Lecture 2

Linear least squares, intro to non-linear least-squares

Poisson Example

- You've discovered a new object! Your theorist friend has a model, and thinks it will flare randomly, with a mean rate of once per month.

- You, an observer, think your friend is wrong. How long would you need to observe to rule out their model?

χ^2

- The PDF of a Gaussian is $\exp(-0.5(x-\mu)^2/\sigma^2)/\mathrm{sqrt}(2\pi\sigma^2)$ with mean μ and standard deviation σ.

- If we have a bunch of data points, which may have different means and standard deviations, then the joint PDF is the product of the PDFs.

- It is often more convenient to work with the log. For many points, $\log(\mathrm{PDF})=\sum -0.5(x_i-\mu_i)^2/\sigma_i^2 - 0.5*\log(2\pi\sigma_i^2)$

- Usually, we know the variance of our data, and want our model to predict the expected value of x_i, which is μ_i. When we compare models, the second part is constant, so we ditch it. log likelihood becomes: $-0.5\sum(x_i-\mu_i)^2/\sigma_i^2$.

- $\sum(x_i-\mu_i)^2/\sigma_i^2$. is χ^2. We can find the maximum likelihood model by minimizing χ^2.

Linear least-squares

- Rewrite $\chi 2$ with matrices: $(x-\mu)^T N^{-1}(x-\mu)$ for noise covariance matrix N. If N has diagonal elements σ^2, this is identical to previous.

- Let's take simple case that our model depends linearly on a small number of parameters: $\mu_i=\sum A_{ij}m_j$ for model parameters m and matrix A that transforms to predicted values. In matricese: $\mu=Am$

- One example: x(t) is a polynomial in time. Then $\mu_i=\sum t_i^j c_j$.

- With this parameterization, $\chi^2=(x-Am)^T N^{-1}(x-Am)$

Least Squares: $\chi^2=(x-Am)^T N^{-1}(x-Am)$

- To find best-fitting model, minimize χ^2. Calculus on matrices works like regular calculus, as long as no orders get swapped.
- $\partial\chi^2/\partial m = -A^T N^{-1}(x-Am)+\ldots=0$ (at minimum)
- We can solve for m: $A^T N^{-1} A m = A^T N^{-1} x$. Or, $m=(A^T N^{-1} A)^{-1} A^T N^{-1} x$

Linear Least-Squares

- Recall matrix description of $-2\ln(L)=\chi^2$ is $(d-p)^T N^{-1}(d-p)$ for data values d and model prediction p. If p is true model, then $\langle d \rangle = p$.
- Take case where predicted values depend linearly on a set of model parameters: $p=Am$, so $\langle d \rangle = Am$
- $\chi^2=(d-Am)^T N^{-1}(d-Am)$. What are the dimensions of the various things?
- What values of *m* are the "best fit"?

LLS cont'd

- Maximizing likelihood equivalent to minimizing χ^2. Take gradient w.r.t what?
- $\nabla\chi^2=-A^TN^{-1}(d-Am)=0$. $A^TN^{-1}Am=A^TN^{-1}d$.
- How can you solve this? Could you multiply by various combinations of A^{-1T} and N to get $m=A^{-1}d$?
- Let's say we have two data sets d_1 and d_2 with best-fit solutions m_1 and m_2. What are the best-fit parameters if we fit (d_1+d_2)?

Parameter Errors

- Since we are scientists, we need errors on our parameters. If we subtract the true model from the data, we can look at the covariance of what's left, $\langle mm^T \rangle$ (why not $\langle m^T m \rangle$?)

- Let's show that $m=(A^T N^{-1} A)^{-1} A^T N^{-1} d$ gives $\langle mm^T \rangle = (A^T N^{-1} A)^{-1}$.

- What are the standard deviations of the parameters?

Worked Example

- What is the best-fit mean and error for a set of uncorrelated gaussian variables with same mean but individual errors?

