Adaptive Methods for Track and Vertex Reconstruction

R. Frühwirth\textsuperscript{a}, A. Strandlie\textsuperscript{b}

\textsuperscript{a}Institute of High Energy Physics
Austrian Academy of Sciences, Vienna

\textsuperscript{b}Gjøvik University College, Gjøvik, Norway

International Workshop for Future Challenges in Tracking and Trigger Concepts
GSI, June 11, 2010
Outline

1 Introduction

2 Track finding
   - Local methods
   - Global methods

3 Track fitting
   - Traditional approach
   - Adaptive approach

4 Vertex reconstruction

5 Conclusions and Outlook
Outline

1. Introduction
2. Track finding
   - Local methods
   - Global methods
3. Track fitting
   - Traditional approach
   - Adaptive approach
4. Vertex reconstruction
5. Conclusions and Outlook
**What is it all about?**

- Track and vertex reconstruction are essential steps in the data analysis chain
- Crucial factor in quality of physics analysis
- Growing importance for high-level trigger
- **Track reconstruction:** determine location, direction and (inverse) momentum of charged tracks
- **Vertex reconstruction:** determine location of interaction point and momenta of participating tracks
- Many basic features in common
Introduction

What is it all about?

- Track and vertex reconstruction are essential steps in the data analysis chain
- Crucial factor in quality of physics analysis
- Growing importance for high-level trigger
- **Track reconstruction:** determine location, direction and (inverse) momentum of charged tracks
- **Vertex reconstruction:** determine location of interaction point and momenta of participating tracks
- Many basic features in common
What is it all about?

- Track and vertex reconstruction are essential steps in the data analysis chain
- Crucial factor in quality of physics analysis
- Growing importance for high-level trigger

**Track reconstruction**: determine location, direction and (inverse) momentum of charged tracks

**Vertex reconstruction**: determine location of interaction point and momenta of participating tracks

Many basic features in common
What is it all about?

- Track and vertex reconstruction are essential steps in the data analysis chain
- Crucial factor in quality of physics analysis
- Growing importance for high-level trigger

**Track reconstruction:** determine location, direction and (inverse) momentum of charged tracks

**Vertex reconstruction:** determine location of interaction point and momenta of participating tracks

Many basic features in common
Introduction

What is it all about?

- Track and vertex reconstruction are essential steps in the data analysis chain
- Crucial factor in quality of physics analysis
- Growing importance for high-level trigger

**Track reconstruction**: determine location, direction and (inverse) momentum of charged tracks

**Vertex reconstruction**: determine location of interaction point and momenta of participating tracks

Many basic features in common
Introduction

What is it all about?

- Track and vertex reconstruction are essential steps in the data analysis chain
- Crucial factor in quality of physics analysis
- Growing importance for high-level trigger

**Track reconstruction:** determine location, direction and (inverse) momentum of charged tracks

**Vertex reconstruction:** determine location of interaction point and momenta of participating tracks

Many basic features in common
Ingredients

- **Track reconstruction**
  - Track model known (analytical or numerical)
  - Observation errors known
  - Process noise known (approximately)
Ingredients

- **Track reconstruction**
  - Track model known (analytical or numerical)
  - Observation errors known
  - Process noise known (approximately)
  - Assignment of observations to tracks unknown
  - Hits from background tracks, electronic noise
  - Mass not always known
**Introduction**

**Ingredients**

- **Track reconstruction**
  - Track model known (analytical or numerical)
  - Observation errors known
  - Process noise known (approximately)
  - Assignment of observations to tracks unknown
  - Hits from background tracks, electronic noise
  - Mass not always known

- **Vertex reconstruction**
  - Track model known (analytical or numerical)
  - Track errors known
  - No process noise
**Ingredients**

- **Track reconstruction**
  - Track model known (analytical or numerical)
  - Observation errors known
  - Process noise known (approximately)
  - Assignment of observations to tracks unknown
  - Hits from background tracks, electronic noise
  - Mass not always known

- **Vertex reconstruction**
  - Track model known (analytical or numerical)
  - Track errors known
  - No process noise
  - Assignment of tracks to vertices unknown
  - Background tracks
Introduction

Three aspects

■ Pattern recognition
  □ Find out which detector hits belong to the same track — highly detector dependent
  ▼ Find out which tracks are produced at the same interaction point — nearly detector independent

■ Estimation
  □ Estimate track parameters at one or several points along the track
  ▼ Estimate vertex location and momenta of attached tracks
  □ ▼ Can be formulated as extended Kalman filter in both cases

■ Test
  □ Test track hypothesis and reject outlying detector hits
  ▼ Test vertex hypothesis and reject outlying tracks
Three aspects

- **Pattern recognition**
  - ✅ Find out which detector hits belong to the same track — highly detector dependent
  - ✅ Find out which tracks are produced at the same interaction point — nearly detector independent

- **Estimation**
  - ✅ Estimate track parameters at one or several points along the track
  - ✅ Estimate vertex location and momenta of attached tracks
  - 🗣️ TV Can be formulated as extended Kalman filter in both cases

- **Test**
  - ✅ Test track hypothesis and reject outlying detector hits
  - ✅ Test vertex hypothesis and reject outlying tracks
Three aspects

- **Pattern recognition**
  - Find out which detector hits belong to the same track — highly detector dependent
  - Find out which tracks are produced at the same interaction point — nearly detector independent

- **Estimation**
  - Estimate track parameters at one or several points along the track
  - Estimate vertex location and momenta of attached tracks
  - Can be formulated as extended Kalman filter in both cases

- **Test**
  - Test track hypothesis and reject outlying detector hits
  - Test vertex hypothesis and reject outlying tracks
Introduction

Classical vs. Adaptive

**Classical approach:**
- Do pattern recognition (track/vertex finding)
- Submit track/vertex candidate to least-squares fit
- Inspect test quantities, identify outlying hits/tracks
- Remove outliers, repeat fit
- ...

**Adaptive approach:**
- Do preliminary pattern recognition or none at all
- Submit hit/track collection to adaptive fit
- Inspect posterior weights of hits/tracks and remove outliers
Classical vs. Adaptive

- **Classical approach:**
  - Do pattern recognition (track/vertex finding)
  - Submit track/vertex candidate to least-squares fit
  - Inspect test quantities, identify outlying hits/tracks
  - Remove outliers, repeat fit
  - ...

- **Adaptive approach:**
  - Do preliminary pattern recognition or none at all
  - Submit hit/track collection to adaptive fit
  - Inspect posterior weights of hits/tracks and remove outliers
Outline

1. Introduction

2. Track finding
   - Local methods
   - Global methods

3. Track fitting
   - Traditional approach
   - Adaptive approach

4. Vertex reconstruction

5. Conclusions and Outlook
Track finding

Global vs. local

- Rough distinction: **local/sequential** and **global/parallel** methods

- **Local** method: generate seeds and complete them to track candidates

- **Global** method: simultaneous clustering of detector hits into track candidates
Track finding

Global vs. local

- Rough distinction: **local/sequential** and **global/parallel** methods
- **Local** method: generate seeds and complete them to track candidates
- **Global** method: simultaneous clustering of detector hits into track candidates
Global vs. local

- Rough distinction: **local/sequential** and **global/parallel** methods
- **Local** method: generate seeds and complete them to track candidates
- **Global** method: simultaneous clustering of detector hits into track candidates
Outline

1 Introduction

2 Track finding
   - Local methods
   - Global methods

3 Track fitting
   - Traditional approach
   - Adaptive approach

4 Vertex reconstruction

5 Conclusions and Outlook
Track finding: Local methods

Some local methods

- Track road
- Track following
- Progressive track finding
Track finding: Local methods

Progressive track finding

- Construct initial track segment (seed)
- Extrapolate seed
- Select best matching hit inside tolerance
- Update track parameters (weighted mean)
- Repeat until last detector layer
- Billoir and Qian (1990a), Billoir and Qian (1990b)
Track finding: Local methods

