Part 2

non-linear problems

example of a straight line fit

the progressive fit

exact constraints

the weighted mean is an LSE

- least squares ('minimum chi-square') estimator

$$\hat{x} = C H^T V^{-1} (m - h_0)$$

$$C \equiv \mathrm{var}(\hat{x}) = \left(H^T V^{-1} H\right)^{-1}$$

- simplest example: weighted mean
  - consider measurements m_i with known uncertainty σ_i
  - assuming they measure the same thing 'x', what value has 'x'?

$$h_i(x) = x \qquad \Longrightarrow \qquad H^T = (1, 1, 1, \cdots)$$

$$\hat{x} = \left(\sum \frac{1}{\sigma_i^2}\right)^{-1} \sum \frac{m_i}{\sigma_i^2} \qquad \mathrm{var}(\hat{x}) = \left(\sum \frac{1}{\sigma_i^2}\right)^{-1}$$

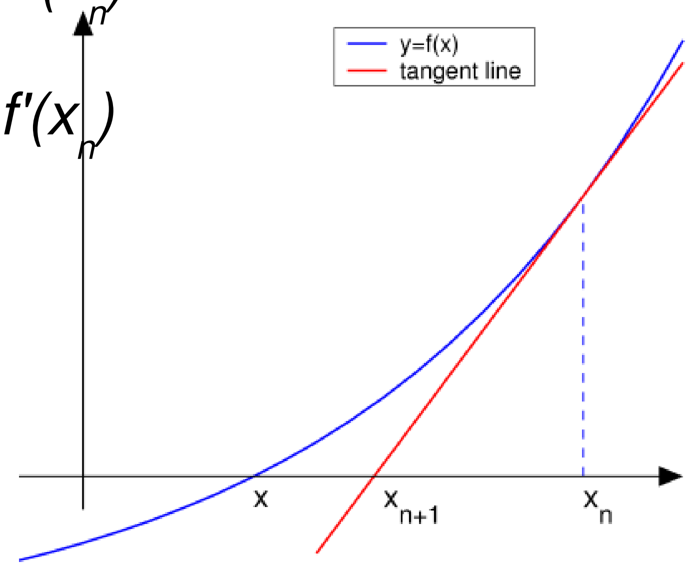
non-linear problems

- what if the measurement model *h(x)* is not linear?
- first derivative of the chi-square now looks like

$$\frac{\mathrm{d}\chi^2}{\mathrm{d}x} = 2\,\frac{\mathrm{d}h(x)}{\mathrm{d}x}^T V^{-1}\,(h(x) - m)$$

where the derivative *dh/dx* now depends on x as well

- use Newton-Raphson to find the zero-crossing '*x*' of a function *f(x)*
  - starting from an estimate $x_{n,}$ evaluate $f(x_n)$ and $f'(x_n)$
  - estimate a better value as $x_{n+1} = x_n - f(x_n) / f'(x_n)$
  - iterate until you're happy with the value of *f(x)*



non-linear problems (II)

- second derivative becomes

$$\frac{\mathrm{d}^2\chi^2}{\mathrm{d}x^2} = \underbrace{2\,\frac{\mathrm{d}h(x)}{\mathrm{d}x}^T V^{-1}\,\frac{\mathrm{d}h(x)}{\mathrm{d}x}}_{\text{this term also appears for a linear model}} + \underbrace{2\,\frac{\mathrm{d}^2h(x)}{\mathrm{d}x^2}^T V^{-1}\,(h(x)-m)}_{\text{this term is new}}$$

- the second term appears because the derivative is not constant
- in track/vertex fit applications we always drop this term, because
  - depending on how poor your starting point is it could actually make the second derivative negative, which is bad
  - if the derivative *dh/dx* varies slowly wrt the resolution, the second term is much smaller than the first
- dropping the 2nd term is equivalent to linearizing the measurement model

non-linear problems (III)

- summarizing: choosing a starting point x_0, we have

$$\frac{1}{2}\left.\frac{\mathrm{d}\chi^2}{\mathrm{d}x}\right|_{x_0} = -H^T V^{-1} \, (m - h(x_0))$$

$$\frac{1}{2}\left.\frac{\mathrm{d}^2\chi^2}{\mathrm{d}x^2}\right|_{x_0} = H^T V^{-1} H$$

$$H \equiv \left.\frac{\mathrm{d}h(x)}{\mathrm{d}x}\right|_{x_0}$$

- which, with Newton-Raphson, gives

$$\hat{x} = x_0 - \left(\frac{\mathrm{d}^2\chi^2}{\mathrm{d}x^2}\right)^{-1} \frac{\mathrm{d}\chi^2}{\mathrm{d}x}$$

expression is just the same as for linear model

- note that the variance can be written as

$$\mathrm{var}\,(x) = 2\left(\frac{\mathrm{d}^2\chi^2}{\mathrm{d}x^2}\right)^{-1}$$

- we now need a *sensible starting point* and *iterations and* repeat the calculation of derivatives and x, until we are 'close enough'

under-constrained problems

- let's look more carefully at this step

$$-2\,H^T V^{-1}\,(m - h_0 - Hx) \;=\; 0$$

minimum X^2 condition

$$\hat{x} \;=\; \underbrace{\left(H^T V^{-1} H\right)^{-1}}_{\text{matrix inversion only possible if determinant not zero!}}\; H^T V^{-1}\,(m - h_0)$$

solution

- if the determinant is zero
  - solution to minimum chi-square condition is not unique
  - some (or all) linear combinations of elements of x are not constrained, which means that they do not have finite variance
  - we call these linear combinations 'unconstrained degrees of freedom'
  - they could be isolated, e.g. by diagonalizing $H^T V^{-1} H$
- example: all linear problems with more parameters than data points
- we will not discuss problems with unconstrained DOFs

chi-square distribution

- consider the sum of N Gaussian distributed random variables (RV) 'r' with unit variance

$$z = \sum_{i=1}^{N} r_i^2$$

- this sum is itself an RV. its distribution *is the chi-square distribution with N degrees of freedom*

$$\mathcal{P}_{\chi^2}(z;N) = \frac{z^{N/2-1}\, e^{-z/2}}{2^{N/2}\,\Gamma(N/2)}$$

$$E(z) = N$$
$$\mathrm{var}(z) = 2N$$

- its cumulative distribution function

$$F(z;N) = \int_z^{\infty} \mathcal{P}(t;N)\,\mathrm{d}t$$

is the probability that a random other 'z' is larger than 'z'

- if 'z' follows a chi-square distribution, then the distribution of F(z) is 'flat' between 0 and 1.

