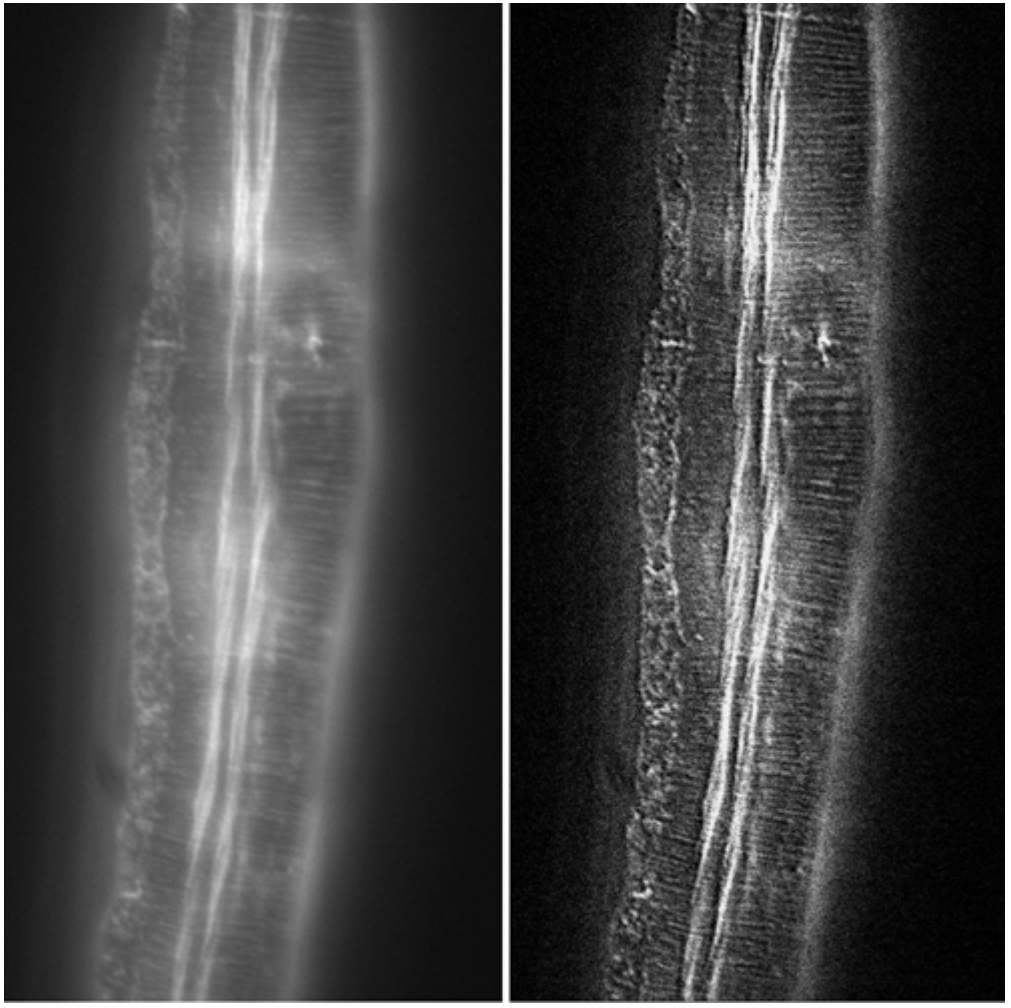
Physics 509: Deconvolution & Unfolding

Scott Oser
Lecture #21

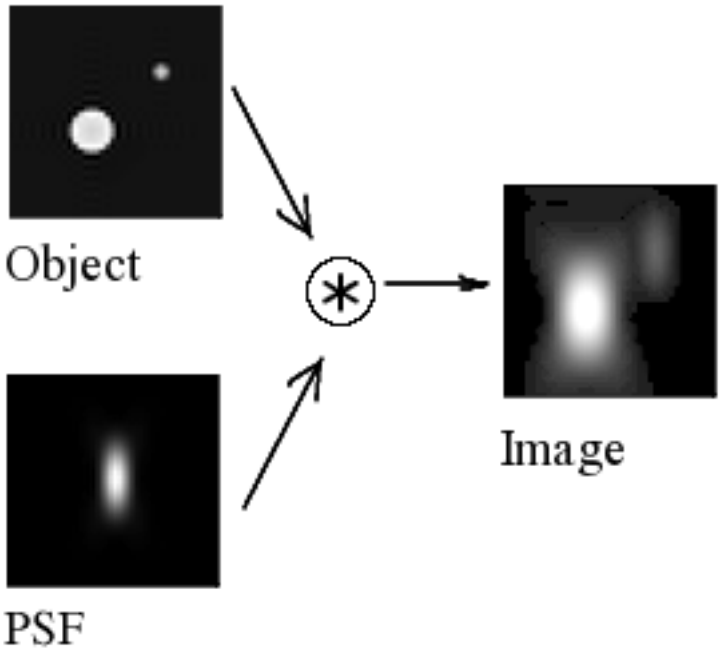


The smearing problem

Measurements generally have errors intrinsic to the apparatus or experimental technique. These errors "smear out" the quantities we're measuring.

An example is the point spread function of a telescope. Due to the finite resolution of lenses, optics, CCDs, etc., a true "point source" of light will produce an extended blob in the image.

Given the final smeared image, can you "unsmear" it to recover the original?



When not to deconvolve

This “unsmearing” procedure is often called “deconvolution” or “unfolding”.

The most important advice I can give about deconvolution is “Don't”.

It's a lot of work, and often produces biased or otherwise unsatisfactory results. Moreover it's often unnecessary.

For example, if you want to compare your measurements to a theory, you could either:

1) deconvolve the effects of the measurement apparatus, and compare the deconvolved result directly with the theory, OR
2) take the theory, convolve it with the response function of the measurement apparatus, then compare the convolved prediction directly with the data. This is called “forward fitting”, and is usually much easier!

When you may have to deconvolve

Sometimes you have to deconvolve anyway ...