Example: progressive track finding
1 Introduction

2 Track finding
   ■ Local methods
   ■ Global methods

3 Track fitting
   ■ Traditional approach
   ■ Adaptive approach

4 Vertex reconstruction

5 Conclusions and Outlook
Track finding: Global methods

Some global methods

- Hough transform
- Legendre transform
- Hopfield network
- Elastic net
- Cellular automaton
Track finding: Global methods

Track finding with a Hopfield network

- First adaptive approach to track reconstruction (Denby, 1988; Peterson, 1989)
- Track segments are neurons of a recursive ANN
- Network weights favor smooth tracks without bifurcations
- Energy function is minimized by gradient descent
- Deterministic annealing helps to find global optimum
- No physical track model
Track finding with a Hopfield network

- First adaptive approach to track reconstruction (Denby, 1988; Peterson, 1989)
- Track segments are neurons of a recursive ANN
- Network weights favor smooth tracks without bifurcations
- Energy function is minimized by gradient descent
- Deterministic annealing helps to find global optimum
- No physical track model
Track finding: Global methods

Track finding with a Hopfield network

- First adaptive approach to track reconstruction (Denby, 1988; Peterson, 1989)
- Track segments are neurons of a recursive ANN
- Network weights favor smooth tracks without bifurcations
- Energy function is minimized by gradient descent
- Deterministic annealing helps to find global optimum
- No physical track model
Track finding: Global methods

Track finding with a Hopfield network

- First adaptive approach to track reconstruction (Denby, 1988; Peterson, 1989)
- Track segments are neurons of a recursive ANN
- Network weights favor smooth tracks without bifurcations
- Energy function is minimized by gradient descent
  - Deterministic annealing helps to find global optimum
  - No physical track model
Track finding: Global methods

Track finding with a Hopfield network

- First adaptive approach to track reconstruction (Denby, 1988; Peterson, 1989)
- Track segments are neurons of a recursive ANN
- Network weights favor smooth tracks without bifurcations
- Energy function is minimized by gradient descent
- Deterministic annealing helps to find global optimum
- No physical track model
Track finding: Global methods

Track finding with a Hopfield network

- First adaptive approach to track reconstruction (Denby, 1988; Peterson, 1989)
- Track segments are neurons of a recursive ANN
- Network weights favor smooth tracks without bifurcations
- Energy function is minimized by gradient descent
- Deterministic annealing helps to find global optimum
- No physical track model
Track finding: Global methods

Example: Hopfield network

From: Stimpfl-Abele and Garrido, CPC 64 (1991) 46
Outline

1 Introduction

2 Track finding
   - Local methods
   - Global methods

3 Track fitting
   - Traditional approach
   - Adaptive approach

4 Vertex reconstruction

5 Conclusions and Outlook
Outline

1. Introduction
2. Track finding
   - Local methods
   - Global methods
3. Track fitting
   - Traditional approach
   - Adaptive approach
4. Vertex reconstruction
5. Conclusions and Outlook
Track fitting: Traditional approach

Least-squares estimation

- Take track candidate and pass it to a least-squares estimator
- Three types: global, recursive, breakpoints
  - **Global**: Set up regression model
  - **Breakpoints**: Estimate track segments and multiple scattering angles
  - **Recursive**: Interpret track as dynamic system and estimate with extended Kalman filter
- All three are **optimal** in the linear model with normal noise
Track fitting: Traditional approach

Least-squares estimation

- Take track candidate and pass it to a least-squares estimator
- Three types: global, recursive, breakpoints
- **Global**: Set up regression model
- **Breakpoints**: Estimate track segments and multiple scattering angles
- **Recursive**: Interpret track as dynamic system and estimate with extended Kalman filter
- All three are **optimal** in the linear model with normal noise
Track fitting: Traditional approach

Least-squares estimation

- Take track candidate and pass it to a least-squares estimator
- Three types: global, recursive, breakpoints
- **Global**: Set up regression model
- **Breakpoints**: Estimate track segments and multiple scattering angles
- **Recursive**: Interpret track as dynamic system and estimate with extended Kalman filter
- All three are **optimal** in the linear model with normal noise
Track fitting: Traditional approach

Least-squares estimation

- Take track candidate and pass it to a least-squares estimator
- Three types: global, recursive, breakpoints
  - Global: Set up regression model
  - Breakpoints: Estimate track segments and multiple scattering angles
  - Recursive: Interpret track as dynamic system and estimate with extended Kalman filter
- All three are optimal in the linear model with normal noise
Track fitting: Traditional approach

**Least-squares estimation**

- Take track candidate and pass it to a least-squares estimator
- Three types: global, recursive, breakpoints
  - **Global**: Set up regression model
  - **Breakpoints**: Estimate track segments and multiple scattering angles
  - **Recursive**: Interpret track as dynamic system and estimate with extended Kalman filter
- All three are optimal in the linear model with normal noise
Track fitting: Traditional approach

Least-squares estimation

- Take track candidate and pass it to a least-squares estimator
- Three types: global, recursive, breakpoints
  - **Global**: Set up regression model
  - **Breakpoints**: Estimate track segments and multiple scattering angles
  - **Recursive**: Interpret track as dynamic system and estimate with extended Kalman filter
- All three are **optimal** in the linear model with normal noise
Track fitting: Traditional approach

Regression

- In general non-linear model:

\[ m = h(x) + \epsilon, \quad \text{Cov}[\epsilon] = V = G^{-1} \]

- Minimize objective function:

\[ M(x) = (m - h(x))^T G (m - h(x)) \]
Track fitting: Traditional approach

Regression

- In general non-linear model:

\[ m = h(x) + \epsilon, \quad \text{Cov}[\epsilon] = V = G^{-1} \]

- Minimize objective function:

\[ M(x) = (m - h(x))^T G (m - h(x)) \]
Track fitting: Traditional approach

- Minimization methods: Gauss-Newton, Newton-Raphson, conjugate gradients, ... 

\[ \tilde{x} = \arg \min M(x) \]
Track fitting: Traditional approach

- Minimization methods: Gauss-Newton, Newton-Raphson, conjugate gradients, ...

\[ \tilde{x} = \arg \min M(x) \]

Test statistics

- Total \( \chi^2 \)

\[ \chi^2 = M(\tilde{x}) \]

- Standardized residuals (pulls)

\[ r = m - h(\tilde{x}), \quad \text{Cov}[r] = V - HH^T G H^{-1} H^T \]

\[ p_i = \frac{r_i}{\sigma[r_i]} \]
Breakpoints

- Explicit estimation of multiple scattering angles
- Prior information about multiple scattering angles is used:

\[
E[\vartheta_p] = 0, \quad \text{var}[\vartheta_p] = \sigma^2(m, p, d, X_0)
\]

\(\vartheta_p\) ... Projected scattering angle

\(m\) ... Mass of the particle

\(p\) ... Momentum of the particle

\(d\) ... Thickness of the material

\(X_0\) ... Radiation length of the material
Track fitting: Traditional approach

**Kalman filter**

- **Recursive**, no large matrices need to be inverted
- Estimated state vectors stay close to the actual track
- Track is described as discrete dynamic system (Frühwirth, 1987)
- **System equation:**

  \[
  x_k = f_k(x_{k-1}) + \delta_k, \quad \text{Cov}[\delta_k] = Q_k
  \]

  - \(x_k\) . . . State vector in layer \(k\) (local track parameters)
  - \(f_k\) . . . Local track model
  - \(\delta_k\) . . . Local process noise (multiple scattering)
**Track fitting: Traditional approach**

**Kalman filter**

- **Recursive**, no large matrices need to be inverted
- Estimated state vectors stay **close** to the actual track
- Track is described as **discrete dynamic system** (Frühwirth, 1987)
- **System equation**:

\[
x_k = f_k(x_{k-1}) + \delta_k, \quad \text{Cov}[\delta_k] = Q_k
\]

