Presenter: Borries Demeler

Topic:
Survey of Numerical Modeling

Copy of Lecture at:

https://demeler.uleth.ca/biophysics/archive/Demeler/

Survey of Numerical Modeling

We will touch on the following subjects:

- Model Building
  - Exact models
  - smoothing
- Theory of Fitting
  - What is a good fit and how can it be measured?
- Parameter Estimation
- Optimization – how do I fit the model to the data?
  - Linear vs. non-linear least squares
  - Linearization of non-linear systems
  - Brute force methods – Grid searches
  - Stochastic Methods – Genetic algorithms
- Effect of noise on analysis
- Statistical analysis and Monte Carlo approaches
- Parallel Implementation

Data Fitting

Modeling involves the description of some *observable data* (experimental measurements) using a mathematical *equation* that describes the underlying physical properties of the experiment.

First, we need to identify a general, mathematical model that can represent the observed data. The *parameters* of the model describe the specifics of the data.

Second, we need to determine the *values* of the parameters in the model that best fit our data. This is accomplished by a fitting algorithm that minimizes the difference between the data and model. Generally, an initial estimate is required that is then improved.

Finally, we need to estimate the error in the parameters we determined in the fitting process and obtain the *confidence intervals*.

Model Building

To build a model, one needs to understand the physical properties of the observed process. Many processes can be described by differential equations. When solved, these equations describe a linear or nonlinear model:

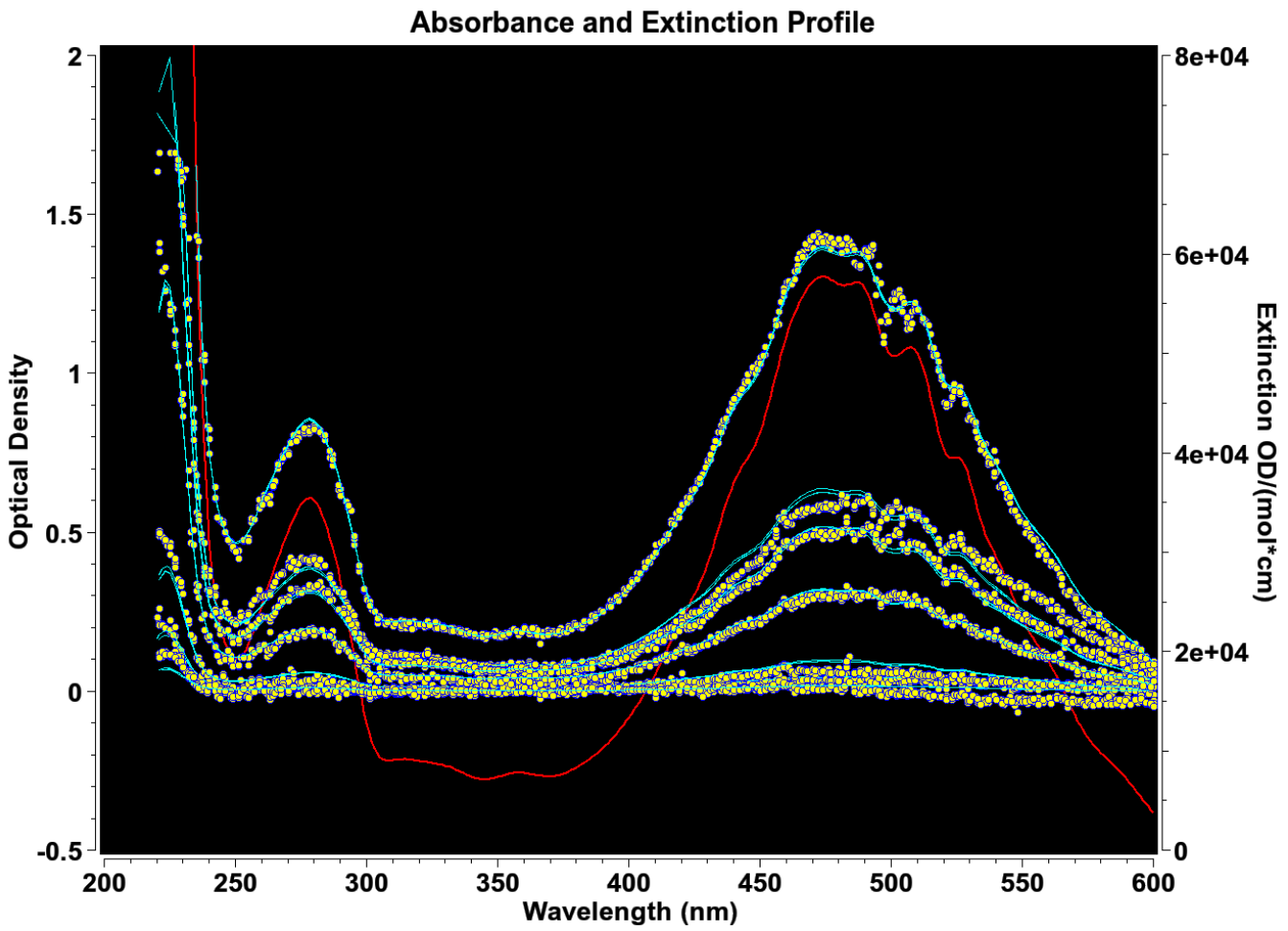
Example – radioactive decay:

Hypothesis: The rate of decay is proportional to the number of nuclei present.