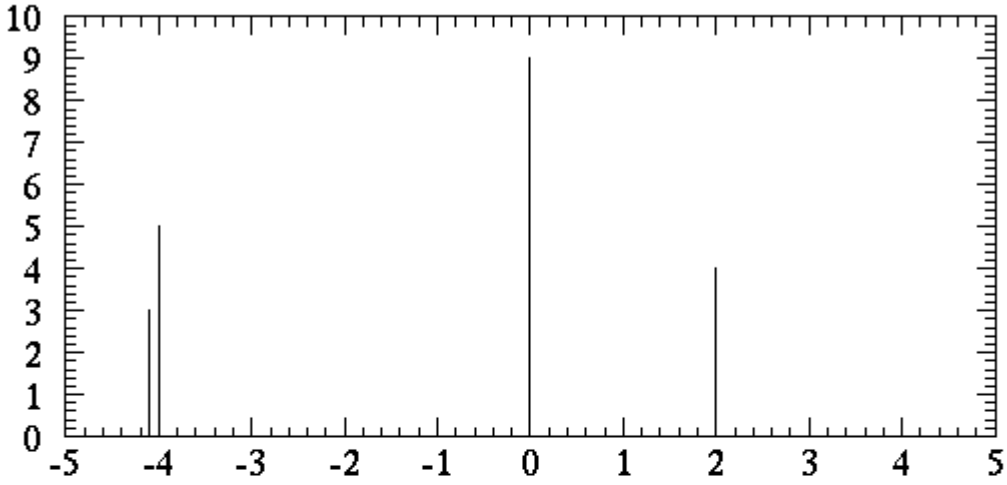
1) You want to compare your measurement with the result from another experiment. The two experiments have different response functions, so to do a head-to-head comparison you have to correct for the measurement effects.

2) You want your data to be easily reanalyzed in 100 years, when no one is alive who remembers what your experiment's response function even looked like.

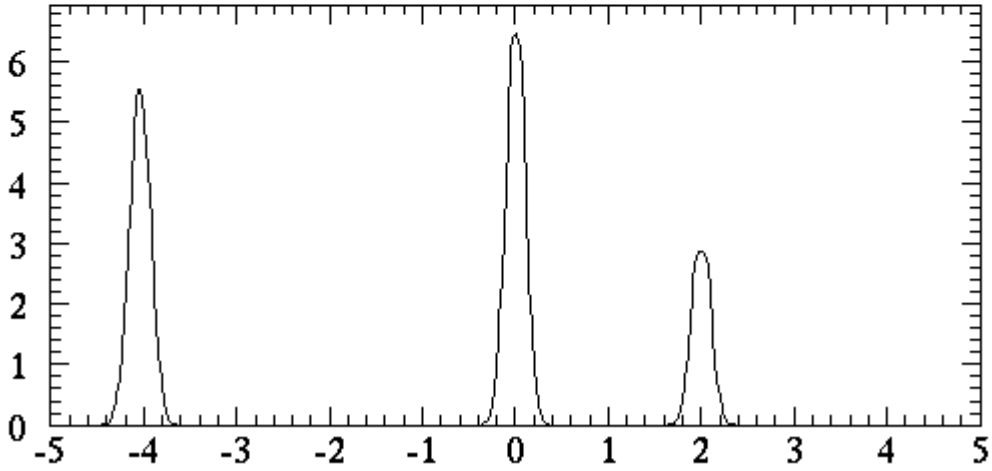
3) You want your result to be in a “human-friendly” format. For example, it's easier to try to unblur a photo and recognize a face than it is to blur a photo of someone you know and try to match it to an image.

The matrix inversion method

Consider some distribution f(y) that we want to estimate. (For example, it could be the true frequency spectrum from a telescope.)



The spectrometer has finite resolution, so that a monochromatic line produces a finite spectrum of some width.



Set up a "response function matrix"

Evidently a count in bin *i* results in signals in multiple bins. There is some response function s(x|y) which tells us the size of the measured signal at x given the presence of a signal at y. Note that

$$\int s(x|y)\,dx = 1$$

We call s(x|y) the "resolution function"/"point spread function"/ "response function".

Although s(x|y) may be continuous, it's easiest to formulate this problem with binned data. The expected amount of signal to be observed in the i^{th} bin is given by

$$\nu_i = \int dy\, f(y) \int_{\text{bin}\,i} dx\, s(x|y)$$

Set up a “response function matrix”

$$\nu_i = \int dy\, f(y) \int_{\text{bin}\, i} dx\, s(x|y)$$

Recast this as a sum over bins in y:

$$\mu_j \equiv \int_{\text{bin}\, j} dy\, f(y)$$

$$\nu_i = \sum_j \frac{\int_{\text{bin}\ j} dy\, f(y) \int_{\text{bin}\, i} dx\, s(x|y)\, \mu_j}{\mu_j}$$

$$= \sum_j R_{ij} \mu_j$$

where

$$R_{ij} \equiv \frac{\int_{\text{bin}\ j} dy\, f(y) \int_{\text{bin}\, i} dx\, s(x|y)}{\int_{\text{bin}\, j} dy\, f(y)}$$

Set up a "response function matrix"

$$\mu_j \equiv \int_{\text{bin } j} dy\, f(y) \qquad \nu_i = \sum_j R_{ij}\,\mu_j \qquad R_{ij} \equiv \frac{\int_{\text{bin } j} dy\, f(y) \int_{\text{bin } i} dx\, s(x|y)}{\int_{\text{bin } j} dy\, f(y)}$$

This formalism relates the expected observed signal size ν_i in the i^{th} bin to the true signal strength μ_j in the j^{th} bin. The idea is that we'll measure the signal size, invert the matrix R, and get the true signal strength μ_j in each bin.

The matrix elements R_{ij} do depend on f(y), which is unknown. But if the bins are narrow enough that we can approximate f(y)=constant across the bin, then f(y) cancels out from the ratio. Or we can put in our best prior estimate for f(y) when we calculate matrix R.

Matrix R can be calculated either from Monte Carlo, direct evaluation of the integral, or measured with calibration sources.

Inverting the response function matrix

$$\nu_i = \sum_j R_{ij}\,\mu_j$$

In this expression, we can have $1 \le i \le N$ and $1 \le j \le M$. (You can have more bins in the observed spectrum than you're fitting for, or vice versa.) But if we consider a square matrix, then we may be able to invert R. (Whether R is invertible depends on the details of s(x|y).)