- \(x_k\) . . . State vector in layer \(k\) (local track parameters)
- \(f_k\) . . . Local track model
- \(\delta_k\) . . . Local process noise (multiple scattering)
Track fitting: Traditional approach

Kalman filter

- **Recursive**, no large matrices need to be inverted
- Estimated state vectors stay **close** to the actual track
- Track is described as **discrete dynamic system** (Frühwirth, 1987)

System equation:

\[ x_k = f_k(x_{k-1}) + \delta_k, \quad \text{Cov}[\delta_k] = Q_k \]

- \( x_k \): State vector in layer \( k \) (local track parameters)
- \( f_k \): Local track model
- \( \delta_k \): Local process noise (multiple scattering)
Track fitting: Traditional approach

Kalman filter

- **Recursive**, no large matrices need to be inverted
- Estimated state vectors stay **close** to the actual track
- Track is described as **discrete dynamic system** (Frühwirth, 1987)
- **System equation**:

  \[ x_k = f_k(x_{k-1}) + \delta_k, \quad \text{Cov}[\delta_k] = Q_k \]

- \( x_k \) ... State vector in layer \( k \) (local track parameters)
- \( f_k \) ... Local track model
- \( \delta_k \) ... Local process noise (multiple scattering)
Track fitting: Traditional approach

- **Measurement equation:**

  \[ m_k = h_k(x_k) + \epsilon_k, \quad \text{Cov}[\epsilon_k] = V_k \]

  - \( m_k \) ... Measurement in layer \( k \)
  - \( h_k \) ... Measurement model
  - \( \epsilon_k \) ... Measurement error

- Kalman filter proceeds **recursively** by alternating two steps:
  1. Prediction
  2. Update
**Track fitting: Traditional approach**

- **Measurement equation:**

  \[ m_k = h_k(x_k) + \epsilon_k, \quad \text{Cov}[\epsilon_k] = V_k \]

  - \( m_k \): Measurement in layer \( k \)
  - \( h_k \): Measurement model
  - \( \epsilon_k \): Measurement error

- Kalman filter proceeds **recursively** by alternating two steps:

  1. Prediction
  2. Update
Track fitting: Traditional approach

**Measurement equation:**

\[ m_k = h_k(x_k) + \epsilon_k, \quad \text{Cov}[\epsilon_k] = V_k \]

- \( m_k \) ... Measurement in layer \( k \)
- \( h_k \) ... Measurement model
- \( \epsilon_k \) ... Measurement error

**Kalman filter proceeds recursively** by alternating two steps

1. Prediction
2. Update
Track fitting: Traditional approach

**Prediction**

- **Propagate** state vector and covariance matrix to the next layer, increment covariance matrix by contributions from multiple scattering and energy loss.
Track fitting: Traditional approach

**Prediction**

- **Propagate** state vector and covariance matrix to the next layer, increment covariance matrix by contributions from multiple scattering and energy loss

**Update**

- Compute a **weighted mean** of the extrapolation and the observation
Track fitting: Traditional approach

Prediction

- Propagate state vector and covariance matrix to the next layer, increment covariance matrix by contributions from multiple scattering and energy loss.

Update

- Compute a weighted mean of the extrapolation and the observation.

Test

- Local $\chi^2$ statistic, ndf equals dimension of $m_k$
- Total $\chi^2$, sum of local $\chi^2$ statistics
Track fitting: Traditional approach

**Prediction and filter step**

\[ x \]

Surface \( k - 1 \)  Scattering matter  Surface \( k \)

Filtered state \( x_{k-1|k-1} \)

Predicted state \( x_{k|k-1} \)

Filtered state \( x_{k|k} \)

Measurement \( m_k \)
Track fitting: Traditional approach

Smoothing

- Optimal estimation of state vectors in each layer
- Standard algorithm, numerically unstable
- Combination of two filters (forward + backward) by a weighted mean, numerically stable
Smoothing

- Optimal estimation of state vectors in each layer
- Standard algorithm, numerically unstable
- Combination of two filters (forward + backward) by a weighted mean, numerically stable

Test statistics

- Local $\chi^2$ s of the filter
- Total $\chi^2$ of the filter (sum of local $\chi^2$s)
- Local $\chi^2$s of the smoother (correlated)
Outline

1. Introduction

2. Track finding
   - Local methods
   - Global methods

3. Track fitting
   - Traditional approach
   - Adaptive approach

4. Vertex reconstruction

5. Conclusions and Outlook
Track fitting: Adaptive approach

Problems of LS estimators

😊 The Kalman filter is LS-estimator, not robust
😊 Difficult to identify multiple outliers (bias)
Track fitting: Adaptive approach

Problems of LS estimators

- The Kalman filter is LS-estimator, **not robust**
- Difficult to identify multiple outliers (bias)

Advantage of adaptive estimators

- **Concurrent** pattern recognition and estimation
  - Defer final decision to fitting stage
  - Complete information available
- **Automatic** suppression of background
  - Reduction of bias
  - No need to remove/add hits during fit
  - “Soft” assignment during entire fit
  - Can be made “hard” after optimal solution has been found
Track fitting: Adaptive approach

Problems of LS estimators

😊 The Kalman filter is LS-estimator, **not robust**
😊 Difficult to identify multiple outliers (bias)

Advantage of adaptive estimators

😊 **Concurrent** pattern recognition and estimation
  - Defer final decision to fitting stage
  - Complete information available

😊 **Automatic** suppression of background
  - Reduction of bias
  - No need to remove/add hits during fit
  - “Soft” assignment during entire fit
  - Can be made “hard” after optimal solution has been found
Track fitting: Adaptive approach

Various implementations

- **Elastic arms**: Ohlsson and Peterson (1992)
  - Based on neural network paradigm

- **Elastic tracking**: Gyulassi and Harlander (1991)
  - Inspired by Radon transform

- **Combinatorial Kalman filter**: Mankel (1997)
  - Full discrete combinatorial exploration

- **Gaussian-sum filter**: Frühwirth (1997)
  - Based on mixture models of noise

- **Deterministic annealing filter**: Frühwirth and Strandlie (1999)
  - Inspired by EM algorithm
Introduction
Track finding
Track fitting
Vertex reconstruction
Conclusions and Outlook

Track fitting: Adaptive approach

Various implementations

- **Elastic arms**: Ohlsson and Peterson (1992)
  - Based on neural network paradigm
- **Elastic tracking**: Gyulassi and Harlander (1991)
  - Inspired by Radon transform
- **Combinatorial Kalman filter**: Mankel (1997)
  - Full discrete combinatorial exploration
- **Gaussian-sum filter**: Frühwirth (1997)
  - Based on mixture models of noise
- **Deterministic annealing filter**: Frühwirth and Strandlie (1999)
  - Inspired by EM algorithm
**Track fitting: Adaptive approach**

**Various implementations**

- **Elastic arms**: Ohlsson and Peterson (1992)
  - Based on neural network paradigm

- **Elastic tracking**: Gyulassi and Harlander (1991)
  - Inspired by Radon transform

- **Combinatorial Kalman filter**: Mankel (1997)
  - Full discrete combinatorial exploration

- **Gaussian-sum filter**: Frühwirth (1997)
  - Based on mixture models of noise

- **Deterministic annealing filter**: Frühwirth and Strandlie (1999)
  - Inspired by EM algorithm
Track fitting: Adaptive approach

Various implementations

- **Elastic arms**: Ohlsson and Peterson (1992)
  - Based on neural network paradigm

- **Elastic tracking**: Gyulassi and Harlander (1991)
  - Inspired by Radon transform

- **Combinatorial Kalman filter**: Mankel (1997)
  - Full discrete combinatorial exploration

- **Gaussian-sum filter**: Frühwirth (1997)
  - Based on mixture models of noise

- **Deterministic annealing filter**: Frühwirth and Strandlie (1999)
  - Inspired by EM algorithm
Track fitting: Adaptive approach

Various implementations

- **Elastic arms**: Ohlsson and Peterson (1992)
  - Based on neural network paradigm

