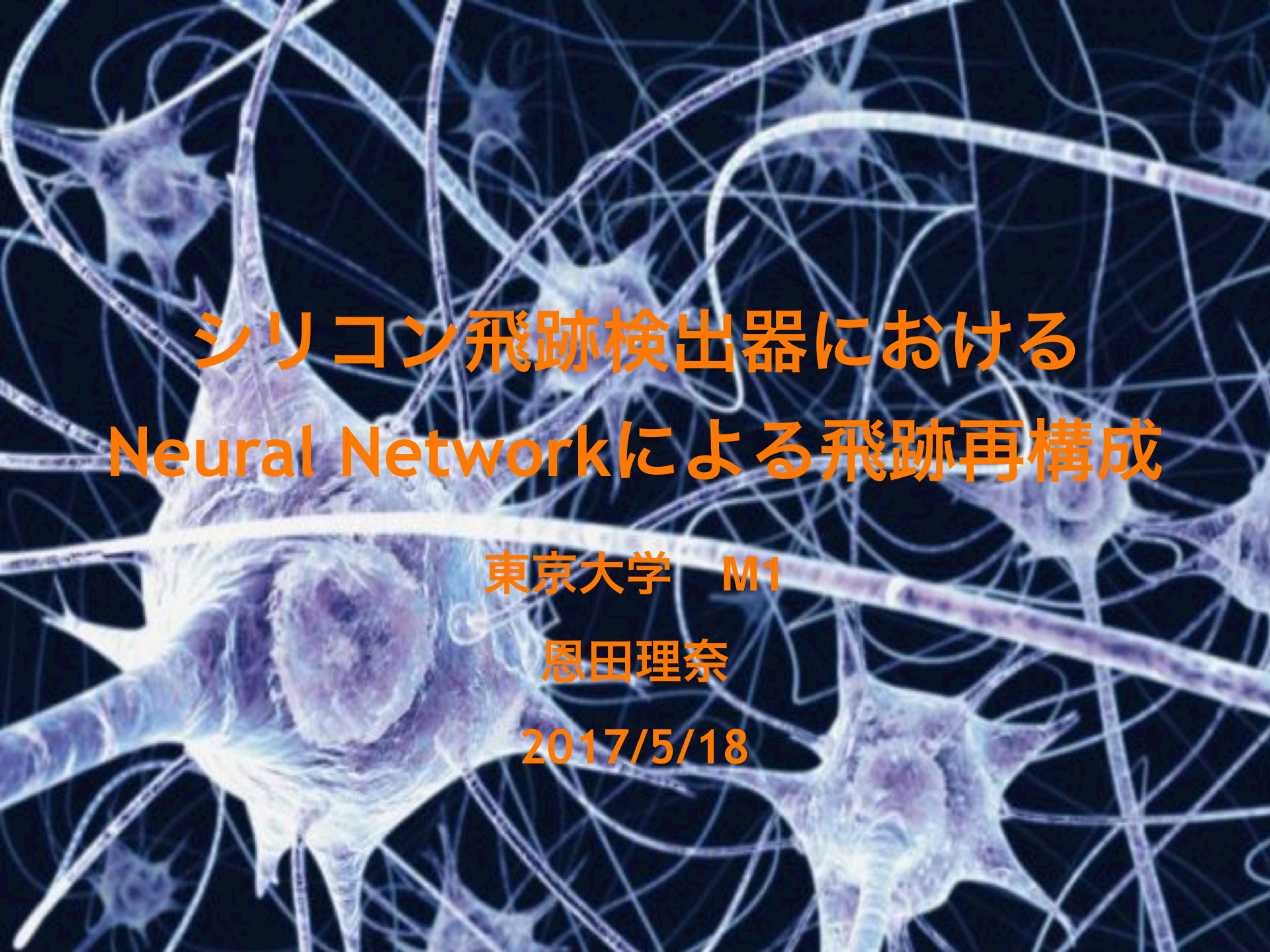


シリコン飛跡検出器における Neural Networkによる飛跡再構成

東京大学　M1

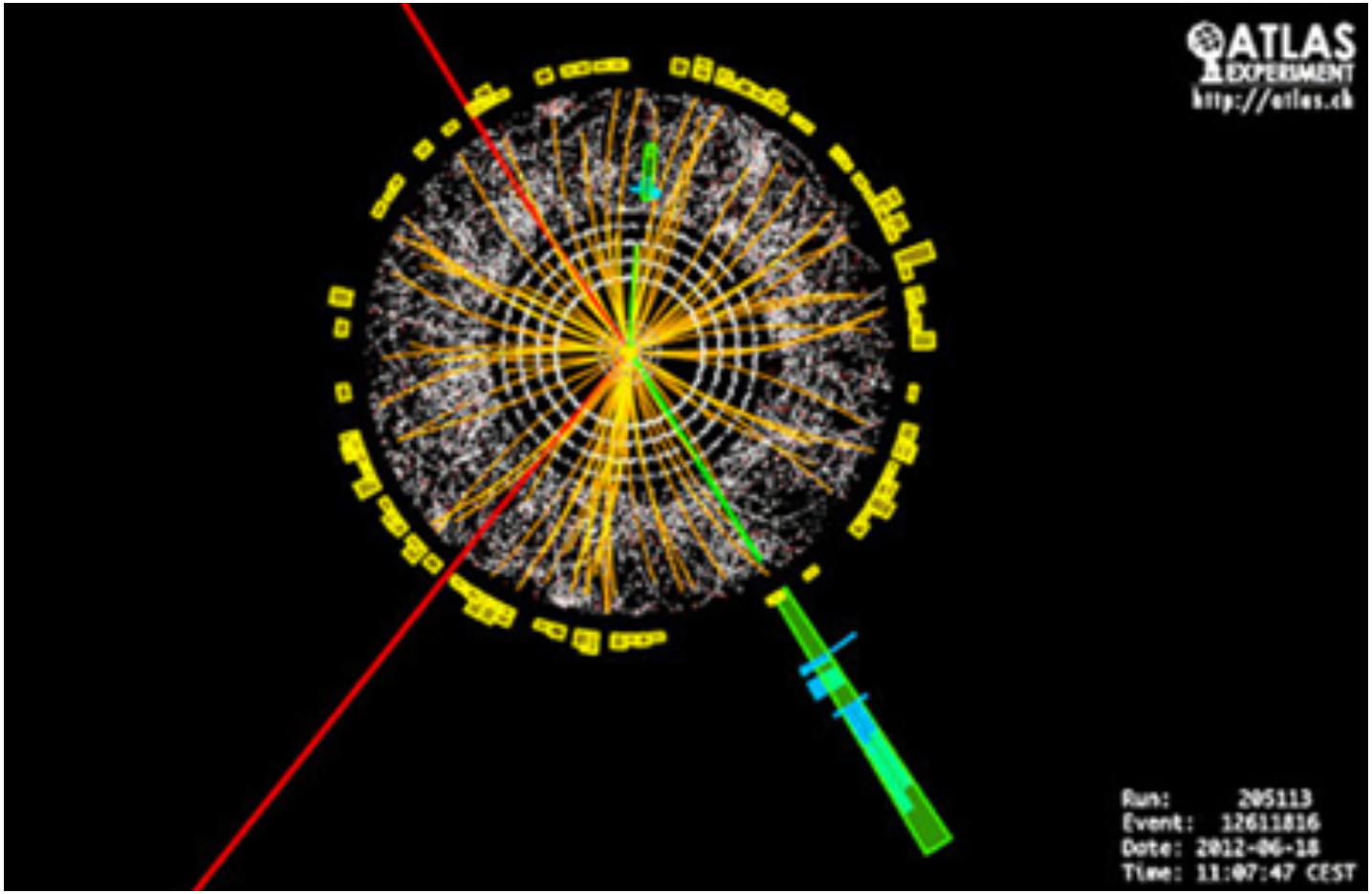
恩田理奈

2017/5/18

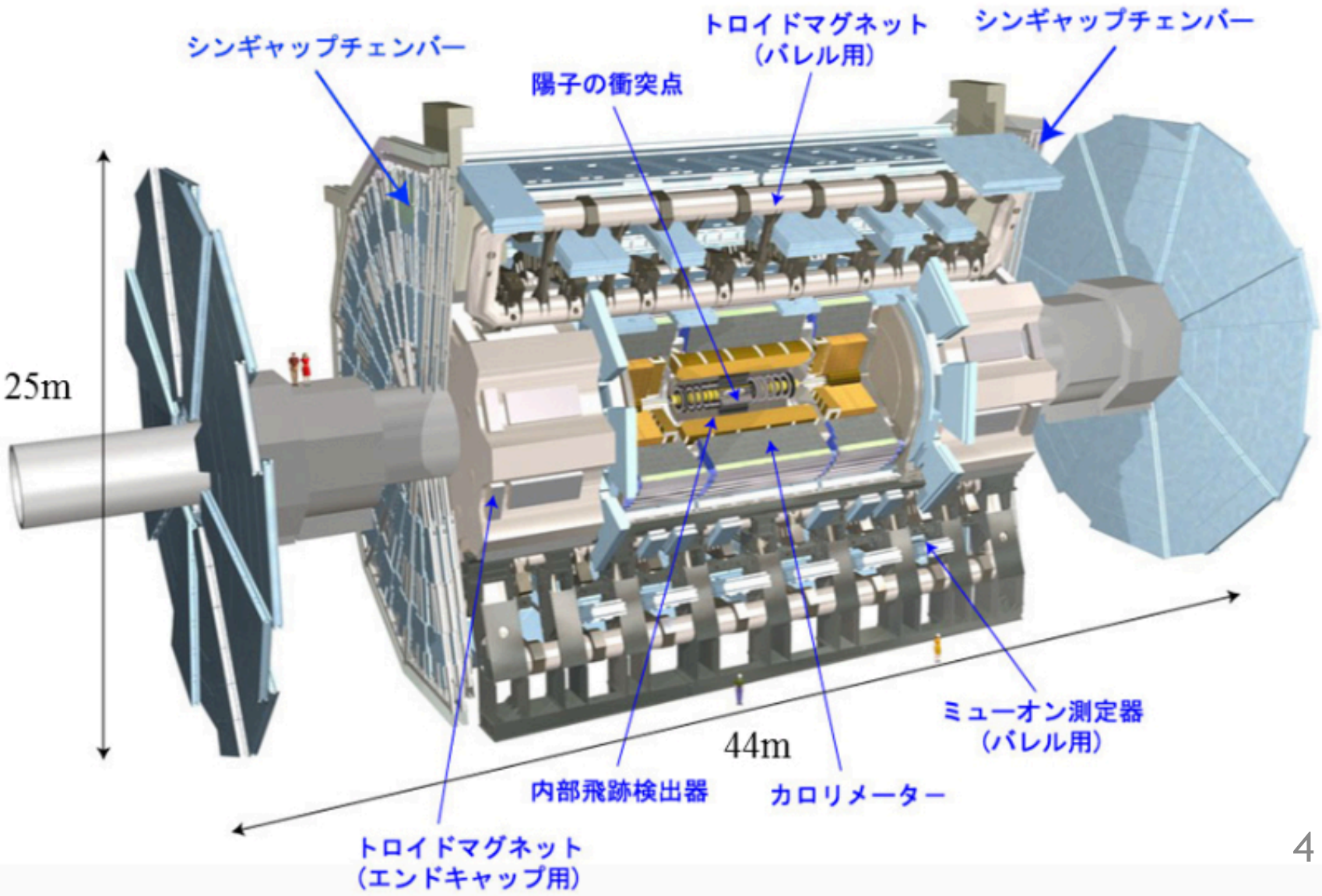
Outline

1. 背景
2. 目的
3. 結果
4. 結論

素粒子実験

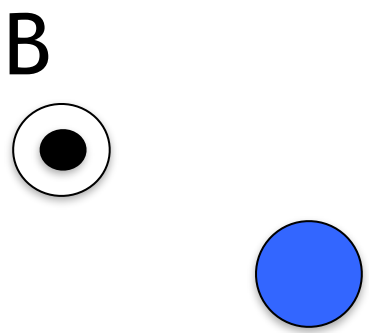

ATLAS
EXPERIMENT
http://atlas.ch
Run: 205113
Event: 12611816
Date: 2012-06-18
Time: 11:07:47 CEST

検出器

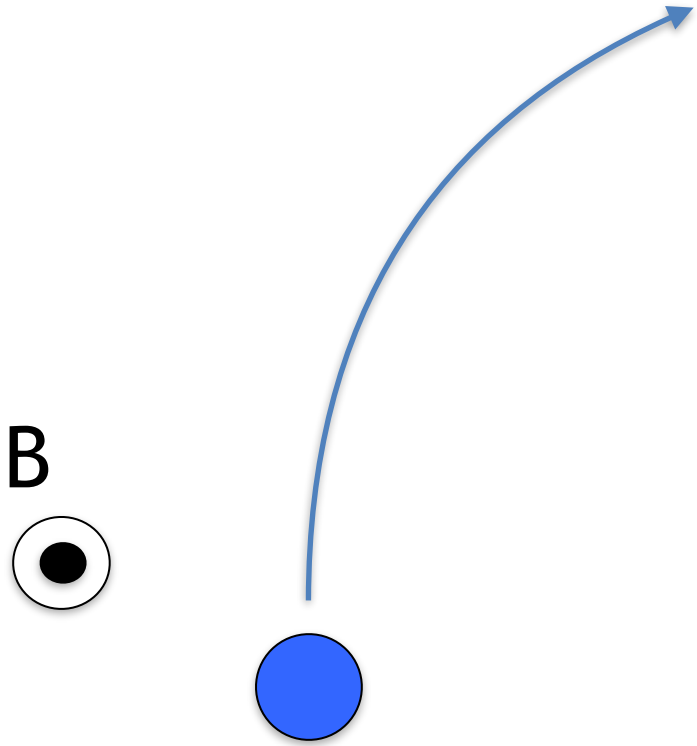

シンギャップチェンバー
トロイドマグネット
（バレル用）
陽子の衝突点
シンギャップチェンバー
25m
44m
ミューオン測定器
（バレル用）
内部飛跡検出器
カロリメーター
トロイドマグネット
（エンドキャップ用）

