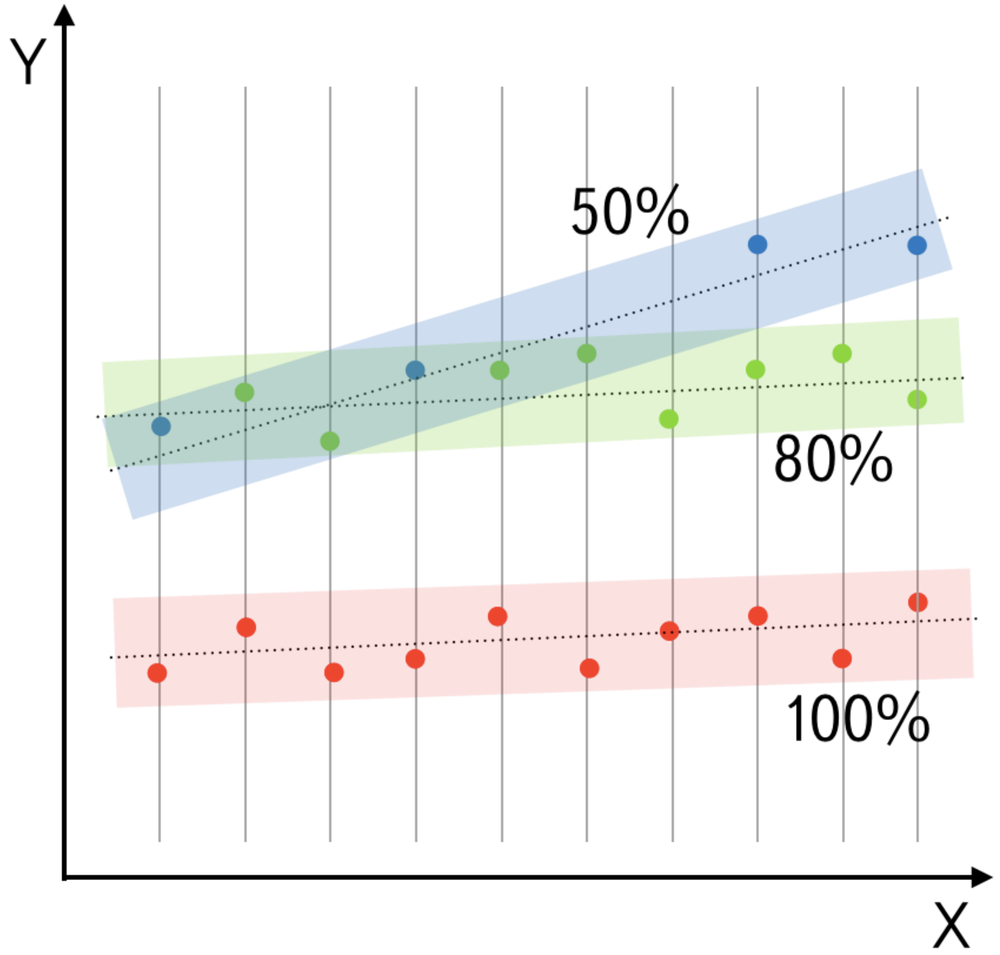
The track finding efficiency is defined as:

$$\epsilon_{track} = \frac{N_{reco\_true\_hits}}{N_{reco\_hits}} * 100\%$$

where N denotes the number of recognized the track's true hits and number of recognized hits respectively.

The track is considered to be reconstructed if its efficiency is higher than, for example, 70%.

This method is stable in the limit of very high track densities.

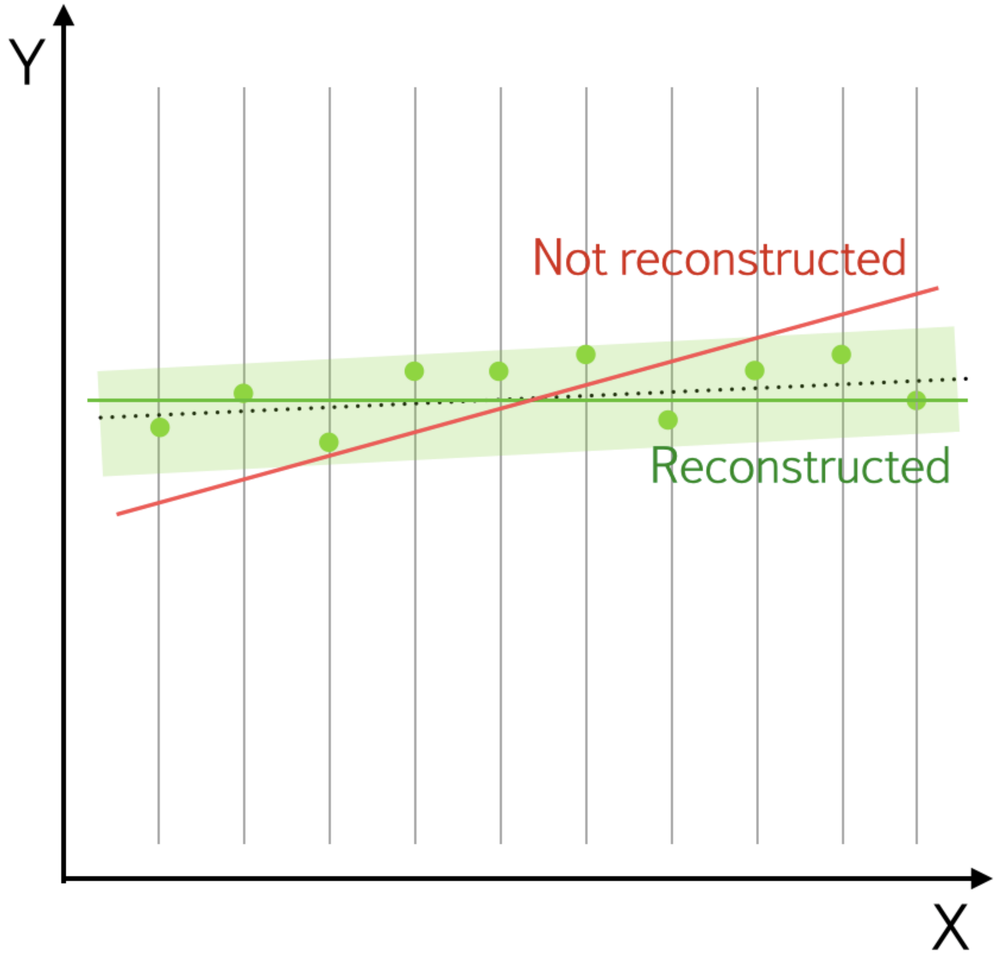


Track Finding Efficiency. Parameter Matching.

The reconstructed parameters of a track are compared with its true parameters. If the parameters are in agreement with the true ones within certain limits, the track is denoted as reconstructed.

The agreement limits are motivated by the physics meaning of the experiment.

The method is not stable in the limit of very high track densities. This can lead to a situation when the track finding efficiency improves with increasing track density.



Track Finding Efficiency

The reconstruction efficiency is defined as:

$$\epsilon_{reco} = \frac{N_{ref}^{reco}}{N_{ref}}$$

where N_{ref}^{reco} is the number of reference tracks that are reconstructed by at least one track.

Number of non-reference tracks ($N_{non-ref}^{reco}$) should also be controlled. Normally the relation:

$$\frac{N_{non-ref}^{reco}}{N_{total} - N_{ref}} \ll \epsilon_{reco}$$

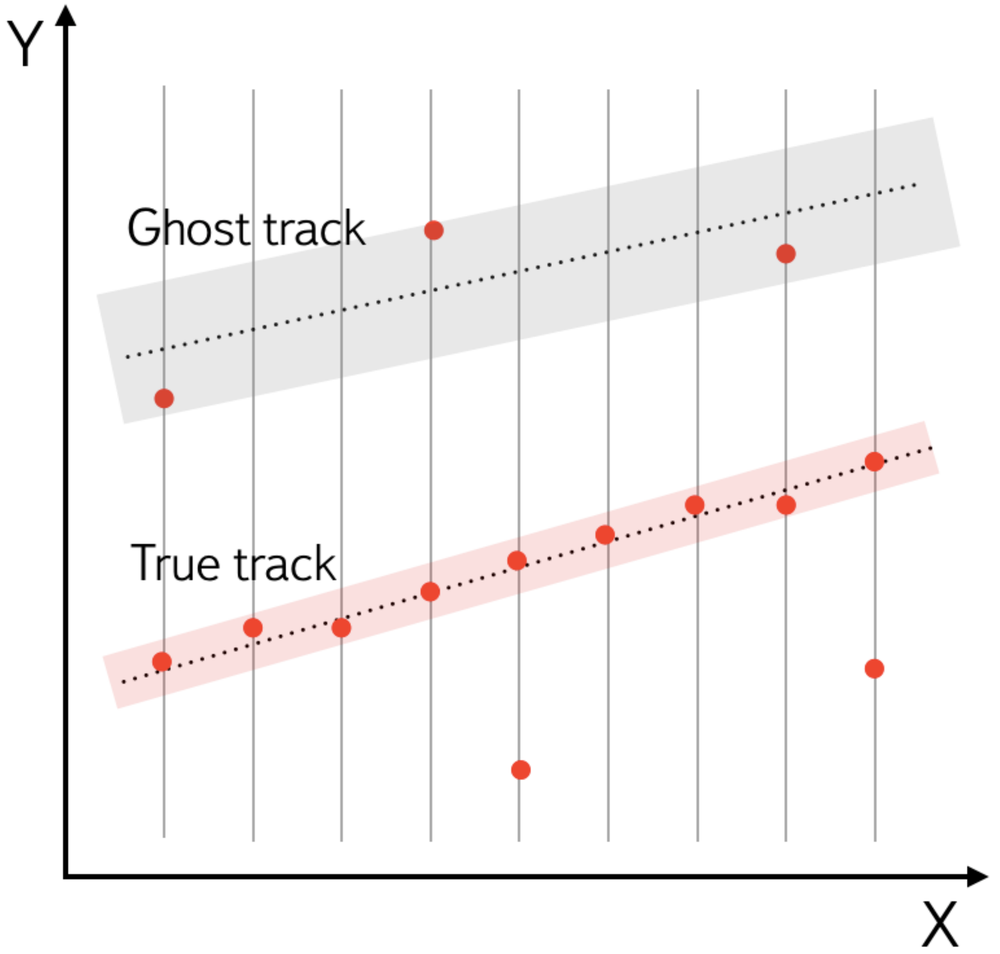
should hold, otherwise the reference criteria might be too strict.

Ghosts

Ghosts are tracks produced by the pattern recognition algorithm that do not reconstruct any true track within or without the reference set.

A ghost rate is defined as:

$$\epsilon_{ghost} = \frac{N^{ghost}}{N_{ref}}$$



Clones

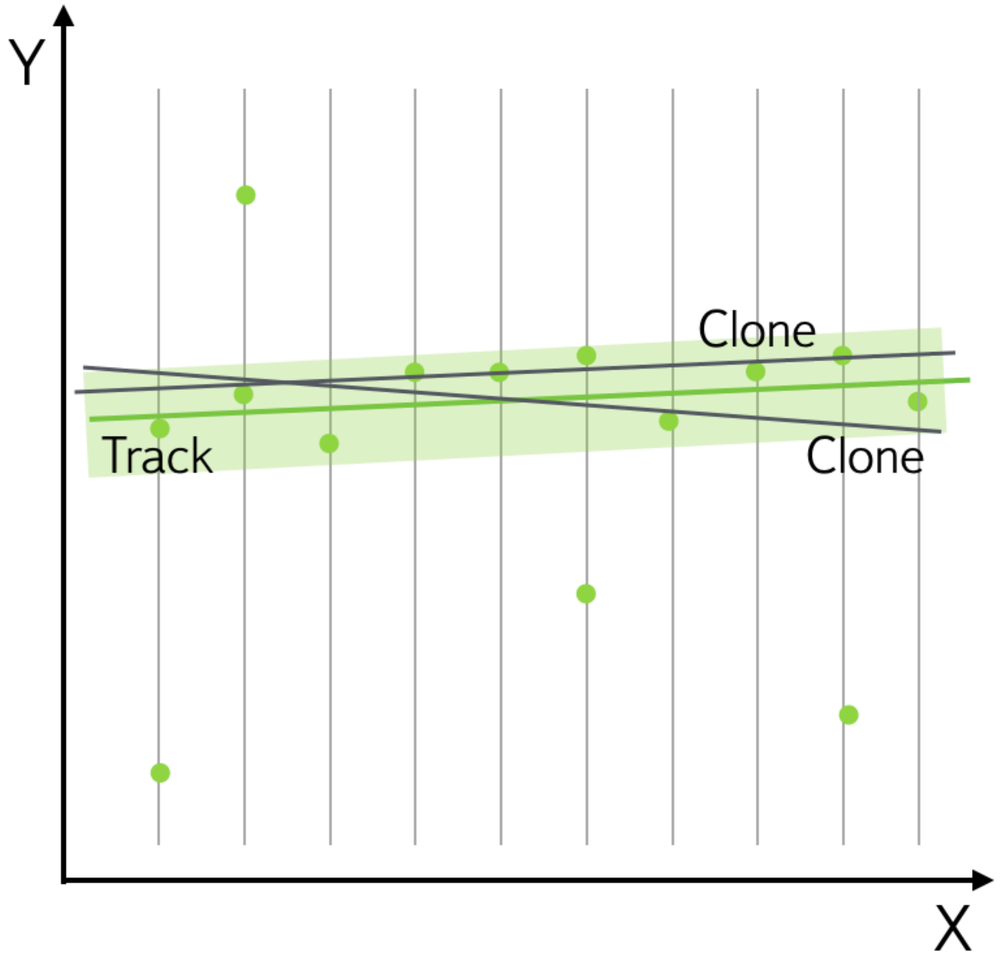
The definitions for efficiency and ghost rate are sensitive to multiple reconstructions of a track. Such redundant reconstructions are sometimes called clones.

For a given track m with N_m^{reco} tracks reconstructing it, the number of clones is

$$N_m^{clone} = \begin{cases} N_m^{reco} - 1, \text{if} N_m^{reco} > 0 \\ 0 \qquad\quad , \text{otherwise} \end{cases}$$

A clone rate then is

$$\epsilon_{clone} = \frac{\sum_m N_m^{clone}}{N_{ref}}$$



Parameter Resolution

The quality of a track parameter X_i estimation is reflected in the parameter residual:

$$R(X_i) = X_i^{rec} - X_i^{true}$$

The normalized parameter residual is

$$P(X_i) = \frac{X_i^{rec} - X_i^{true}}{\sqrt{C_{ii}}}$$

which is often called the pull of this parameter. C in the equation above is the parameter covariance matrix. Ideally, the pull should follow a Gaussian distribution with a mean value of zero and a standard deviation of one.

Interplay

Considered metrics:

- Reference set
- Tracks finding efficiencies (hits and parameters matching)
- Ghost and clone rates
- Parameter resolution

It is advisable to use several of the above metrics in combination to measure quality of the tracks recognition methods.

Tracks Recognition

Global Methods

Definition of the Global Methods

〉 Global methods of the tracks recognition process all hits in a similar way.

〉 The result should be independent of the starting point or the order in which hits are processed.

〉 This is unlike the local methods which depend on suitable seeds for track candidates.

Global Methods

1. Template Matching

- Simple Template Matching
- RANSAC

2. Transformation Methods

- The Radon Transform
- Hough transform

3. Neural Network Techniques

- The Denby-Peterson method
- Cellular Automaton
- Rotor models of Hopfield networks

Template Matching

Definition of the Template Matching

The method is used when the number of possible tracks patterns, valid combinations of hits, is finite and the complexity limited enough to handle them all.

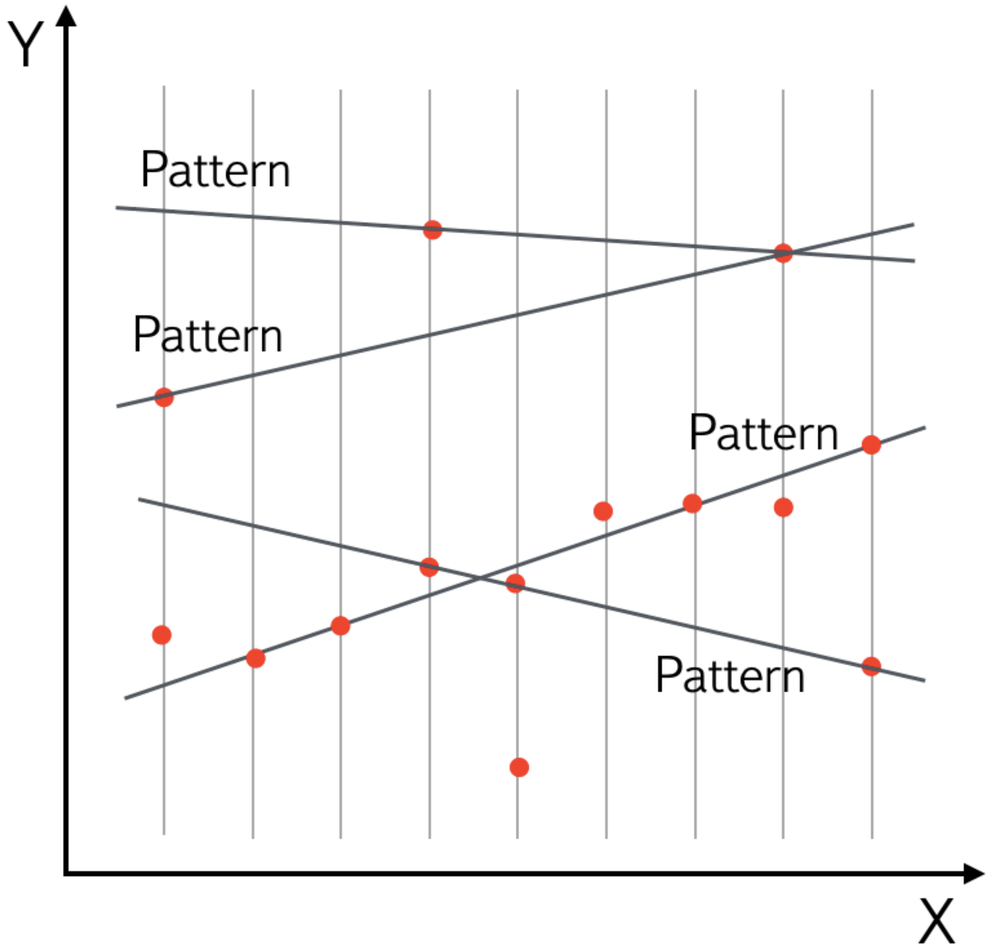
For the each track pattern a template can be defined. An example of the template is that a track should have particular shape (straight line, circle), should pass through the n hits or should starts and ends in special areas.

If a pattern satisfies a template, the pattern is denoted as a track candidate.

Simple Template Matching

Simple Template Matching

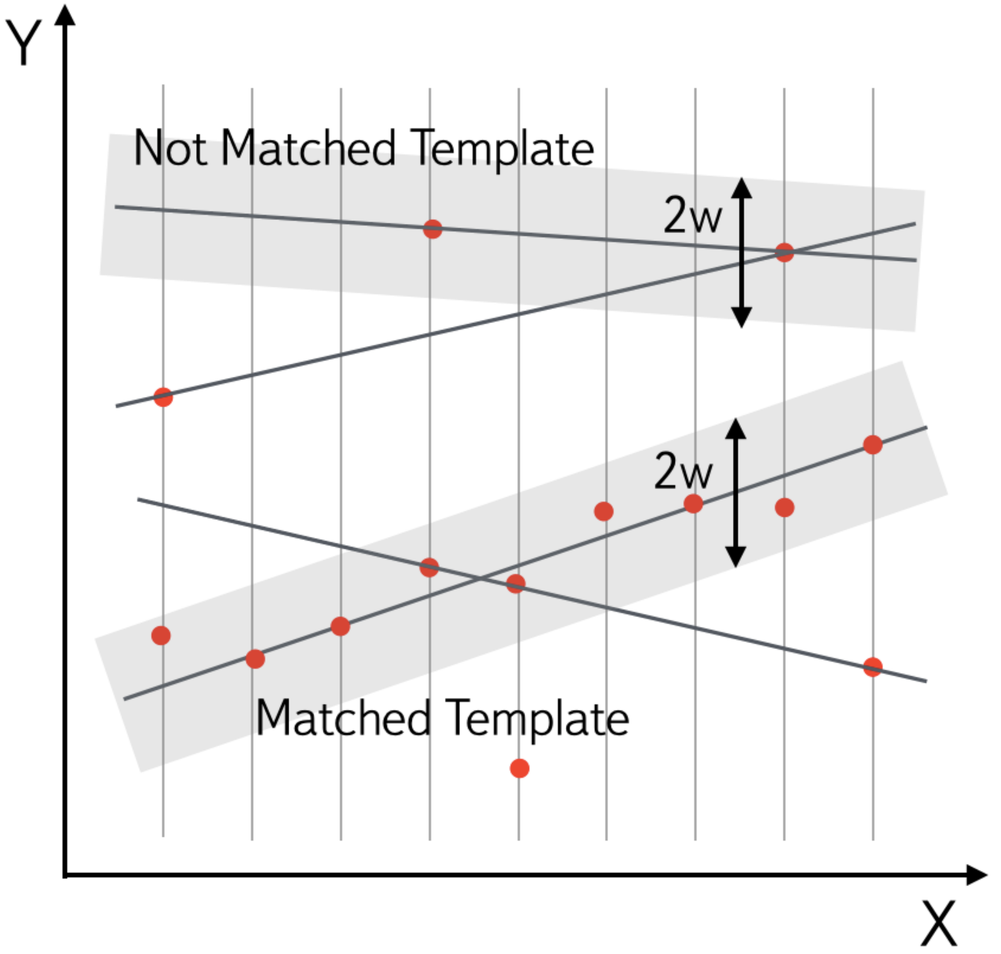
For straight tracks a pattern can be defined as a straight line which passes through at least two hits.



Simple Template Matching

Then, a template can be defined as a straight line which passes through at least 2 hits and contains as least 8 hits within a window width (w) from the line.

The hits which match the template are marked as a track candidate.



Simple Template Matching

Simple Algorithm:

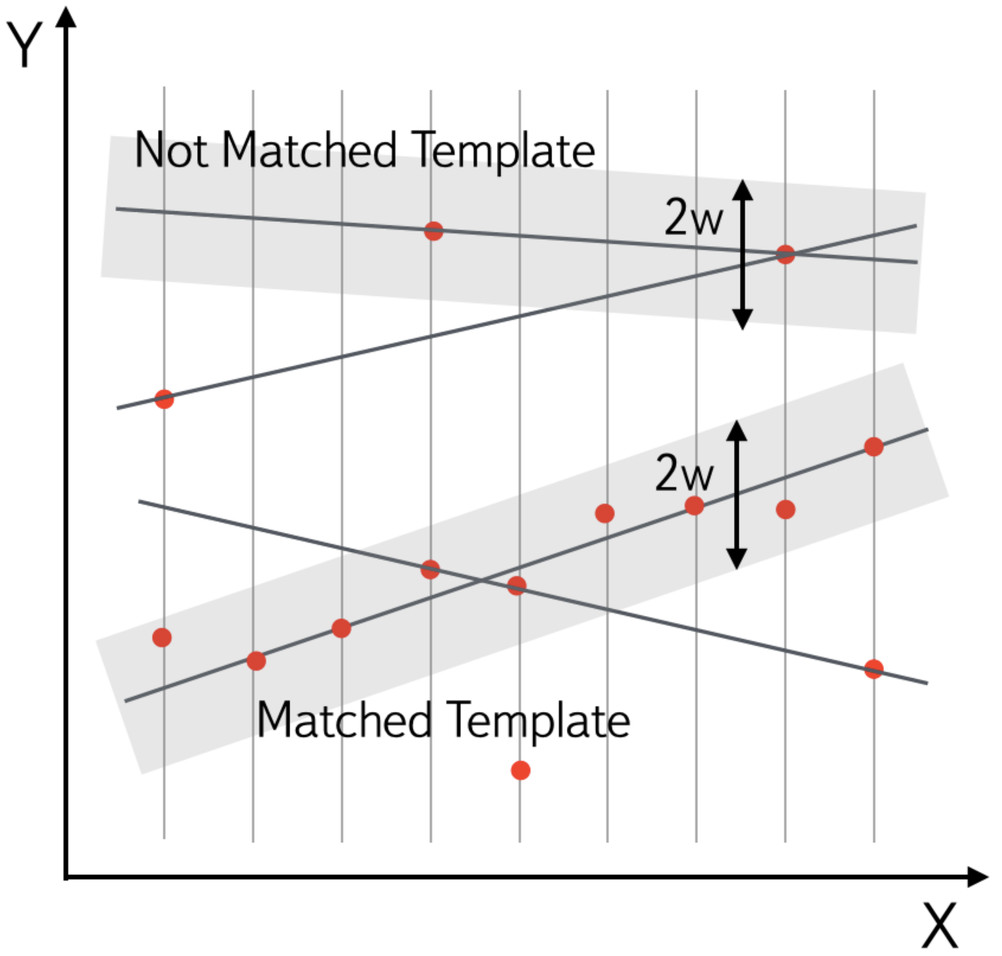
1. Select start and end hits which don't belong to any recognized tracks before.

2. Build a line through these hits.

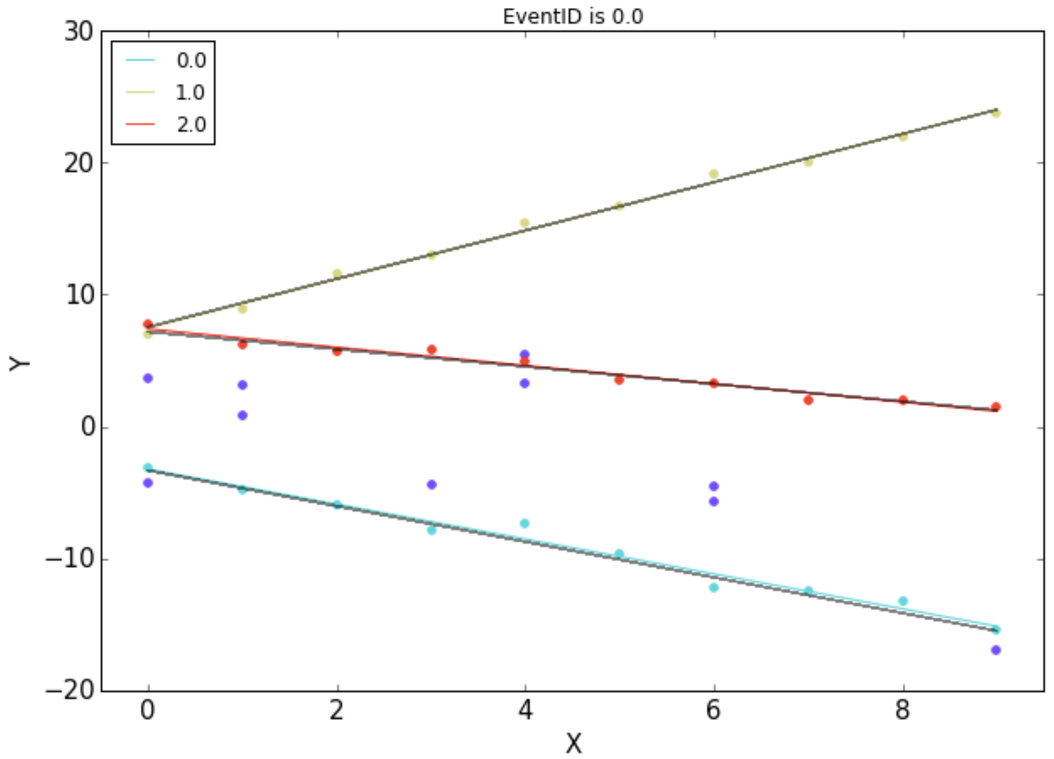
3. Count hits within a window width from the line.