The above relation is an exact relation giving the expected signal in each bin, given the true underlying distribution μ. Suppose we actually go measure the strength n_i in each bin. An obvious estimator for the true distribution is then:

$$\vec{\mu} = R^{-1}\vec{n}$$

The matrix method

Consider the following response function operating on binned data:

- 36% of the signal stays in its original bin
- 44% of the signal spills into an adjoining bin (22% each way)
- 20% of the signal spills into two bins (10% in each direction)

		10%
		22%
100%	→	36%
		22%
		10%

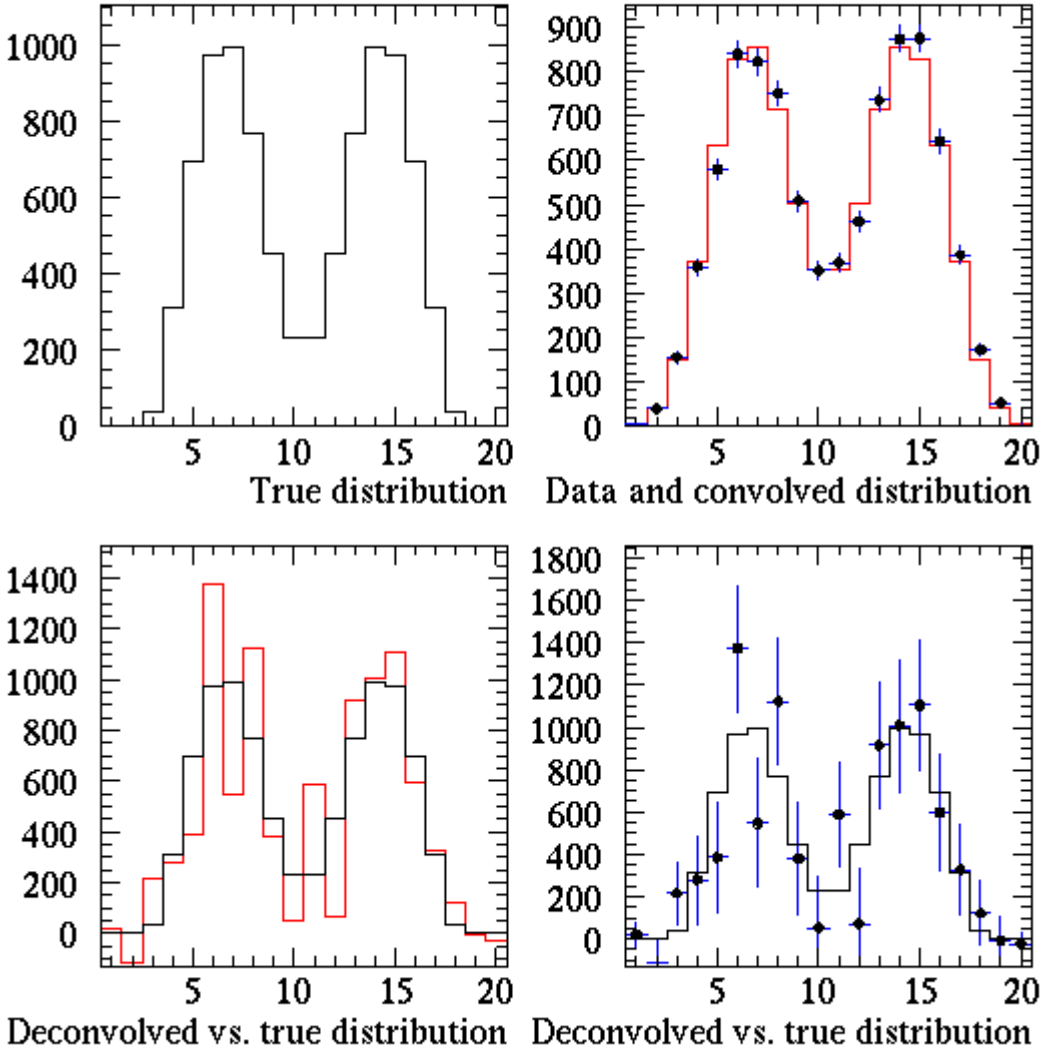
The matrix method

Consider the true distribution (upper left).

Smearing it gives the distribution in the upper right. Points are data drawn from this distribution. Each bin has ~3-10% accuracy.

Bottom left compares deconvolved data from matrix method with true distribution---ugh!

Bottom right shows error bars---very big!