- the value of F(z) is sometimes called the 'chi-square probability'

minimum chi-square of the LSE

- let's look again at the chi-square of our linear model

$$\chi^2 = \sum_i \left(\frac{m_i - h_0 - H x}{\sigma_i} \right)^2$$

- if everything Gaussian, then for if $x = x^{true}$ this is distributed as $\mathcal{P}_{\chi^2}(z; N)$
- let's now look at the minimum chi-square in the LSE

$$\hat{\chi}^2 = (m - h_0 - H\hat{x})^T V^{-1} (m - h_0 - H\hat{x})$$

- filling in the solution for x-hat, we can rewrite this, for any x_0 (!)

$$\hat{\chi}^2 = \underbrace{(m - h_0 - Hx_0)^T V^{-1} (m - h_0 - Hx_0)}_{X^2 \text{ of residuals for } x=x_0} - \underbrace{(\hat{x} - x_0)^T C^{-1} (\hat{x} - x_0)}_{X^2 \text{ of difference between } x \text{ and } x_0}$$

minimum chi-square of the LSE

- now apply this for $x_0 = x^{true}$

dimension N dimension M

$$\hat{\chi}^2 \;=\; (m - h_0 - Hx^{\rm true})^T V^{-1} (m - h_0 - Hx^{\rm true}) \;-\; (\hat{x} - x^{\rm true})^T C^{-1} (\hat{x} - x^{\rm true})$$

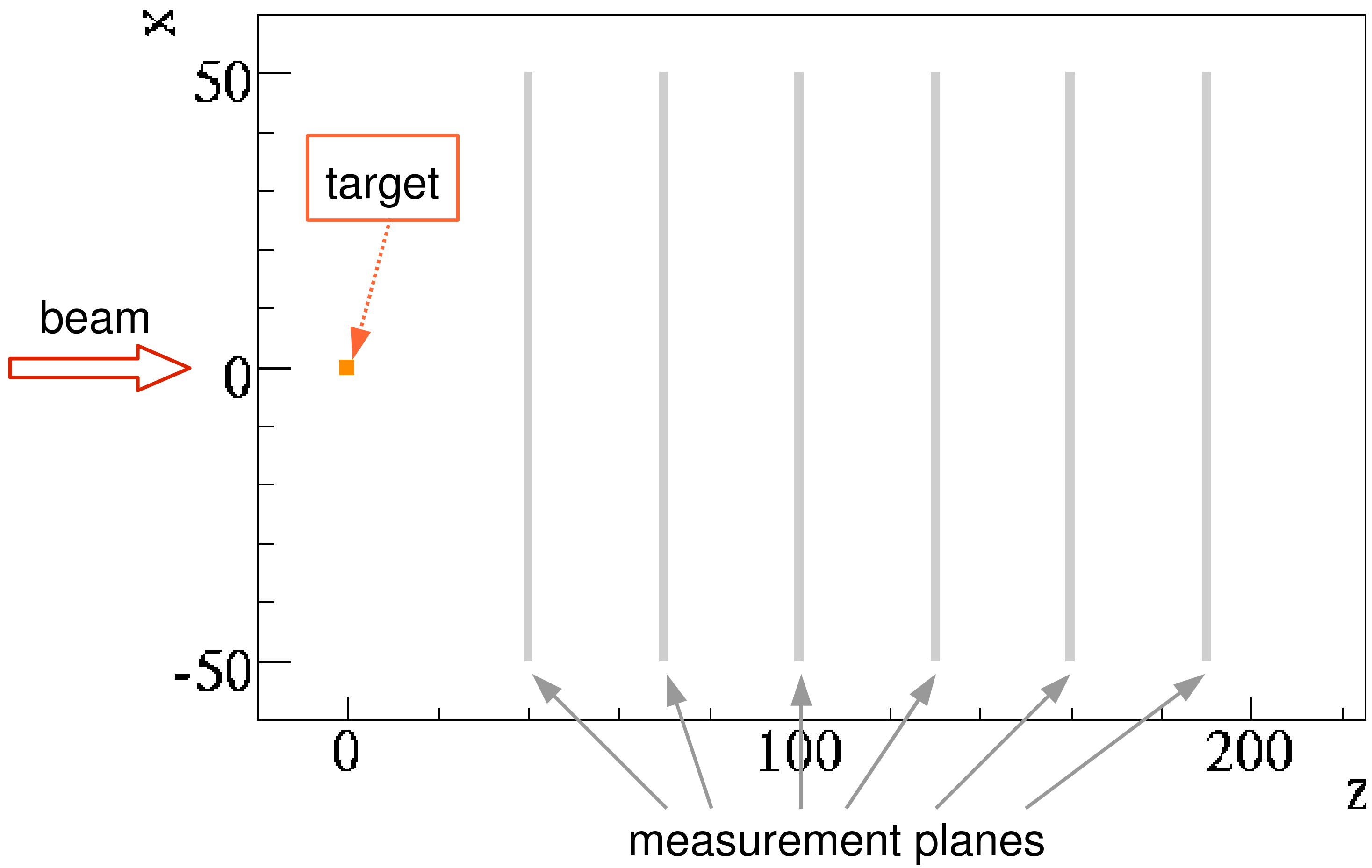
- the first term is the chi-square for x^{true}, distributed as $\mathcal{P}_{\chi^2}(z; N)$
- the second is also a real chi-square:
  - because any linear combination of Gaussian RVs is still Gaussian
  - if x has dimension M, this chi-square is distributed as $\mathcal{P}_{\chi^2}(z; M)$
- however, two terms are fully correlated: all random perturbations in the right term originate from those in the left term
- as a result (without proof) things cancel and we get $\mathcal{P}_{\chi^2}(\hat{\chi}^2; N - M)$
- its expectation value is thus $E(\hat{\chi}^2) \;=\; N - M$
- in words: “by fitting we have removed M degrees of freedom from X^2”

enough theory?

let's fit something

toy track fit: detector

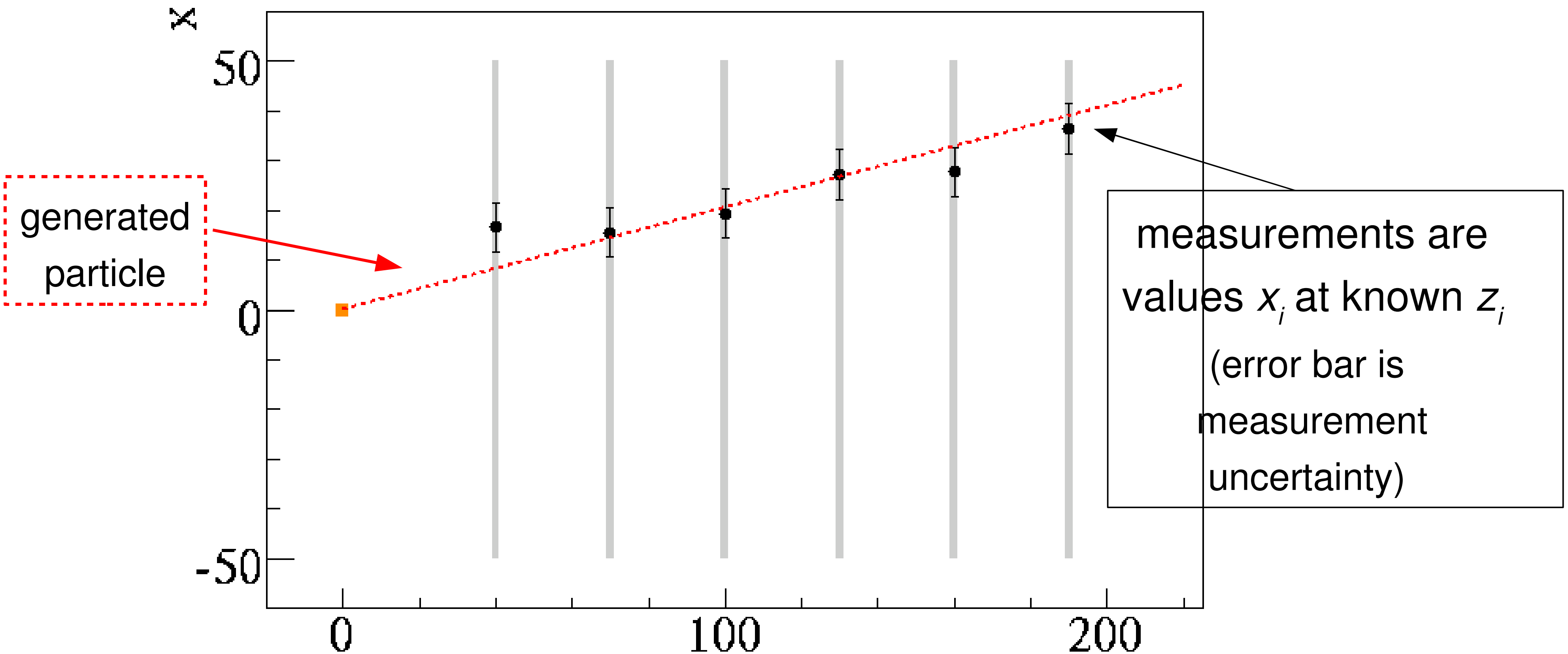
- detector: 6 measurement planes in the x-z plane