4. If number of the hits is larger than minimum number, mark these hits as recognized track.

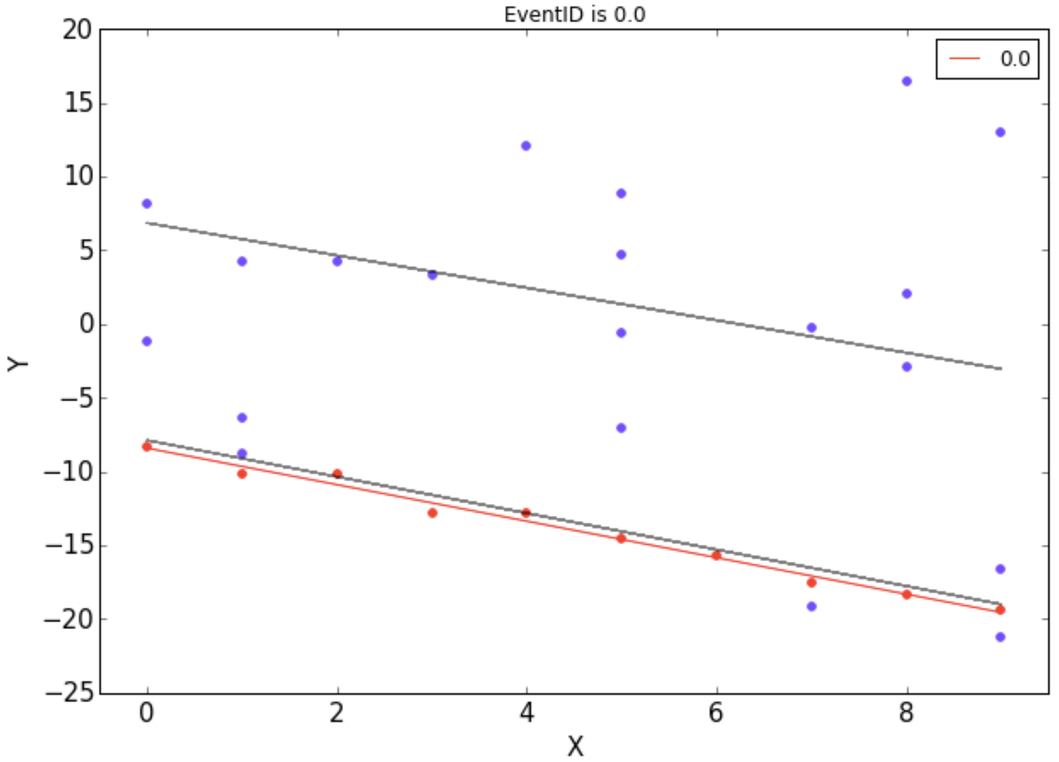
5. Repeat 1-4 steps until all pairs of hits are not selected.



Simple Template Matching. Demonstration.



Well tuned algorithms finds tracks with hight reconstruction efficiency and small ghost and clone rates.



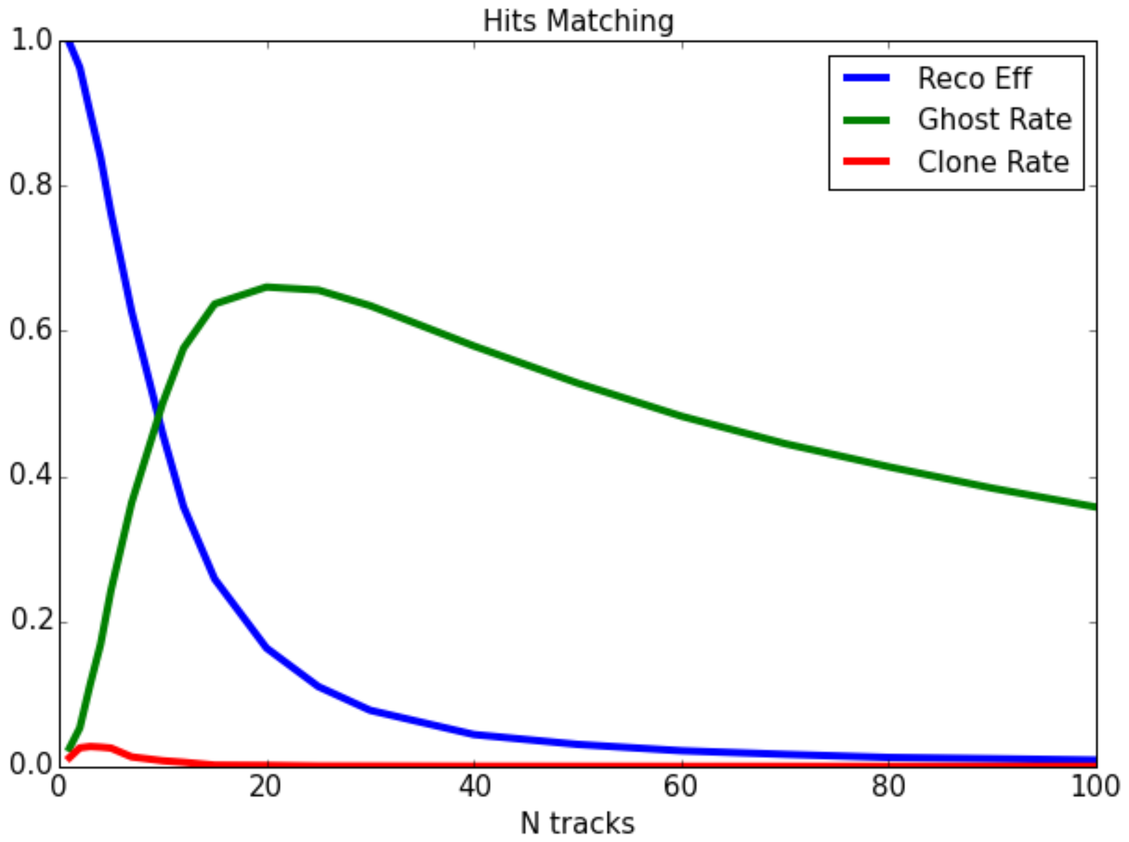
Increasing the window width or/and decreasing minimum number of hits for a track lead to growth of number of ghost.

Simple Template Matching. Demonstration.

Hits matching is used to measure the tracks recognition quality.

With increasing the number of tracks the track candidates contains more hits from different true tracks. This leads to decreasing the reconstruction efficiency (to zero) and increasing the ghost rate.

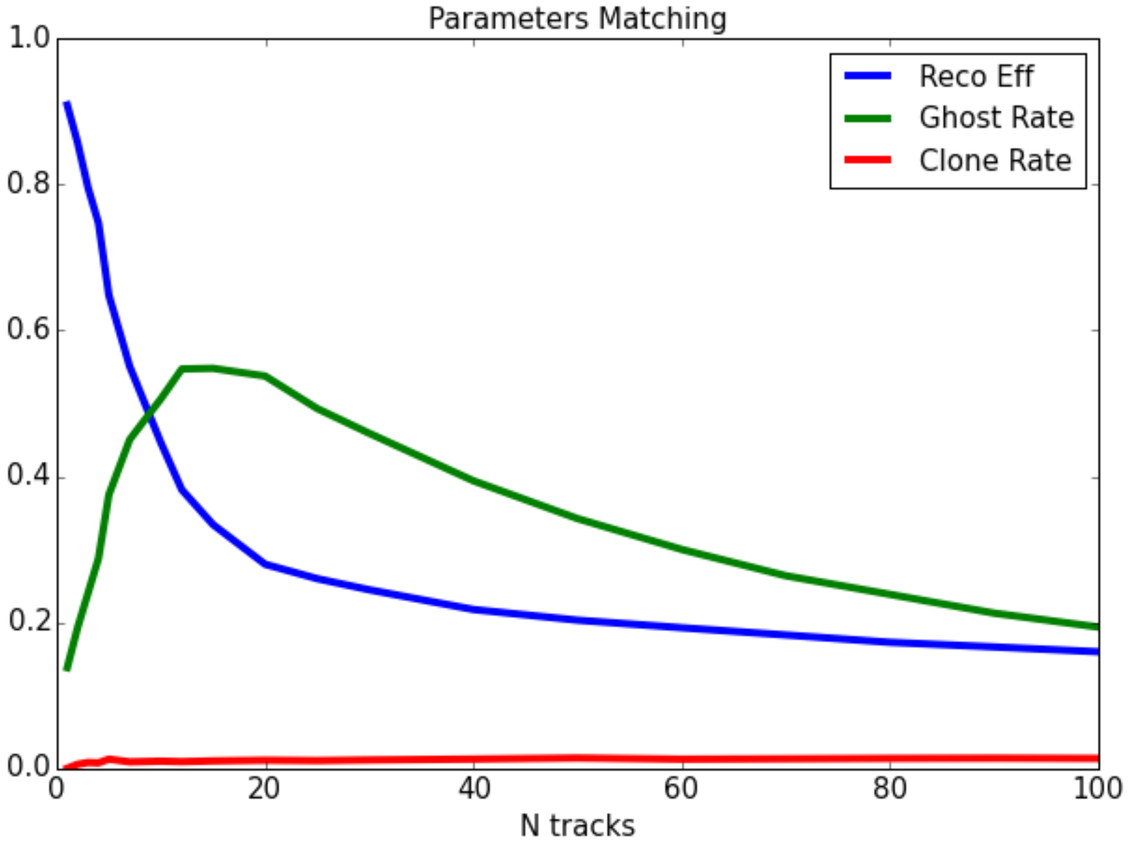
Moreover, due to the constant window width the algorithm finds less track candidates with increasing the number of the true tracks. This leads to decreasing the reconstruction efficiency and the ghost rate for the large number of true tracks.



Simple Template Matching. Demonstration.

Parameters matching is used to measure the tracks recognition quality.

In contrast with the hits matching, the reconstruction efficiency based on the parameters matching does not fall so quickly in large number of tracks. This is due to that the algorithms finds track candidates with parameters close to the true tracks ones.



Simple Template Matching

Advantages of the method:

- Very simple
- Works well for small number of tracks

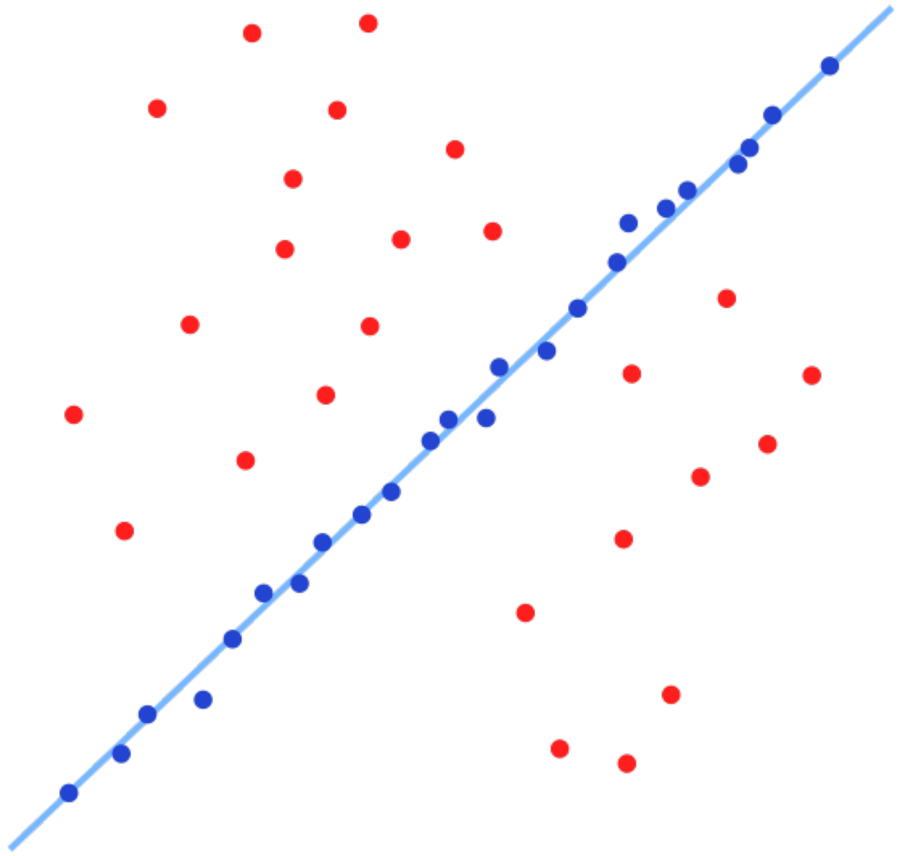
Limitations of the method:

- Works bad for large number of tracks
- Computational not efficient for a large number of complex patterns

RANSAC

Definition of the RANSAC

RANSAC (RANdom SAmple Consensus) is an iterative method for regression problem with samples contaminated with outliers.



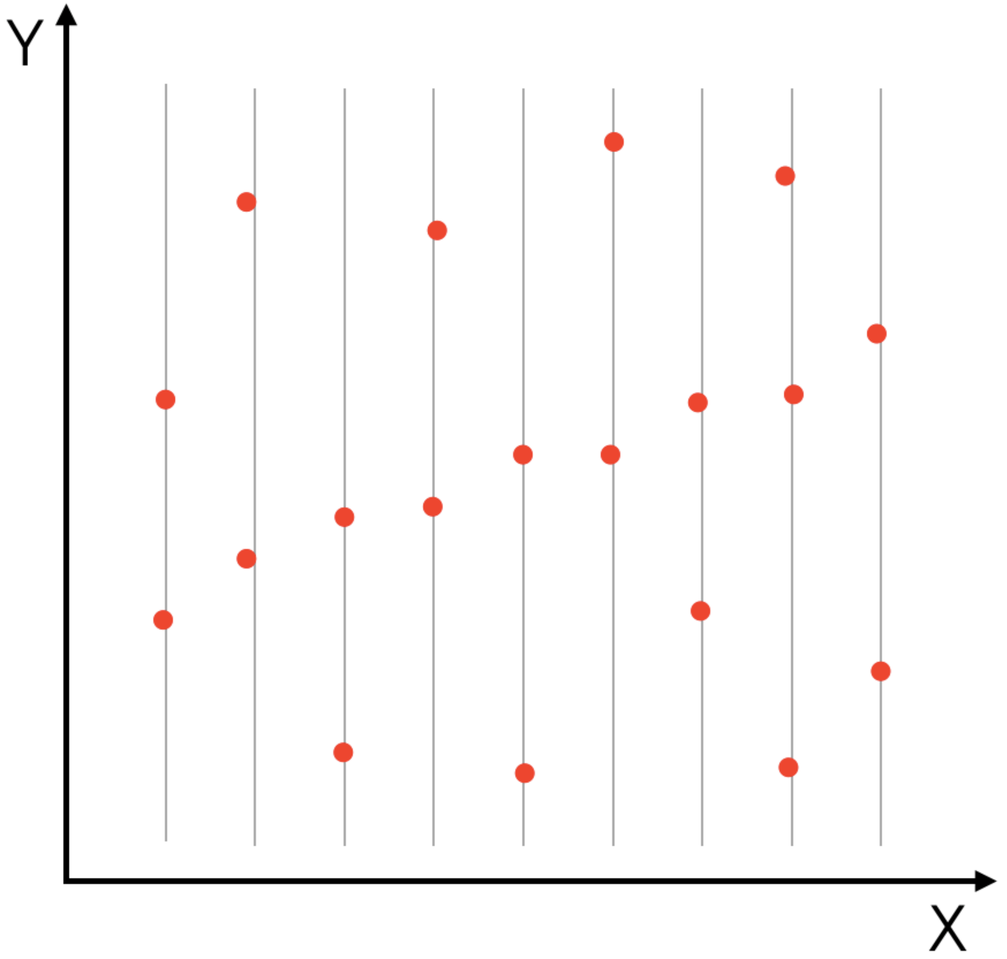
The RANSAC iterates two steps:

1. Fitting a model using a random sample subset.

2. Verification of the model using the entire sample.

RANSAC

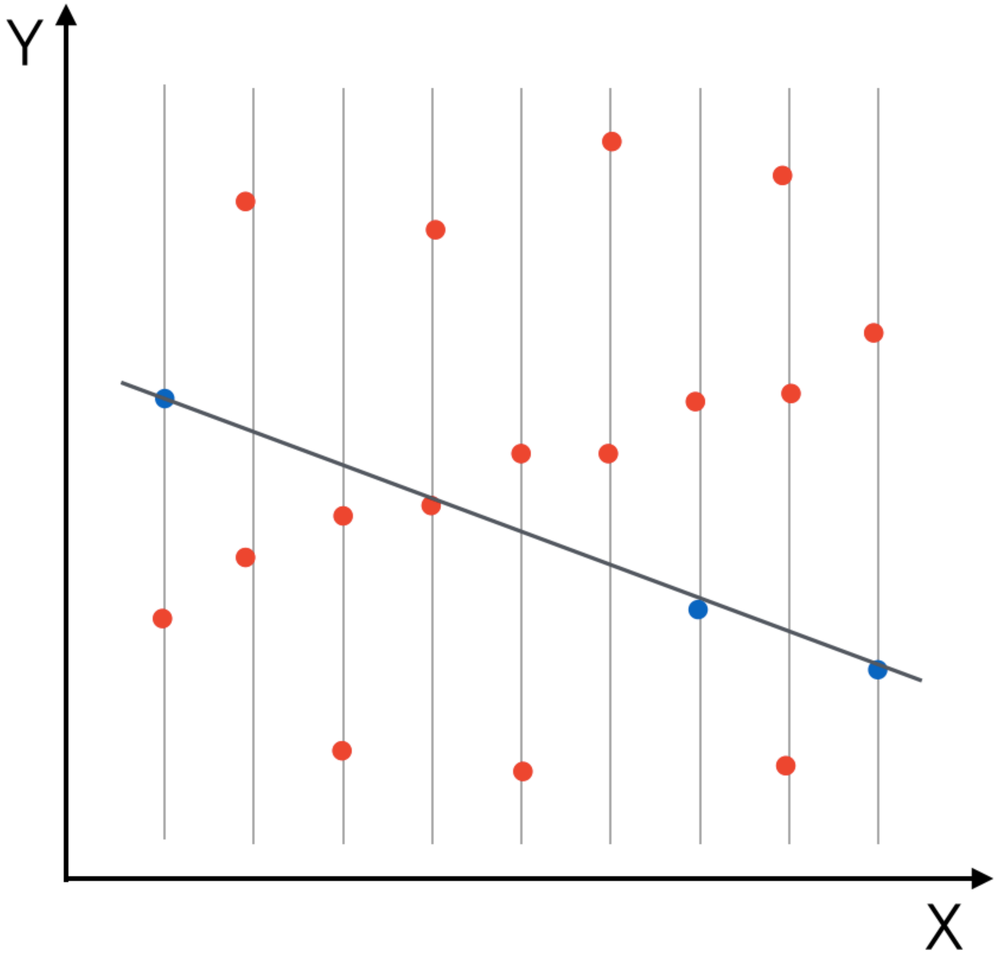
Consider a problem of fitting a linear model with observed data which contains outliers.



RANSAC

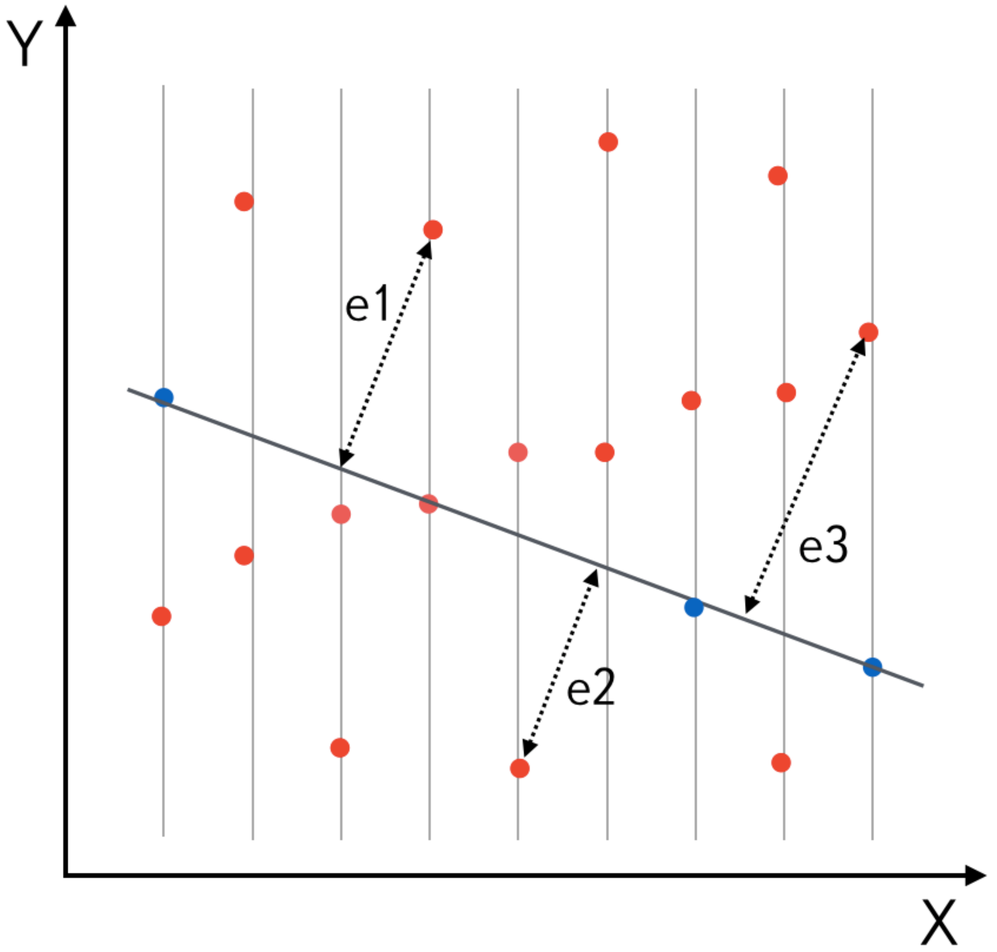
1. The RANSAC selects a random subset of the hits. For example, 3 hits.

2. Then, the linear model is fitted using these subset.



RANSAC

3. Calculate Error of the data with respect to the fitted model.



RANSAC

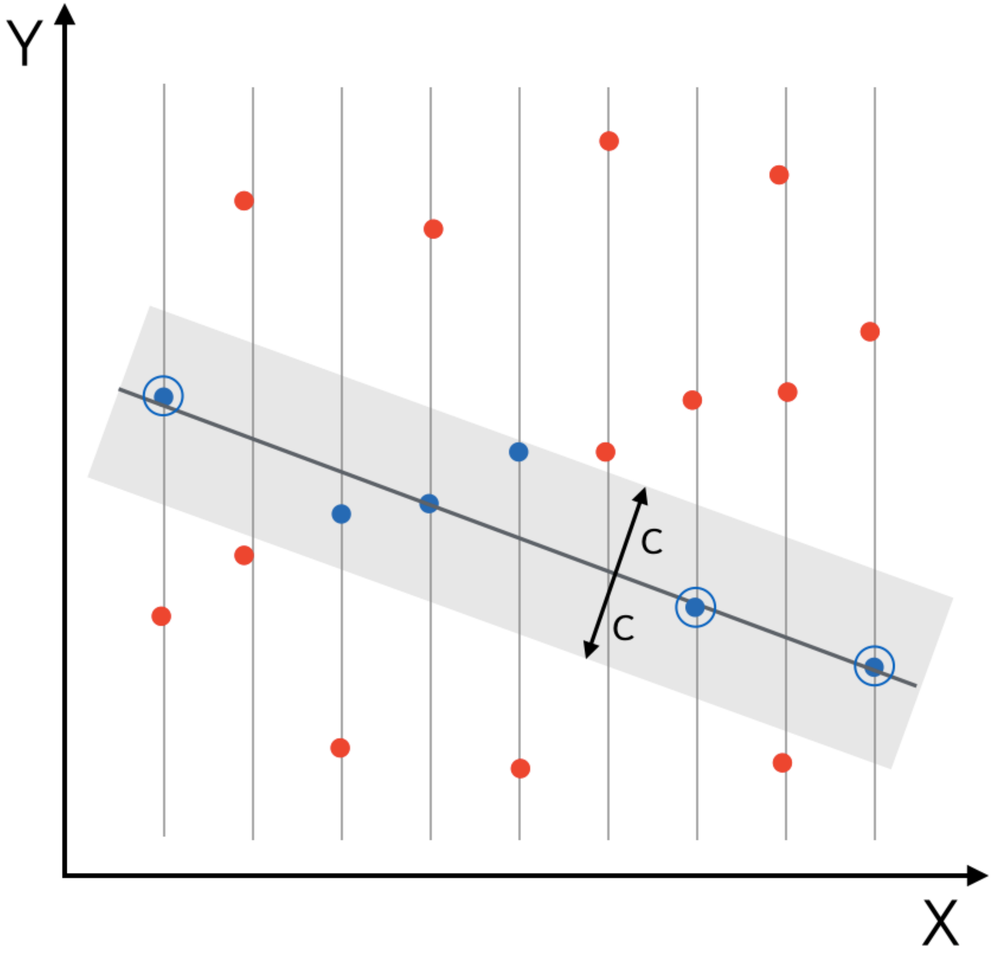
4. Number of inlier candidates is calculated. A hit is marked as inlier when:

$$|e| < c$$

where e is the error of the hit with respect to the fitted model, c is the error threshold.

