Stats Review
Lecture 20

What we’ve gone over

- What is probability
- Common probability distributions
  - Normal / Gaussian, Poisson
- Interpretation of a measurement
  - A data point is a distribution
- Counting statistics, Poisson process
- Least squares / chi2 fitting to find a model’s best fit to data
- Chi2 statistics to find parameter errors

Probability

- Basic properties
  - $\sum_{\text{all outcomes}}$ Probability = 1
  - The probability of something must be between 0 and 1!
- Distributions
  - PMF - probability mass function, the probability of discrete outcomes
  - PDF - probability density function, the probability of continuous outcomes

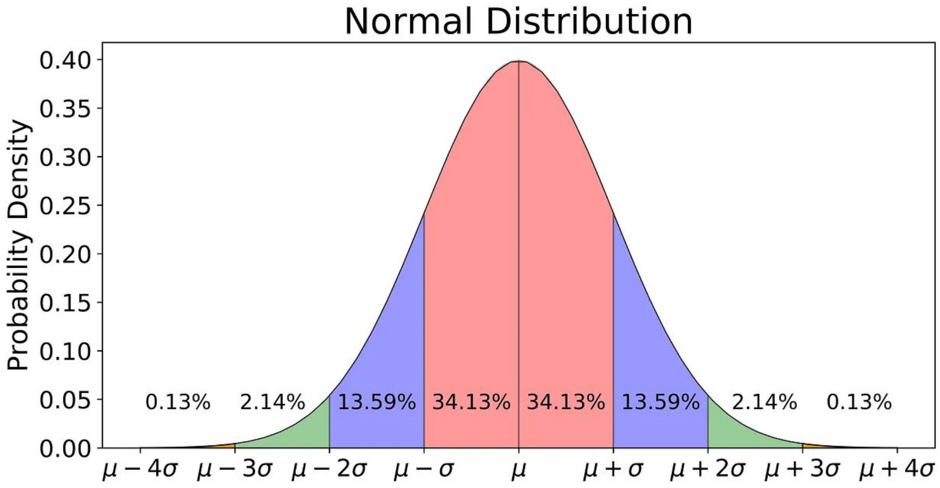
$$P(-2 \leq x \leq 2) = \int_{-2}^{2} p(x)\, dx$$

Normal distribution

One of the most common probability distributions is a Normal distribution

Sometimes also known as a Gaussian

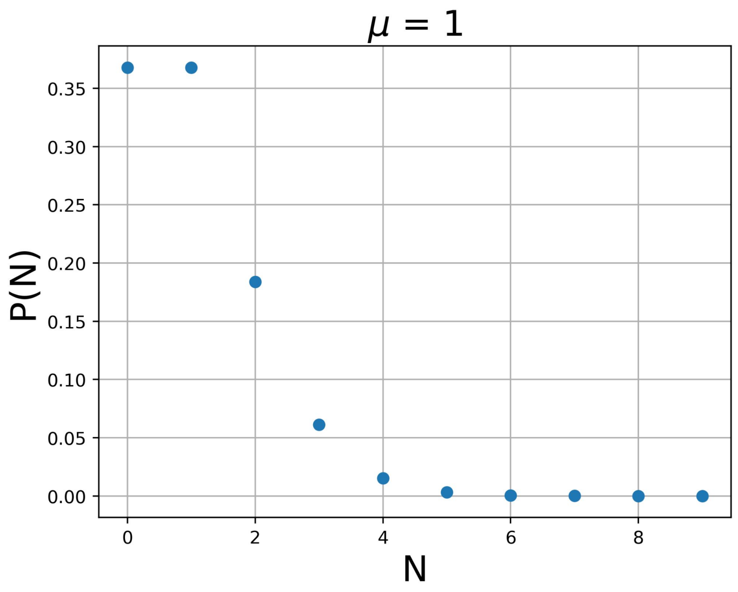
$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$



Poisson Distribution

The probability distribution of the number arrivals (or counts) in a given amount of time is given by the Poisson distribution

A PMF



$$P(N;\mu) = \frac{(\mu)^N e^{-\mu}}{N!}$$

N = number of counts or arrivals

μ = average counts

μ = average rate * time

! means factorial

How to interpret observations/measurements

Measurements of non-discrete parameters are never exact

There will always be some error to a measurement

There are 2 types of errors -

- **Random**: makes fluctuations of measurements above and below the actual value
    - These should cancel out over many observations
    - Creates a spread in your measurements that should average to actual value
- Systematic: creates an offset in one direction relative to the actual value
    - These do not average away over many observations
    - Creates a bias in your measurements

We’re going to focus on random errors

Taking a measurement

A measurement is never exact, there is always some error to it

This depends on what you’re measuring, but these errors often follow a Gaussian

Taking a measurement

A measurement is never exact, there is always some error to it

This depends on what you’re measuring, but these errors often follow a Gaussian

Say for example you measure the energy of a photon to be 9.5 keV, but your device has a known Gaussian 1-sigma error of 1 keV

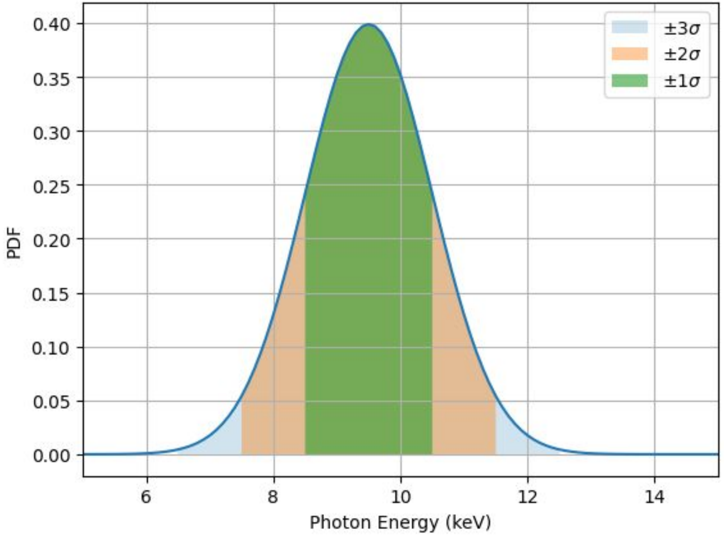
Taking a measurement

A measurement is never exact, there is always some error to it

This depends on what you're measuring, but these errors often follow a Gaussian

Say for example you measure the energy of a photon to be 9.5 keV, but your device has a known Gaussian 1-sigma error of 1 keV

This would be the PDF of the actual photon energy



Taking a measurement - by counting

One type of measurement is counting

- While the number of counts are exact
- The average rate is not exact
- Your measurement of the average rate is again a distribution

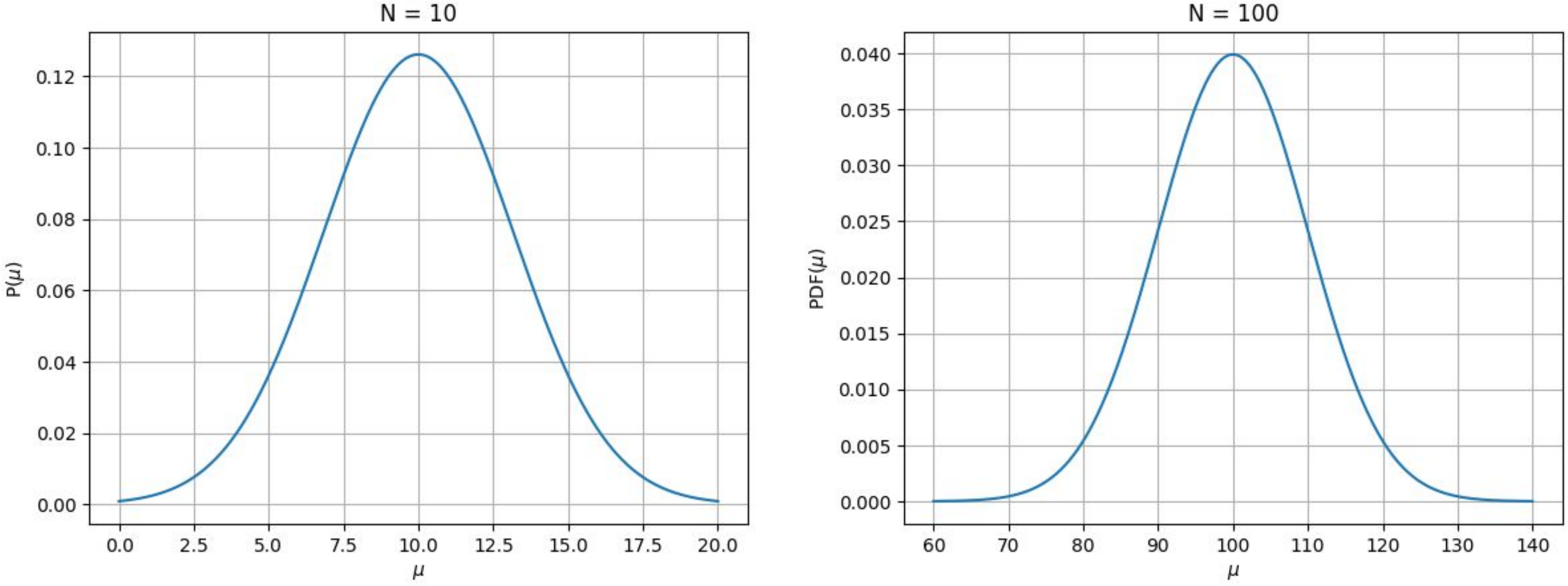
If the number of counts is high enough the Poisson distribution can be approximated as a Gaussian

- The variance of a Poisson is equal to the expected counts (avg rate x time)
- Std deviation = sqrt(counts)

Say we measure 10 counts or 100 counts

Our error PDF for the “expected counts” is a Gaussian with sigma = sqrt(N)