- **Elastic tracking**: Gyulassi and Harlander (1991)
  - Inspired by Radon transform

- **Combinatorial Kalman filter**: Mankel (1997)
  - Full discrete combinatorial exploration

- **Gaussian-sum filter**: Frühwirth (1997)
  - Based on mixture models of noise

- **Deterministic annealing filter**: Frühwirth and Strandlie (1999)
  - Inspired by EM algorithm
Track fitting: Adaptive approach

Combinatorial Kalman filter

- Extension of progressive track finding
- Full combinatorial exploration
- Several candidates are propagated in parallel
- Generate a branch for each compatible hit
- Generate a branch with a missing hit (optional)
- Limit growth by dropping branches
  - with bad total chi-square
  - with too many missing hits
  - which are subsets of other candidates
- Select “best” branch as final track
Track fitting: Adaptive approach

Combinatorial Kalman filter

- Extension of progressive track finding
- Full combinatorial exploration
- Several candidates are propagated in parallel
- Generate a branch for each compatible hit
- Generate a branch with a missing hit (optional)
- Limit growth by dropping branches
  - with bad total chi-square
  - with too many missing hits
  - which are subsets of other candidates
- Select “best” branch as final track
Track fitting: Adaptive approach

Combinatorial Kalman filter

- Extension of progressive track finding
- Full combinatorial exploration
- Several candidates are propagated in parallel
- Generate a branch for each compatible hit
- Generate a branch with a missing hit (optional)
- Limit growth by dropping branches
  - with bad total chi-square
  - with too many missing hits
  - which are subsets of other candidates
- Select “best” branch as final track
Track fitting: Adaptive approach

Combinatorial Kalman filter

- Extension of progressive track finding
- Full combinatorial exploration
- Several candidates are propagated in parallel
- Generate a branch for each compatible hit
- Generate a branch with a missing hit (optional)
- Limit growth by dropping branches
  - with bad total chi-square
  - with too many missing hits
  - which are subsets of other candidates
- Select “best” branch as final track
Track fitting: Adaptive approach

Combinatorial Kalman filter

- Extension of progressive track finding
- Full combinatorial exploration
- Several candidates are propagated in parallel
- Generate a branch for each compatible hit
- Generate a branch with a missing hit (optional)
- Limit growth by dropping branches
  - with bad total chi-square
  - with too many missing hits
  - which are subsets of other candidates
- Select “best” branch as final track
Track fitting: Adaptive approach

Combinatorial Kalman filter

- Extension of progressive track finding
- Full combinatorial exploration
- Several candidates are propagated in parallel
- Generate a branch for each compatible hit
- Generate a branch with a missing hit (optional)
- Limit growth by dropping branches
  - with bad total chi-square
  - with too many missing hits
  - which are subsets of other candidates
- Select “best” branch as final track
Track fitting: Adaptive approach

Combinatorial Kalman filter

- Extension of progressive track finding
- Full combinatorial exploration
- Several candidates are propagated in parallel
- Generate a branch for each compatible hit
- Generate a branch with a missing hit (optional)
- Limit growth by dropping branches
  - with bad total chi-square
  - with too many missing hits
  - which are subsets of other candidates
- Select “best” branch as final track
Track fitting: Adaptive approach

Combinatorial Kalman filter

- Extension of progressive track finding
- Full combinatorial exploration
- Several candidates are propagated in parallel
- Generate a branch for each compatible hit
- Generate a branch with a missing hit (optional)
- Limit growth by dropping branches
  - with bad total chi-square
  - with too many missing hits
  - which are subsets of other candidates
- Select “best” branch as final track
Track fitting: Adaptive approach

Combinatorial Kalman filter

- Extension of progressive track finding
- Full combinatorial exploration
- Several candidates are propagated in parallel
- Generate a branch for each compatible hit
- Generate a branch with a missing hit (optional)
- Limit growth by dropping branches
  - with bad total chi-square
  - with too many missing hits
  - which are subsets of other candidates
- Select “best” branch as final track
Track fitting: Adaptive approach

Combinatorial Kalman filter

- Extension of progressive track finding
- Full combinatorial exploration
- Several candidates are propagated in parallel
- Generate a branch for each compatible hit
- Generate a branch with a missing hit (optional)
- Limit growth by dropping branches
  - with bad total chi-square
  - with too many missing hits
  - which are subsets of other candidates
- Select “best” branch as final track
Track fitting: Adaptive approach

Example: Combinatorial Kalman filter

From: R. Mankel, NIM A 395 (1997) 169
Track fitting: Adaptive approach

The Gaussian-sum filter

- Kalman filter suboptimal with long-tailed noise
- Tails in the measurement errors
- Tails in the angular distribution of multiple scattering
- Energy loss by ionization and bremsstrahlung is non-Gaussian
- Modeling by a Gaussian sum (mixture)
- Implementation by parallel Kalman filters
Track fitting: Adaptive approach

Mathematics of the GSF

- **\( x \)** is track state at a material layer or a measurement layer.
- Its **prior** is a mixture of **\( K \)** multivariate Gaussians:

\[
f(x) = \sum_{k=1}^{K} \pi_k \varphi(x; x_k, C_k), \quad \sum_{k=1}^{K} \pi_k = 1
\]

- At a material layer, the process noise (multiple scattering or bremsstrahlung) is modelled by a mixture of **\( M \)** Gaussians:

\[
g(x) = \sum_{m=1}^{M} w_m \varphi(x; \mu_m, Q_m), \quad \sum_{m=1}^{M} w_m = 1
\]

- **Posterior** is mixture of all pairwise convolutions:

\[
p(x) = \sum_{k=1}^{K} \sum_{m=1}^{M} \pi_k w_m \varphi(x; x_k + \mu_m, C_k + Q_m)
\]
Track fitting: Adaptive approach

- At a measurement layer, the measurement error is modelled by a Gaussian mixture with $M$ components:

$$h(m) = \sum_{m=1}^{M} \omega_m \varphi(m; Hx + c_m, V_m), \quad \sum_{m=1}^{M} \omega_m = 1$$

- Posterior is computed via Bayes’ theorem:

$$p(x) = \sum_{k=1}^{K} \sum_{m=1}^{M} p_{km} \varphi(x; x_{km}, C_{km})$$

with

$$x_{km} = x_k + C_{km} H^T V_m^{-1} (m - c_m - Hx_k)$$

$$C_{km} = [C_k^{-1} + H^T V_m^{-1} H]^{-1}$$

$$p_{km} \propto \pi_k \omega_m \varphi(m; Hx_k + c_m, V_m + HC_k H^T), \quad \sum_k \sum_m p_{km} = 1$$
Track fitting: Adaptive approach

Implementation of the GSF

- Exponentially increasing number of components in the posterior distributions
- For practical purposes the number of components has to be limited
- Depending on the application, this can be achieved by keeping the $N$ components with largest posterior weights, or by merging components being close in parameter space, closeness defined by an appropriate similarity metric
- **Computationally intensive**, only in special situations
Track fitting: Adaptive approach

GSF with electrons

Normalized momentum residuals of electrons without (left) and with (right) the vertex constraint at $p_T = 10\text{ GeV}/c$. GSF (KF) results are shown as solid (open) histograms.
Track fitting: Adaptive approach

Elastic arms, deformable templates

- First truly adaptive estimator
- Arms or templates are parameterized tracks
- Concurrent solution of two optimization problems
  - Continuous: minimize least-squares objective function
  - Discrete: decide which hit belongs to which template
- Discrete problem is transformed into a continuous one by deterministic annealing
- Minimization of the resulting non-quadratic energy function at each temperature
Track fitting: Adaptive approach

Elastic arms, deformable templates

- First truly adaptive estimator
- Arms or templates are parameterized tracks
- Concurrent solution of two optimization problems
  - Continuous: minimize least-squares objective function
  - Discrete: decide which hit belongs to which template
- Discrete problem is transformed into a continuous one by deterministic annealing
- Minimization of the resulting non-quadratic energy function at each temperature
Track fitting: Adaptive approach