運動量の計算

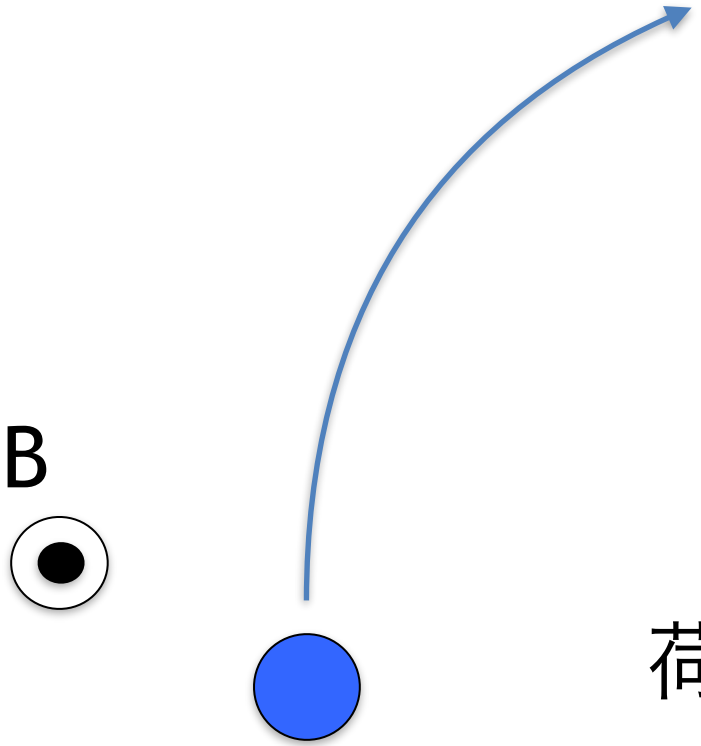
運動量の計算



運動量の計算



運動量の計算

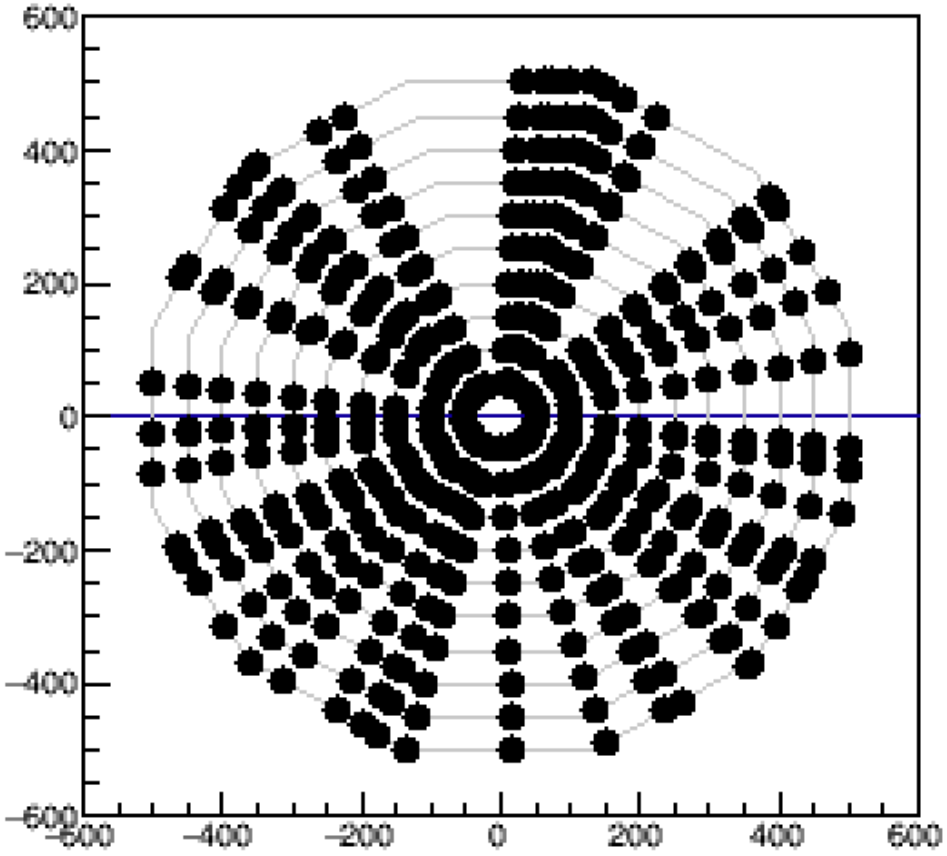


$$m\frac{v^2}{r} = qvB$$

$$\Leftrightarrow\ p = mv = qBr$$

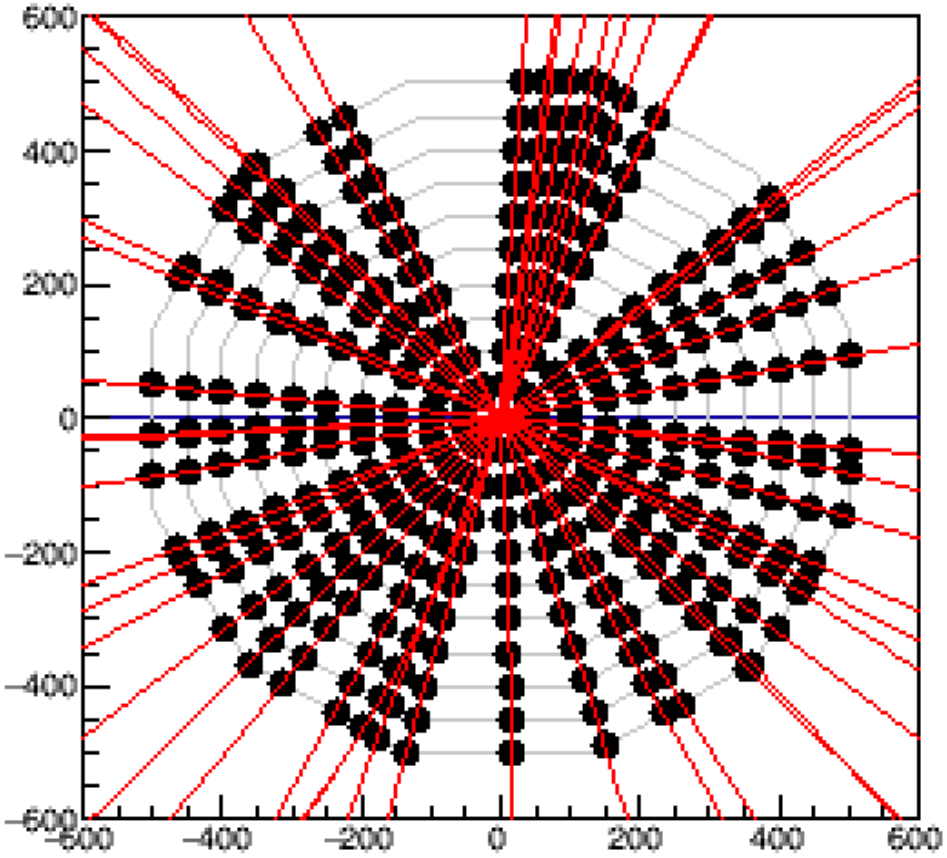
荷電粒子の円軌道の半径より
運動量を求めることができる

飛跡再構成



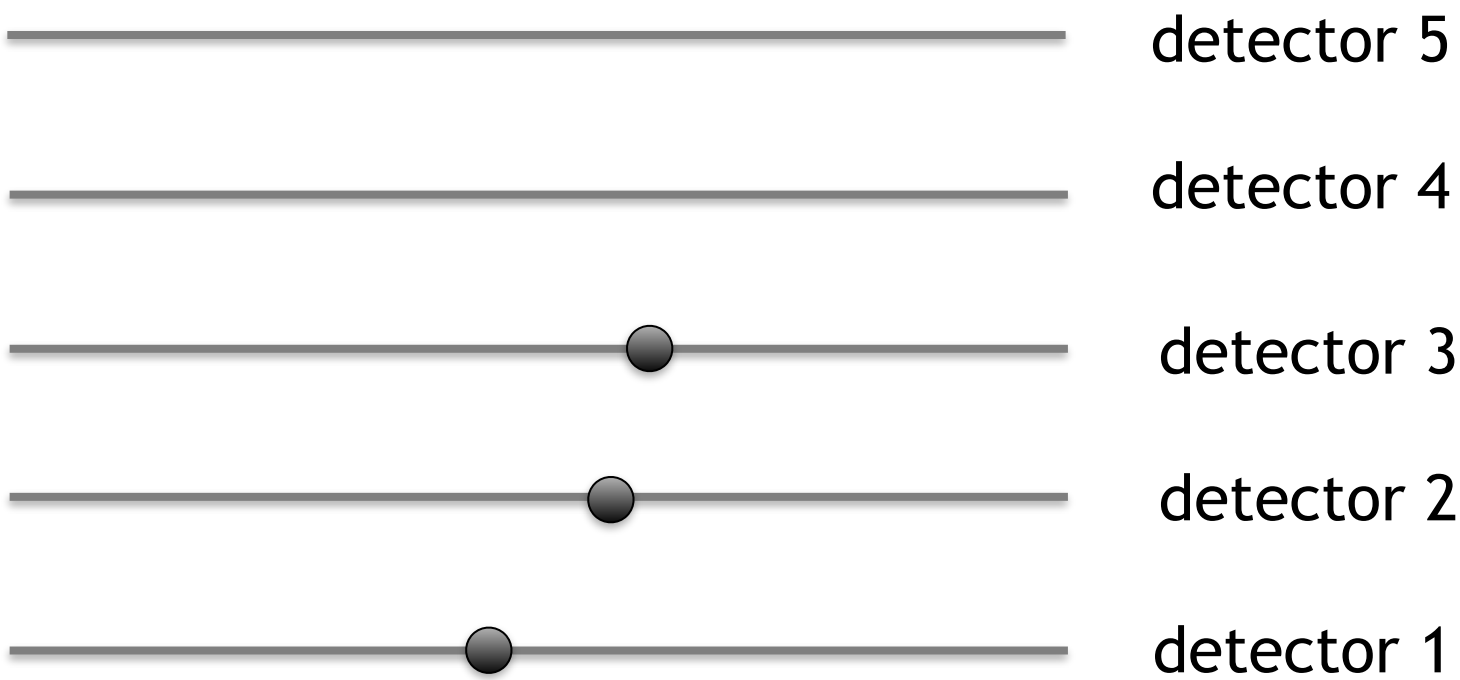
選んだhitの組み合わせを円軌道でfitし、適切なものを選ぶ

飛跡再構成



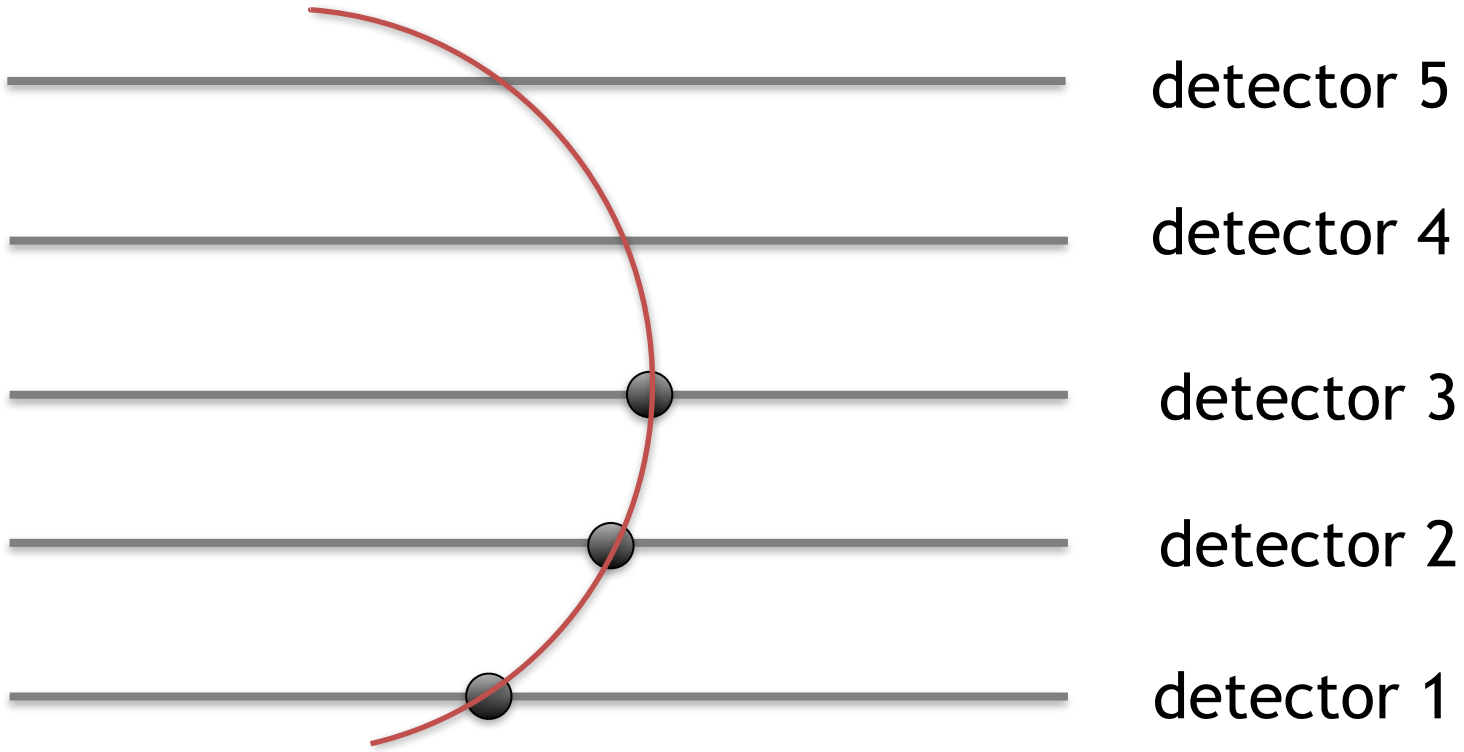
選んだhitの組み合わせを円軌道でfitし、適切なものを選ぶ

Kalman Filter



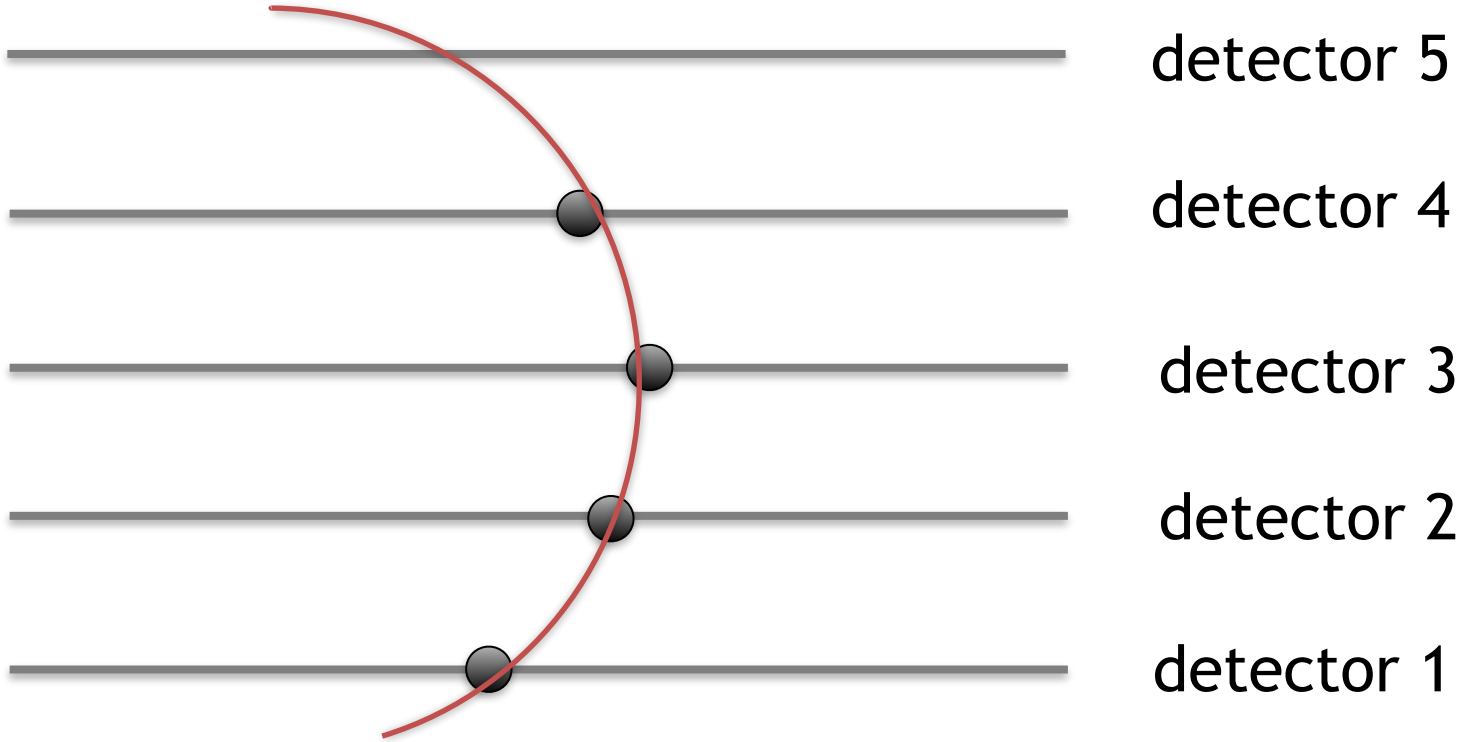
detectorのhitを加えてparameterを補正していく

Kalman Filter



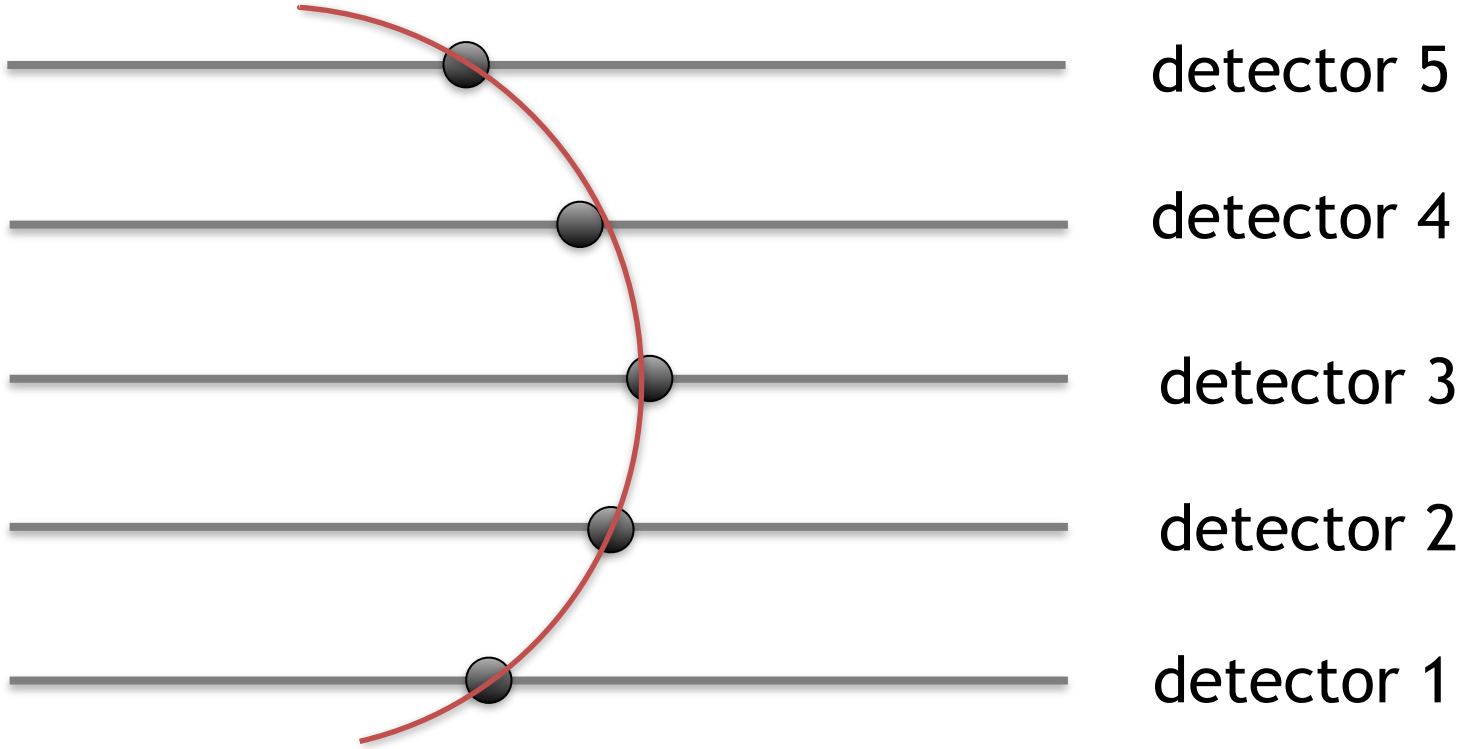
detectorのhitを加えてparameterを補正していく

Kalman Filter



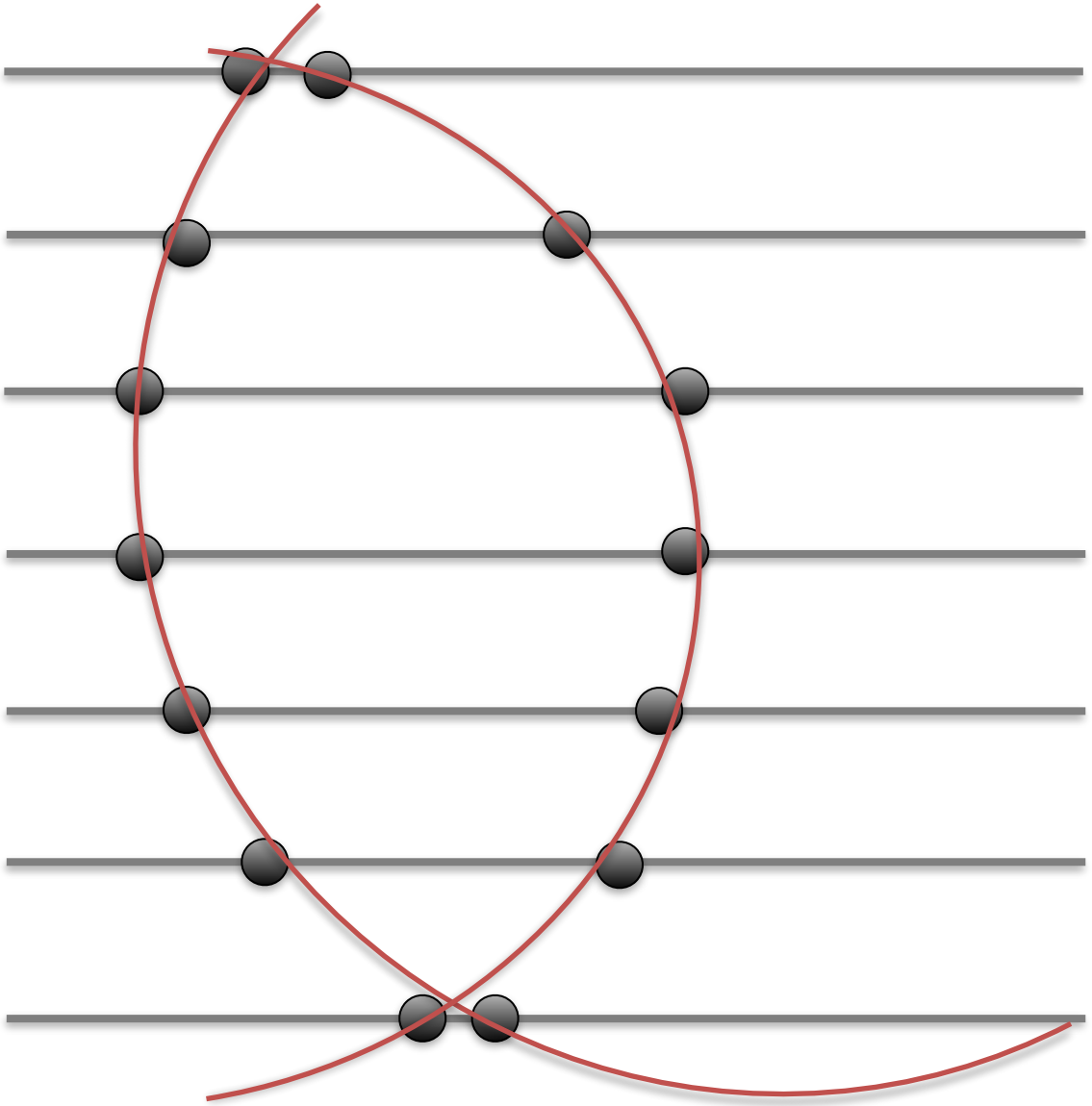
detectorのhitを加えてparameterを補正していく

Kalman Filter

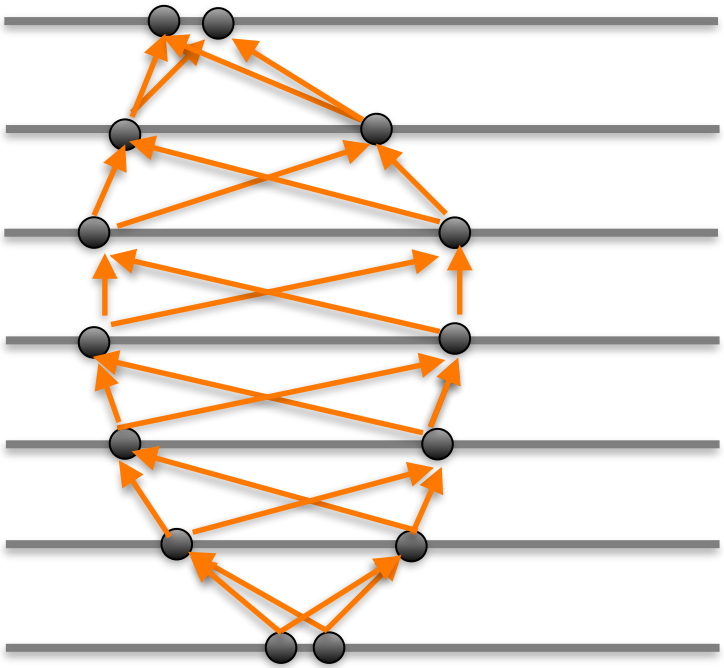


detectorのhitを加えてparameterを補正していく

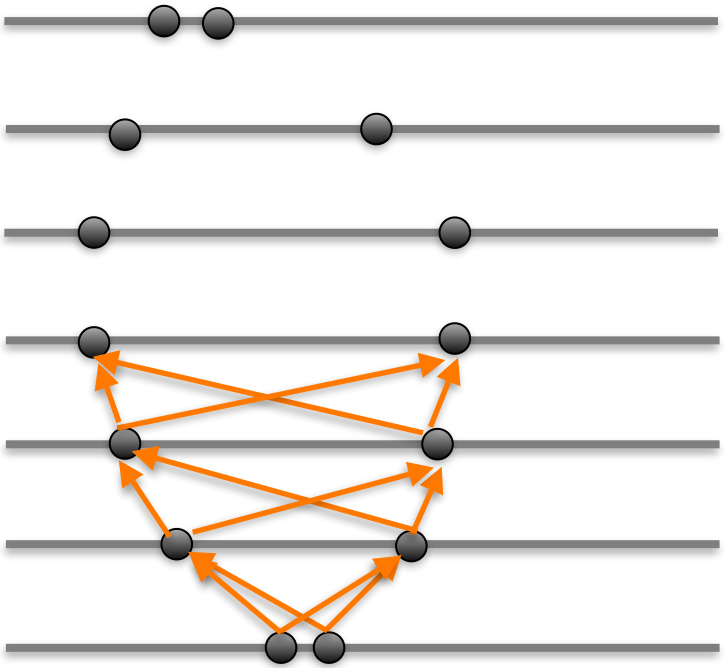
Least Squares vs Kalman Filter



Least Squares vs Kalman Filter

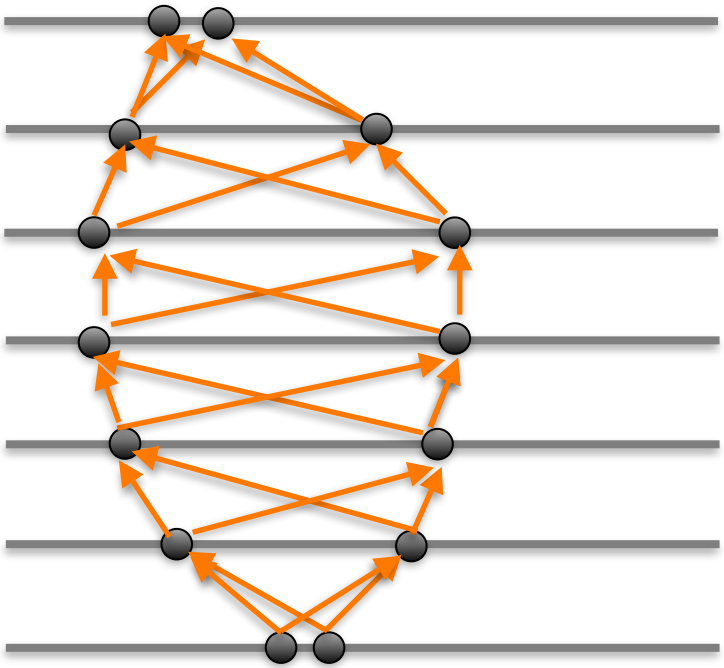


$2^7 = 128$

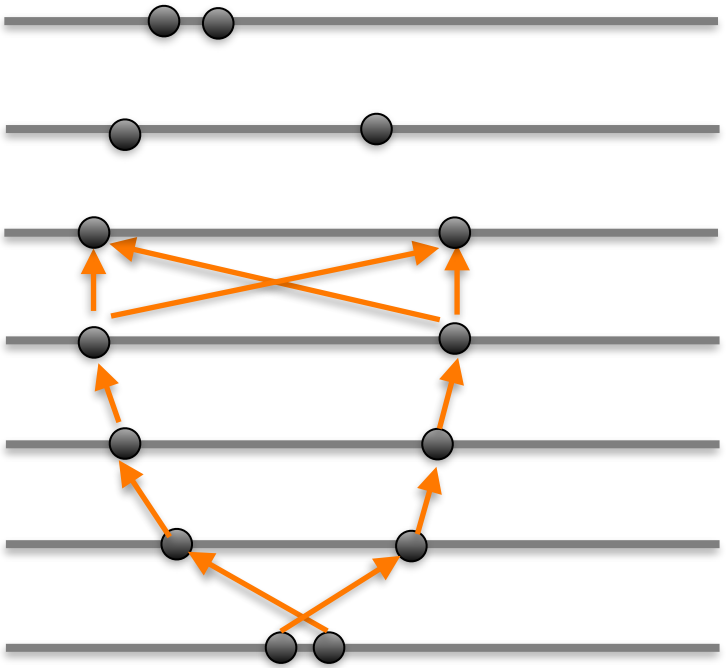


$2^4 = 16$

Least Squares vs Kalman Filter

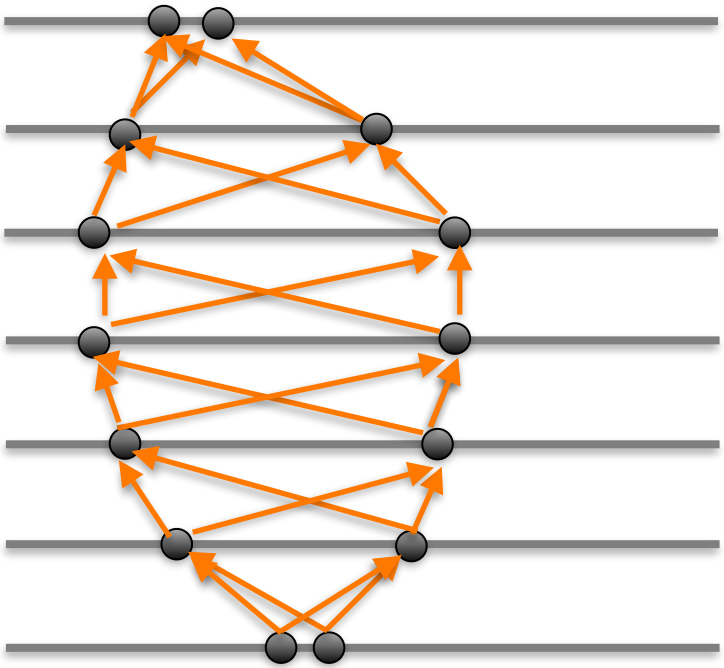


$2^7 = 128$

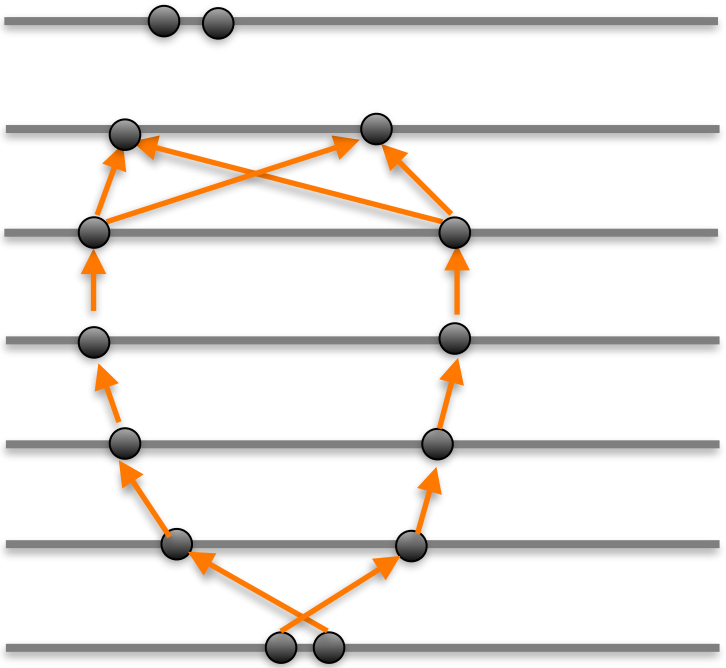


$2^4 + 2 \times 2 = 20$

Least Squares vs Kalman Filter

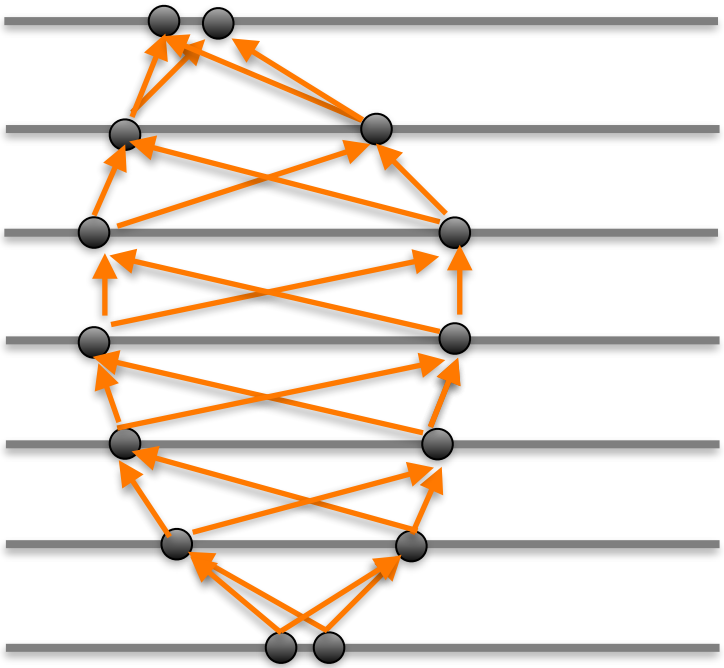


$$2^7 = 128$$



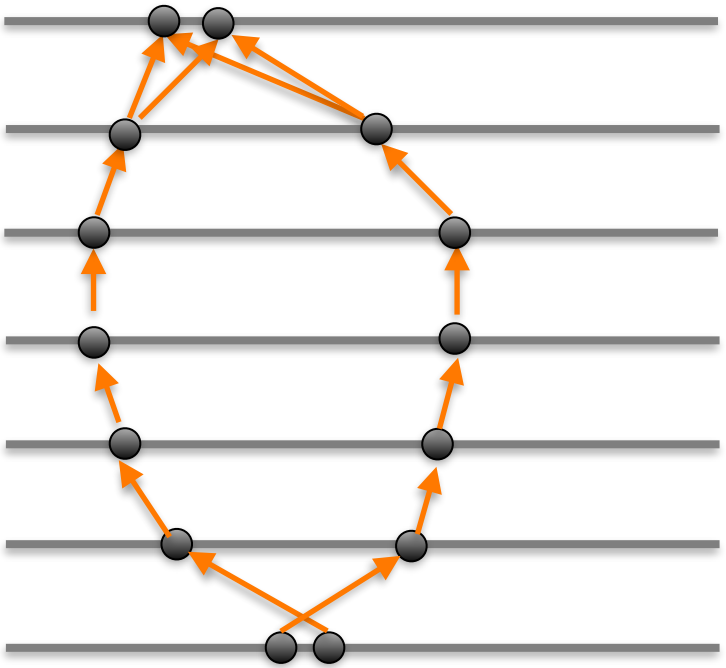
$$2^4 + 2 \times 2 + 2 \times 2 = 24$$

Least Squares vs Kalman Filter



$$2^7 = 128$$

~ (#track)^(#detector)



$$2^4 + 2 \times 2 + 2 \times 2 + 2 \times 2 = 28$$

~ (#track)^4

計算が速い！

目的

Neural Networkを用いて、飛跡再構成をより効率よく、短時間で行う

Neural Network

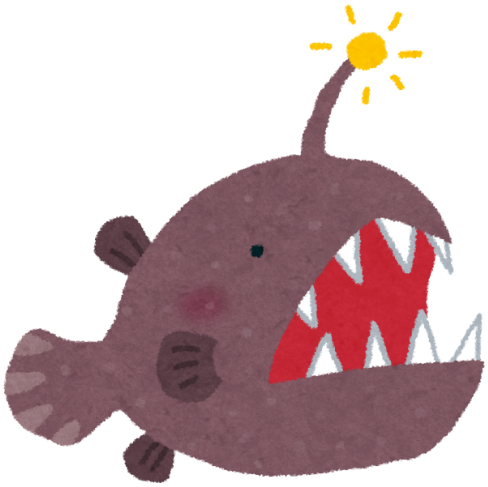
脳機能に見られるいくつかの特性を計算機上のシミュレーションによって表現することを目指した数学モデル