- $A=?$ Show that $A^T N^{-1} A = \sum(\sigma_i^{-2})$, $A^T N^{-1} d = \sum d_i/\sigma_i^2$.

- Define weights $w_i=\sigma_i^{-2}$. Then $m=\sum w_i d_i/\sum w_i$. Variance of our estimator is $1/\sum w_i$.

Worked Example 2

- Let's assume that N is constant and diagonal, and we have a single paramers.

- Show LHS = $\sum(m_i^2/\sigma^2)$

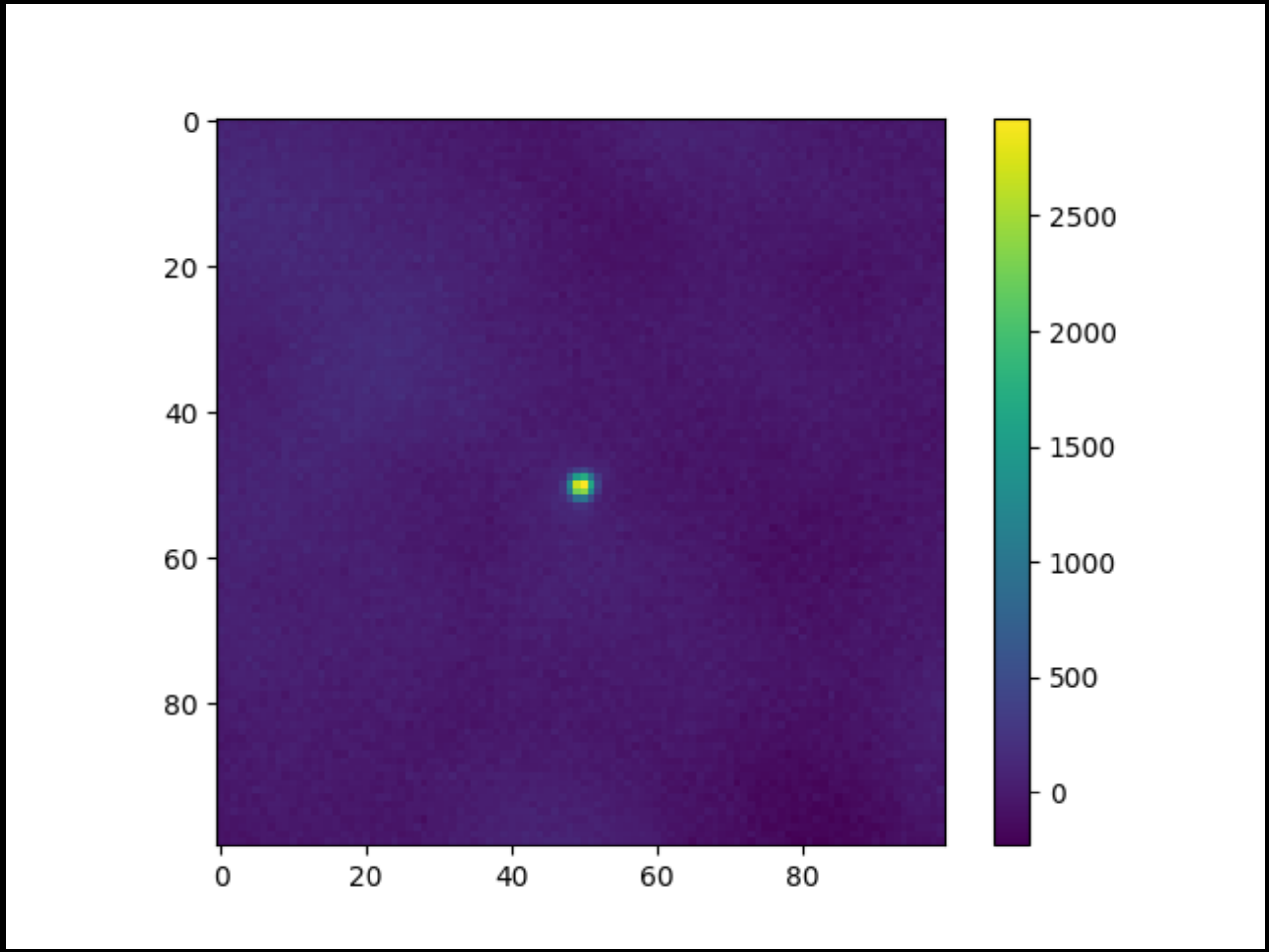
- RHS=$\sum(d_i m_i/\sigma^2)$

- Best-fit amplitude is RHS/LHS = $\sum(d_i m_i)/\sum m_i^2$