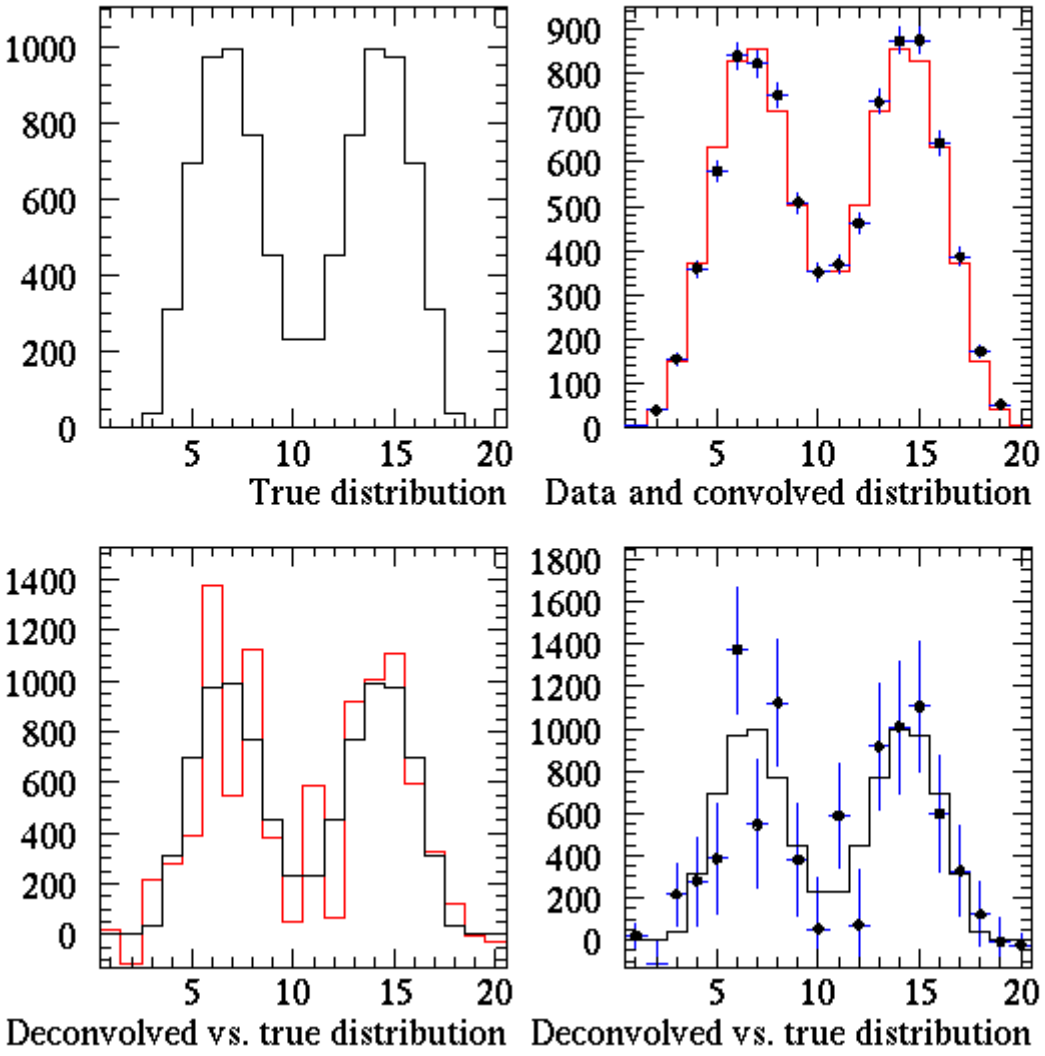


What a nasty result!

The deconvolved data looks very little like the true distribution (see bottom left---big zig zags!)

When error bars are added, the deconvolved data at least looks consistent with the true distribution, but the error bars are much bigger than the 3-10% per bin we had on the raw data!

What's going wrong?



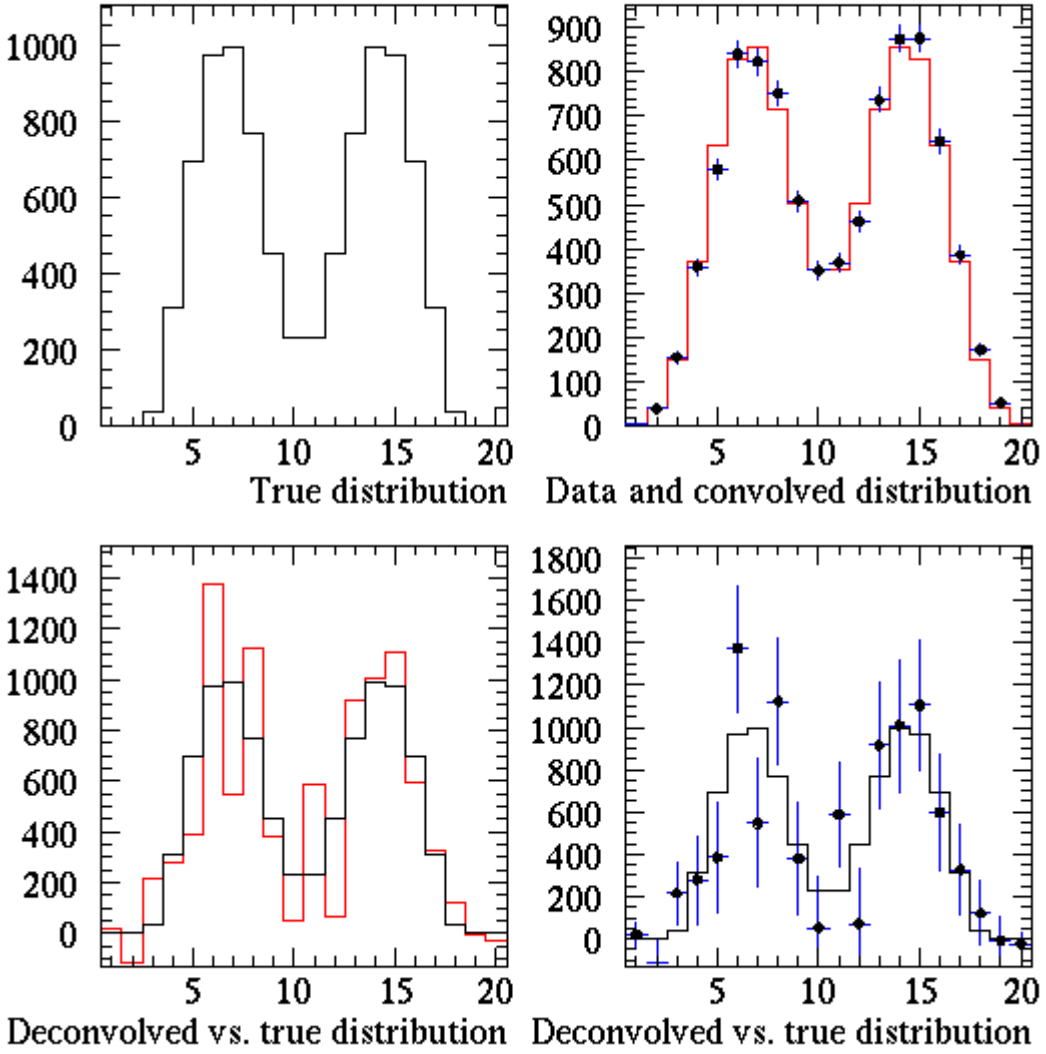
Nothing is wrong ... nothing to see here ... move along, people .

Nasty facts:

1) Deconvolution by the matrix method “works”---if applied to the expected convolved distribution, we get back exactly the true distribution.

2) But the procedure amplifies noise! The 3-10% random scatter in the data gets amplified into the ~20% errors on individual bins from the deconvolved data.

3) Underlying problem is that smearing washes out sharp peaks---so “unsmearing” has to create sharp peaks. Any small random fluctuations get blown up.



The matrix inversion method is unbiased and has minimal variance

These sound like nice properties! Remember that

$$\vec{\mu} = R^{-1}\vec{n}$$

So to calculate the covariance matrix of μ we just calculate:

$$\mathrm{cov}(\mu_i, \mu_j) = \mathrm{cov}\left(\sum_k (R^{-1})_{ik} n_k, \sum_l (R^{-1})_{jl} n_l\right)$$

$$\mathrm{cov}(\mu_i, \mu_j) = \sum_{k,l} (R^{-1})_{ik} (R^{-1})_{jl} \mathrm{cov}(n_k, n_l)$$

Often $\mathrm{cov}(n_k, n_l) = \delta_{kl}$.

The off-diagonal covariances may be large, and often alternate signs.
The reason why the errors is so big on individual bins is that neighboring bins are negatively correlated. So we get a zig zag pattern in the residuals, and the the sum of several bins will have smaller uncertainties than individual bins do.

(By the way, note that this method is equivalent to ML or least squares fitting.)

What's wrong with the matrix inversion method's result?