利用例：　画像認識、音声認識

Neural Network

脳機能に見られるいくつかの特性を計算機上のシミュレーションによって表現することを目指した数学モデル

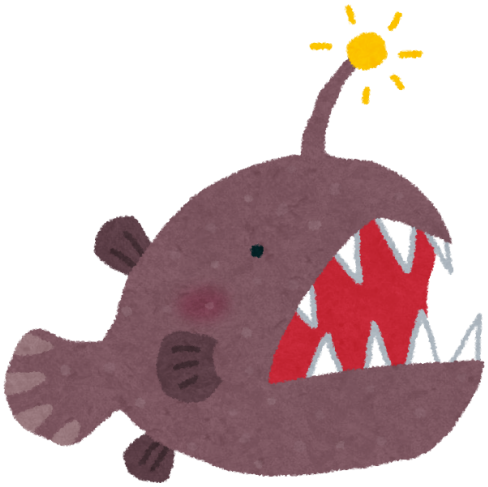
利用例：　画像認識、音声認識



Neural Network

脳機能に見られるいくつかの特性を計算
レーションによって表現することを目
ル

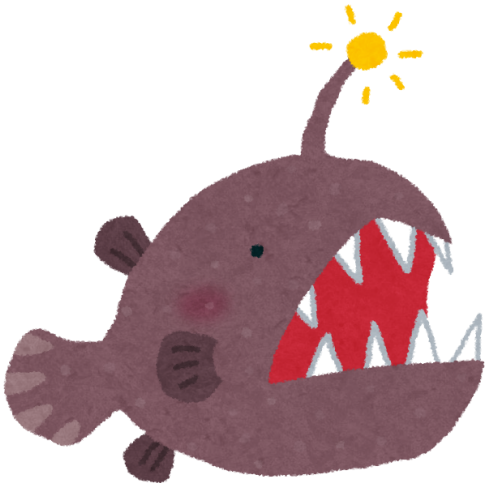
利用例：　画像認識、音声認識



Neural Network

脳機能に見られるいくつかの特性を計算機上のシミュレーションによって表現することを目指した数学モデル

利用例：　画像認識、音声認識

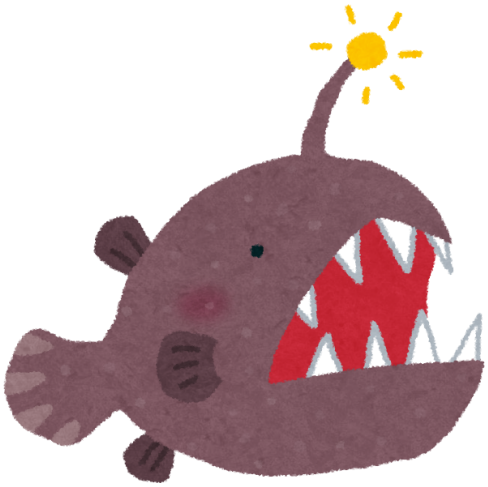


Neural Network

Neural Network

脳機能に見られるいくつかの特性を計算
レーションによって表現することを目
ル

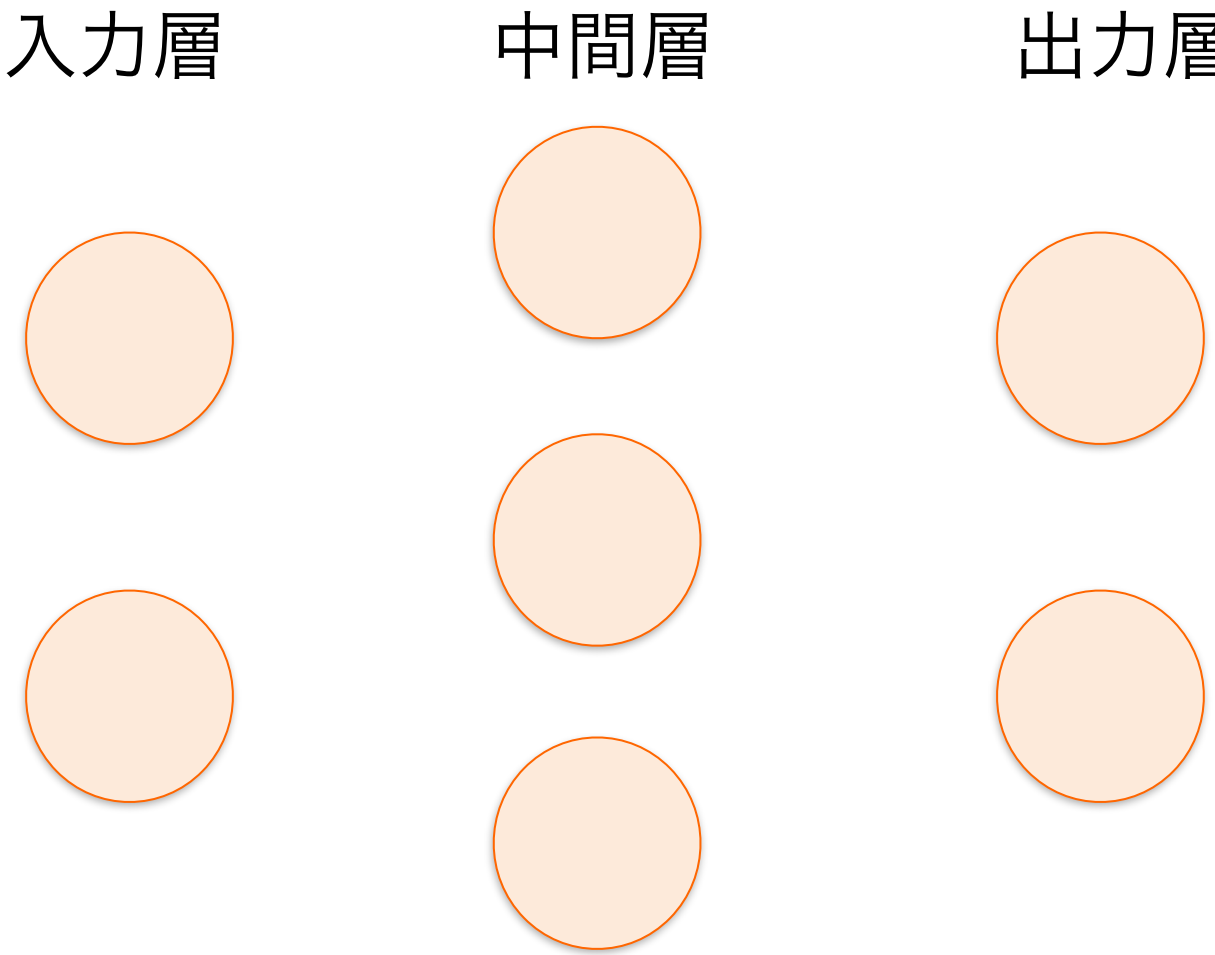
利用例：　画像認識、音声認識



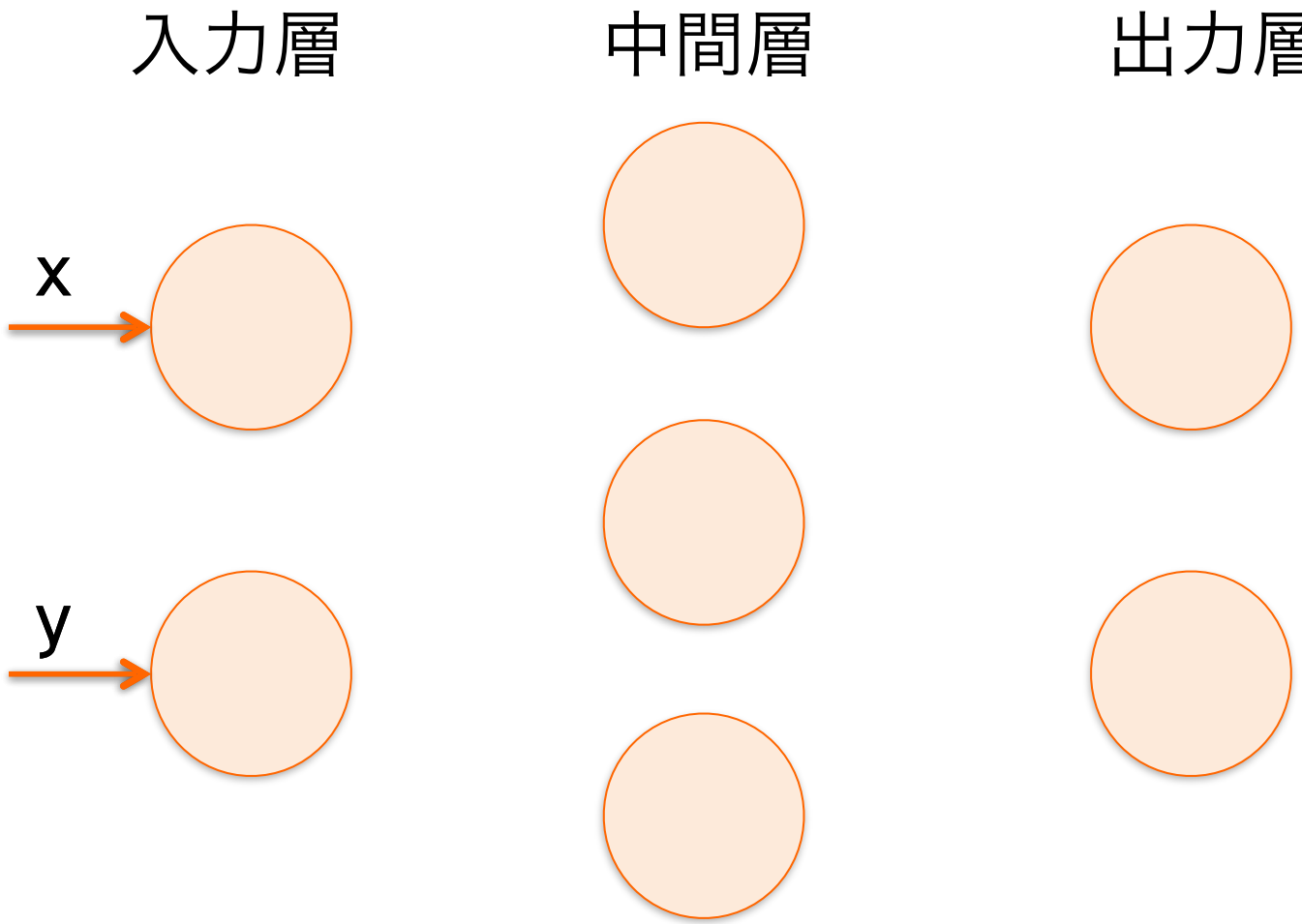
チョウチンアンコウ

Neural Network

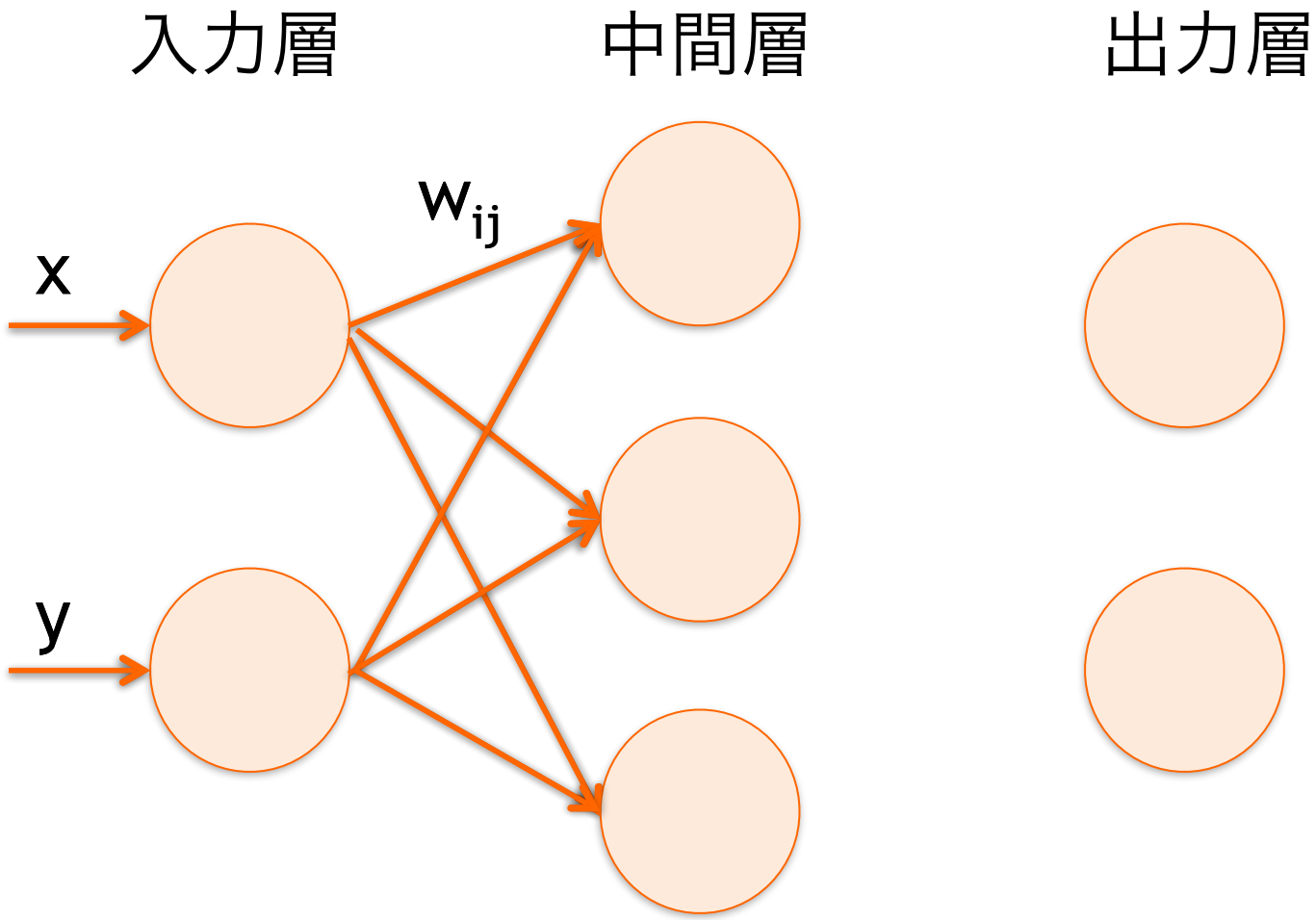
Neural Network



Neural Network

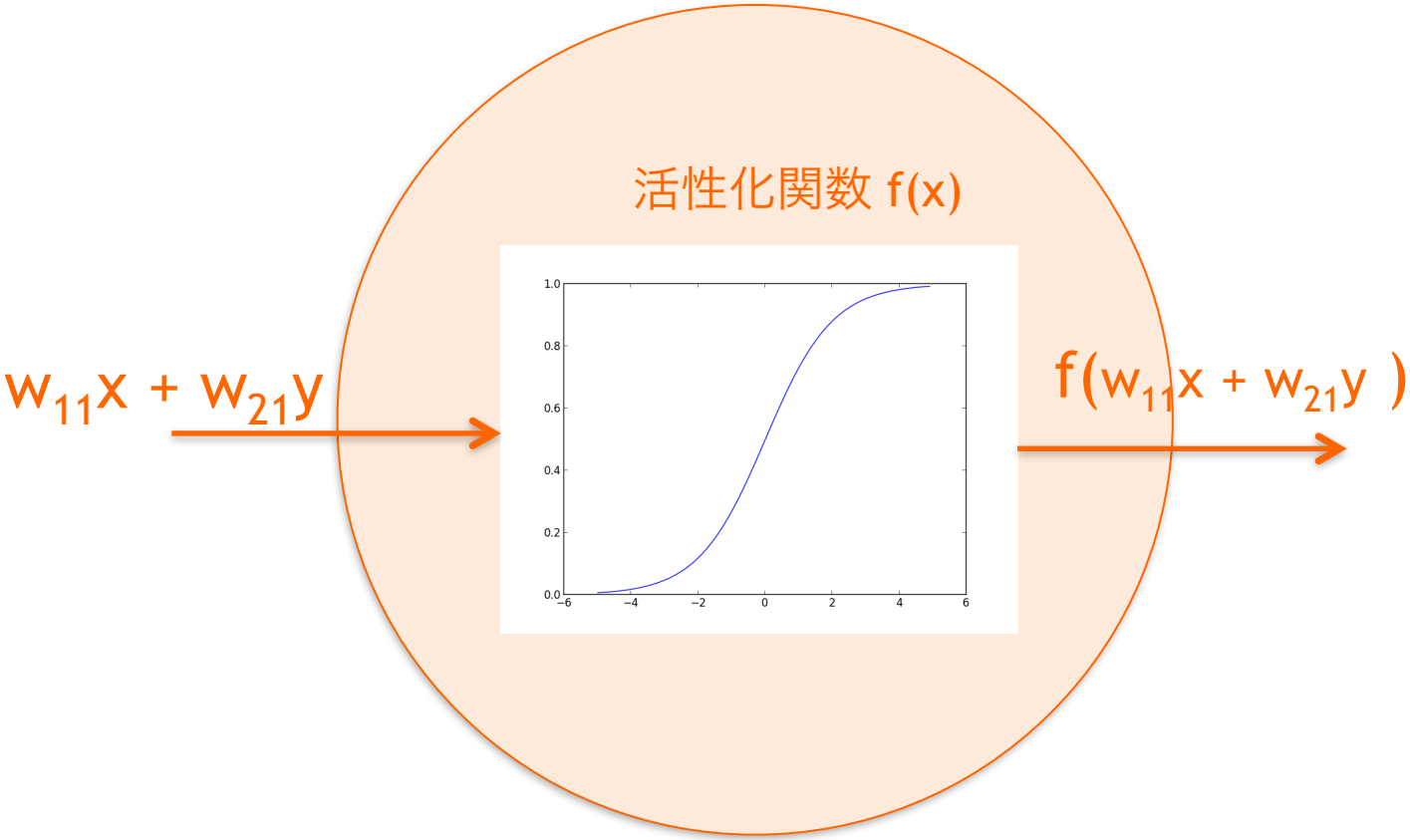


Neural Network

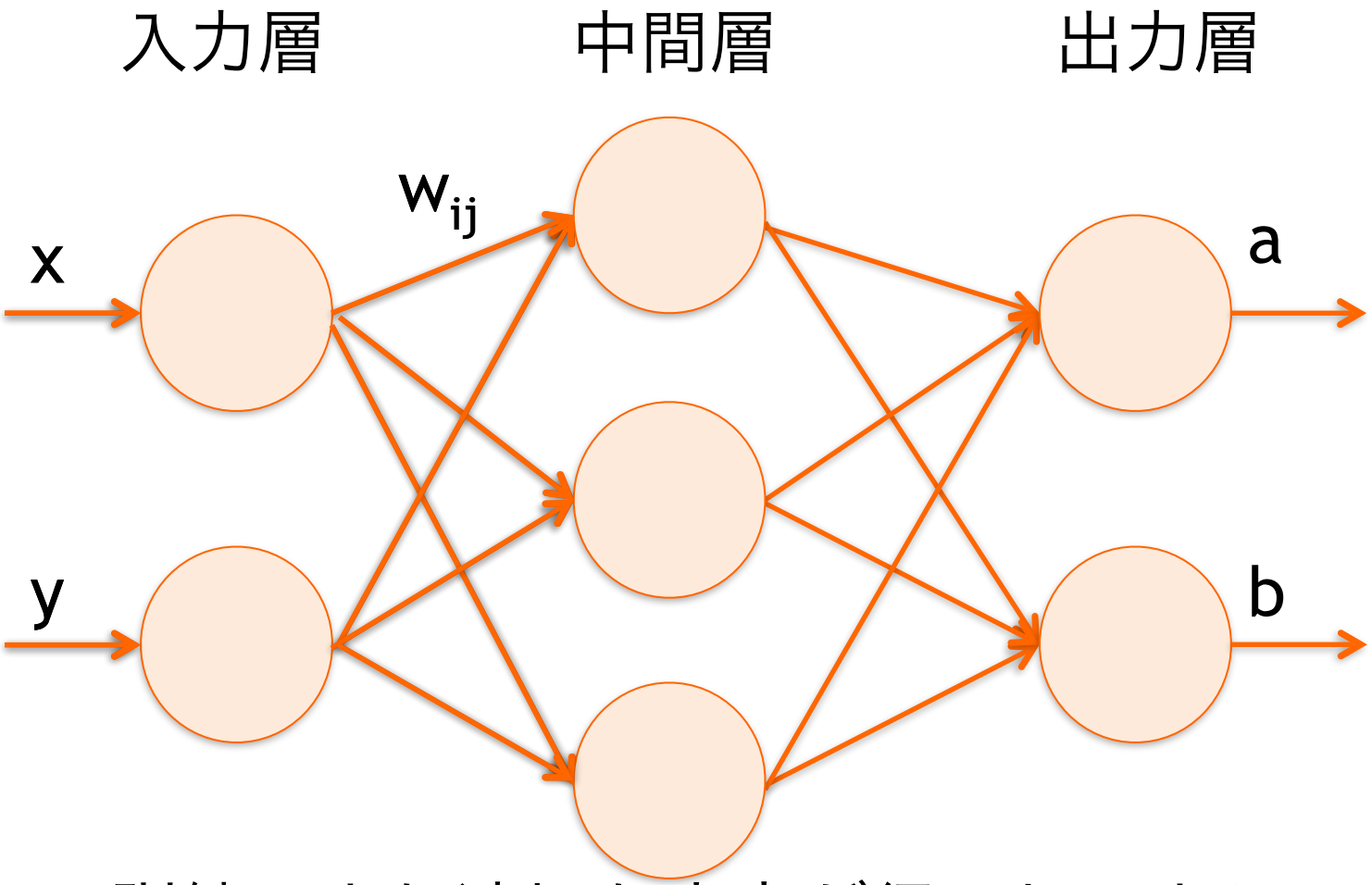


Neural Network

中間層

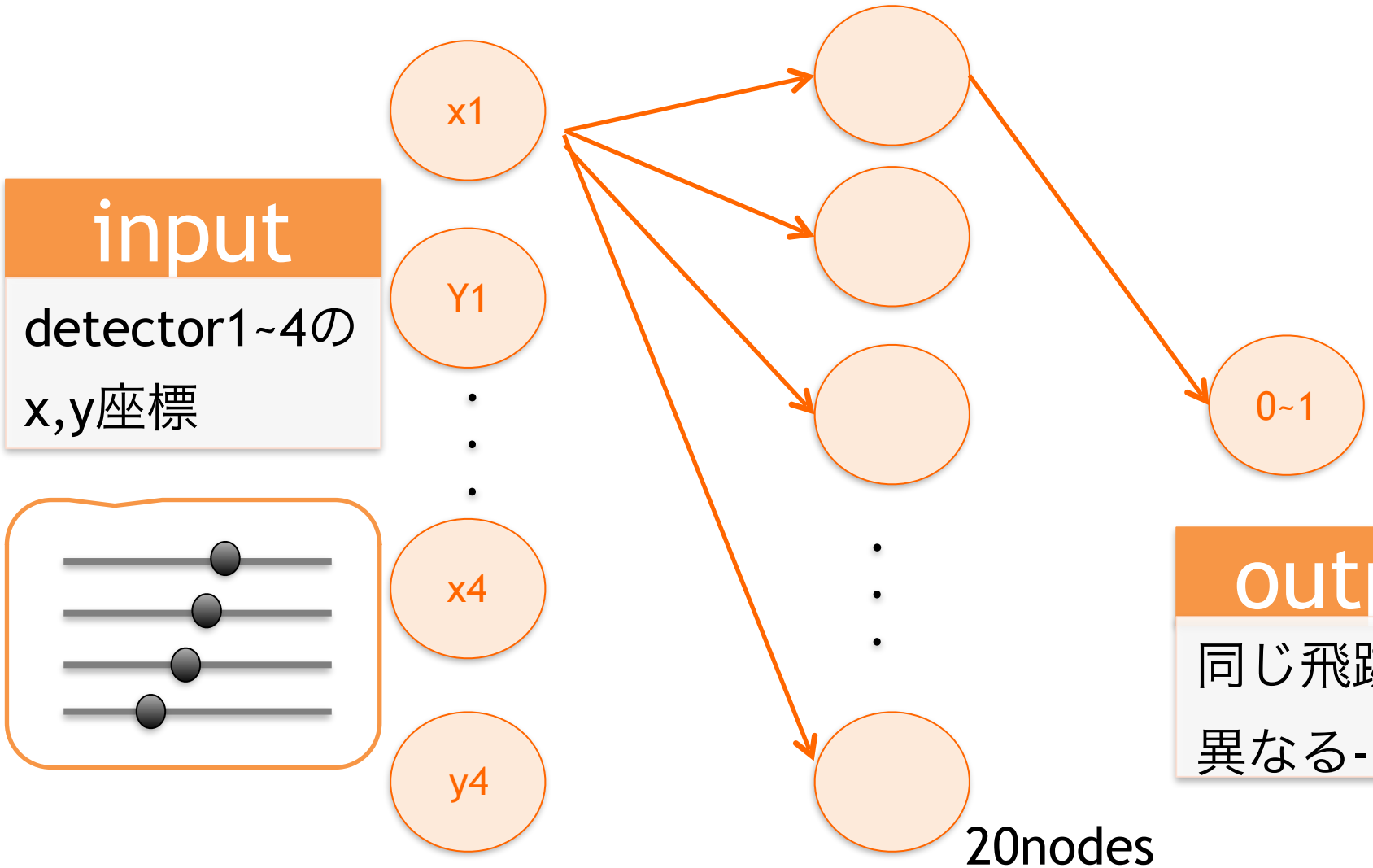


Neural Network



訓練により適切な出力が得られるように重みを計算する

4 Hits Net



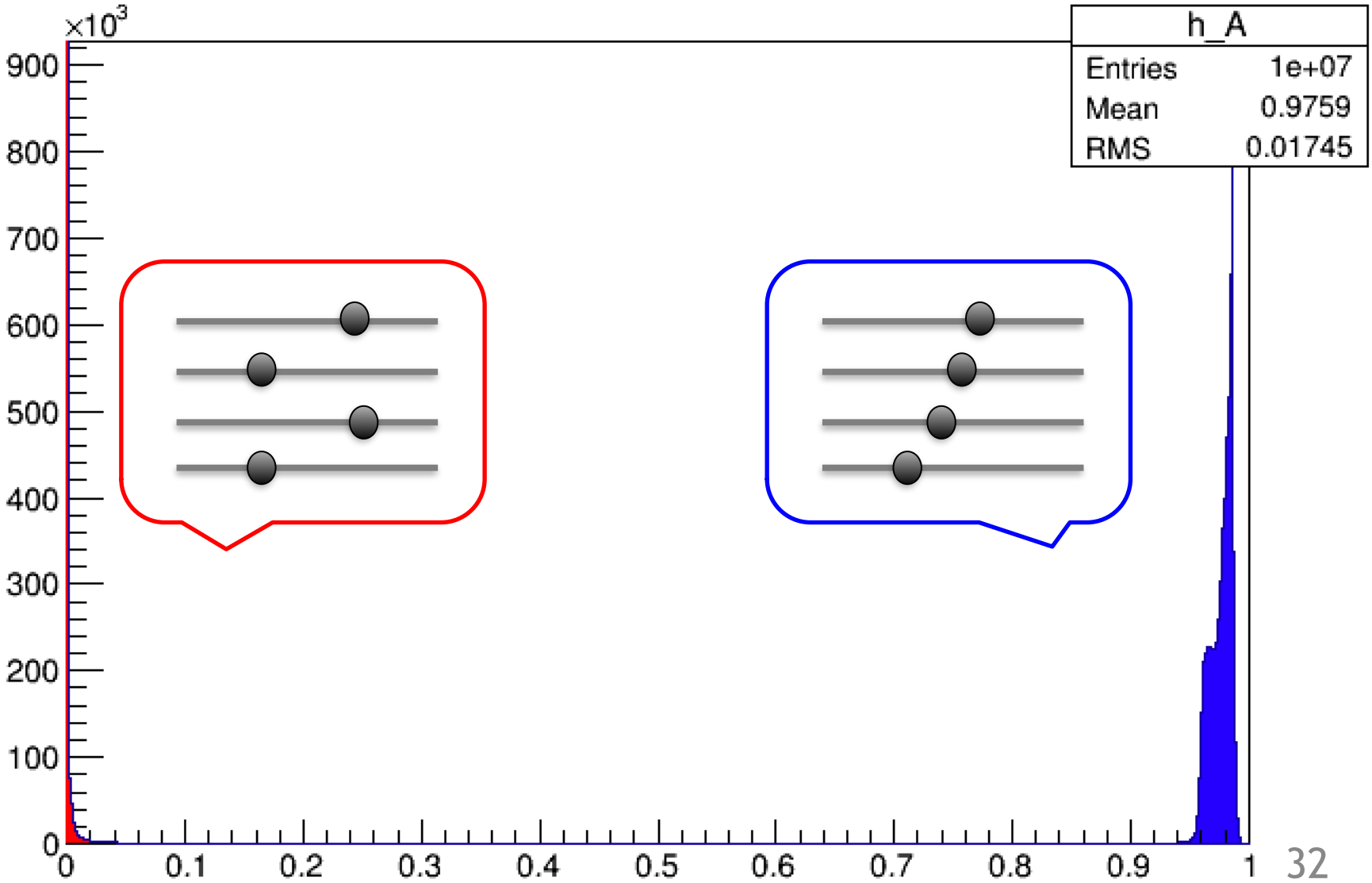
input

detector1~4の
x,y座標

output

同じ飛跡->1
異なる->0

Training Result

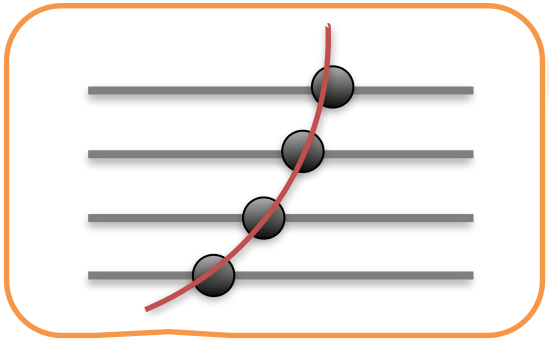


Hit on Track Net

input

detector1~3の
3点を通る飛跡
のparameter

detector4の
x,y座標



1/R

φ

x4

y4

0~1

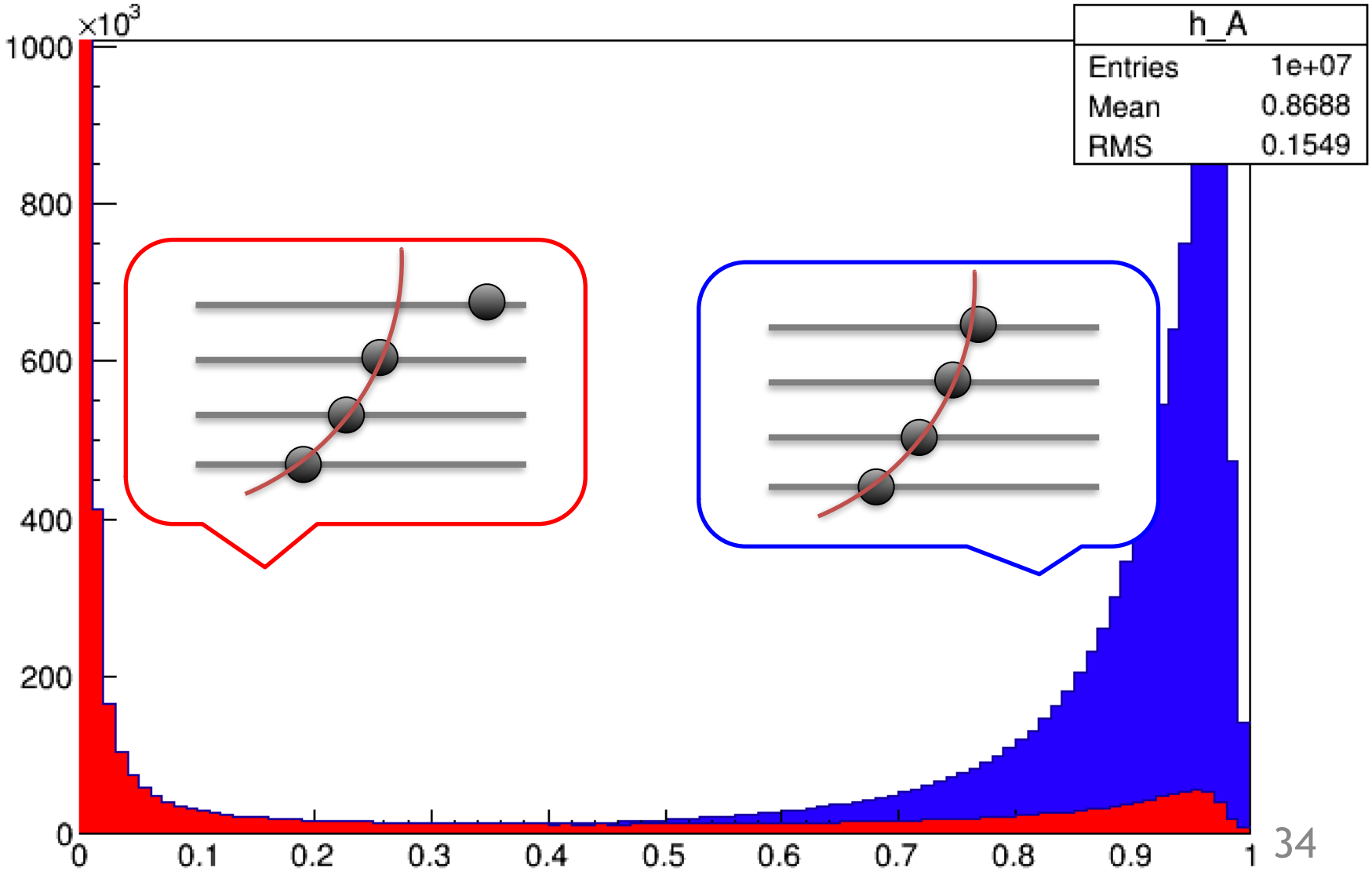
output

同じ飛跡->1

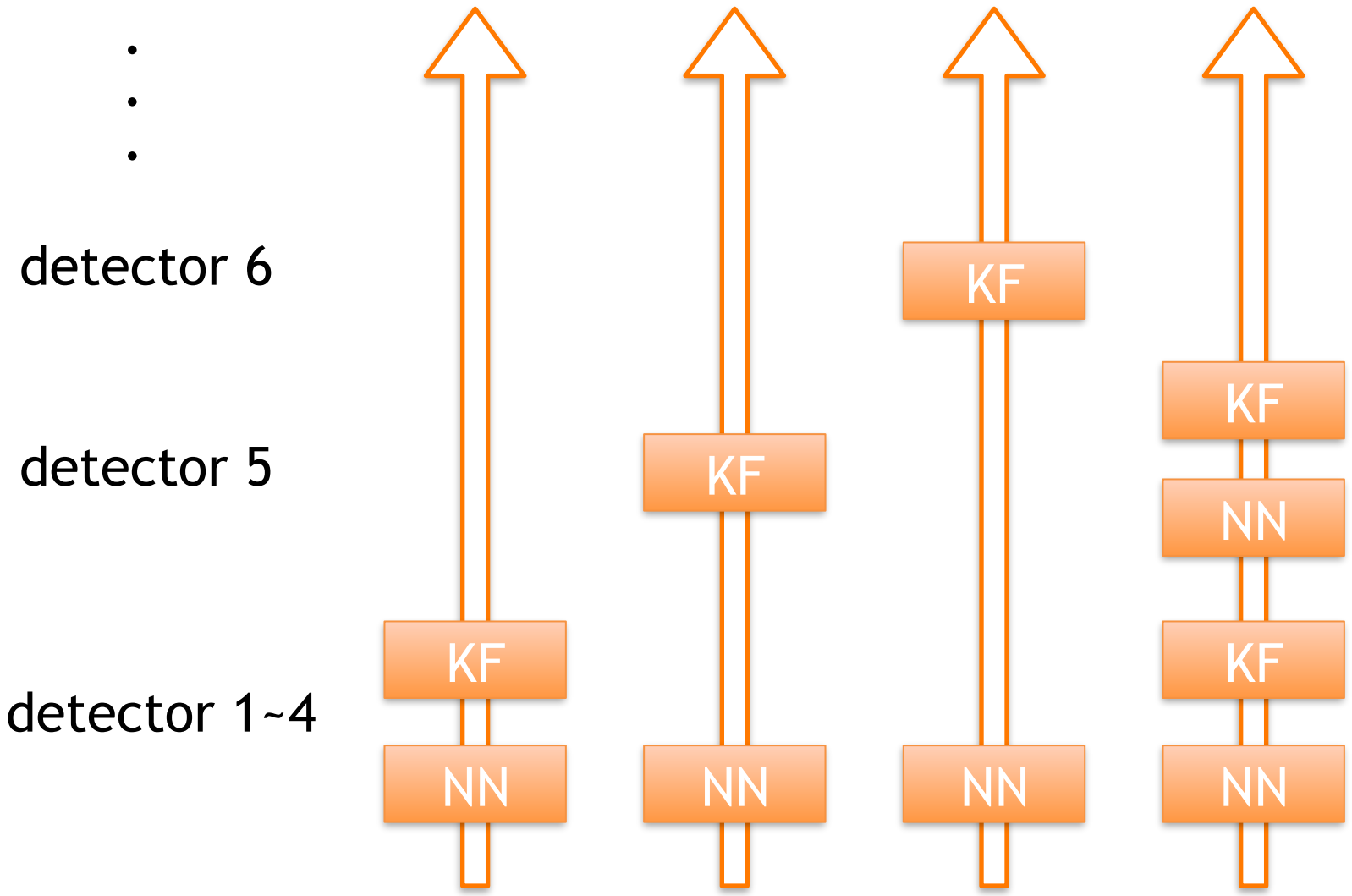
異なる->0

30nodes

Training Result

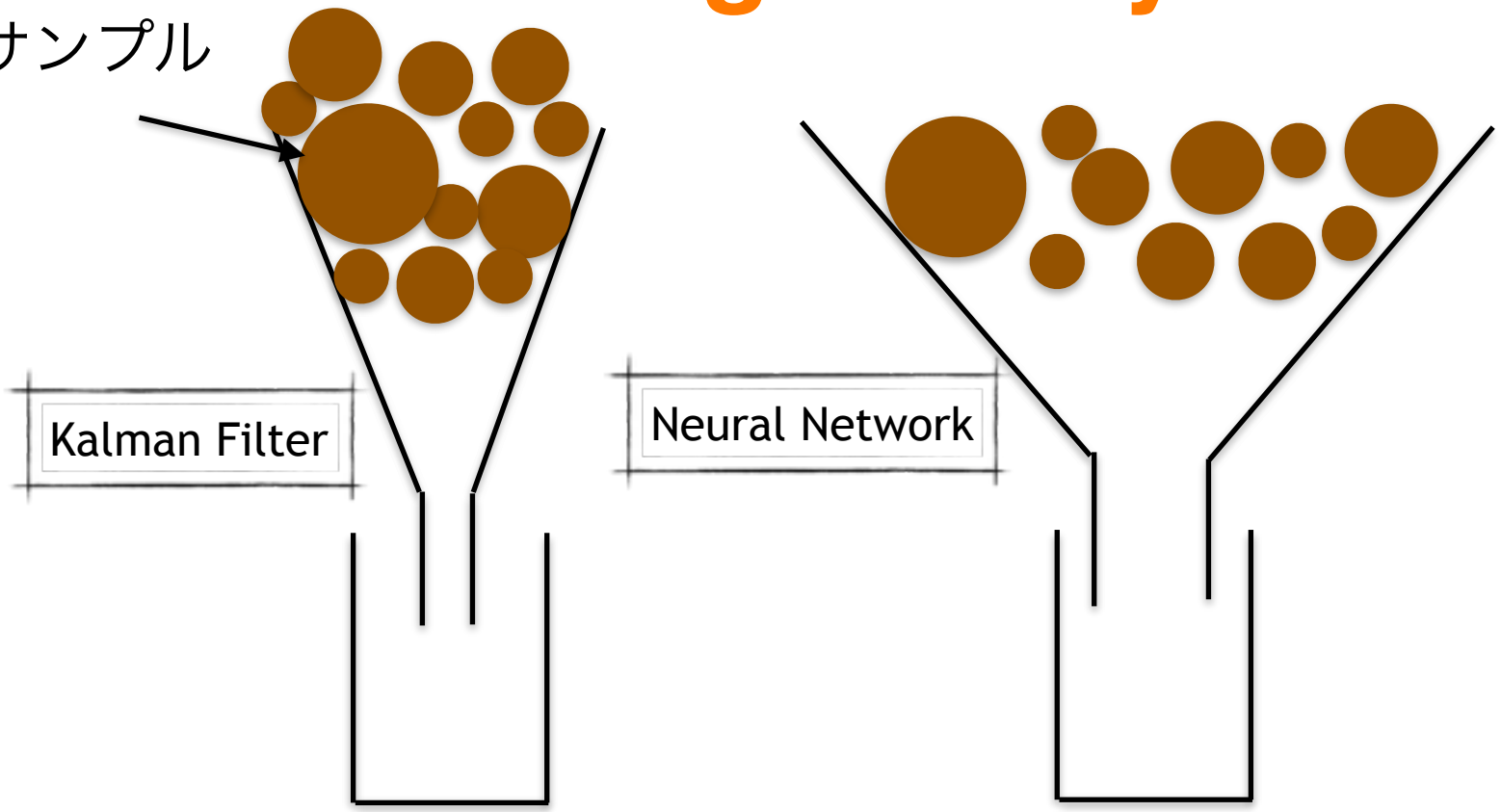


How to Use the Network?



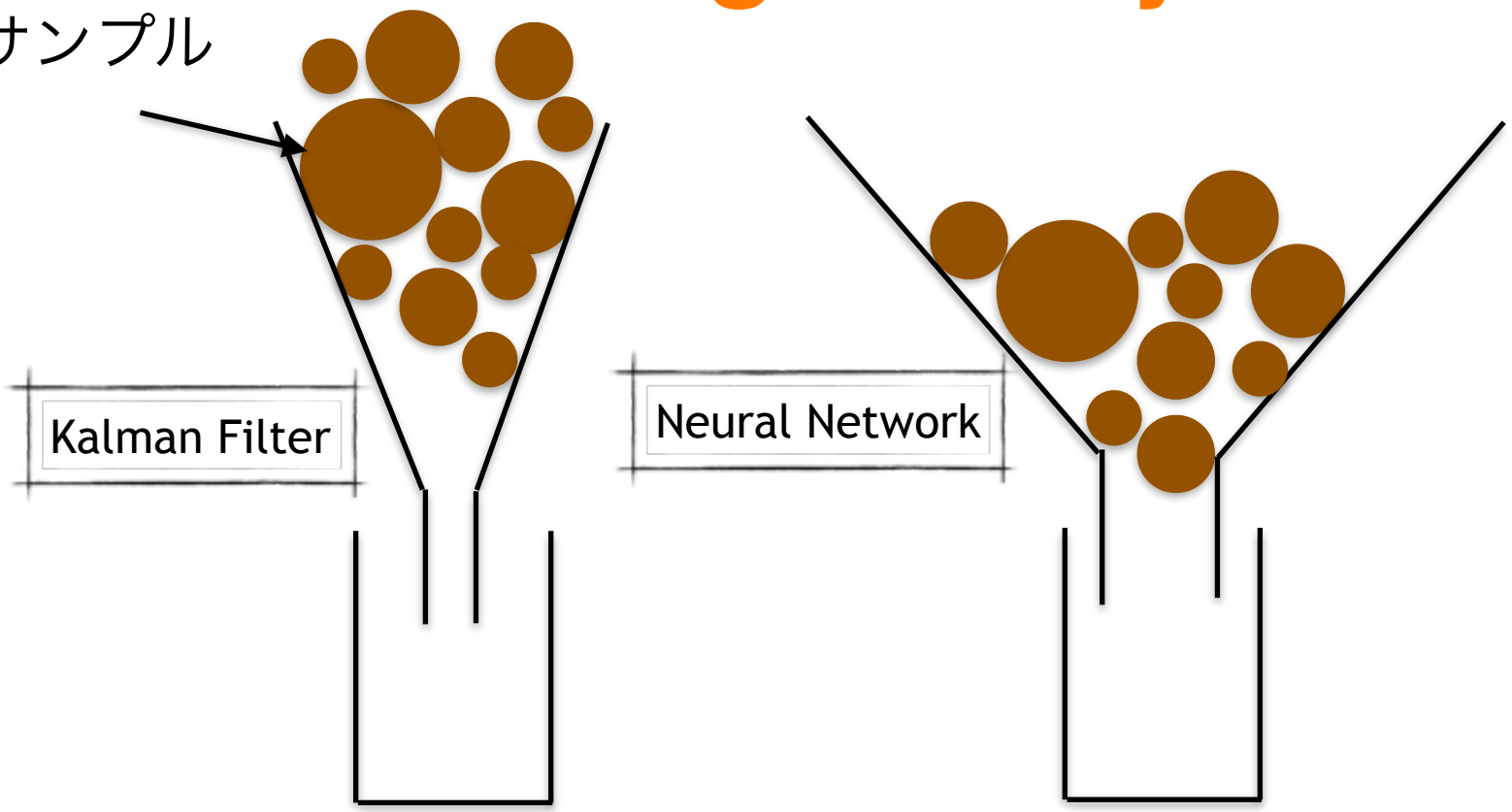
Filtering Quality

計算サンプル



Filtering Quality

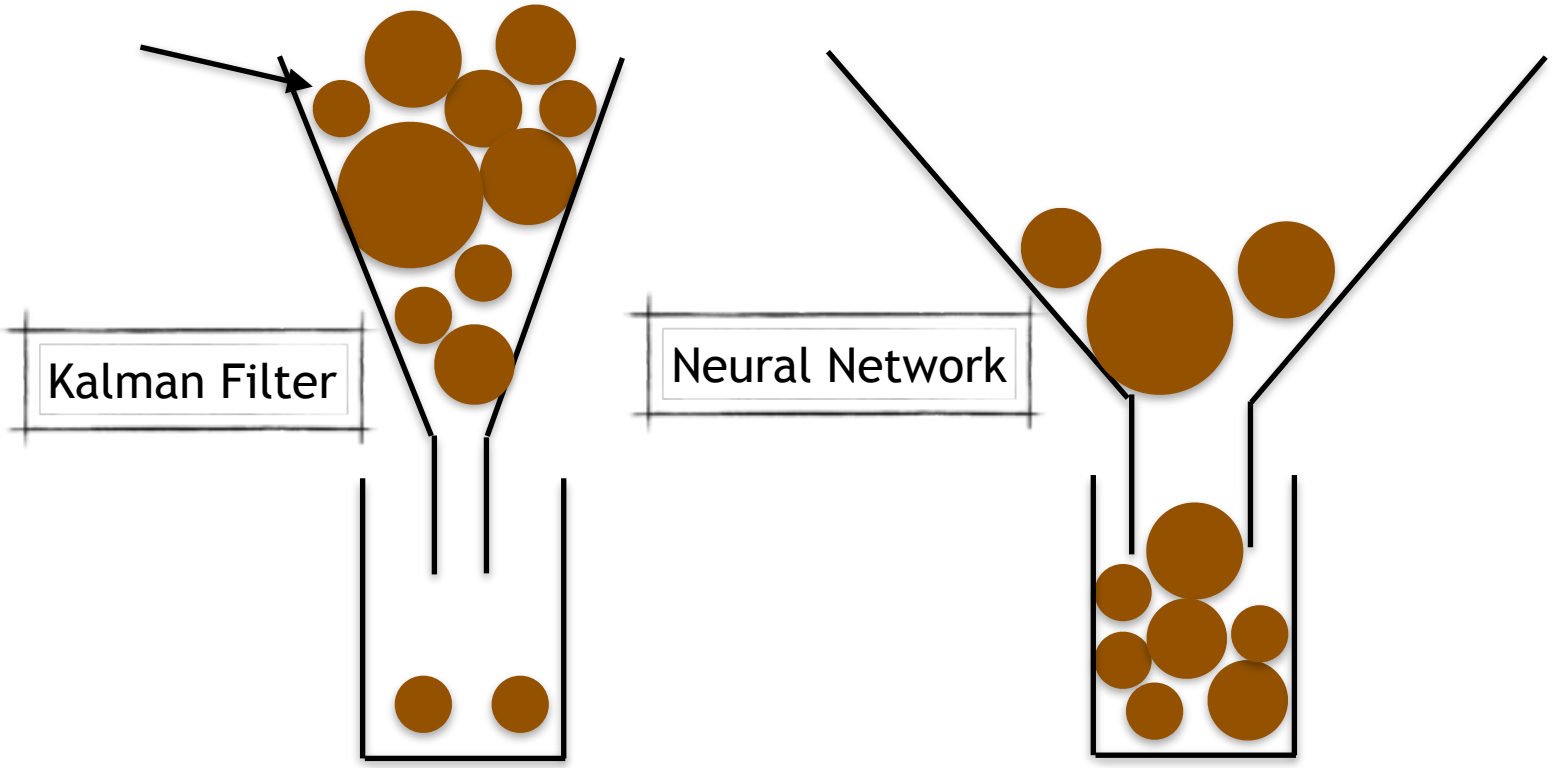
計算サンプル



速さ　　Kalman Filter < Neural Network

Filtering Quality

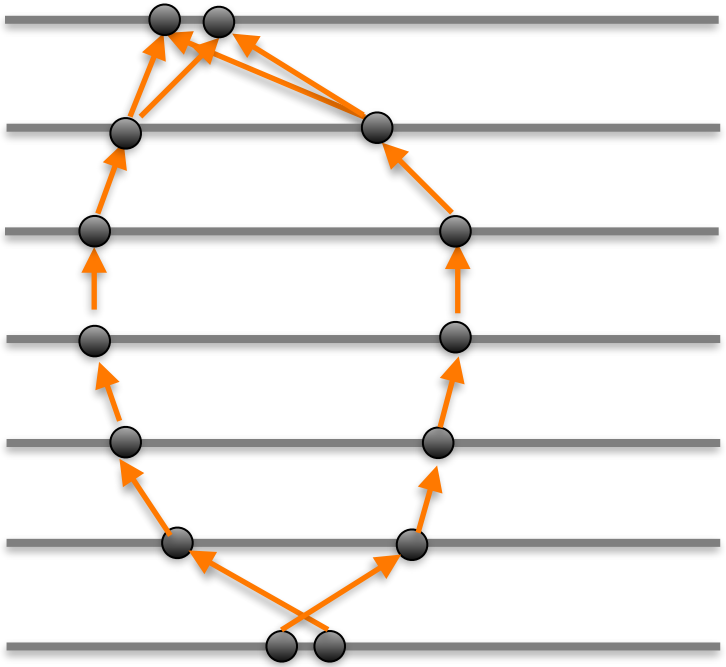
計算サンプル



速さ　　Kalman Filter < Neural Network

精度　　Kalman Filter > Neural Network

Kalman Filter

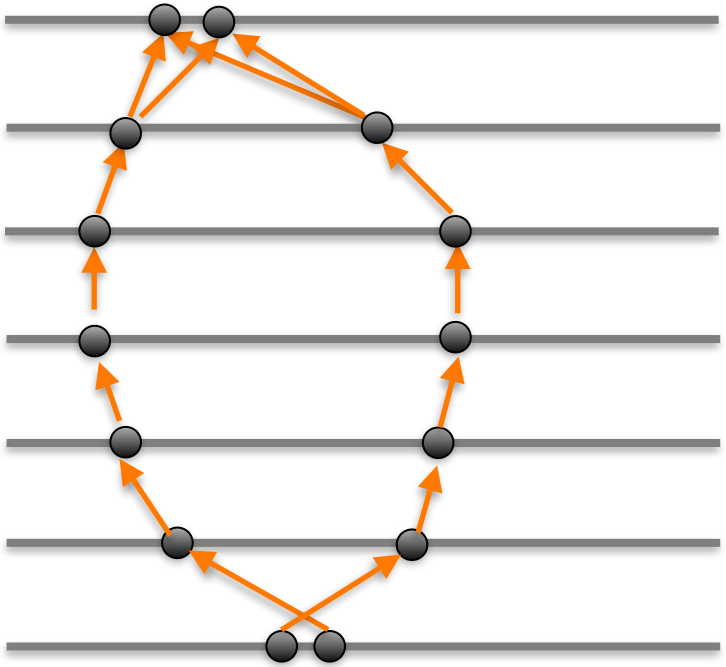


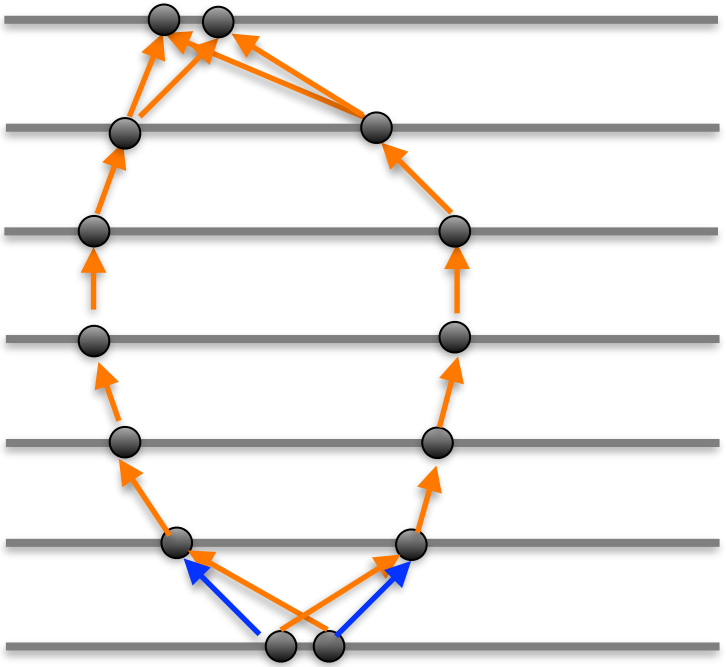
$$2^4 + 2 \times 2 + 2 \times 2 + 2 \times 2 = 28$$