- Error=1/sqrt(RHS) = σ/sqrt($\sum m_i^2$). If there are ~n model points with value ~1, this turns into σ/sqrt(n), as roughly expected.

Example - Source in ACT Data

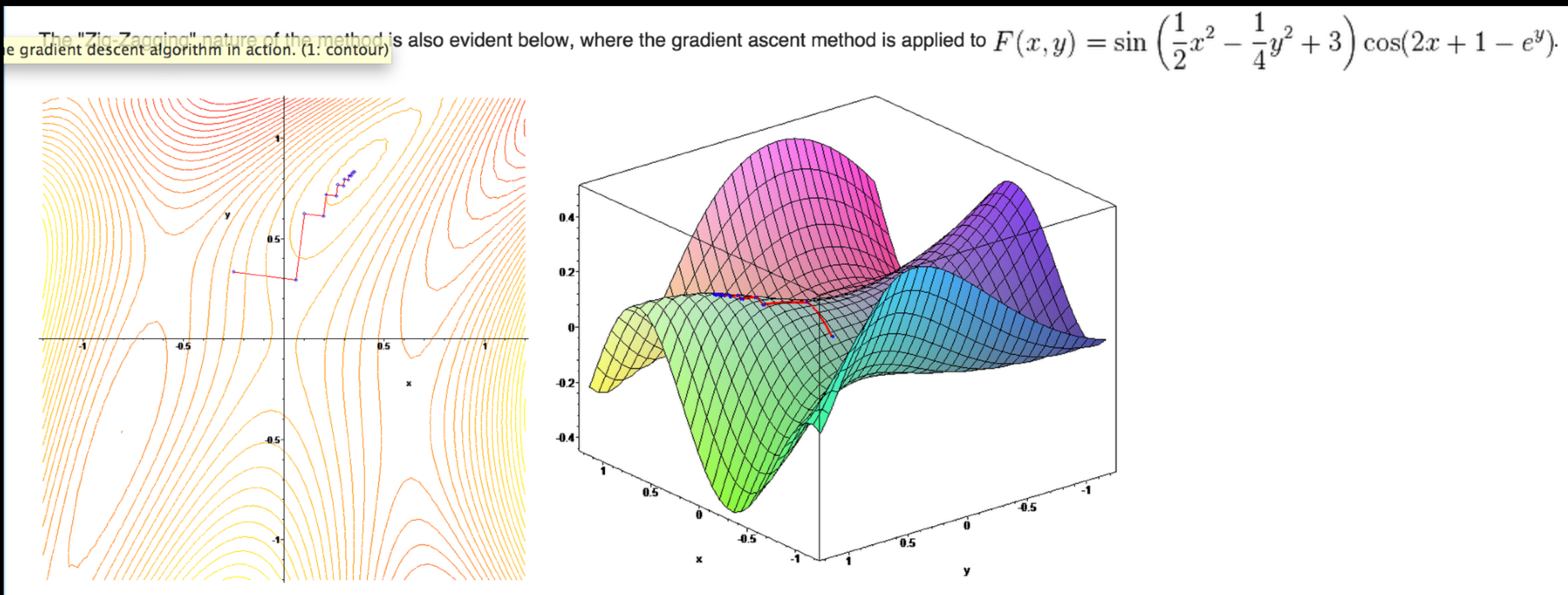
- Let's fit the amplitude of a source in ACT data.
- Look at find_act_flux.py.
- Let's try to guess a Gaussian, fit amplitude to it.
- Map is saved as FITS. Ability to read/write/manipulate FITS images extremely useful in astronomy!