5. The 1-4 steps are repeated until the maximum number of iteration is not exceeded.

6. A model with maximum number of inliers is returned.



RANSAC

The RANSAC requires to set the following parameters:

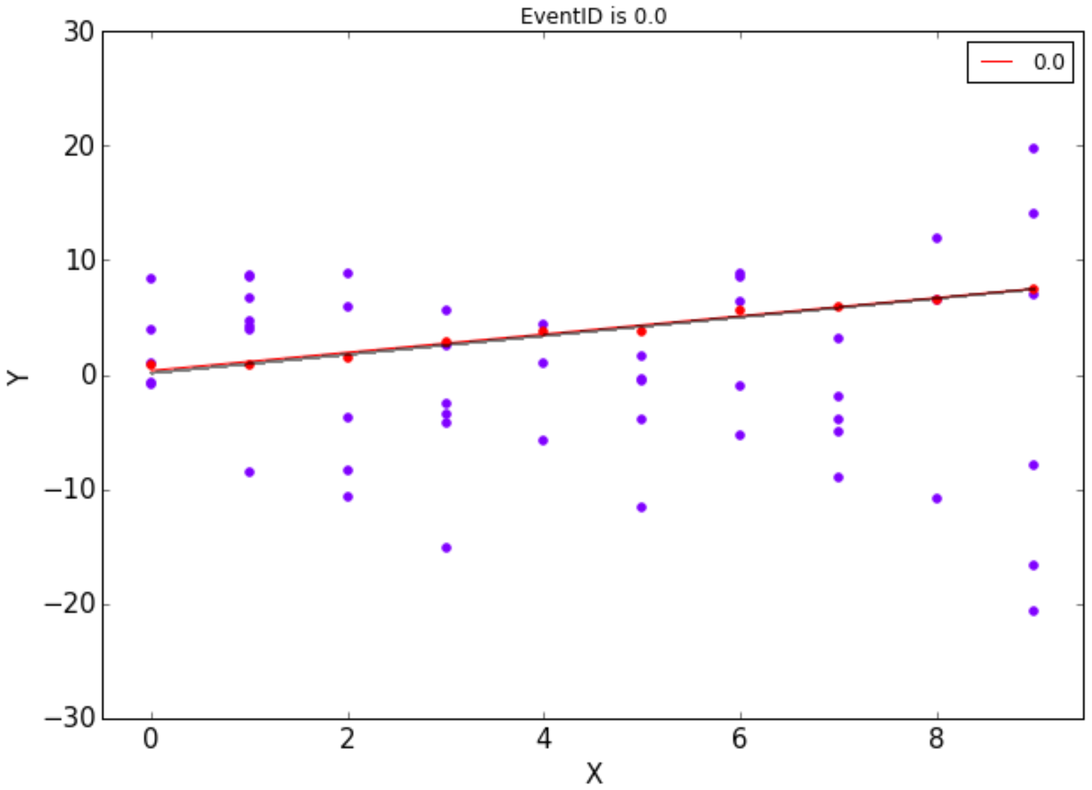
n - number of hits in a random sample subset,
k - number of iterations,
c - the error threshold.

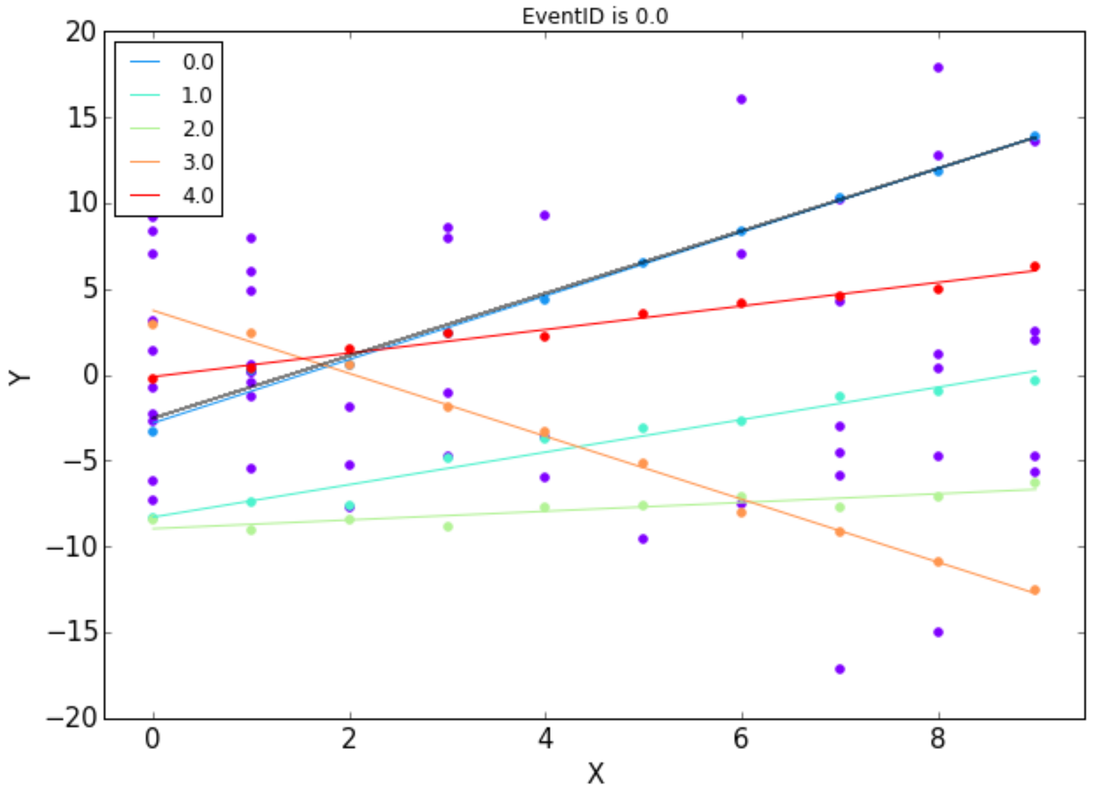
Number of iterations can be chosen by the following way:

$$k = \frac{\log(1-p)}{\log(1-w^n)}$$

where w is probability to select inlier from all hits, p is probability that a random sample subset of size n will consists of only inliers.

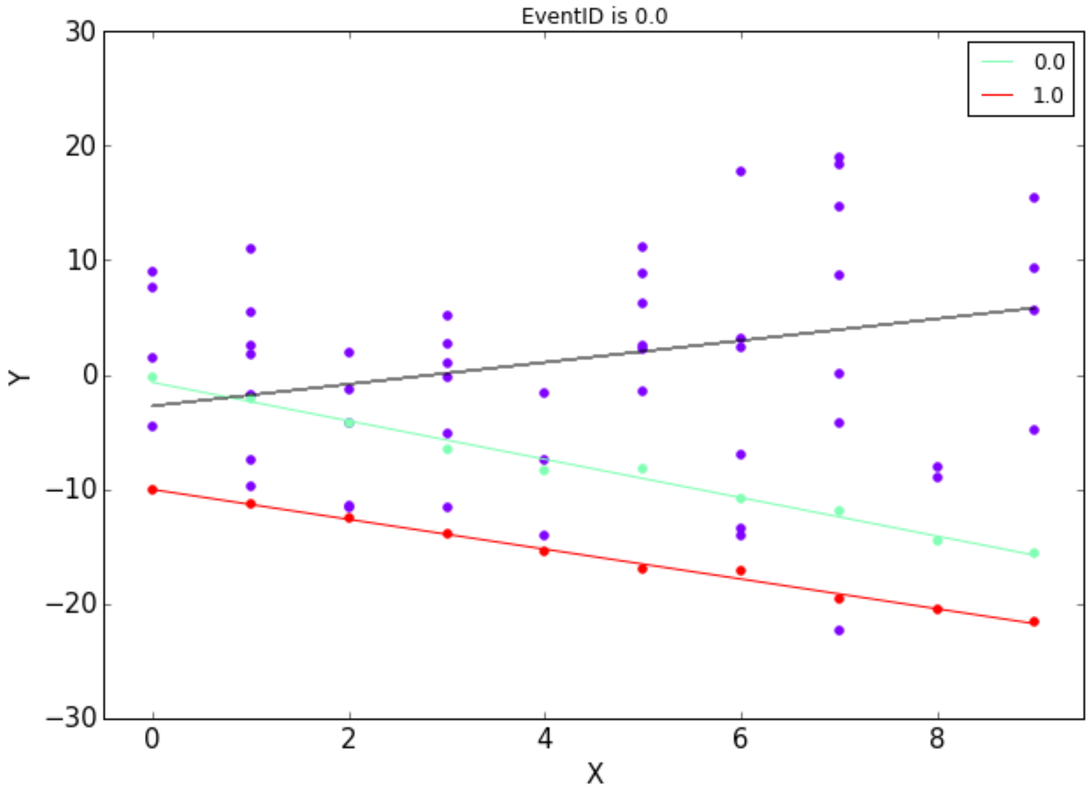
RANSAC.Demonstration.

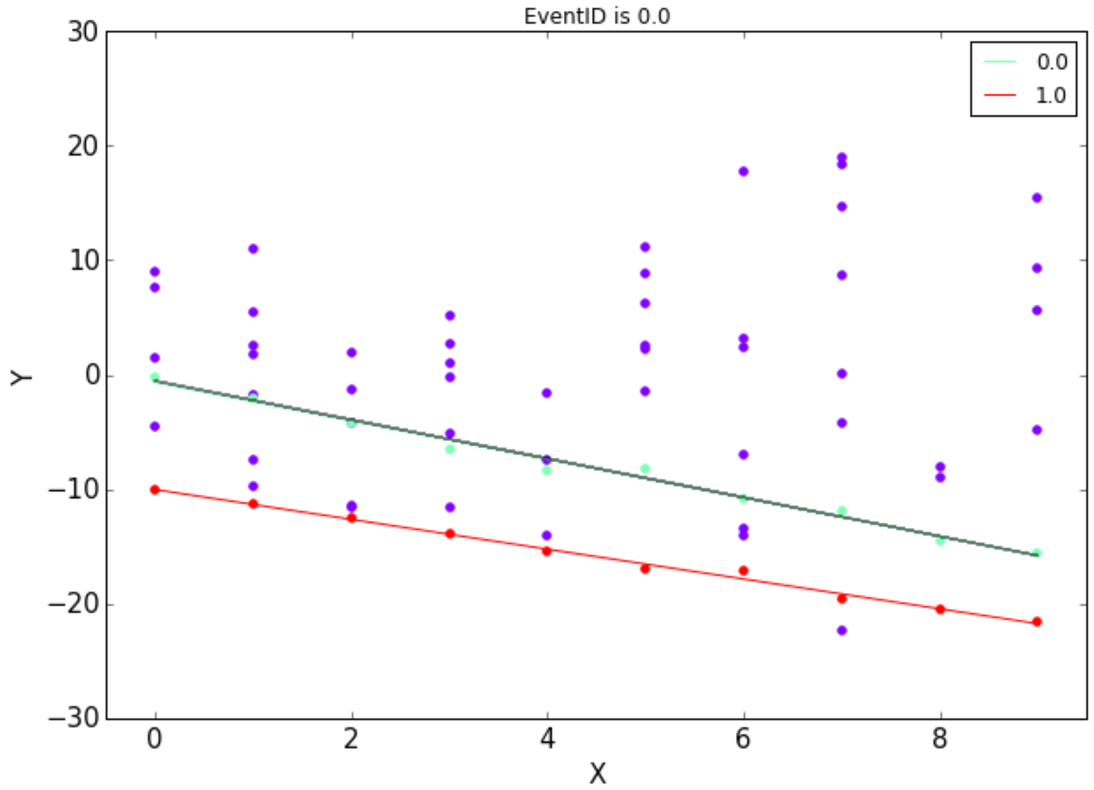




RANSAC works well with highly contaminated data and with data that contains several tracks.

RANSAC.Demonstration.





Not properly selected parameters and small number of iteration can lead to the non optimal solution.

RANSAC

Advantages of the method:

- Robust estimation of the model's parameters
- Works well with significant number of outliers

Limitations of the method:

- Optimal solution needs a large number of iterations
- Estimates only one model for a particular data set

RANSAC Family

Algorithm Generalization:

1. The RANSAC selects a random subset of the hits. For example, 3 hits.

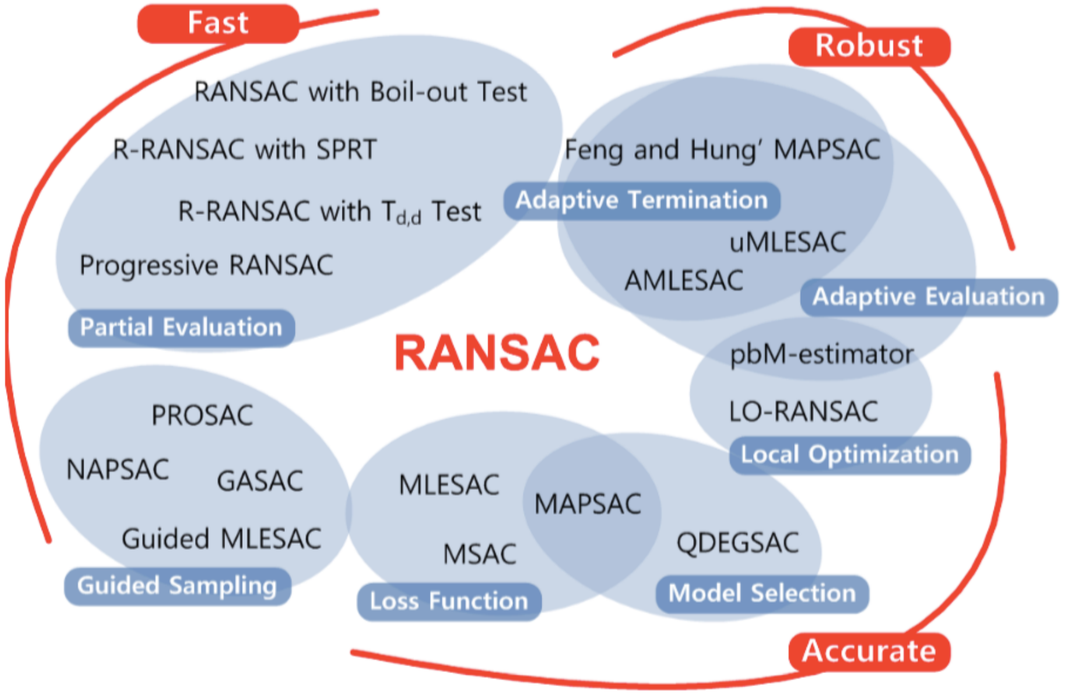
2. Then, the linear model is fitted using these subset.

3. Calculate Error of the data with respect to the fitted model.

4. The best model selection.

5. The 1-4 steps are repeated until the maximum number of iteration is not exceeded.

6. A model with maximum number of inliers is returned.



RANSAC Family

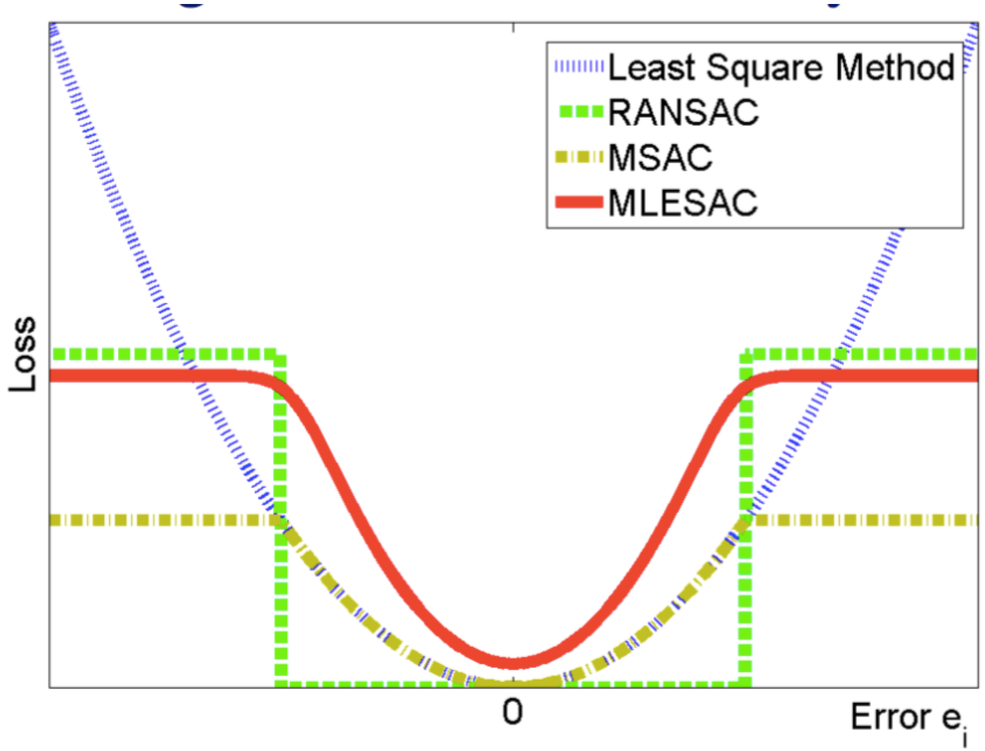
The best model selection can be considered as an optimization problem:

$$M_{best} = arg\min_{M}\{\sum_{d\in D} Loss(Err(d;M))\}$$

where M is a model, D is data, Loss is loss function, Err is error function. The RANSAC uses:

$$\text{Loss}(e) = \begin{cases} 0 & |\,e\,| < c \\ \text{const} & \text{otherwise} \end{cases}$$

where c is an error threshold. This loss function is equal to the counting of the inliers for the each model.



Loss functions

Link: http://www.bmva.org/bmvc/2009/Papers/Paper355/Paper355.pdf

Transformation Methods

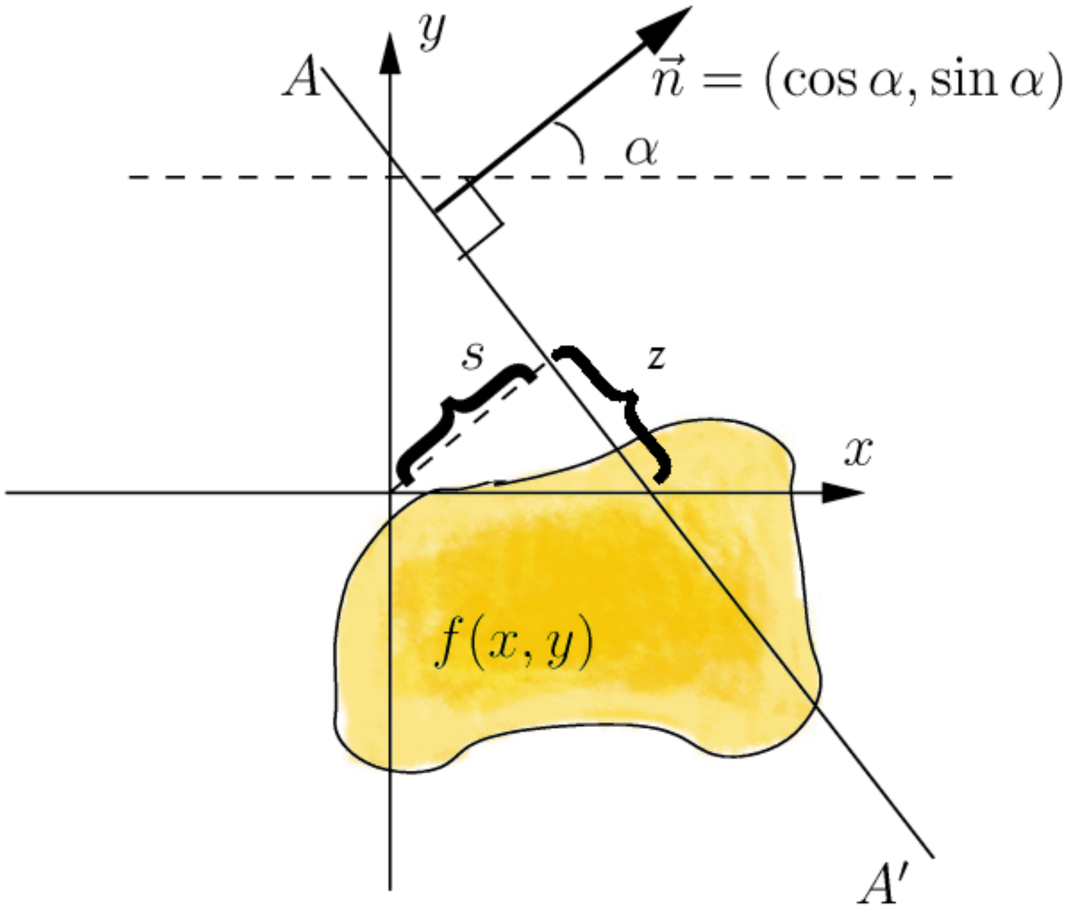
The Radon Transform

Definition of the Radon Transform

Let $f(\mathbf{x}) = f(x,y)$ is continuous function in $\mathbf{R}^2$.

The Radon Transform, Rf, is a function defined on the space of straight lines L in $\mathbf{R}^2$ by the line integral along each such line:

$$Rf(L) = \int_L f(\mathbf{x})\,|d\mathbf{x}|$$



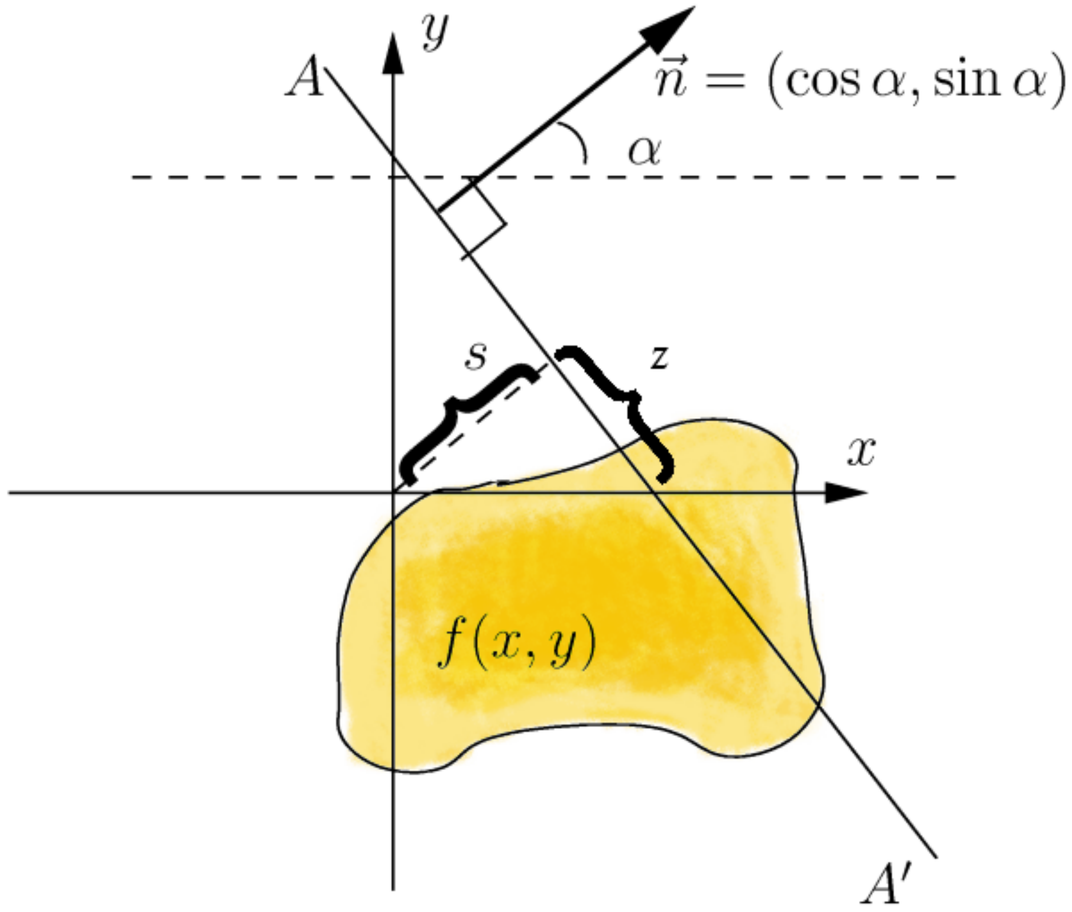
Definition of the Radon Transform

The parametrization of any straight line L with respect to arc length z can always be written:

$$(x(z), y(z)) = \Big((z \sin\alpha + s\cos\alpha), (-z\cos\alpha + s\sin\alpha) \Big)$$

where s is the distance of L from the origin and a is the angle the normal vector to L makes with the x axis. Then, the Radon transform:

$$\begin{aligned} Rf(\alpha, s) &= \int_{-\infty}^{\infty} f(x(z), y(z))\, dz \\ &= \int_{-\infty}^{\infty} f\big((z\sin\alpha + s\cos\alpha), (-z\cos\alpha + s\sin\alpha)\big)\, dz \end{aligned}$$

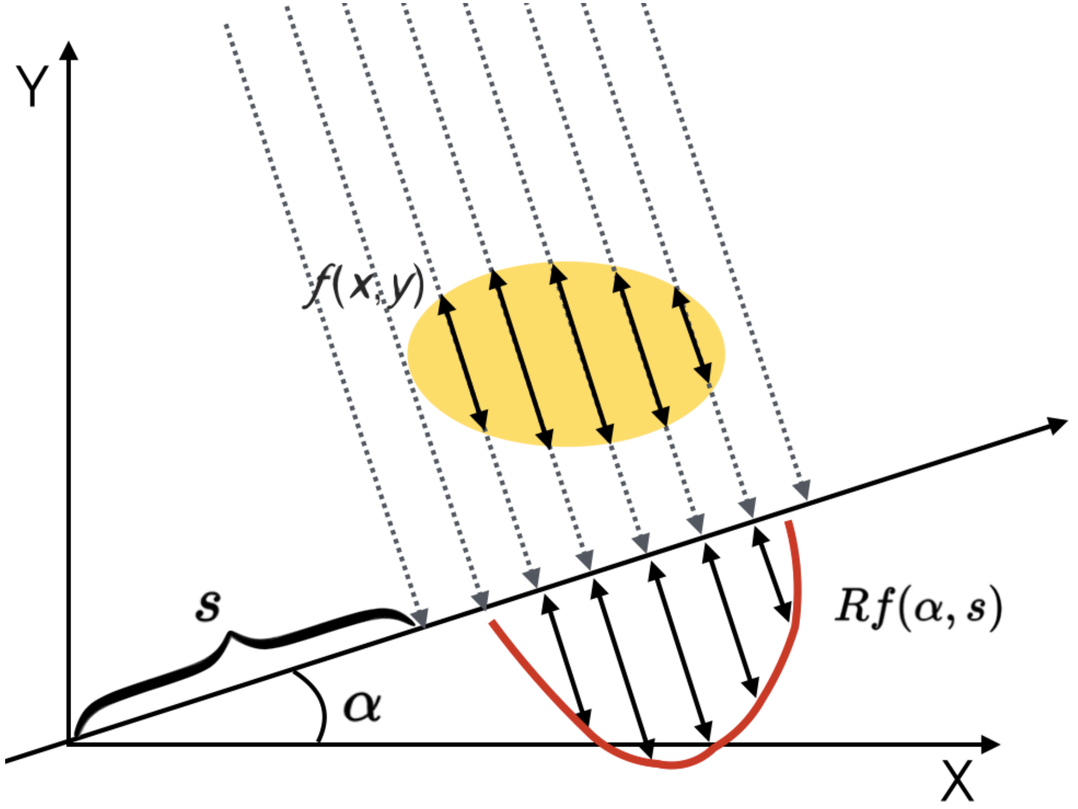


The Radon Transform. Interpretation.