Filtering結果の#track

元々の#track

Kalman Filter





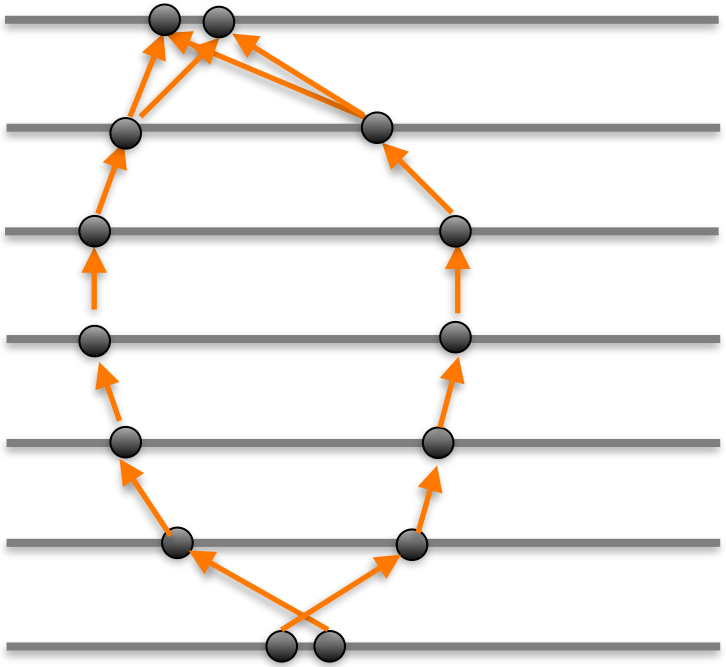
$2^4 + 2 \times 2 + 2 \times 2 + 2 \times 2 = 28$

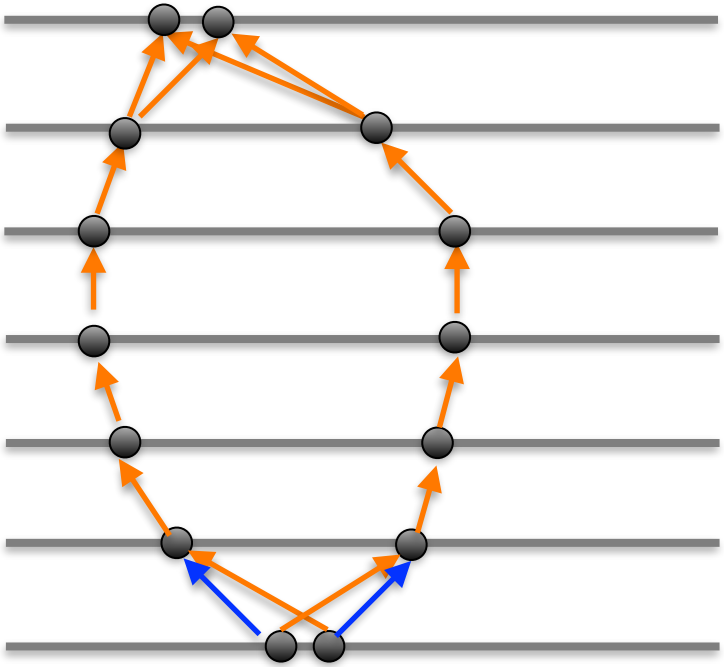
$2^4 + 4 \times 2 + 4 \times 2 + 4 \times 2 = 40$

Filtering結果の#track

元々の#track

Kalman Filter



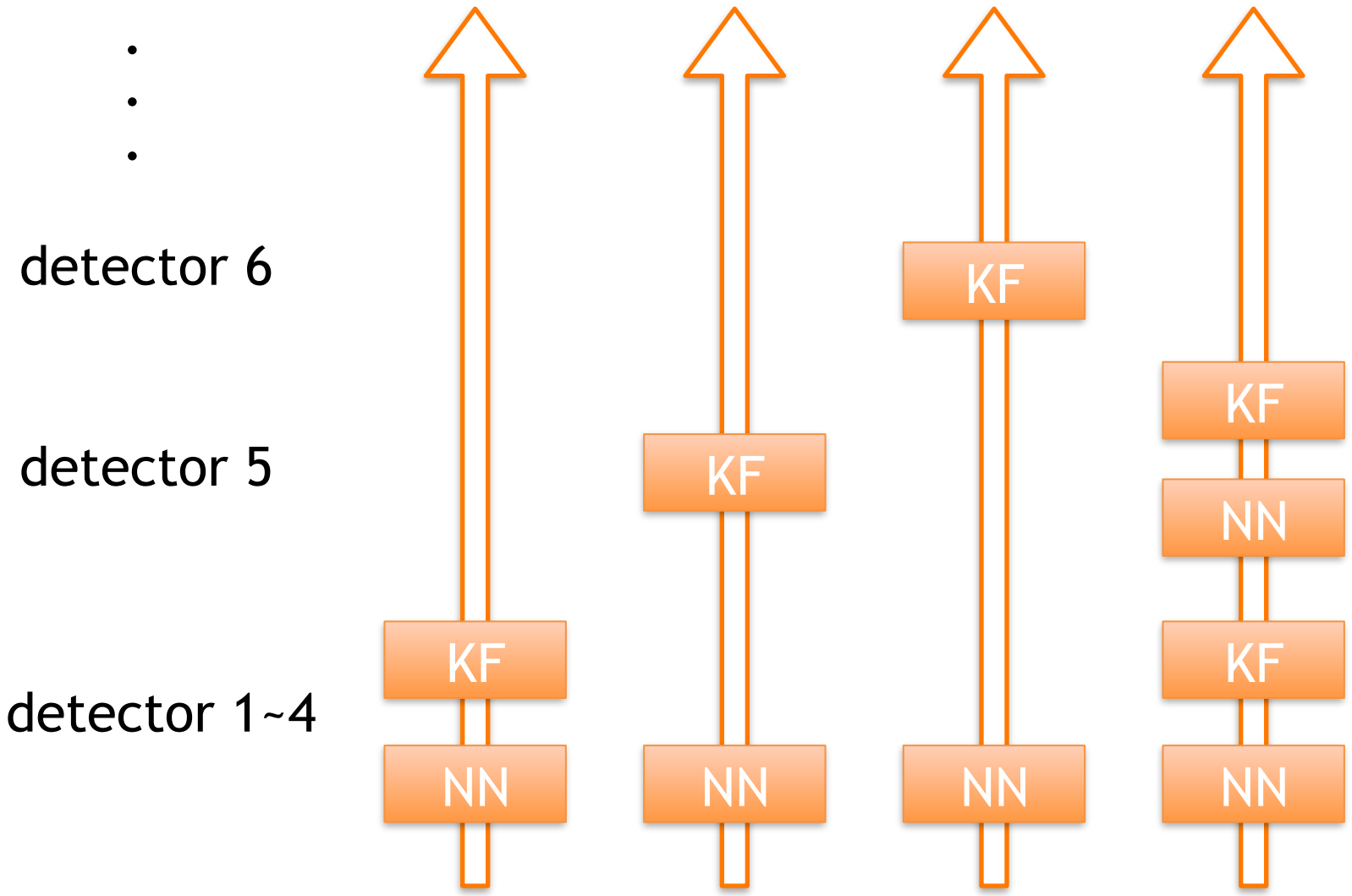


$2^4 + 2 \times 2 + 2 \times 2 + 2 \times 2 = 28$

$2^4 + 4 \times 2 + 4 \times 2 + 4 \times 2 = 40$

早い段階で候補を減らす必要がある

How to Use the Network?



How to Use the Network?

-
-
-

detector 6

detector 5

detector 1~4

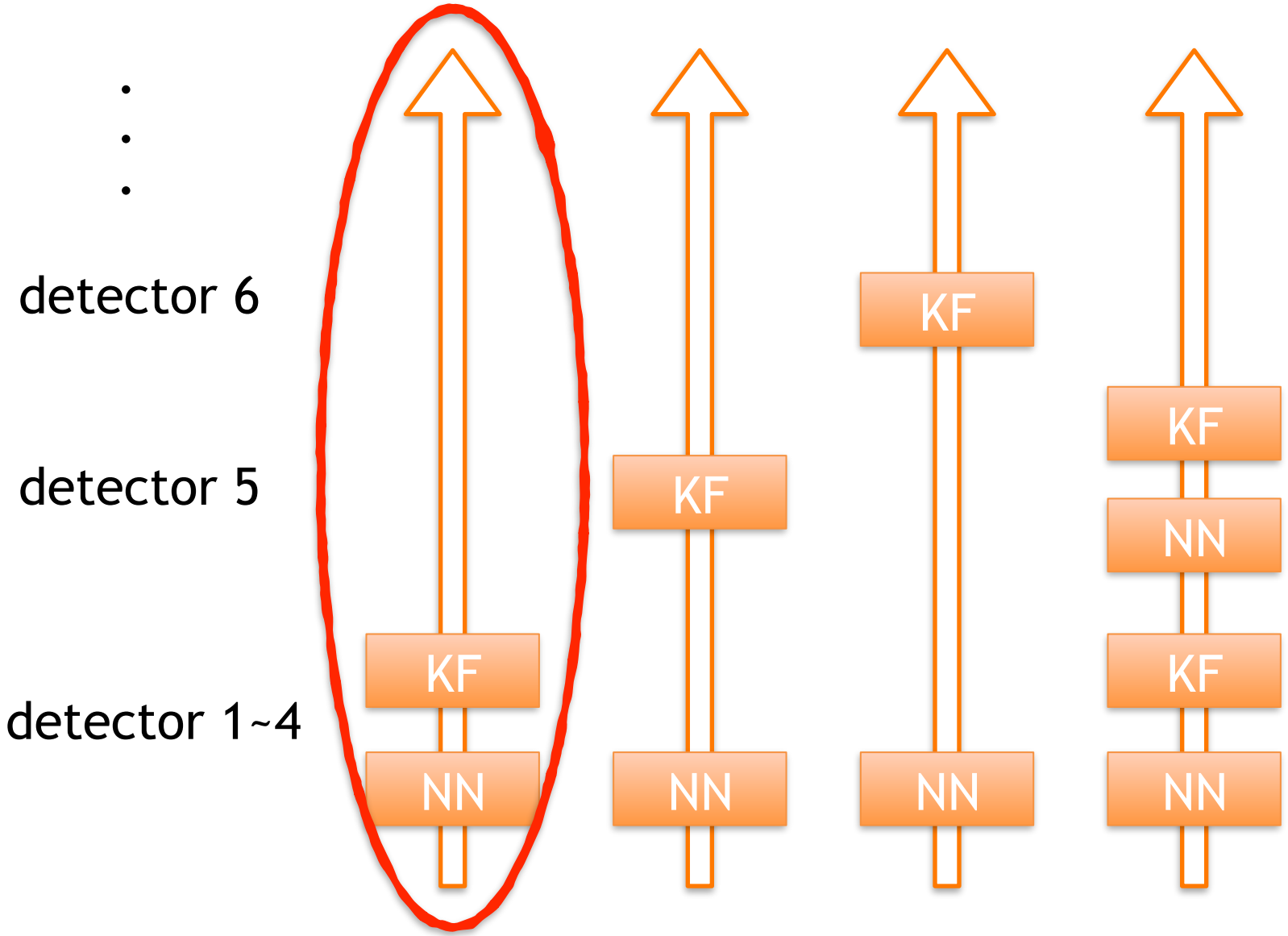
早い段階で候補を減らす必要がある

速さ Kalman Filter < Neural Network
精度 Kalman Filter > Neural Network

KF
NN
KF
NN
KF
NN

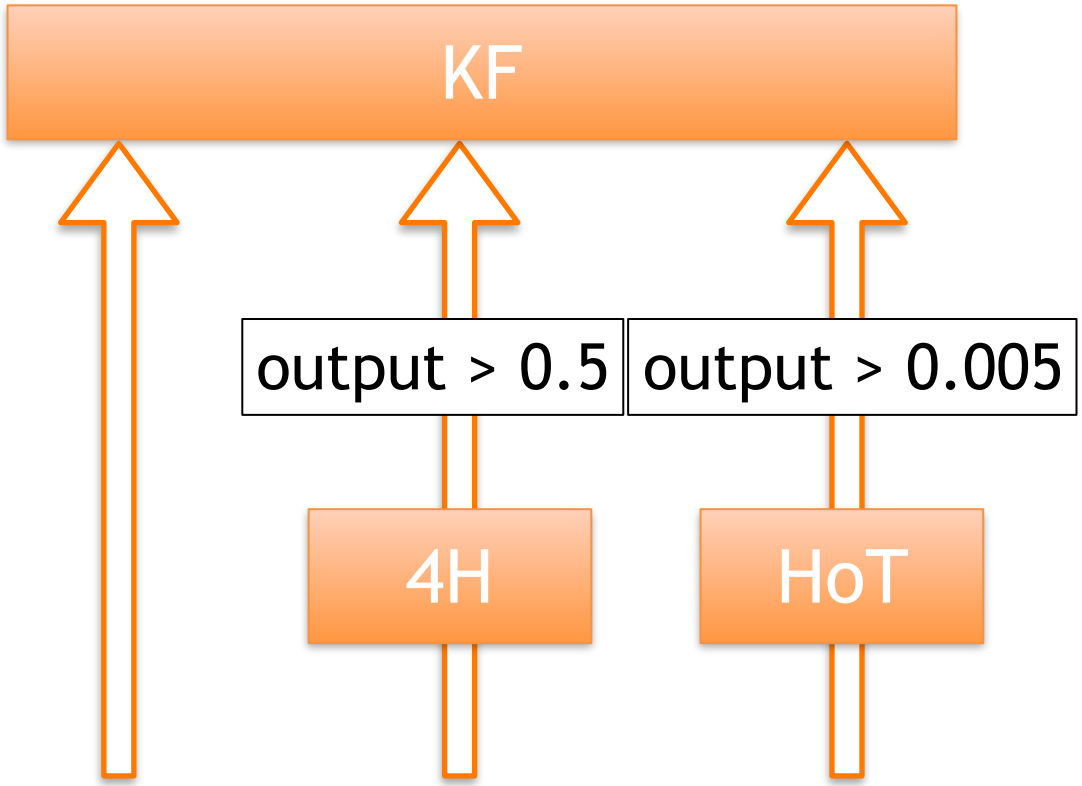
チョウチンアンコウ

How to Use the Network?



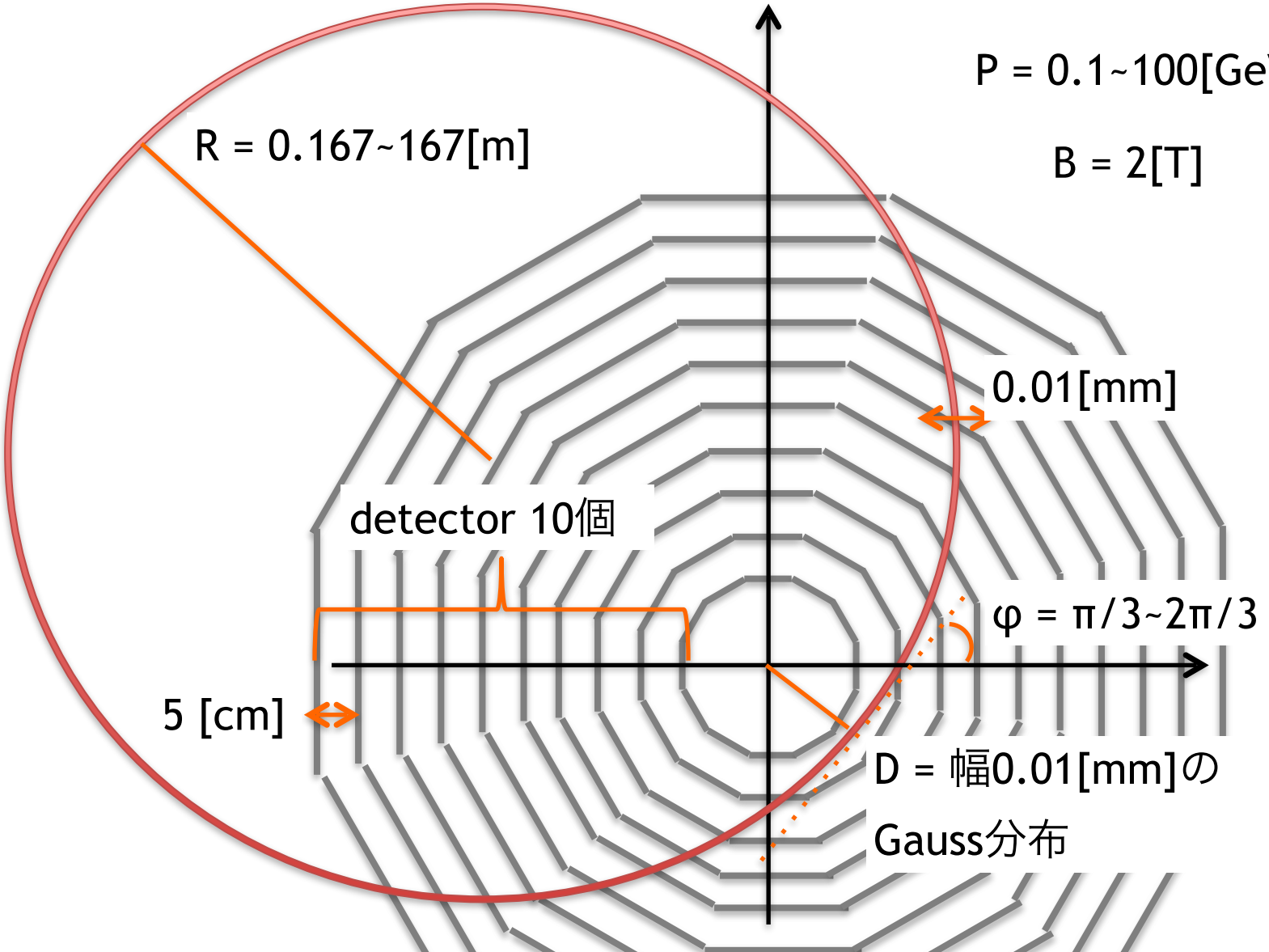
Calculation Methods

飛跡を計算

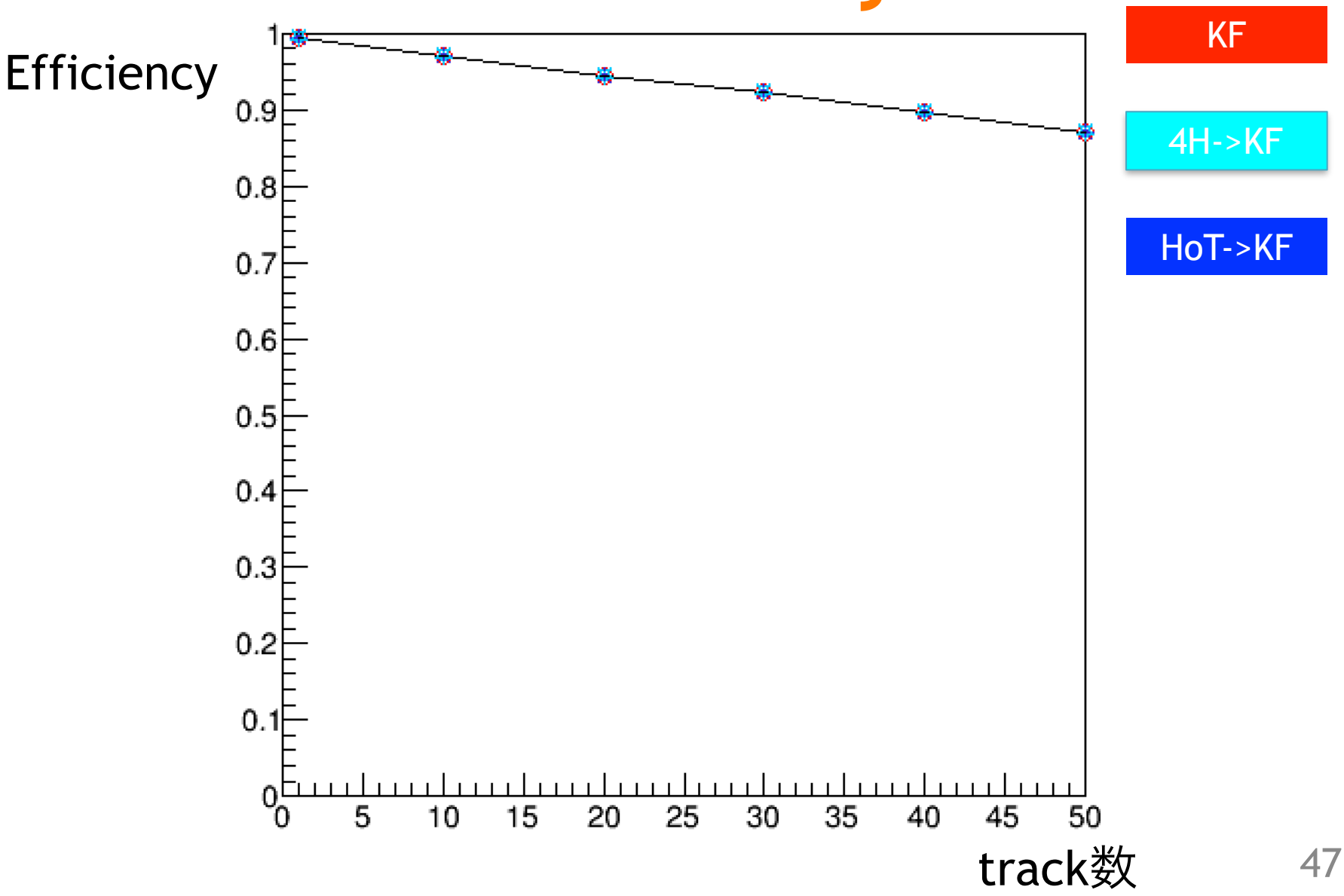


detector1~4のhitsの組み合わせが同一飛跡に乗るか判定

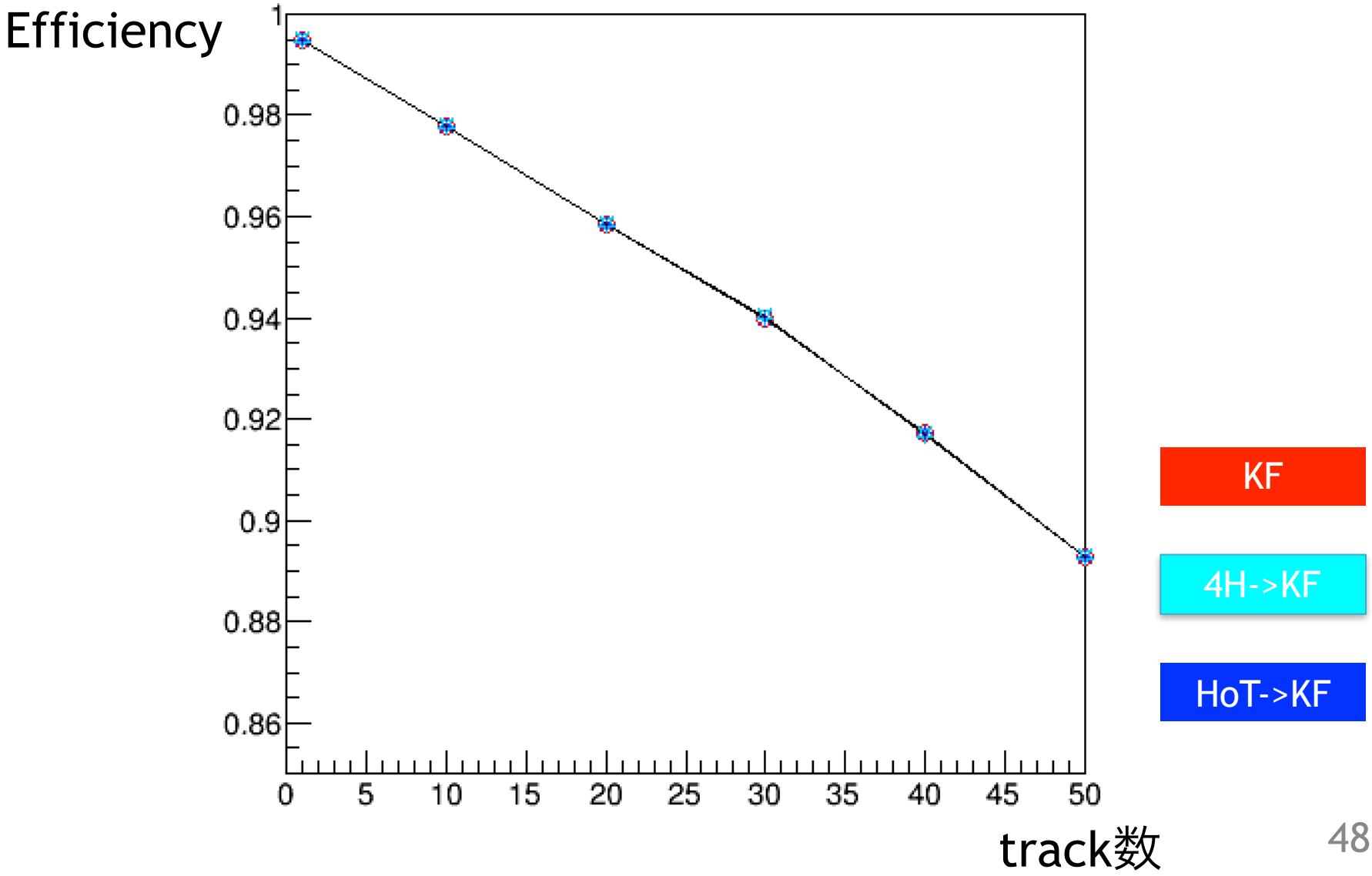
Setting for Reconstruction



Efficiency

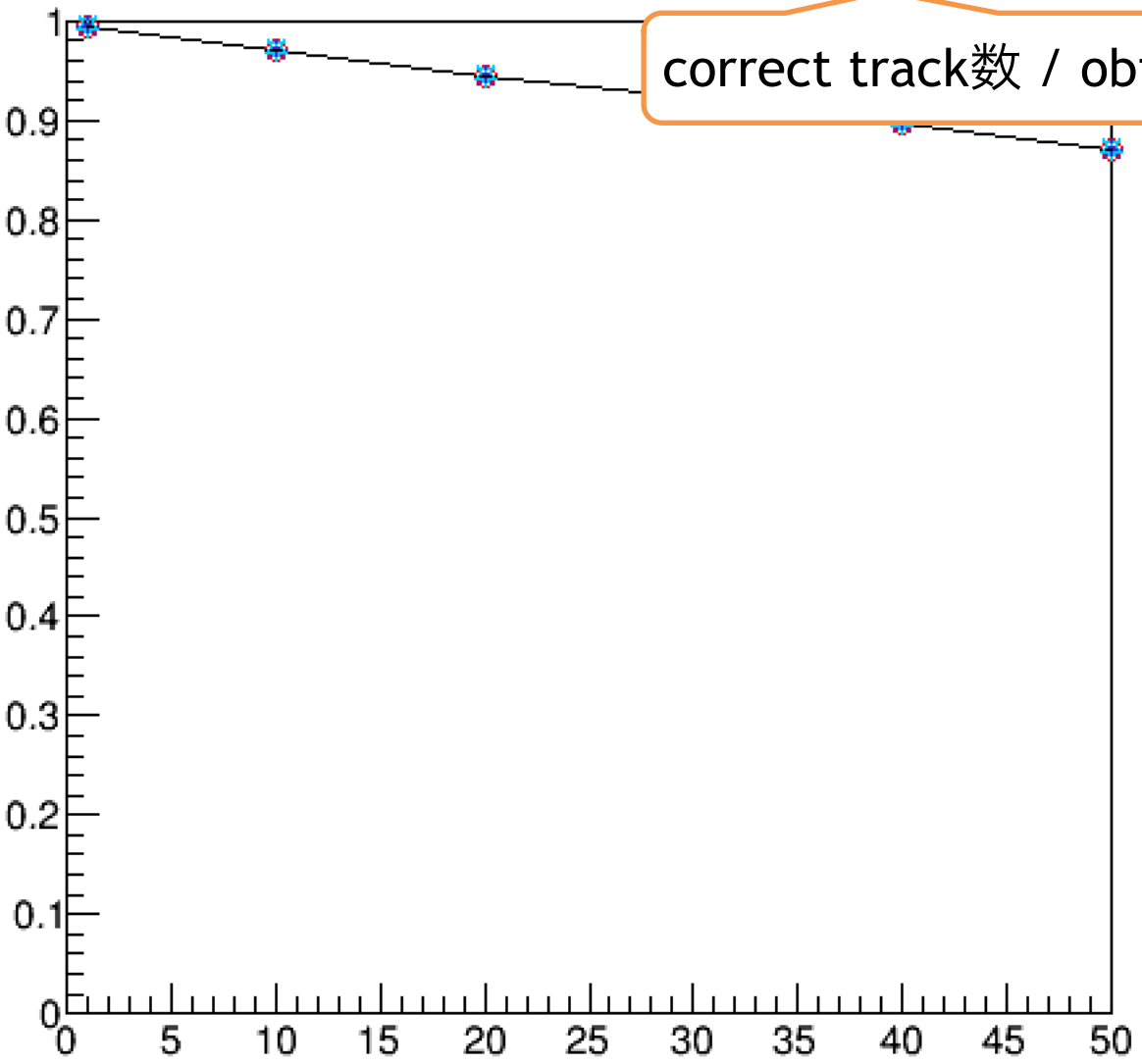

Efficiency
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
5
10
15
20
25
30
35
40
45
50
track数
KF
4H->KF
HoT->KF

Efficiency


Efficiency
1
0.98
0.96
0.94
0.92
0.9
0.88
0.86
0
5
10
15
20
25
30
35
40
45
50
track数
KF
4H->KF
HoT->KF

Accuracy Rate

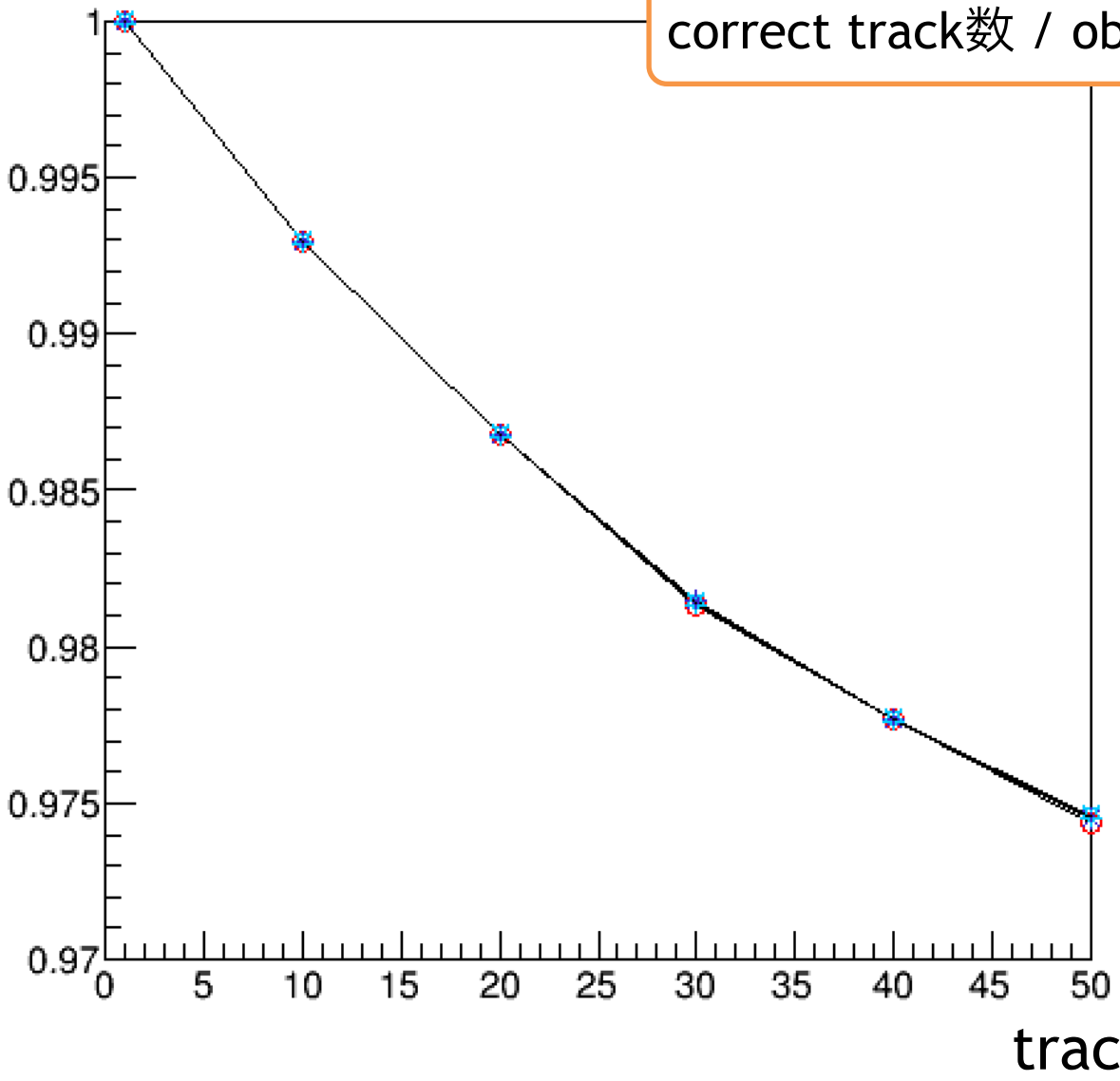
Accuracy Rate



Accuracy Rate

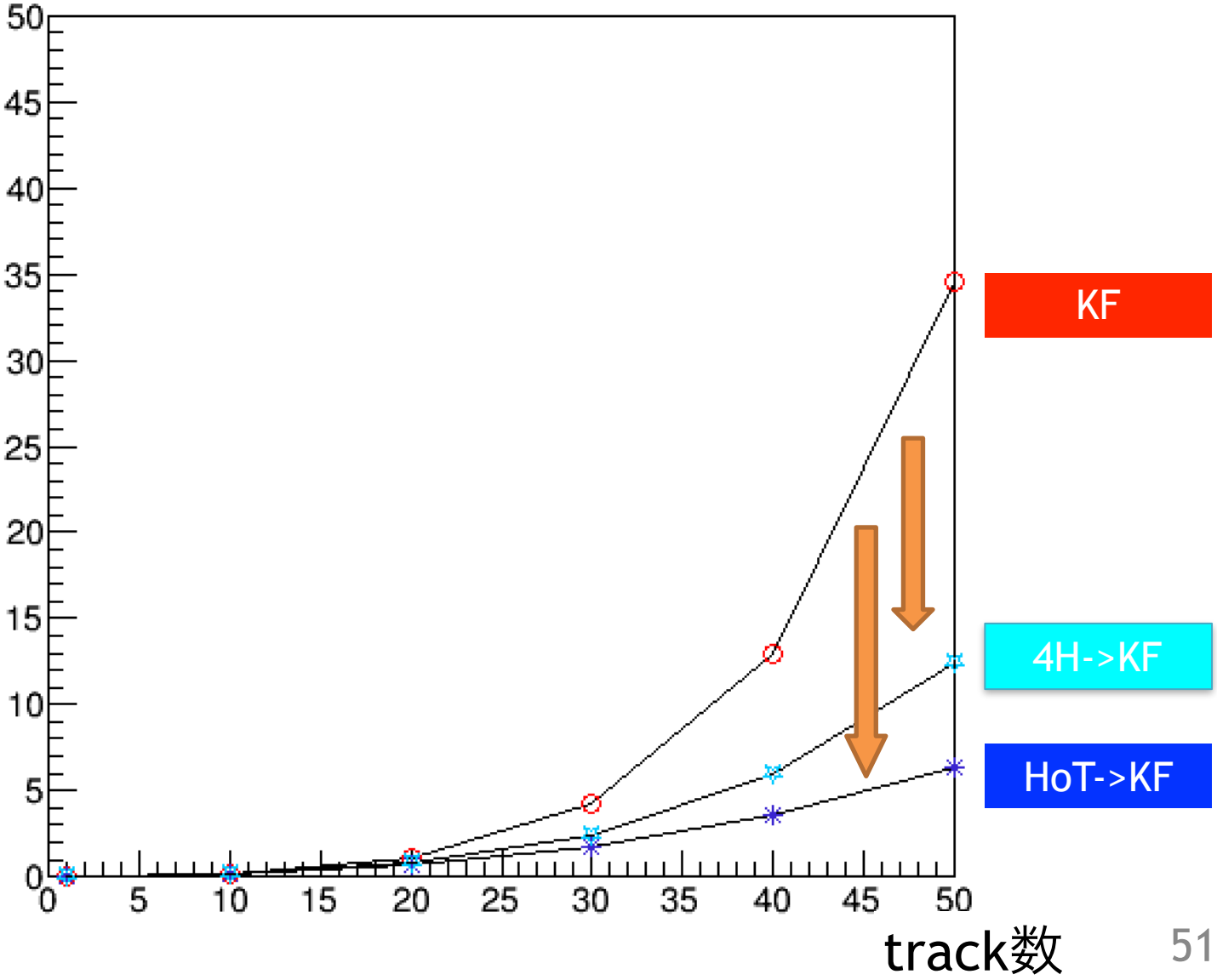
Accuracy Rate

correct track数 / obtained track数



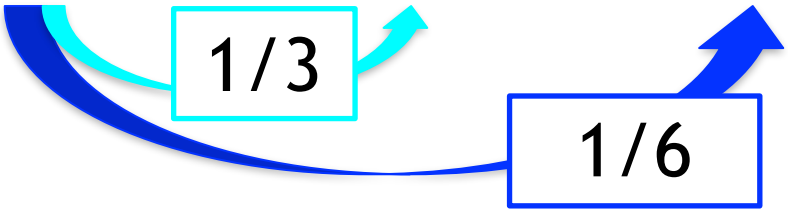
Calculation Time

Calculation Time [s]