Nonlinear Fitting

- Sometimes data depend non-linearly on model parameters
- Examples are Gaussian and Lorentzian ($a/(b+(x-c)^2)$)
- Often significantly more complicated - cannot reason about global behaviour from local properties. May be multiple local minima
- Many methods reduce to how to efficiently find the "nearest" minimum.
- One possibility - find steepest downhill direction, move to the bottom, repeat until we're happy. Called "steepest descent."
- How might this end badly?

Steepest Descent



From wikipedia. Zigagging is inefficient.

Better: Newton's Method

- linear: $\langle d \rangle = Am$. Nonlinear: $\langle d \rangle = A(m)$ $\chi^2 = (d - A(m))^T N^{-1} (d - A(m))$
- If we're "close" to minimum, can linearize. $A(m) = A(m_0) + \partial A/\partial m * \delta m$
- Now have $\chi^2 = (d - A(m_0) - \partial A/\partial m \, \delta m)^T N^{-1} (d - A(m_0) - \partial A/\partial m \, \delta m)$
- What is the gradient?

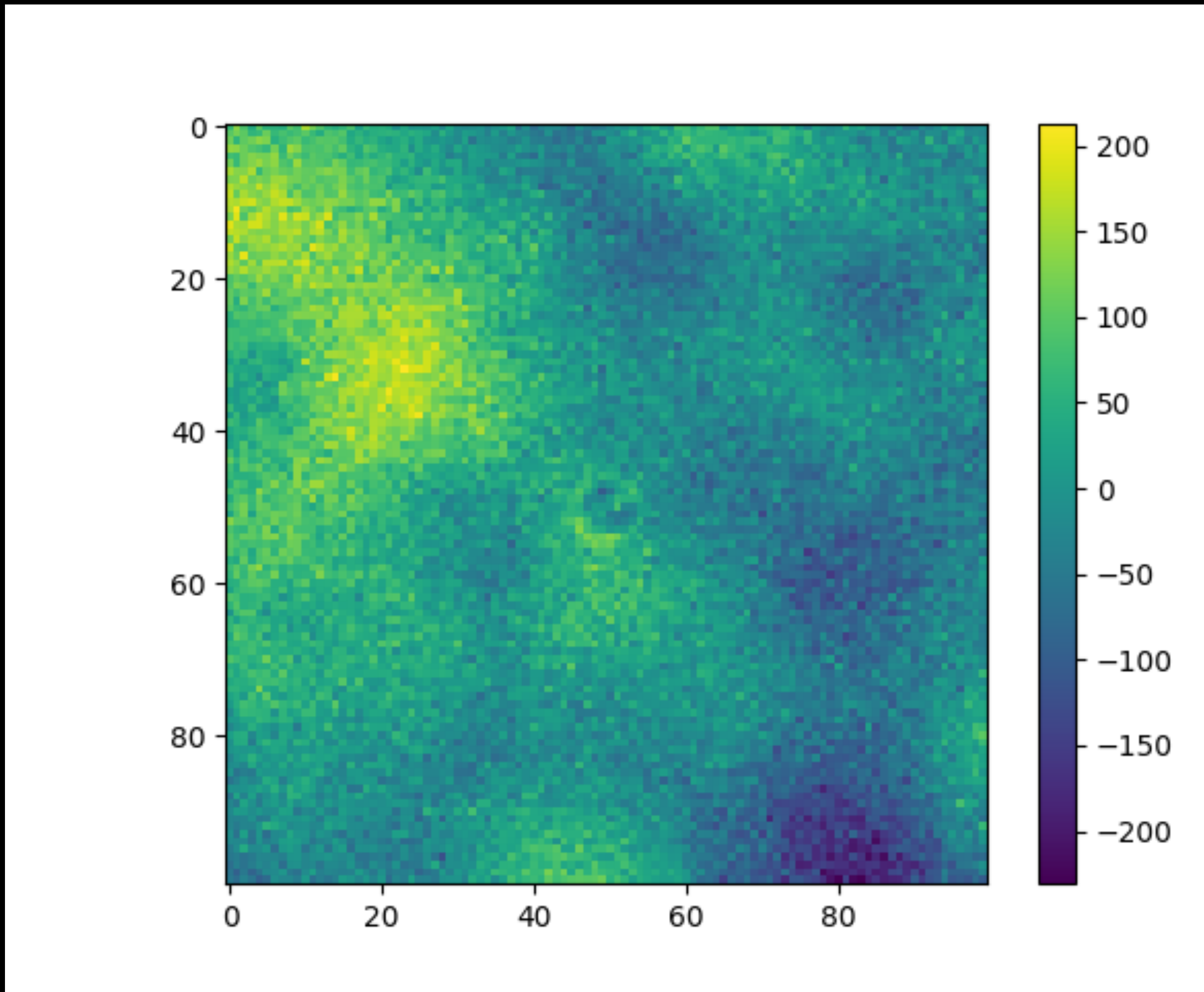
Newton's Method ctd