Elastic arms, deformable templates

- First truly adaptive estimator
- Arms or templates are parameterized tracks
- Concurrent solution of two optimization problems
  - Continuous: minimize least-squares objective function
  - Discrete: decide which hit belongs to which template
- Discrete problem is transformed into a continuous one by deterministic annealing
- Minimization of the resulting non-quadratic energy function at each temperature
Track fitting: Adaptive approach

Elastic arms, deformable templates

- First truly adaptive estimator
- Arms or templates are parameterized tracks
- Concurrent solution of two optimization problems
  - Continuous: minimize least-squares objective function
  - Discrete: decide which hit belongs to which template
- Discrete problem is transformed into a continuous one by **deterministic annealing**
- Minimization of the resulting non-quadratic energy function at each temperature
Track fitting: Adaptive approach

Elastic arms, deformable templates

- First truly adaptive estimator
- Arms or templates are parameterized tracks
- Concurrent solution of two optimization problems
  - Continuous: minimize least-squares objective function
  - Discrete: decide which hit belongs to which template
- Discrete problem is transformed into a continuous one by deterministic annealing
- Minimization of the resulting non-quadratic energy function at each temperature
Track fitting: Adaptive approach

Elastic tracking

- Define the energy function by a sum over measurements $i$ and tracks $j$:

$$R_V(t) = -\sum_{i=1}^{n} \sum_{j=1}^{m} V(\chi_{ij}^2, t)$$

with

$$V(\chi_{ij}^2, t) = \frac{w^2(t) (I)}{\chi_{ij}^2 + w^2(t)}$$

where $\chi_{ij}^2$ measures the **distance** of observation $i$ from track $j$

- $w^2(t)$ is chosen large in the beginning to smooth out the energy surface.

- It is decreased to a value compatible with the standard deviation of the measurement error.
Elastic tracking

- Define the energy function by a sum over measurements \( i \) and tracks \( j \):

\[
R_V(t) = - \sum_{i=1}^{n} \sum_{j=1}^{m} V(\chi_{ij}^2, t)
\]

with

\[
V(\chi_{ij}^2, t) = \frac{w^2(t)(I)}{\chi_{ij}^2 + w^2(t)}
\]

where \( \chi_{ij}^2 \) measures the distance of observation \( i \) from track \( j \)

- \( w^2(t) \) is chosen large in the beginning to smooth out the energy surface.

- It is decreased to a value compatible with the standard deviation of the measurement error.
**Track fitting: Adaptive approach**

**Elastic tracking**

- Define the energy function by a sum over measurements $i$ and tracks $j$:

$$R_V(t) = - \sum_{i=1}^{n} \sum_{j=1}^{m} V(\chi_{ij}^2, t)$$

with

$$V(\chi_{ij}^2, t) = \frac{w^2(t) (I)}{\chi_{ij}^2 + w^2(t)}$$

where $\chi_{ij}^2$ measures the **distance** of observation $i$ from track $j$

- $w^2(t)$ is chosen large in the beginning to smooth out the energy surface.

- It is decreased to a value compatible with the standard deviation of the measurement error.
Track fitting: Adaptive approach

Deterministic annealing filter (DAF)

- Same principle as elastic arms
- Minimization by EM algorithm, implemented as iterated re-weighted Kalman filter
- Easy to deal with process noise
- Observations are assigned weights
- Iteration of two principal steps
  1. Full Kalman filter + smoother, using the current weights
  2. Calculation of weights, using current estimates
- The iteration ends when the weights are stable
Track fitting: Adaptive approach

Deterministic annealing filter (DAF)

- Same principle as elastic arms
- Minimization by EM algorithm, implemented as \textit{iterated re-weighted Kalman filter}
- Easy to deal with process noise
- Observations are assigned \textit{weights}
- Iteration of two principal steps
  1. Full Kalman filter+smoother, using the current weights
  2. Calculation of weights, using current estimates
- The iteration ends when the weights are stable
Track fitting: Adaptive approach

Deterministic annealing filter (DAF)

- Same principle as elastic arms
- Minimization by EM algorithm, implemented as **iterated re-weighted Kalman filter**
- Easy to deal with process noise
  - Observations are assigned **weights**
  - Iteration of two principal steps
    1. Full Kalman filter + smoother, using the current weights
    2. Calculation of weights, using current estimates
- The iteration ends when the weights are stable
Track fitting: Adaptive approach

Deterministic annealing filter (DAF)

- Same principle as elastic arms
- Minimization by EM algorithm, implemented as *iterated re-weighted Kalman filter*
- Easy to deal with process noise
- Observations are assigned *weights*
- Iteration of two principal steps
  1. Full Kalman filter+smoother, using the current weights
  2. Calculation of weights, using current estimates
- The iteration ends when the weights are stable
Track fitting: Adaptive approach

Deterministic annealing filter (DAF)

- Same principle as elastic arms
- Minimization by EM algorithm, implemented as \textit{iterated}
  \textit{re-weighted Kalman filter}
- Easy to deal with process noise
- Observations are assigned \textbf{weights}
- Iteration of two principal steps
  1. Full Kalman filter+smoother, using the current weights
  2. Calculation of weights, using current estimates
- The iteration ends when the weights are stable
Track fitting: Adaptive approach

Deterministic annealing filter (DAF)

- Same principle as elastic arms
- Minimization by EM algorithm, implemented as **iterated re-weighted Kalman filter**
- Easy to deal with process noise
- Observations are assigned **weights**
- Iteration of two principal steps
  
  1. Full Kalman filter + smoother, using the current weights
  2. Calculation of weights, using current estimates
- The iteration ends when the weights are stable
Track fitting: Adaptive approach

Deterministic annealing filter (DAF)

- Same principle as elastic arms
- Minimization by EM algorithm, implemented as **iterated re-weighted Kalman filter**
- Easy to deal with process noise
- Observations are assigned **weights**
- Iteration of two principal steps
  1. Full Kalman filter + smoother, using the current weights
  2. Calculation of weights, using current estimates
- The iteration ends when the weights are stable
Track fitting: Adaptive approach

Deterministic annealing filter (DAF)

- Same principle as elastic arms
- Minimization by EM algorithm, implemented as **iterated re-weighted Kalman filter**
- Easy to deal with process noise
- Observations are assigned **weights**
- Iteration of two principal steps
  1. Full Kalman filter+smoother, using the current weights
  2. Calculation of weights, using current estimates
- The iteration ends when the weights are stable
Track fitting: Adaptive approach

**Definition of the weights**

- **Weight of observation** $i$ in layer $k$:

\[
p_{ik} = \frac{\exp(-\chi_{ik}^2/2T)}{\exp(-\chi_{cut}^2/2T) + \sum_j \exp(-\chi_{jk}^2/2T)}
\]

- $\chi_{ik}^2$ measures the **distance** of observation $i$ in layer $k$ from the smoothed track state in layer $k$.
- $\chi_{cut}^2$ is the **cut-off** parameter.
- $T$ is the **annealing** factor (temperature).
- For a single observation $p_{ik} = 0.5$ if $\chi_{ik}^2 = \chi_{cut}^2$. 
Track fitting: Adaptive approach

Definition of the weights

- Weight of observation $i$ in layer $k$:

$$p_{ik} = \frac{\exp(-\chi_{ik}^2/2T)}{\exp(-\chi_{\text{cut}}^2/2T) + \sum_j \exp(-\chi_{jk}^2/2T)}$$

- $\chi_{ik}^2$ measures the **distance** of observation $i$ in layer $k$ from the smoothed track state in layer $k$.

- $\chi_{\text{cut}}^2$ is the **cut-off** parameter.

- $T$ is the **annealing** factor (temperature).

- For a single observation $p_{ik} = 0.5$ if $\chi_{ik}^2 = \chi_{\text{cut}}^2$. 

