MORE LIKELIHOOD

PARAMETER ESTIMATION

$$f(x;\theta) = \frac{1}{\theta}e^{-x/\theta}$$

r.v. parameter

$$\vec{x} = (x_1, \ldots, x_n)$$

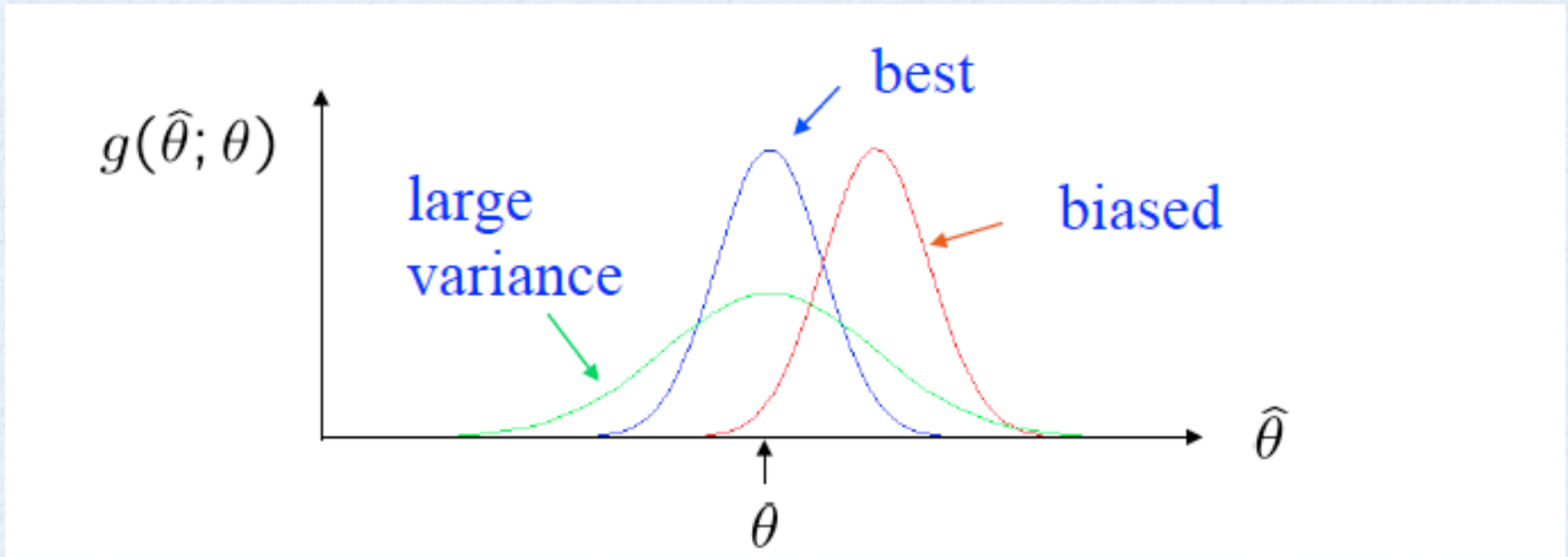
- Suppose we have some known distribution f and some measured data values
- We want to find some estimation of the parameter theta
- we call it the "estimator"

$$\hat{\theta}(\vec{x})$$

PRESENTATION TOPICS

- Still only 3 have indicated what they would like to do. If you don't tell me by Sunday I will assign them.
- They will be done during the last 3 classes randomly assigned...

WHAT ARE WE LOOKING FOR?



- We would like an estimator that has little or zero bias (i.e. it estimates the true value of the parameter not something else) - we call this the “systematic error”
- We would like an estimator that has a small variance - we call this the “statistical error”

- Suppose the entire result of an experiment (set of measurements) is a collection of numbers x, and suppose the joint pdf for the data x is a function that depends on a set of parameters θ:

$$f(\vec{x};\vec{\theta})$$

- Now evaluate this function with the data obtained and regard it as a function of the parameter(s). This is the likelihood function:

$$L(\vec{\theta}) = f(\vec{x};\vec{\theta})$$

LIKELIHOOD

- Consider n independent observations of x: x1, ..., xn, where x follows f (x; θ). The joint pdf for the whole data sample is:

$$f(x_1, \ldots, x_n; \theta) = \prod_{i=1}^{n} f(x_i; \theta)$$

- The likelihood function is

$$L(\vec{\theta}) = \prod_{i=1}^{n} f(x_i; \vec{\theta})$$

EXAMPLE: EXPONENTIAL

- Consider the exponential probability distribution function (for example probability of a particle with lifetime τ to decay at time t)

$$f(t;\tau) = \frac{1}{\tau} e^{\frac{-t}{\tau}}$$

- Suppose we have data t1, t2, ..tn

- The likelihood of any particular value τ is the product of the probability density function evaluated at that τ with the observed data points

$$L(\tau) = \prod_{i=1}^{n} \frac{1}{\tau} e^{\frac{-t_i}{\tau}}$$

Let's evaluate it

- Recall we can maximize the likelihood by minimizing the -log likelihood
- since the log of a product is just the sum of the logs we can write it:

$$lnL(\tau) = ln \prod_{i=1}^{n} \frac{1}{\tau} e^{\frac{-t_i}{\tau}} = \sum_{i=1}^{n} (ln \frac{1}{\tau} - \frac{t_i}{\tau})$$

DOING THE MATH

- Setting the derivative equal to 0 and solving for τ and using the properties of the log

$$\frac{\delta lnL(\tau)}{\delta \tau} = 0$$

$$\frac{\delta lnL(\tau)}{\delta \tau} = -\sum_{i=1}^{i=n} \frac{1}{\tau} + \sum_{i=1}^{i=n} \frac{t_i}{\tau^2} = 0$$

- We arrive at the simple result that the best estimate of the lifetime is just the mean value of the measured values

- Seems to make sense!