Mathematical model: $$\frac{\partial N}{\partial t} = -a\,N$$

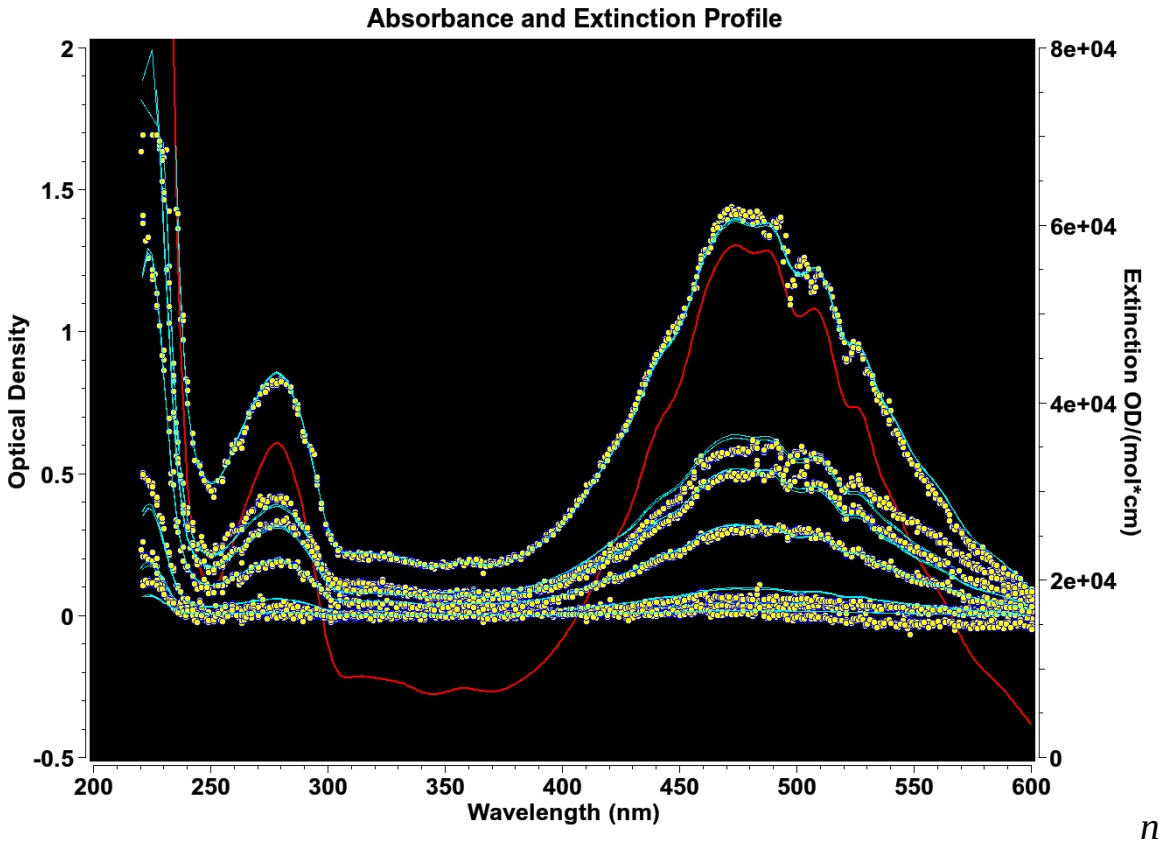
Solve: $$N = N_0\, e^{-a(t-t_0)} + b$$

You start with some experimental data...



Absorption data from multiple concentrations fitted to a sum of Gaussian functions

You start with some experimental data...



$$OD = a\sum_{i=1}^{n} c_i e^{\left[\frac{-(x_i-\mu_i)^2}{2\sigma_i^2}\right]} + b$$

Absorption data from multiple concentrations fitted to a sum of Gaussian functions

Method of Least Squares

Fitting Data to a Model by the Method of Least Squares:

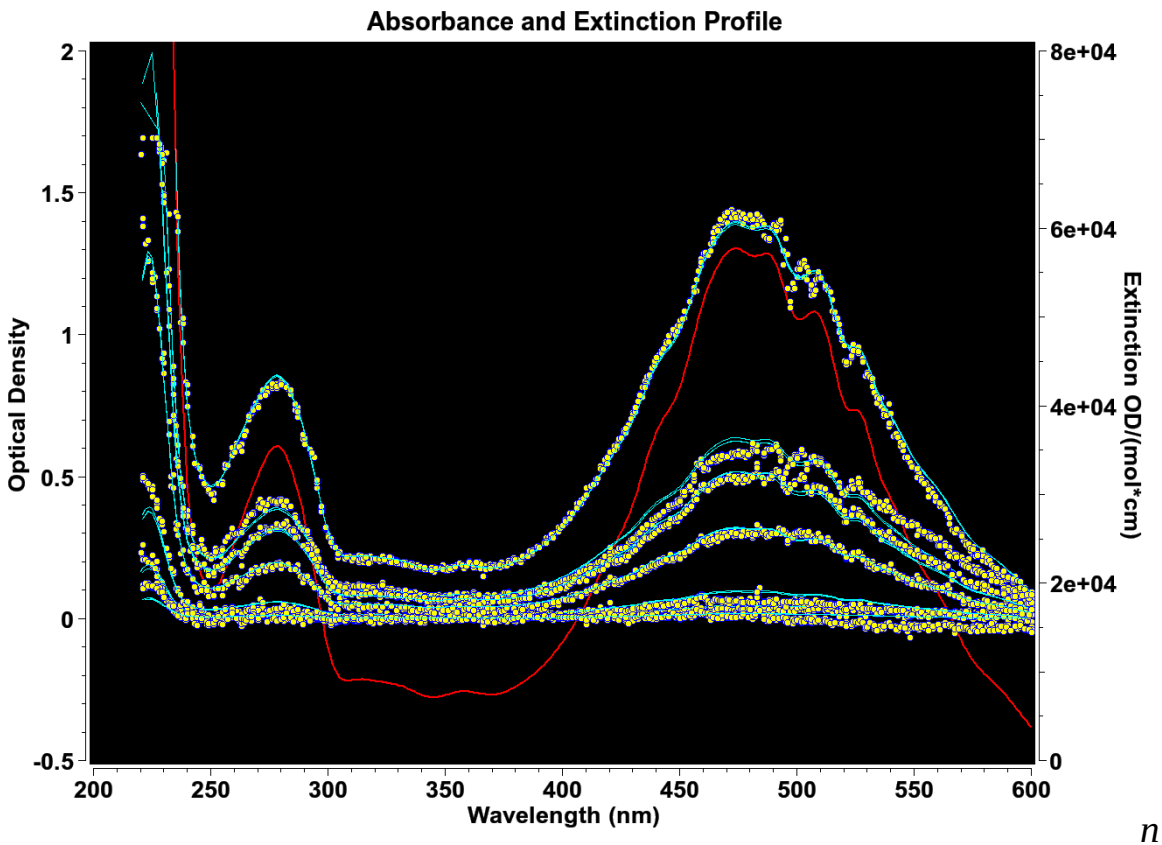
Any observable process that influences the measurement needs to be accounted for in order for the model to yield meaningful results. The object is then to minimize the residuals between the model and the data:

$$MIN \sum_{i=1}^{n} \left(\frac{Data_i - (Model_i)}{uncertainty_i} \right)^2 = MIN \ \chi^2$$

Extracting parameters from a simulated solution by fitting the model to experimental data is called an *"Inverse Problem"*

A non-parametric fit is used to smooth data for display, where the intrinsic model is of little interest, and hence the parameters are not needed.

You start with some experimental data...



Parameters aren't really useful here (except outer amplitudes), model is only used to find a smoothing shape that can be scaled to the appropriate concentration

$$OD = a\sum_{i=1}^{n} c_i e^{\left[\frac{-(x_i-\mu_i)^2}{2\sigma_i^2}\right]} + b$$

Absorption data from multiple concentrations fitted to a sum of Gaussian functions

Method of Least Squares

$$MIN \sum_{i=1}^{n} \left(\frac{Data_i - (Model_i)}{uncertainty_i} \right)^2 = MIN \ \chi^2$$

Assumptions made in the Method of Least Squares:

The model is a truthful representation of reality

All error is associated with the dependent variable. We can scale the reliability of each observation with an uncertainty factor σ_i.

All experimental noise is considered to be of Gaussian distribution

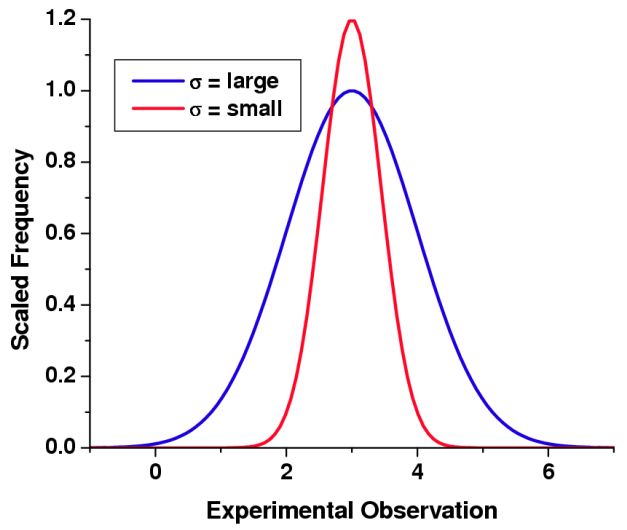
Experimental Uncertainties

The uncertainty of a measurement can be determined by repeating the experiment several times. Each time, a slightly different value is obtained for the experimental observation. Assuming a Gaussian distribution of errors in the measurement, one can determine the standard deviation σ of the distribution of measurement values, and use σ to set error bars on a measurement and to scale the contribution of a datapoint to the sum of the residuals.

The standard deviation can be calculated by using this formula:

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2}$$

where $\bar{x}$ is the average of all measurements.