- “Expected counts” = true rate x exposure

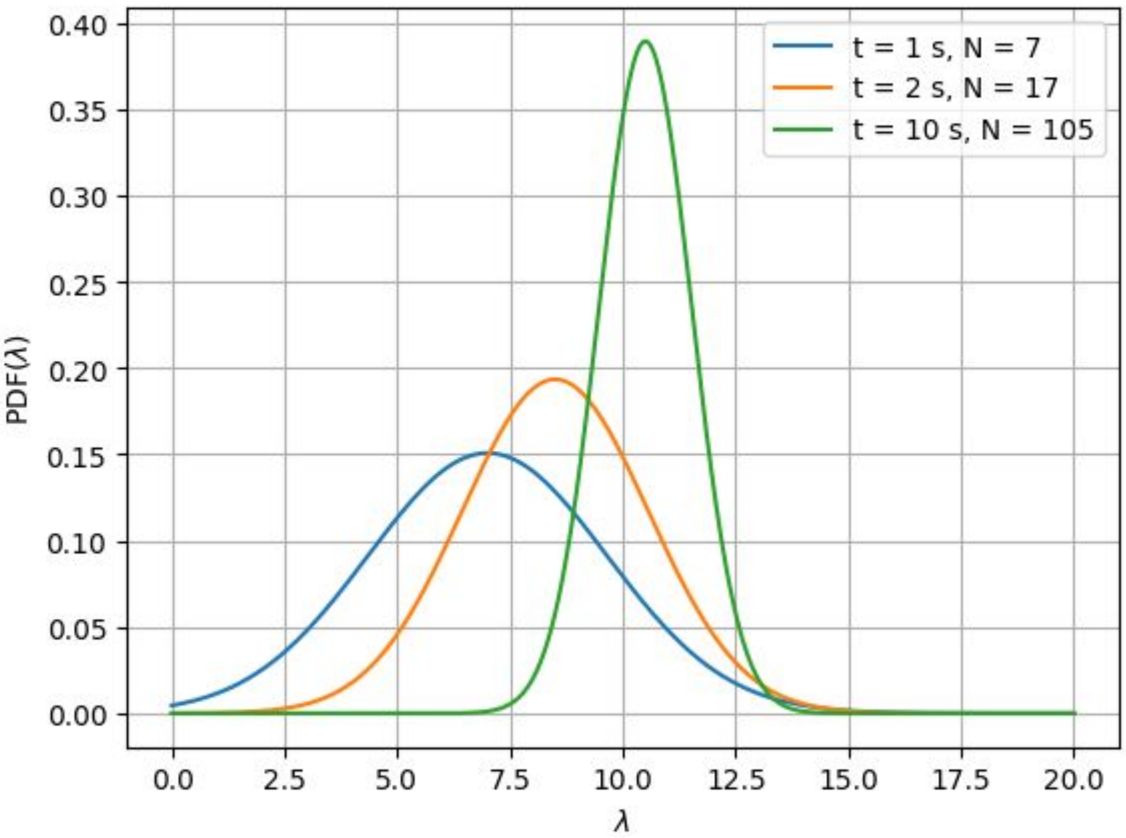


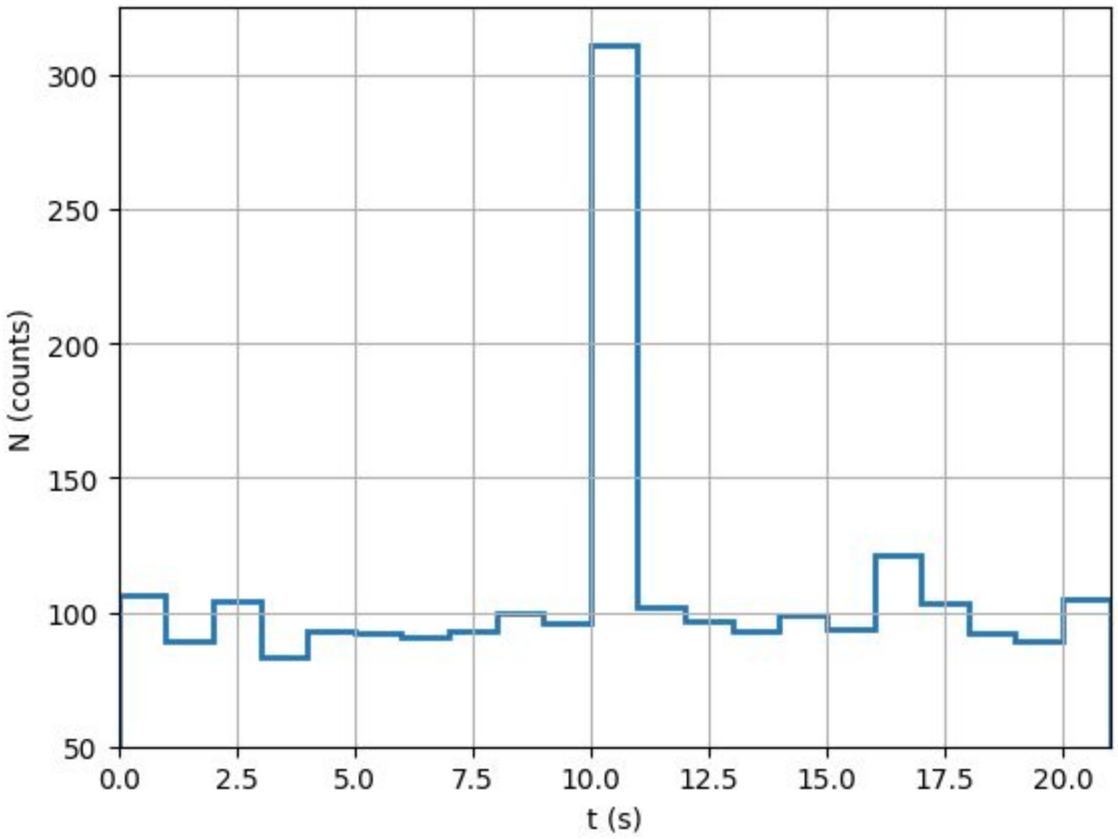
Say true rate = 10 / s

Here’s 3 outcomes at 3 different exposures

The error PDF on the rate is a Gaussian

Sigma = sqrt(N) / t




300
250
200
150
100
50
N (counts)
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
t (s)

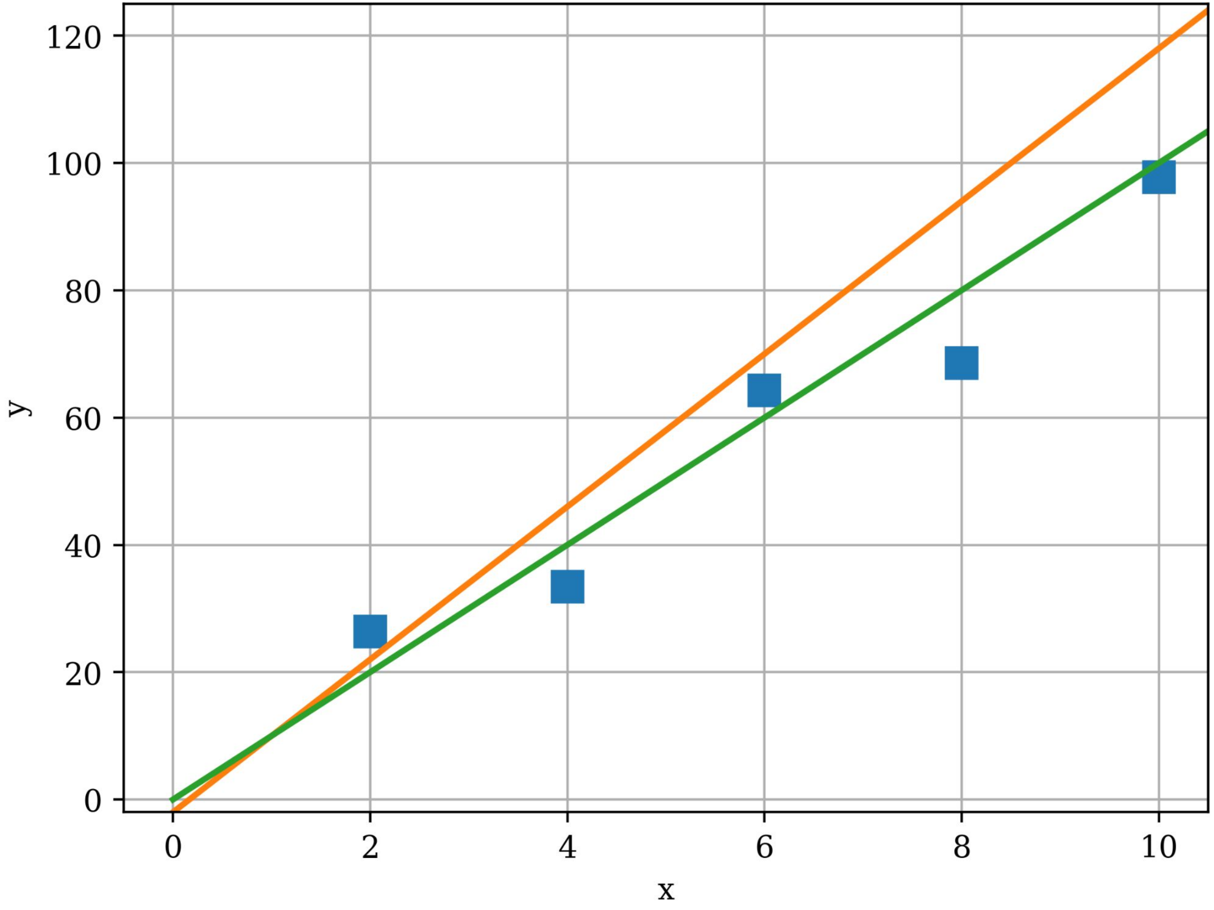
- These are all single observations
- Typically we analyze a set of observations (a set of data points)

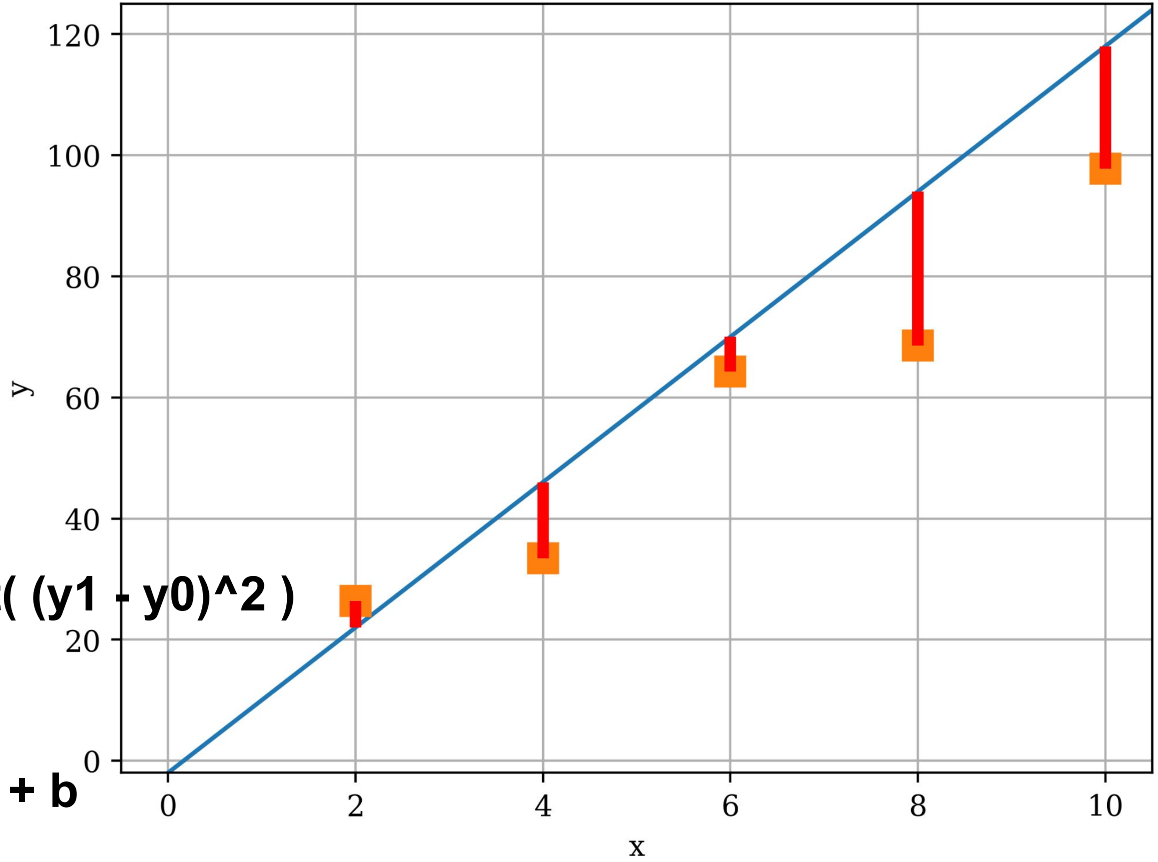
- To analyze a data set we apply a physically motivated model
- Model - a function that estimates what we observe as a function of parameters
    - Ex: describing the speed of a car with constant acceleration as a function of time
    - $v(t) = a*t + v(t=0)$
- To extract information from the data we compare the data to the model and find what parameters of the model “best” describes/fits the data

How can we choose between 2 lines

Which describes the data better?

How about the one closest to the data points?

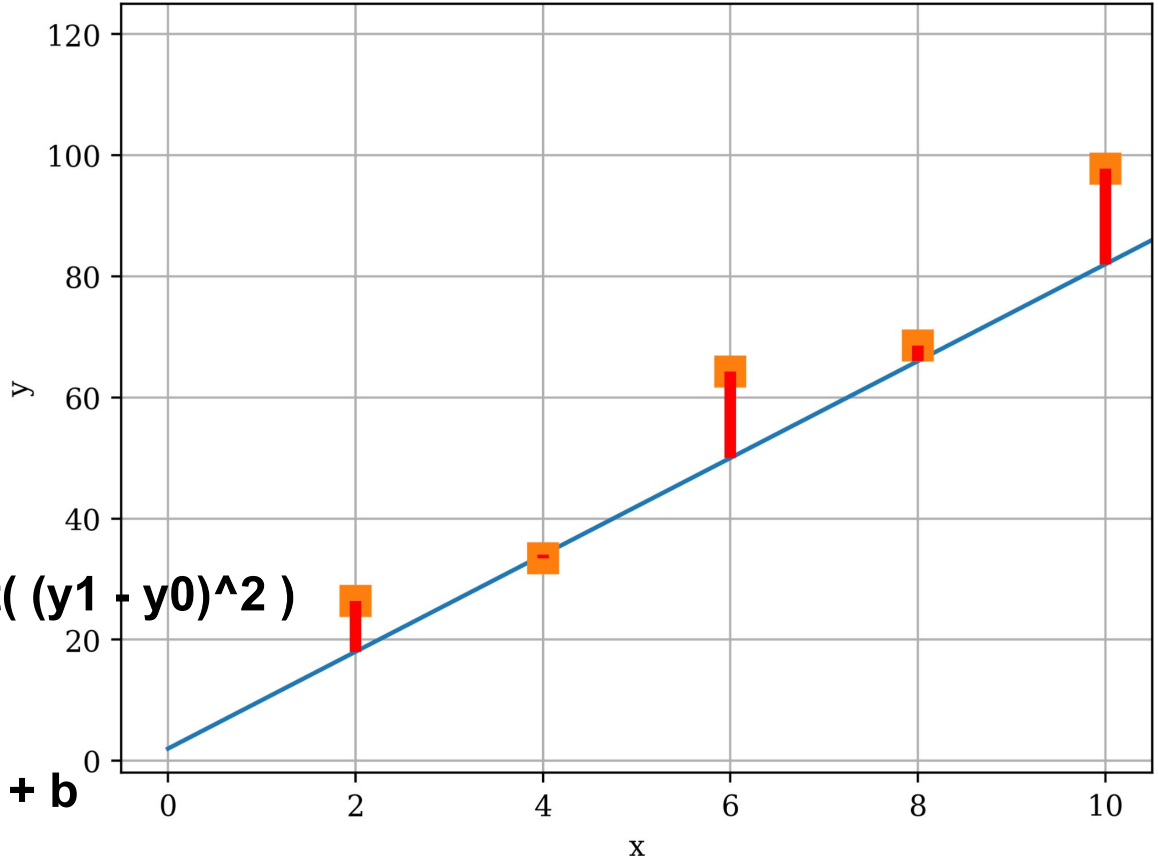




distance = sqrt((y1 - y0)^2)