Consider f(x,y) as a transparency function of an object in 2D.

Let light rays go in direction perpendicular to the vector $(\cos\alpha, \sin\alpha)$.

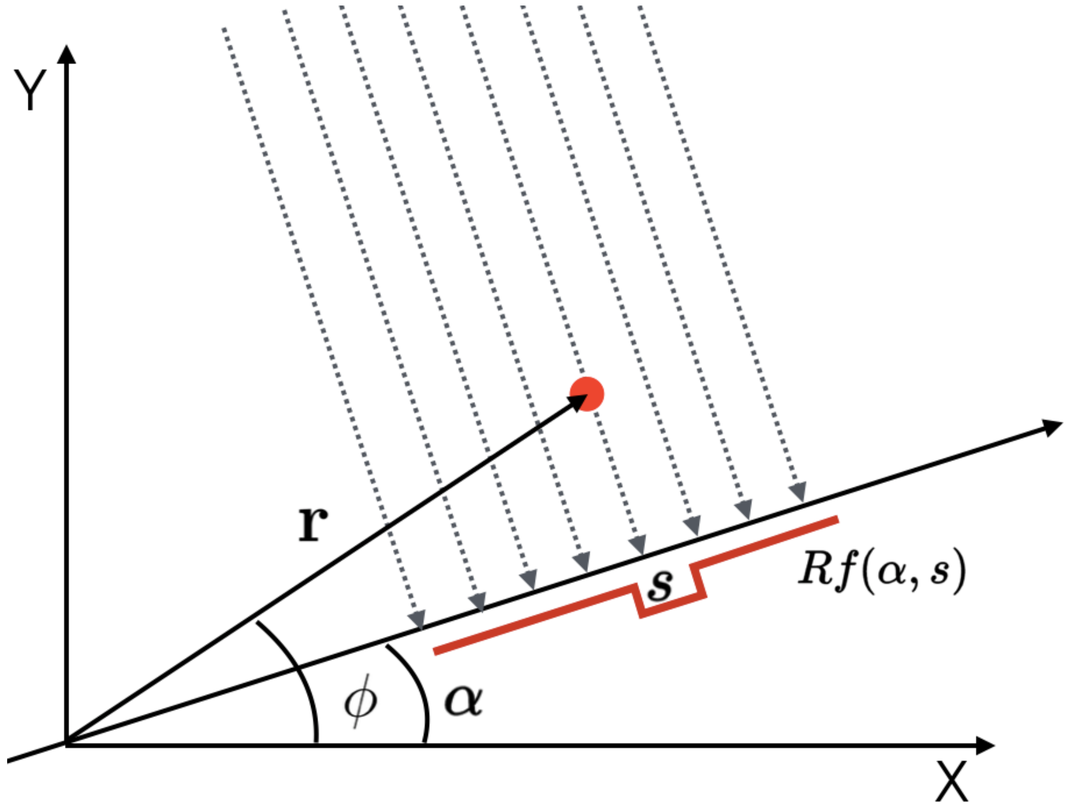
Then the Radon Transformation $Rf(\alpha, s)$ can be interpreted as intensity distribution of the light passed through the object.



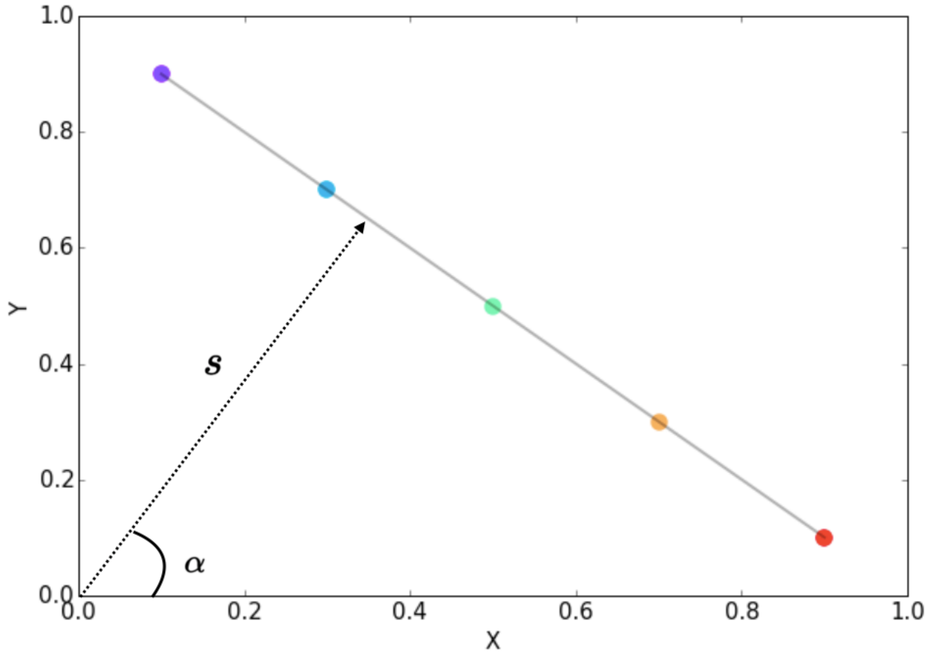
The Radon Transform

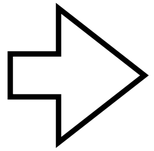
Consider a point with coordinates (x_0, y_0). Then, the Radon transformation represented as

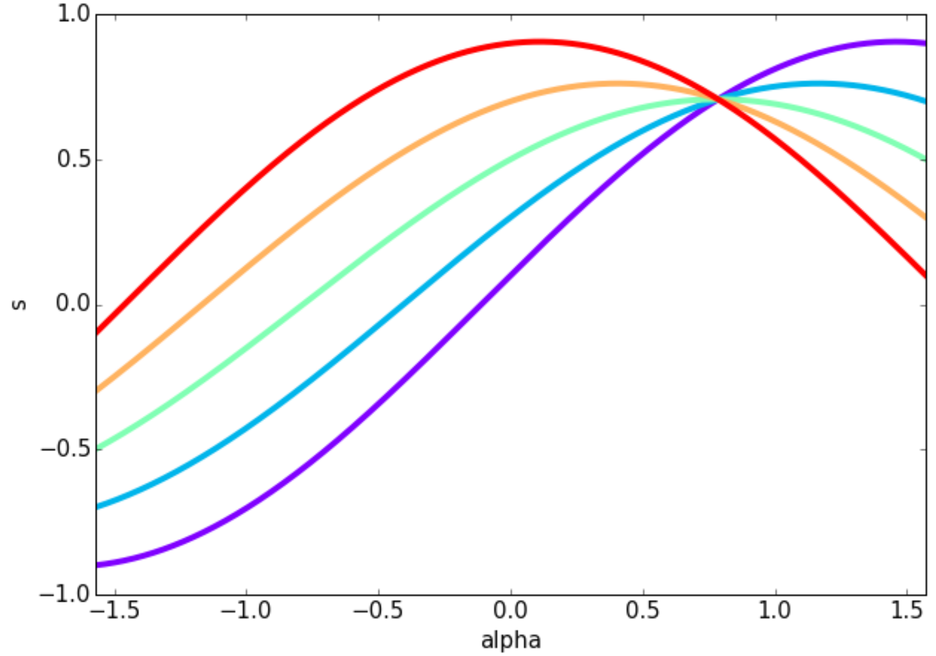
$$Rf(\alpha, s) = \begin{cases} 1 & \text{if } s = r\cos(\phi - \alpha) \\ 0 & \text{otherwise} \end{cases}$$



The Radon Transform

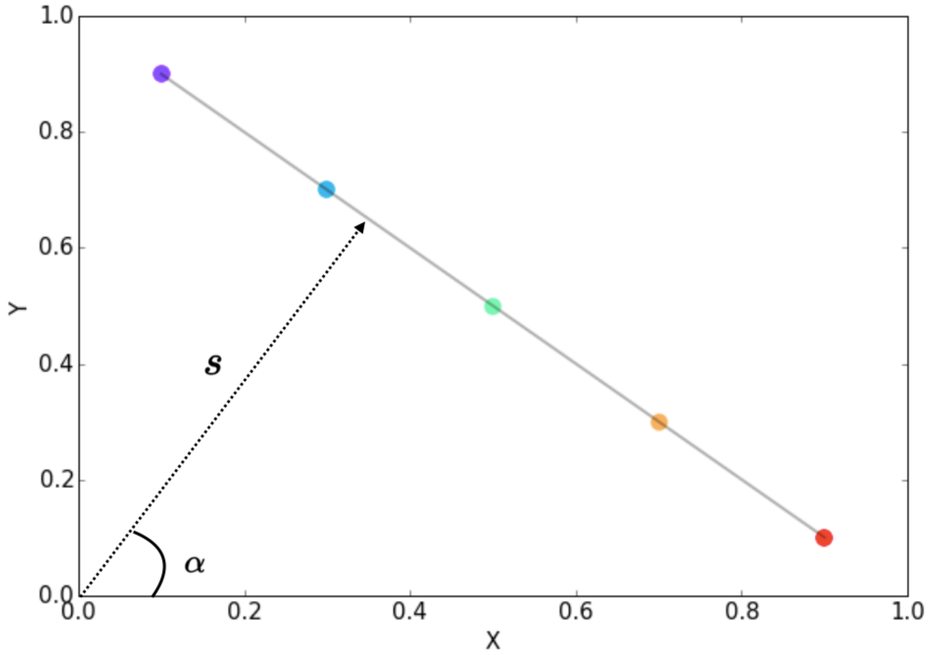


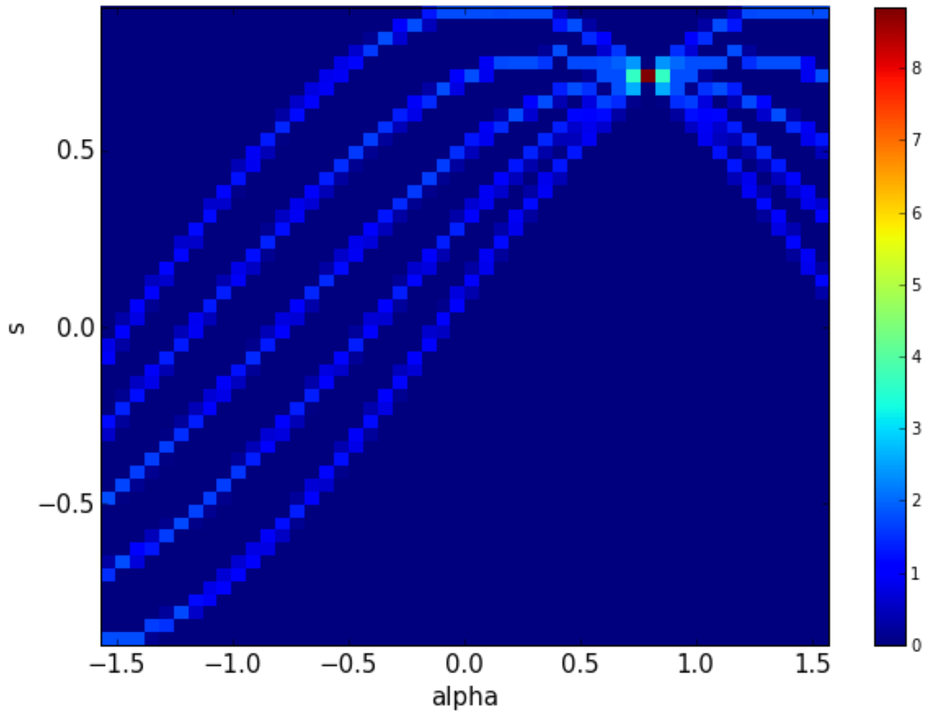




The Radon transform convert each point in (x,y) space to the cosine curve in (a, s) space. Intersection of the curves corresponds to the track parameters.

The Radon Transform





The point in (a, s) space with maximum value of the Rf(x,y) corresponds to the track's parameters.

The Hough Transform

Definition of the Hough Transform

The Hough transform is a method of finding instances within a certain class by voting procedure in a parameters space of the class.

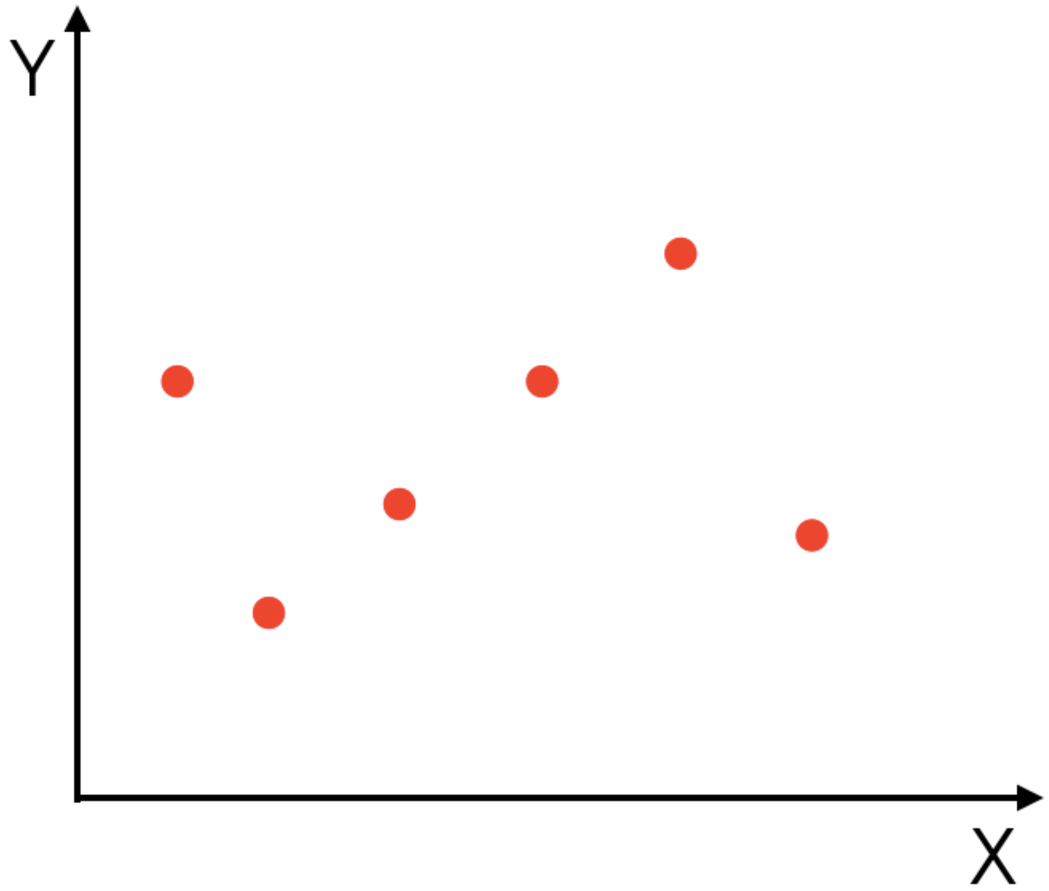
The Hough Transform

Consider a set of hits. Let search lines among the hits.

A line can be parametrized as

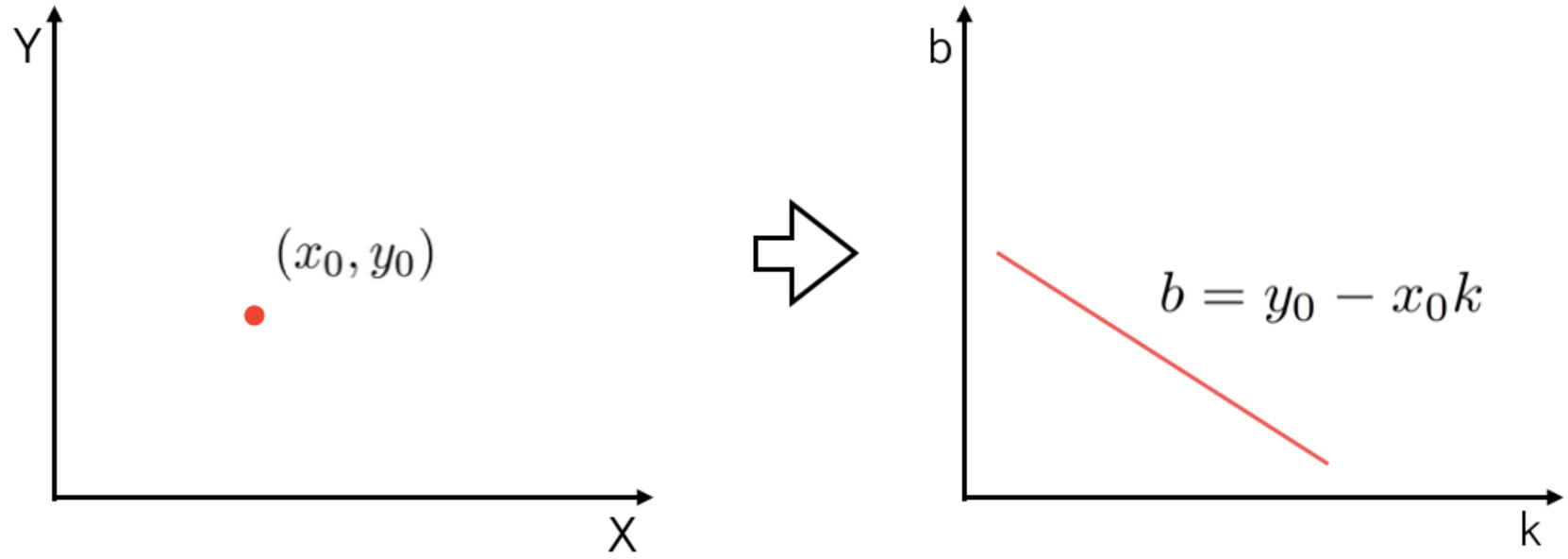
$$y = kx + b$$

where k, b are parameters of the line.



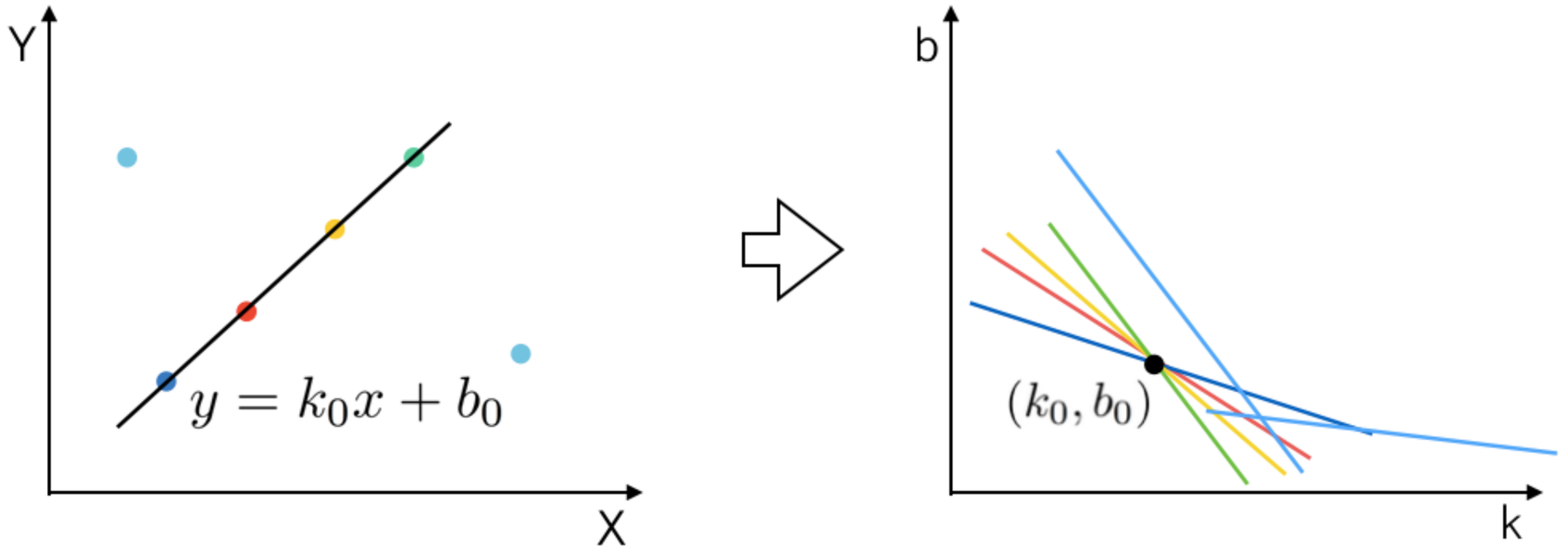
The Hough Transform

The Hough transform convert a point from (x,y) space to the curve in (k, b) space of the parameters. Each point of the curve in (k, b) space represents parameters of the line, that can go trough the point in (x, y) space.



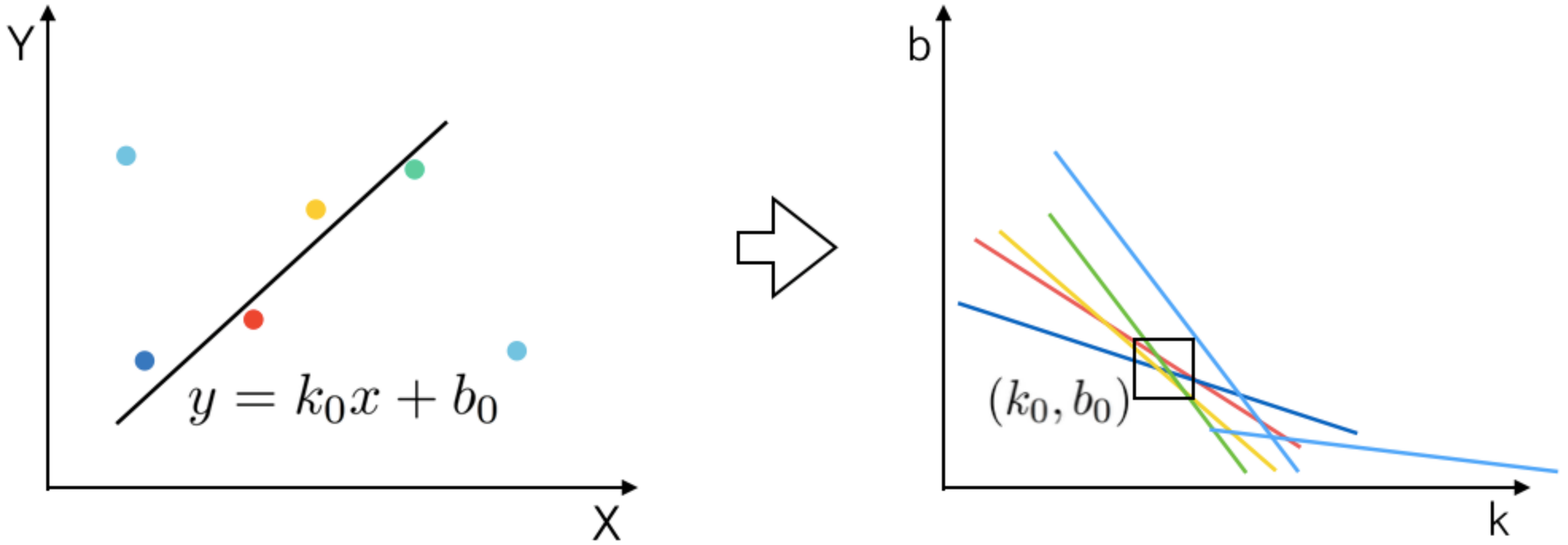
The Hough Transform

Since set of hits lie on the straight line, the curves in the parameter space should intersect in one point. This point corresponds the line's parameters.



The Hough Transform

Due to noise in coordinates of the hits, the optimal parameters of the track are in a cell in (k, b) space where the largest number of curves pass through.



The Hough Transform

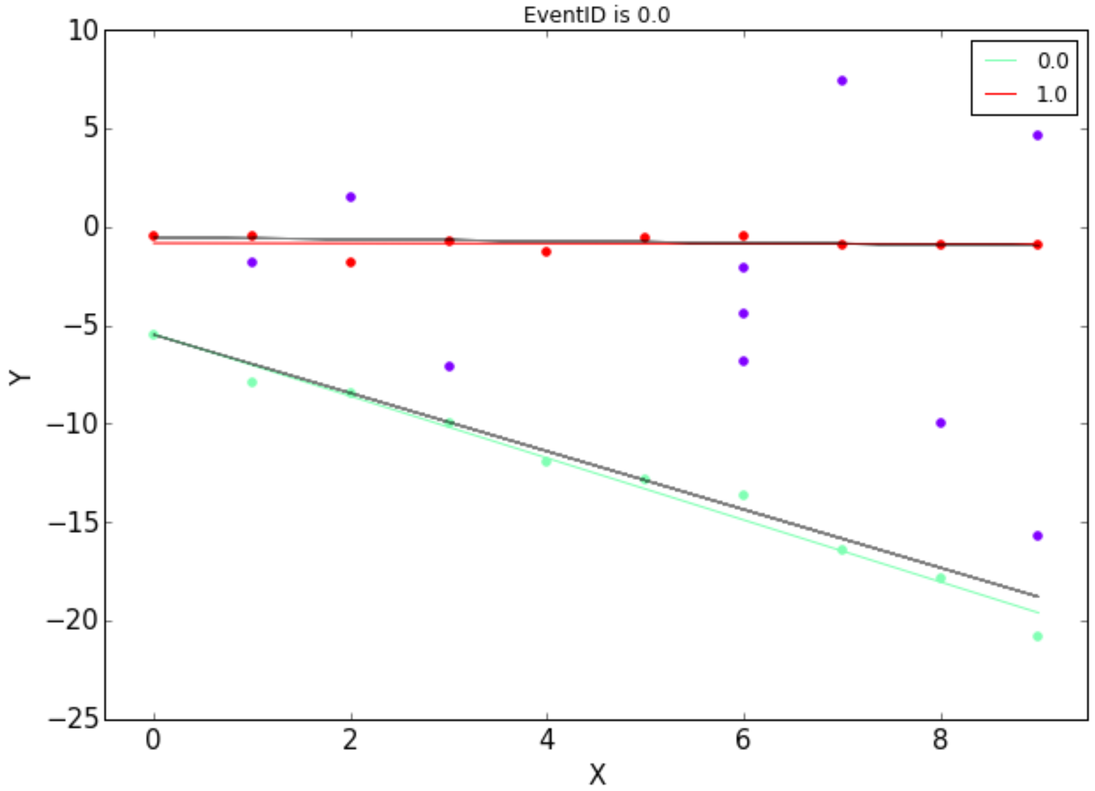
Algorithm:

1. Transform each hit using the Hough transform (or Radon transform).

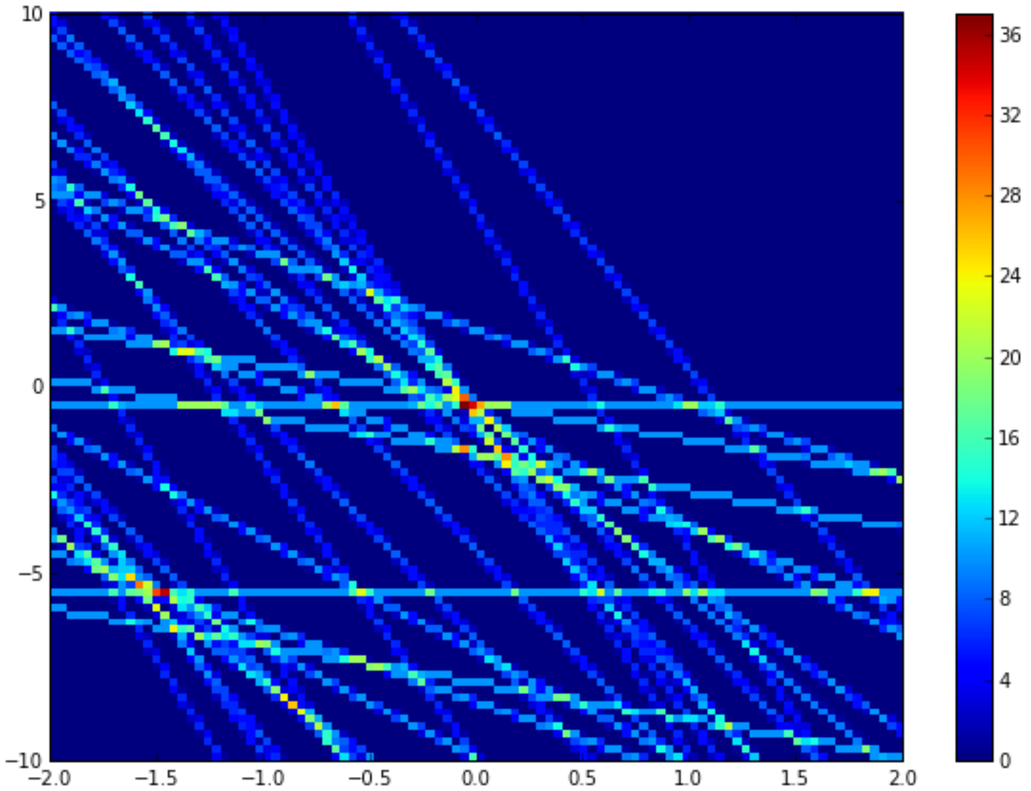
2. Histogram the curves of the transformed hits.