Fitting Basics

For a straight line we have: y = a + bx

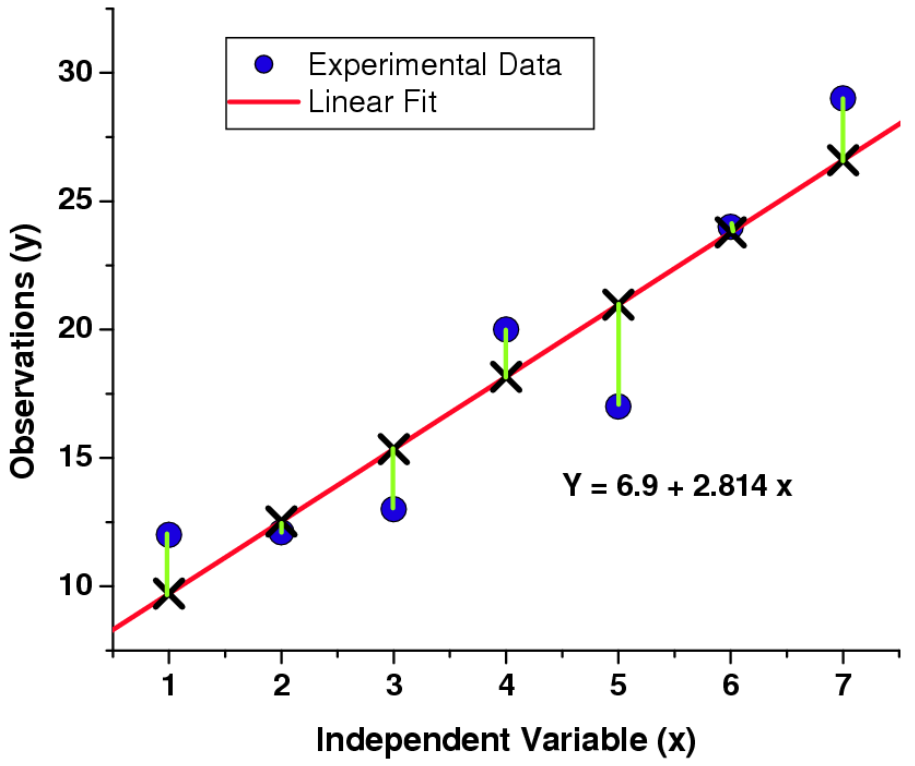
The least squares equation Is given by:

$$\sum_{i=1}^{n}\left(\frac{D_i - M_i}{\sigma_i}\right)^2$$

The distances are measured perpendicular to the data.

The object is to find the equation of the straight line that minimizes the distance between the straight line and the data points.

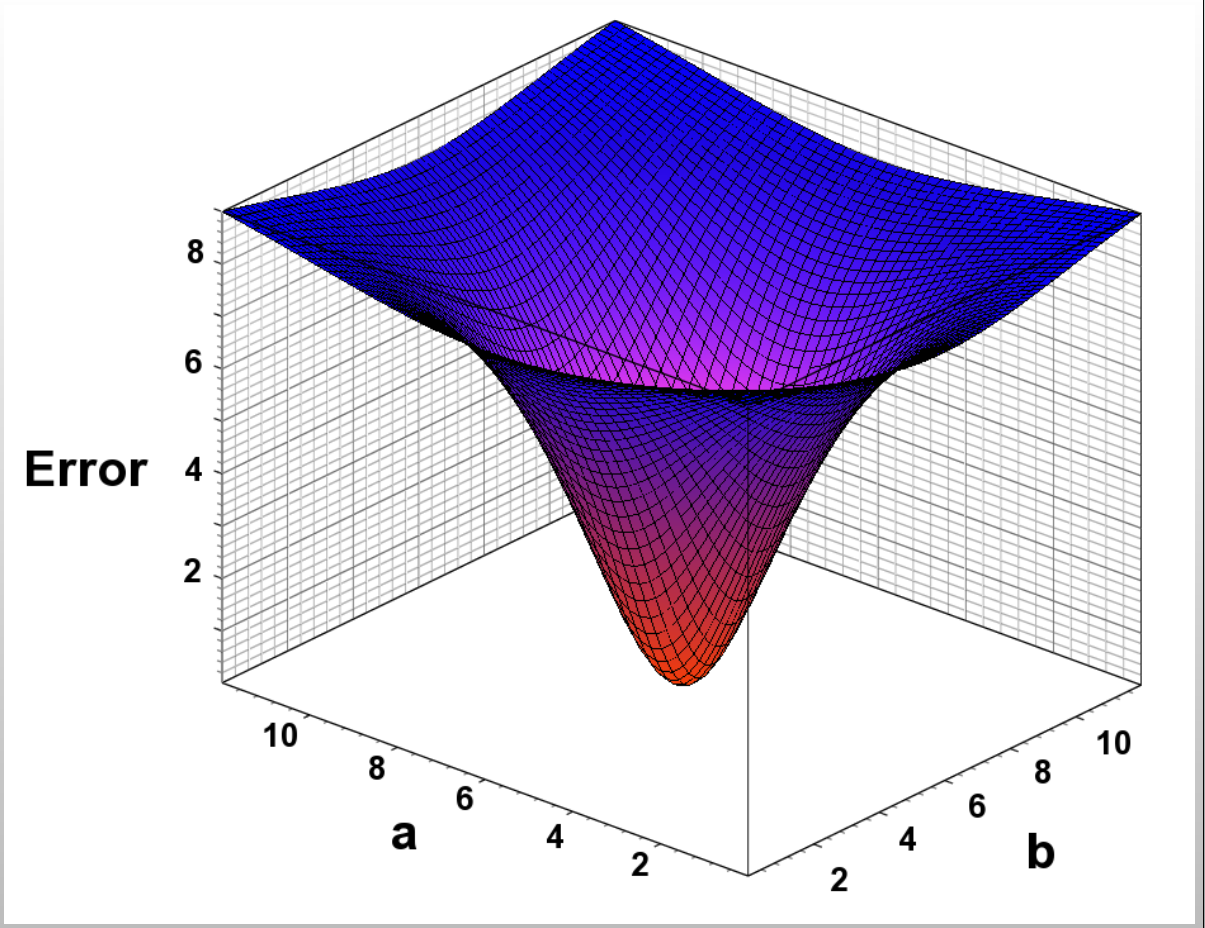
$$\sum_{i=1}^{n}\left(\frac{D_i - a - b\,x_i}{\sigma_i}\right)^2 = \chi^2(a,b)$$