- each plane (at known z) measures the x-coordinate with uncertainty σ

toy track fit: generator

- generator: single particle, with straight trajectory (yes ... it's called a line)



- I am now going to show you a really complicated way of fitting a straight line through these point

- the good thing is that once you understand this, you can fit (almost) any kind of line, through (almost) any kind of data set, without MINUIT

toy track fit: track model

- let's choose this parameterization of the straight line

$$\begin{pmatrix} x \\ z \end{pmatrix} = \begin{pmatrix} x_0 \\ 0 \end{pmatrix} + \lambda \begin{pmatrix} t_0 \\ 1 \end{pmatrix}$$

$$\begin{array}{lcl} x_0 & : & x \text{ position at } z = 0 \\ t_0 & : & \text{slope at } z = 0 \end{array}$$

- so, vector of track parameters (sorry, once more a change in notation)

$$\boldsymbol{\alpha} = \begin{pmatrix} x_0 \\ t_0 \end{pmatrix}$$

- measurement model for hit in plane at z_i

$$h_i(\boldsymbol{\alpha}) = x_0 + z_i t_0 \qquad \boldsymbol{H}_i = \begin{pmatrix} 1 \\ z_i \end{pmatrix}$$

- this is a linear model: let's anyway use the expressions for the non-linear model. since the model is linear, we can choose a simple expansion point, e.g. (0,0)

chi-square derivatives

- evaluate the 1st and 2nd derivative of the chi-square at α=0

$$\frac{1}{2}\frac{\mathrm{d}\chi^2}{\mathrm{d}\alpha} = -\sum_{i=1}^{N} H_i^T \frac{1}{\sigma_i^2} x_i = -\frac{N}{\sigma^2}\begin{pmatrix} \langle x_i \rangle \\ \langle x_i z_i \rangle \end{pmatrix}$$

$$\frac{1}{2}\frac{\mathrm{d}^2\chi^2}{\mathrm{d}\alpha^2} = \frac{N}{\sigma^2}\begin{pmatrix} 1 & \langle z_i \rangle \\ \langle z_i \rangle & \langle z_i^2 \rangle \end{pmatrix}$$

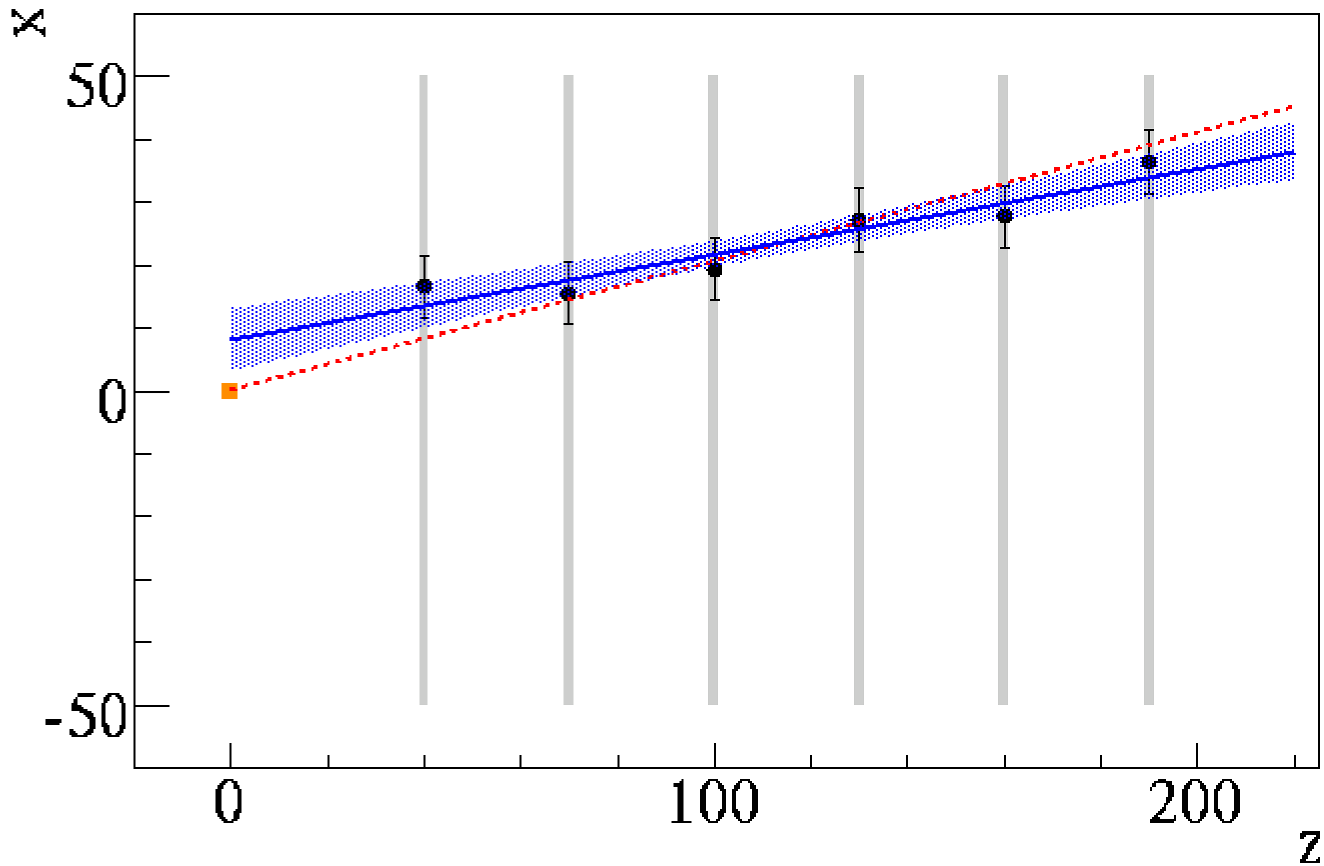
- I will not write down solution because this is where we would normally stop writing things down and use a computer. however, it is still instructive to look at the covariance matrix

$$\mathrm{var}\left(\alpha\right) = \frac{\sigma^2}{N}\frac{1}{\langle z_i^2 \rangle - \langle z_i \rangle^2}\begin{pmatrix} \langle z_i^2 \rangle & -\langle z_i \rangle \\ -\langle z_i \rangle & 1 \end{pmatrix}$$