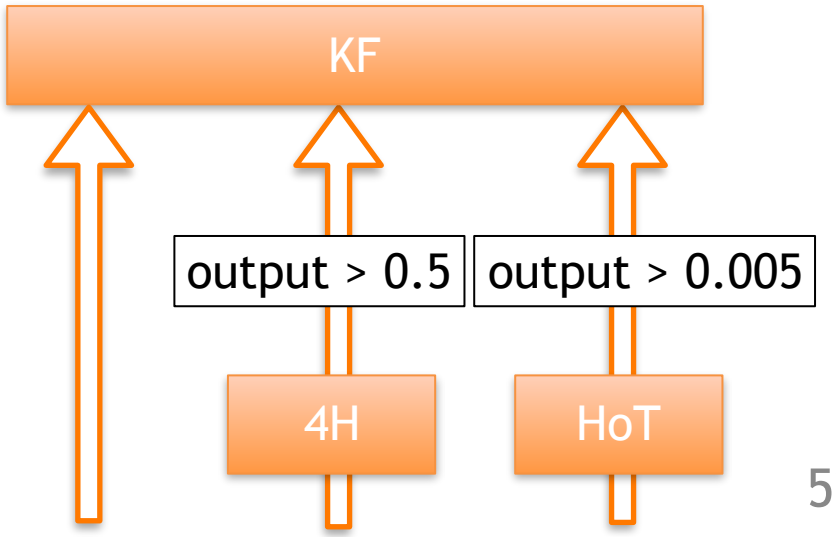


Calculation Time for 50 tracks

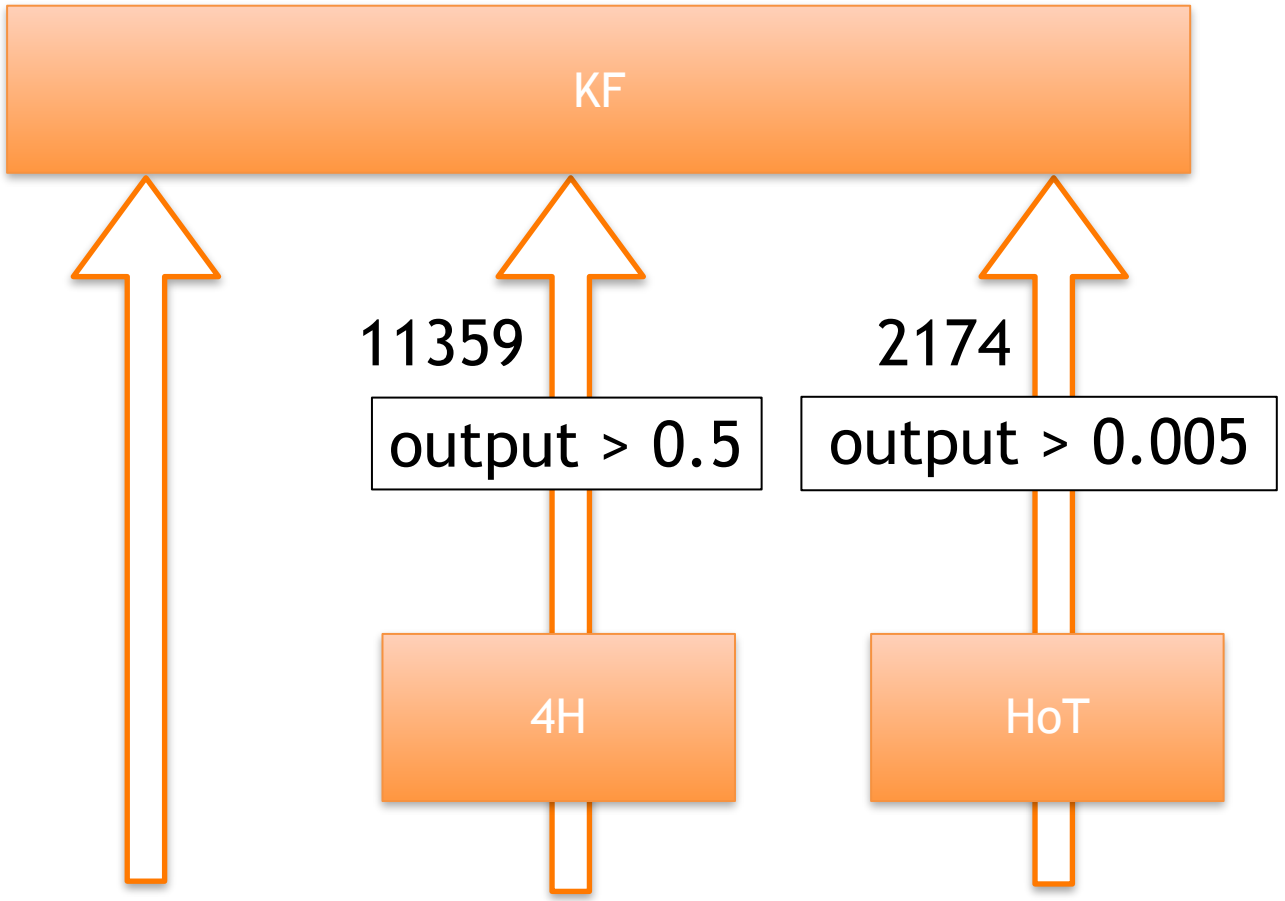
KF	4H -> KF	HoT -> KF
34.55s	12.40s	6.35s



計算する組み合わせの数を
大幅にカットできる



Inputs for 50 tracks

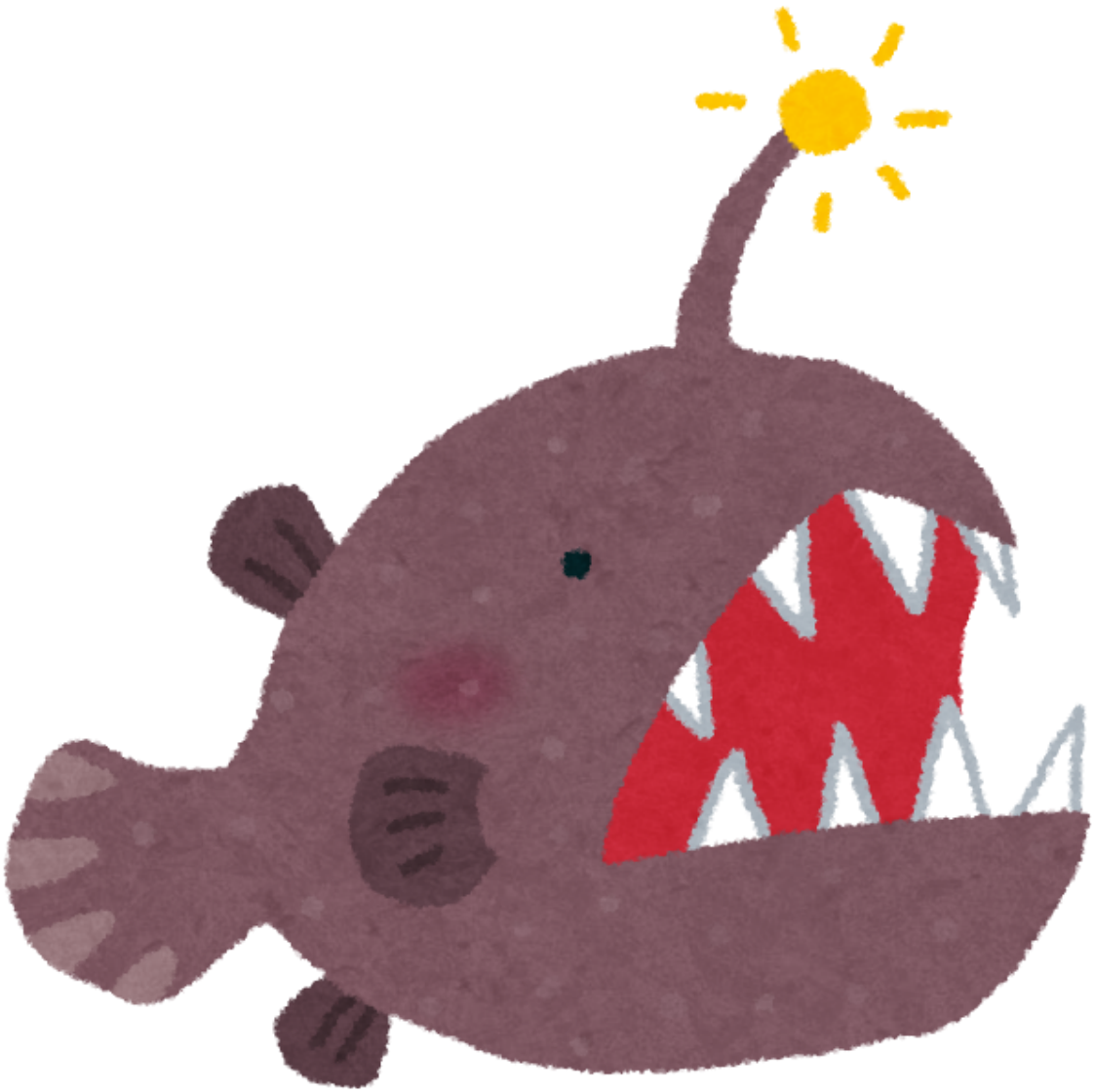


50^4 = 6250000 6250000 6250000

Conclusion

✓Neural Networkを用いることにより、飛跡再構成を同等の効率で短時間に行うことができた

➢実際の実験条件に近づける

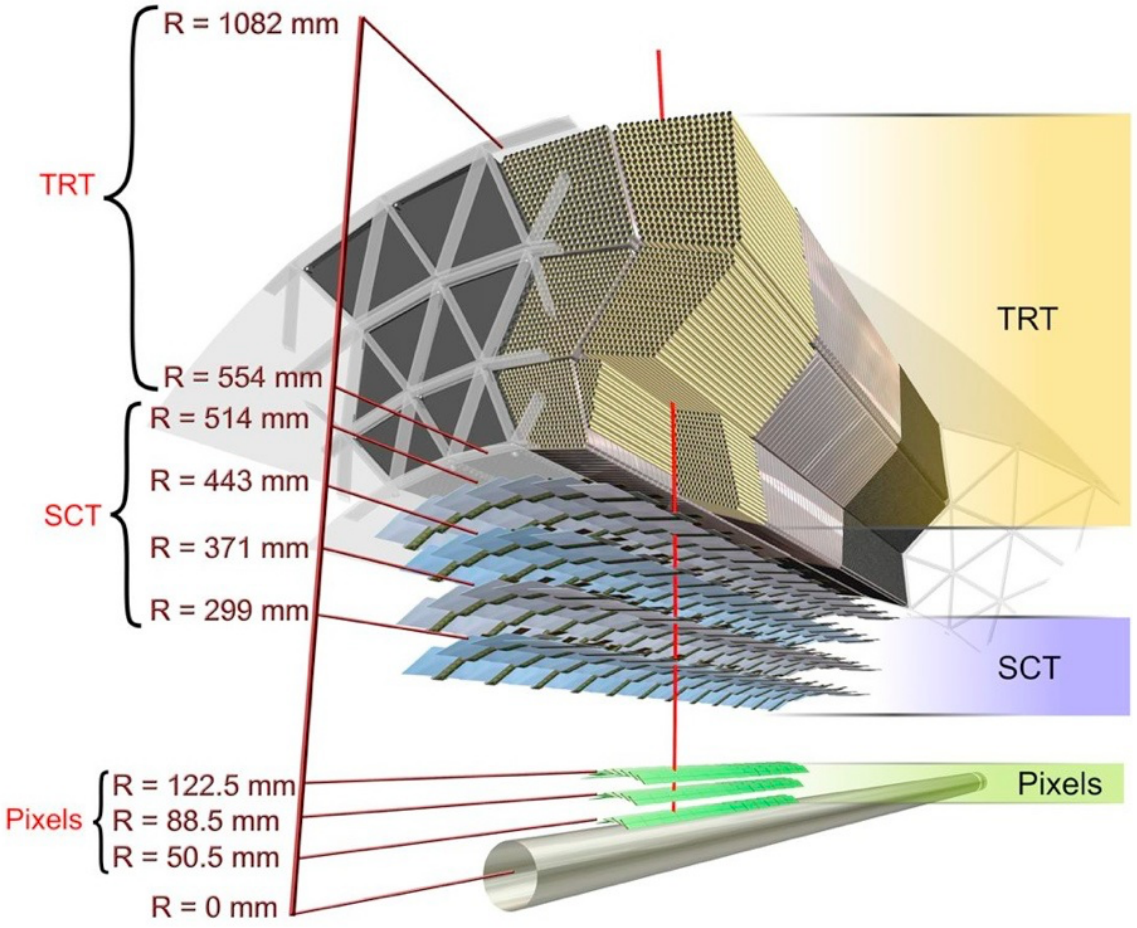


Backup

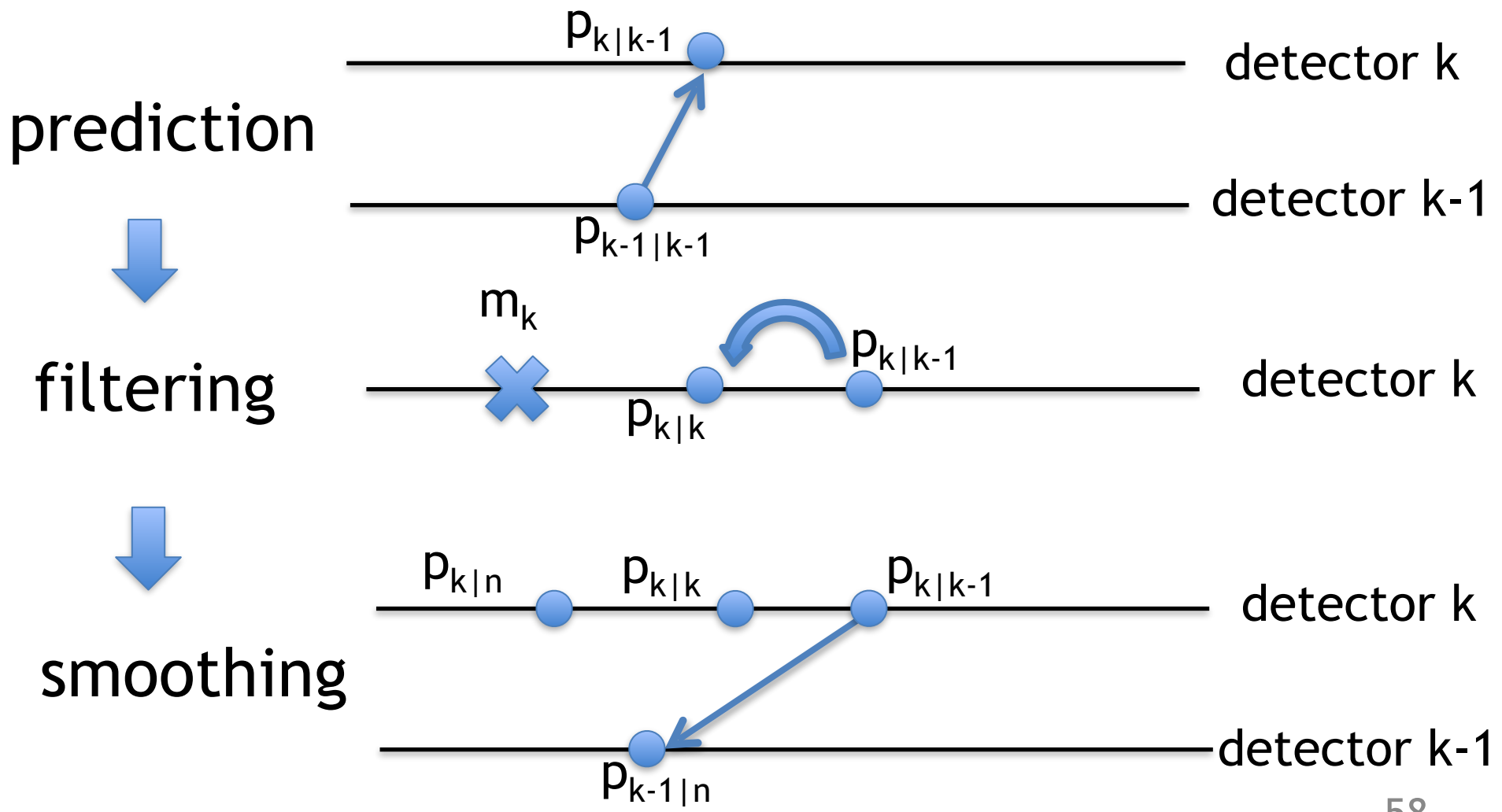
参考文献

- 足立修一 ・ 丸田一郎 (2012) 『カルマンフィルタの基礎』　東京電機大学出版局
- R.Fruihwirth (1987) “Application of Kalman Filtering to Track and Vertex Fitting”, Nuclear Instruments and Methods in Physics Research A262 p.444-450
- 斎藤康毅 (2016) 『ゼロから作るDeep Learning』 オライリー・ジャパン

ATLAS の内部飛跡検出器



Kalman Filter



Kalman Filter

n次元状態ベクトルx(k)、観測データy(k)が

$$\mathbf{x(k+1) = Ax(k) + Bv(k)}$$

$$\mathbf{y(k) = C^T x(k) + w(k)}$$

という状態空間モデルに従うとする。

ただし、Aはn×n行列、Bはn×r行列、Cはp×n行列

vは平均値０、共分散行列Qのr次元システム雑音ベクトル。wは平均値０、共分散行列Rのp次元観測雑音ベクトル。

更新アルゴリズム

- 予想ステップ

  事前状態推定値：$x_{k|k-1} = A\, x_{k-1|k-1}$

  事前誤差共分散行列：$P_{k|k-1} = A\, P_{k-1|k-1}\, A^T + B\, Q\, B^T$

- フィルタリングステップ

  カルマンゲイン行列：$G_k = P_{k-1|k-1}\, C^T \left(C\, P_{k-1|k-1}\, C^T + R \right)^{-1}$

  状態推定値：$x_{k|k} = x_{k|k-1} + G_k \left(y(k) - C\, x_{k|k-1} \right)$

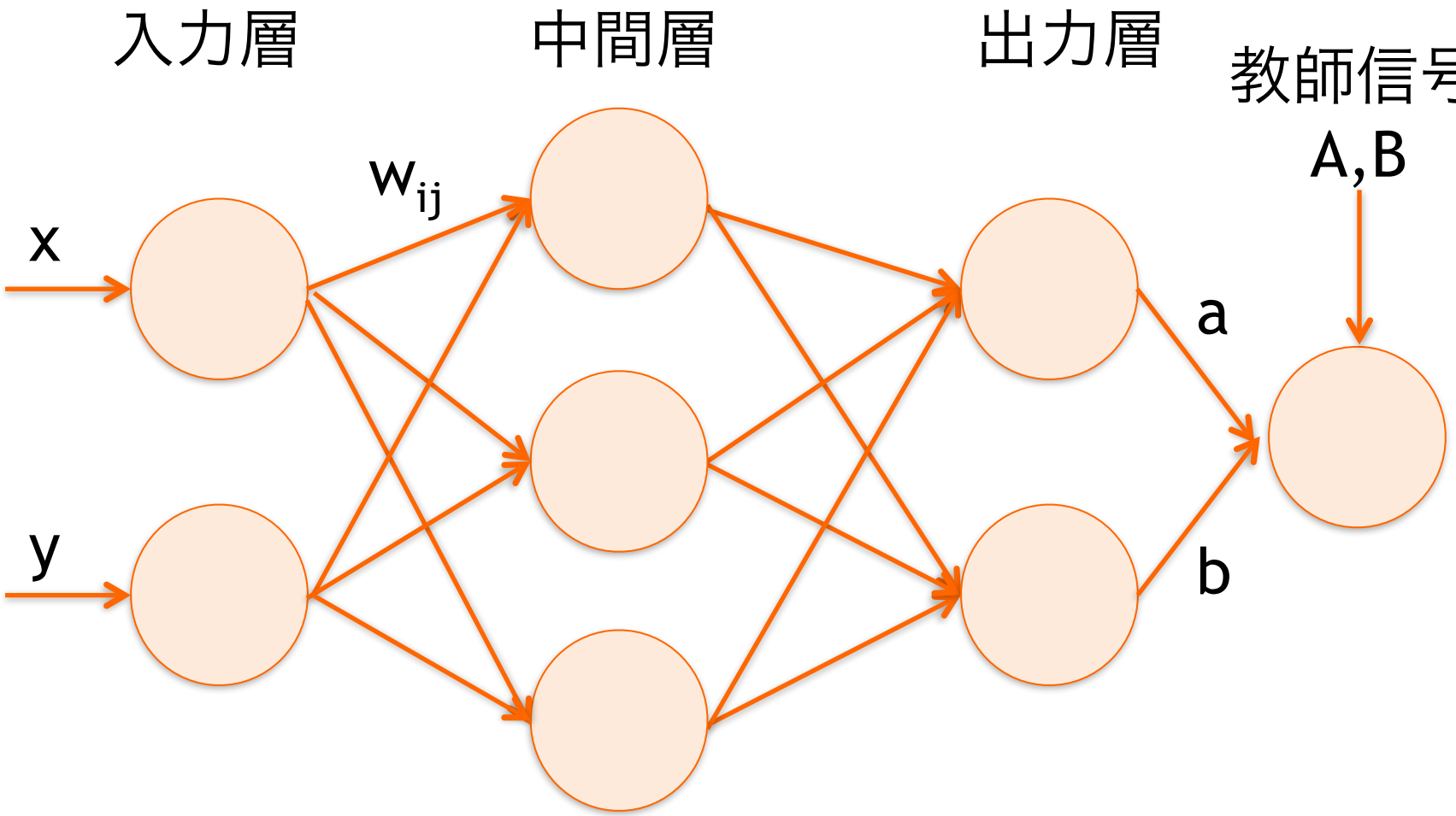
  事後誤差共分散行列：$P_{k|k} = \left(I - G_k\, C \right) P_{k|k-1}$

- **平滑ステップ**

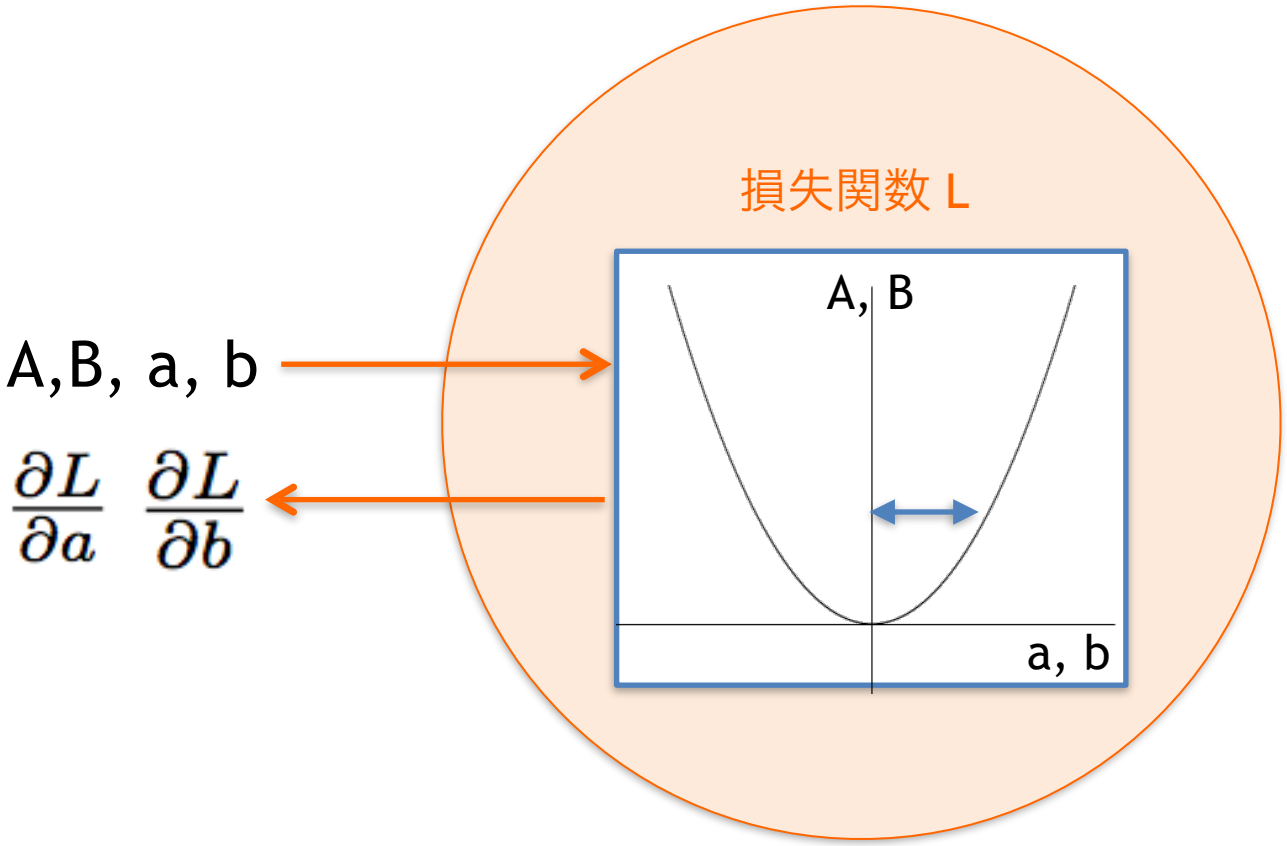
$$x_{k|n} = x_{k|k} + P_{k|k}\, A\, P_{k+1|k}^{-1} \left(x_{k+1|n} - x_{k+1|k} \right)$$

$$P_{k|n} = P_{k|k} + P_{k|k}\, A\, P_{k+1|k}^{-1} \left(P_{k+1|n} - P_{k+1|k} \right) \left(P_{k|k}\, A\, P_{k+1|k}^{-1} \right)^T$$