3. Tracks parameters correspond to the bins with >N entries.

The Hough Transform. Demonstration.



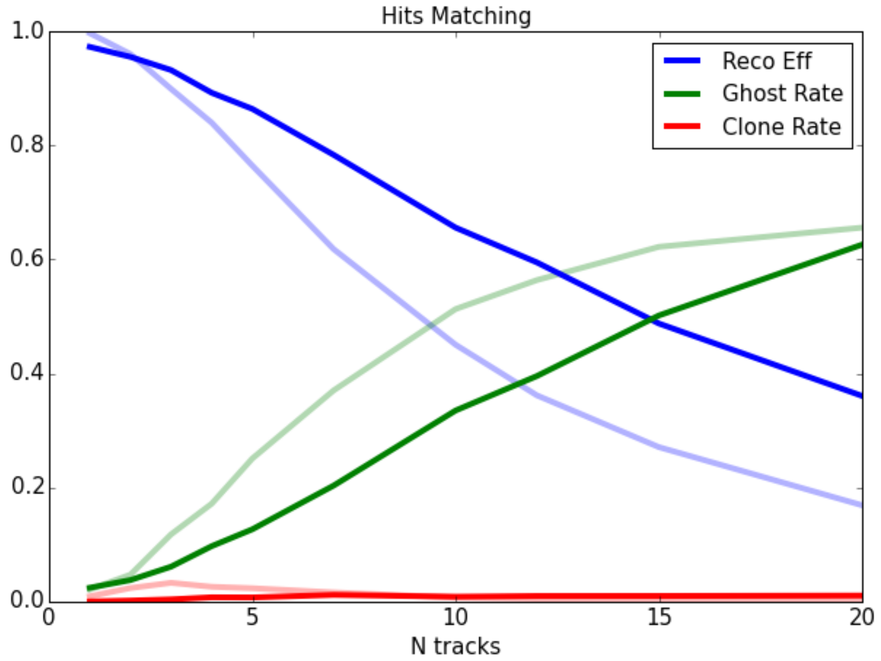
Tracks in hits space

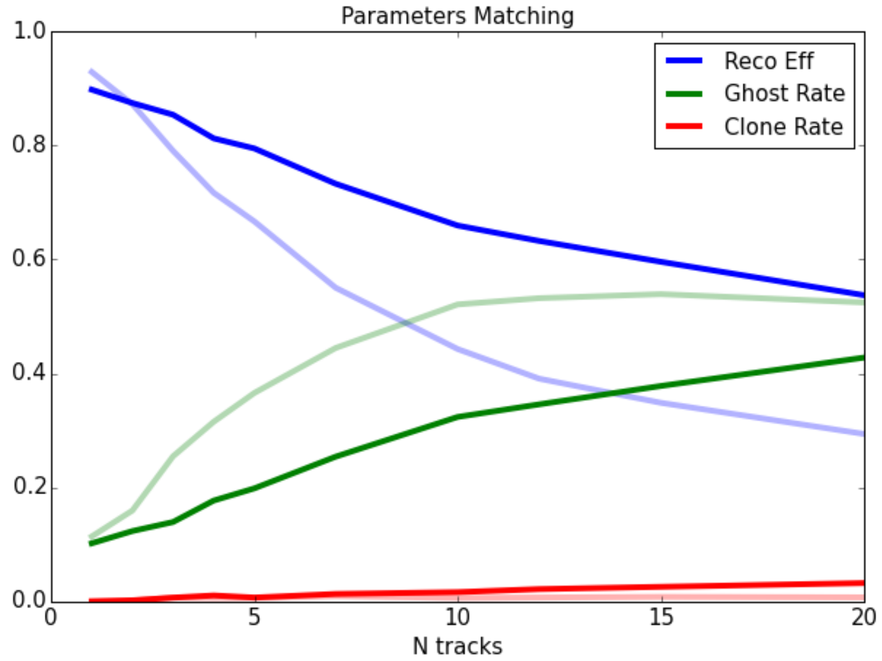


Hough transform of the hits

The Hough Transform. Demonstration.

Quality metrics for the Hough Transform and for the Simple Template Matching (transparent):





The Hough Transform demonstrates better reconstruction efficiency and ghost rate, especially for the large number of tracks.

The Hough Transform

Advantages of the method:

- Works well with several tracks
- Works with higher dimensions
- Can be used for circle tracks

Limitations of the method:

- Difficult to find maxima in parameters space
- Complex track shape leads to the increasing dimensionality of the parameter space

Neural Networks

The Hopfield NN

The Hopfield NN

Structure of the Hopfield network:

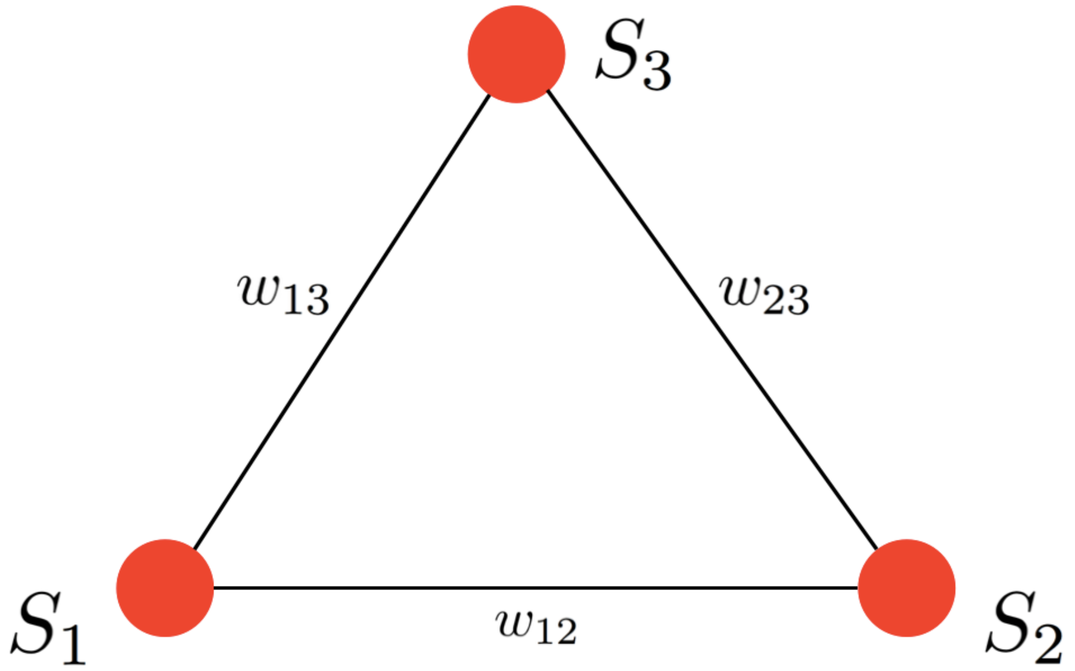
1. Each neuron has only two possible states S_i .

2. Every pair of the neurons have a symmetric connection with weight w_{ij} .

3. The state of a neuron calculates as:

$$S_i = \Theta(\sum_j w_{ij} S_j - \theta_i)$$

where Θ is activation function, that takes zero value for negative arguments, and one otherwise; θ_i is threshold value.



The Hopfield NN

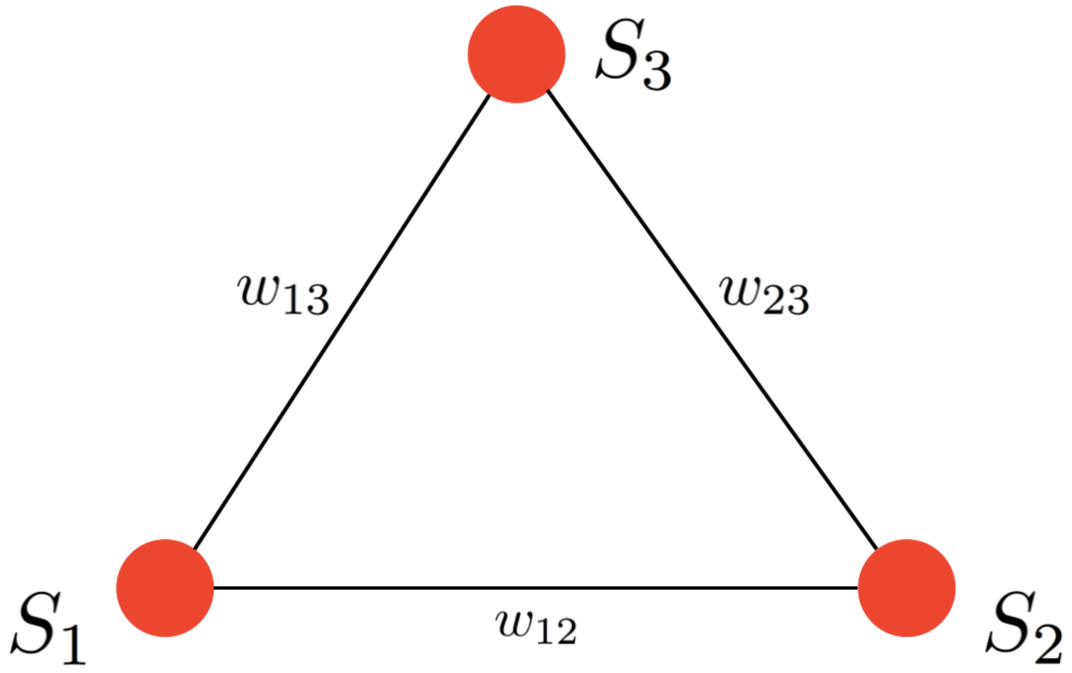
Training:

1. Set init states of the neurons.
2. Update the states of the each neuron using the rule:

$$S_i = \Theta(\sum_j w_{ij} S_j - \theta_i)$$

3. Repeat the step 2 while the states are not stable.

The states update can be done by one neuron at a time (asynchronous), or by all units at the same time(synchronous).



The Hopfield NN

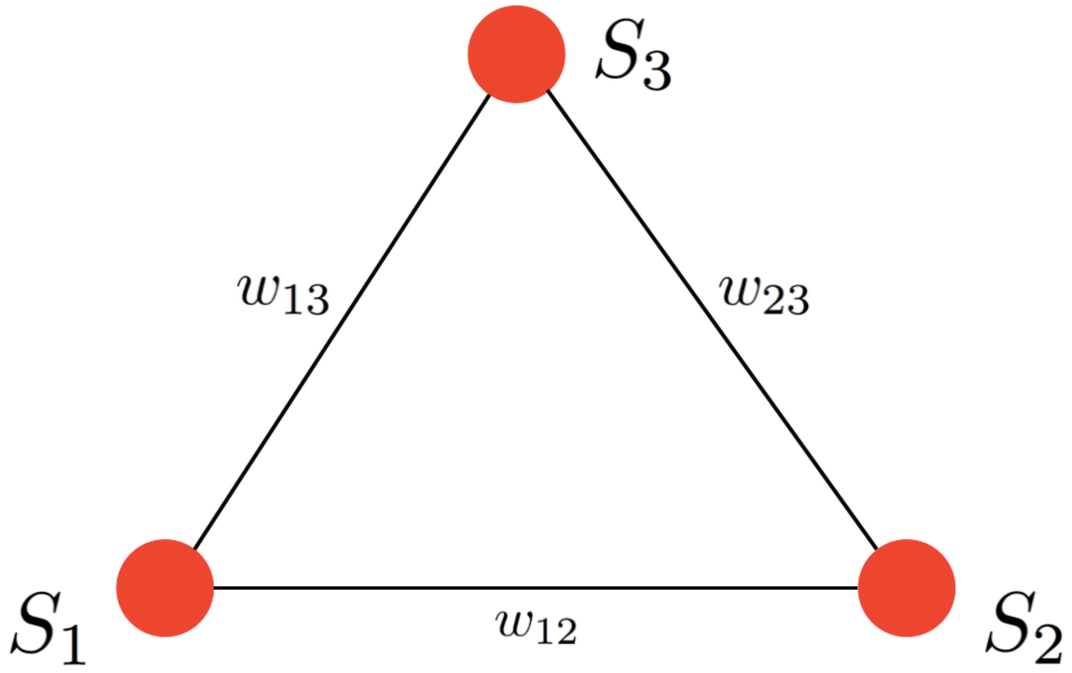
Hopfield's theorem:

The energy function

$$E = -\frac{1}{2}\sum_{i,j} w_{ij} S_i S_j + \sum_i \theta_i S_i$$

of a Hopfield network has local minima corresponding to the network stability points.

All neural network methods of tracks recognition are based on this theorem.



The Denby-Peterson Method

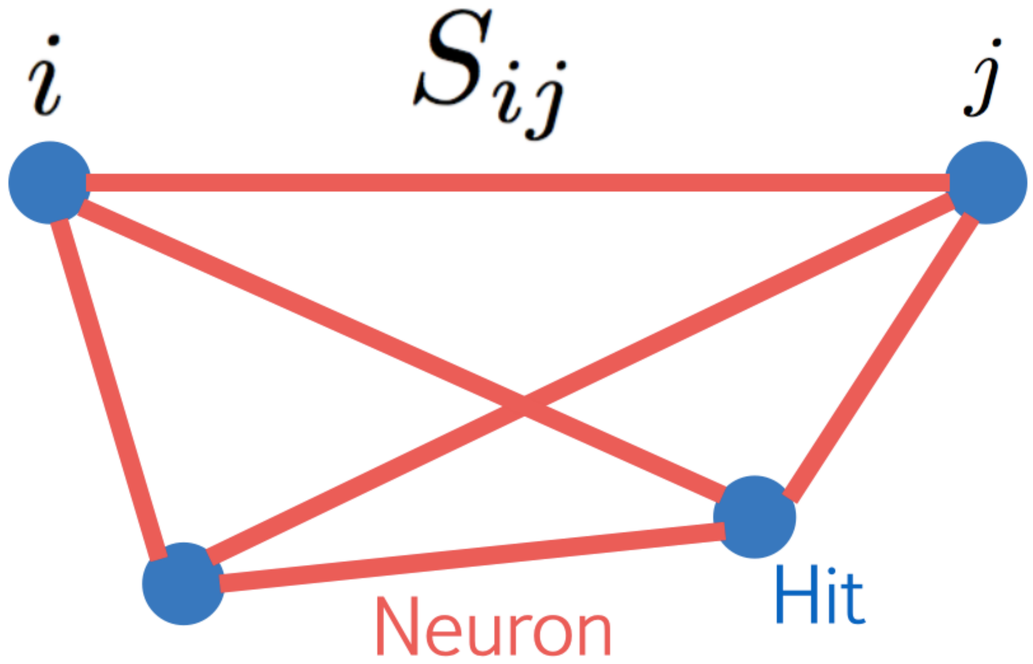
The Denby-Peterson Method

A network structure:

1. Each pair of hits is connected.
2. A connection represents one Hopfiel neuron.
3. Each neuron has two different types:

S_{ij} = 1 if two hits belong to the same track

S_{ij} = 0 if two hits belong to the different tracks

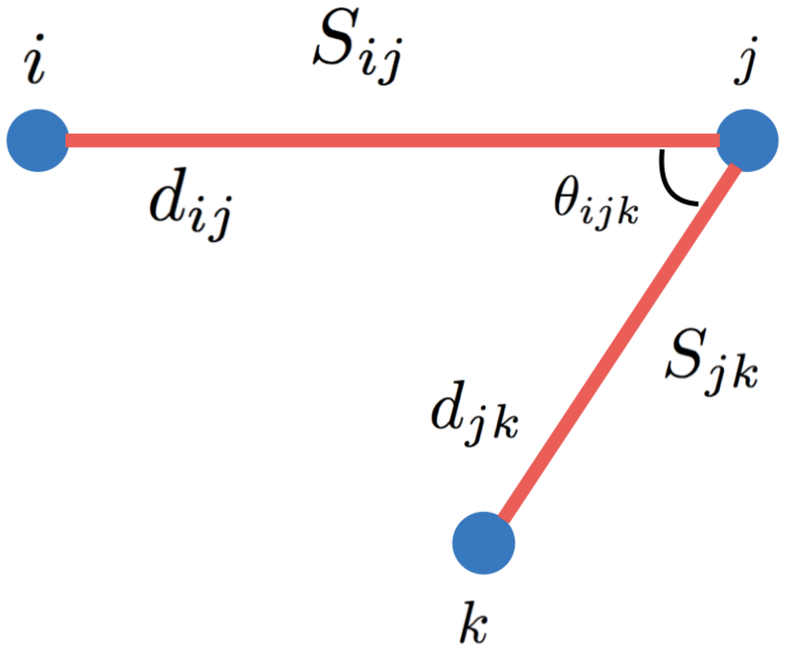


The Denby-Peterson Method

Energy between two neurons can be written:

$$E_{ijk} = -\frac{-\cos^m(\theta_{ijk})}{d_{ij}+d_{jk}} S_{ij} S_{jk}$$

Th idea of the expression is that neurons in true tracks are parallel. The energy has minima for the neurons from the same track.



The Denby-Peterson Method

$$E = -\frac{1}{2}\sum \frac{-\cos^m \theta_{ijk}}{d_{ij}+d_{jk}} S_{ij}S_{jk} + \frac{\alpha}{2}\left(\sum_{l\neq j} S_{ij}S_{il} + \sum_{k\neq i} S_{ij}S_{kj}\right) + \frac{\delta}{2}\left(\sum S_{kl} - N\right)^2$$

'cost function':

- θ_{ijl}: angle between neurons ij and jl

d_{ij}: length of neuron ij

penalty function against bifurcations

penalty function to balance number of active neurons against number of hits

Alpha, delta and m are adjustable parameters

The Denby-Peterson Method

A simple track recognition algorithm:

1. Create neurons. Initialize their states with some values.

2. Cut of neurons on their length to reduce their number.

3. Minimize the energy with respect to the rule:

$$s_{ij} = \frac{1}{2}\left(1 + \text{sign}\left(-\frac{\partial E}{\partial s_{ij}}\right)\right)$$

4. Iterate the step 3 while the states are not converged.

5. Find tracks as groups of connected neurons in active states.

The Denby-Peterson Method

Minimization with discrete states is not very stable. The solution is to define continuous states and use the following update function:

$$v_{ij} = \frac{1}{2}(1 + tanh(-\frac{\delta E}{\delta v_{ij}}\frac{1}{T}))$$

where v is continuous states and temperature T is adjustable parameter.

The Denby-Peterson Method

A track recognition algorithm:

1. Create neurons. Initialize their states with some values.

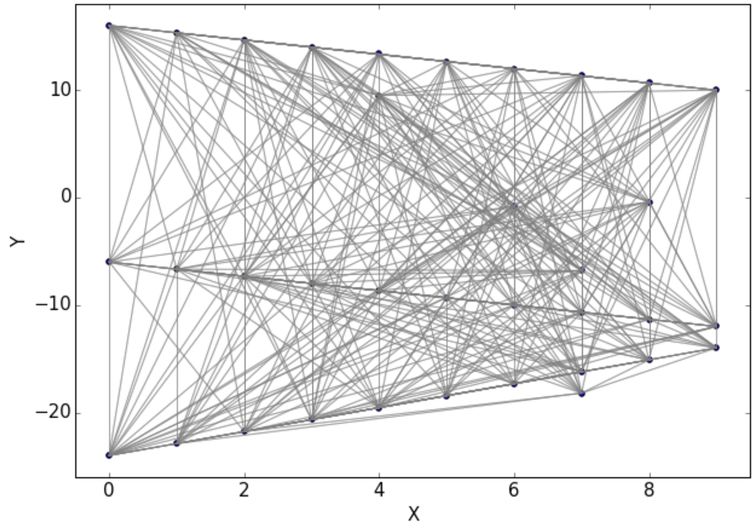
2. Cut of neurons on their length to reduce their number.

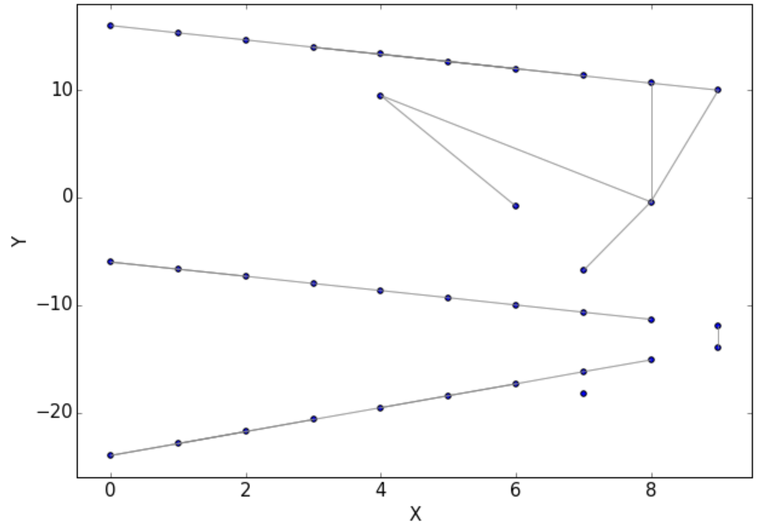
3. Calculate the new states for the all neurons using the previous equation.

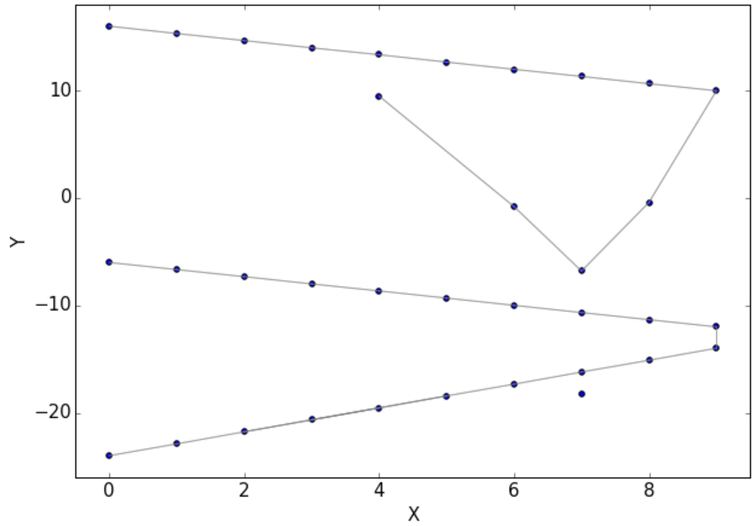
4. Iterate the step 3 while the states are not converged. The temperature between the iteration can be reduced.

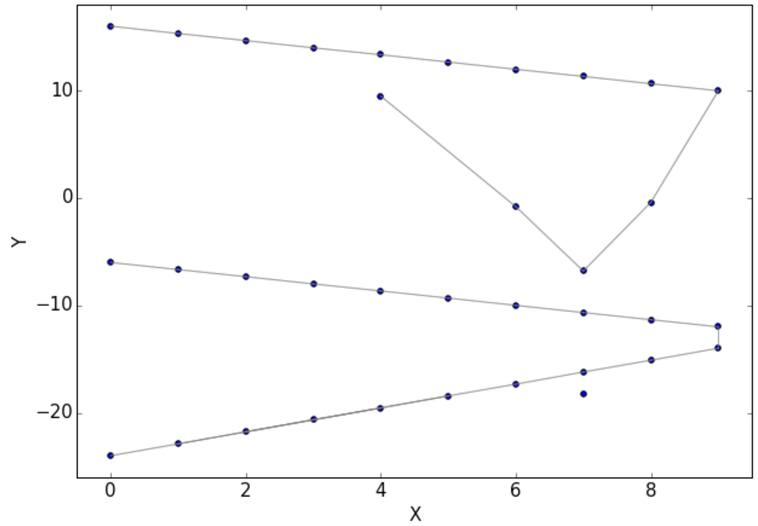
5. Find tracks as groups of connected neurons in active states.

The Denby-Peterson Method. Demonstration.