- Gradient trickier - $\partial A/\partial m$ depends in general on m, so there's a second derivative
- Two terms: $\nabla\chi^2=(-\partial A/\partial m)^T N^{-1}(d-A(m_0)-\partial A/\partial m\ \delta m)-(\partial^2 A/\partial m_i \partial m_j\ \delta m)^T N^{-1}(d-A(m_0)-\partial A/\partial m\ \delta m)$
- If we are near solution $d \approx A(m_0)$ and δm is small, so first term has one small quantity, second has two. Second term in general will be smaller, so usual thing is to drop it.
- Call $\partial A/\partial m$ A_m. Call $d-A(m_0)$ r. Then $\nabla\chi^2 \approx -A_m^T N^{-1}(r-A_m \delta m)$
- We know how to solve this! $A_m^T N^{-1} A_m\ \delta m = A_m^T N^{-1} r$

How to Implement

- Start with a guess for the parameters: m_0.
- Calculate model $A(m_0)$ and local gradient A_m. Gradient can be done analytically, but also often numerically.
- Solve linear system $A_m^T N^{-1} A_m \, \delta m = A_m^T N^{-1} r$
- Set $m_0 \rightarrow m_0 + \delta m$.
- Repeat until δm is "small". For χ^2, change should be << 1 (why?).