Track fitting: Adaptive approach

Definition of the weights

- Weight of observation $i$ in layer $k$:

$$p_{ik} = \frac{\exp(-\chi^2_{ik}/2T)}{\exp(-\chi^2_{\text{cut}}/2T) + \sum_j \exp(-\chi^2_{jk}/2T)}$$

- $\chi^2_{ik}$ measures the **distance** of observation $i$ in layer $k$ from the smoothed track state in layer $k$.
- $\chi^2_{\text{cut}}$ is the **cut-off** parameter.
- $T$ is the **annealing** factor (temperature).
- For a single observation $p_{ik} = 0.5$ if $\chi^2_{ik} = \chi^2_{\text{cut}}$. 
Track fitting: Adaptive approach

Definition of the weights

- Weight of observation $i$ in layer $k$:

$$p_{ik} = \frac{\exp(-\chi_{ik}^2/2T)}{\exp(-\chi_{cut}^2/2T) + \sum_j \exp(-\chi_{jk}^2/2T)}$$

- $\chi_{ik}^2$ measures the distance of observation $i$ in layer $k$ from the smoothed track state in layer $k$
- $\chi_{cut}^2$ is the cut-off parameter
- $T$ is the annealing factor (temperature)
- For a single observation $p_{ik} = 0.5$ if $\chi_{ik}^2 = \chi_{cut}^2$
Track fitting: Adaptive approach

Definition of the weights

- Weight of observation $i$ in layer $k$:

$$p_{ik} = \frac{\exp(-\chi_{ik}^2/2T)}{\exp(-\chi_{cut}^2/2T) + \sum_j \exp(-\chi_{jk}^2/2T)}$$

- $\chi_{ik}^2$ measures the distance of observation $i$ in layer $k$ from the smoothed track state in layer $k$
- $\chi_{cut}^2$ is the cut-off parameter
- $T$ is the annealing factor (temperature)
- For a single observation $p_{ik} = 0.5$ if $\chi_{ik}^2 = \chi_{cut}^2$
Track fitting: Adaptive approach

Weight function without competition

Weight function of an observation without competition
Track fitting: Adaptive approach

Example: 1D data with outliers

- Estimate location of the bulk of the data

![Graph showing 1D data with outliers]
Track fitting: Adaptive approach

Evolution of the objective function

- $M(\mu; c, T)$ for $m=5$, $T=5$
- $M(\mu; c, T)$ for $m=5$, $T=2$
- $M(\mu; c, T)$ for $m=5$, $T=1$
- $M(\mu; c, T)$ for $m=5$, $T=0.1$
Track fitting: Adaptive approach

Example: 2D clustering of peptides
Track fitting: Adaptive approach

Weight function with competition

- If there are several observations in a detector layer, they may compete with each other.
- A matching observation suppresses the other ones.
- Deterministic Annealing helps to reach the optimal solution:
  - At the start $T \gg 1$
  - During the iteration $T$ is stepped down
  - The final value is $T = 1$
- Well-known technique of global optimization, e.g. with ANNs.
- At $T > 0$ the association is “soft”.
- “Cooling down” to $T = 0$ yields “hard” association. Not necessarily optimal!
Track fitting: Adaptive approach

**Weight function with competition**

- If there are several observations in a detector layer, they may *compete* with each other.
- A matching observations *suppresses* the other ones.
- **Deterministic Annealing** helps to reach the optimal solution:
  - At the start $T \gg 1$.
  - During the iteration $T$ is stepped down.
  - The final value is $T = 1$.

- Well-known technique of global optimization, e.g. with ANNs.
- At $T > 0$ the association is “soft”.
- “Cooling down” to $T = 0$ yields “hard” association. Not necessarily optimal!
Track fitting: Adaptive approach

Weight function with competition

- If there are several observations in a detector layer, they may compete with each other.
- A matching observations suppresses the other ones.
- **Deterministic Annealing** helps to reach the optimal solution:
  - At the start $T \gg 1$
  - During the iteration $T$ is stepped down
  - The final value is $T = 1$

- Well-known technique of global optimization, e.g. with ANNs
- At $T > 0$ the association is “soft”
- “Cooling down” to $T = 0$ yields “hard” association. Not necessarily optimal!
Track fitting: Adaptive approach

**Weight function with competition**

- If there are several observations in a detector layer, they may **compete** with each other.
- A matching observation suppresses the other ones.
- **Deterministic Annealing** helps to reach the optimal solution:
  - At the start $T \gg 1$
  - During the iteration $T$ is stepped down
  - The final value is $T = 1$
- Well-known technique of global optimization, e.g. with ANNs.
- At $T > 0$ the association is “soft”
- “Cooling down” to $T = 0$ yields “hard” association. Not necessarily optimal!
Weight function with competition

- If there are several observations in a detector layer, they may compete with each other.
- A matching observation suppresses the other ones.
- **Deterministic Annealing** helps to reach the optimal solution:
  - At the start $T \gg 1$
  - During the iteration $T$ is stepped down
  - The final value is $T = 1$
- Well-known technique of global optimization, e.g. with ANNs.
- At $T > 0$ the association is “soft.”
- “Cooling down” to $T = 0$ yields “hard” association. Not necessarily optimal!
Track fitting: Adaptive approach

Weight function with competition

- If there are several observations in a detector layer, they may compete with each other.
- A matching observations suppresses the other ones.
- **Deterministic Annealing** helps to reach the optimal solution.
  - At the start $T \gg 1$
  - During the iteration $T$ is stepped down.
  - The final value is $T = 1$
- Well-known technique of global optimization, e.g. with ANNs.
- At $T > 0$ the association is “soft”.
- “Cooling down” to $T = 0$ yields “hard” association. Not necessarily optimal!
Track fitting: Adaptive approach

Weight function of the DAF

Weight function of an observation with competition
Track fitting: Adaptive approach

Example: DAF with and without annealing

From: Strandlie and Zerubia, CPC 123 (1999) 77
Track fitting: Adaptive approach

History of the DAF

- DAF was evaluated in CMS (Winkler, 2002)
- Studies with single tracks
- Studies in different physics contexts
- Implemented in CMS offline framework
- Implemented in ATLAS offline framework
- Studies of track finding with the DAF (Strandlie and Frühwirth, 2006)
Track fitting: Adaptive approach

Extension to multi-track fit

- Multi-track filter (MTF) was first studied by Strandlie and Frühwirth (2000), using pairs of tracks in the ATLAS barrel TRT.

- Several competition schemes are possible:

1. **Competition between hits.** Competition between all hits for each track, but no competition between the tracks. Equivalent to the DAF.

2. **Competition between tracks.** Competition between all tracks for each hit, but no competition between the hits. Equivalent to the original version of the EAA.

3. **Global competition.** All tracks compete for all hits.

4. **Competition between tracks and between mirror hits.** Refinement of scheme 2, specific for detectors with mirror hits.
### Performance of multi-track fit

<table>
<thead>
<tr>
<th>Noise level</th>
<th>Competition scheme 1</th>
<th>Competition scheme 2</th>
<th>Competition scheme 3</th>
<th>Competition scheme 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0%</td>
<td>281</td>
<td>36.2</td>
<td>4.52</td>
<td>2.84</td>
</tr>
<tr>
<td>10%</td>
<td>270</td>
<td>58.7</td>
<td>5.35</td>
<td>4.35</td>
</tr>
<tr>
<td>20%</td>
<td>388</td>
<td>100.9</td>
<td>6.26</td>
<td>7.06</td>
</tr>
<tr>
<td>30%</td>
<td>358</td>
<td>185.3</td>
<td>7.19</td>
<td>9.51</td>
</tr>
<tr>
<td>40%</td>
<td>409</td>
<td>238.9</td>
<td>9.50</td>
<td>12.44</td>
</tr>
<tr>
<td>50%</td>
<td>653</td>
<td>301.6</td>
<td>11.66</td>
<td>17.65</td>
</tr>
</tbody>
</table>

Relative generalized variance of the Multi-Track Fit with competition schemes 1–4, at various noise levels with mirror hits. The baseline is a single-track fit of all tracks separately. From: Strandlie and Frühwirth (2000)
Outline