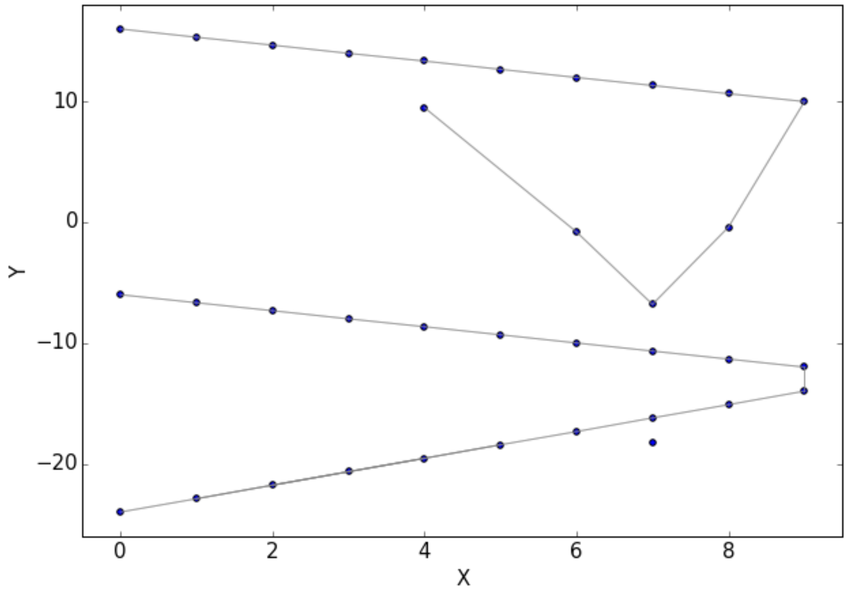




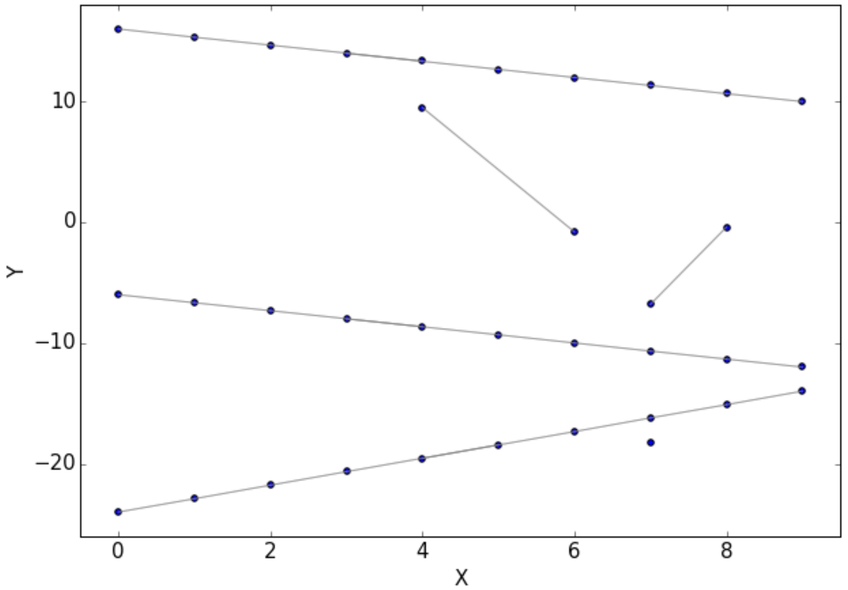




The Denby-Peterson Method. Demonstration.



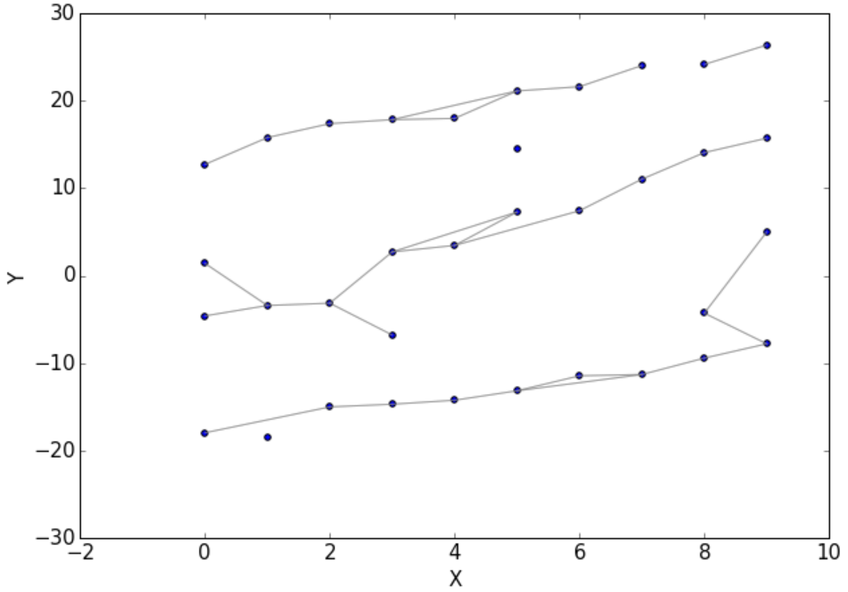
Original network



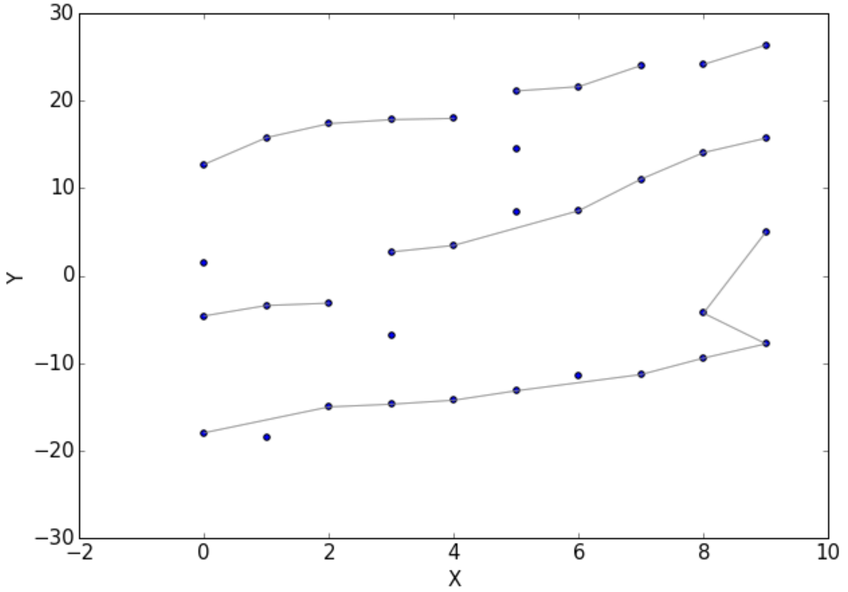
Network after cutting neurons

Cut neurons so, that each hit has no more than 2 neurons with angle between them near to 180 degree.

The Denby-Peterson Method. Demonstration.



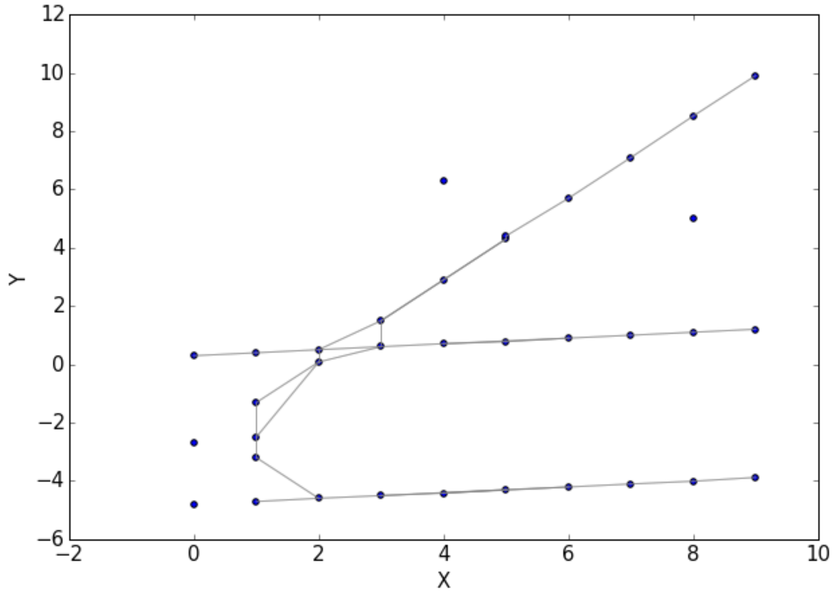
Original network



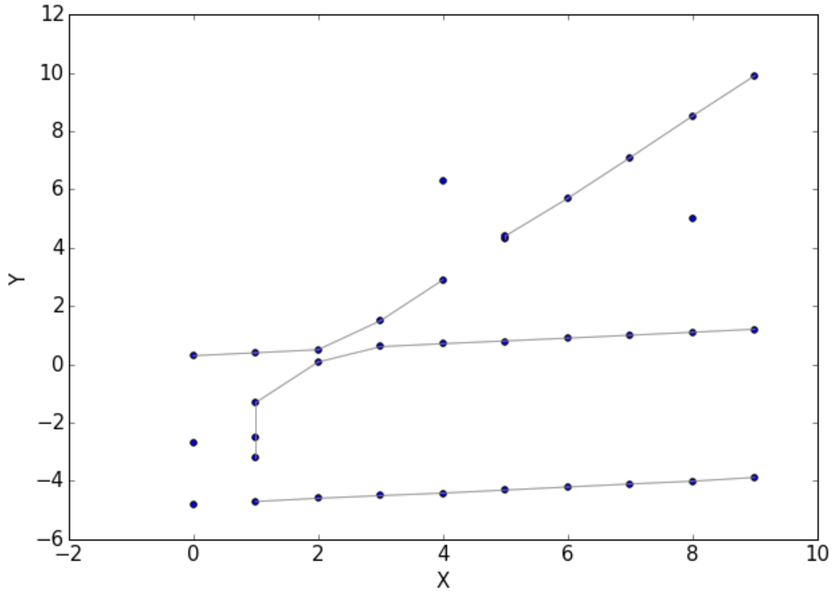
Network after cutting neurons

Noise can lead to the more complicated patterns and to the tracks segmentation.

The Denby-Peterson Method. Demonstration.



Original network

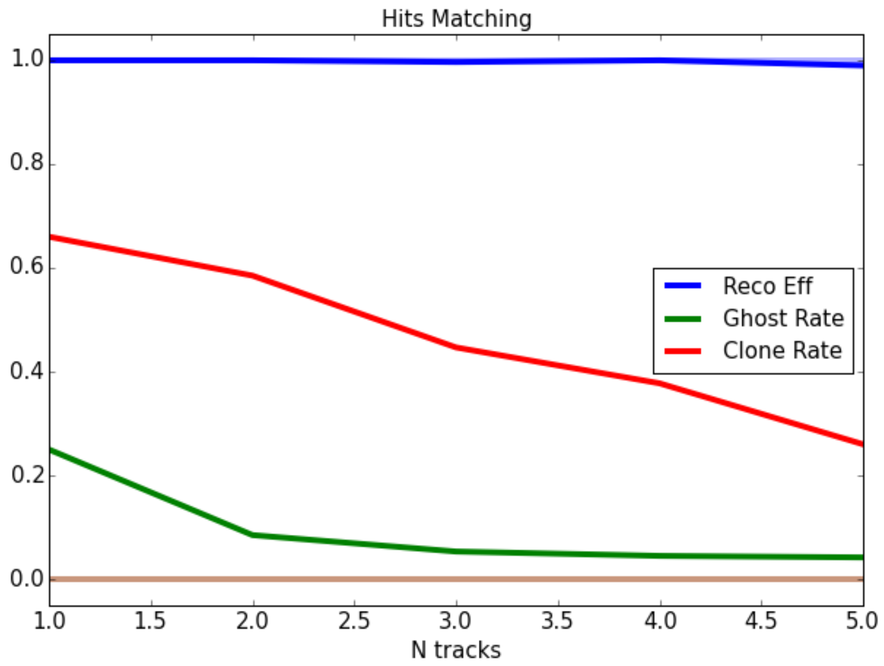


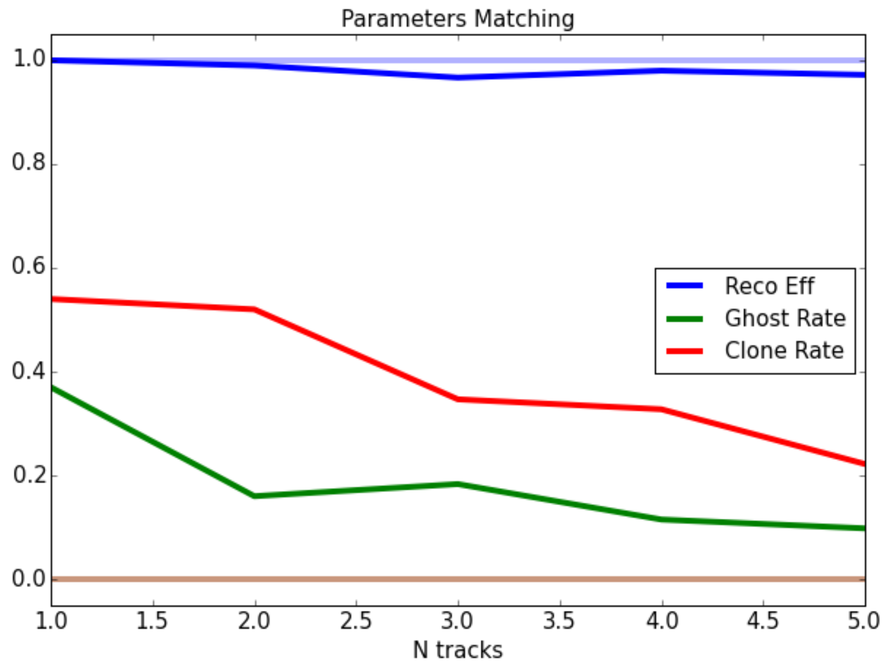
Network after cutting neurons

It is difficult to separate intersected tracks. It is needed more complicated separation algorithms.

The Denby-Peterson Method. Demonstration.

Quality metrics for the Denby-Peterson method and for the Simple Template Matching (transparent) for non intersecting tracks:

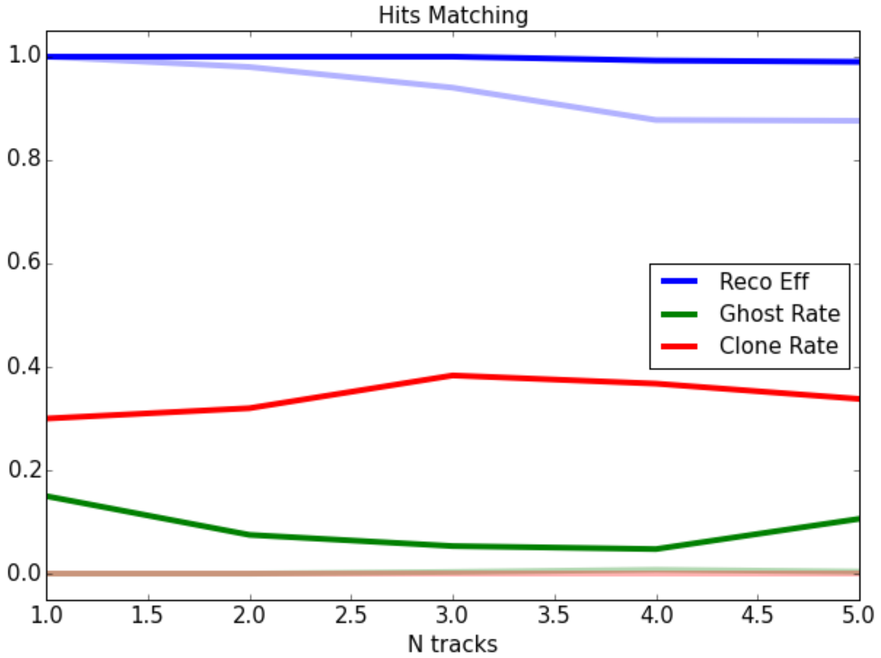


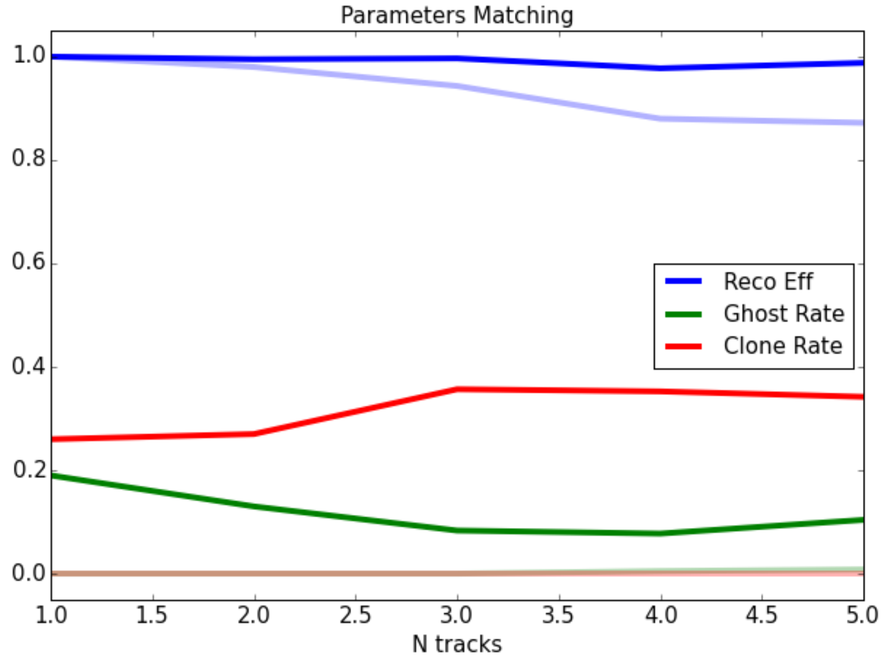


The Denby-Peterson method demonstrates good reconstruction efficiency, but large clone and ghost rates. This is due to the tracks segmentations and segments with noise hits.

The Denby-Peterson Method. Demonstration.

Quality metrics for the Denby-Peterson method and for the Simple Template Matching (transparent) for intersecting tracks:





The Denby-Peterson method demonstrates better reconstruction efficiency, but large clone rate. This is due to the tracks segmentations and intersections. Segments with noise hits lead to the hight clone rate.

The Denby-Peterson Method

Advantages of the method:

- Works without actual knowledge of a track model
- Indifferent about the shape of the track
- Robust against noise

Limitations of the method:

- The optimization is more difficult
- Requires large computational resources

Cellular Automaton

Cellular Automaton

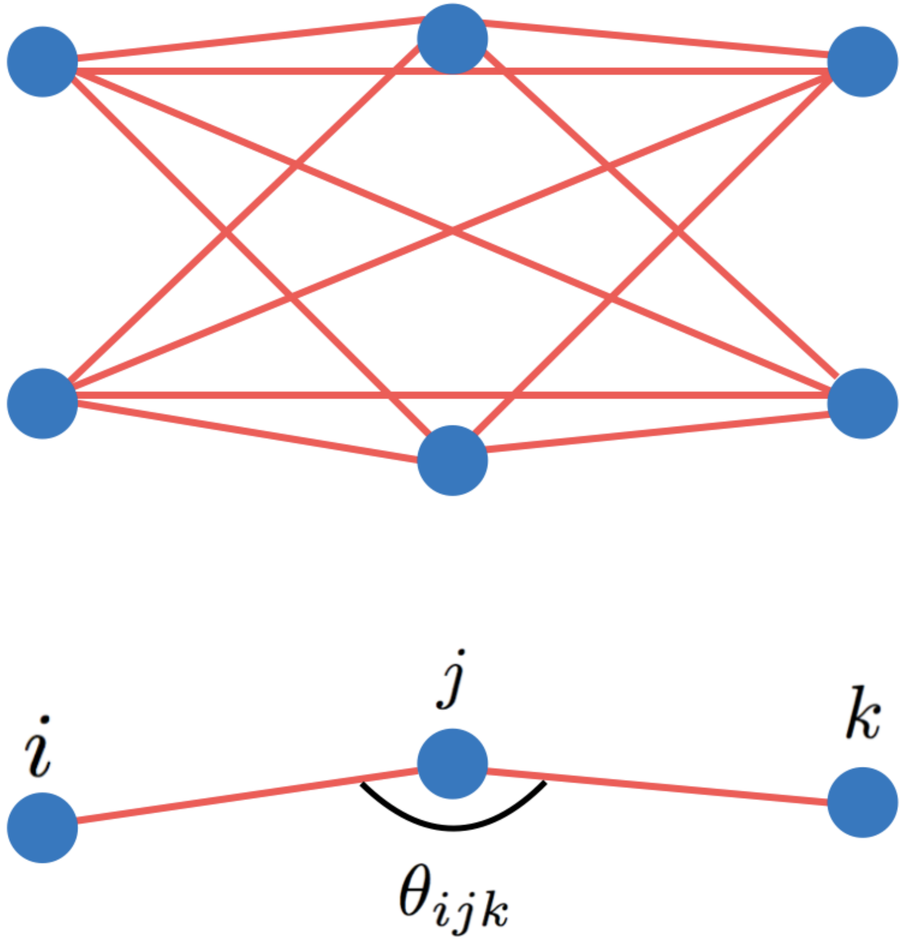
Similar to Denby-Peterson, but simpler.

1. Neurons connect hits from different layers.

2. Each neuron has integer-valued state S_{ij} , initialized at 1.

3. Two neuron belong to the same track if

$$\theta_{ijk} > \theta^{min}$$



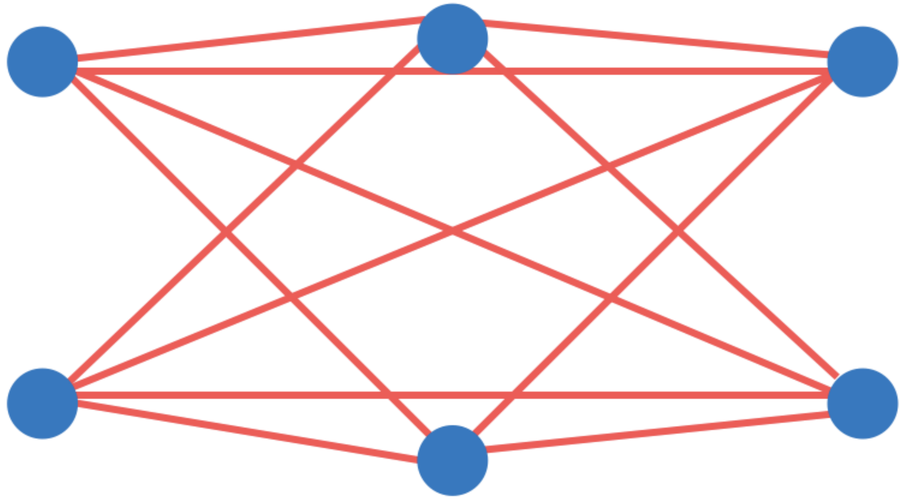
Cellular Automaton

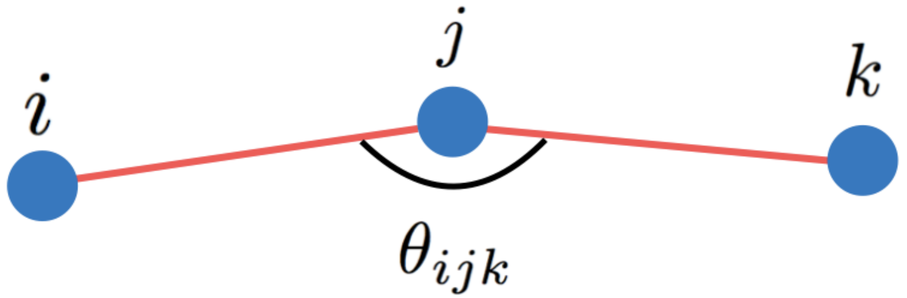
4. The states are updated simultaneously by looking at neighbours in layer before it:

$$S_{jk} = max\{S_{ij} | \theta_{ijk} > \theta^{min}\} + 1$$

5. Repeat step 4 until the cells are stable.

6. Select tracks by starting form a neuron with the highest state value.





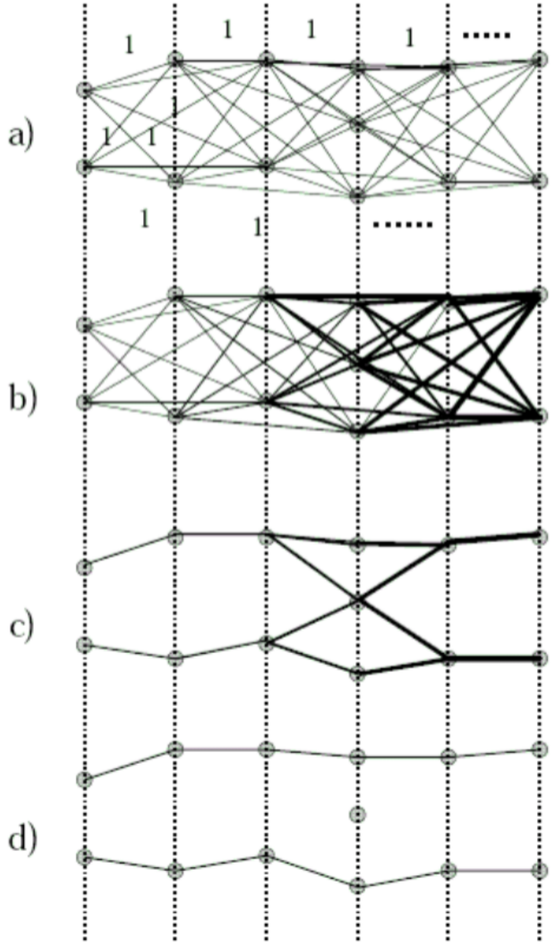
Cellular Automaton. Demonstration.

a) Initialization.

b) End of the evolution. State value indicates by the thickness.

c) Selection of the longest tracks started from the neurons with the heights states.

d) Additional selection to remove overlaps.