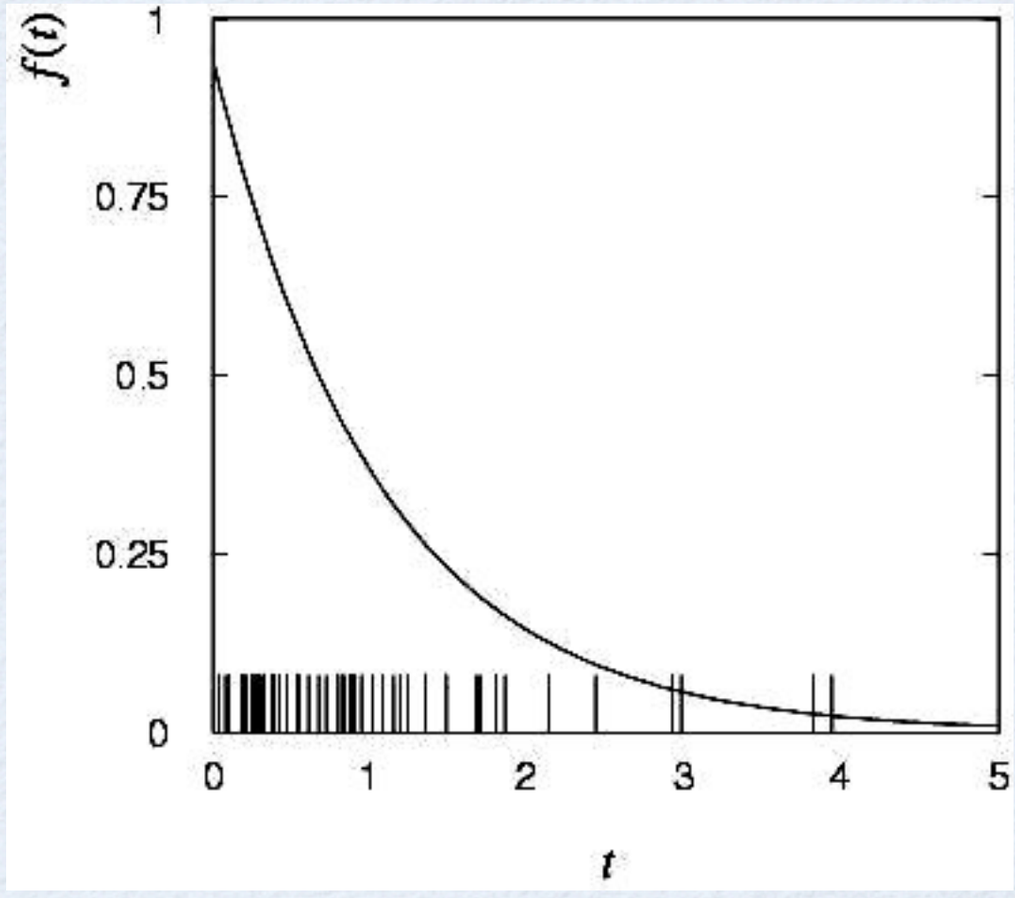
$$\sum_{i=1}^{i=n} \tau = \sum_{i=1}^{i=n} t_i$$

$$n\tau = \sum_{i=1}^{i=n} t_i$$

$$\tau = \frac{1}{n} \sum_{i=1}^{i=n} t_i$$

NUMERIC EXAMPLE

- Setting τ=1 and generating 50 random numbers from the distribution
- We find our estimate to be τ = 1.062
- Seems to work!



What do we do in general?

- For the lifetime example - the likelihood is simple enough that we can just solve it analytically. In general this is not true.
- We can however do this numerically as well...

```
sum = 0;
for (i=0; i<N; i++) {
  sum += log(tau) + t[i]/tau;
}

Find minimum sum by scanning in tau
```

In general we do something like this...

EXPECTATION VALUE

To see if there is a bias - we can compute the expectation value of our estimator for τ

$$E(\hat{\tau}) = E(\frac{1}{n}\sum_{i=1}^{n} t_i) = \frac{1}{n}\sum_{i=1}^{n} E(t_i)$$

$$E(\hat{\tau}) = \frac{1}{n}\sum_{i=1}^{n} \int t_i \frac{1}{\tau} e^{-\frac{t_i}{\tau}}\, dt_i$$

By explicit computation we see that our estimator is unbiased for all values of n

$$E(\hat{\tau}) = te^{-\frac{t}{\tau}}|_0^\infty - \int_0^\infty e^{-\frac{t}{\tau}}\, dt$$

$$E(\hat{\tau}) = \tau$$

CONSIDER A SLIGHT MODIFICATION

$$f(t;\tau) = \frac{1}{\tau} e^{\frac{-t}{\tau}}$$

- What if we made a simple substitution and used the width rather than the lifetime and wrote

let $\Gamma = \frac{1}{\tau}$

$$f(t,\Gamma) = \Gamma e^{-\Gamma t}$$

AGAIN LIKELIHOOD

$$-lnL(\Gamma) = -\sum_{i=1}^{n} ln(\Gamma e^{-\Gamma t_i}) = \sum_{i=1}^{n}(-ln\Gamma + \Gamma t_i)$$

$$-ln(L) = -Nln\Gamma + \Gamma \sum_{i=1}^{n} t_i$$

$$\frac{\delta(-ln(L)}{\delta\Gamma} = 0 = \frac{-N}{\Gamma} + \sum_{i=1}^{n} t_i$$

SO

$$\hat{\Gamma} = \frac{n}{\sum_{i=1}^{N} t_i} \qquad \hat{\Gamma} = (\frac{1}{\hat{\tau}})$$

BIAS?

- You might think because the estimate for τ is unbiased that the estimate for Γ must also be.

BIAS?

- You might think because the estimate for τ is unbiased that the estimate for Γ must also be.
- Unfortunately this is not correct!

if you do the integral and sums...

$$E(\hat{\Gamma}) = \Gamma \frac{n}{n-1}$$

How does this happen?

$$E(x) = \int x f(x)$$

- Recall the definition of the expectation value

$$E(\frac{1}{x}) = \int x f(\frac{1}{x})$$

$$E(\frac{1}{x}) \neq E(x)$$

for example by explicit calculation

$$f(x) = Ax$$

How does this happen?

- Recall the definition of the expectation value
- Therefore if E(x) is unbiased E(1/x) must be biased!

$$E(x) = \int x f(x)$$

$$E(\frac{1}{x}) = \int x f(\frac{1}{x})$$

$$E(\frac{1}{x}) \neq E(x)$$

for example by explicit calculation

$$f(x) = Ax$$

UNCERTAINTY FROM LIKELIHOOD

- We would also like an estimate of the uncertainty
- Expand Log Likelihood in Taylor Series about minimum

$$lnL(\theta) = lnL_{\hat{\theta}} + \frac{1}{2}\frac{\delta^2 lnL}{\delta\theta^2}|_{\hat{\theta}}(\theta - \hat{\theta})^2$$

so cutting it off we can write

$$L(\theta) = L_{\hat{\theta}} e^{\frac{1}{2}\frac{\delta^2 lnL}{\delta\theta^2}|_{\theta=\hat{\theta}}(\theta-\hat{\theta})^2}$$

so..

$$L(\theta) = L_{\hat{\theta}} e^{\frac{1}{2}\frac{\delta^2 lnL}{\delta\theta^2}|_{\theta=\hat{\theta}}(\theta-\hat{\theta})^2}$$

- Around a minimum the likelihood function looks like a Gaussian with a variance given by:

$$\sigma^2 = -\frac{1}{\frac{\delta^2 lnL}{\delta\theta^2}\Big|_{\theta=\hat{\theta}}}$$

Meaning of Likelihood

$$L(t_i|\tau) = \prod_{i=1}^{n} P(t_i|\tau)$$

- The likelihood function is a product of probabilities

- It tells us something about the probability of data given a particular τ

- A priori it does not tell us (directly)about the probability of a particular τ given data

- Why not?