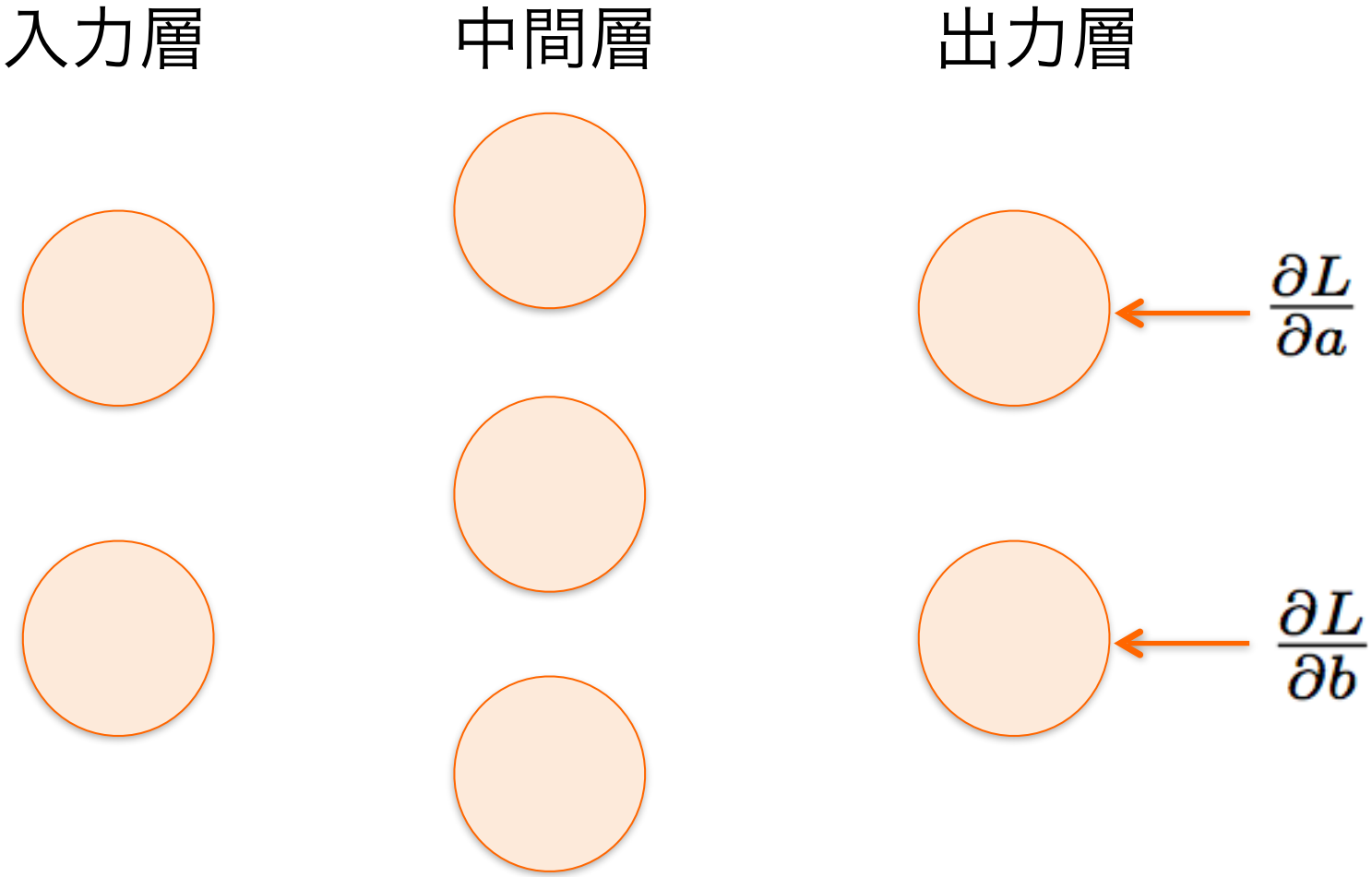
Training Phase



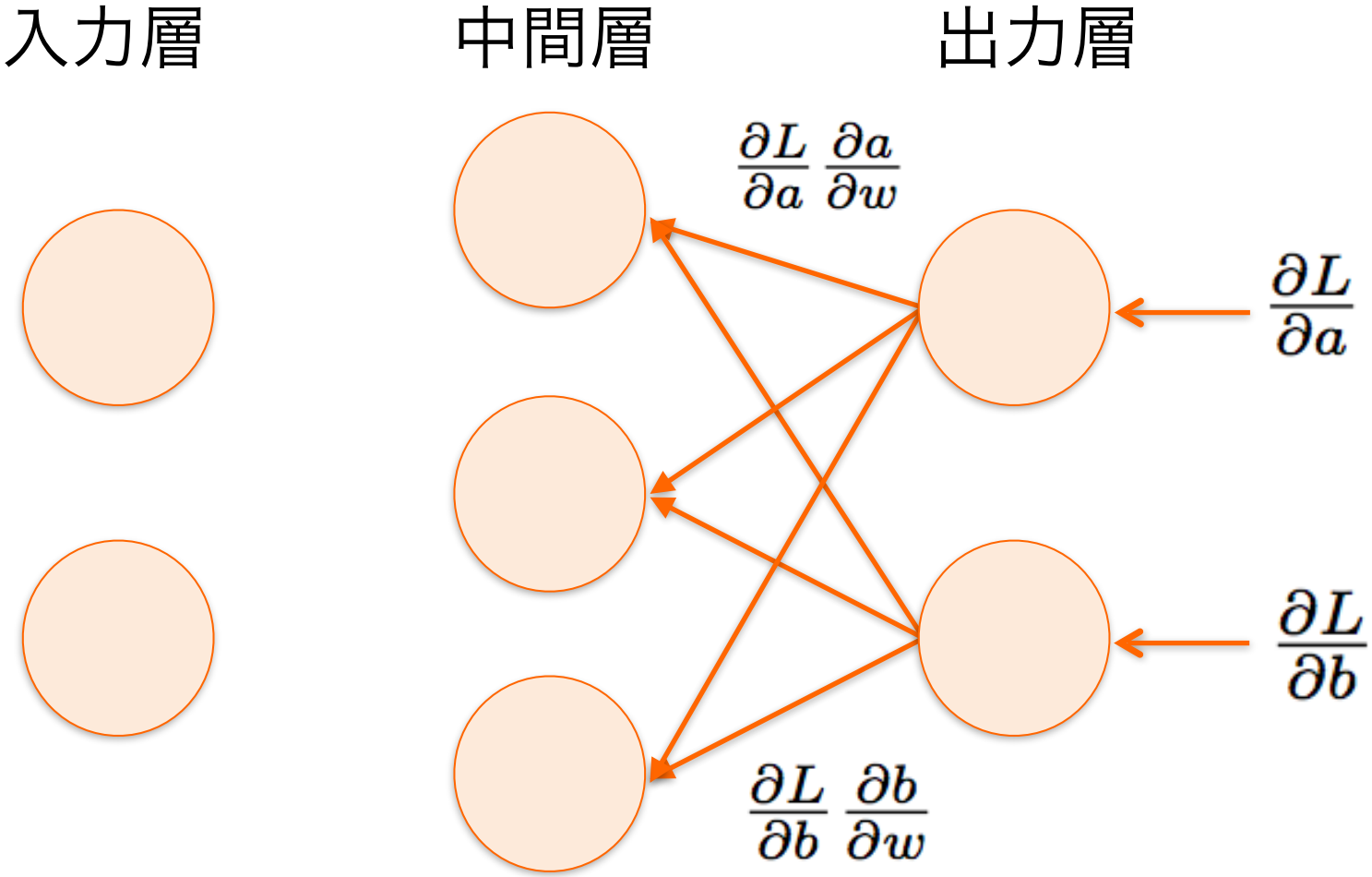
Training Phase



Training Phase



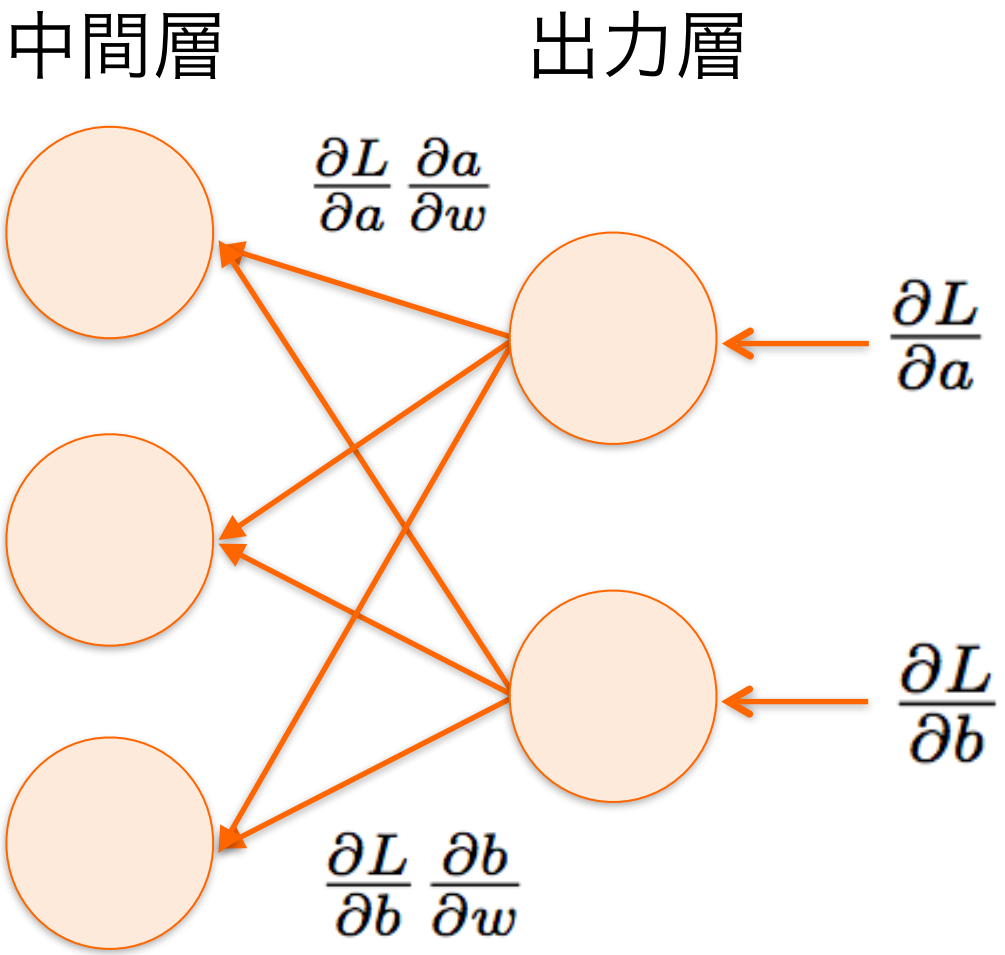
Training Phase



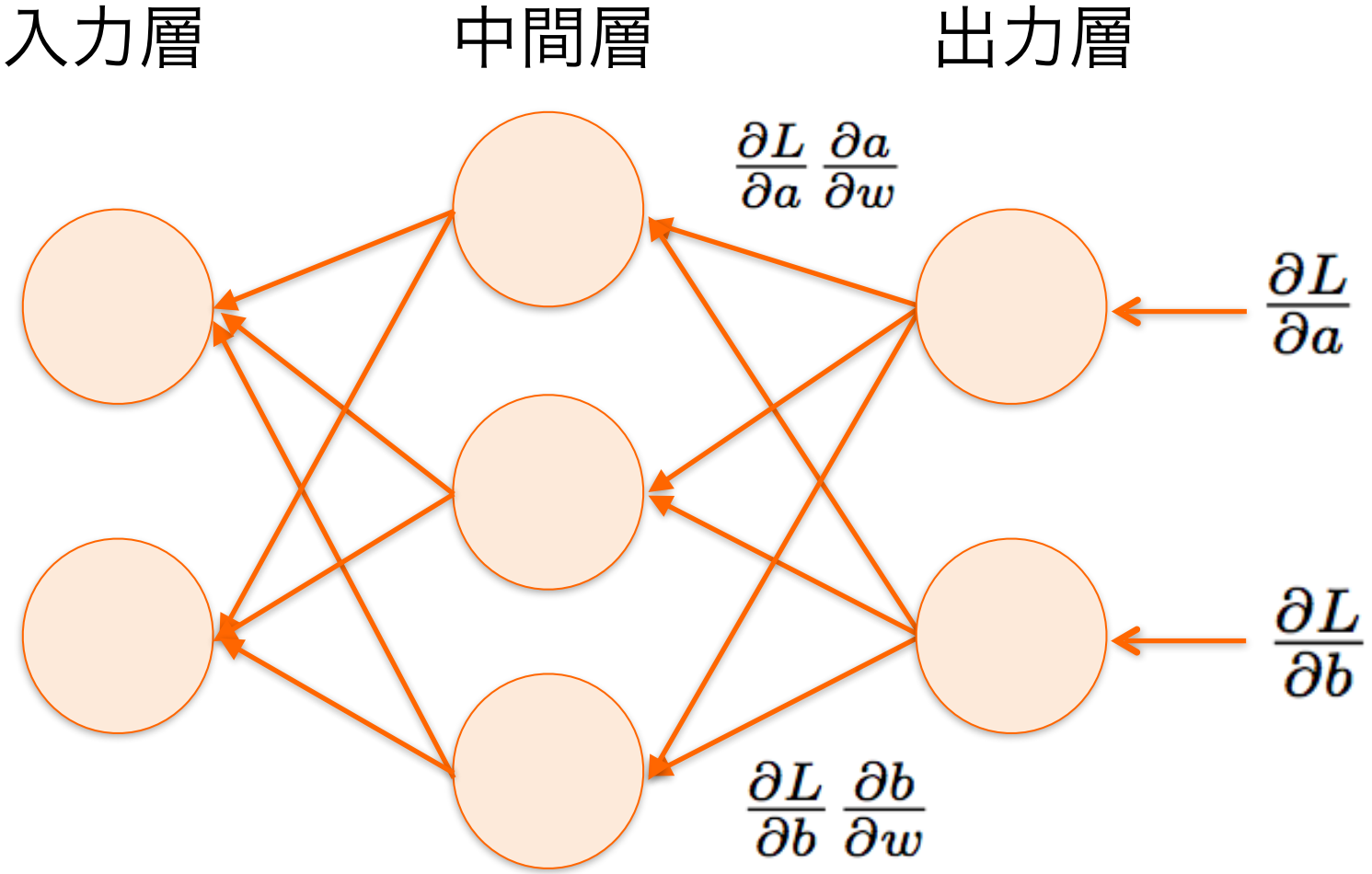
Training Phase

$$w = w - \eta \frac{\partial L}{\partial w}$$

η：学習係数



Training Phase

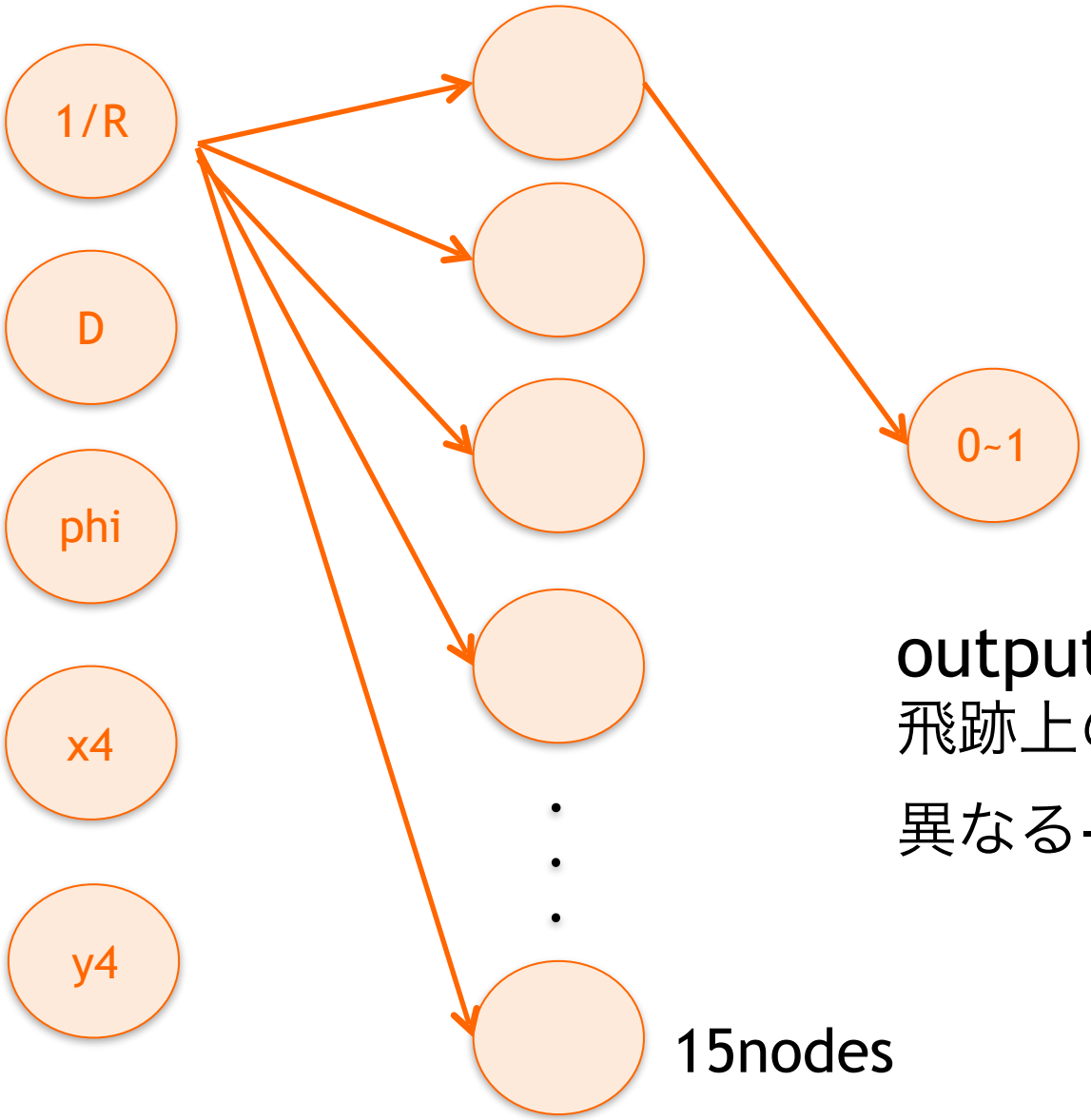


Hit on Track Net with D

input

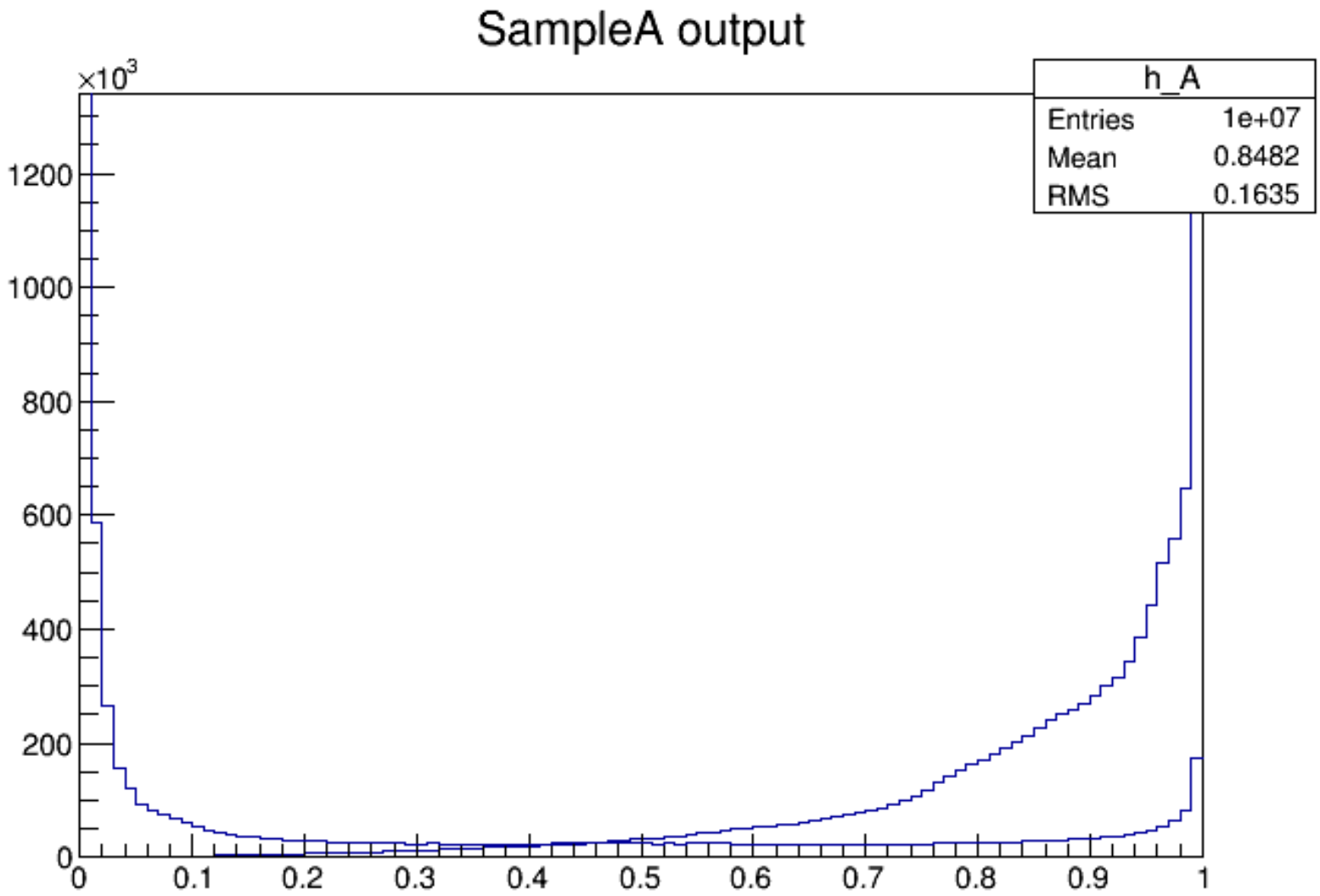
detector1~3の3点を通る飛跡のparameter

detector4のx,y座標

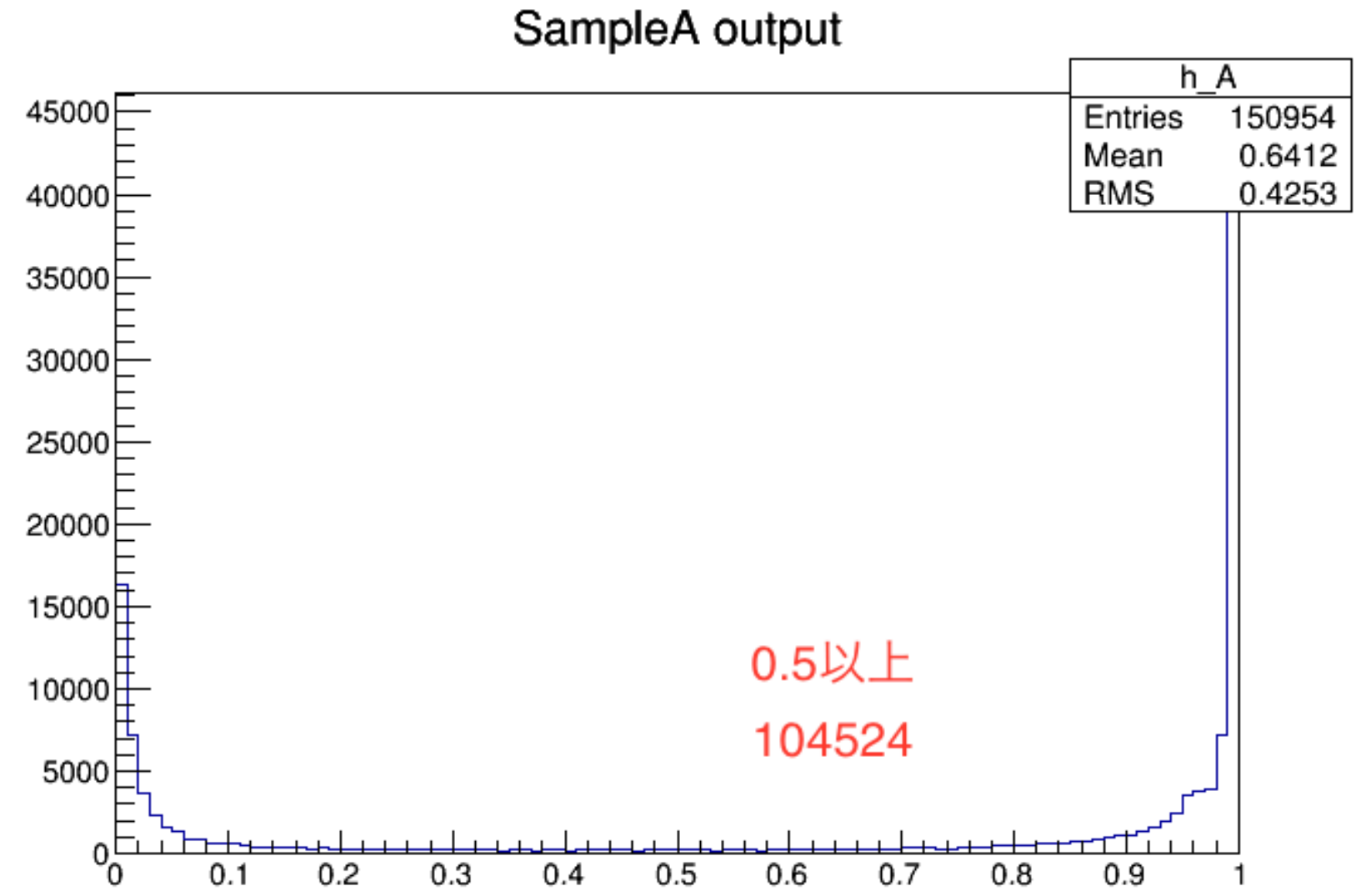


output
飛跡上の点->1
異なる->0

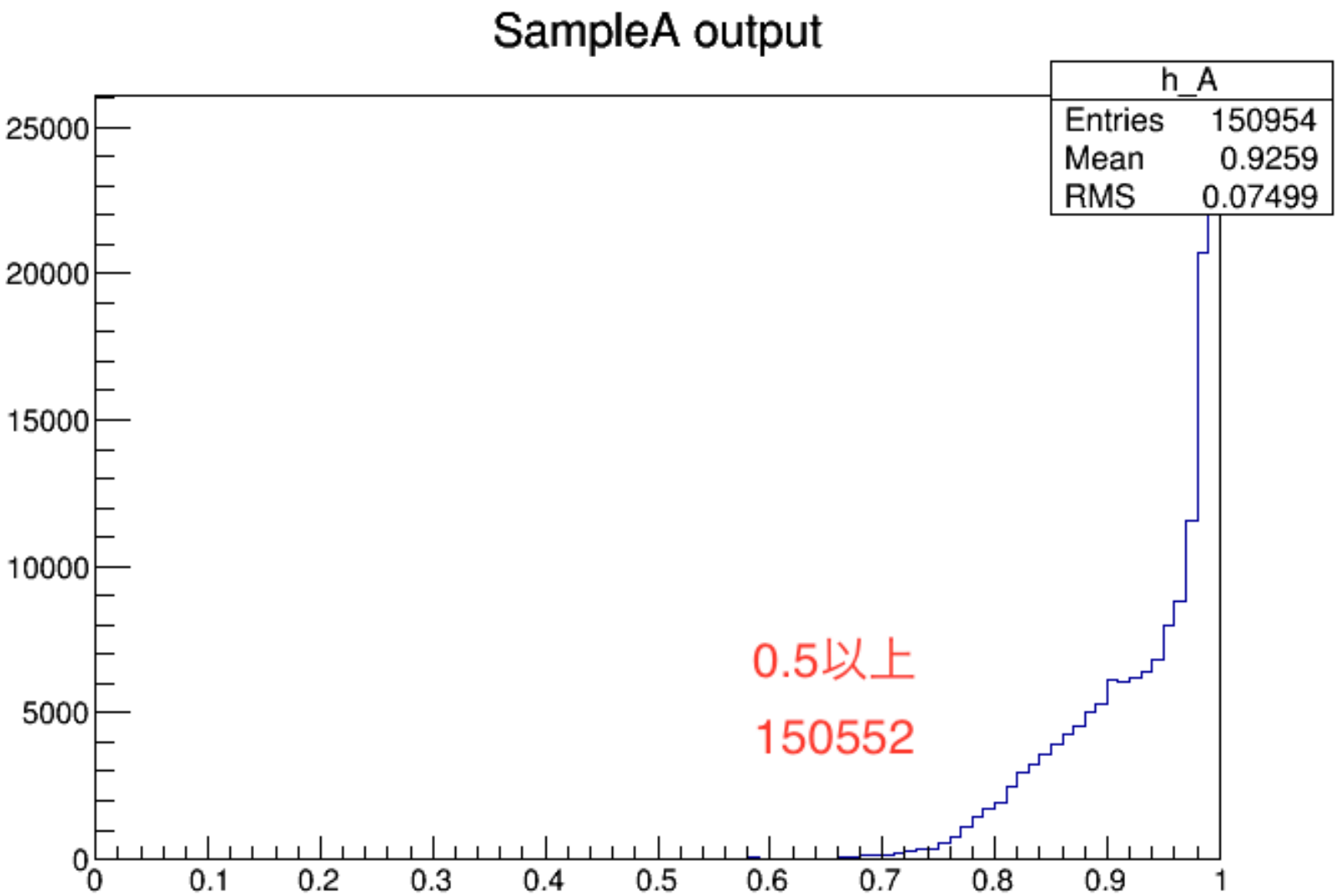
Training Result
input parameters = true parameters



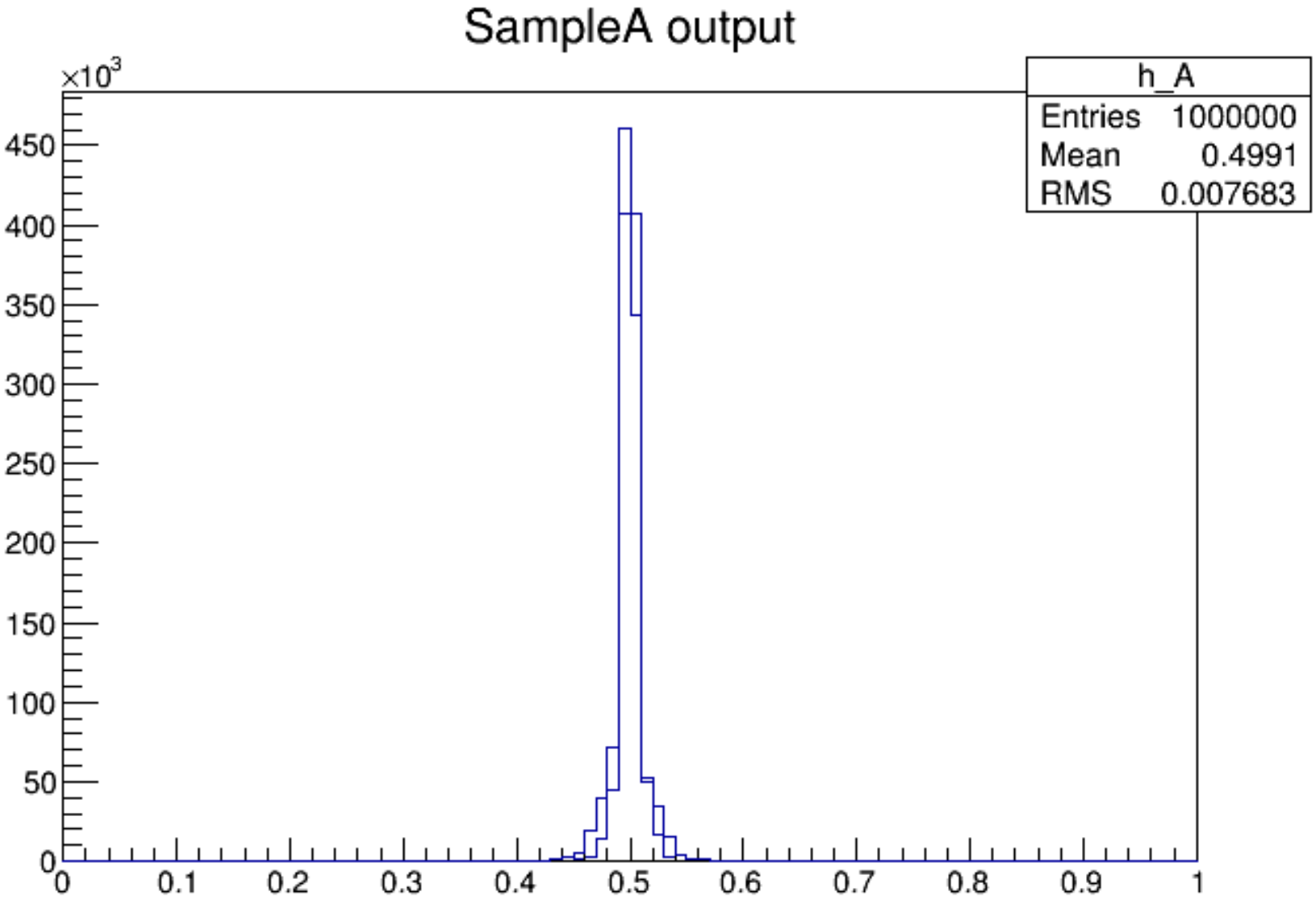
NeuralNet Output
input parameters = calculated