Cellular Automaton

Advantages compared with the Denby-Peterson method:

- Less neurons
- Simpler optimization
- Require less computational resources

Rotor Models

Rotor Models

In rotor model each hit has its own rotor. Energy function has the following form:

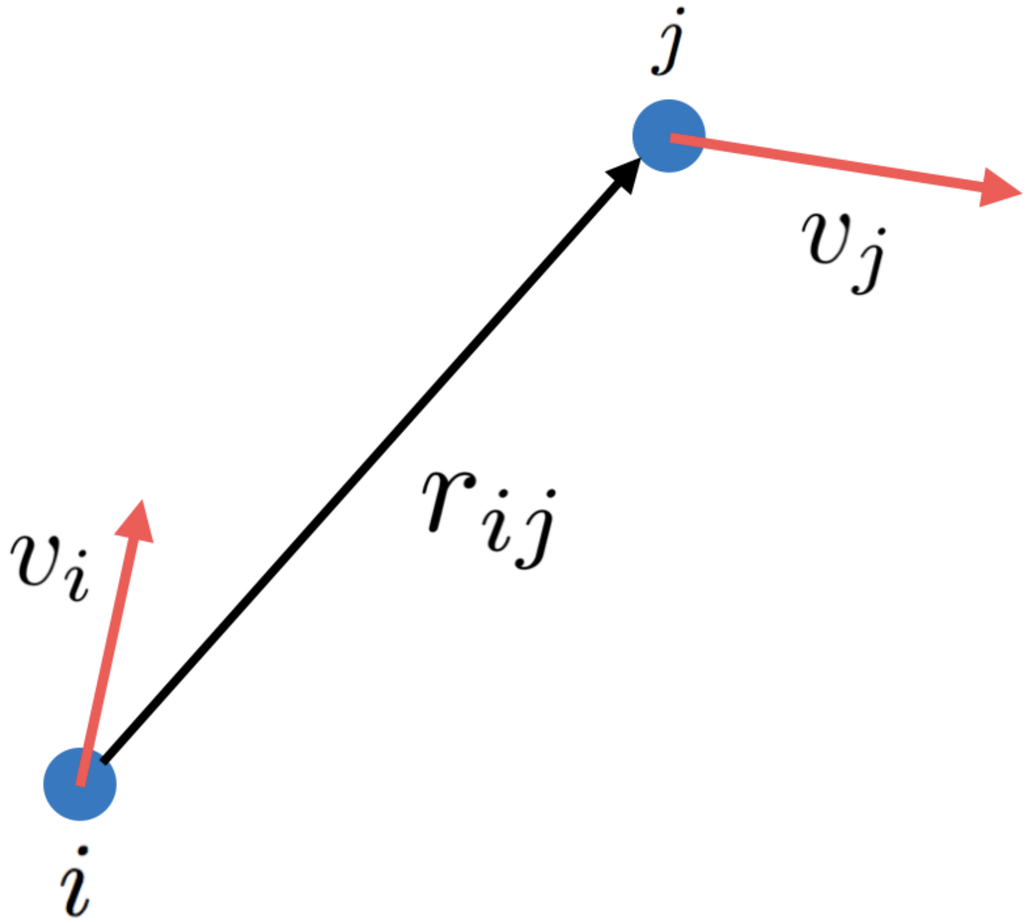
$$E = -\frac{1}{2}\sum_{ij}\frac{1}{|r_{ij}|^m}v_i v_j - \frac{1}{2}\alpha\sum_{ij}\frac{1}{|r_{ij}|^m}(v_i r_{ij})^2$$

The first term forces neighbouring rotors to be close to each other.

The second term is in charge of the same between rotors and track-segments.

The rotors update rule:

$$v_i = tanh(-\frac{\delta E}{\delta v_i}\frac{1}{T})$$



Rotor Models

The model for the circle tracks can be simplified:

$$\dot{E} = -\frac{1}{2}\sum_{ij} v_i W_{ij} v_j$$

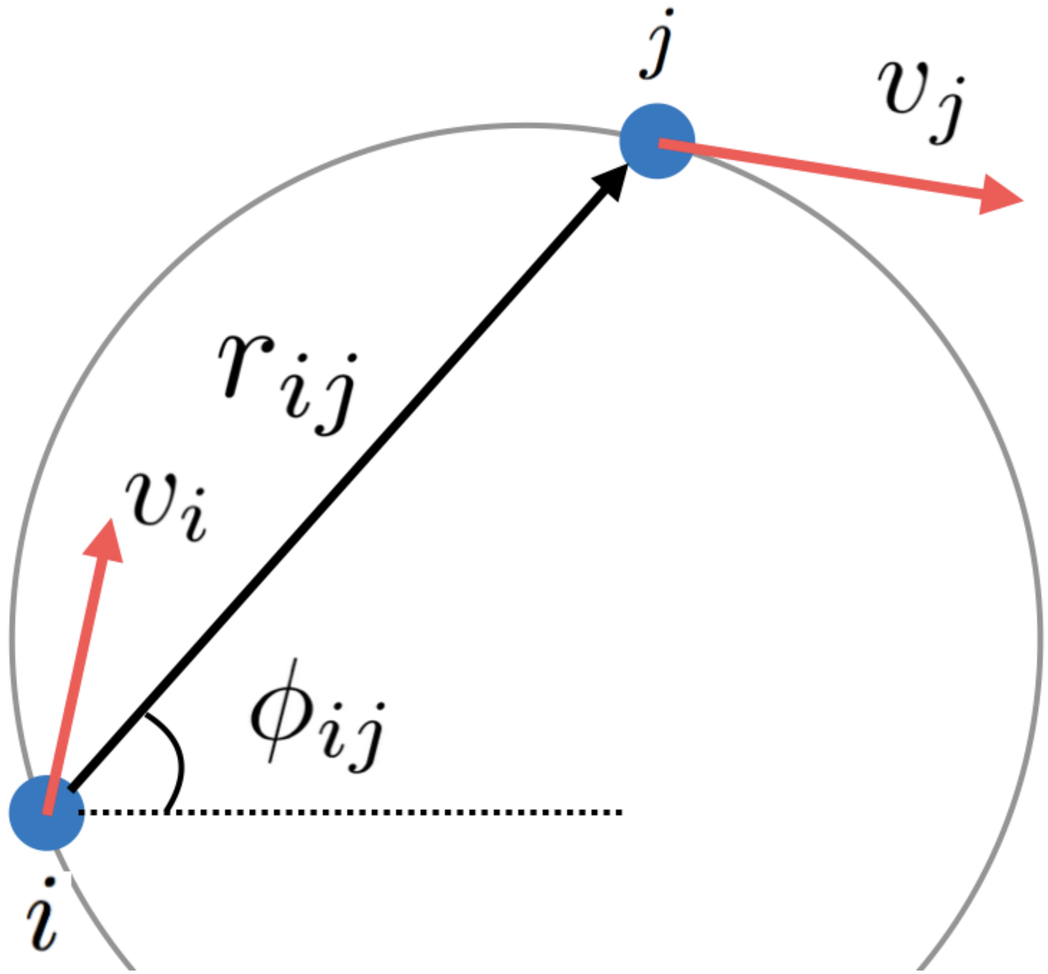
where W:

$$W_{ij} = \begin{pmatrix} \cos 2\phi_{ij} & \sin 2\phi_{ij} \\ \sin 2\phi_{ij} & -\cos 2\phi_{ij} \end{pmatrix}$$

Then a rotor evolution can be written:

$$v_i = \tanh\left(\frac{\sum_j W_{ij} v_j}{T}\right)$$

where temperature T is adjustable parameter.



Rotor Models

A track recognition algorithm:

1. Create rotors. Initialize them with some values.

2. Update the rotors using the evolution expression from the previous slides.

3. Iterate the step 2 while the rotors are not converged. The temperature between the iteration can be reduced.

4. Find tracks as groups of codirectional rotors. The larger a rotor's module the higher it influences on other rotors.

Elastic Arms

〉 Elastic arms or deformable templates is method for the tracks recognition based on neural nets. The method combines stages of tracks recognition and fitting in one procedure.

〉 For example, the Denby-Peterson method finds something straight. Elastic arms method can find other templates (circles or other curves).

〉 Powerful method even in high tracks density case.

〉 Hard to optimize, requires a lot of computational resources.

More information: http://bit.ly/28LUPSy

Summary of Global Methods

1. Template Matching
   - Simple Template Matching
   - RANSAC
2. Transformation Methods
   - The Radon Transform
   - Hough transform
3. Neural Network Techniques
   - The Denby-Peterson method
   - Cellular Automaton
   - Rotor models of Hopfield networks
   - Elastic Arms

Tracks Recognition

Local Methods

Definition of the Local Methods

〉 Local methods (track following methods) reconstruct track hit by hit.

〉 The track following procedure starts from generating a track seeds, i.e. initial track candidates formed by just minimal set of hits.

〉 A parametric track model is needed for the extrapolation along the track and adding new hits to the track.

Local Methods

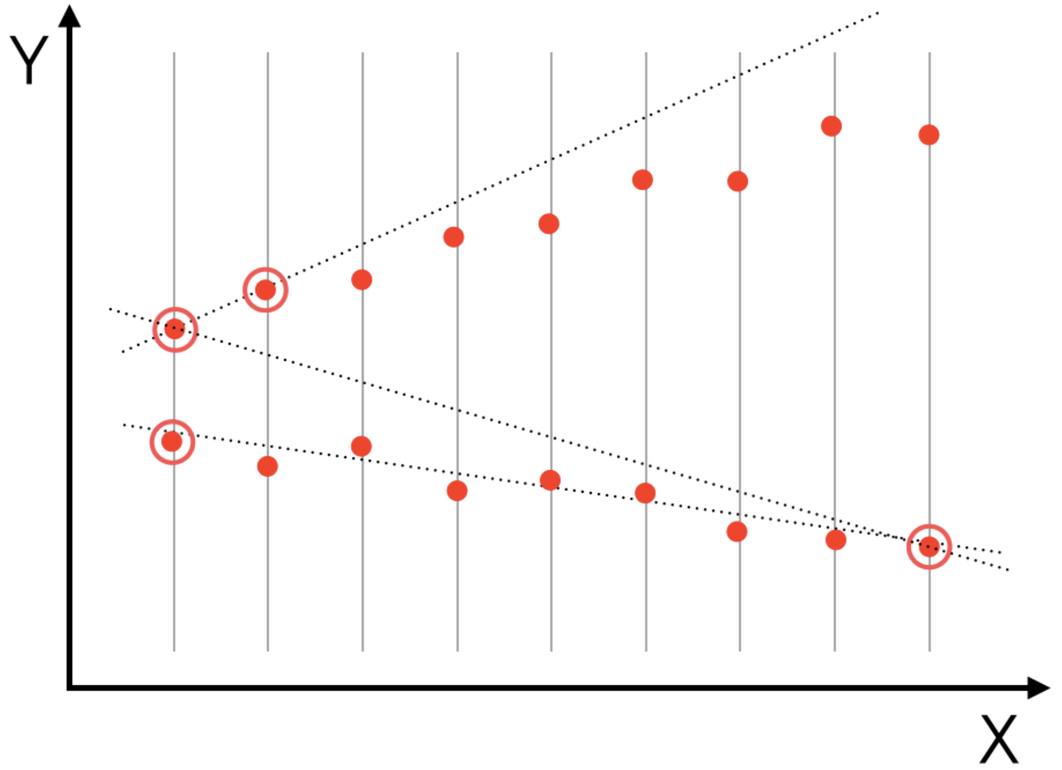
1. Track Seeds
2. Naive Track Following
3. The Combinatorial Filter
4. The Kalman Filter

Track Seeds

Track Seeds

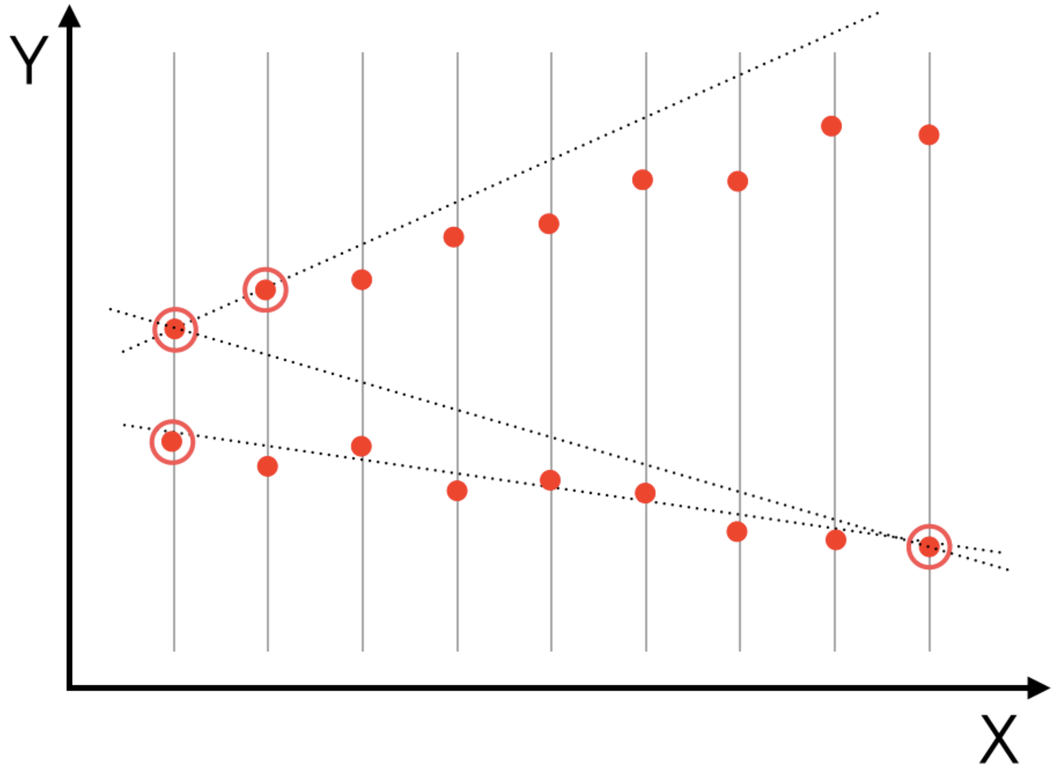
There are different way how seeds can be constructed.

The most suitable way is determined by the physics and geometry of the problem.



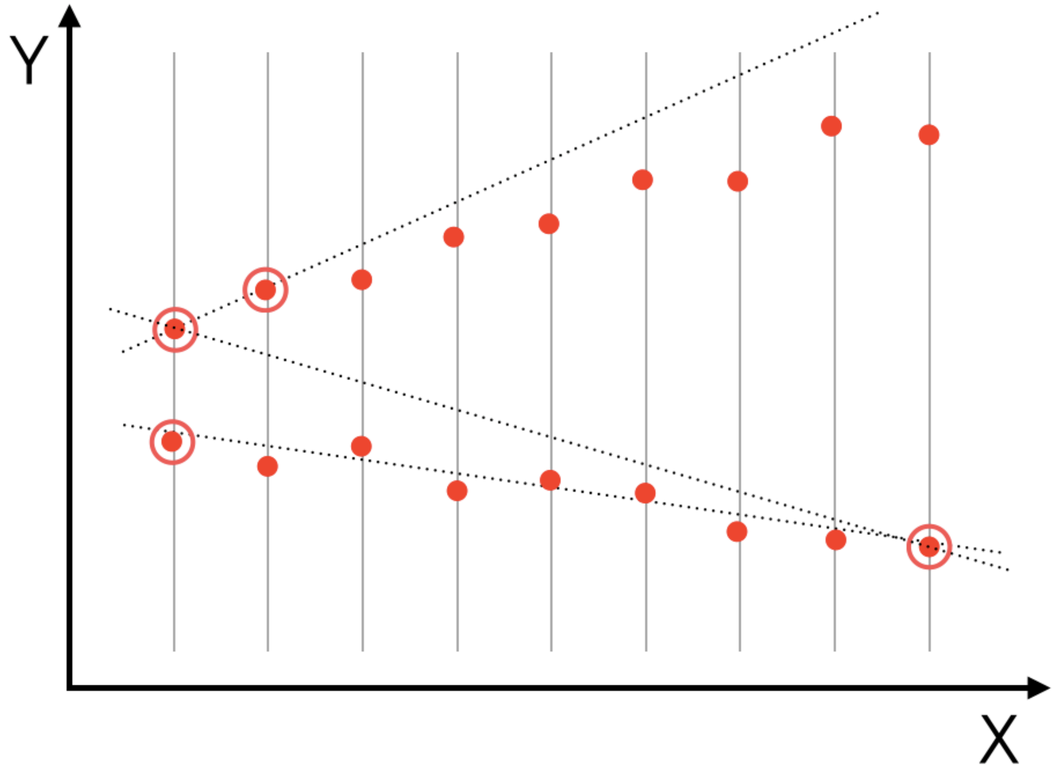
Track Seeds

Seeds which are constructed from the hits in closest layers generally have limited track's parameters precision. But the rate of fake seeds is relatively small, since most wrong combinations tend to obtain a steep slope that is incompatible with the relevant physical tracks and can be discarded immediately.



Track Seeds

Seeds which are constructed from the hits in distant layers generally have better track's parameters precision. But number of these seeds in much higher.

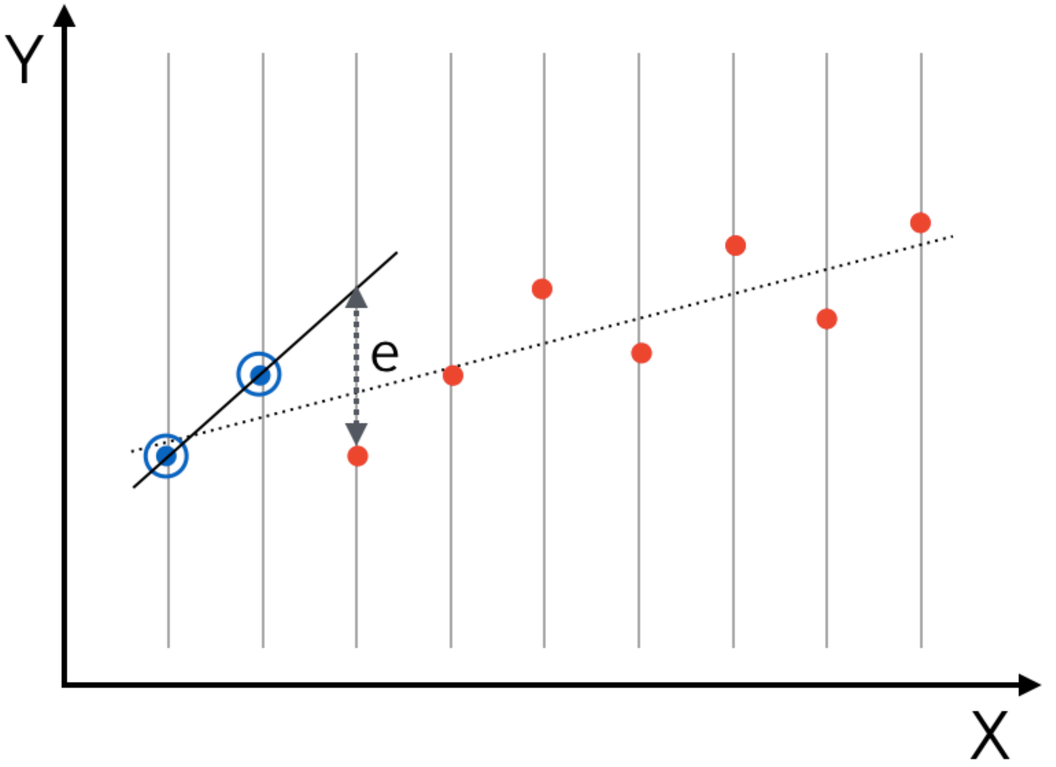


Naive Track Following

Naive Track Following

1. Start from a seed. Extrapolate the trajectory to the layer, where the next hit is expected.

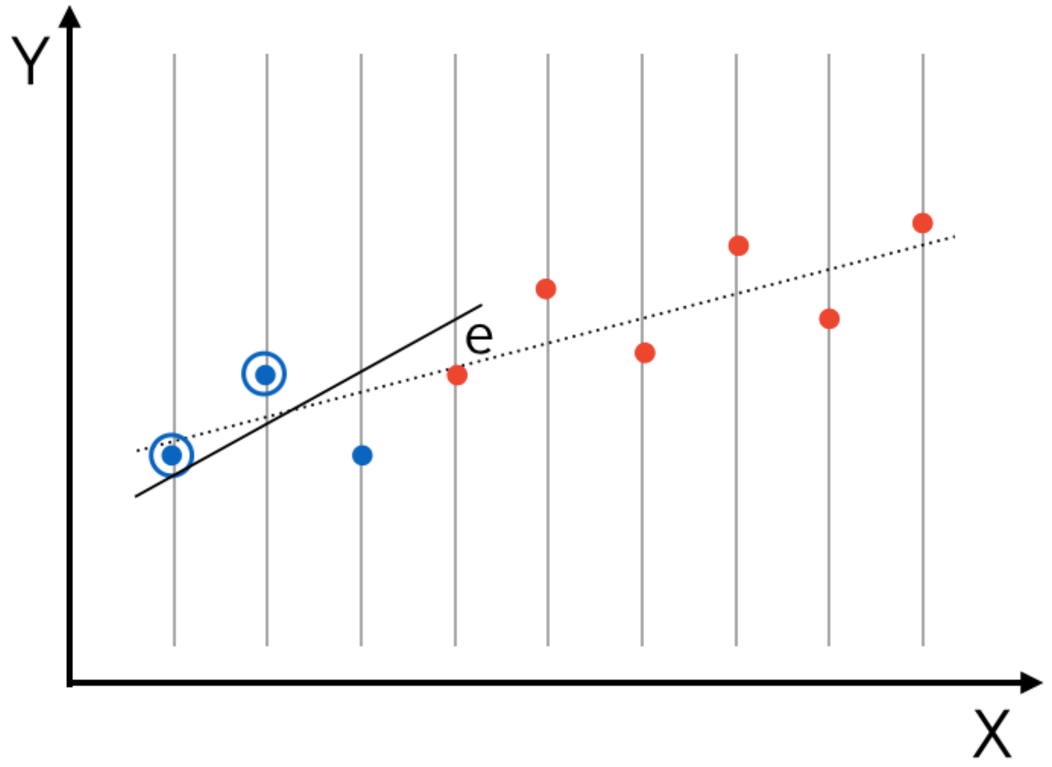
2. If suitable hit is found, add it to the track. If there are several hits, select the closest one to the extrapolated trajectory.



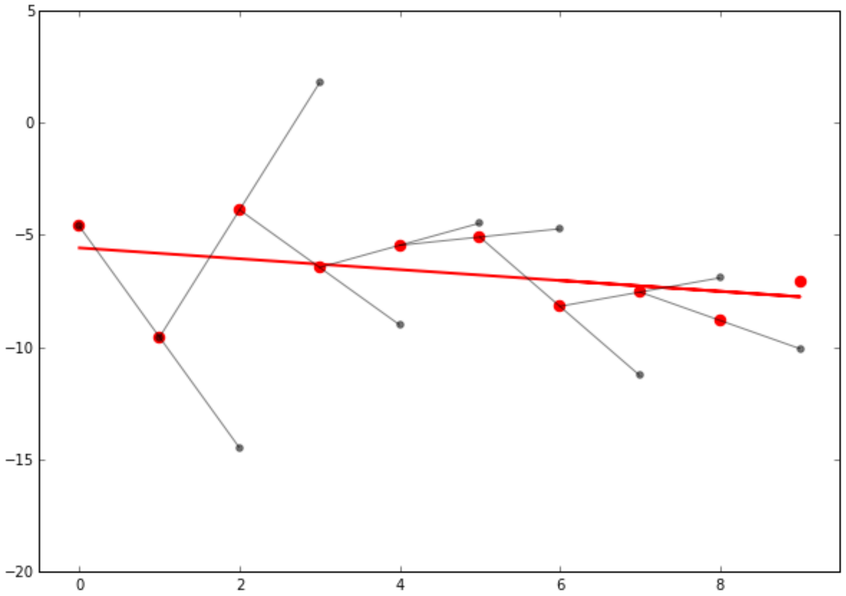
Naive Track Following

3. Extrapolate the trajectory to the next layer using all hits previously added to the track or just several last ones.

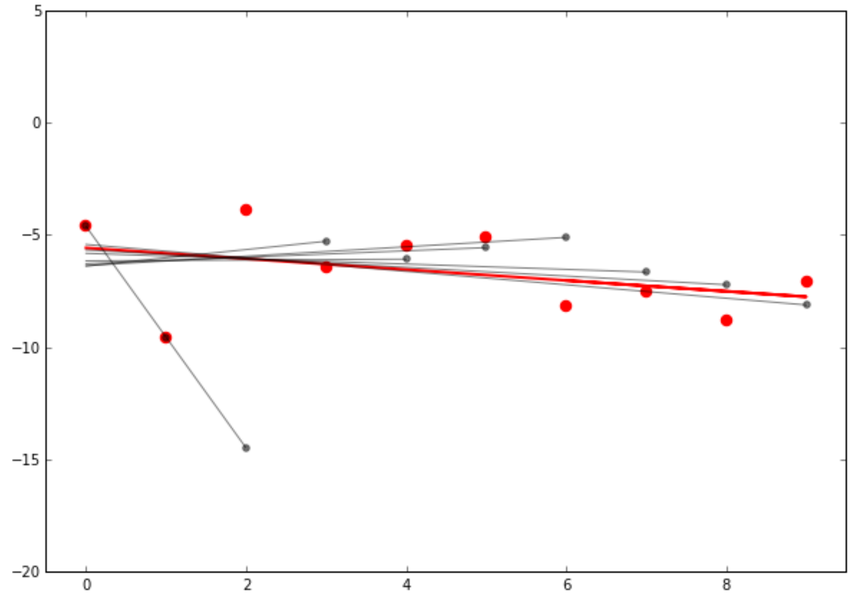
4. Continue the procedure until the end of the tracking area is reached, or suitable hits can not be found.



Naive Track Following. Demonstration.



Extrapolation based on two last hits.

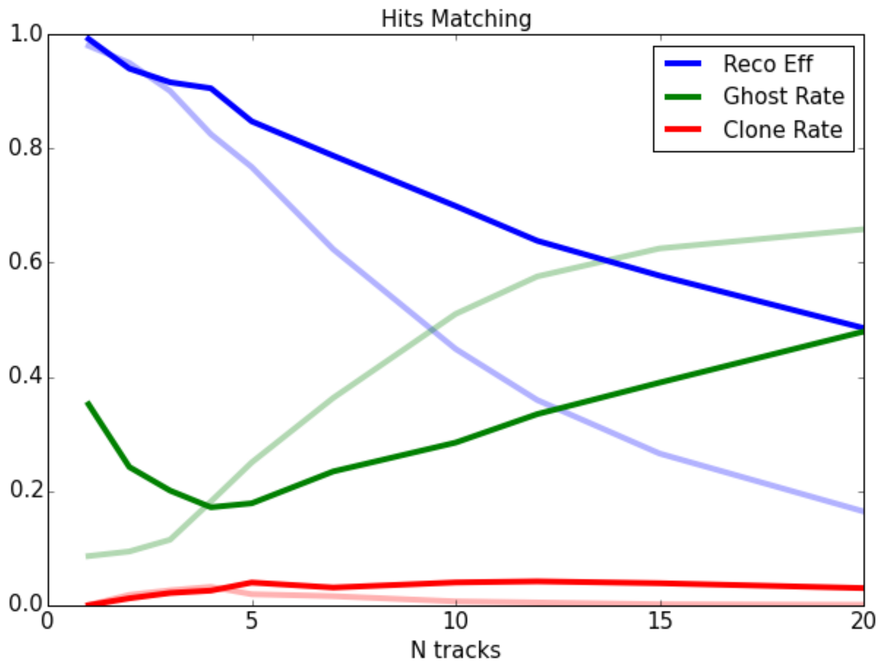


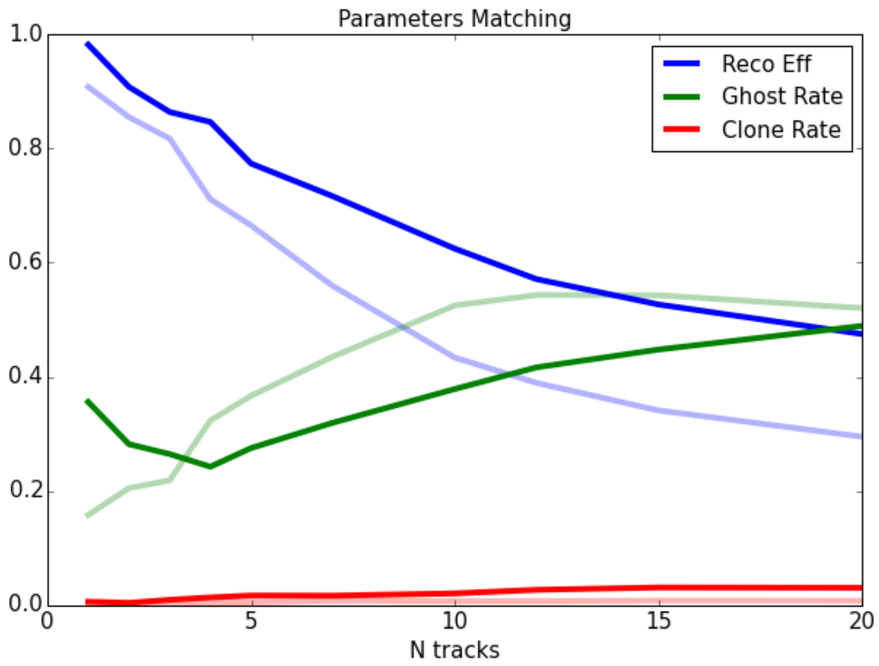
Extrapolation based on all hits previously added.