IN GENERAL

$$P(A|B) \neq P(B|A)$$

FOR EXAMPLE

- Say I had a bird as a pet and I told you that it was a crow and asked you what color it was.
- What would you say?

ALL CROWS ARE BLACK

BUT

- If i told you it was a black bird and asked what type of bird it was...
- What would you say?

NOT ALL BLACK BIRDS ARE CROWS

so..

$$P(crow|blackbird) \neq P(blackbird|crow)$$

OR

≠

BAYES THEOREM

- Bayes theorem tells us how to relate the two!

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

IN OUR CASE

- P(data) - is really just a normalization factor
- P(τ) is our `prior' distribution

$$P(\tau|data) = \frac{P(data|\tau)P(\tau)}{P(data)}$$

IN OUR CASE

- P(data) - is really just a normalization factor
- P(τ) is our `prior' distribution
- If we assume all τ are equally likely then and ONLY then do we have

$$P(\tau|data) = \frac{P(data|\tau)P(\tau)}{P(data)}$$

$$P(\tau|data) = P(data|\tau)$$

REWRITING IT...

$$lnL(\theta) = logL|_{\hat{\theta}} - \frac{1}{2}\frac{(\theta - \hat{\theta})^2}{\sigma^2}$$

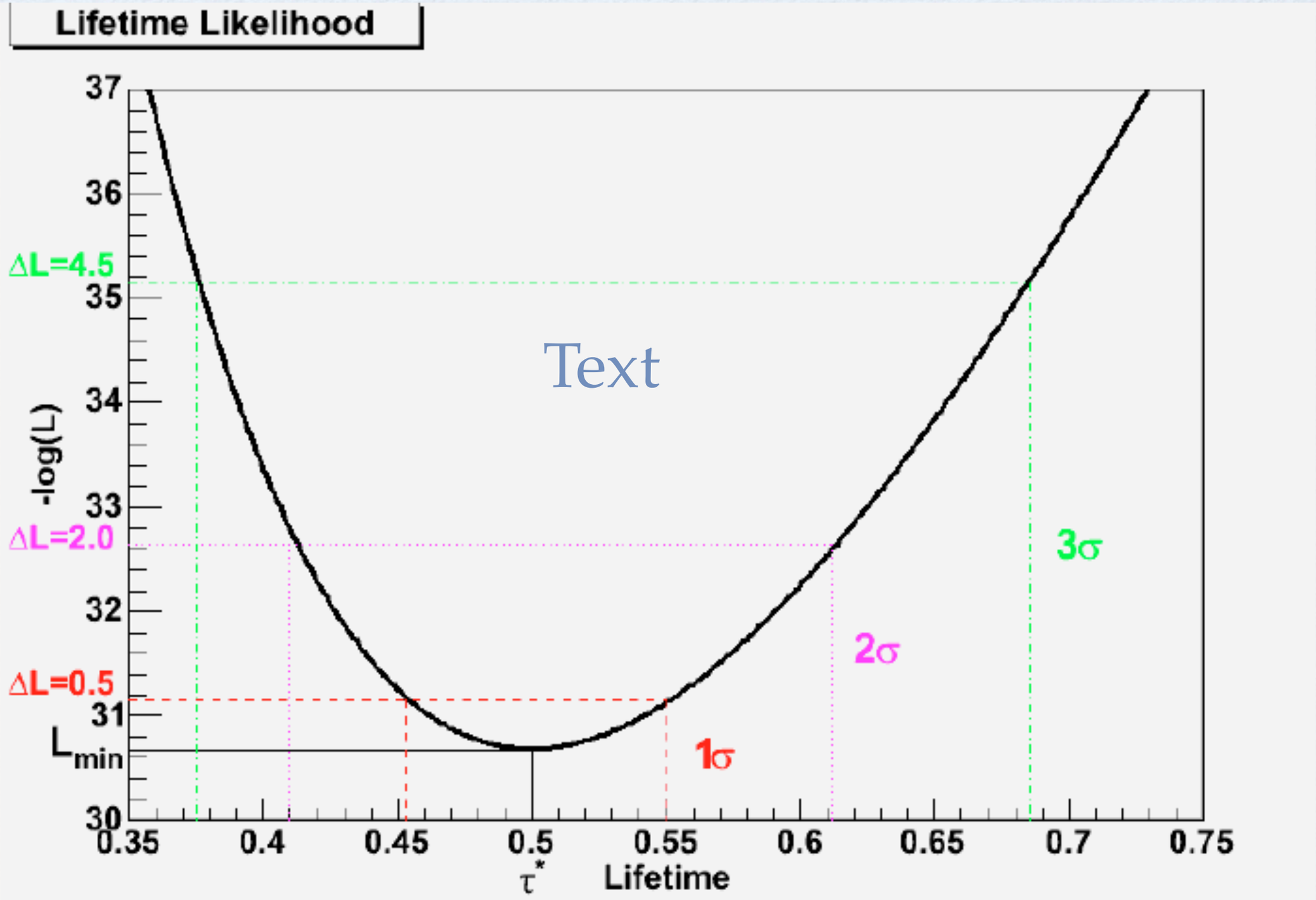
- So the value for the likelihood being 1,2,..n standard deviations away from the central value is just:

$$-lnL(\hat{\theta} \pm 1\sigma) = -logL|_{\hat{\theta}} + \frac{1}{2}$$

$$-lnL(\hat{\theta} \pm 2\sigma) = -logL|_{\hat{\theta}} + 2$$

$$-lnL(\hat{\theta} \pm n\sigma) = -logL|_{\hat{\theta}} + \frac{1}{2}n^2$$

GRAPHICALLY



SO IN OUR EXAMPLE

$$-lnL = Nlog\tau + \frac{1}{\tau} + \frac{1}{\tau}\sum_{i=1}^{n} t_i$$

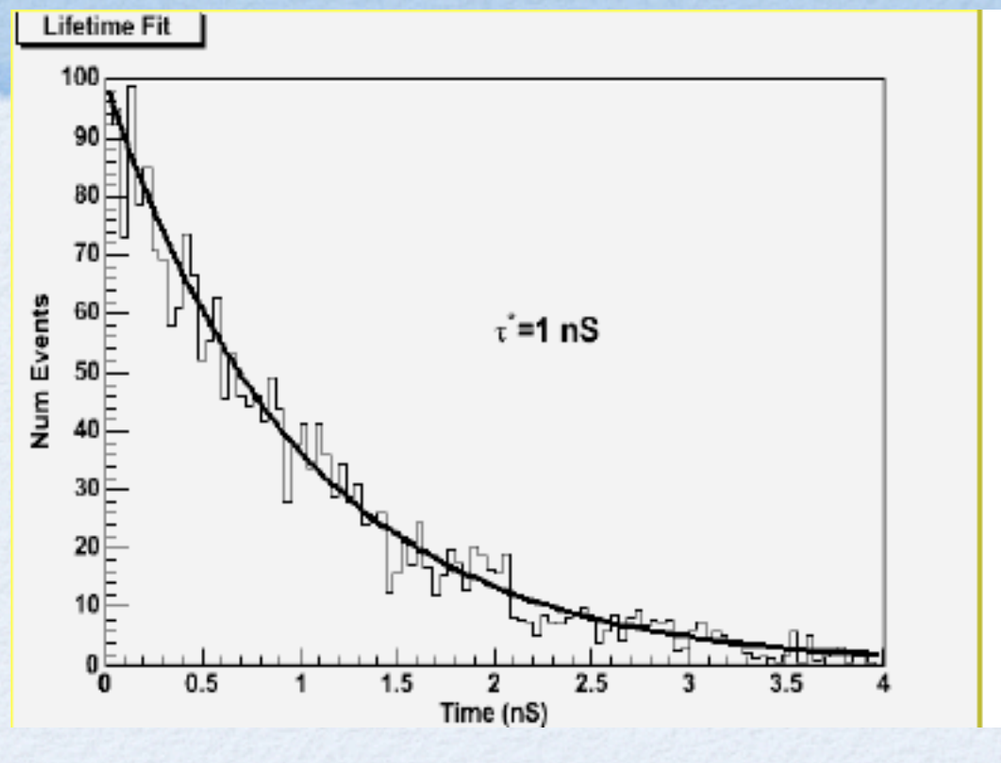
$$\frac{\delta(-lnL)}{\delta\tau}\frac{N}{\tau} - \frac{1}{\tau^2}\sum_{i=1}^{n} t_i$$

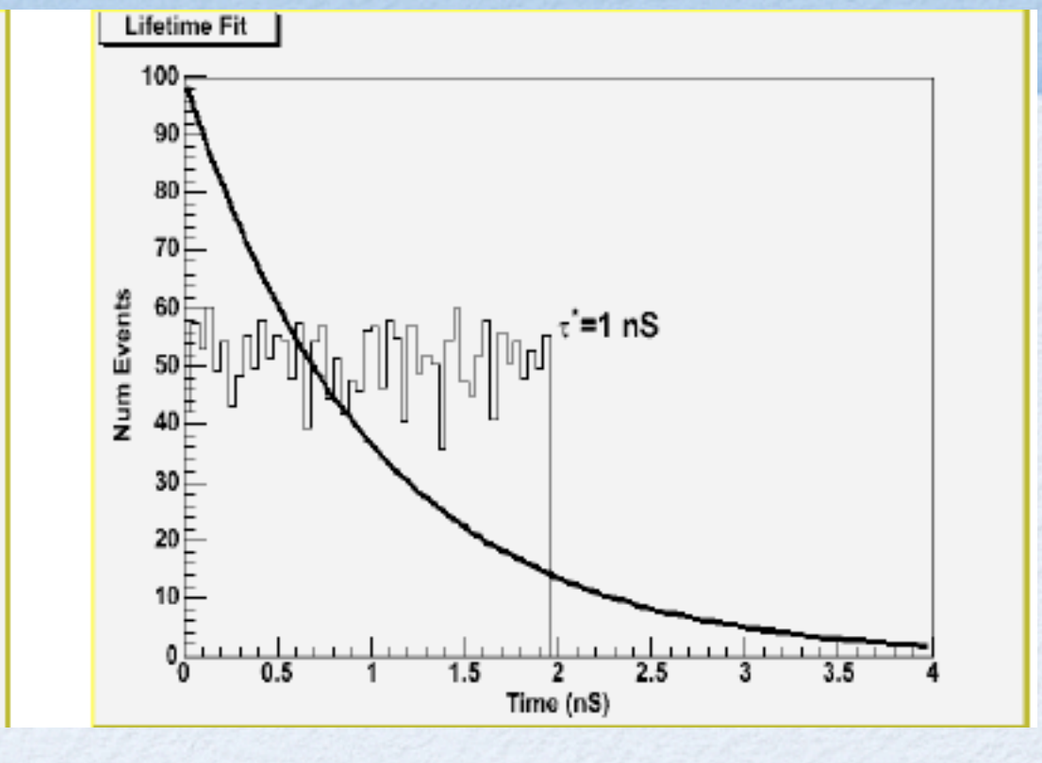
$$\frac{\delta(-lnL)}{\delta\tau} = -\frac{N}{\hat{\tau}^2} + \frac{2}{\hat{\tau}^3}\sum_{i=1}^{n} t_i$$

$$= -\frac{N}{\hat{\tau}^2} + \frac{2N}{\hat{\tau}^2} = \frac{N}{\hat{\tau}^2}$$

$$\sigma = \frac{\hat{\tau}}{\sqrt{n}}$$

GOODNESS OF FIT





- Consider the two data sets above, made into histograms for visualization. Both result in the same ML estimator for the lifetime.
- Surely, data set 1 is "more" likely than data set 2, right?
- Surely, since it is more exponential, the value of the likelihood function for the 1st should be larger than for the 2nd, right? Each events "probability" should be higher, resulting in a net larger likelihood.

OOPS

Unfortunately this is wrong!

$$-LnL = nln\tau + \frac{1}{\tau}\sum_{i=1}^{n} t_i$$

$$\hat{\tau} = \frac{1}{n}\sum_{i=1}^{n} t_i$$

so..

$$-lnL_{min} = Nln(\hat{\tau}) + \frac{1}{\hat{\tau}}\sum_{i=1}^{n} t_i$$

$$= nln(\frac{1}{n}\sum t_i) + \frac{n}{\sum t_i}\sum t_i$$

$$= n(-lnn + 1 + ln\sum t_i) = n(ln\hat{\tau} + 1)$$

Any distribution with he same sum of times produces the same likelihood!

No measure of goodness of fit!