It's relatively easy to do, unbiased, and has the minimum possible variance. It's actually a ML estimator! What's not to like?

The result isn't wrong, just not very desirable. Particularly the covariances between bins are large, and the result just doesn't look much like the true original distribution.

To do better we need to add more constraints, such as a smoothness criterion.

The price we pay will be to produce a bias in the estimator. Basically what we want to do is to create some biased estimators, incorporating our prior beliefs of what the correct answer looks like (such as assumptions about how smooth the answer should be), that will happen to have an acceptably small bias if our assumptions are correct. We trade bias for smoothness.

Method of correction factors

Here's a very simple alternative to the matrix inversion method:

$$\hat{\mu}_i = C_i n_i$$

where the C_i are "correction factors" that convert from measured values to the true values in each bin.

You calculate the correction factors using Monte Carlo and some assumed true distribution:

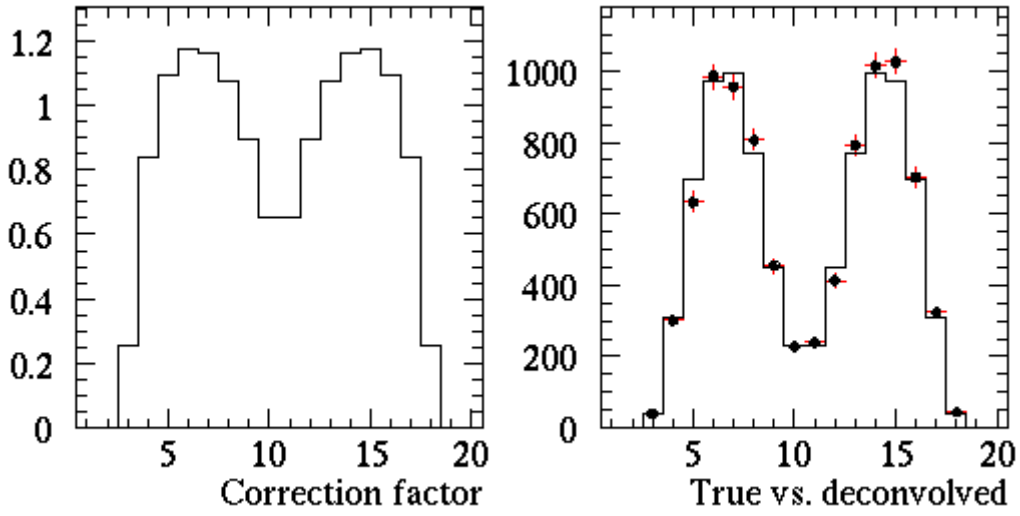
$$C_i = \frac{\mu_i^{MC}}{\nu_i^{MC}}$$

This is nothing other than the ratio of the true signal strength μ in a bin to the measured signal strength ν.

Results from the method of correction factors

Left: correction factors
Right: comparison of true distribution with result from correction factor method---it looks great!

But there's a cheat ...



We used the true distribution to calculate the correction factors. This introduces a bias:

$$b_i = \left(\frac{\mu_i^{MC}}{\nu_i^{MC}} - \frac{\mu_i}{\nu_i^{sig}} \right) \nu_i^{sig}$$

The direction of the bias is usually to pull the answer towards the model prediction. So there is significant model dependency. Imagine what result you'd get if applying this to a flat distribution!

Regularized unfolding

The “matrix inversion” method is actually just the maximum likelihood method. We were unhappy that it amplified noise, but it does actually maximize the likelihood.

We'd be willing to accept a “worse” fit to the data (as measured by the likelihood) provided that it is “smoother” by some criteria we will need to specify. Define some “smoothness function” S(μ), that gets bigger when the extracted distribution becomes smoother. We want to find the solution that maximizes S(μ) while still giving a “reasonable” fit:

$$\ln L(\vec{\mu}) \geq \ln L_{max} - \Delta \ln L$$

We choose the parameter Δ ln L based on how far we're willing to allow the solution to vary from the best fit.

A trade-off

We want to maximize S(μ) while still giving a "reasonable" fit to the data---that is, satisfy this constraint:

$$\ln L(\vec{\mu}) \geq \ln L_{max} - \Delta \ln L$$

Using variational calculus you can show that this is equivalent to maximizing:

$$\Phi(\vec{\mu}) = \alpha \ln L_{max} + S(\vec{\mu})$$

The parameter α ln L depends on Δ ln L. If α=0, we ignore the data entirely and just report whichever distribution has maximal smoothness. If $\alpha \to \infty$ then we ignore the smoothness critierion and just get back the maximum likelihood result.

Our job: choose a good ***regularization function*** S(μ), and decide how much you're willing to let the smoothed version vary from the best fit.

Tikhonov regularization

What is a good regularization function? (How do you define "smoothness"?)

Tikhonov regularization judges smoothness based on the derivative of the deconvolved distribution:

$$S[f] = -\int dx \left(\frac{d^k f}{dx^k} \right)^2$$

Here f(x) is the deconvolved distribution. The minus sign makes this term big when the derivative is small.

With k=2 we generally favor linear PDFs, while penalizing curvature.

You can use any value of k, or linear combinations of different order derivatives.

MaxEnt regularization

What if we use Shannon's entropy measure as the regularization function?

$$S = -\sum_{i=1}^{N} \frac{\mu_i}{\mu_{tot}} \ln \frac{\mu_i}{\mu_{tot}}$$

Using this form of regularization yields the distribution with the maximum entropy that is consistent with whatever tolerance you've placed on the deviation from the best-fit point.

The entropy is actually related to how many ways there are to distribute M objects over N different bins:

$$\Omega(\vec{\mu}) = \frac{\mu_{tot}!}{\mu_1! \mu_2! \ldots \mu_N!}$$

+ lots of algebra =

$$\ln \Omega(\vec{\mu}) \approx \mu_{tot} S(\vec{\mu})$$

Connection of MaxEnt with Bayesian methods

By Bayes theorem we estimate:

$$f(\vec{\mu}|\vec{n}) \propto L(\vec{n}|\vec{\mu})\, p(\vec{\mu})$$

If we use a maximum entropy prior for p(μ), we have

$$p(\vec{\mu}) = \Omega(\vec{\mu}) \approx \exp(\mu_{tot} S(\vec{\mu}))$$

The peak of the Bayesian posterior distribution is that which maximizes:

$$\ln f(\vec{\mu}|\vec{n}) \propto \ln L(\vec{n}|\vec{\mu}) + \ln p(\vec{\mu})$$

$$\ln f(\vec{\mu}|\vec{n}) \propto \ln L(\vec{n}|\vec{\mu}) + \mu_{tot} S(\vec{\mu})$$

This is just regularization with entropy, and with the constant α that controls how much we're allowed to deviate from best fit equal to $1/\mu_{tot}$.