- note
  - uncertainty on track parameters is proportional to hit uncertainty
  - its inversely proportional to sqrt(N)
  - uncertainty on the slope is inverse proportional to the spread in z

toy track fit: results of the LSE

- this is a result of the fit to the event we have seen before



- the blue line is the best fit 'trajectory'
- the blue band is the uncertainty on the x-coordinate for given z
- let me show you how that was calculated

'transporting' the track state

- we parameterized our track model at a fixed z-position z=0

$$\begin{pmatrix} x \\ z \end{pmatrix} = \begin{pmatrix} x_0 \\ 0 \end{pmatrix} + \lambda \begin{pmatrix} t_0 \\ 1 \end{pmatrix}$$

- we could have taken any other point. as a function of that point z, the track parameters are related to the parameters at z=0 by

$$\alpha(z) = \begin{pmatrix} x_0 \\ t_0 \end{pmatrix} + \begin{pmatrix} z t_0 \\ 0 \end{pmatrix} = \boldsymbol{F} \alpha_0 \qquad \text{with} \qquad \boldsymbol{F} = \begin{pmatrix} 1 & z \\ 0 & 1 \end{pmatrix}$$

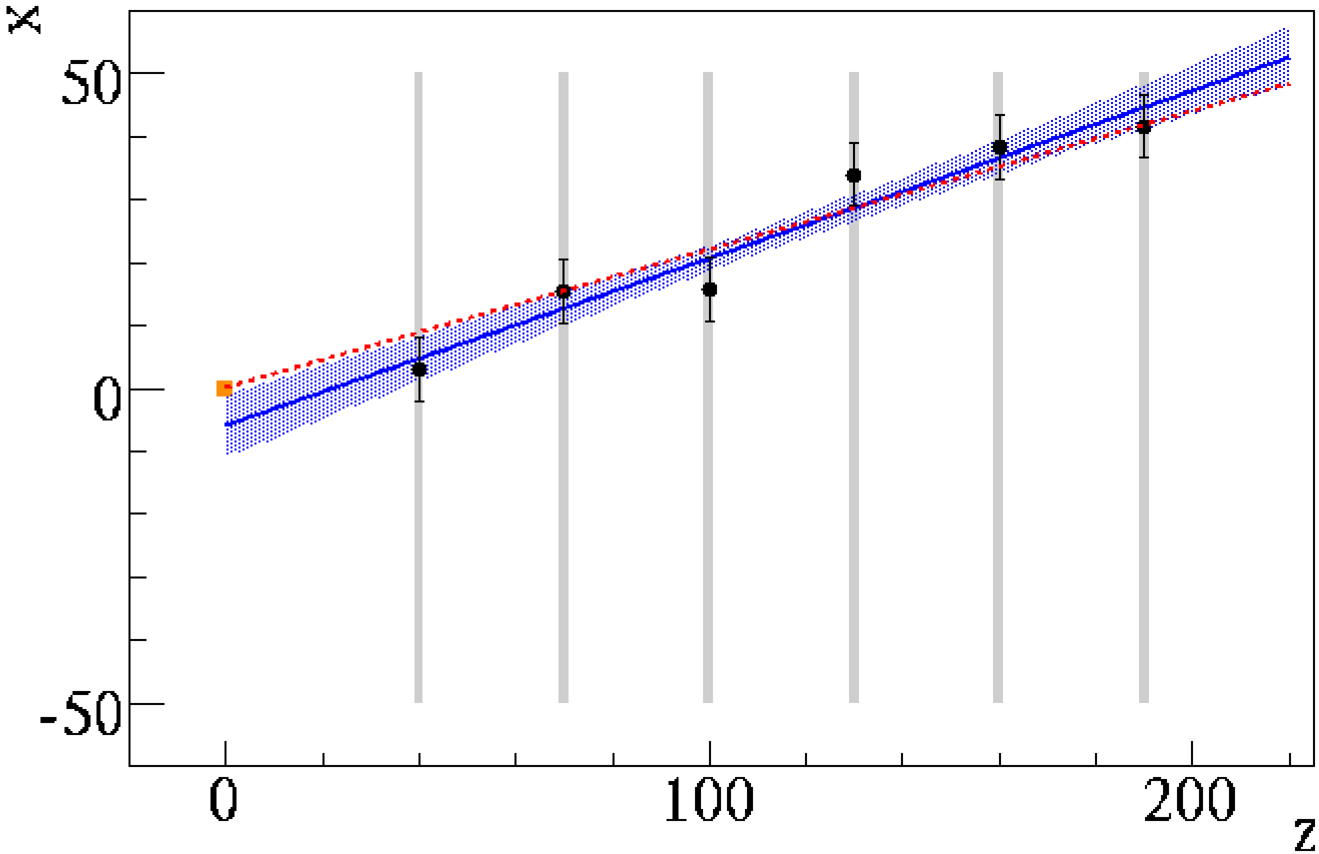
- we sometimes call the matrix F the 'transport matrix'
- the variance of the track parameters along z is then

$$\mathrm{var}\left(\alpha(z)\right) = \boldsymbol{F}\, \mathrm{var}\left(\alpha\right)\, \boldsymbol{F}^T$$

(just the familiar error propagation)

- for the error in x we find: $\mathrm{var}\left(x(z)\right) = \dfrac{\sigma^2}{N} \dfrac{\left\langle z_i^2 \right\rangle - 2\left\langle z_i \right\rangle z + z^2}{\left\langle z_i^2 \right\rangle - \left\langle z_i \right\rangle^2}$