NeuralNet Output
input parameters = true parameters

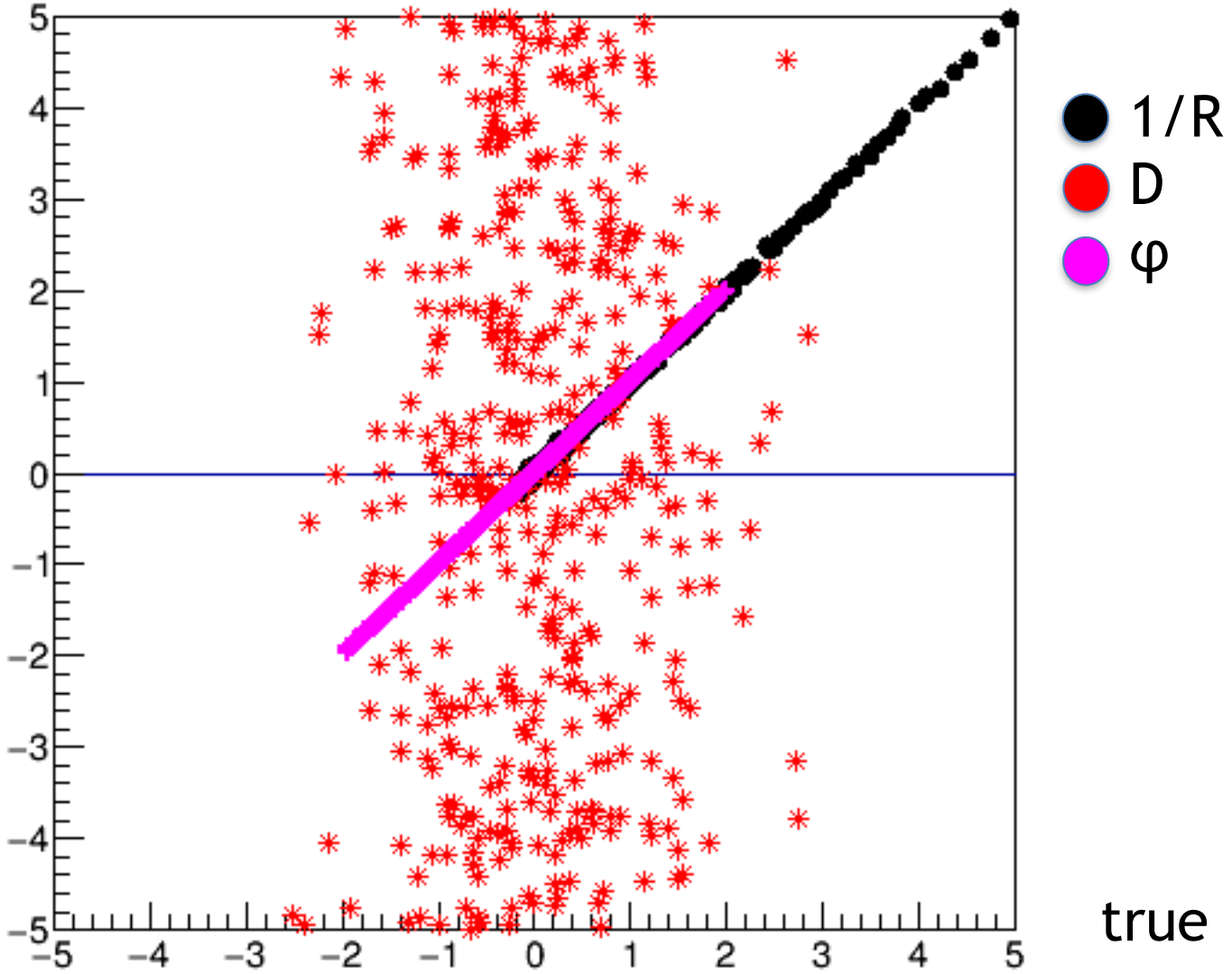


Training Result
input parameters = calculated



true-cal parameters

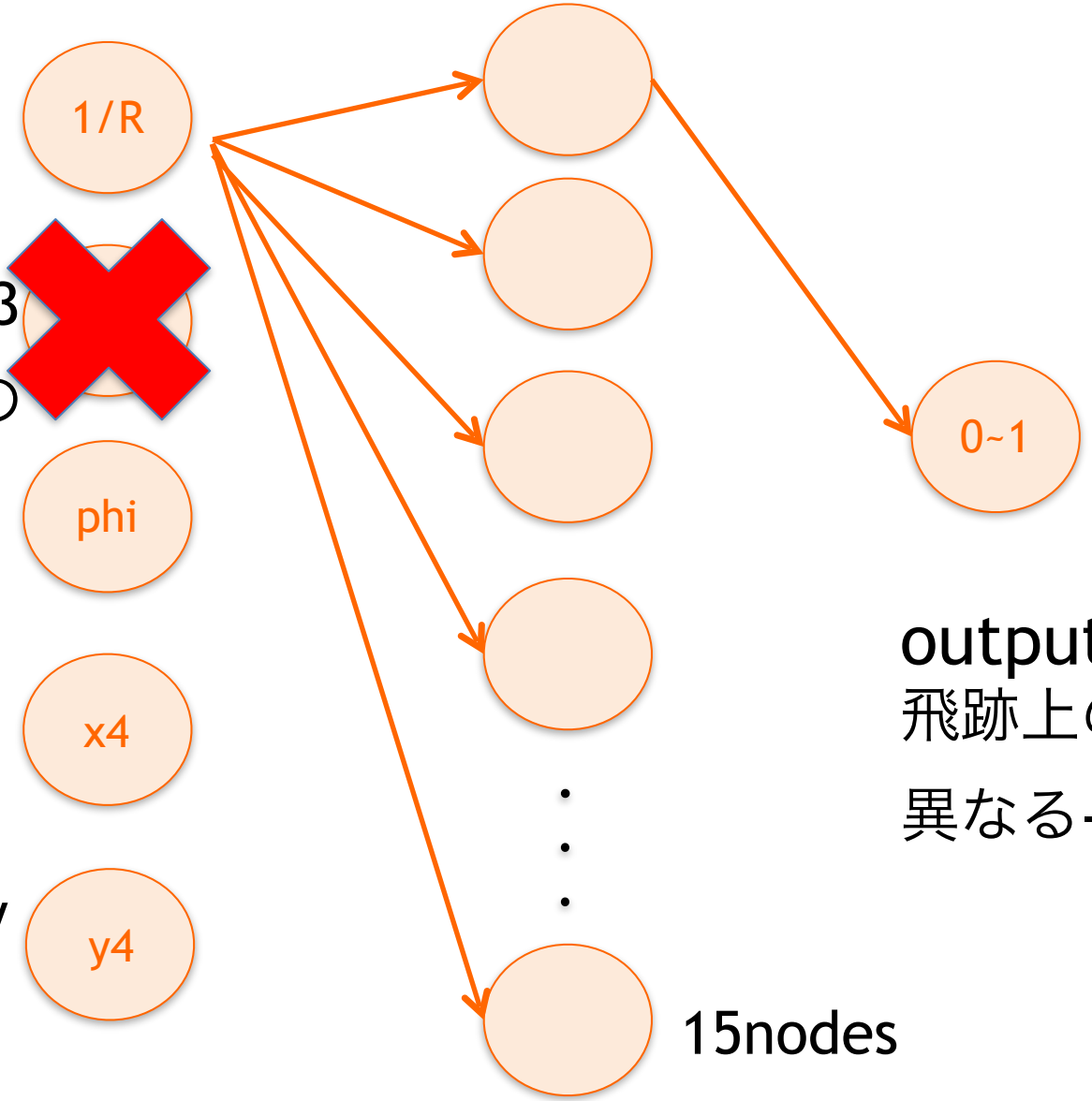
calculated



Hit on Track Net without D

input

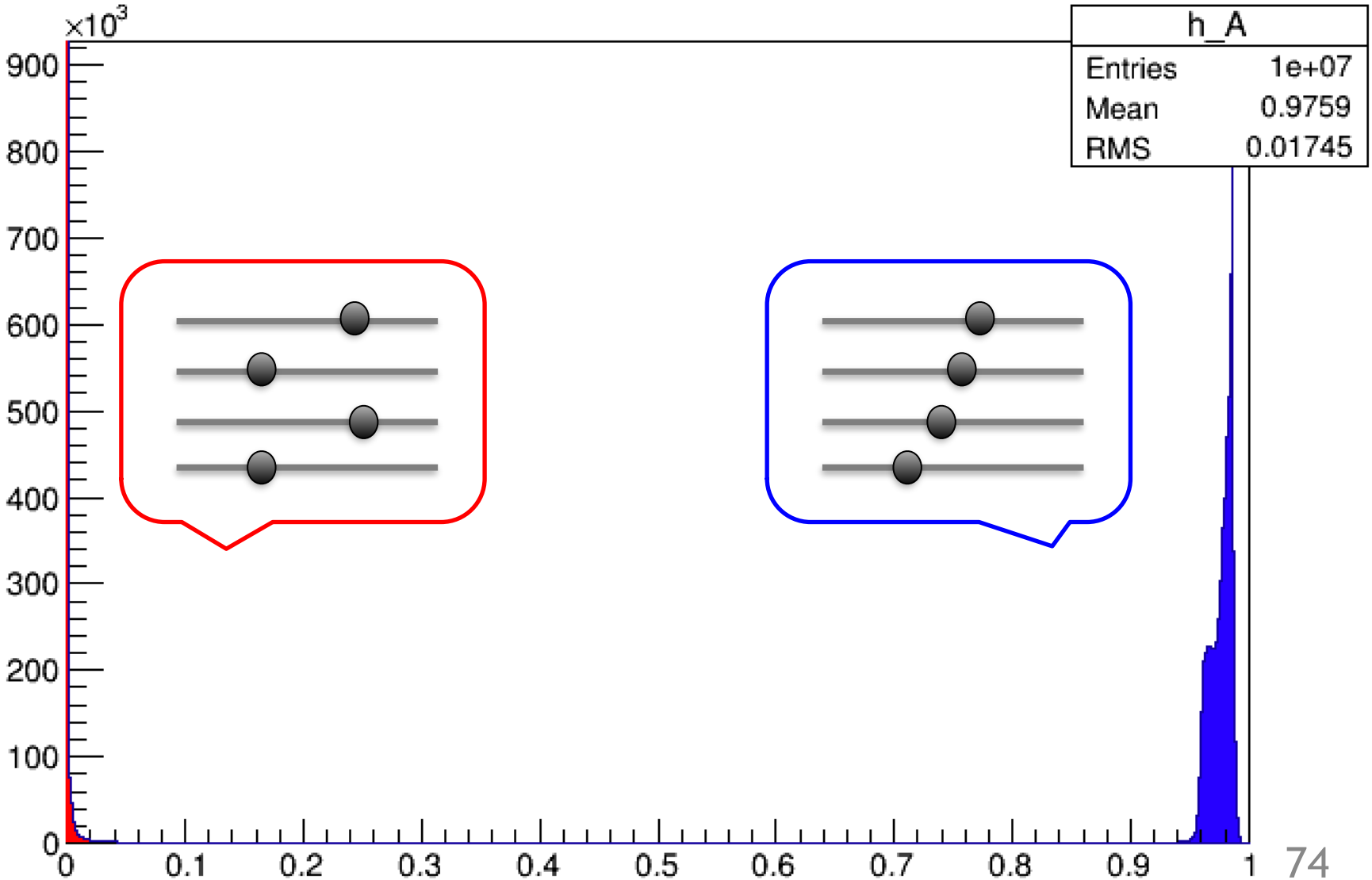
detector1~3の3点を通る飛跡のparameter



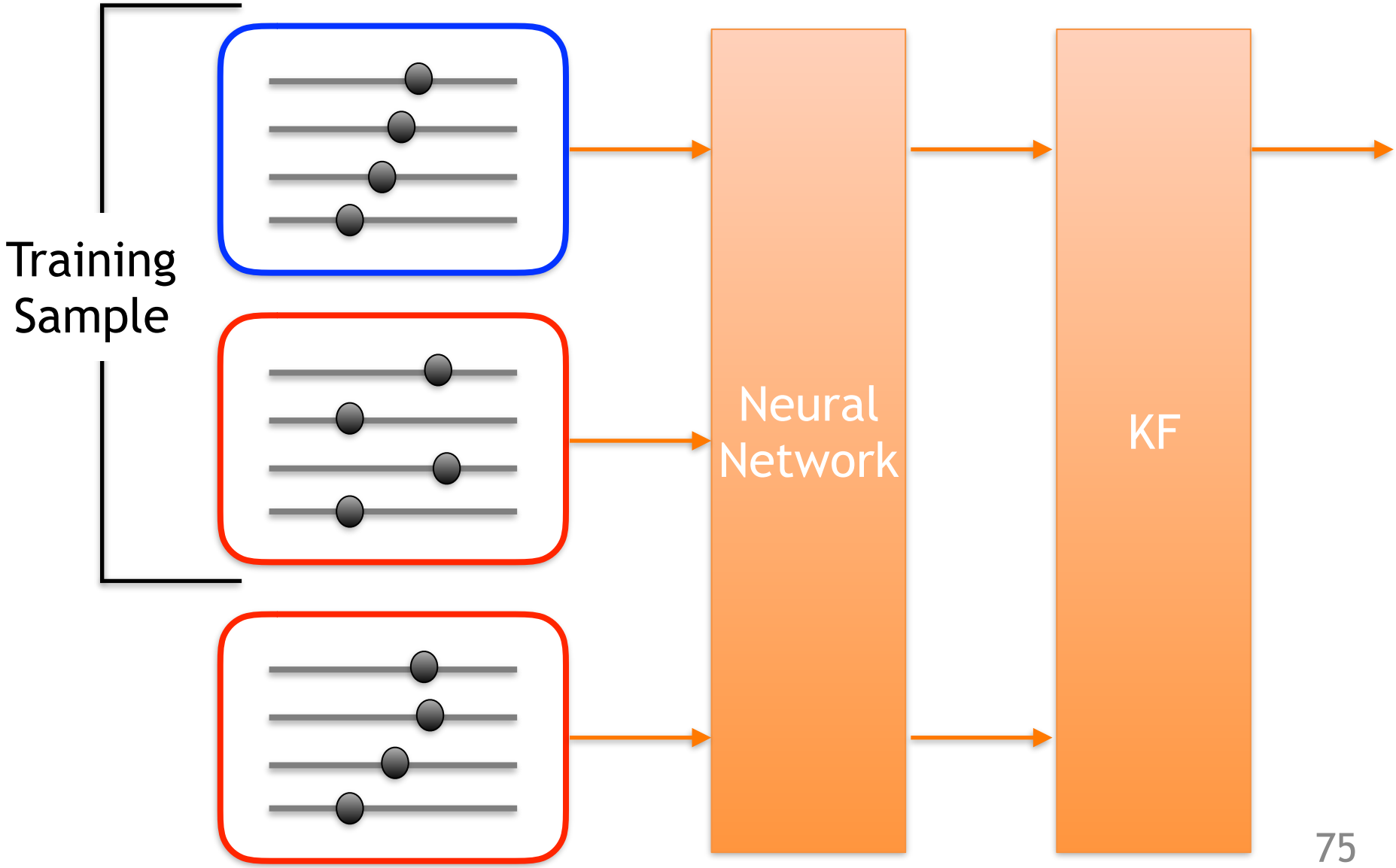
detector4のx,y座標

output
飛跡上の点->1
異なる->0

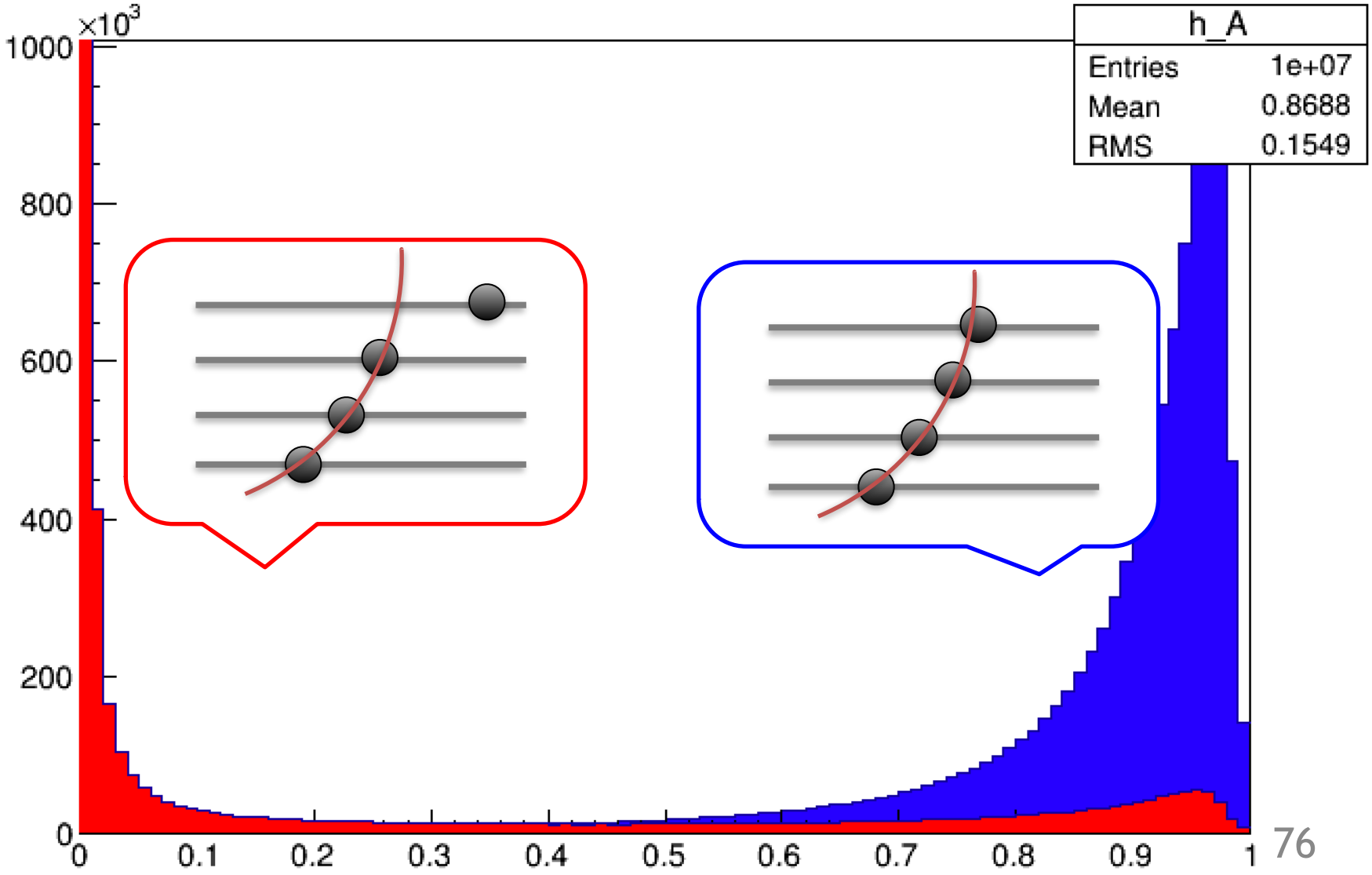
Training Result



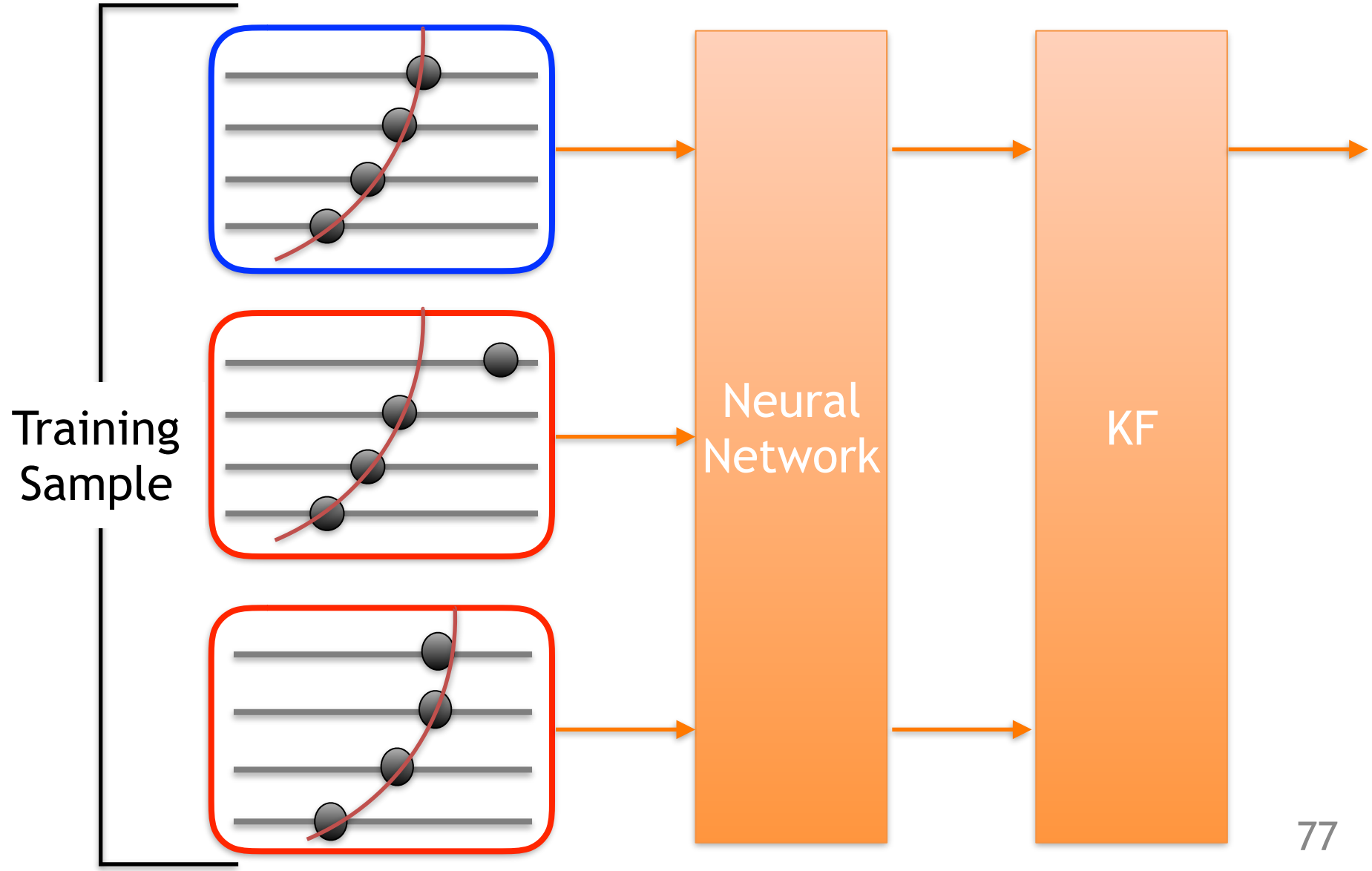
Use of 4 Hits Net



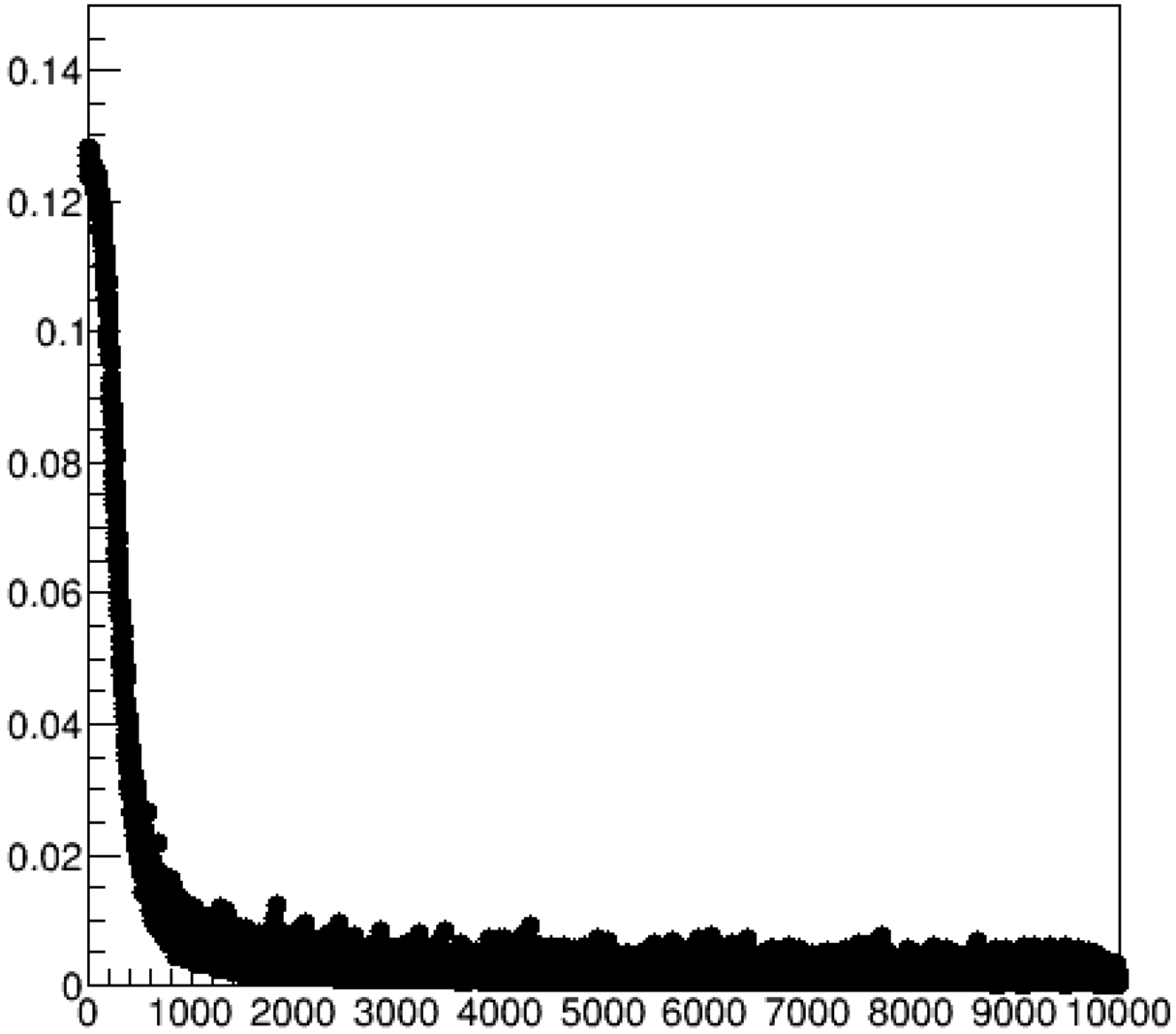
Training Result



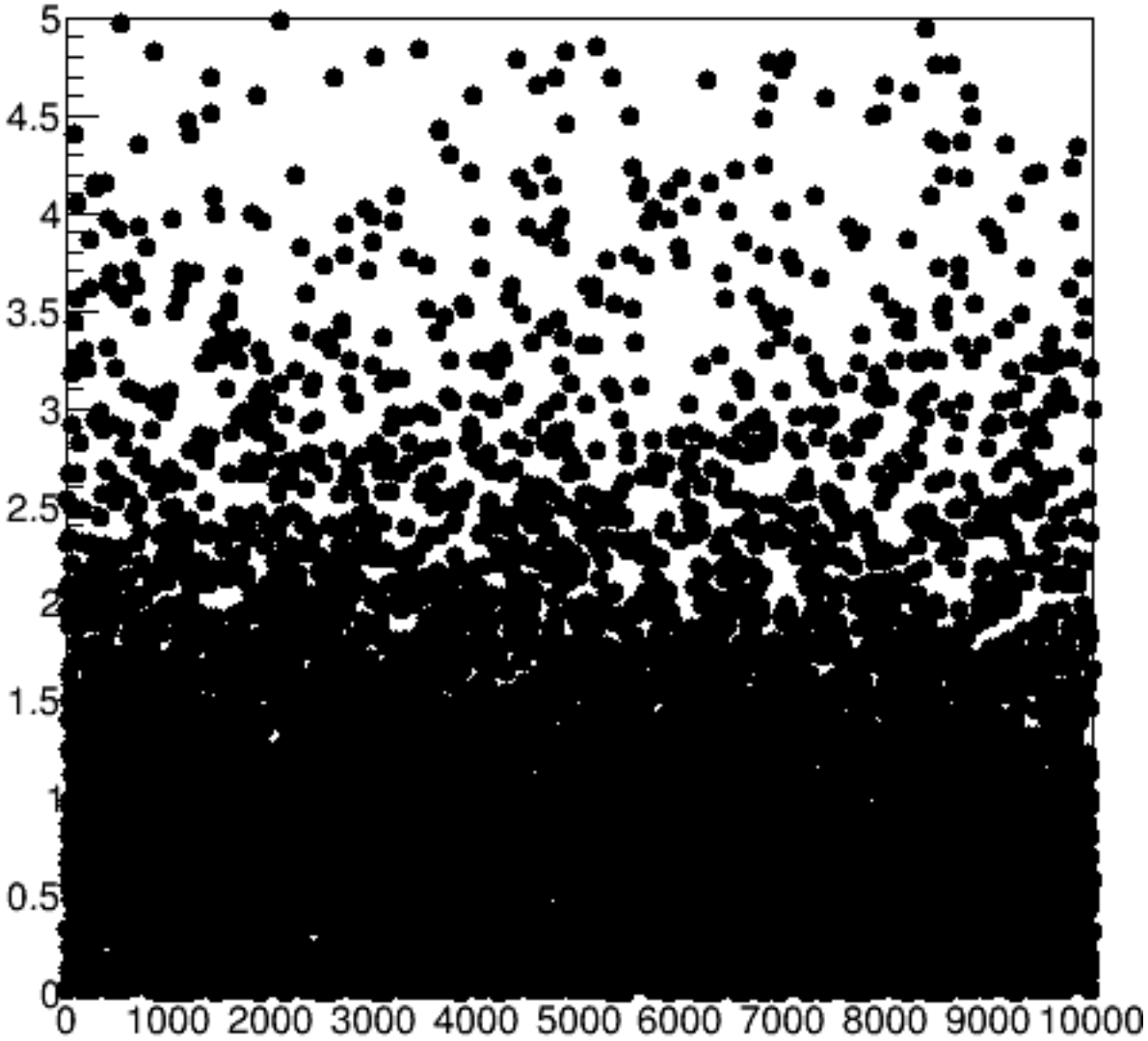
Use of Hit on Track Net



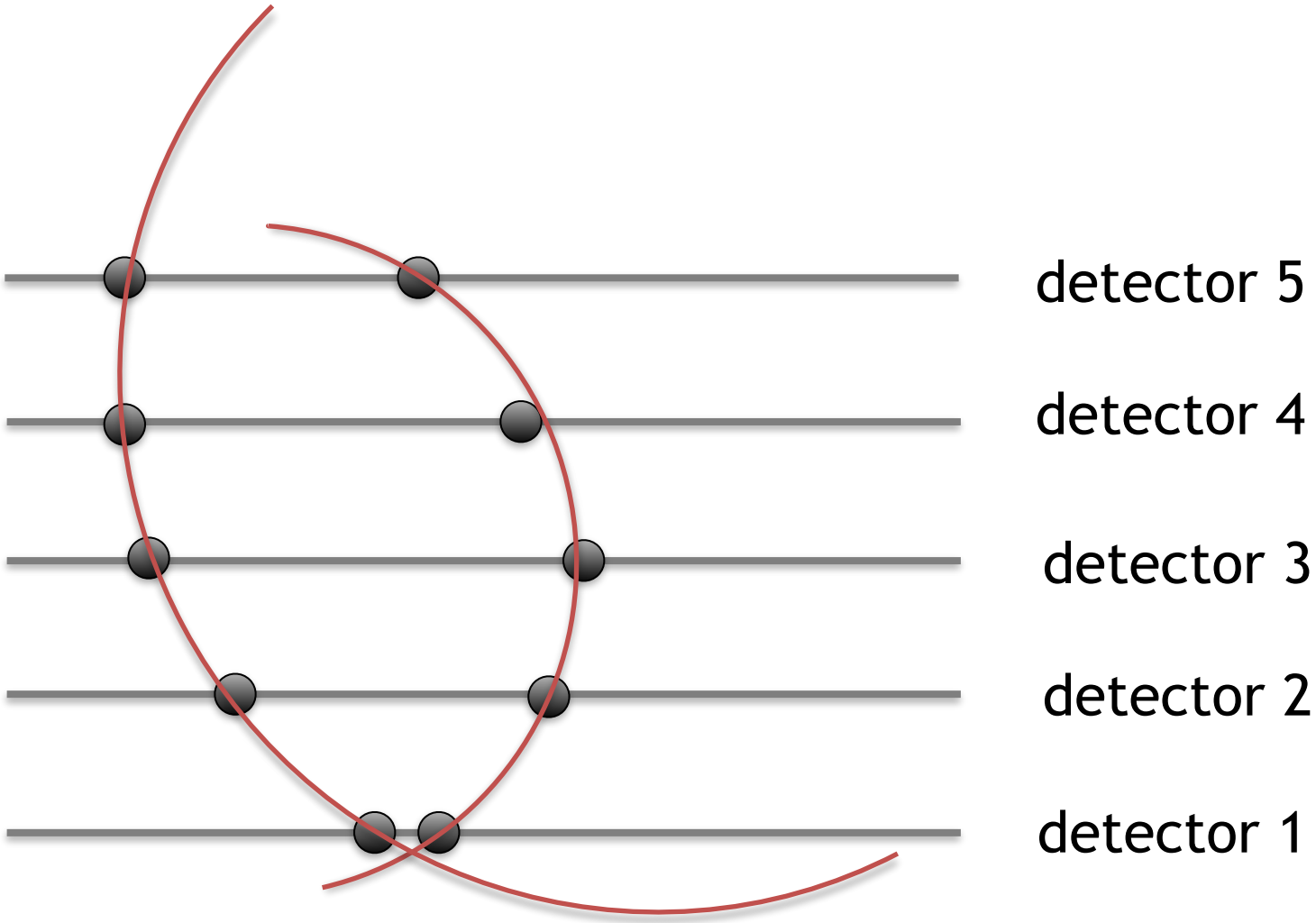
Loss Function



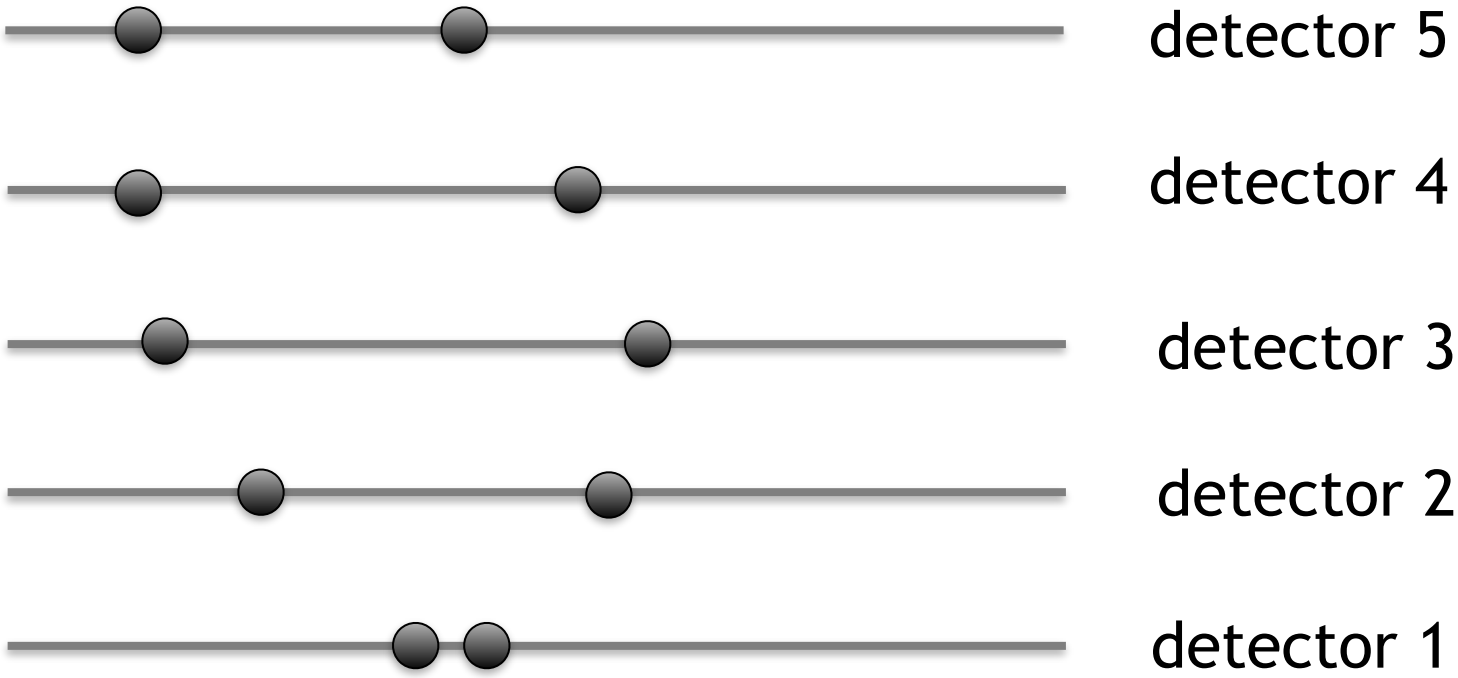
Loss Function



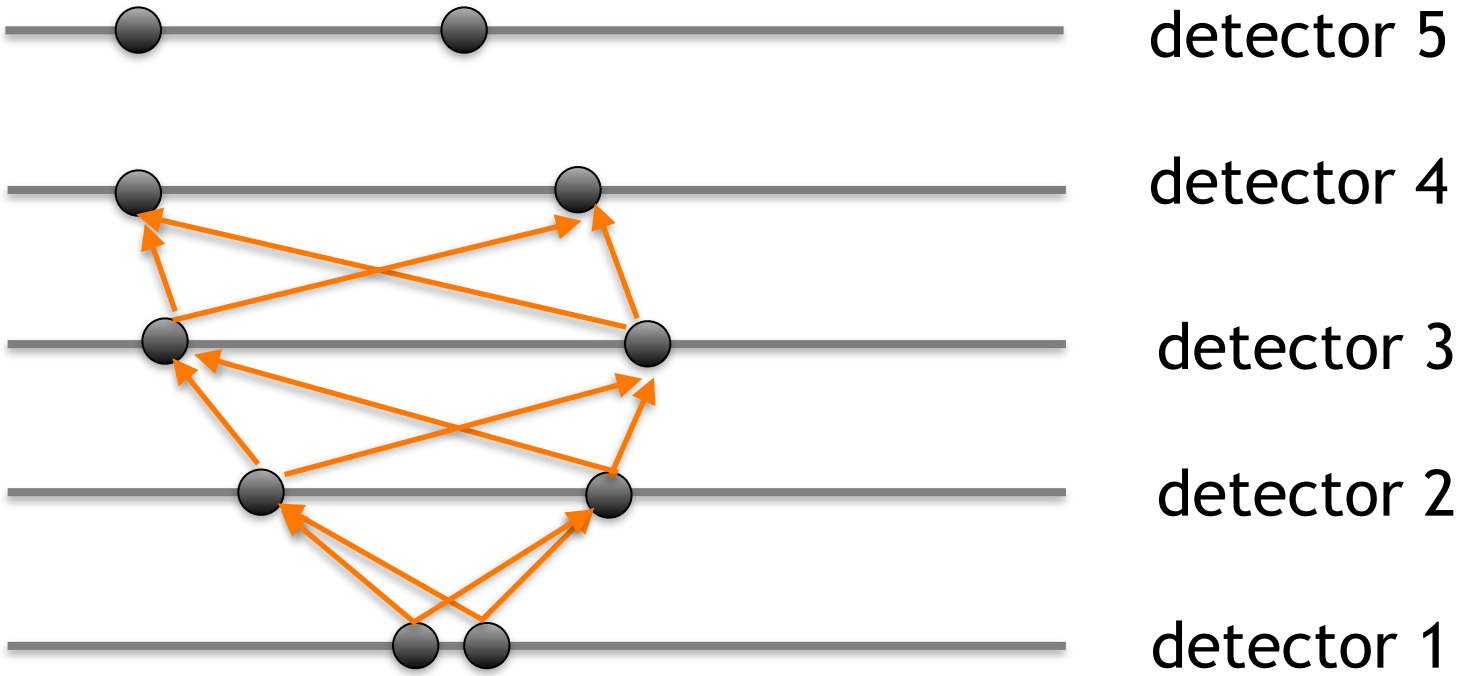
Calculation Methods



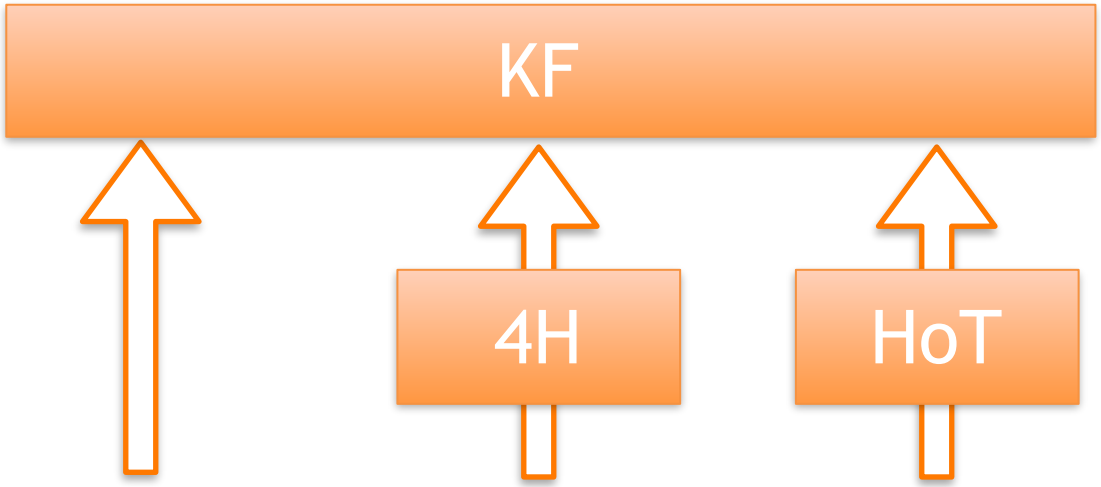
Calculation Methods

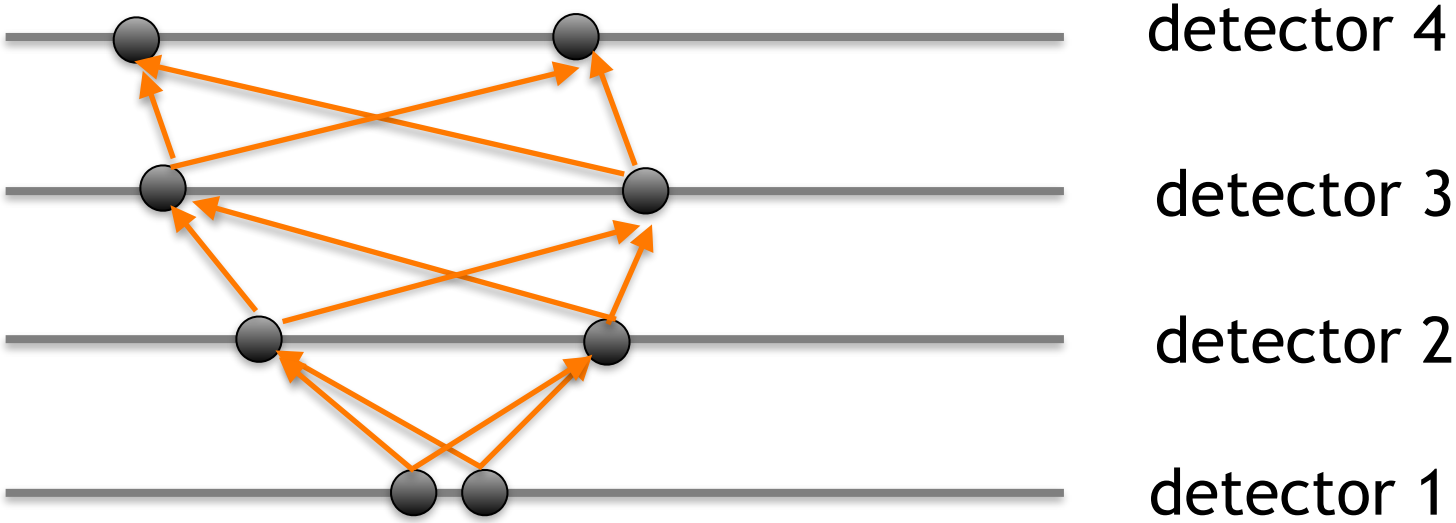