Extrapolation based on last several hits is suitable when shape of a track is unknown. On the other hand, the extrapolation based on all hits previously added to the track is more precisely. This leads to the more accurate track reconstruction.

Naive Track Following. Demonstration.

Quality metrics for the Naive Track Following and for the Simple Template Matching (transparent):





The Naive Track Following demonstrates better reconstruction efficiency and ghost rate for the large number of tracks.

Naive Track Following

Advantages of the method:

- Easy to apply
- Well for moderate track density
- Requires reasonable computational resources

Limitations of the method:

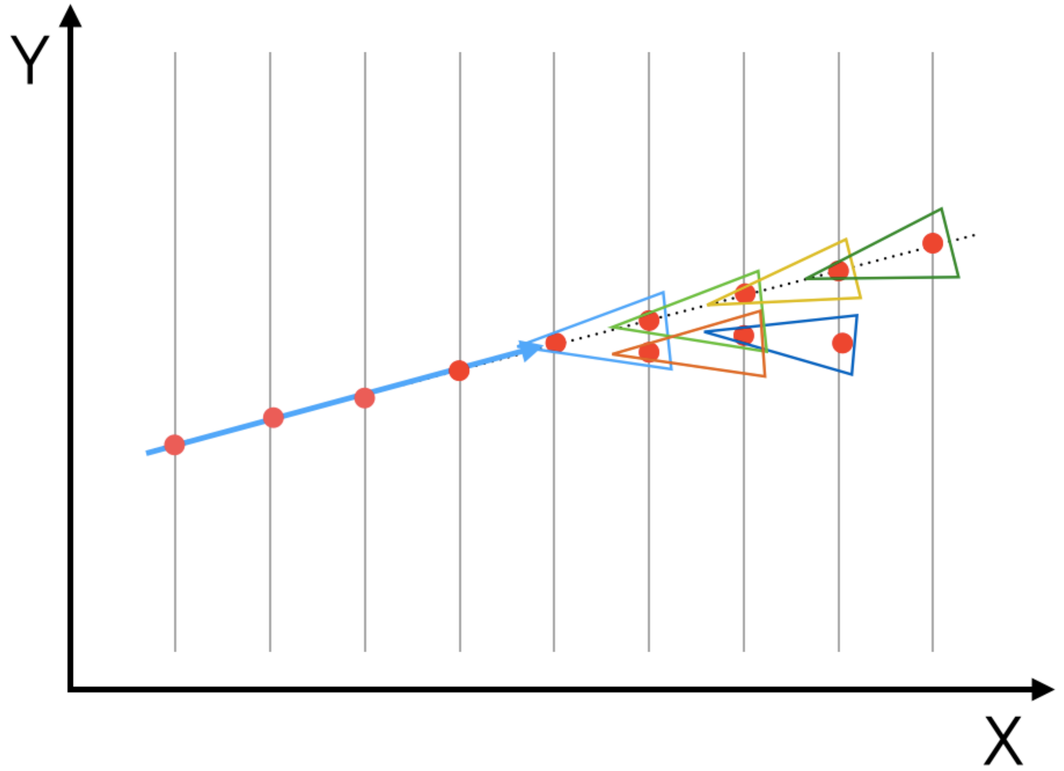
- Has strong limitations for large track density
- Sensitive to the missing and wrong hits

The Combinatorial Filter

The Combinatorial Filter

In each track following step, each hit which is suitable within wide tolerance creates new branch of the track following procedure. In result, tree of track candidates is created.

Then, the best candidates are selected base on chi-square values, number of hits, number of missed hits, etc..



The Combinatorial Filter

Advantages of the method:

- Potentially unbeatable in terms of track efficiency
- Good in hight track density case, where no one obvious path to be followed

Limitations of the method:

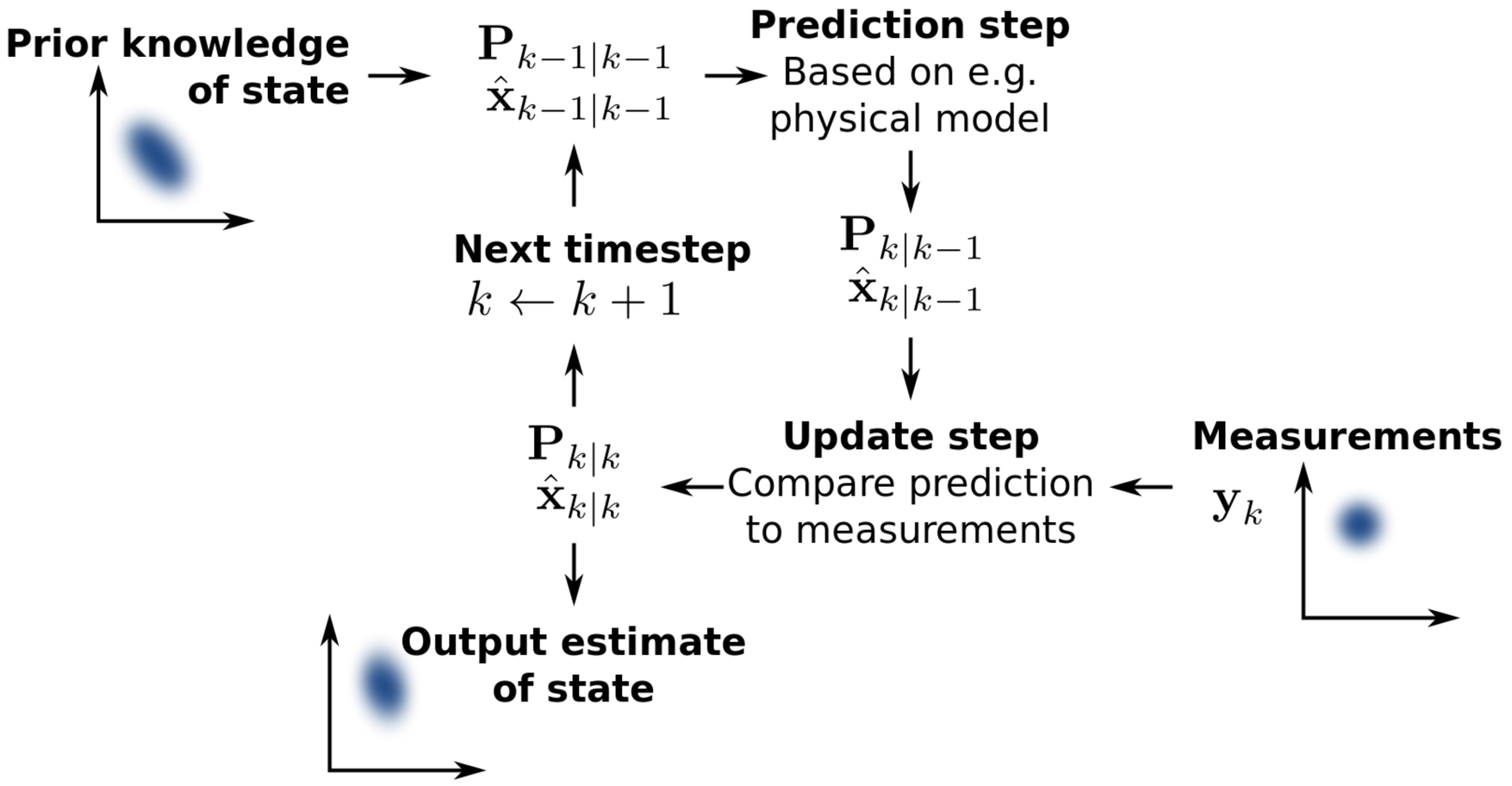
- Requires large computational resources for high number of combinations

The Kalman Filter

The Kalman Filter

«The Kalman Filter is an algorithm that uses a series of measurements observed over time, containing statistical noise and other inaccuracies, and produces estimates of unknown variables that tend to be more precise than those based on a single measurement alone, by using Bayesian inference and estimating a joint probability distribution over the variables for each timeframe.» [wikipedia]

The Kalman Filter



The Kalman Filter

The model assumes that the true state x evolves in time in following way:

$$x_k = F_k x_{k-1} + w_k$$

where F is the state transition matrix, w is the process noise with multivariate normal distribution:

$$w_k \sim N(0, Q_k)$$

An observation z of the true state x is made according to

$$z_k = H_k x_k + v_k$$

where H is the observation matrix, v is the observation noise with multivariate normal distribution:

$$v_k \sim N(0, R_k)$$

The Kalman Filter

Predict:

Predicted (a priori) state estimate $\hat{x}_{k|k-1} = F_k \hat{x}_{k-1|k-1}$

Predicted (a priori) estimate covariance $P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_k$

Update:

Measurement residual $\hat{y}_k = z_k - H_k \hat{x}_{k|k-1}$

Residual covariance $S_k = H_k P_{k|k-1} H_k^T + R_k$

Optimal Kalman gain $K_k = P_{k|k-1} H_k^T S_k^{-1}$

Updated (a posteriori) state estimate $\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k \hat{y}_k$

Updated (a posteriori) estimate covariance $P_{k|k} = (I - K_k H_k) P_{k|k-1}$

The Kalman Filter. Demonstration.

Let consider a linear track in (x, y) plane:

$$y_{hit} = \beta x_{hit} + \alpha$$

Then, the true state vector and the transition matrix in the Kalman Filter model can be written as

$$x_k = \begin{pmatrix} \beta \\ \alpha \end{pmatrix}_k \qquad F_k = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

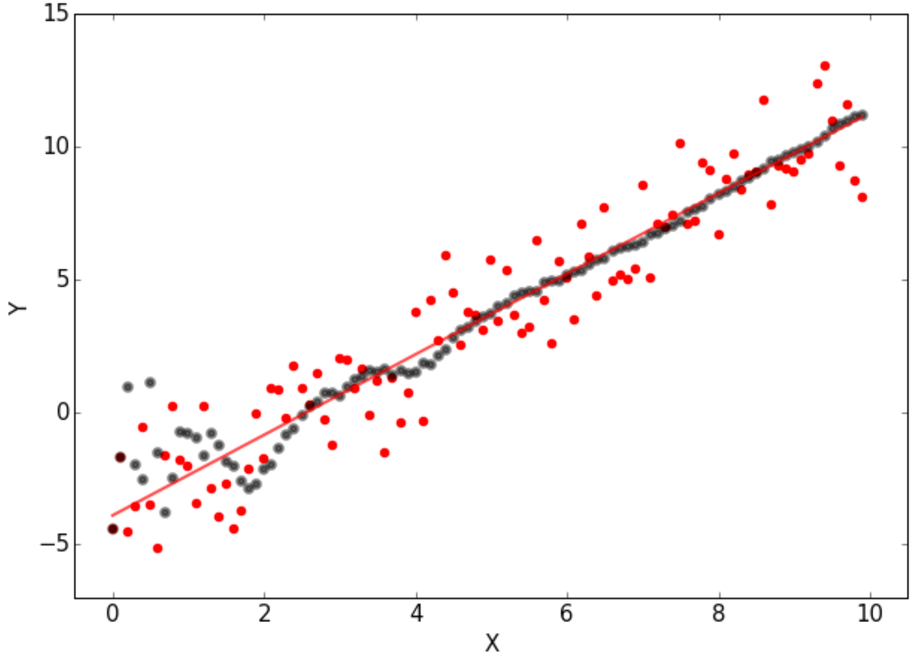
For the observations we take y coordinates of the hits:

$$z_k = y_{hit}$$

Then, the observation matrix in the model:

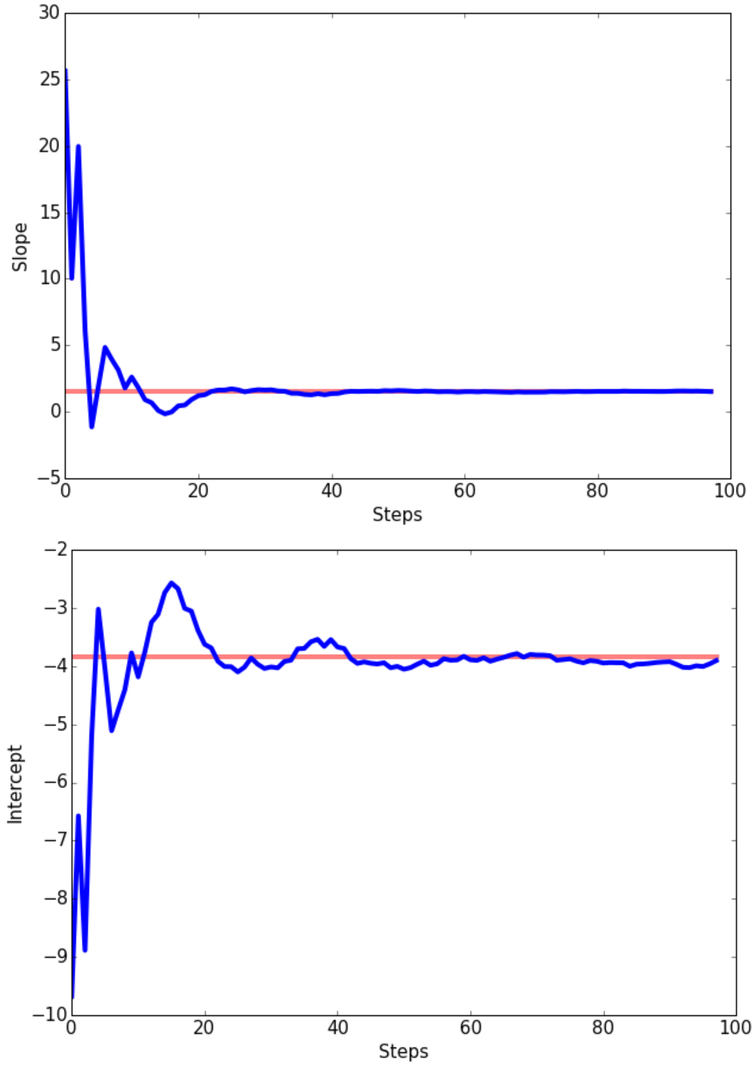
$$H_k = \begin{pmatrix} x_{hit} & 1 \end{pmatrix}$$

The Kalman Filter. Demonstration.



Red: hits of the track; Black: predictions of the Kalman filter

$$\hat{z}_k = H_k x_k$$



The state evolution

The Kalman Filter and Track Following

1. Start from a seed.

2. Use predict stage of the Kalman filter model:

$$\hat{x}_{k|k-1} = F_k \hat{x}_{k-1|k-1}$$
$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_k$$

3. Calculate a new measurement residual value. Hits with too large residual values are rejected.

$$\hat{y}_k = z_k - H_k \hat{x}_{k|k-1}$$

4. Update the state and its covariance:

$$S_k = H_k P_{k|k-1} H_k^T + R_k$$
$$K_k = P_{k|k-1} H_k^T S_k^{-1}$$
$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k \hat{y}_k$$
$$P_{k|k} = (I - K_k H_k) P_{k|k-1}$$

The Kalman Filter and Track Following

5. Calculate filtered residual and observation covariance:

$$\tilde{y}_k = (I - H_k K_k)\tilde{y}_k$$
$$\tilde{R}_k = (I - H_k K_k)R_k$$

6. Calculate chi-square contribution of the filtered hit:

$$\chi^2_{k,F} = \tilde{y}_k^T \tilde{R}_k \tilde{y}_k$$

7. Use the chi-square contribution for the more precisely hits selection.

8. Continue the procedure until the end of the tracking area is reached, or suitable hits can not be found.

9. The state vector at the last filtered hit contains the full information from all hits.

The Kalman Filter and Smoother Equations

The full state vector can be passed upstream with the Rauch—Tung—Striebel smoother equations:

$$C_k = P_{k|k} F_{k+1}^T P_{k+1|k}^{-1}$$
$$\hat{x}_{k|n} = \hat{x}_{k|k} + C_k(\hat{x}_{k+1|n} - \hat{x}_{k+1|k})$$
$$P_{k|n} = P_{k|k} + C_k(P_{k+1|n} - P_{k+1|k})C_k^T$$
$$\hat{y}_{k|n} = z_k - H_k \hat{x}_{k|n}$$

The smoothing proceeds step by step in the direction opposite to that of the filter. This procedure improves the model predictions on early steps of the track following procedure.

The Kalman Filter

Advantages of the method:

- Exploit all knowledge from the accumulated hits for the track reconstruction
- Requires small computational resources
- Easy to apply

Limitations of the method:

- Has limitations for large track density

Summary of the Local Methods

1. Track Seeds
2. Naive Track Following
3. The Combinatorial Filter
4. The Kalman Filter

Tracks Recognition

Combining Track Algorithms

Combining Track Algorithms

- Track recognition strategies depend on a detector design. So, think how you will find tracks before building the detector.
- Most robust strategies are combinations of the different track recognition methods.
- Unfortunately, there is no universal solution for the method combination.

Combining Track Algorithms

Remember, that each track recognition method has its advantages and limitations:

〉 Neural network methods are flexible and does not require knowledge of the track model . These methods are patient to the tracks which deviate from their models, but have large ghost and clone rates.

〉 Hough transform and template matching work well with particular track models. The methods are intolerant to the track which are deviate from their models.

〉 Local methods require just local knowledge about the tracks. Work well with low and moderate tracks density. Tolerant to the curve tracks.

Combining Track Algorithms

Choosing a track finding strategy takes into account:

〉 A detector design, its geometry and physics of an experiment.

〉 Advantages and limitation of different model. Foe example, some of them can be used for preselecting hits for the other ones.

Tracks Recognition in Real Experiments

Tracks Recognition in Real Experiments

Silicon Strip Detectors:

〉 Semiconductor-based devices structured in strips typically down to widths of 25 µm.

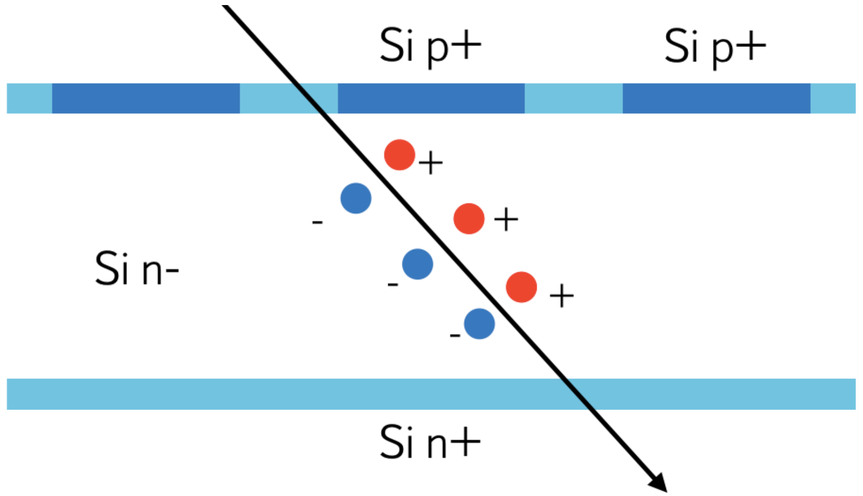
〉 When a particle goes through a strip pair of an electron and a hole is created. This can be registered as impulse.

〉 High spatial resolution.

〉 Resilience against radiation damage.

〉 Too expensive for coverage of large volumes.

〉 Works in 2D and 3D.



Tracks Recognition in Real Experiments

Straw tubes and wire chambers:

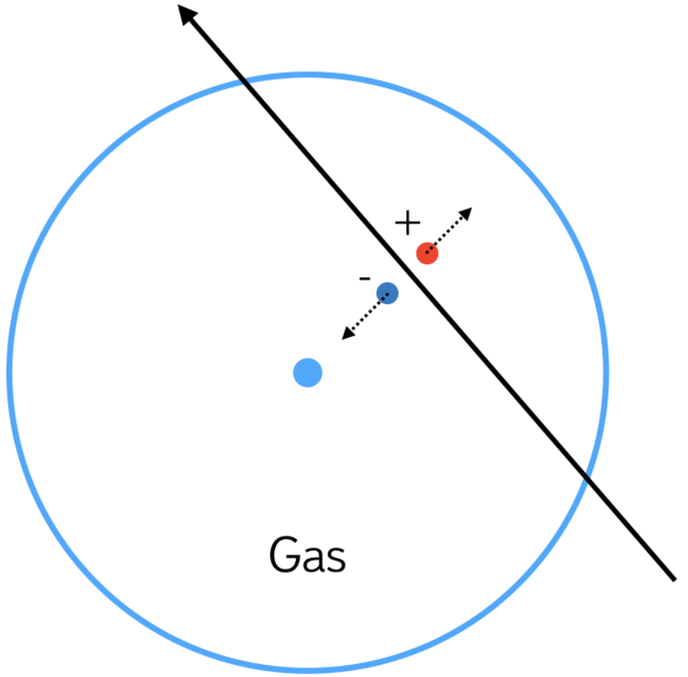
〉 When a particle goes through the tube, it ionize gas inside the tube. Under the voltage the election and ion go in different directions. This can be registered as impulse.

〉 We do not know where exactly along the tube the particle passed.

〉 But know track distance from the wire.

〉 Cheap for coverage of large volumes.

〉 Works only in 2D. For 3D it’s needed several stereo planes.



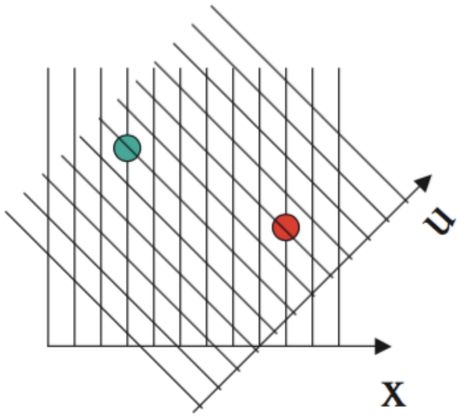
Tracks Recognition in Real Experiments

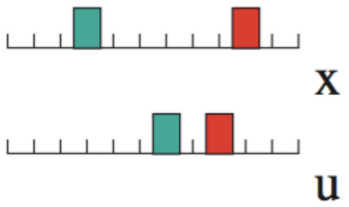
Often, 3D tracks recognition is done by using 2D projections:

〉 Find tracks in one view first, then combine with hits in other views.

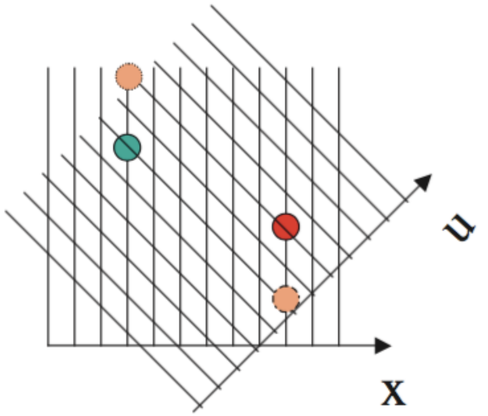
〉 Find tracks in two projections, then combine.

Using several stereo planes of straw tubes for the 3D tracks reconstruction leads to the space points ambiguity.





mirror points

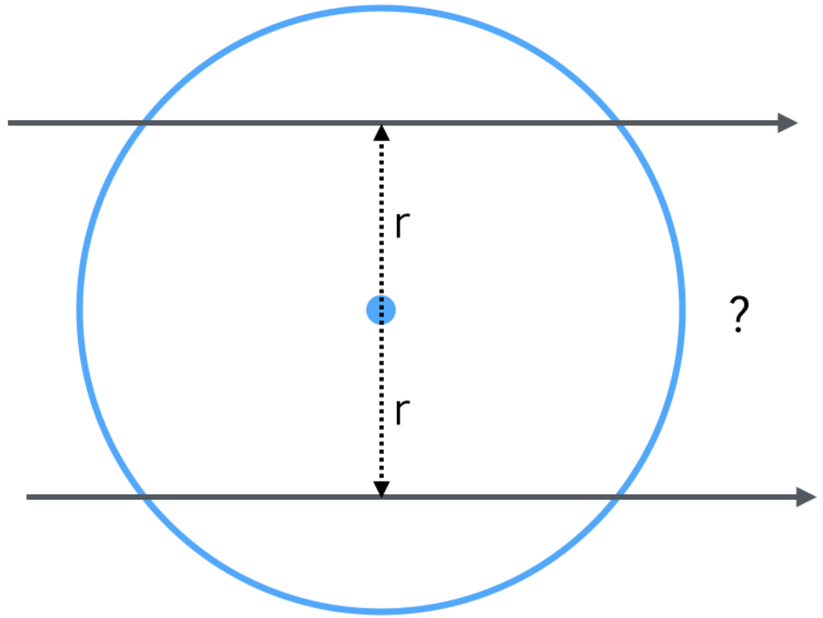


Tracks Recognition in Real Experiments

A straw tube can estimate distance between a track and the tube's wire. But it can not estimate on which side the track passed.

This problem is called left-right ambiguity. This leads to the several alternative tracks candidates.

Track recognition strategy should takes this into account.



Tracks Recognition

Sources

Sources

This course was prepared using the following sources:

1. «Pattern Recognition and Event Reconstruction in Particle Physics Experiments» http://arxiv.org/abs/physics/0402039

2. «Pattern recognition» http://bit.ly/28KWfct

3. «Pattern recognition in HEP» http://bit.ly/28LUPSy

4. «Современные методы обработки данных в физике высоких энергий» http://www1.jinr.ru/Pepan/v-33-3/v-33-3-11.pdf

5. «Performance Evaluation of RANSAC Family» http://www.bmva.org/bmvc/2009/Papers/Paper355/Paper355.pdf

6. https://en.wikipedia.org/wiki/Kalman_filter