1. Introduction

2. Track finding
   - Local methods
   - Global methods

3. Track fitting
   - Traditional approach
   - Adaptive approach

4. Vertex reconstruction

5. Conclusions and Outlook
Vertex finding

- Hierarchical clustering (agglomerative or divisive)
- Topological finding, Radon transform (Jackson, 1997)
- Minimum spanning tree (Hillert, 2008)
- Multi-layer perceptron (Lindsey and Denby, 1991)
- Adaptive vertex reconstructor (see below)
Vertex reconstruction

**Vertex finding**

- Hierarchical clustering (agglomerative or divisive)
- Topological finding, Radon transform (Jackson, 1997)
- Minimum spanning tree (Hillert, 2008)
- Multi-layer perceptron (Lindsey and Denby, 1991)
- Adaptive vertex reconstructor (see below)
Vertex reconstruction

**Vertex finding**

- Hierarchical clustering (agglomerative or divisive)
- Topological finding, Radon transform (Jackson, 1997)
- Minimum spanning tree (Hillert, 2008)
- Multi-layer perceptron (Lindsey and Denby, 1991)
- Adaptive vertex reconstructor (see below)
**Vertex finding**

- Hierarchical clustering (agglomerative or divisive)
- Topological finding, Radon transform (Jackson, 1997)
- Minimum spanning tree (Hillert, 2008)
- Multi-layer perceptron (Lindsey and Denby, 1991)
- Adaptive vertex reconstructor (see below)
Vertex finding

- Hierarchical clustering (agglomerative or divisive)
- Topological finding, Radon transform (Jackson, 1997)
- Minimum spanning tree (Hillert, 2008)
- Multi-layer perceptron (Lindsey and Denby, 1991)
- Adaptive vertex reconstructor (see below)
Adaptive vertex estimation

- Concepts of adaptive estimators can be transferred almost one-to-one from track to vertex fitting
- Algorithm is called **Adaptive Vertex Fitter (AVF)** (Waltenberger, 2004)
- Implemented as **iterated re-weighted Kalman filter**
- Outlying tracks are automatically down-weighted
- Resulting estimator is **highly robust**, but much easier to compute than other robust estimators such as LMS or LTS
- Extension to **Multi-Vertex Fitter (MVF)**: vertices compete for the tracks
Vertex reconstruction

Adaptive vertex estimation

- Concepts of adaptive estimators can be transferred almost one-to-one from track to vertex fitting
- Algorithm is called Adaptive Vertex Fitter (AVF) (Waltenberger, 2004)
- Implemented as iterated re-weighted Kalman filter
- Outlying tracks are automatically down-weighted
- Resulting estimator is highly robust, but much easier to compute than other robust estimators such as LMS or LTS
- Extension to Multi-Vertex Fitter (MVF): vertices compete for the tracks
Adaptive vertex estimation

- Concepts of adaptive estimators can be transferred almost one-to-one from track to vertex fitting
- Algorithm is called *Adaptive Vertex Fitter (AVF)* (Waltenberger, 2004)
- Implemented as *iterated re-weighted Kalman filter*
- Outlying tracks are automatically down-weighted
- Resulting estimator is *highly robust*, but much easier to compute than other robust estimators such as LMS or LTS
- Extension to *Multi-Vertex Fitter (MVF)*: vertices compete for the tracks
Adaptive vertex estimation

- Concepts of adaptive estimators can be transferred almost one-to-one from track to vertex fitting.
- Algorithm is called **Adaptive Vertex Fitter (AVF)** (Waltenberger, 2004).
- Implemented as **iterated re-weighted Kalman filter**.
- Outlying tracks are automatically down-weighted.
- Resulting estimator is **highly robust**, but much easier to compute than other robust estimators such as LMS or LTS.
- Extension to **Multi-Vertex Fitter (MVF)**: vertices compete for the tracks.
Vertex reconstruction

Adaptive vertex estimation

- Concepts of adaptive estimators can be transferred almost one-to-one from track to vertex fitting
- Algorithm is called Adaptive Vertex Fitter (AVF) (Waltenberger, 2004)
- Implemented as iterated re-weighted Kalman filter
- Outlying tracks are automatically down-weighted
- Resulting estimator is highly robust, but much easier to compute than other robust estimators such as LMS or LTS
- Extension to Multi-Vertex Fitter (MVF): vertices compete for the tracks
Adaptive vertex estimation

- Concepts of adaptive estimators can be transferred almost one-to-one from track to vertex fitting
- Algorithm is called **Adaptive Vertex Fitter (AVF)** (Waltenberger, 2004)
- Implemented as **iterated re-weighted Kalman filter**
- Outlying tracks are automatically down-weighted
- Resulting estimator is **highly robust**, but much easier to compute than other robust estimators such as LMS or LTS
- Extension to **Multi-Vertex Fitter (MVF)**: vertices compete for the tracks
Primary vertices estimated with AVF

Beamspot profile, CMS, first collisions at $\sqrt{s} = 900$ GeV
Vertex reconstruction

Adaptive Vertex Reconstructor (AVR)

- Vertex finding by *iterated AVF* (Waltenberger, 2008)
  - Fit all tracks to a common vertex, using the AVF
  - Remove all tracks with weight above threshold
  - Fit all remaining tracks to a common vertex, using the AVF
  - Repeat until no valid vertex can be fitted

- Implemented and successfully validated in CMS offline software
Adaptive Vertex Reconstructor (AVR)

- Vertex finding by *iterated AVF* (Waltenberger, 2008)
  - Fit all tracks to a common vertex, using the AVF
  - Remove all tracks with weight above threshold
  - Fit all remaining tracks to a common vertex, using the AVF
  - Repeat until no valid vertex can be fitted

- Implemented and successfully validated in CMS offline software
Vertex reconstruction

Adaptive Vertex Reconstructor (AVR)

- Vertex finding by *iterated AVF* (Waltenberger, 2008)
  - Fit all tracks to a common vertex, using the AVF
  - Remove all tracks with weight above threshold
  - Fit all remaining tracks to a common vertex, using the AVF
  - Repeat until no valid vertex can be fitted

- Implemented and successfully validated in CMS offline software
Vertex reconstruction

Adaptive Vertex Reconstructor (AVR)

- Vertex finding by *iterated AVF* (Waltenberger, 2008)
  - Fit all tracks to a common vertex, using the AVF
  - Remove all tracks with weight above threshold
  - Fit all remaining tracks to a common vertex, using the AVF
  - Repeat until no valid vertex can be fitted

- Implemented and successfully validated in CMS offline software
Vertex reconstruction

Adaptive Vertex Reconstructor (AVR)

- Vertex finding by *iterated AVF* (Waltenberger, 2008)
  - Fit all tracks to a common vertex, using the AVF
  - Remove all tracks with weight above threshold
  - Fit all remaining tracks to a common vertex, using the AVF
  - Repeat until no valid vertex can be fitted

- Implemented and successfully validated in CMS offline software
Vertex reconstruction

Adaptive Vertex Reconstructor (AVR)

- Vertex finding by *iterated AVF* (Waltenberger, 2008)
  - Fit all tracks to a common vertex, using the AVF
  - Remove all tracks with weight above threshold
  - Fit all remaining tracks to a common vertex, using the AVF
  - Repeat until no valid vertex can be fitted

- Implemented and successfully validated in CMS offline software
RAVE Toolbox

- CMS algorithms (KF, AVF, AVR, ...) packed into detector-independent vertex reconstruction toolkit: RAVE (Waltenberger, 2007)
- Download from http://projects.hepforge.org/rave
- Used in new Belle2 framework
- Used for ILC studies (ILD, SiLC)
**RAVE Toolbox**

- CMS algorithms (KF, AVF, AVR, ...) packed into detector-independent vertex reconstruction toolkit: **RAVE** (Waltenberger, 2007)
- Download from [http://projects.hepforge.org/rave](http://projects.hepforge.org/rave)
- Used in new Belle2 framework
- Used for ILC studies (ILD, SiLC)
RAVE Toolbox