y0 = data

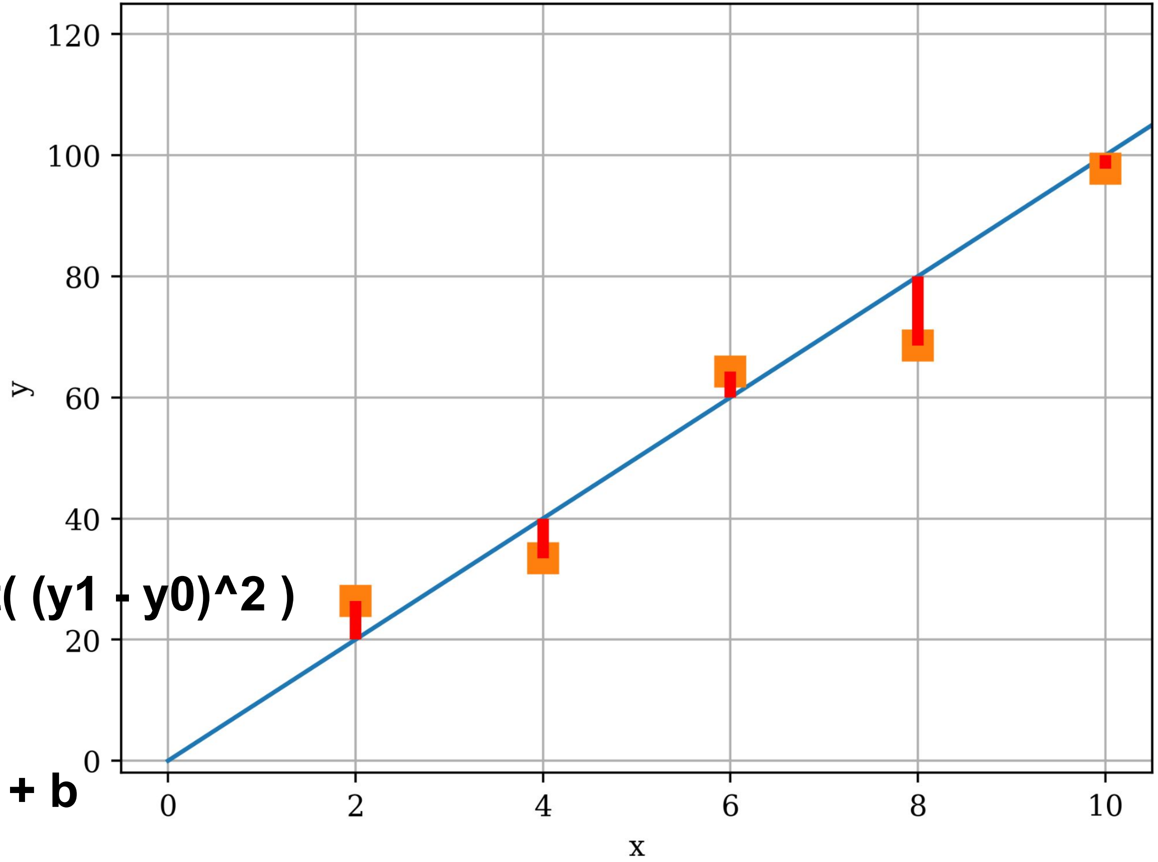
y1 = line = m*x + b



distance = sqrt((y1 - y0)^2)

y0 = data

y1 = line = m*x + b



distance = sqrt((y1 - y0)^2)

y0 = data

y1 = line = m*x + b

Least Squares

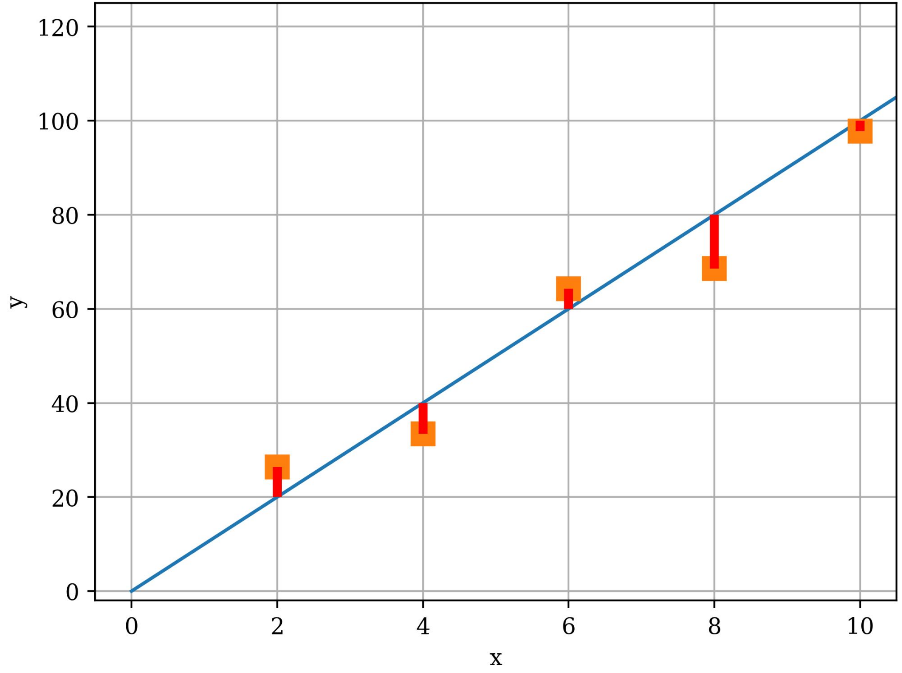
This is known as least squares fitting

a very common method

$S = \Sigma_i \,(\, y(x_i) - \; y_i \,)^2$

Then find the form of y(x) that minimizes S

Here y(x) = m*x + b



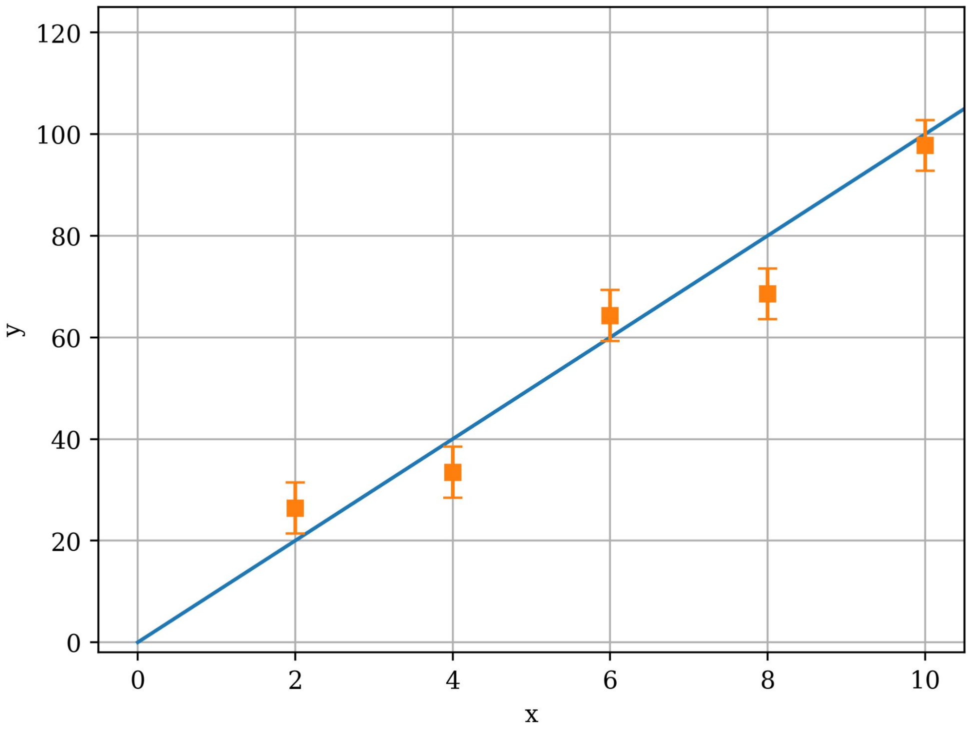
Let’s make this realistic

These data points are observations, they should have some error to them

Let’s say they have a Gaussian error

Here’s 1 sigma error bars

How can we take into account the error doing least squares?



Let's make this realistic

How can we take into account the error doing least squares?

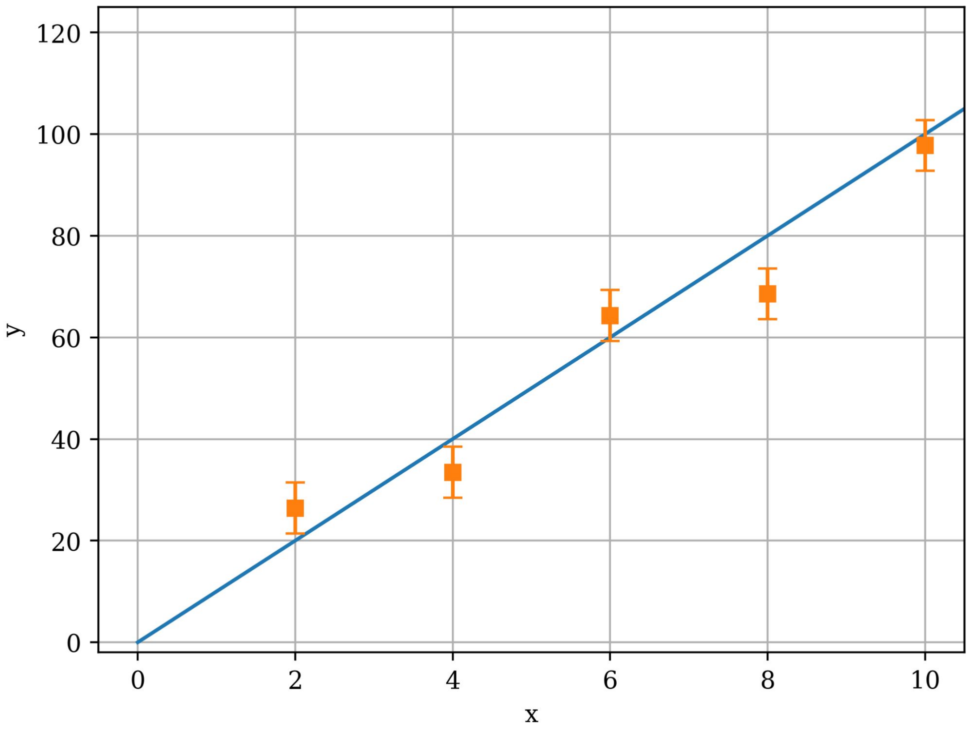
Let's divide the distance by σ

$s_i = ((y(x_i) - y_i) / \sigma_i)^2$

$chi2 = \Sigma_i((y(x_i) - y_i) / \sigma_i)^2$

It's now how many σ's is it from the line

y(x) is your model, m and b are your model parameters



Chi2 statistics to find model parameter errors

Run a bunch of experiments,

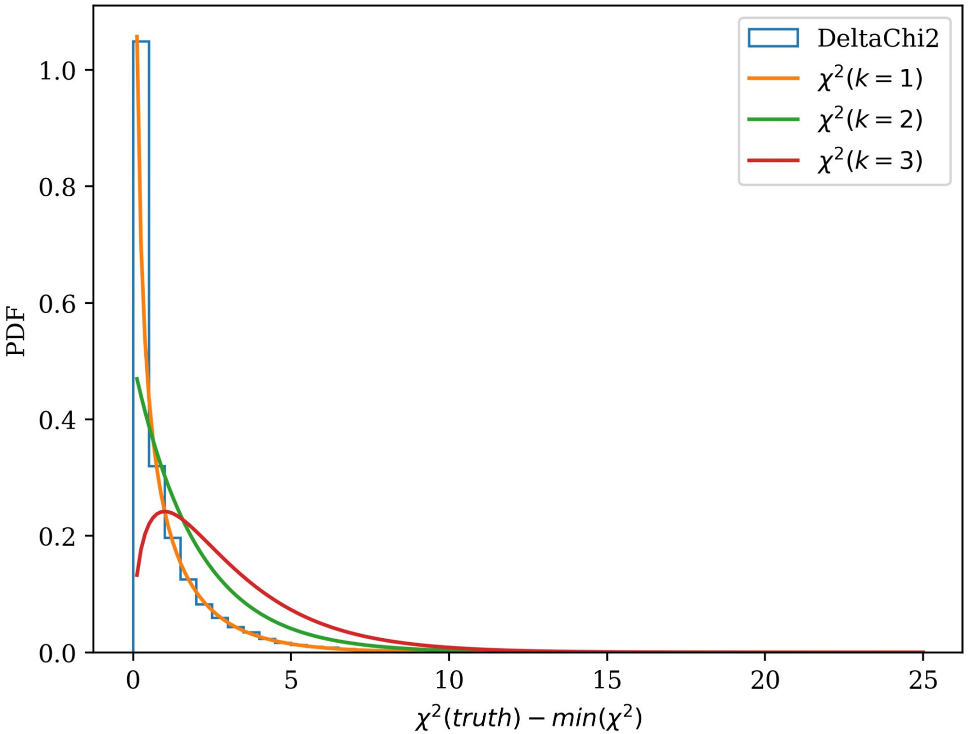
chi2(a = Truth) - min(chi2)

It looks like a chi2 dist with k = 1!