EXAMPLE CODE

- I have placed on blackboard a piece of code which shows a simple example of how to do a likelihood fit
- Let's walk through it
- You can find it as LikelihoodFit.C

LET'S LOOK

```
// ------------------------------------------------------------------------- //
void LikelihoodFit() {
// ------------------------------------------------------------------------- //

  // Set the showing of statistics and fitting results (0 means off, 1111 means all on):
  gROOT->Reset();
  gStyle->SetOptStat(1111);
  gStyle->SetOptFit(1111);

  // Set the graphics:
  gStyle->SetStatBorderSize(1);
  gStyle->SetCanvasColor(0);

// Parameters of the problem:
const int Ntimes = 100;      // Number of times
double tau_truth = 1.0;      // We choose (like Gods!) the lifetime.
vector<double> t;            // Vector of times

bool verbose = false;        // Should the program print (a lot) or not? (true/false)
TRandom3 r;

int Nbin_exp = 50;           // Number of bins in histogram
TH1F *Hist_Exp = new TH1F("Hist_Exp", "Hist_Exp", Nbin_exp, 0.0, 10.0);
```

Starts with some style stuff

We are going to generate some fake data and put it in this histogram

ASIDE: STANDARD TEMPLATE LIBRARY

- note the code
- vector<double>
- what does this mean?

```
// ------------------------------------------------------------------------------- //
void LikelihoodFit() {
// ------------------------------------------------------------------------------- //

  // Set the showing of statistics and fitting results (0 means off, 1111 means all on):
  gROOT->Reset();
  gStyle->SetOptStat(1111);
  gStyle->SetOptFit(1111);

  // Set the graphics:
  gStyle->SetStatBorderSize(1);
  gStyle->SetCanvasColor(0);

  // Parameters of the problem:
  const int Ntimes = 100;      // Number of times
  double tau_truth = 1.0;      // We choose (like Gods!) the lifetime.
  vector<double> t;            // Vector of times

  bool verbose = false;        // Should the program print (a lot) or not? (true/false)
  TRandom3 r;

  int Nbin_exp = 50;           // Number of bins in histogram
  TH1F *Hist_Exp = new TH1F("Hist_Exp", "Hist_Exp", Nbin_exp, 0.0, 10.0);
```

Standard Template Library

- Turns out - there are many simple things that people want to do all the time in C++
- Store data in an ordered way
- Run standard algorithms on data (sort, find, etc..)

C++ HAS A WAY

- Unless you are doing something very special and you know something special about your data (i.e. way it is organized, how it is ordered, etc) it is generally easiest to use the standard template library to take care of this

- C++ experts who have written optimized code how to store and sift through data

- Available to you at no cost!

EXAMPLE VECTOR<DOUBLE>

- a vector is much like an array (i.e. an ordered list of numbers)
- an array is just a declaration of a set of numbers like
  - double x[3] = {1.2,1.3,0};
- A vector acts very much like an array in that it is a set of things but it also has many built in special functions
  - unlike an array it can be dynamically sized (you don't need to know its size when you declare it)
  - and a lot of other neat things

VECTOR

```
template <class T>
class vector {
   public:
      vector();
      vector(const vector<T>& originalMap);

      typedef implementation_specific_class_1 iterator;
      typedef implementation_specific_class_2 const_iterator;

      bool empty() const;          // true iff logical length is 0
      long size() const;           // returns logical length of vector
      void clear();                // empties the vector, sets size to 0

      void push_back(const T& elem);
      void pop_back();

      T& operator[](int i);
      const T& operator[](int i) const;
      iterator insert(iterator where, const T& elem);
      iterator erase(iterator where);

      iterator begin();
      iterator end();
      const_iterator begin() const;
      const_iterator end() const;
};
```

STL ALSO HAS ALGORITHMS

```
#include <algorithm>
#include <iostream>
#include <vector>

static bool sort_using_greater_than(double u, double v)
{
   return u > v;
}

int
main()
{
   std::vector<double> v;

   v.push_back(0.3);
   v.push_back(0.1);
   v.push_back(1.2);
   v.push_back(0.01);

   std::sort(v.begin(), v.end(), sort_using_greater_than);

   for(std::vector<double>::size_type index = 0; index < v.size(); ++index)
      std::cout << v[index] << std::endl;
}
```

Many containers, many algorithms

- Google C++ Standard Template Library
- Best by learning by example!

GENERATE THE DATA

```
// ------------------------------------
// Generate data:
// ------------------------------------
for (int i=0; i < Ntimes; i++) {
  t.push_back(-tau_truth*log(r.Uniform()));        // Exponential with lifetime tau.
  // t.push_back(r.Exp(tau_truth));                // Could also use this line!
  Hist_Exp->Fill(t[i]);
  if (verbose) printf("  %5.2f", t[i]);
}
if (verbose) printf("\n\n");
```

- Generate a random number according to the distribution we want
- push it back into our fake data histogram as if we measured this from data

SCAN VALUES

```
// Define the number of tau values to test in Chi2 and LLH:
const int Ntau = 1500;
double min_tau = 0.5;
double max_tau = 2.0;
double delta_tau = (max_tau-min_tau) / Ntau;

// Loop over hypothesis for the value of tau and calculate Chi2 and LLH:
// ------------------------------------------------------------------
double tau[Ntau+1], llh[Ntau+1];

double llh_minval  = 999999.0;  double llh_minpos  = 0.0;

for (int itau=0; itau < Ntau+1; itau++) {
   double tau_hypo = min_tau + double(itau)*delta_tau;   // Scan in values of tau
   tau[itau] = tau_hypo;
   // Unbinned likelihood (from loop over events):
   llh[itau]  = 0.0;
   for (int iev=0; iev < Ntimes; iev++) {
      llh[itau] += -2.0*log(1.0/tau_hypo*exp(-t[iev]/tau_hypo));
    }
    if (llh[itau] < llh_minval) {llh_minval = llh[itau]; llh_minpos = tau_hypo;}
}
```

- for one parameter - we don't need fancy code. Just scan through the range in small steps
- Compute likelihood for different values of tau and store it

FIT THE RESULT

- Could simply draw it but ..
- But easier to see and allows to calculate uncertainty