- let's look at a few more events

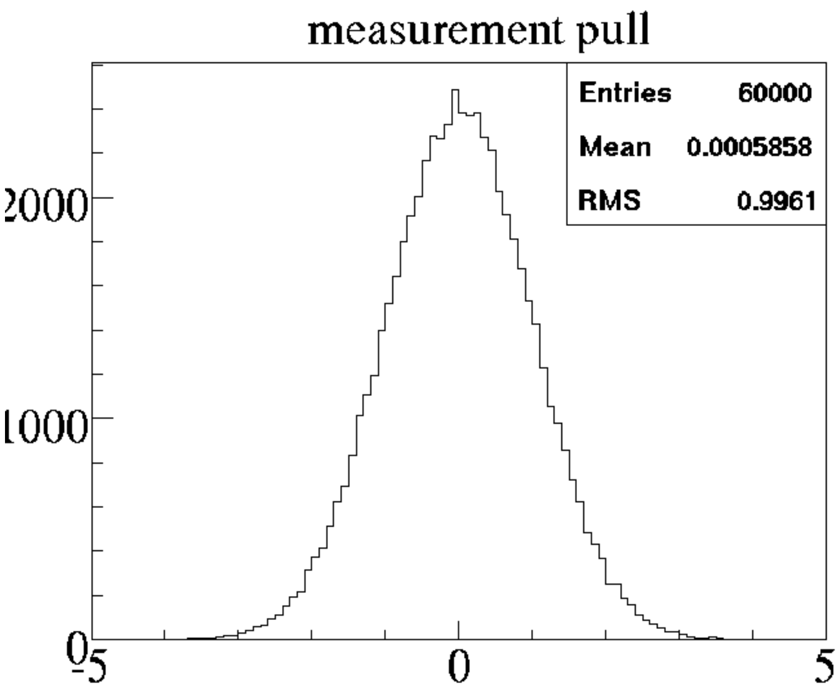


- note that the error band is always most narrow in the middle of the track

$$\mathrm{var}\left(x(z)\right) = \frac{\sigma^2}{N}\frac{\left\langle z_i^2\right\rangle - 2\left\langle z_i\right\rangle z + z^2}{\left\langle z_i^2\right\rangle - \left\langle z_i\right\rangle^2}$$

testing the fit

- first thing to check is that *input* to track fit makes sense
- look at *pull-distribution* of measurements

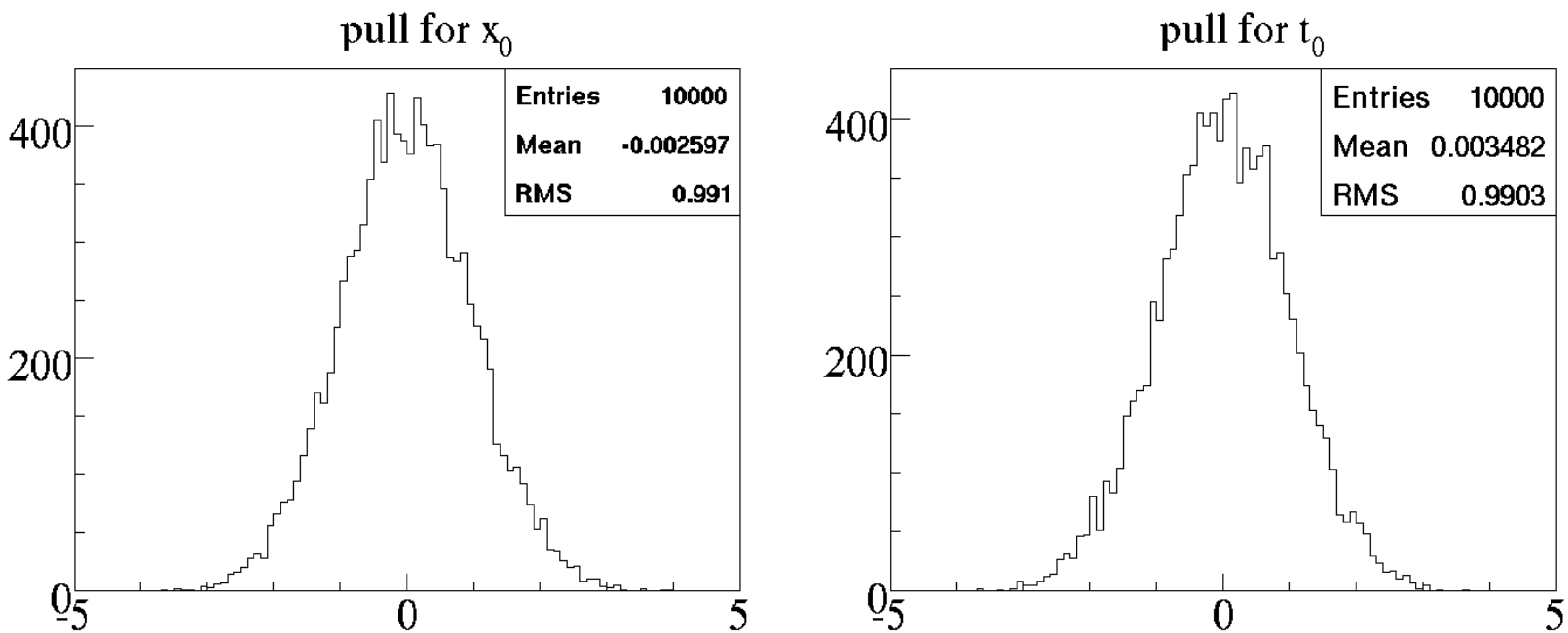


$$\text{measurement pull} \; = \; \frac{x_i - x_i^{\text{true}}}{\sigma_i}$$

- pull distribution: any distribution of $\dfrac{a - E(a)}{\sqrt{\mathrm{var}\,(a)}}$
- square of pull is chi-square of 1 degree-of-freedom
- pull gives more information because it is signed (e.g. can see model bias)

parameter pulls

- next we look at pulls of the *output* of the fit



- these kind of pulls still require knowledge of the 'true' trajectory
- we can also form pulls that do not require the 'truth', like the 'residual' pull

residual pull

- measurement residuals: $r_i = m_i - h_i(x)$
- covariance matrix for residuals (not entirely trivial. your homework)

$$R \equiv \mathrm{var}(r) = V - HCH^T$$

note minus sign!
(100% correlation)