Cross-entropy regularization

The MaxEnt regularization favors a deconvolved distribution that is flatter (closer to uniform).

If we really have no prior beliefs about the true distribution, or if we believe the distribution is likely to be pretty flat, this is probably the right thing to do.

But if you think the most likely true distribution isn't flat, but has some shape q(μ), then we may want to use cross-entropy instead:

$$K(f;q) = -\sum_{i=1}^{M} f_i \ln \frac{f_i}{M\, q_i}$$

If the reference distribution q is flat (q_i =1/M), this reduces to the regular entropy.

Choosing the balance: bias vs. variance

We started with:

$$\Phi(\vec{\mu}) = \alpha \ln L_{max} + S(\vec{\mu})$$

and attempted to maximize this given some value of α (which determines how much weight we give to smoothness vs. getting the best fit).The values of μ_i that maximize this are our smoothed estimate.

The estimators depend on the data, and so themselves are random variables with non-zero variances:

$$\hat{\vec{\mu}} = \hat{\vec{\mu}}(\vec{n})$$

They also have some bias as a result of the smoothing. To know the exact bias you need to know the true distribution. But if we assume that the μ values we got for our data equal the true distribution, we can estimate the expected bias, averaging over hypothetical data sets:

$$b_i = E(\mu_i) - \mu_i \approx E(\hat{\mu}_i) - \hat{\mu}_i$$

Choosing the balance: bias vs. variance 2

How do we pick α?

$$\Phi(\vec{\mu}) = \alpha \ln L_{max} + S(\vec{\mu})$$

We pick some criterion, such as (1) minimizing the mean squared error:

$$\text{mean squared error} = \frac{1}{M}\sum_{i=1}^{M}\left(\text{Var}(\mu_i) + \hat{b}_i^2\right)$$

(2) Or instead we may choose to allow each bin to contribute ~1 to the χ^2.

$$2\Delta \ln L = 2(\ln L_{max} - \ln L) = N$$

(3) Or we look at the estimates of the biases and their variances:

$$\chi_b^2 = \sum_{i=1}^{M} \frac{\hat{b}_i^2}{Var(\hat{b}_i)}$$

We set this equal to M, the number of bins fitted for---basically we require the average bias in each bin to be 1σ from zero.

Which of these should you choose?

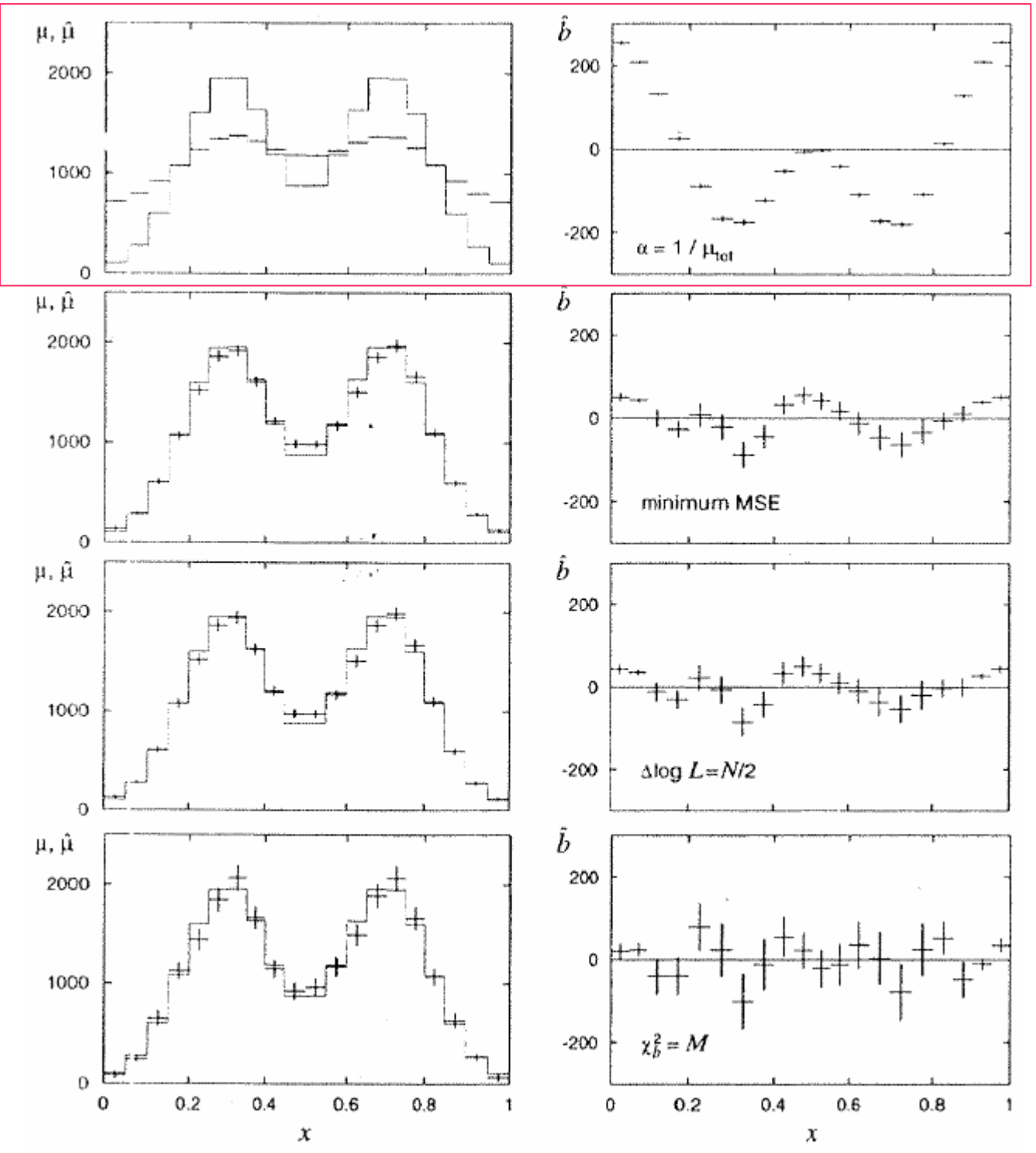
Damned if I know.