Why?

Both chi2's are calculated using the same data

the only remaining degree of freedom is the best fit a



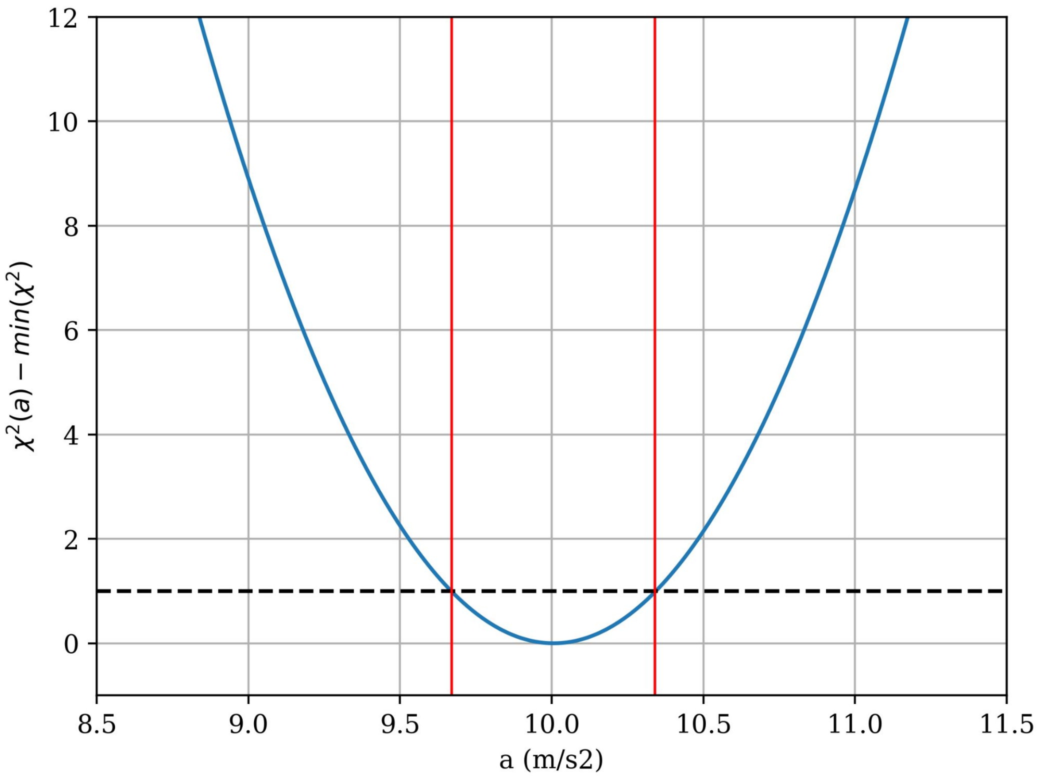
Then prob that a is in the region where

chi2 <= min(chi2) + 1

is also 68%

68% confidence that a is between 9.67 and 10.34 m/s2

a = 10 +/- 0.33 m/s2



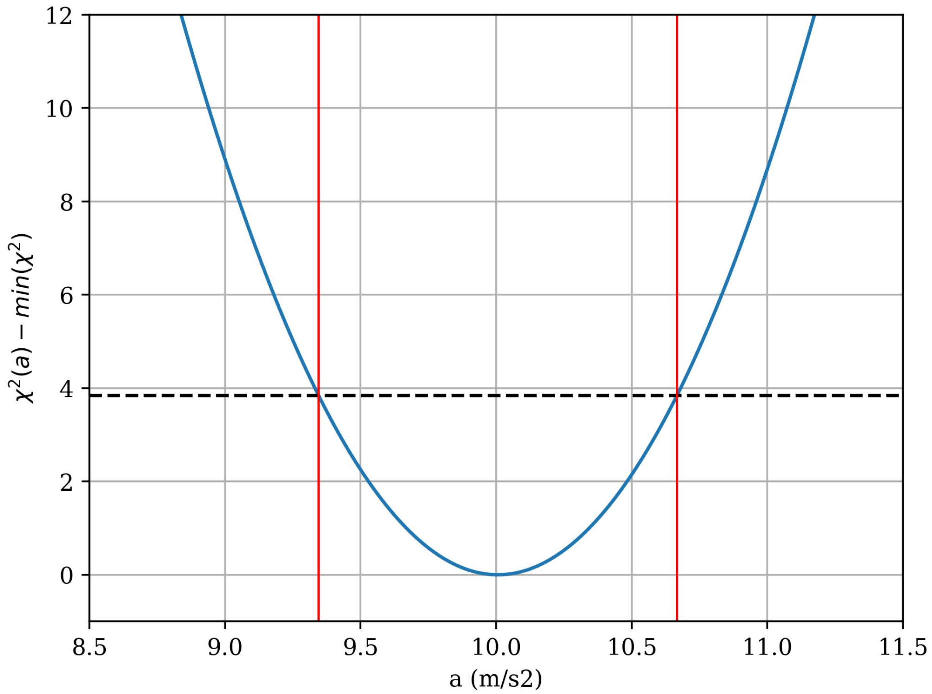
Different data than last lecture
Best-fit a ~ 10 m/s2

How about 95%?

stats.chi2.ppf(0.95, 1) = 3.84

95% confidence 9.34 - 10.67

a = 10 +/- 0.67 m/s2



What if we don’t know for sure that $v0 = 0$ m/s?

What if we don’t know for sure that v0 = 0 m/s?

Then we have another free parameter

v(t) = a*t + v0

What if we don’t know for sure that v0 = 0 m/s?

Then we have another free parameter

$v(t) = a*t + v0$

How do we find the best solution and errors now?

What if we don't know for sure that v0 = 0 m/s?

Then we have another free parameter

v(t) = a*t + v0

How do we find the best solution and errors now?

chi2(a, v0)

Now it's a 2D parameter space

To find the chi2 min we now have to scan over 2 dimensions

We can still brute force this, but this gets harder and harder to do in higher dimensions

To find the chi2 min we now have to scan over 2 dimensions

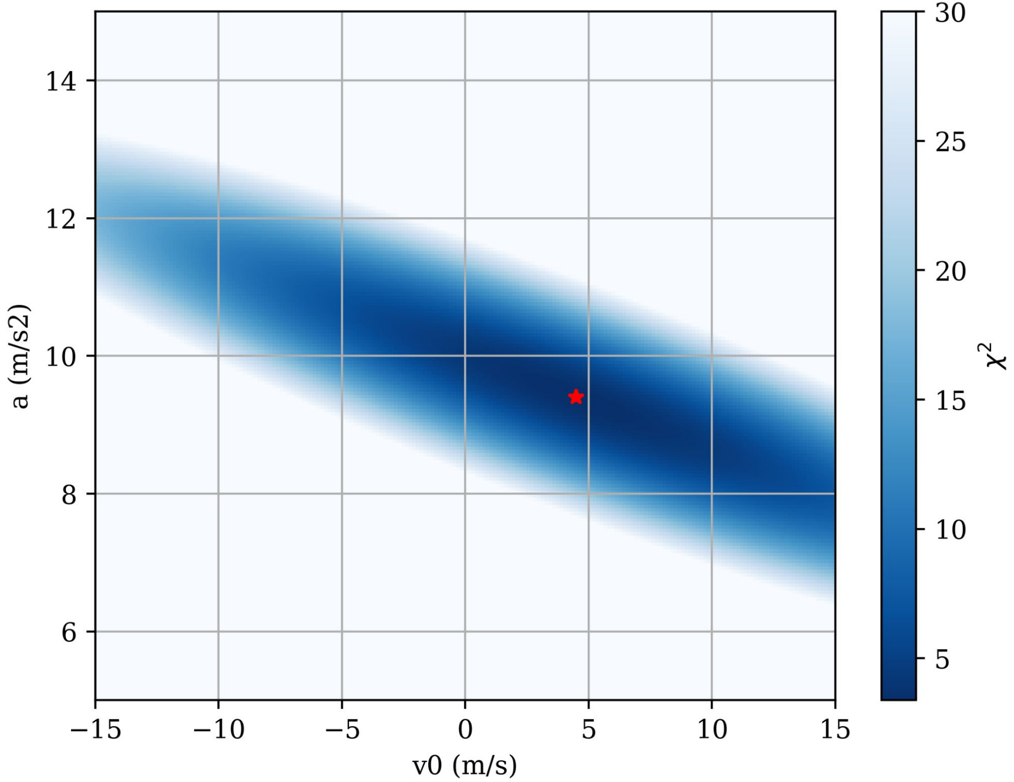
We can still brute force this, but this gets harder and harder to do in higher dimensions

- Need a 2D grid of a and v0 values
- Calculate chi2 at each grid point
- Find where the min chi2 is

To find the chi2 min we now have to scan over 2 dimensions

We can still brute force this, but this gets harder and harder to do in higher dimensions

- Need a 2D grid of a and v0 values
- Calculate chi2 at each grid point
- Find where the min chi2 is



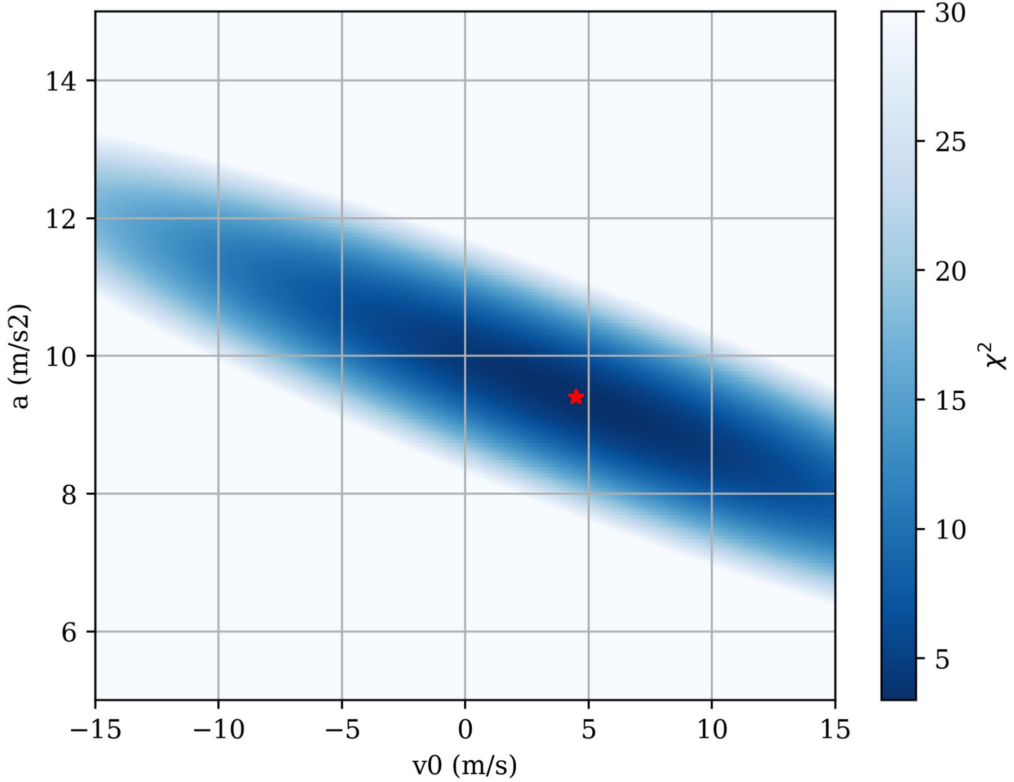
To find the chi2 min we now have to scan over 2 dimensions

We can still brute force this, but this gets harder and harder to do in higher dimensions

- Need a 2D grid of a and v0 values
- Calculate chi2 at each grid point
- Find where the min chi2 is

Best a = 9.4 m/s2

Best v0 = 4.5 m/s



What about the error PDF or confidence levels?

What about the error PDF or confidence levels?

We can still use Δchi2!

What about the error PDF or confidence levels?

We can still use Δchi2!

Though now we have 2 free parameters

- so 2 degrees of freedom ($k = 2$)

What about the error PDF or confidence levels?

We can still use Δchi2!

Though now we have 2 free parameters

- so 2 degrees of freedom ($k = 2$)

But how do we map the 1D bounds to 2D?

What about the error PDF or confidence levels?