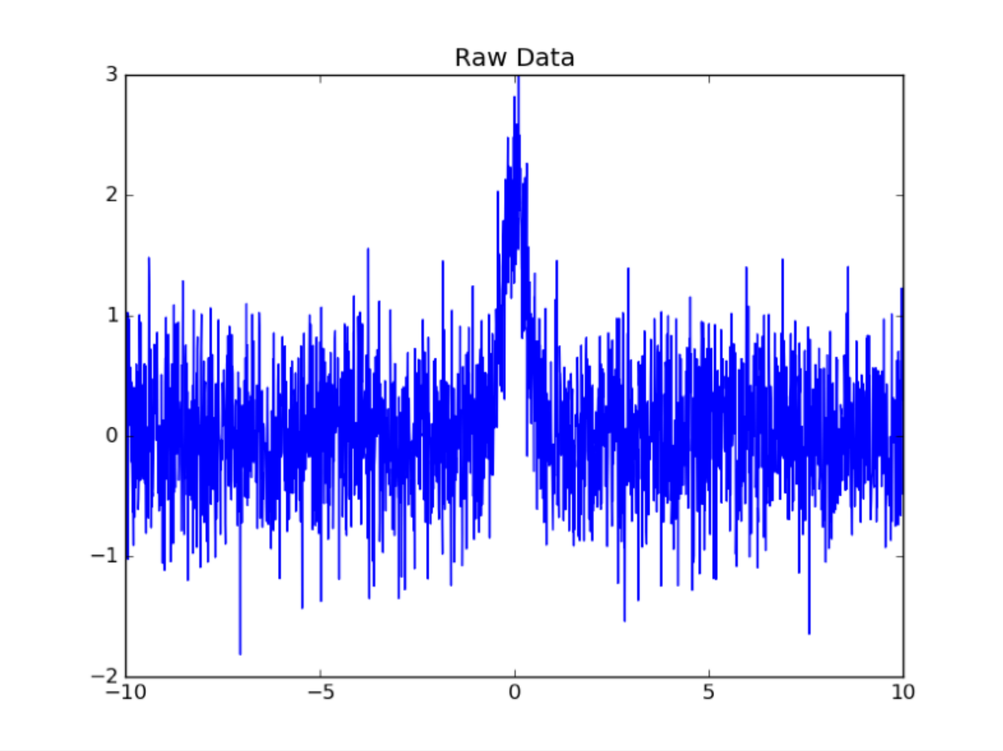
ACT Map Example



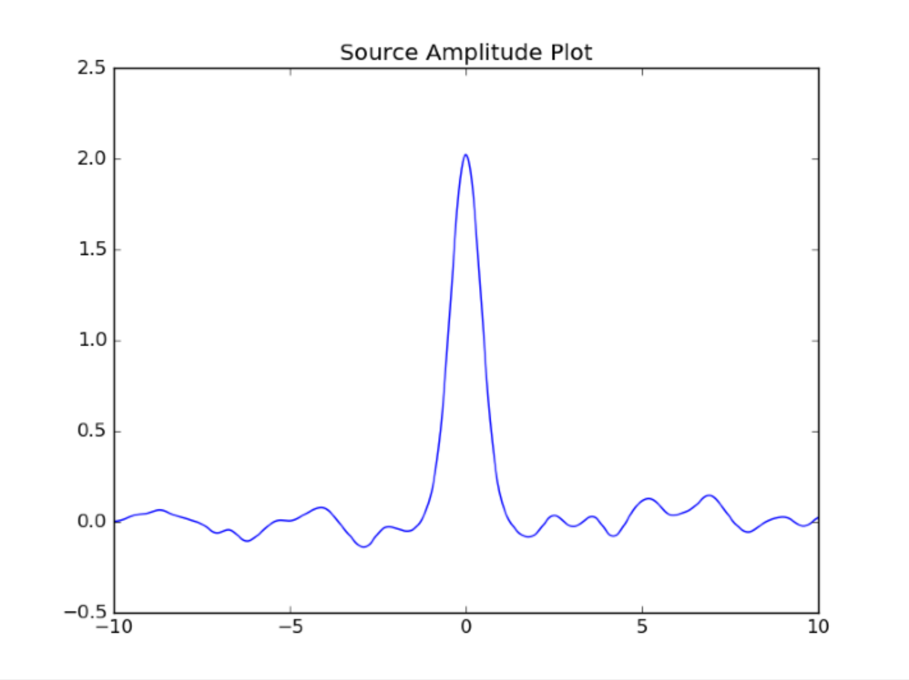
- Look at fit_act_flux_newton.py
- This implements numerical derivatives w/ Newton's method to fit a Gaussian (including sigma, dx, dy) to the ACT data.
- How should we estimate the noise, and hence the parameter uncertainties?