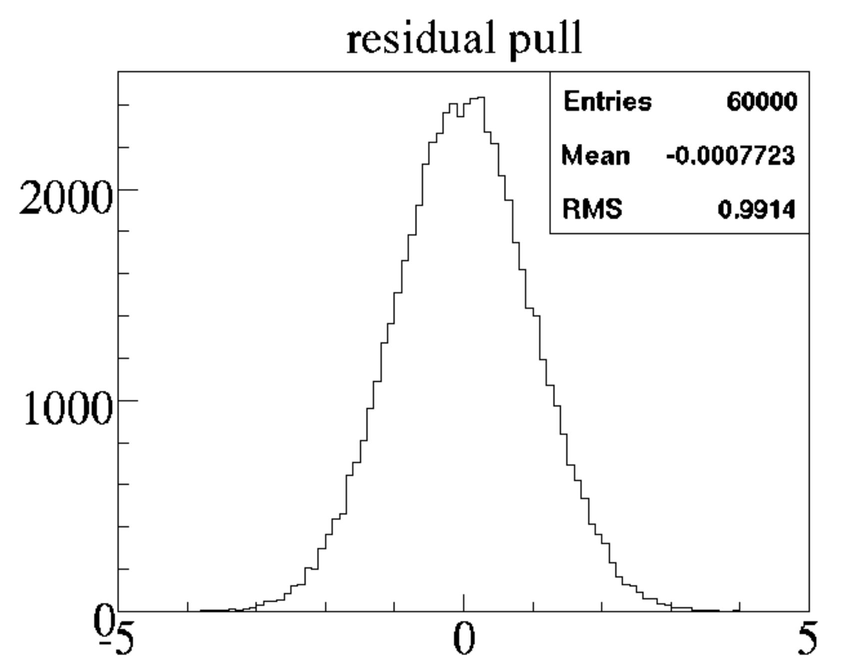
variance of m

variance of x

$$C = \left(H^T V^{-1} H\right)^{-1}$$

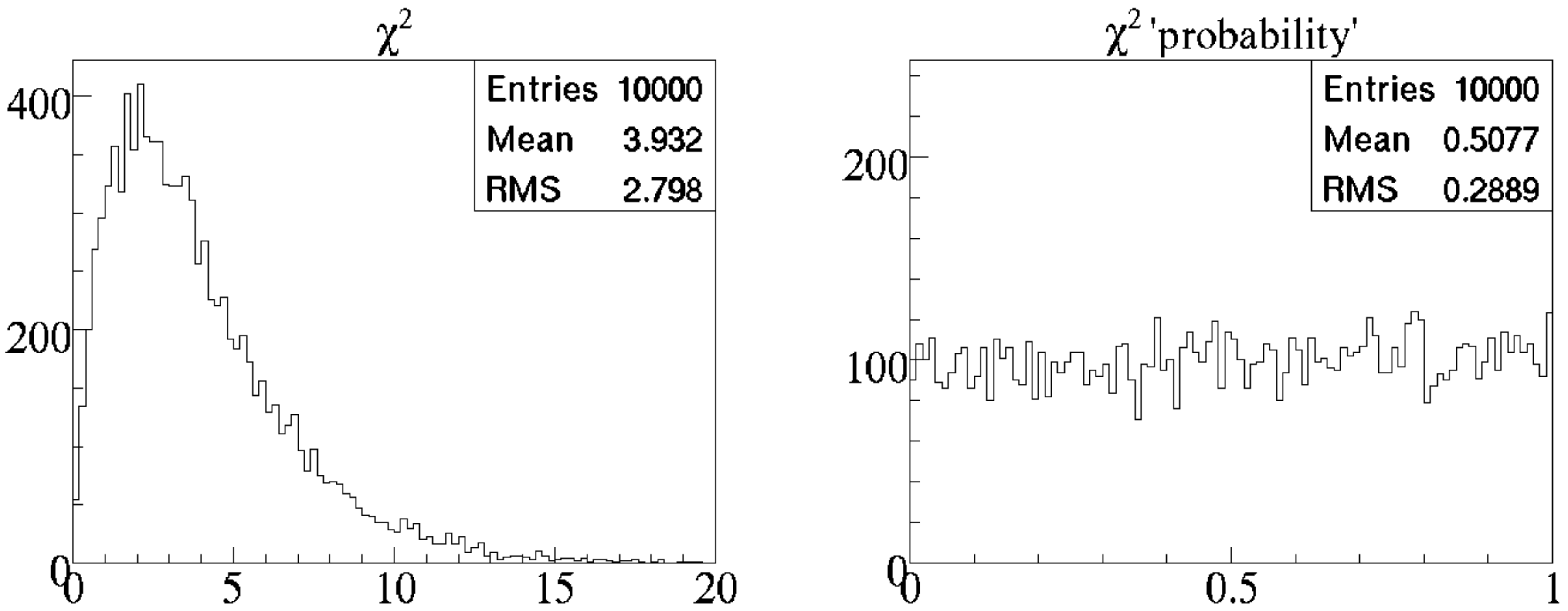
- matrix R is singular. its rank is N-M
  (it has M zero eigenvalues)

- using this expression we can now calculate pull of residuals



chi-square

- if the residual pull looks good, chi-square must look good, because it is basically just the sum of residuals-squared



$$\begin{aligned} E(\chi^2) &= N - M &= 4 \\ \sqrt{var(\chi^2)} &= \sqrt{2(N-M)} &= \sqrt{8} \end{aligned}$$

applying constraints sequentially

- suppose we split the set of constraints in two (m_1 and m_2 still vectors)

$$\chi^2 = (m_1 - h_1(x))^T V_1^{-1} (m_1 - h_1(x)) + (m_2 - h_2(x))^T V_2^{-1} (m_2 - h_2(x))$$

- let's first minimize the first term

$$\chi_1^2 = (m_1 - h_1(x))^T V_1^{-1} (m_1 - h_1(x))$$

as you know, assuming a linear model, the solution is

$$\begin{aligned} C_1 &= \left(H_1^T V_1^{-1} H_1\right)^{-1} \\ x_1 &= x_0 + C_1 H_1^T V_1^{-1} (m_1 - h_1(x_0)) \end{aligned}$$

where x_0 was an arbitrary starting point (you can choose $x_0=0$)

- how can we reuse this result to find the minimum chi-square solution to the total set of constraints?

applying constraints sequentially (II)

- well, use the solution x1 as a constraint to form this chi-square

$$
\begin{aligned}
\chi^{2'} &= (x_1 - x)^T C_1^{-1} (x_1 - x) \ + \ (m_2 - h_2(x))^T V_2^{-1} (m_2 - h_2(x)) \\
&= \begin{pmatrix} x_1 - x \\ m_2 - h_2(x) \end{pmatrix}^T \begin{pmatrix} C_1^{-1} & 0 \\ 0 & V_2^{-1} \end{pmatrix} \begin{pmatrix} x_1 - x \\ m_2 - h_2(x) \end{pmatrix}
\end{aligned}
$$

- the derivative matrix is now $H = \begin{pmatrix} 1 \\ H_2 \end{pmatrix}$

  and the solution becomes

$$
\begin{aligned}
C &= \left(C_1^{-1} + H_2^T V_2^{-1} H_2 \right)^{-1} \\
x &= x'_0 \ + \ C \ \left(C_1^{-1}(x_1 - x'_0) + H_2^T V_2^{-1} (m_2 - h_2(x'_0)) \right)
\end{aligned}
$$

- after substituting x_1 and C_1, the result is equal to the result we would have obtained had we minimized the original chi-square
- conclusion: for linear models we can apply constraints sequentially
- caveat: the intermediate problems should not be under-constrained

gain matrix and weighted mean formalisms

- note that in the expression of the final solution the expansion point is still arbitrary

$$x = x_0' + C \left(C_1^{-1}(x_1 - x_0') + H_2^T V_2^{-1}(m_2 - h_2(x_0')) \right)$$

- an obvious choice is to use x1 as the expansion point

$$x = x_1 + C\, H_2^T V_2^{-1}(m_2 - h_2(x_1))$$

gain matrix formalism

- we can also write this expression as

$$x = C \left(C_1^{-1} x_1 + H_2^T V_2^{-1}(m_2 \underbrace{- h_2(x_1) + H_2 x_1)}_{\text{this is constant for linear models}} \right)$$

weighted means formalism

- two understand the subtle difference, we need to talk about time consumption of calculations

matrix inversions

- let's look more carefully at this

$$x = x_1 + C\, H_2^T V_2^{-1} (m_2 - h_2(x_1))$$

diagonal matrix, trivial to invert

$$C = (C_1^{-1} + H_2^T V_2^{-1} H_2)^{-1}$$

not diagonal matrices

expensive to invert if dimension is large

- so, to calculate C we need two 'expensive' matrix inversions
- it turns out we can do something smarter if the dimension of m_2 is small