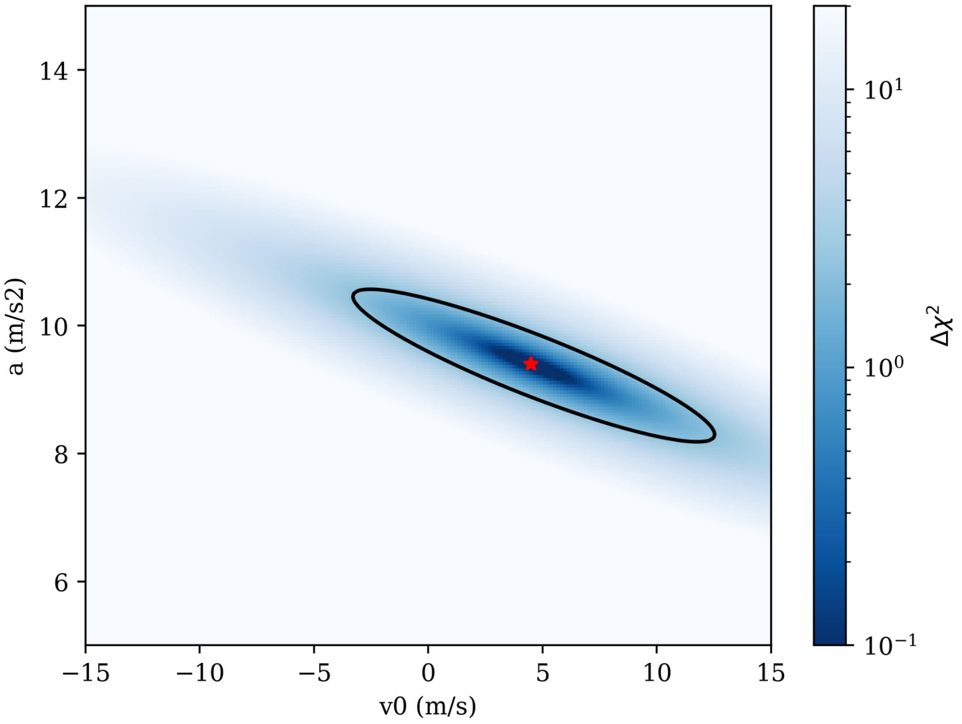
We can still use Δchi2!

Though now we have 2 free parameters

- so 2 degrees of freedom (k = 2)

But how do we map the 1D bounds to 2D?

- A 2D contour!

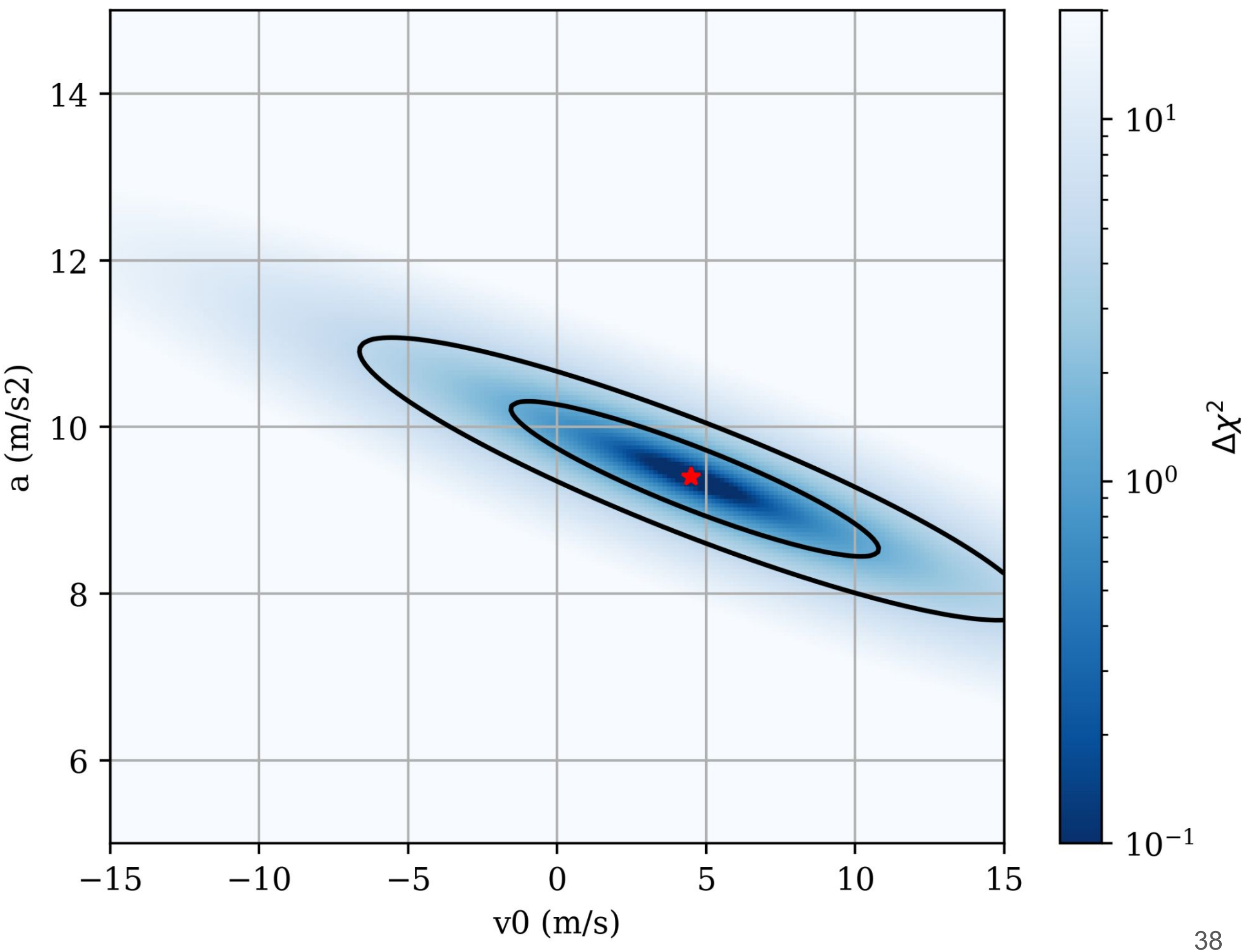


68% contour

stats.chi2.ppf(0.5, 2) = 1.39

stats.chi2.ppf(0.9, 2) = 4.6

90% probability true a and v0 is inside the 90% contour

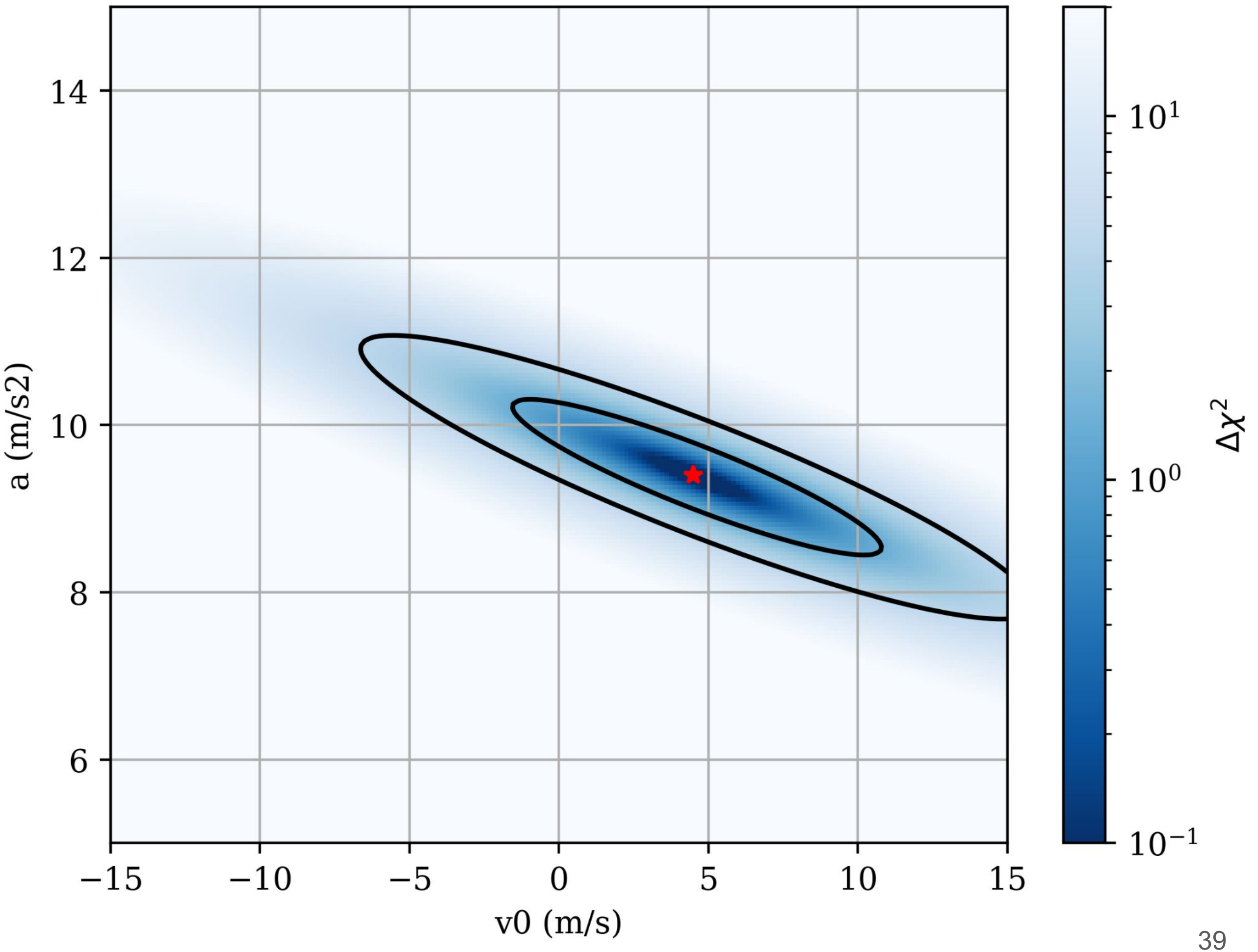


stats.chi2.ppf(0.5, 2) = 1.39

stats.chi2.ppf(0.9, 2) = 4.6

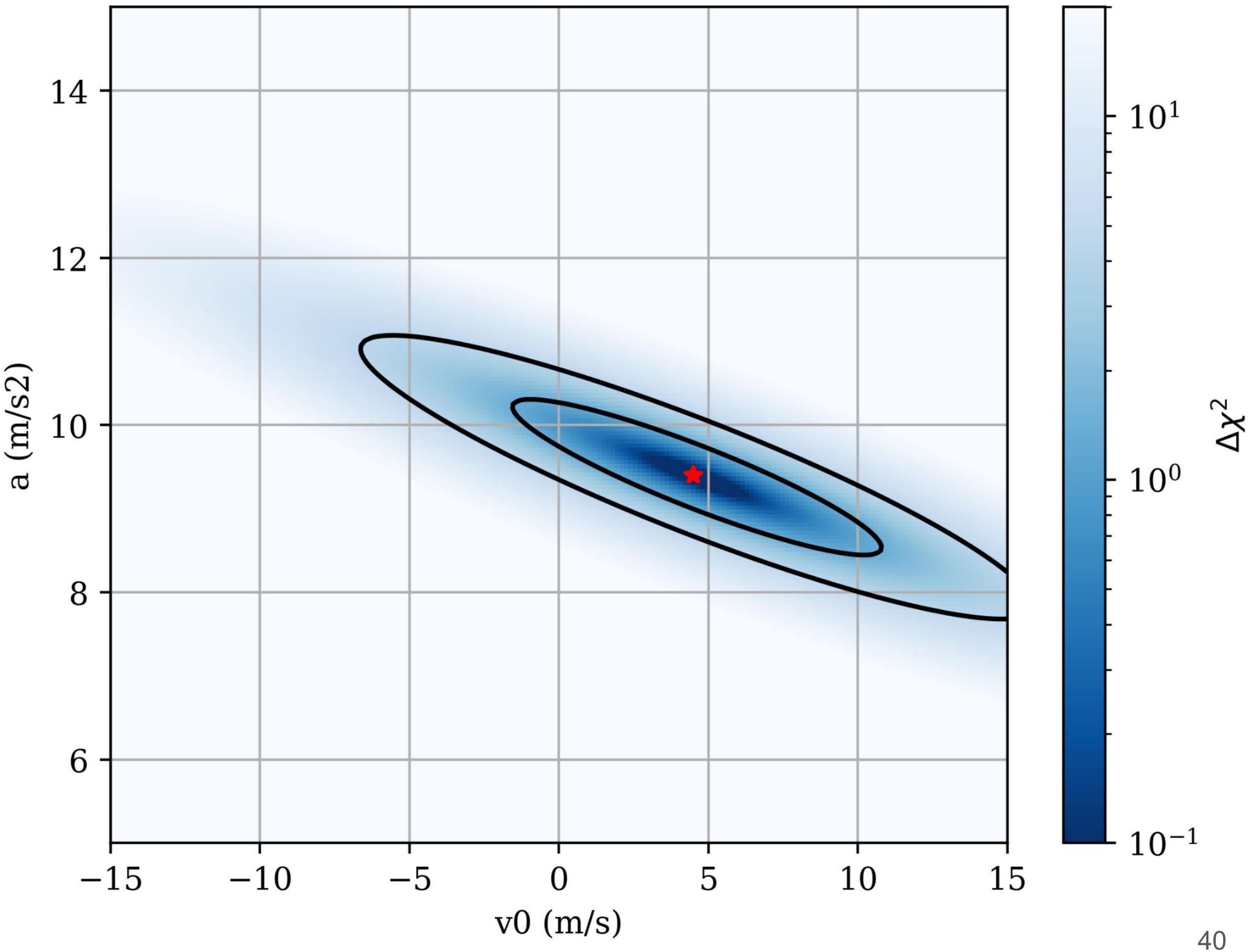
90% probability true a and v0 is inside the 90% contour

So how do we get an error bar from this?