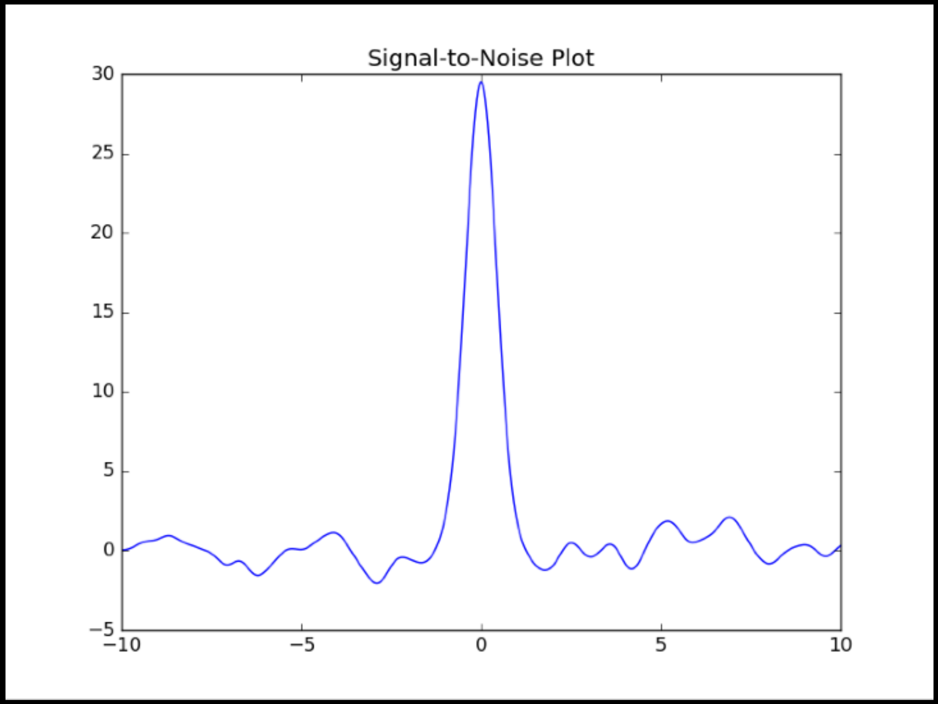
Code Example

```
import numpy
from matplotlib import pyplot as plt

dx=0.01
noise=0.5
Ninv=1.0/noise**2
x=numpy.arange(-10,10,dx)
n=len(x)


x0=0
amp_true=2.0

sig=0.3
template=numpy.exp(-0.5*(x-x0)**2/sig**2)

dat=template*amp_true+numpy.random.randn(n)*noise

snr=numpy.zeros(n)
amp=numpy.zeros(n)
dat_filt=Ninv*dat
denom=(numpy.dot(template,Ninv*template))
rt_denom=numpy.sqrt(denom)
for i in range(n):
    template=numpy.exp(-0.5*(x-x[i])**2/sig**2)
    rhs=numpy.dot(template,dat_filt)
    snr[i]=rhs/rt_denom
    amp[i]=rhs/denom
```




```
plt.clf();
plt.plot(x,snr);
plt.title('Signal-to-Noise Plot')
plt.savefig('snr_plot.png')

plt.clf();
plt.plot(x,amp)
plt.title('Source Amplitude Plot')
plt.savefig('amp_plot.png')

plt.clf();
plt.plot(x,dat)
plt.title('Raw Data')
plt.savefig('dat_raw.png')
```





But wait!

- We took $\sum d(t)a(t-\tau)$. But, this is just the correlation of d with a. We can do this quickly using Fourier transforms.

- Alternatively, let $a^\lozenge=a(-t)$. Then this is $\sum d(t)a^\lozenge(\tau-t)=d\otimes a^\lozenge$. By convolution theorem, this is IFT(FT(d)*FT($a^\lozenge$)).

- However, $FT=\sum f(x)\exp(-2\pi ikx/N)$. $FT(f(-x))=\sum f(-x)\exp(-2\pi ikx/N)=\sum f(x)\exp(2\pi ikx/N)=F^*(k)$. So, our output is just $IFT(FT(d)FT^*(a))$.

- NB - if a is symmetric, when A(k) is real (why?) and we can skip the conjugates.

How about changing noise?

- Often we're searching for things where noise is varying - say, searching for sources in a map that is deeper in the middle.

- Top: $A^T N^{-1} d$ is easy - correlate A with $N^{-1}d$.

- Bottom: $A^T N^{-1} A$ we can correlate N^{-1} with A^2.