Fitting Basics

What is an error surface?

Each parameter combination a, b results in a unique error when fitted to the experimental data.
The optimal solution occurs where the error is the smallest.
Ideally, the error surface is continuously differentiable.



Error surface for some function *y = F(a, b)*

Solving the Least Squares Equation

The minimum in the differences occur where the derivative of our objective function with respect to the parameters is zero, so we need to differentiate it with respect to the parameters of interest, a and b:

$$\sum_{i=1}^{n}\left(\frac{D_i-(a+b\,x_i)}{\sigma_i}\right)^2 = \chi^2(a,b)$$

$$\frac{\partial\chi^2}{\partial a} = 0 = -2\sum_{i=1}^{n}\left(\frac{D_i-a-b\,x_i}{\sigma_i^{\,2}}\right)$$

$$\frac{\partial\chi^2}{\partial b} = 0 = -2\sum_{i=1}^{n}\left(\frac{x_i(D_i-a-b\,x_i)}{\sigma_i^{\,2}}\right)$$

Solving the Least Squares Equation

This leads to a system of linear equations:

$$\sum_{i=1}^{n} \frac{a}{\sigma_i^2} + \sum_{i=1}^{n} \frac{b\,x_i}{\sigma_i^2} = \sum_{i=1}^{n} \frac{D_i}{\sigma_i^2}$$

$$\sum_{i=1}^{n} \frac{a\,x_i}{\sigma_i^2} + \sum_{i=1}^{n} \frac{b\,x_i^2}{\sigma_i^2} = \sum_{i=1}^{n} \frac{D_i\,x_i}{\sigma_i^2}$$

Or, in matrix form:

$$\begin{bmatrix} \sum_{i=1}^{n} \frac{1}{\sigma_i^2} & \sum_{i=1}^{n} \frac{x_i}{\sigma_i^2} \\ \sum_{i=1}^{n} \frac{x_i}{\sigma_i^2} & \sum_{i=1}^{n} \frac{x_i^2}{\sigma_i^2} \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} \frac{D_i}{\sigma_i^2} \\ \sum_{i=1}^{n} \frac{D_i\,x_i}{\sigma_i^2} \end{bmatrix}$$

Solving the Least Squares Equation

Let:

$$A = \begin{bmatrix} \sum_{i=1}^{n} \frac{1}{\sigma_i^2} & \sum_{i=1}^{n} \frac{x_i}{\sigma_i^2} \\ \sum_{i=1}^{n} \frac{x_i}{\sigma_i^2} & \sum_{i=1}^{n} \frac{x_i^2}{\sigma_i^2} \end{bmatrix}, \quad X = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \quad \text{and } B = \begin{bmatrix} \sum_{i=1}^{n} \frac{D_i}{\sigma_i^2} \\ \sum_{i=1}^{n} \frac{D_i x_i}{\sigma_i^2} \end{bmatrix}$$

(where $x_1 = a$ (i.e., intercept) and $x_2 = b$ (i.e., slope))

In matrix notation: AX = B, with solution $A^{-1}AX = A^{-1}B = X$

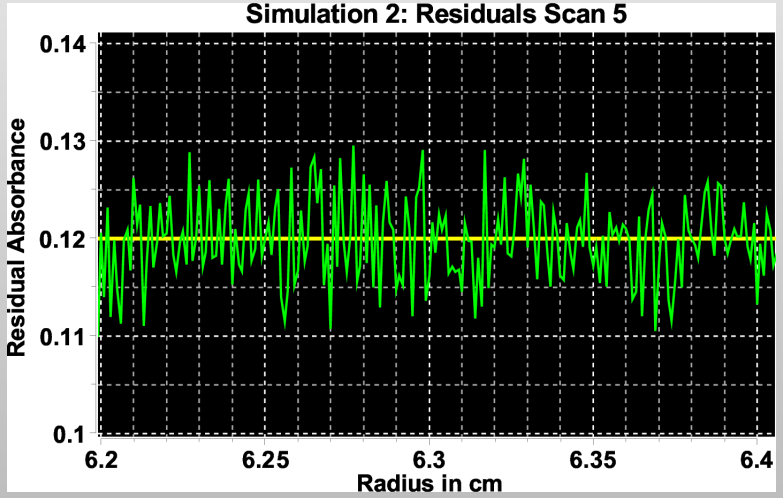
The equations can be solved either by inverting A or by using Cramer's Rule:

$$x_1 = \frac{b_1 a_{22} - b_2 a_{12}}{a_{11} a_{22} - a_{12} a_{21}}, \quad x_2 = \frac{b_2 a_{11} - b_1 a_{21}}{a_{11} a_{22} - a_{12} a_{21}}$$