So how do we get an error bar from this?

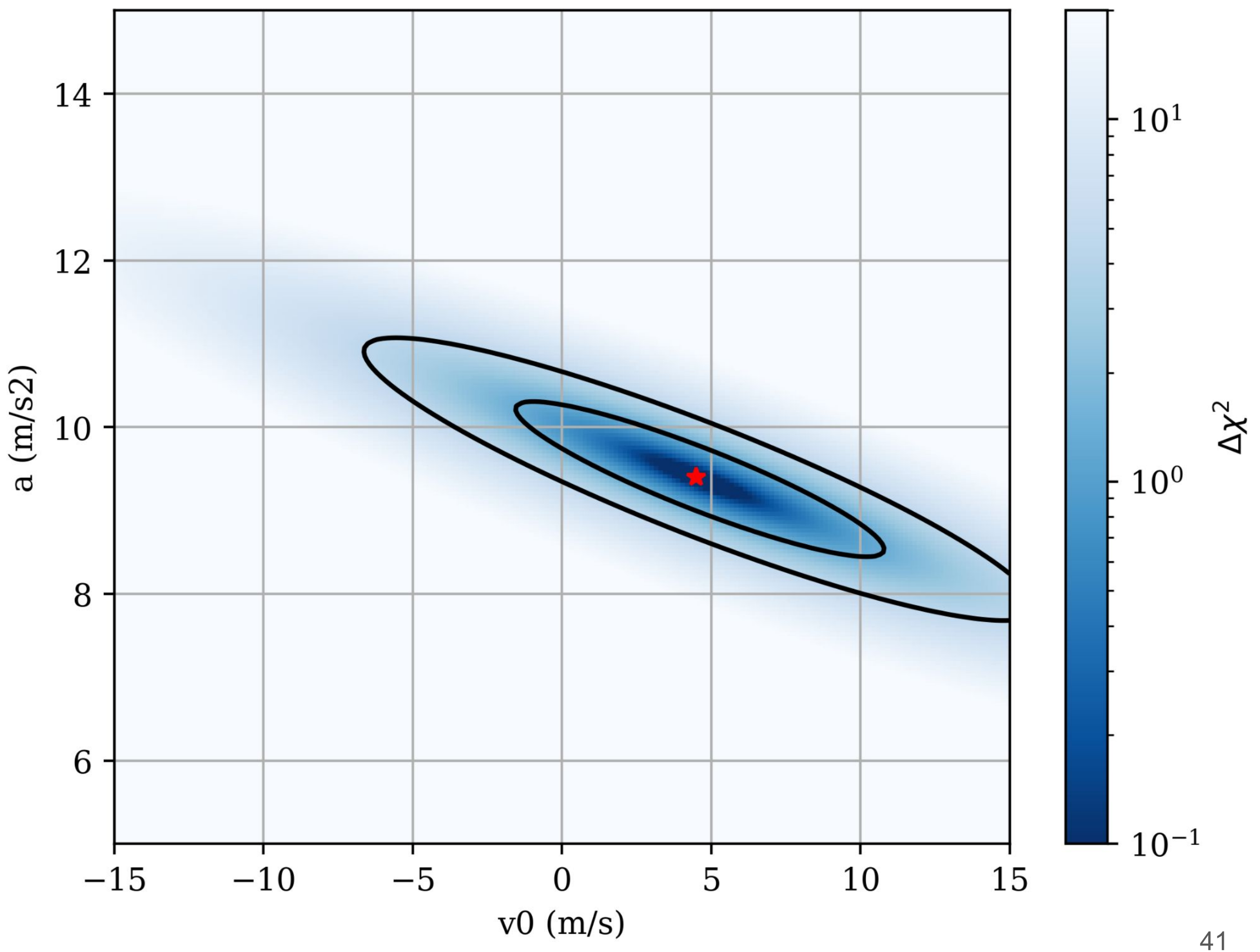
The a and v0 values that give lines that can describe the data are correlated



So how do we get an error bar from this?

The a and v0 values that give lines that can describe the data are correlated

The possible values of a, depend on v0

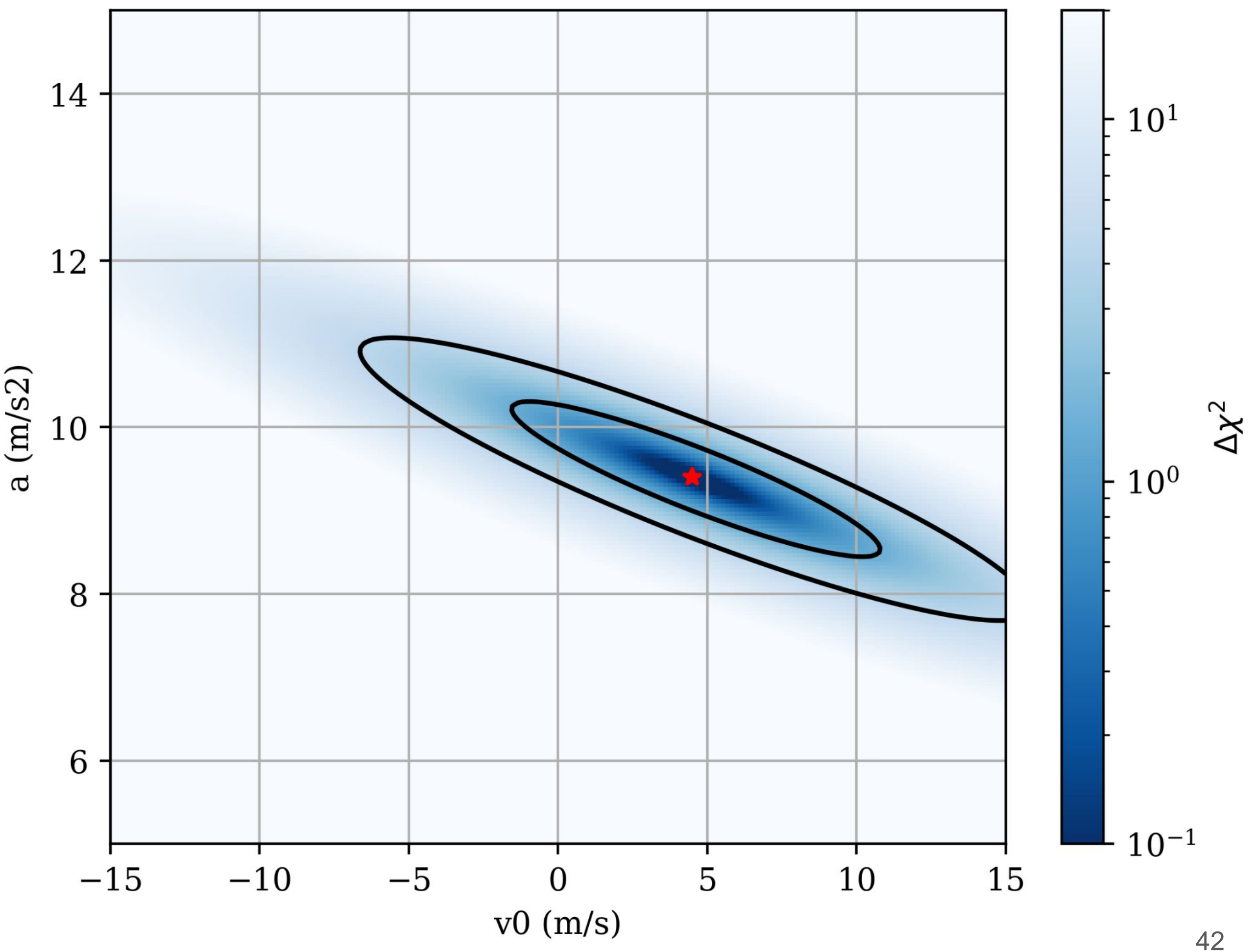


So how do we get an error bar from this?

The a and v0 values that give lines that can describe the data are correlated

The possible values of a, depend on v0

How to present this depends on what you want to know



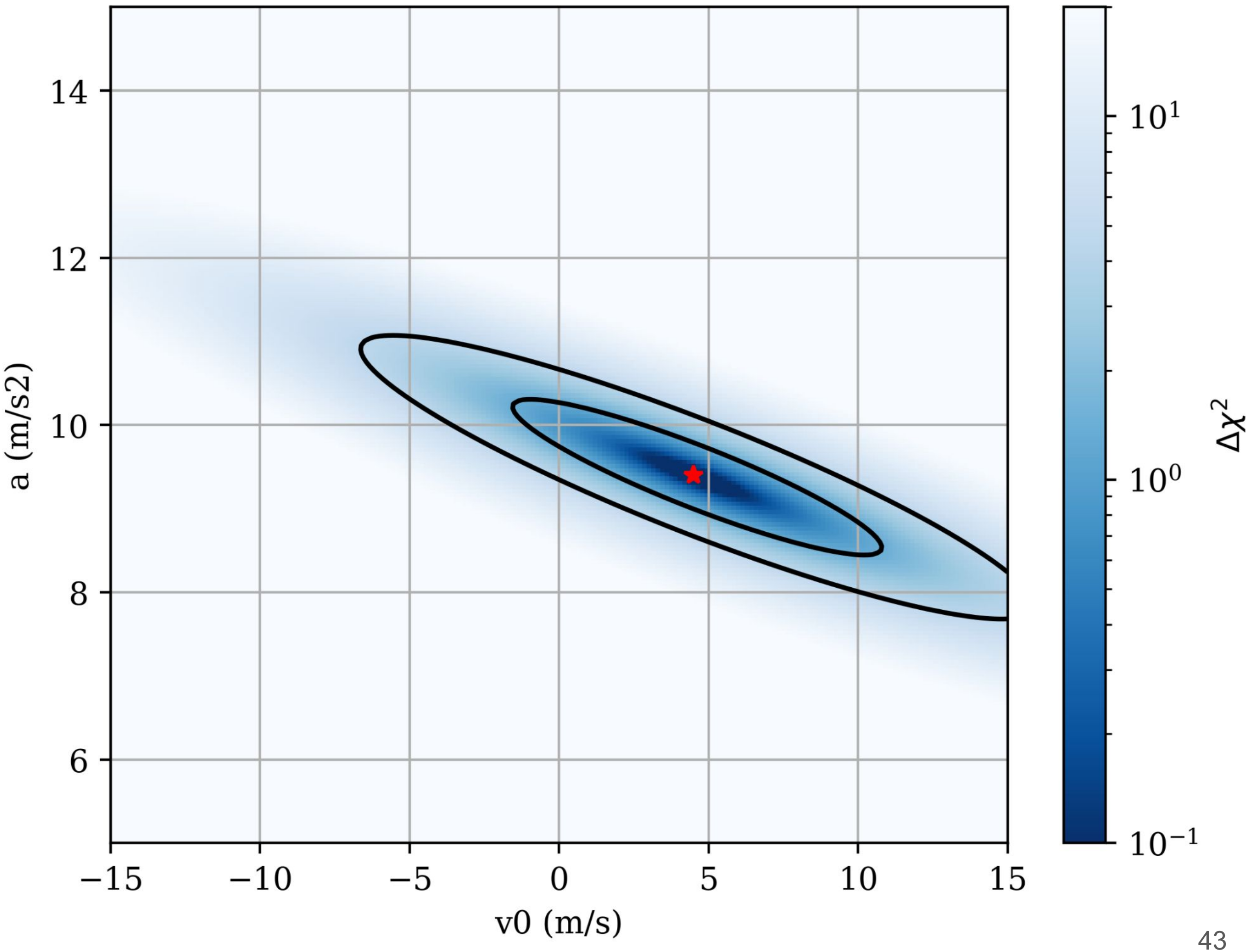
So how do we get an error bar from this?