Probably there's no universal solution. Depending on your application you may be more concerned about bias or more concerned about reducing the variances.

Bias=0 gives huge variances (this are just the ML estimators)

You can get the variance to equal zero by letting α=0 and ignoring your data completely, which also seems stupid.

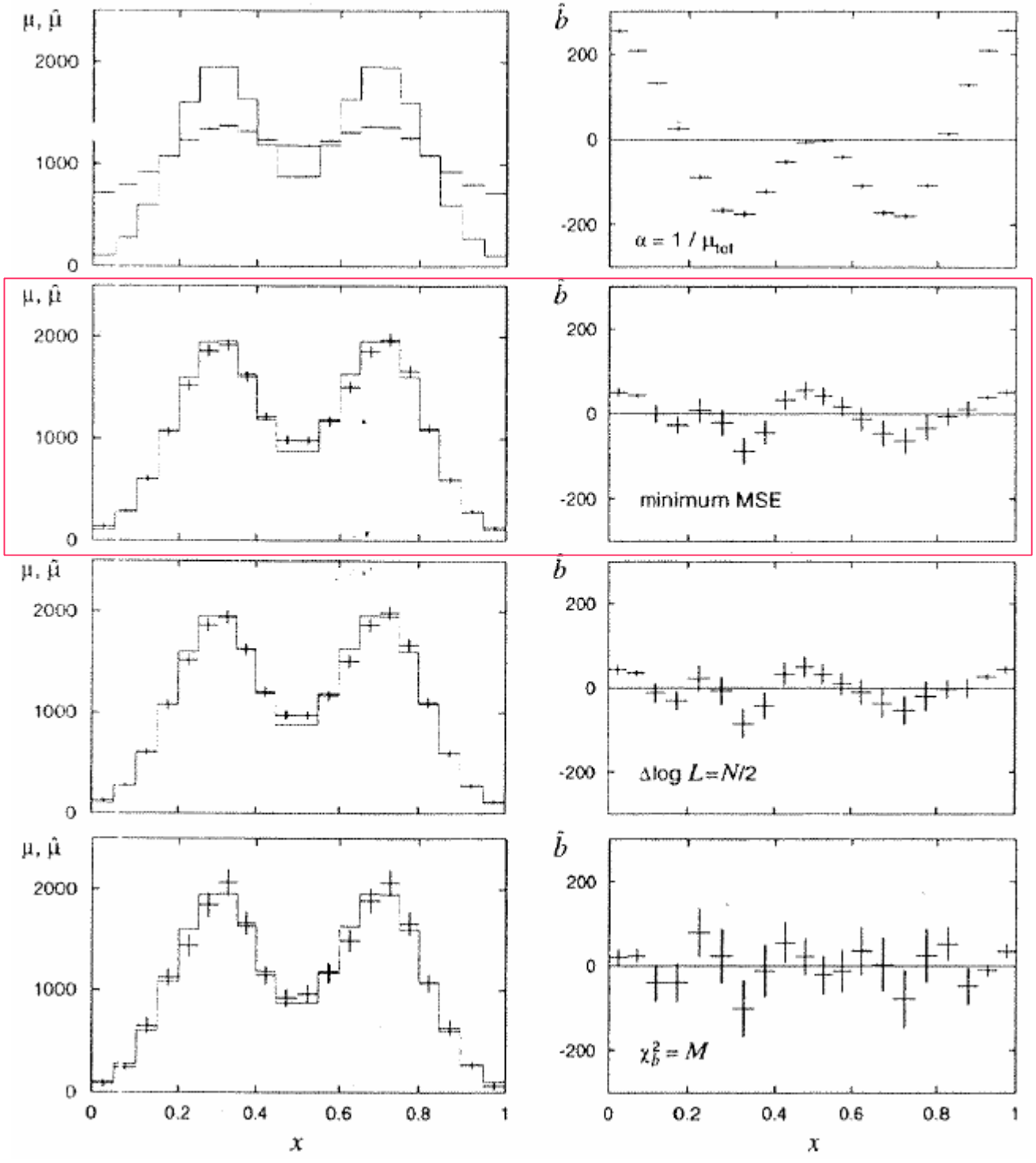
Caveat emptor.



Top left: true and deconvolved distribution using Bayesian prescription (using maximum entropy prior with α fixed by Bayes.)

Top right: bias estimated from the data itself

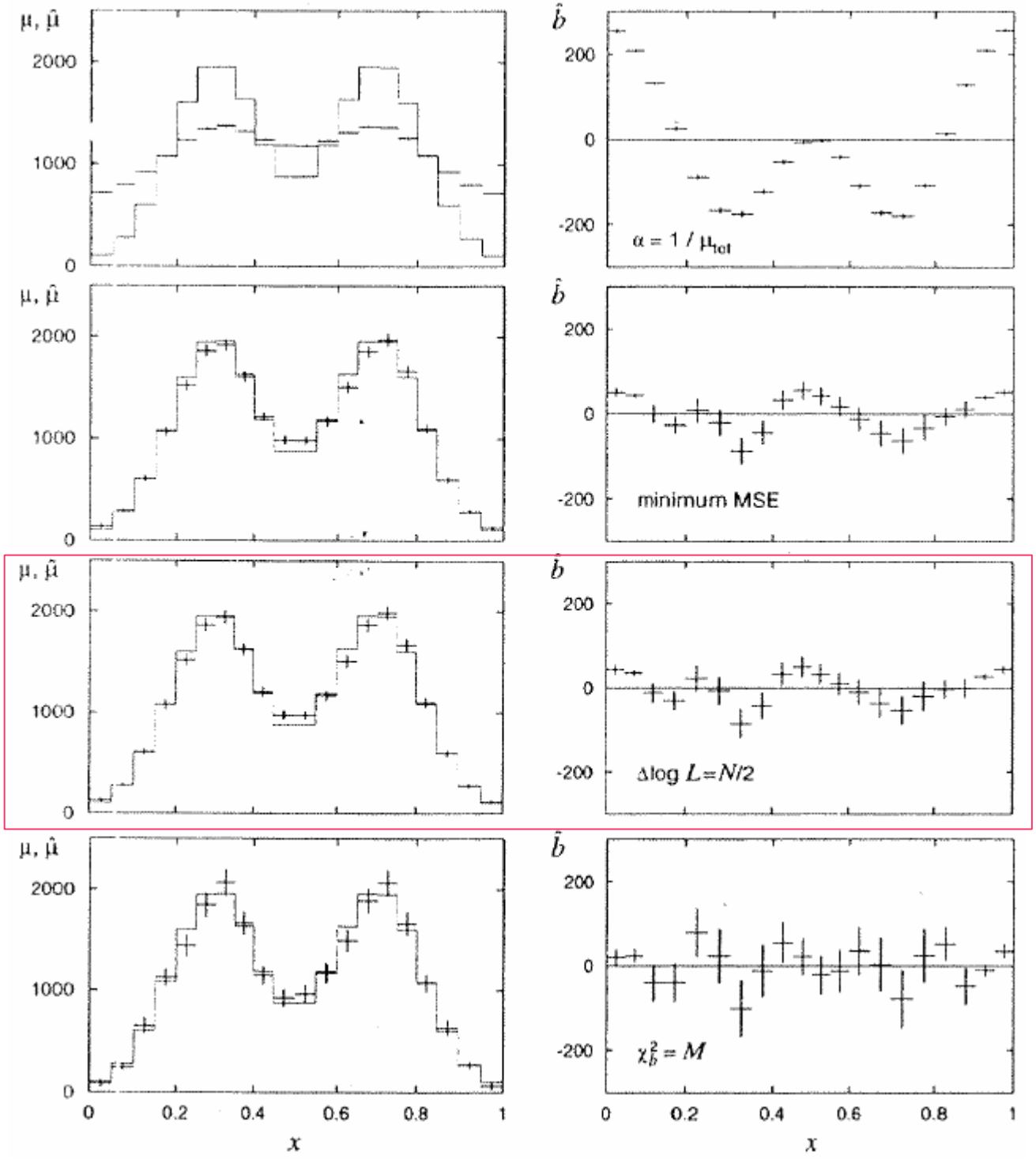
The bias is quite significant---the raw MaxEnt method here favors flat distributions. For the true distribution it doesn't work well. But we only know that in hindsight---if you had prior knowledge that the true distribution wasn't flat, you wouldn't use MaxEnt!