Calculation Methods

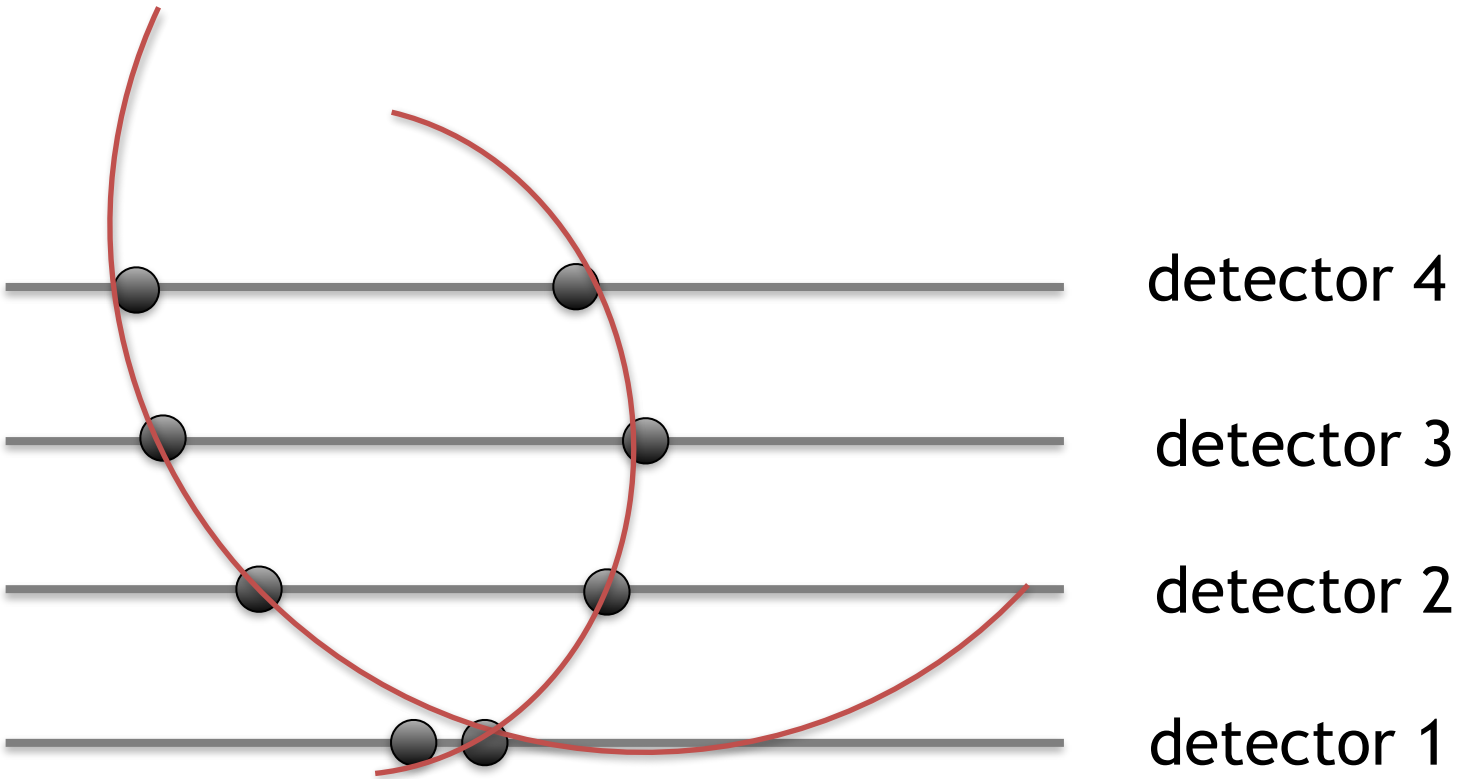


Calculation Methods

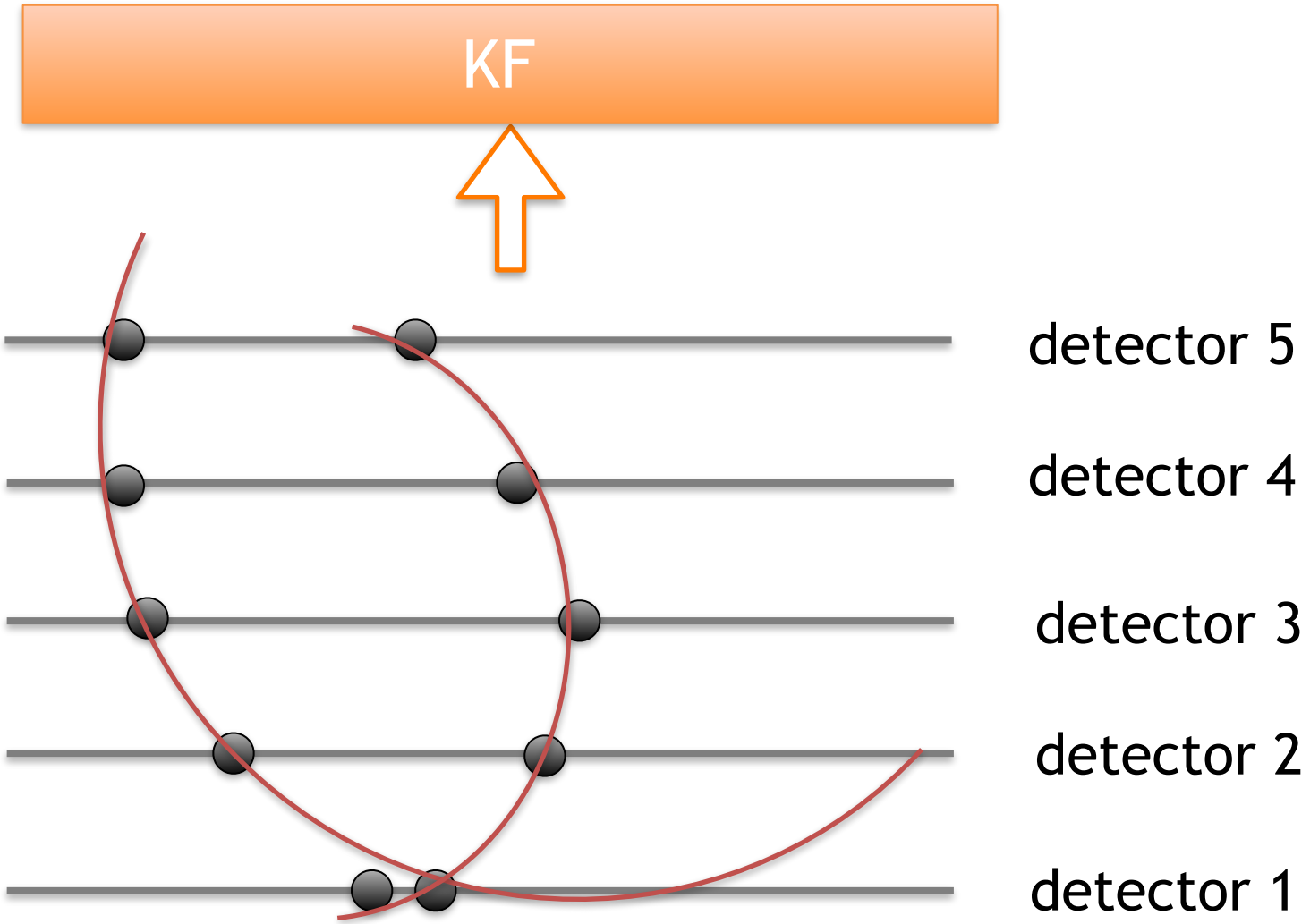




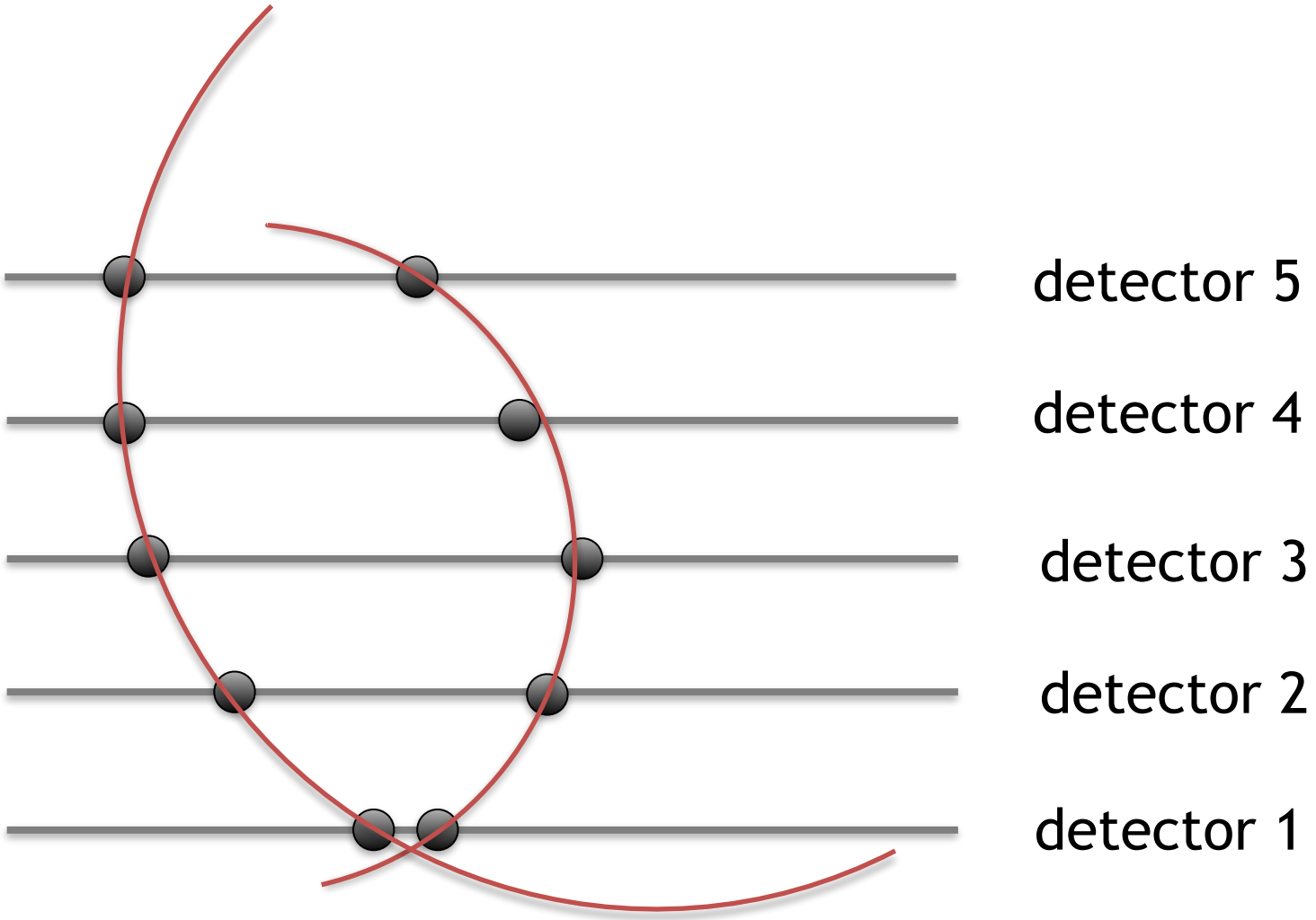
Calculation Methods



Calculation Methods

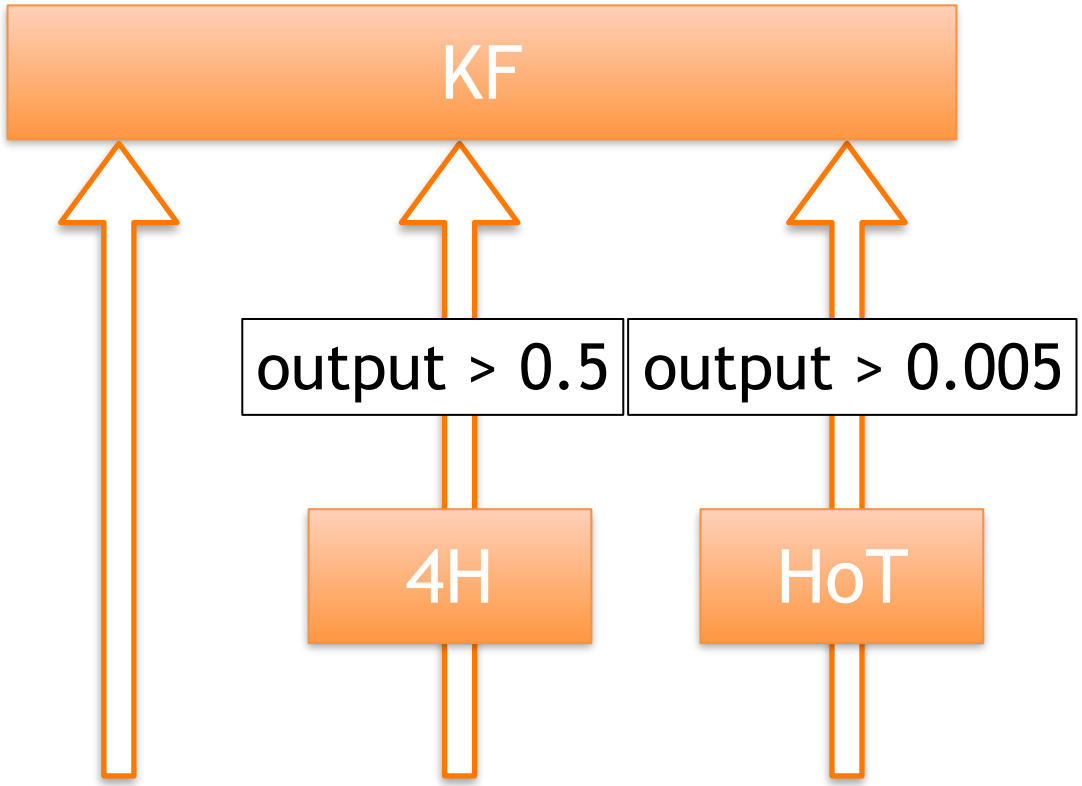


Calculation Methods



Calculation Methods

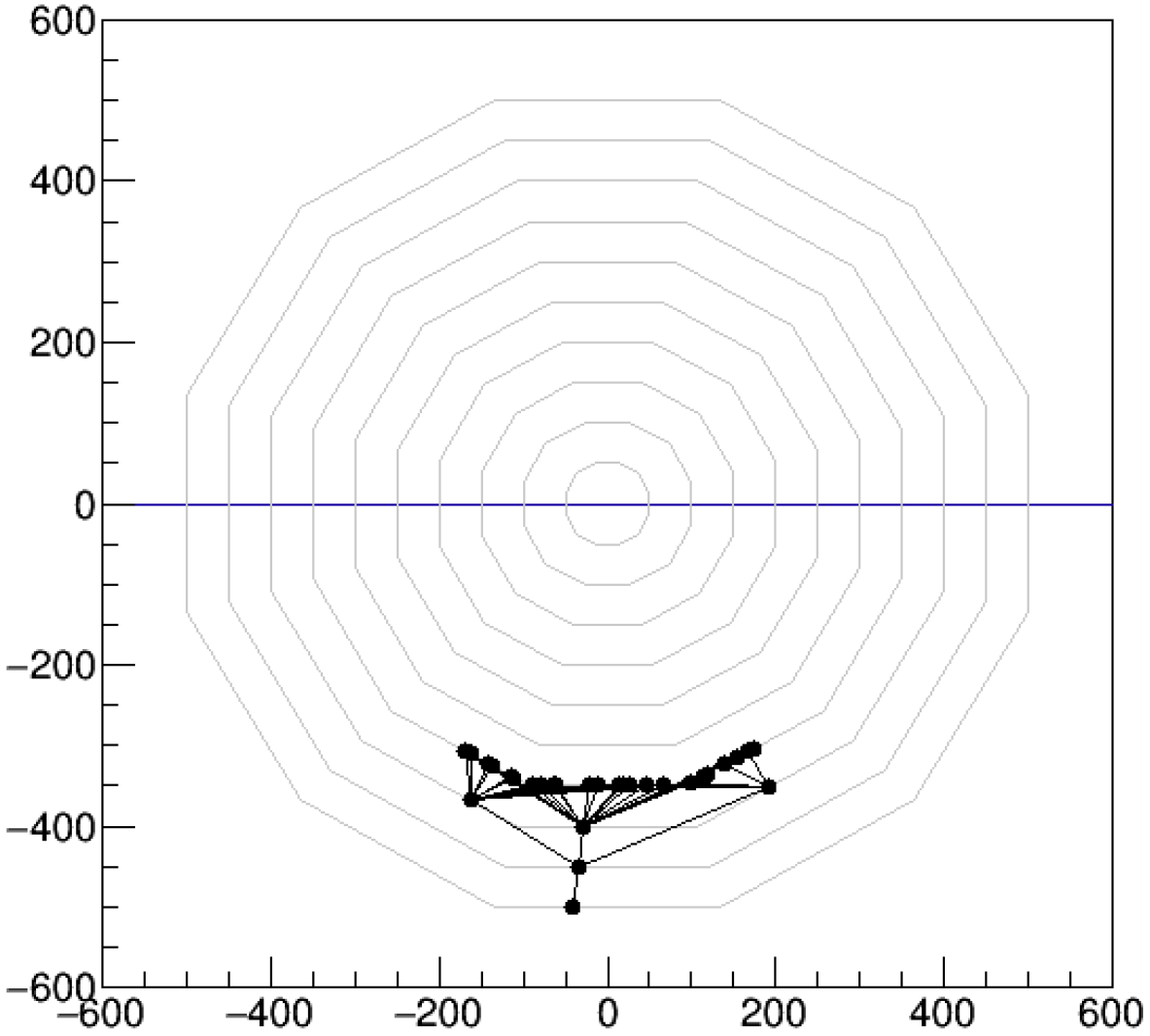
飛跡を計算



detector1~4のhitsの組み合わせが同一飛跡に乗るか判定

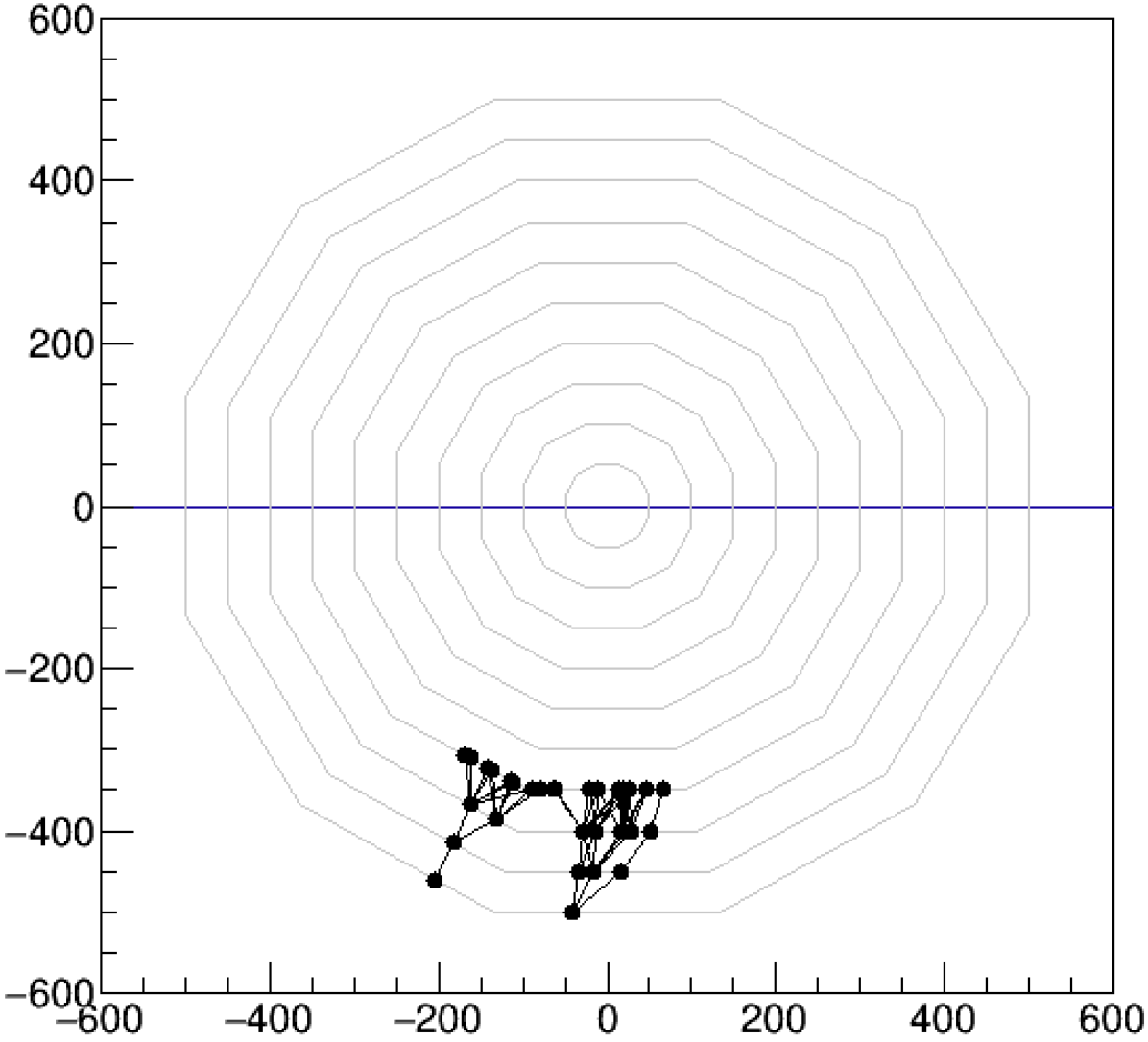
KF Inputs for 50 tracks

#50



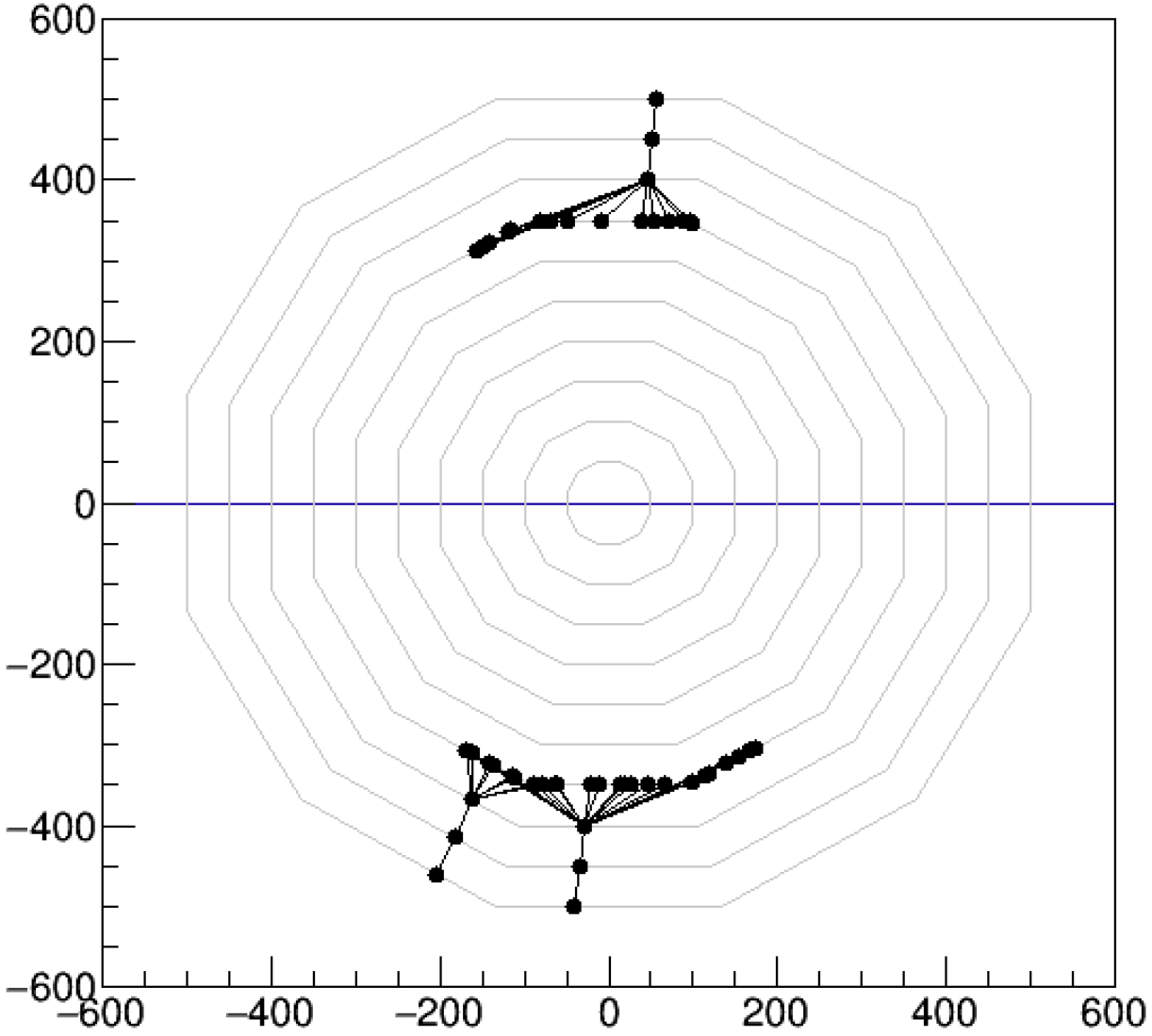
KF Inputs for 50 tracks

#50
with 4H



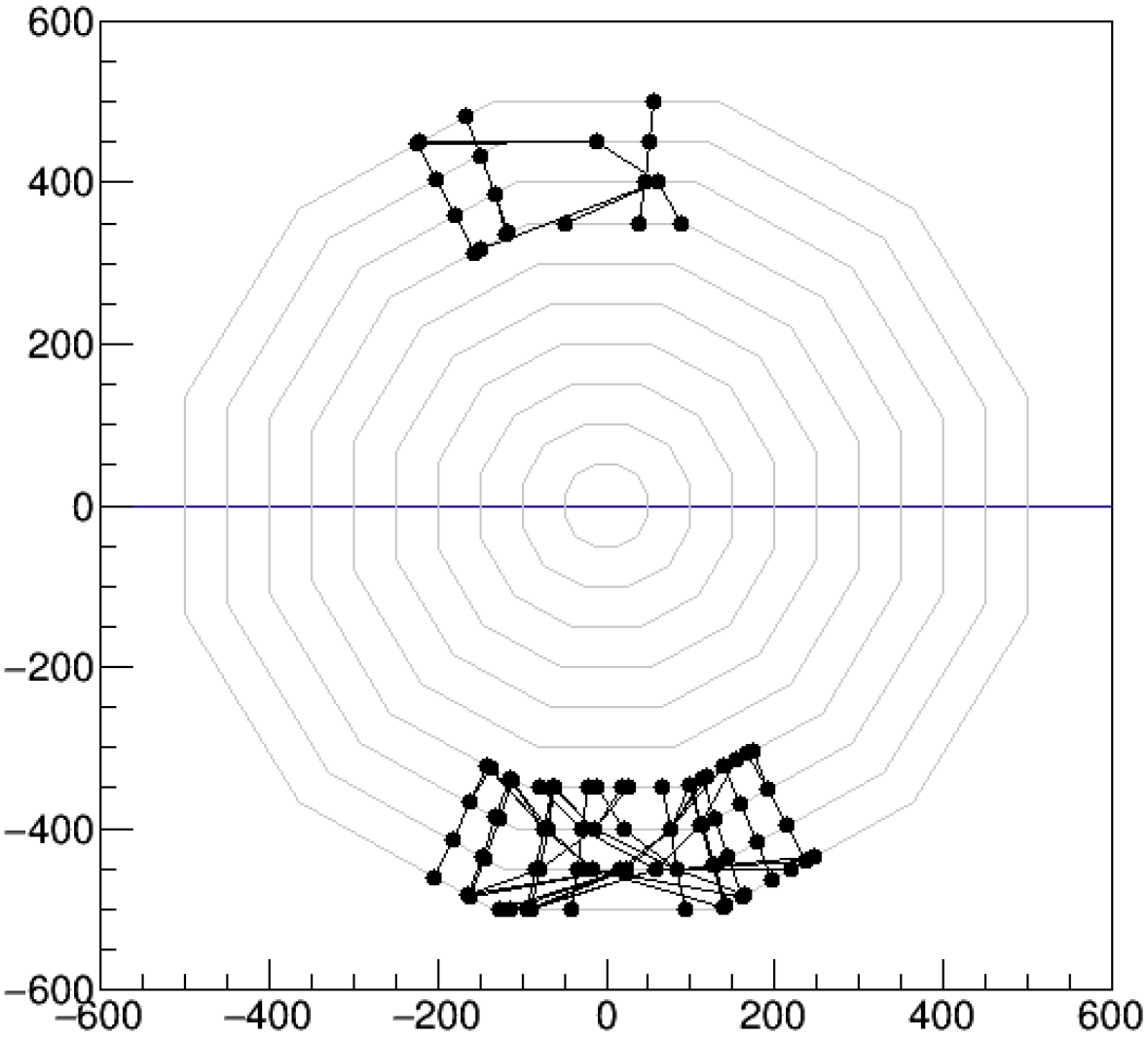
KF Inputs for 50 tracks

#50
with HoT



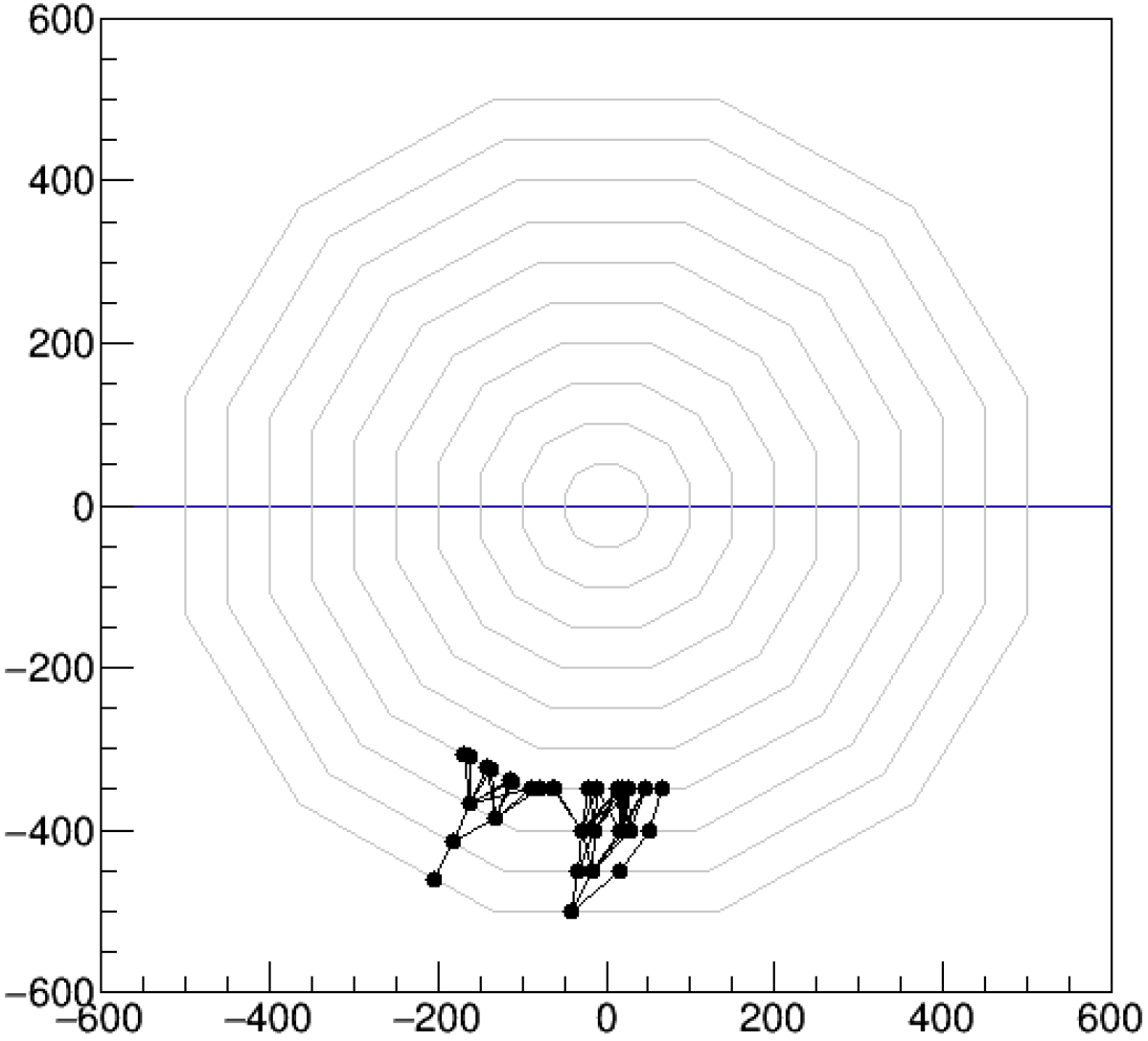
KF Outputs for 50 tracks

#50



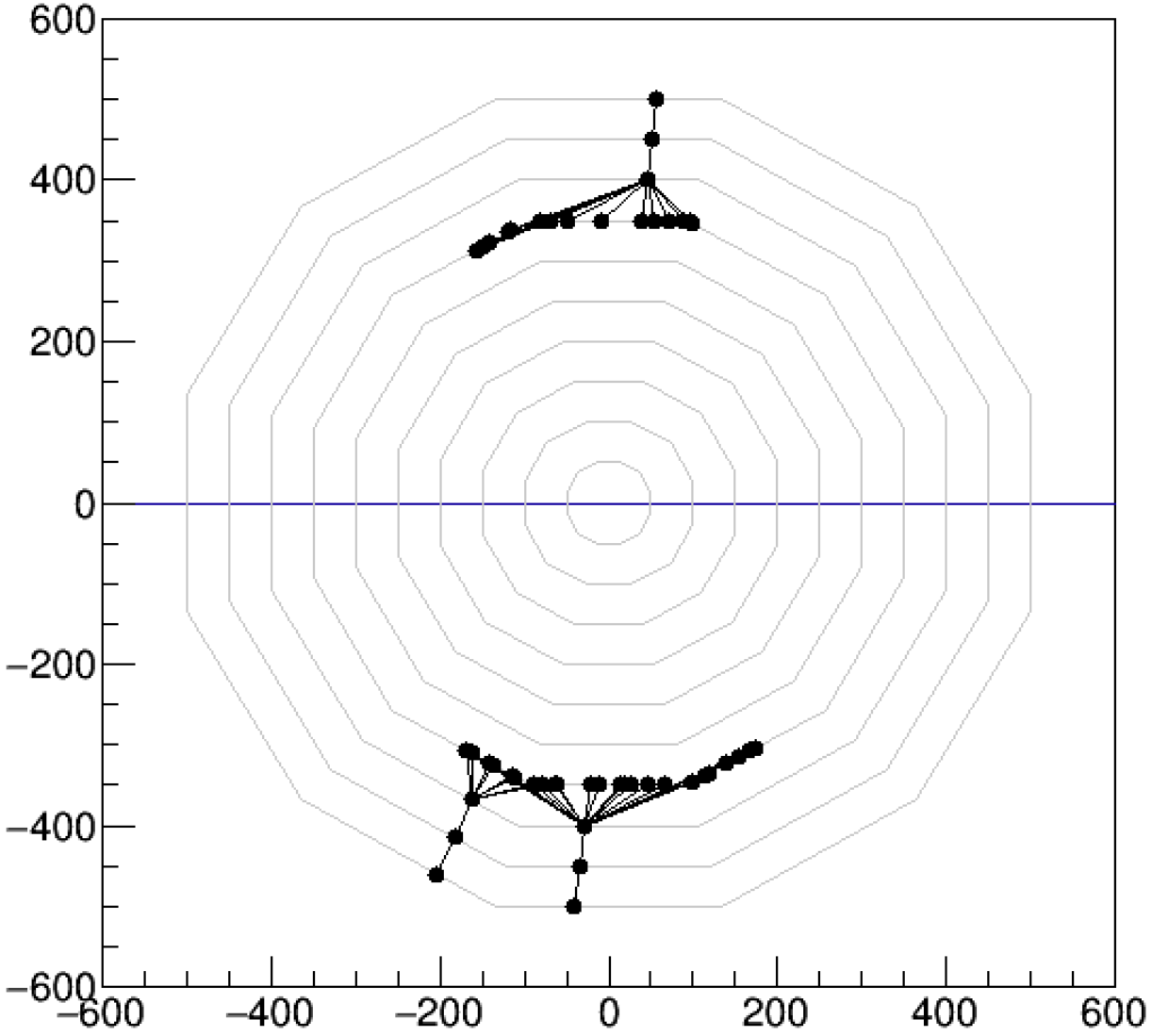
4H Outputs for 50 tracks

#50



HoT Outputs for 50 tracks

#50



流れ

- ３点でparameter初期化
- |D|>10をカット
- 点を加えてparameter更新
- v>10σをカット
- parameter収束判定
- |D|>10をカット
- S/n<10 -> OK
- 加えたdataのv>10σ -> 加えずに次のdetectorへ

v>10σをカット