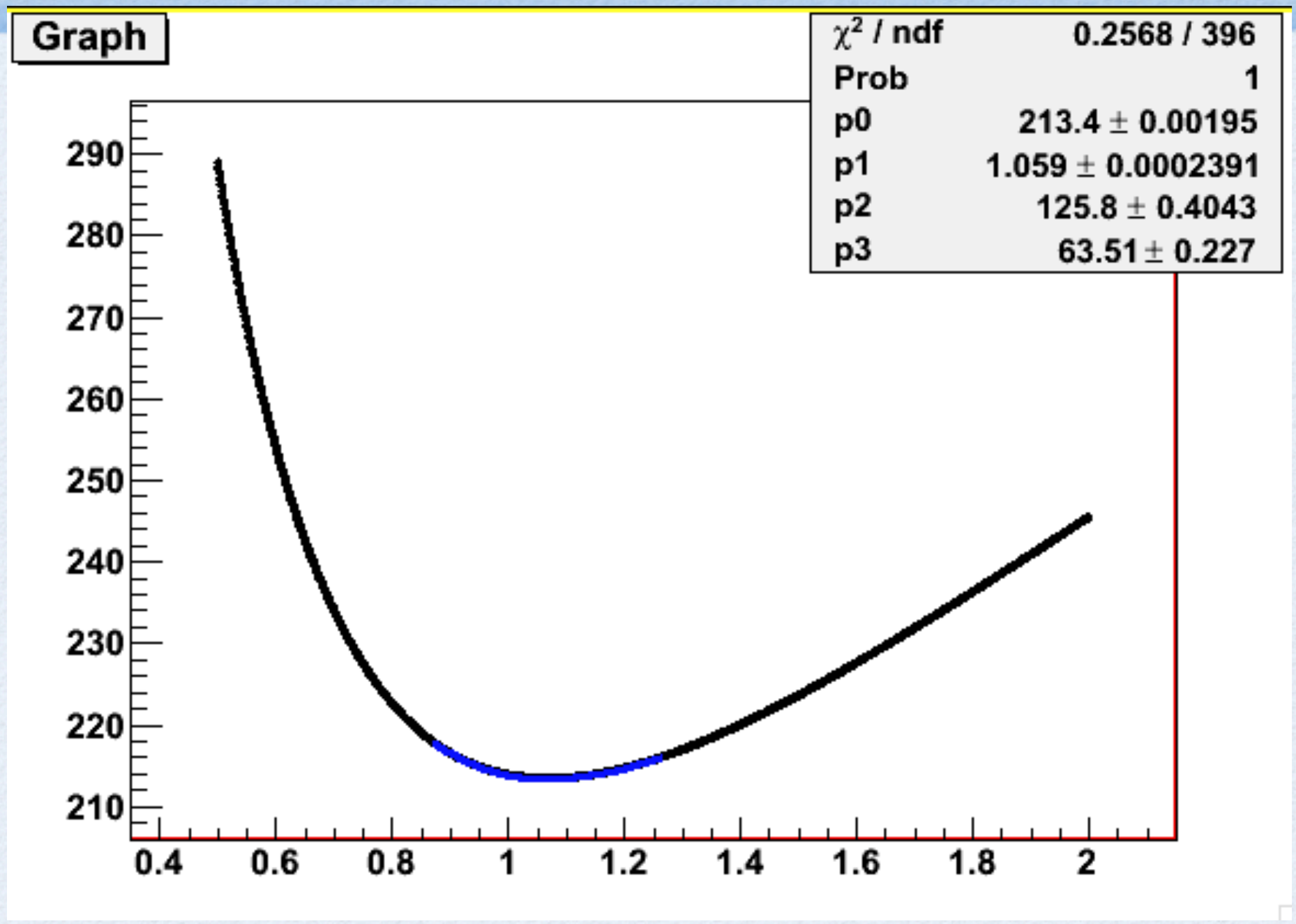
```
// Unbinned Likelihood:
TCanvas* c_llh = new TCanvas("c_llh","",140,80,600,450);
TGraphErrors* graph_llh = new TGraphErrors(Ntau+1, tau, llh);
TF1 *fit_llh = new TF1("fit_llh", func_asympara, llh_minpos - 0.20, llh_minpos + 0.20, 4);
fit_llh->SetParameters(llh_minval, llh_minpos, 20.0, 20.0);
fit_llh->SetLineColor(4);
graph_llh->SetMarkerStyle(20);
graph_llh->SetMarkerSize(0.4);
graph_llh->Fit("fit_llh","R");
graph_llh->Draw("AP");

c_llh->Update();
```

RUN IT

- In the directory where you have the file do:
- root
- .L LikelihoodFit.C
- LikelihoodFit()

RESULTS


Graph
χ^2 / ndf 0.2568 / 396
Prob 1
p0 213.4 ± 0.00195
p1 1.059 ± 0.0002391
p2 125.8 ± 0.4043
p3 63.51 ± 0.227
290
280
270
260
250
240
230
220
210
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2

More complicated

- For simple problems - like the one we just did. It is pretty straightforward to both construct the likelihood function and minimize it
- For complicated functions in many parameters this is not the case!
- Multidimensional minimization is much harder!!

CHECK TO SEE IF IT WORKS

- Change the lifetime by a bit (be careful that the range you scan includes the true value)
- You should find that the -ln(likelihood) will change it's position is at its minimum

ADDING CONSTRAINTS

- Sometimes we are in the situation where we have external information. For example if we are fitting a spectrum where we have well known particles with well known properties (that are much better known than we can measure) and we would like to include that within our analysis

- Since the likelihood function is a product of probability distribution functions we can modify our likelihood by multiplying a ‘prior’ probability distribution to the likelihood

EXAMPLE

Let's add a Gaussian constraint

$$L = P(\tau|\tau_{pdg},\sigma_{pdg})\prod_{i=1}^{n} P(t_i|\tau)$$

so then..

$$-lnL = -lnP(\tau|\tau_{pdg},\sigma_{pdg}) + \sum_{i}^{n} lnP(t_i|\tau)$$

$$= -ln(\frac{1}{\sqrt{(2\pi)}\sigma_{pdg}} e^{-\frac{(\tau-\tau_{pdg})^2}{2\sigma_{pdg}^2}}) + \sum_{i}^{n} lnP(t_i|\tau)$$

$$= \sum_{i=1}^{n} lnP(t_i|\tau) + \frac{(\tau-\tau_{pdg})^2}{2\sigma_{pdg}^2} + ln\sqrt{2\pi}\sigma_{pdg}$$

original

wandering

1/2 σ from pdg

constant

value costs 1/2 unit of likelihood

What does this do?

- By including this 'constraint' or penalty in the likelihood we enhance the likelihood in the region close to this value.

- We don't want to fix it because we want our data to count !

- So we add a term which penalizes it for moving away from the value where we have other information (other experimental results)

IMPERFECT DETECTOR

- Of course our detectors aren't perfect and the resolution can affect the spectrum that we measure

- Let's Convolute the function with a gaussian resolution indicating that we know that we have an uncertainty on every measurement we make

$$P(t \mid \tau) = Exp(t', \tau) \otimes G(0, \sigma)$$

$$= \frac{1}{\sqrt{2\pi}\sigma\tau} \int_0^\infty e^{-t'/\tau} e^{-\frac{(t-t')^2}{2\sigma^2}} dt'$$

EXAMINE THE EXPONENT

- Expanding it out we can write it as...