Goodness of fit

The quality of the fit is determined by the randomness of the residuals and the root mean square deviation (RMSD).

The randomness of the residuals can be measured by determining the runs. Runs (R) are the number of consecutive positive (*p*) or negative (*n*) residuals from the mean.



$$R_T = \frac{R - \bar{R}}{\sigma_R}$$

$$\bar{R} = \frac{2np}{n+p} + 1$$

$$\sigma_R^2 = \frac{2np(2np - n - p)}{(n+p)^2(n+p-1)}$$

The R_T value is a measure of randomness and can be compared to a normal table to find out the probability of the test being random

Linear Models

The equation of a straight line is considered a *linear equation*:

$$y = a + bx$$

This equation is *linear* in the coefficients that are fitted. The equation doesn't have to be that of a straight line to be considered linear.

$$y = a + bx + cx^2 + dx^3 + ex^4 + \ldots$$

As long as the coefficients are linear, the equation is considered a linear fitting equation, no matter how wildly nonlinear the terms of the independent variable are:

$$y = a + b(x - \sin(x^3)) + c\, e^{-(4-3x)} + d \ln(3x^4) + \ldots$$

In general, we can write for any linear equation:

$$y = a_1X_1 + a_2X_2 + a_3X_3 + a_4X_4 + \ldots$$

where X_i can be any nonlinear term.

Linear Models

Linearization of a nonlinear equation:

Turn $y = ae^{bx}$ **into a linear function of the form:**

$$y = a + bx$$

take log on both sides:

Fitting the log of y reduces the nonlinear equation to a linear equation, y* = a* + bx, where y* = ln y and a* = ln a.

$$\ln y = \ln a + bx$$

Parameter Constraints

Sometimes, we may want to constrain the value of a parameter – for example, we don't want the amplitude of an exponential to turn negative during fitting:

$$y = a\, e^{bx} + c = e^{\ln(a) + bx} + c$$

By making the transformation to fitting the log of a number we can assure that the number itself will never be negative (negative amplitudes don't make sense in many physical models).

Nonlinear Regression

Why is it such a big deal if an equation is linear or nonlinear? It turns out that nonlinear functions need to be fitted using iterative approaches, while linear functions can be fitted in a single iteration, so it helps to have the objective function in a linear form. For nonlinear systems, a Jacobian is defined.

The idea: Iteratively improve the parameter estimates by following along the gradient of the error function in the direction of maximum "improvement". This requires knowledge of the partial derivatives for each parameter at each point in the experiment. We build the Jacobian matrix:

$$J = \begin{bmatrix} \left(\frac{\partial X}{\partial a_1}\right)_{x1} & \left(\frac{\partial X}{\partial a_2}\right)_{x1} & \cdots & \left(\frac{\partial X}{\partial a_n}\right)_{x1} \\ \left(\frac{\partial X}{\partial a_1}\right)_{x2} & \left(\frac{\partial X}{\partial a_2}\right)_{x2} & \cdots & \left(\frac{\partial X}{\partial a_n}\right)_{x2} \\ \cdots & \cdots & \cdots & \cdots \\ \left(\frac{\partial X}{\partial a_1}\right)_{xm} & \left(\frac{\partial X}{\partial a_2}\right)_{xm} & \cdots & \left(\frac{\partial X}{\partial a_n}\right)_{xm} \end{bmatrix}$$

Nonlinear Regression

Our equation is: $J * R = \Delta y$, **with** $R = a - g$

where J **is the Jacobian matrix,** g **is the current parameter estimate,** a **is the adjustment made to the parameter estimate in the current iteration, this is the value we need to find.** Δy **is the difference between the experimental data and the model**

Solve for a**:** $J^T J \ R = J^T \Delta y$, **substitute** $J^T \Delta y = B$,
Option 1: use inverse: $(J^T J)^{-1} J^T J \ R = R = (J^T J)^{-1} J^T \Delta y$
Option 2: $J^T J$ **is positive definite, so use Cholesky decomposition:**
$J^T J = LL^T$
$L(L^T R) = B$, **substitute** $L^T R = Z$, **to get** $LZ = B$ **and solve for** Z **using forward substitution, then solve for** R **using backward substitution:** $L^T R = Z$, **then solve for** a **to get the adjustment for the parameter.**

Iterate until converged.