The a and v0 values that give lines that can describe the data are correlated

The possible values of a, depend on v0

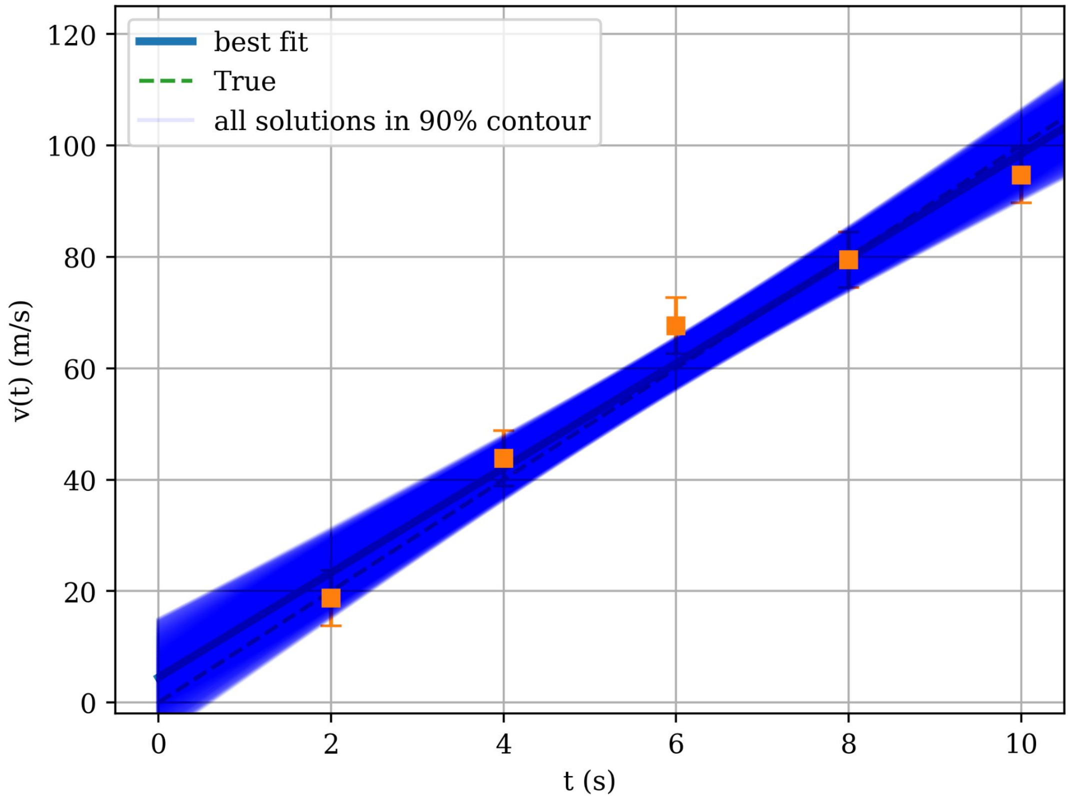
How to present this depends on what you want to know

This shows us all the allowed parameter space



We can also map this to our v(t) vs t plot

What are the allowed v values as a function of t



What if we instead wanted to know the error bar or PDF of 1 of the parameters

What if we instead wanted to know the error bar or PDF of 1 of the parameters

Say we don’t care what a is, we only care about what v0 is

- Did that dragster have a rolling start?

What if we instead wanted to know the error bar or PDF of 1 of the parameters

Say we don’t care what a is, we only care about what v0 is

- Did that dragster have a rolling start?

In this case a is what’s called a **nuisance parameter**

- An unknown free parameter in our model that we are not interested in

Getting rid of a nuisance parameter

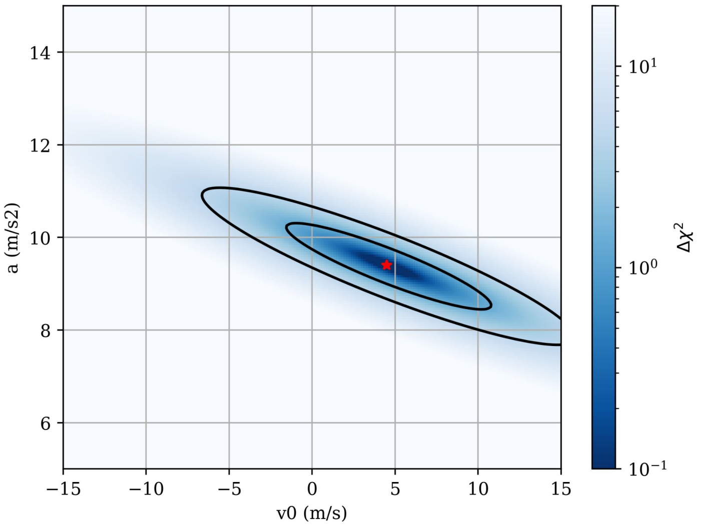
To “get rid” of a nuisance parameter you do what’s called profiling.

You have a 2D parameter space, but you can reduce that by minimizing chi2 over a, for each v0.

Getting rid of a nuisance parameter

To “get rid” of a nuisance parameter you do what’s called profiling.

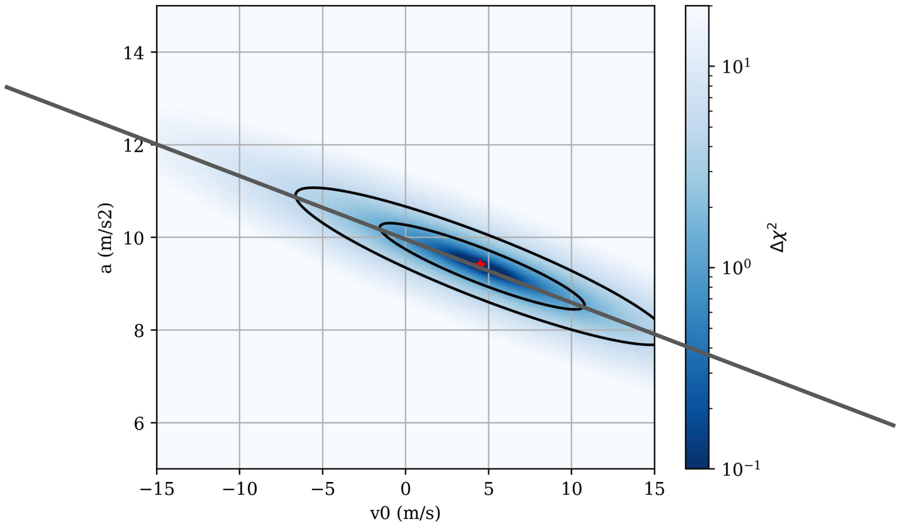
You have a 2D parameter space, but you can reduce that by minimizing chi2 over a, for each v0.



Getting rid of a nuisance parameter

To “get rid” of a nuisance parameter you do what’s called profiling.

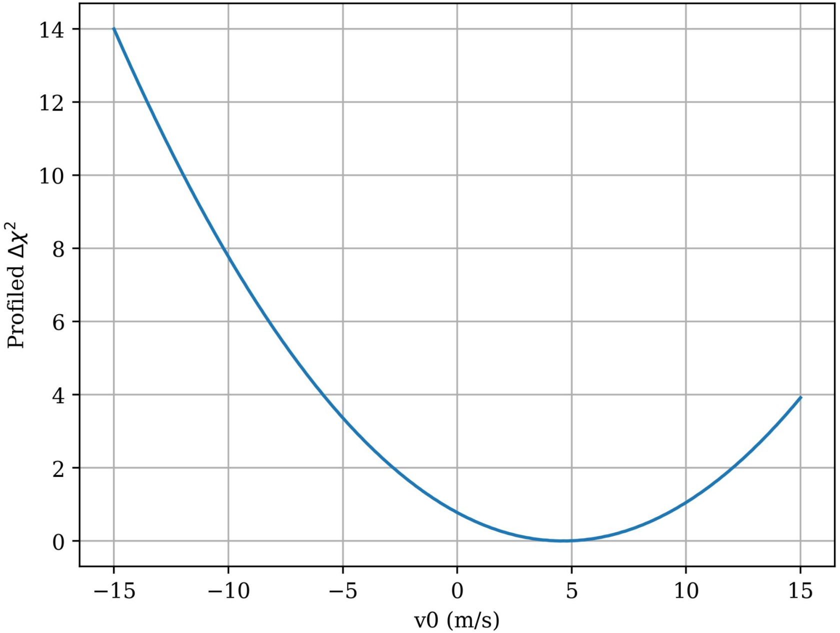
You have a 2D parameter space, but you can reduce that by minimizing chi2 over a, for each v0.



Getting rid of a nuisance parameter

To “get rid” of a nuisance parameter you do what’s called profiling.

You have a 2D parameter space, but you can reduce that by minimizing chi2 over a, for each v0.

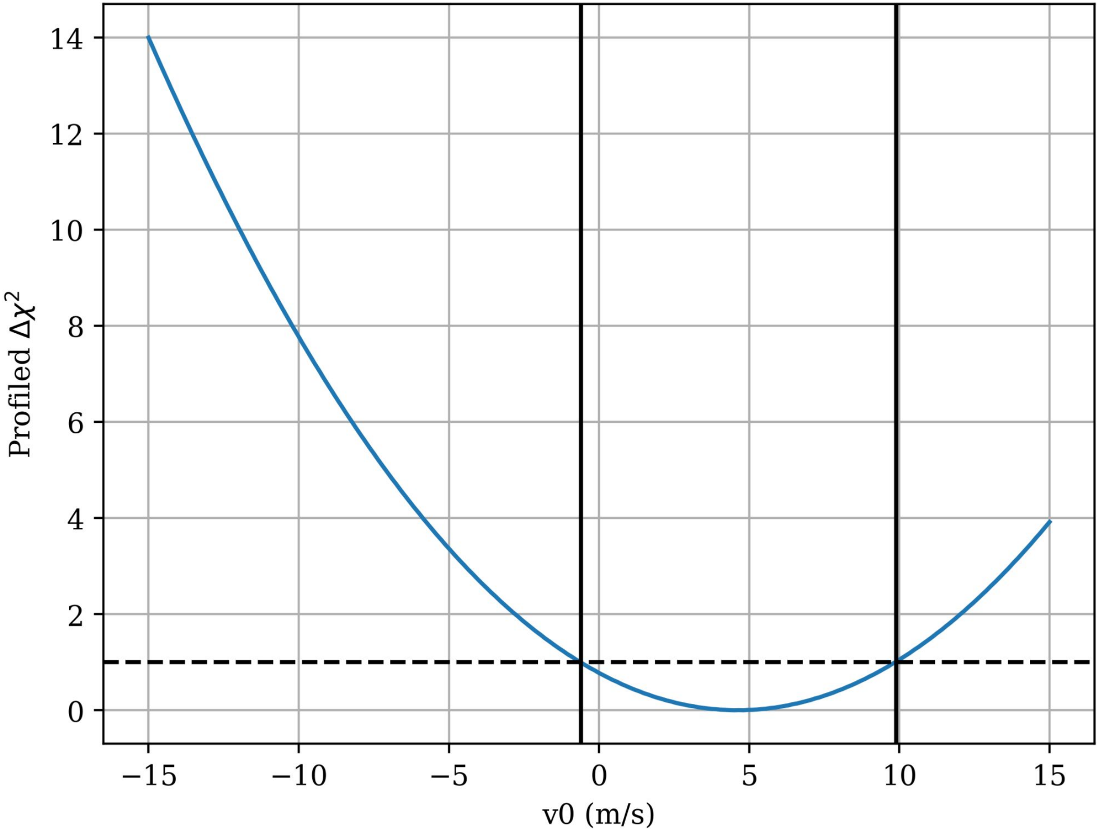


Getting rid of a nuisance parameter

We can use our chi2 with 1 dof again to find our 1D 68% error bar

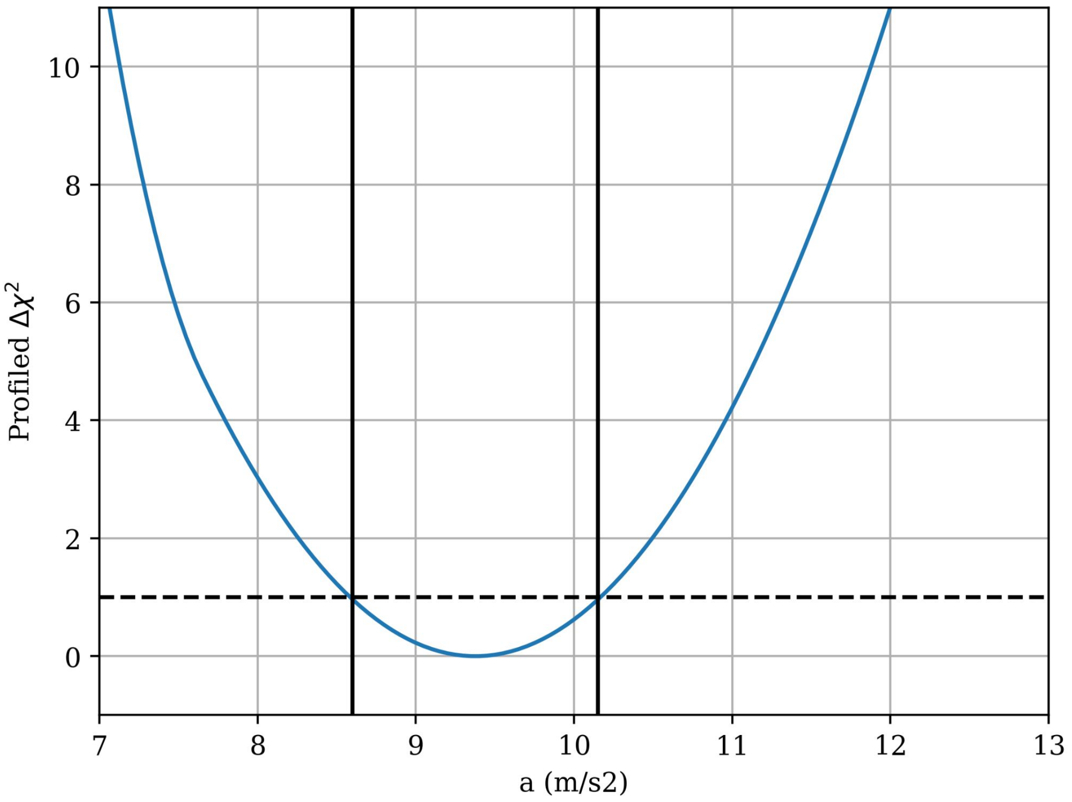
- We reduced the dof by constraining 1 of them