$$
\begin{aligned}
&= -\frac{1}{2\sigma^2}(t'^2 - 2t't + t^2 + 2\frac{\sigma^2}{\tau}t') \\
&= -\frac{1}{2\sigma^2}(t'^2 - 2t'(t - \frac{\sigma^2}{\tau}) + t^2) \\
&= -\frac{1}{2\sigma^2}\left[(t' - (t - \frac{\sigma^2}{\tau}))^2 + 2\frac{\sigma^2 t}{\tau} + \frac{\sigma^4}{\tau^2}\right] \\
&= -\frac{1}{2\sigma^2}(t' - (t - \frac{\sigma^2}{\tau}))^2 - \frac{t}{\tau} - \frac{\sigma^2}{2\tau^2}
\end{aligned}
$$

So that

PDF becomes

$$P(t \mid \tau) = \frac{1}{\sqrt{2\pi}\sigma\tau} \int_0^\infty e^{-t/\tau} e^{-\frac{\sigma^2}{2\tau^2}} e^{-\frac{(t' - (t - \frac{\sigma^2}{\tau}))^2}{2\sigma^2}} dt'$$

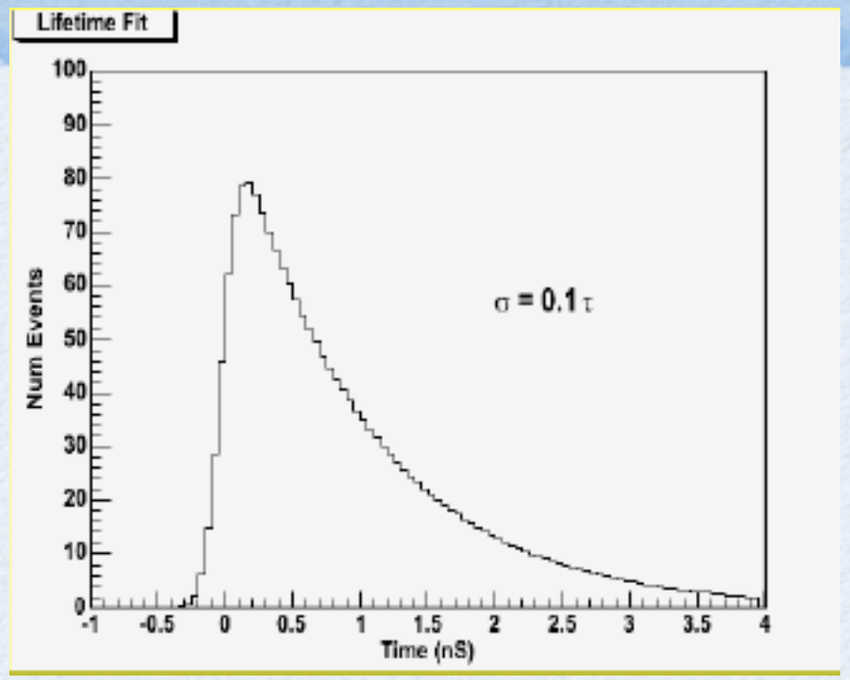
$$= \frac{1}{\tau} e^{-t/\tau} e^{-\frac{\sigma^2}{2\tau^2}} \frac{1}{\sqrt{2\pi}\sigma} \int_0^\infty e^{-\frac{(t' - (t - \frac{\sigma^2}{\tau}))^2}{2\sigma^2}} dt'$$

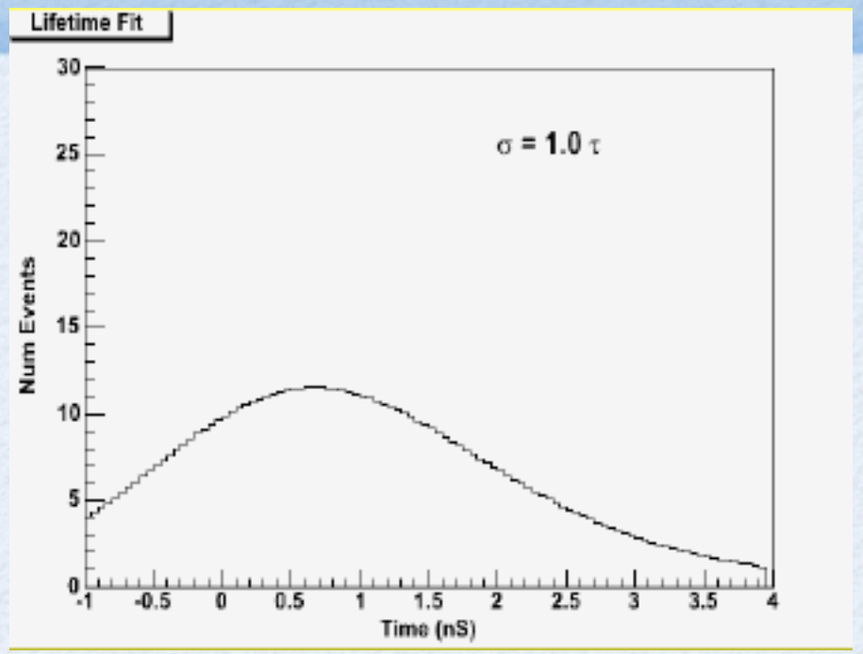
$$= \frac{1}{\tau} e^{-t/\tau} e^{-\frac{\sigma^2}{2\tau^2}} \frac{1}{\sqrt{2\pi}\sigma} \int_{-(t - \frac{\sigma^2}{\tau})}^\infty e^{-\frac{t''^2}{2\sigma^2}} dt'$$

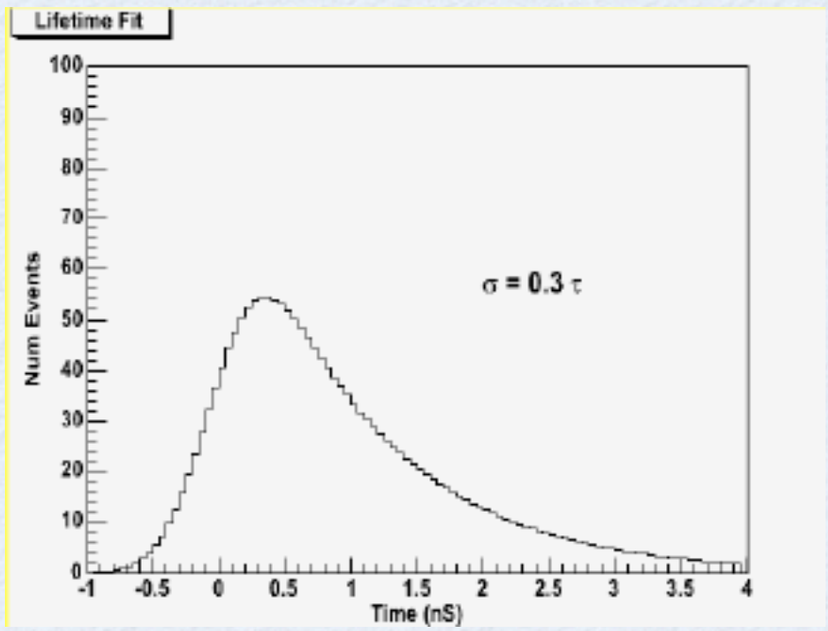
$$= \frac{1}{\tau} e^{-t/\tau} e^{-\frac{\sigma^2}{2\tau^2}} \frac{1}{\sqrt{\pi}} \int_{-\frac{1}{\sqrt{2}}(\frac{t}{\sigma} - \frac{\sigma}{\tau})}^\infty e^{-x^2} dx$$

$$= \frac{1}{\tau} e^{-t/\tau} e^{-\frac{\sigma^2}{2\tau^2}} \frac{1}{2} erfc(-\frac{1}{\sqrt{2}}(\frac{t}{\sigma} - \frac{\sigma}{\tau}))$$