Optimization Methods

Linear Optimization:

Straight line fits

Generalized linear least squares - single iteration fitting of objective functions of the type:

$$y = a_0 + \sum_{i=1}^{n} a_i X_i, \quad -\infty < a_i < +\infty$$

NNLS (non-negative constrained least squares):

$$y = a_0 + \sum_{i=1}^{n} a_i X_i, \quad a_i \geq 0$$

Multidimensional spectrum analyses (HPC recommended)

Non-parametric fits (B-splines, polynomial smoothing, etc)

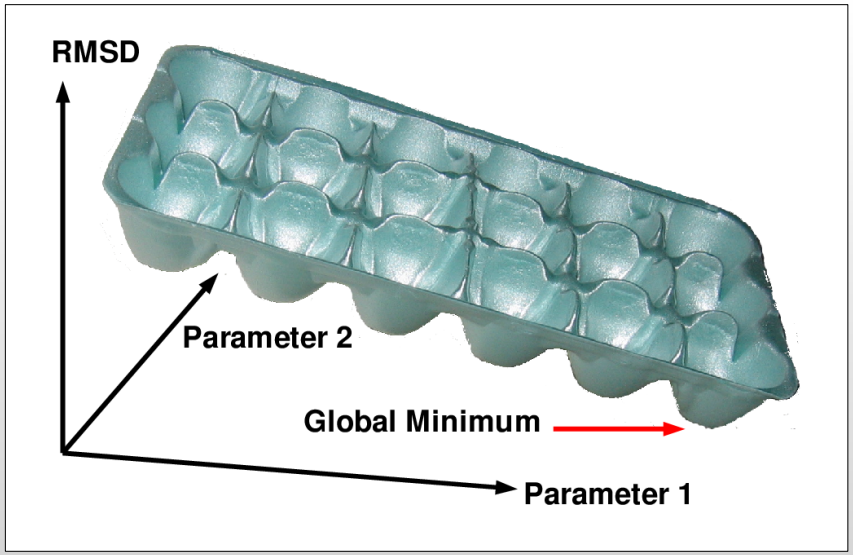
Optimization Methods

Nonlinear Optimization using Gradient Descent Methods for functions of the type:

$$y = F(a_i, x_i)$$

Levenberg-Marquardt (stable, robust, works well even if initial guesses are rather far away from optimum)
Gauss-Newton methods
Quasi-Newton (works well near optimum)
Conjugate gradients
Tangent approximation methods (derivatives are not required)
Neural networks

The Optimization Challenge:



Problem with nonlinear least squares optimization:

For multi-component systems, the nonlinear least squares fitting algorithm gets easily stuck in local minima and the solution depends on the starting points. Problem gets worse with more parameters (i.e., multiple components).

Optimization Methods

Stochastic Methods

Monte Carlo
Simulated Annealing
Random walk
Genetic Algorithms

Optimization Methods

Comparison Stochastic vs Deterministic Fitting Methods:

Stochastic:

- **Large search space possible**
- **Generally slow converging**
- **Excellent convergence properties if given enough time**
- **Compute-intensive**
- **Suitable for many parameters**
- **Good for ill-conditioned error surfaces**
- **Derivatives not needed**

Deterministic:

- **Small search space**
- **Suitable for a few parameters only**
- **Well-conditioned error surface**
- **Very fast converging**
- **Requires derivatives**

Noise & Data Modeling Considerations

Fitting of noisy data prevents unique solutions – multiple solutions are possible.

We need to *minimize noise* when modeling data.

There are three ways to reduce or eliminate noise:

1. **fit the noise**
2. **maintain an exceptionally well tuned instrument**
3. **design your experiment to optimize the quality of the data**

There are two noise types:

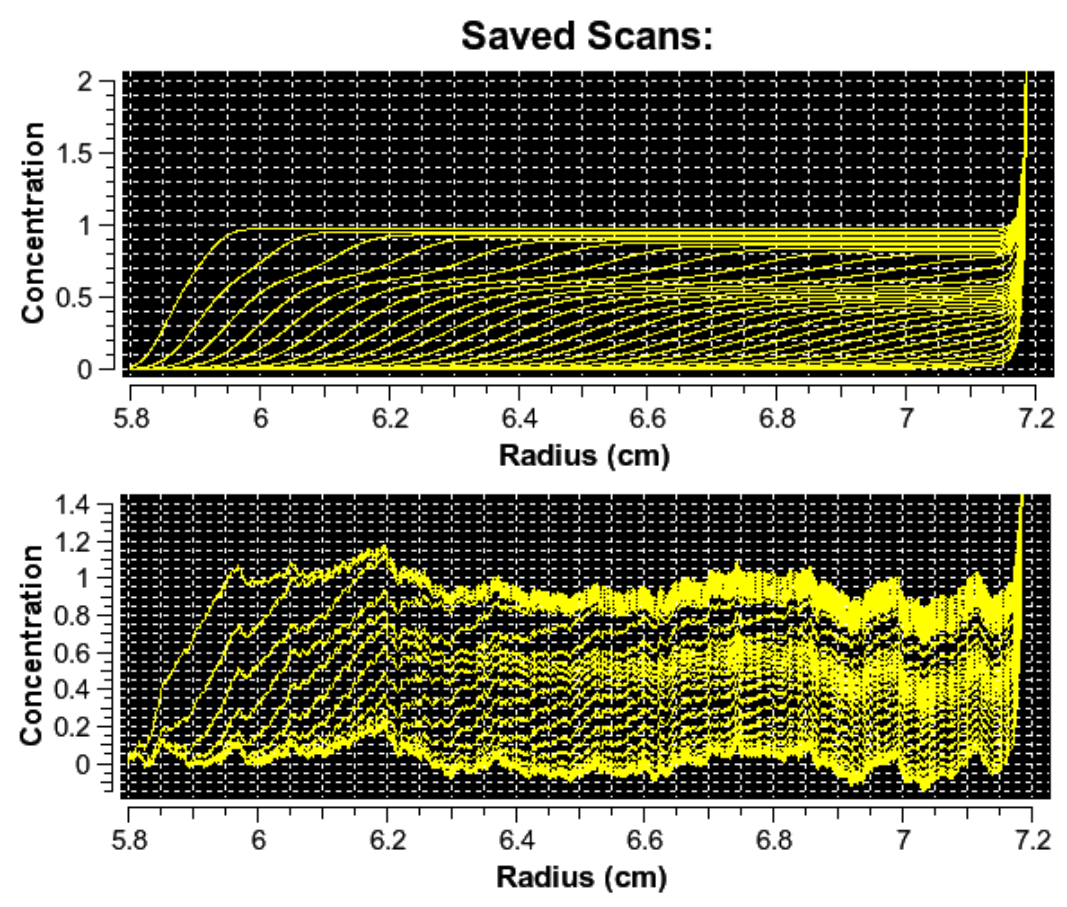
1. Systematic noise: Signal comes from a systematic source that is not part of the parametric model (finger print on the lens of a camera) and is highly correlated with some feature of the experiment.