- Amplitude stays ratio of the two (bottom is now a vector instead of a scalar), and SNR is top/$\sqrt{}$(bottom)

Matched Filter

- We are beginning on a very widespread class of techniques called *matched filters*.
- In various forms, they show up in GW analysis, photometry in optical images, finding galaxy clusters in CMB maps, finding radar return echos…
- Extendable to non-independent noise, multiple (possibly correlated) simultaneous/multi-frequency datasets, many others.
- All arises from writing down χ^2 and minimizing. Make a habit of this.

Linear Algebraing up χ^2 (from last slides)

- Usual expression is $\sum(x_i-\mu_i)^2/2\sigma_i^2$
- Let N be diagonal matrix with $N_{ii}=\sigma_i^2$.
- Element-wise, $(x-\mu)^T N^{-1}(x-\mu)$ is identically χ^2.
- I can put orthogonal matrices ($V^T=V^{-1}$) in without changing anything: $(x-\mu)^T V^T V N^{-1} V^T V(x-\mu)$.
- In new, rotated coordinates: x->Vx, μ->Vμ, N->VNV^T, χ^2 remains unchanged. Show that expectation of (rotated) x_i noise times x_j noise = (rotated) N_{ij}?

Stationary Noise

- With this plus posted note, we can work out N^{-1} for correlated but stationary noise.
- N^{-1} operator becomes Fourier divide by noise power spectrum transform.
- Numerator becomes $IFT(FT(a)^{*}FT(d)/N^{FT})$

Expectation/Variance

- expectation of a random variable $\langle x \rangle$ = average value of many realizations. Mathematically=$\int x p(x) dx$.
- Variance is scatter (squared) about the mean - $\mathrm{Var}(x)= \langle (x-\langle x \rangle)^2 \rangle$. Show this equals $\langle x^2 \rangle - \langle x \rangle^2$.
- Expectation and variance fully describe a Gaussian random variable.
- Let c be a constant. What is $\langle cx \rangle$? What is $\mathrm{Var}(cx)$?
- Let y be another random variable. What is $\langle x+y \rangle$? What is $\mathrm{Var}(x+y)$?

Covariance

- Let's look at Var(x+y): $<(x+y)^2>-<x+y>^2=<x^2+2xy+y^2>-<x>^2-2<x><y>-<y>^2=<x^2>-<x>^2+2<xy>-2<x><y>+<y^2>-<y>^2$.
- =Var(x)+Var(y)+2Cov(x,y), where covariance defined to be $<xy>-<x><y>$.
- If x and y are uncorrelated, what is the variance of (x+y)? And of (x-y)?

Stability

- Let's fit polynomials (polyfit.py). How did that go? Why?

- Let's ignore N for now, and use SVD of A - $A=USV^T$, where U is orthogonal (and rectangular), S is diagonal, and V is orthogonal (and square).

- $ATA = VSU^TUSV^T=VS^2V^T$. $(ATA)^{-1}=VS^{-2}V^T$, so if an entry of S was very small, it becomes very large.

- By writing out an analytic cancellation, we can get rid of one copy of S, making problem better behaved numerically: $VS^2V^Tm=VSU^Td$. V and S are square and invertible (usually!), so leaves us with $SV^Tm=U^Td$. No squaring of S… or, $m=VS^{-1}U^Td$.

- For polys, real solution is to switch bases to e.g. Legendre, Chebyshev… Good idea to check condition number (ratio of largest to smallest entries of S) before trying LLSQs.

Worked Example 2

- Let’s fit a 1-parameter template to data, but possibly want to shift it (e.g. fitting a for a source amplitude at various positions).
- If A is n by 1, then $A^T N^{-1} A$ is a scalar.
- now we have $m = A^T N^{-1} d / (A^T N^{-1} A)$.
- If N is “constant” (i.e. $N_{ij} = f(i-j)$) and we shift the template $A_i \rightarrow A_{i+\delta}$, how does the denominator depend on δ?
- Up to an overall constant, we can make $N^{-1}d$ and dot it against the various shifted A’s.