- CMS algorithms (KF, AVF, AVR, ...) packed into detector-independent vertex reconstruction toolkit: RAVE (Waltenberger, 2007)
- Download from http://projects.hepforge.org/rave
- Used in new Belle2 framework
- Used for ILC studies (ILD, SiLC)
Vertex reconstruction

RAVE Toolbox

- CMS algorithms (KF, AVF, AVR, ...) packed into detector-independent vertex reconstruction toolkit: **RAVE** (Waltenberger, 2007)
- Download from [http://projects.hepforge.org/rave](http://projects.hepforge.org/rave)
- Used in new Belle2 framework
- Used for ILC studies (ILD, SiLC)
Outline

1. Introduction

2. Track finding
   - Local methods
   - Global methods

3. Track fitting
   - Traditional approach
   - Adaptive approach

4. Vertex reconstruction

5. Conclusions and Outlook
Adaptive estimators are useful tools for track and vertex reconstruction (Strandlie and Frühwirth, 2010)

Background is automatically down-weighted, no need for iterative rejection of outliers

Competition between hits or vertices is possible

Annealing helps to reach the globally optimal solution

Implementations are built on existing methods (Kalman filter)

Resistant to high levels of noise, important for experiments at Super-LHC and the upgraded B-factory at KEK
Summary

- **Adaptive estimators** are useful tools for track and vertex reconstruction (Strandlie and Frühwirth, 2010)

- **Background** is automatically down-weighted, no need for iterative rejection of outliers

- **Competition** between hits or vertices is possible

- **Annealing** helps to reach the globally optimal solution

- **Implementations** are built on existing methods (Kalman filter)

- **Resistant** to high levels of noise, important for experiments at Super-LHC and the upgraded B-factory at KEK
Summary

- **Adaptive estimators** are useful tools for track and vertex reconstruction (Strandlie and Frühwirth, 2010)
- **Background** is automatically down-weighted, no need for iterative rejection of outliers
- **Competition** between hits or vertices is possible
- **Annealing** helps to reach the globally optimal solution
- **Implementations** are built on existing methods (Kalman filter)
- **Resistant** to high levels of noise, important for experiments at Super-LHC and the upgraded B-factory at KEK
Summary

- **Adaptive estimators** are useful tools for track and vertex reconstruction (Strandlie and Frühwirth, 2010)
- **Background** is automatically down-weighted, no need for iterative rejection of outliers
- **Competition** between hits or vertices is possible
- **Annealing** helps to reach the globally optimal solution
- **Implementations** are built on existing methods (Kalman filter)
- **Resistant** to high levels of noise, important for experiments at Super-LHC and the upgraded B-factory at KEK
Conclusions and Outlook

Summary

- **Adaptive estimators** are useful tools for track and vertex reconstruction (Strandlie and Frühwirth, 2010)
- **Background** is automatically down-weighted, no need for iterative rejection of outliers
- **Competition** between hits or vertices is possible
- **Annealing** helps to reach the globally optimal solution
- **Implementations** are built on existing methods (Kalman filter)
- **Resistant** to high levels of noise, important for experiments at Super-LHC and the upgraded B-factory at KEK
Summary

- **Adaptive estimators** are useful tools for track and vertex reconstruction (Strandlie and Frühwirth, 2010)
- **Background** is automatically down-weighted, no need for iterative rejection of outliers
- **Competition** between hits or vertices is possible
- **Annealing** helps to reach the globally optimal solution
- **Implementations** are built on existing methods (Kalman filter)
- **Resistant** to high levels of noise, important for experiments at Super-LHC and the upgraded B-factory at KEK
Open questions

- **Parallelization** To which extent can adaptive methods be parallelized?
- **Very high noise levels** What are the optimal track finding methods at very high noise levels?
- **Sequential track fits or multi-track fit** Is a multi-track fit superior to sequential adaptive single-track fits?
- **Sequential vertex fits or multi-vertex fit** Is a multi-vertex fit superior to sequential adaptive single-vertex fits?
- **Usage of posterior weights** How can the posterior weights of DAF or GSF be optimally used in subsequent stages of the analysis (vertex fit, kinematic fit)?
Open questions

- **Parallelization** To which extent can adaptive methods be parallelized?

- **Very high noise levels** What are the optimal track finding methods at very high noise levels?

- **Sequential track fits or multi-track fit** Is a multi-track fit superior to sequential adaptive single-track fits?

- **Sequential vertex fits or multi-vertex fit** Is a multi-vertex fit superior to sequential adaptive single-vertex fits?

- **Usage of posterior weights** How can the posterior weights of DAF or GSF be optimally used in subsequent stages of the analysis (vertex fit, kinematic fit)?
Open questions

- **Parallelization** To which extent can adaptive methods be parallelized?

- **Very high noise levels** What are the optimal track finding methods at very high noise levels?

- **Sequential track fits or multi-track fit** Is a multi-track fit superior to sequential adaptive single-track fits?

- **Sequential vertex fits or multi-vertex fit** Is a multi-vertex fit superior to sequential adaptive single-vertex fits?

- **Usage of posterior weights** How can the posterior weights of DAF or GSF be optimally used in subsequent stages of the analysis (vertex fit, kinematic fit)?
Conclusions and Outlook

Open questions

■ **Parallelization** To which extent can adaptive methods be parallelized?

■ **Very high noise levels** What are the optimal track finding methods at very high noise levels?

■ **Sequential track fits or multi-track fit** Is a multi-track fit superior to sequential adaptive single-track fits?

■ **Sequential vertex fits or multi-vertex fit** Is a multi-vertex fit superior to sequential adaptive single-vertex fits?

■ **Usage of posterior weights** How can the posterior weights of DAF or GSF be optimally used in subsequent stages of the analysis (vertex fit, kinematic fit)?
Open questions

- **Parallelization** To which extent can adaptive methods be parallelized?
- **Very high noise levels** What are the optimal track finding methods at very high noise levels?
- **Sequential track fits or multi-track fit** Is a multi-track fit superior to sequential adaptive single-track fits?
- **Sequential vertex fits or multi-vertex fit** Is a multi-vertex fit superior to sequential adaptive single-vertex fits?
- **Usage of posterior weights** How can the posterior weights of DAF or GSF be optimally used in subsequent stages of the analysis (vertex fit, kinematic fit)?
Thanks to:

Thomas Speer
Wolfgang Adam
Wolfgang Waltenberger
Matthias Winkler
Selected References (1)

- P. Billoir and S. Qian, NIM A 294 (1990) 219
- P. Billoir and S. Qian, NIM A 295 (1990) 492
- T. Alexopoulos et al., NIM A 592 (2008) 456
- B. Denby, CPC 49 (1988) 429
- C. Peterson, NIM A 279 (1989) 537
- G. Stimpfl-Abele and L. Garrido, CPC 64 (1991) 46
- I. Kisel and V. Kovalenko, CPC 98 (1996) 45
- I. Kisel et al., NIM A 387 (1997) 433
- I. Abt et al., NIM A 489 (2002) 389
- R. Frühwirth, NIM A 262 (1987) 444
- M. Ohlsson and C. Peterson, CPC 71 (1992) 77
- M. Gyulassi and M. Harlander, CPC 66 (1991) 31
- R. Mankel, NIM A 395 (1997) 169
Selected References (2)

- R. Frühwirth, CPC 100 (1997) 1
- R. Frühwirth and A. Strandlie, CPC 120 (1999) 197
- A. Strandlie and J. Zerubia, CPC 123 (1999) 77
- D.J. Jackson, NIM A 388 (1997) 247
- C. Lindsey and B. Denby, NIM A 302 (1991) 217
- W. Waltenberger et al., NIM A 581 (2007) 549
- A. Strandlie and R. Frühwirth, Rev. Mod. Phys. 82 (2010) 1419