FOR VARIOUS WIDTHS







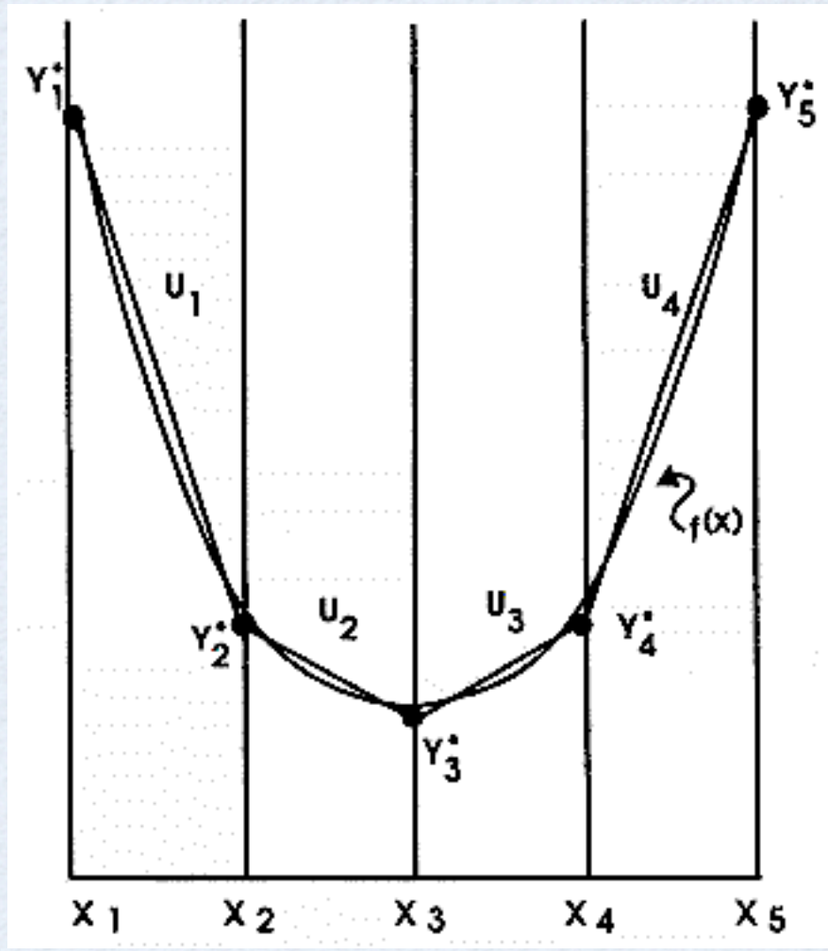
- Larger our resolution the more Gaussian it looks

Combining Measurements

- The maximum likelihood method can be used to combine results from two experiments
  - Results from Higgs searches share likelihood curves
  - Simply make a combine likelihood by multiplying the two

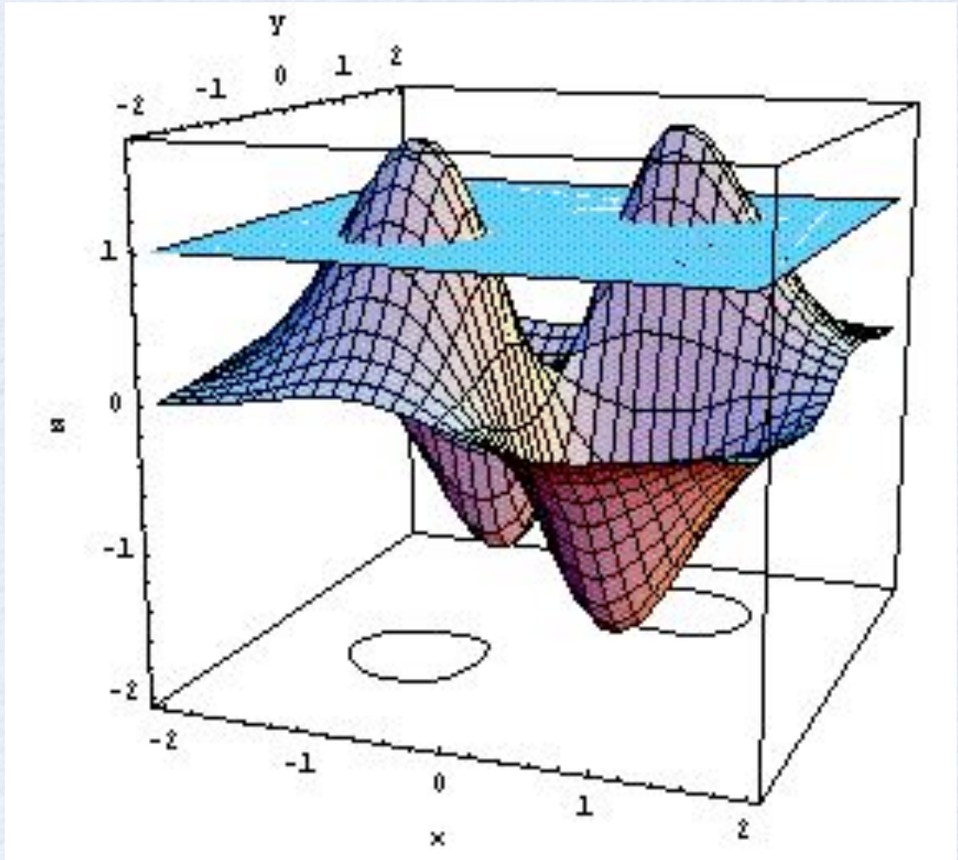
SINGLE PARAMETER

- For a single parameter you can just scan through the full range methodically
- Make a graph
- Difficult to make a mistake and if you do easy to figure it out



MANY PARAMETERS

- Parameter space becomes MUCH more complex.
- Can get stuck in local minima and converge on the wrong answer!



FORTUNATELY

- For complicated cases their is a small industry of tools
- I would suggest using them
- We will discuss some in the next week...