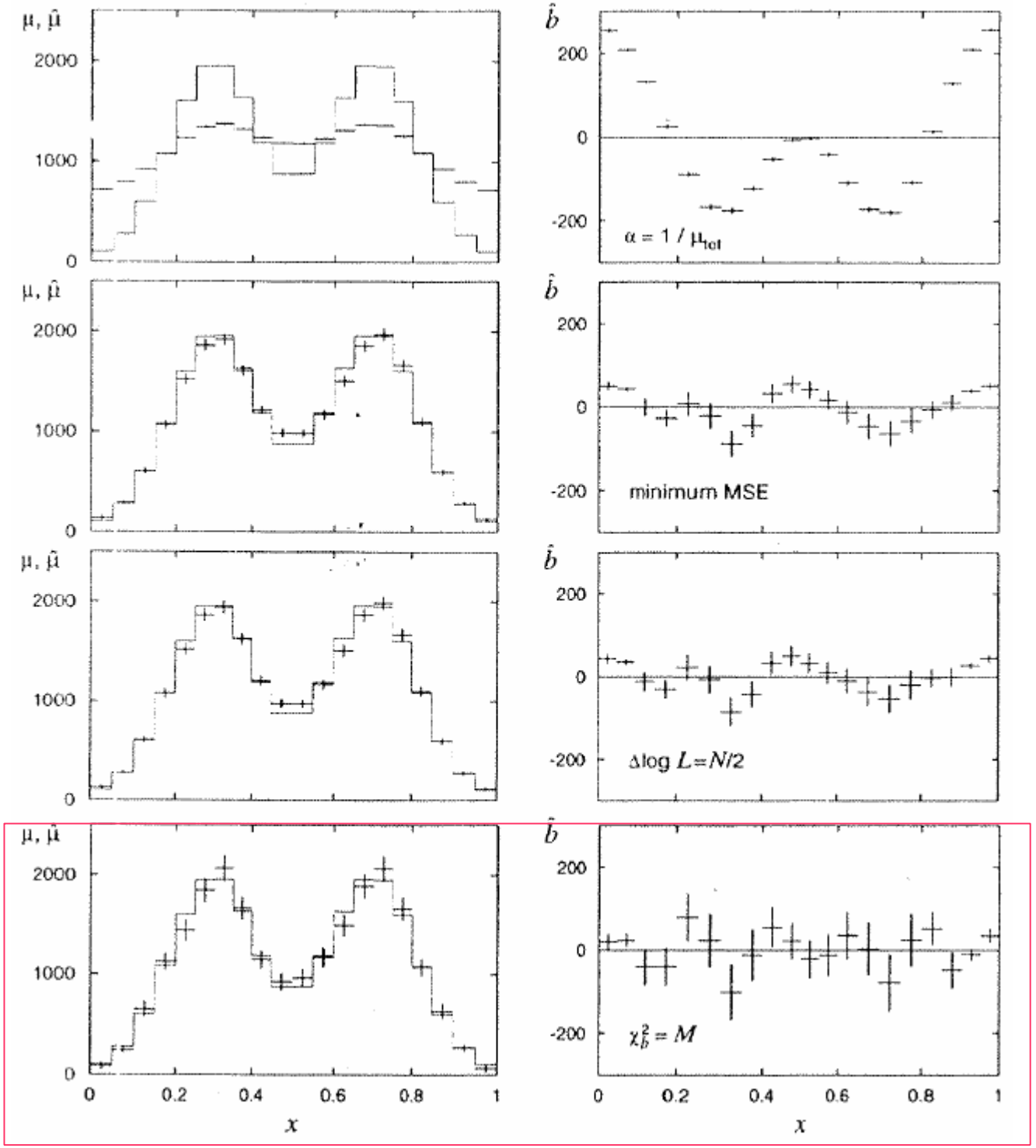
Minimizing the mean squared error gives the second row.

Biases are now much smaller, but inconsistent with zero.

Variance of estimators is much bigger than before.



Third row: smaller biases still, but generally similar to the minimum squared error method.



Bottom row: putting the estimated typical bias in a bin to be 1σ away from zero.

Biases are all very nice, but of course variances are also bigger (compare the error bars on the top row with those on the bottom).

Parting words on deconvolving

Avoid deconvolving.

If you must, read the extensive literature. I've only given you a taste of what's out there. A good starting point is “Statistical Data Analysis”, by Glen Cowan. The latter examples are taken from there.

If you have meaningful prior information of any sort, consider a Bayesian analysis.