v0 = 4.5 +/- 5.2 m/s



We can do the same with a

a = 9.4 +/- 0.8 m/s2

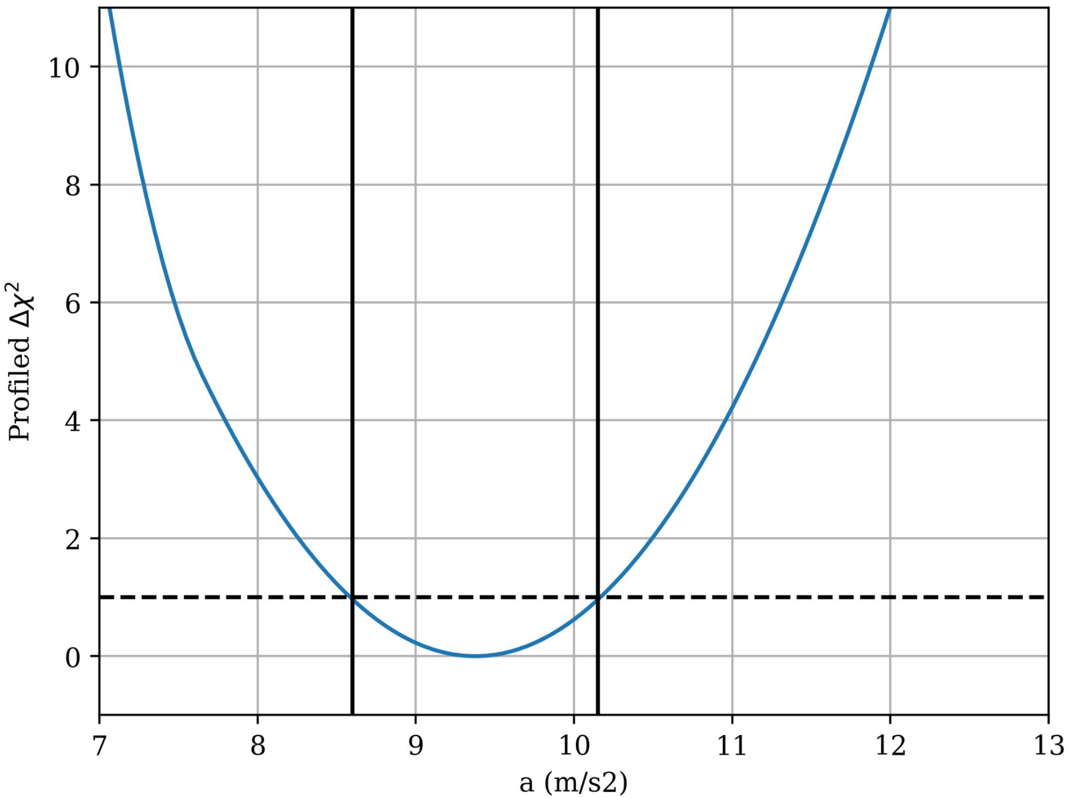


We can do the same with a

a = 9.4 +/- 0.8 m/s2

Previously with v0=0 we got

a = 10 +/- 0.33 m/s2



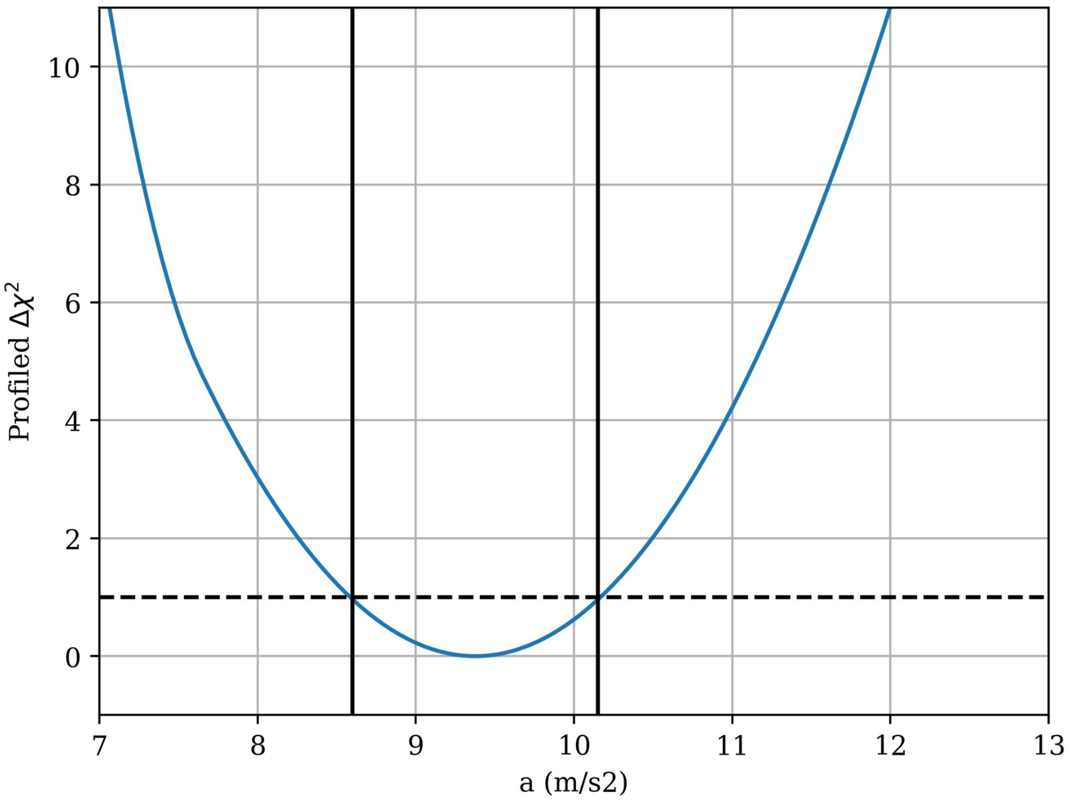
We can do the same with a

a = 9.4 +/- 0.8 m/s2

Previously with v0=0 we got

a = 10 +/- 0.33 m/s2

Why did the error bar get bigger?



We can do the same with a

a = 9.4 +/- 0.8 m/s2

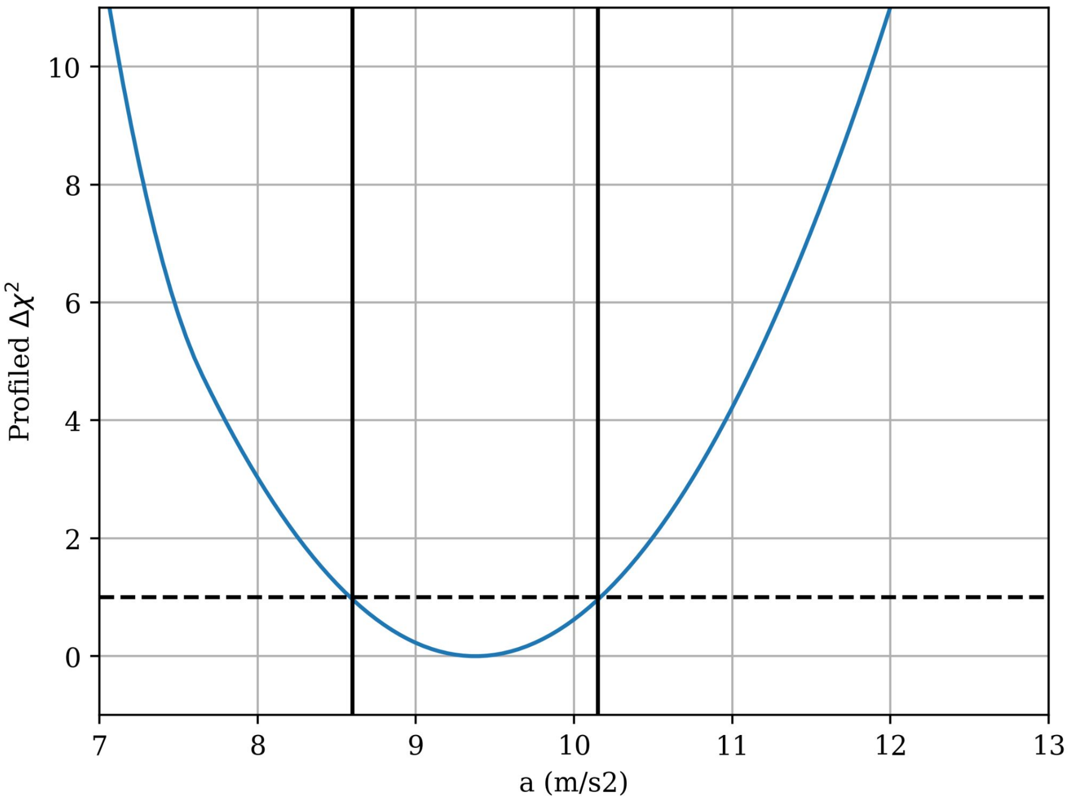
Previously with v0=0 we got

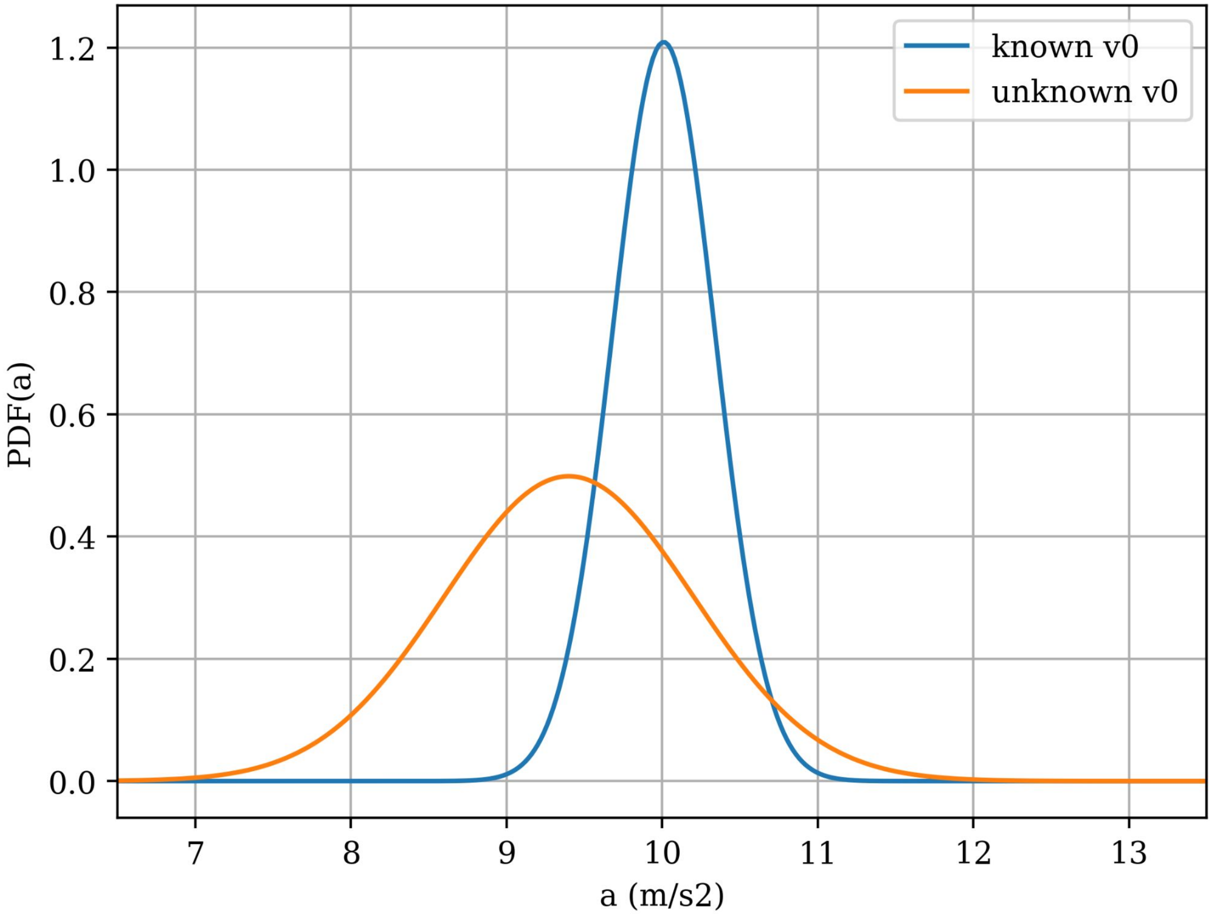
a = 10 +/- 0.33 m/s2

Why did the error bar get bigger?

We have less information

- v0 could be anything




known v0
unknown v0
PDF(a)
a (m/s2)
0.0
0.2
0.4
0.6
0.8
1.0
1.2
7
8
9
10
11
12
13

Summary

- Each observation/measurement in our data set has its own error
- Counting is a type of measurement
  - Follows Poisson statistics
  - Std dev on “expected counts” = sqrt(counts)
- A model can be applied to extract information from a dataset
  - Chi2 calculates the “distance” from model expectation to actual data
  - min(chi2) gives “best” parameters
  - Chi2 statistics can be used to find the error around those “best” parameters