Kalman gain matrix

- make use of following matrix identity (homework: verify!)

$$(C^{-1} + H^T V^{-1} H)\, H^T\, V^{-1} = C\, H^T\, (V + HCH^T)^{-1}$$

to rewrite the gain matrix solution as

$$x = x_1 + K(m_2 - h_2(x_1))$$

$$K = C_1 H_2^T (V_2 + H_2 C_1 H_2^T)^{-1}$$

- single matrix inversion now concerns matrix with dimension of m_2.
- if this dimension is small (e.g. if we add a single data point), the calculation is very fast
- this is the essential ingredient to the Kalman Filter

Kalman Filter

- developed to determine evolution of state vector (called 'trajectory') of dynamical system in time
- with each new measurement, knowledge about trajectory improves
- example: following a rocket on a radar screen
  - with a global fit, you would need to refit complete trajectory with each new measurement
  - kalman fit much faster, which is important in this case ...
- main benefits of Kalman filter in track and vertex fitting
  - local treatment of multiple scattering (tomorrow morning, part 4)
  - pattern recoginition: possibility to decide, based on current knowledge of track, if a new hit is really part of it (tomorrow morning, part 5)

application: a weighted mean

- suppose m_2 is just another estimate of x with covariance C_2
- in the weighted mean formalism the combination would give

$$x \;=\; \left(C_1^{-1} + C_2^{-1}\right)^{-1} \;\left(C_1^{-1} x_1 \;+\; C_2^{-1} x_2\right)$$

- where the gain matrix expression looks like

$$x \;=\; x_1 + C_1 \;\left(C_1 + C_2\right)^{-1} \left(x_2 - x_1\right)$$

- it is easy to verify that these two expressions lead to identical results
- if the dimension of x is larger than 1, the 2[nd] expression is computationally much simpler, since it involves only a single matrix inversion

covariance matrix in the gain formalism

- there exist several expressions for the covariance matrix

$$C = (1 - KH_2)\, C_1$$

$m^3 + O(m^2)$
fast

$$C = (1 - KH_2)\, C_1\, (1 - KH_2)^T + KV_2K^T$$

$3m^3 + O(m^2)$
stable but slow

$$C = (1 - 2KH_2)\, C_1 + K\,(V_2 + H_2C_1H_2^T)\, K^T$$

$m^3 + O(m^2)$
stable and fast

- note that covariance matrix calculation does not require any extra matrix inversions
- expressions above differ in computation speed and in sensitivity to small errors in the gain matrix K
- such small errors can occur because of finite floating point precision affecting the matrix inversion
- see also NIM.A552:566-575,2005

global versus progressive fit

- global fit: apply constraints simultaneously
  - calculate chi-square derivatives once, summing over all constraints
  - requires single calculation of solution to system with M equations
  - hard to include 'process noise' (multiple scattering)

- progressive fit (Kalman filter, recursive fit): apply constraints sequentially
  - update solution after each constraint
  - requires M calculations of solution to system with single equation (provided the constraint is 1dimensional)
  - constraints must be uncorrelated
  - easy to include process noise

Kalman filter for track fitting

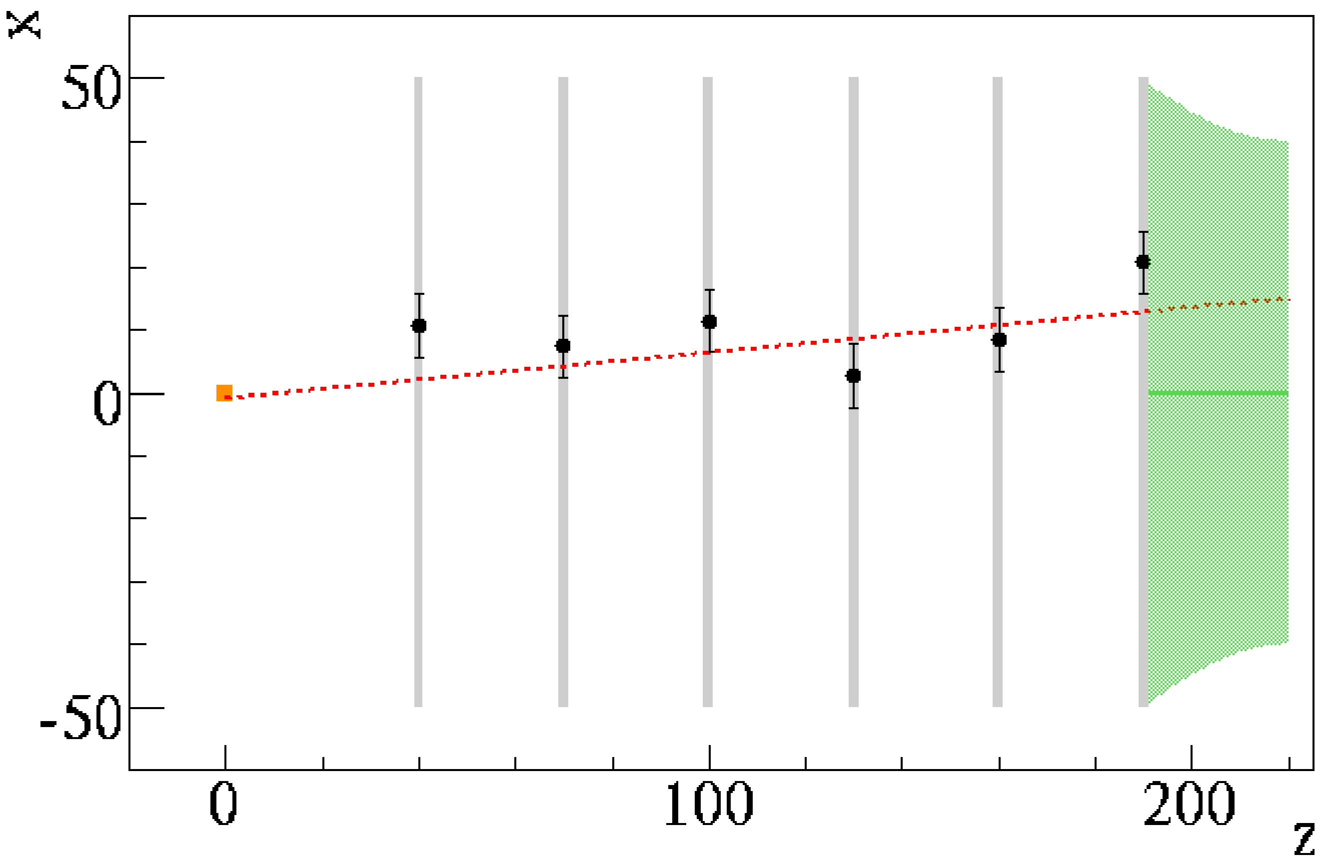
- your main reference: Fruhwirth, NIM-A262, pp 444, 1987
- the Kalman filter is based on the gain matrix formalism
  - start with an estimate x_0, C_0 of the track parameters. C_0 must be large compared to final expected variance
  - add measurements one-by-one updating track parameters at each step

$$
\begin{aligned}
K_k &= C_{k-1} H_k^T (V_k + H_k C_{k-1} H_k^T)^{-1} \\
x_k &= x_{k-1} + K_k (m_k - h_k(x_{k-1})) \\
C_k &= (1 - K_k H_k) \, C_{k-1}
\end{aligned}
$$

- for linear problems, the result is identical to the global fit
- the real advantage of the KF becomes clear only when we discuss multiple scattering

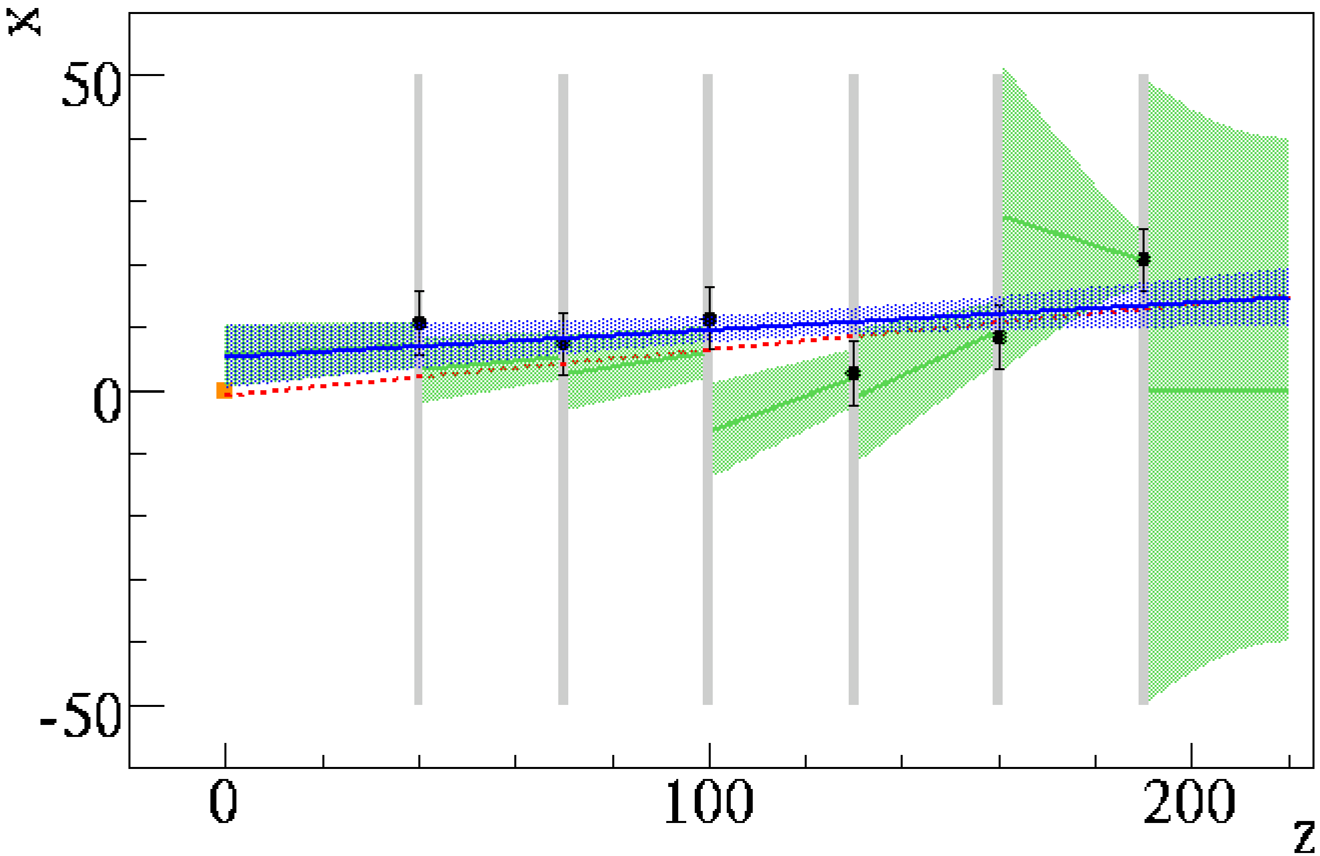
Kalman filter for our toy simulation

- this animation shows how the track state changes when hits are added



Kalman filter for our toy simulation

- this animation shows how the track state changes when hits are added



- the result of the global fit is shown in blue

measurement constraints

- up till now, contributions to chi-square looked like

$$\Delta\chi^2 \;=\; (m_i - h_i(x))^T \, V_i^{-1} \, (m_i - h_i(x))$$

- I'll call this type of contribution a ***measurement constraint***
- it can be more generally written as

$$\Delta\chi^2 \;=\; g_i(x)^T \, V_i^{-1} \, g_i(x)$$

- with the LSE we solve the over-constrained set of equations

$$\forall_i \; g_i(x) \;=\; 0$$

  using the assigned inverse variance as a weight

- but now suppose that we have a relation between the parameters *x* that we want to be *exactly* satisfied?

exact constraints

- ***exact constraint*** expresses exact relation between the parameters *x*
  - for example: suppose x is a 4-vector with 4x4 covariance matrix and we want it to have exactly length m_B
- sometimes it is possible to simply eliminate 1 of the parameters
- more generic solution: add an extra term to the chi-square

$$\Delta\chi^2 = \lambda_j \, g_j(x)$$

- the parameter λ is a lagrange multiplier
- we now minimize the total chi-square wrt to λ and *x* simultaneously
- taking the derivative to lambda, you see how this imposes the constraint

$$0 = \frac{\mathrm{d}\chi^2}{\mathrm{d}\lambda_j} = g_j(x)$$

exact constraints in the progressive fit

- in the progressive fit, we can eliminate the lagrange multiplier

$$\chi^2_k = (x - x_{k-1})^T {C_{k-1}}^{-1} (x - x_{k-1}) + 2\lambda_k^T g_k(x)$$

$$\Downarrow$$

linearize around x_{k-1}: $g_k(x) = g_k(x_{k-1}) + G_k (x - x_{k-1})$

$$0 = \frac{1}{2}\frac{\mathrm{d}\chi^2}{\mathrm{d}x} = {C_{k-1}}^{-1}(x - x_{k-1}) + G_k^T \lambda_k$$

$$0 = \frac{\mathrm{d}\chi^2}{\mathrm{d}\lambda} = g_k(x_{k-1}) + G_k (x - x_{k-1})$$

$$\Downarrow$$

solve, eliminate λ

$$x_k = x_{k-1} - K_k \, g_k(x_{k-1})$$

$$C_k = (1 - K_k G_k) \, C_{k-1} \, (1 - K_k G_k)^T$$

$$K_k = C_{k-1} G_k^T \left(G_k C_{k-1} G_k^T \right)^{-1}$$

- not surprising: expressions are identical to those for a measurement constraint with V=0!
- so, it is easy to include exact constraints in a progressive fit