2. Stochastic (random noise): Noise is (hopefully) Gaussian in distribution and uncorrelated to any feature of the experiment

(1) can often be fitted and accounted for, which (2) must be minimized.

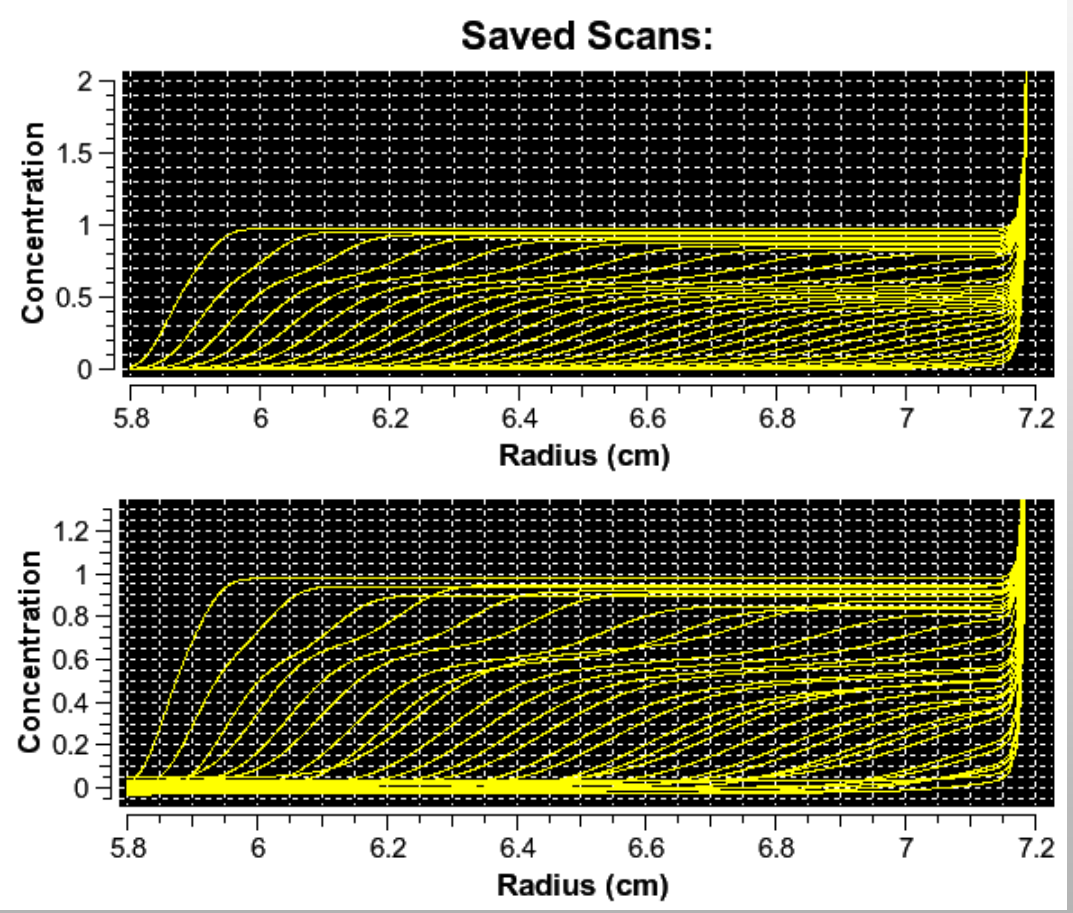
Different Types of Noise - Systematic

Time Invariant noise: **Noise is different for each radial position, but the same offset for each scan, and hence time independent.**



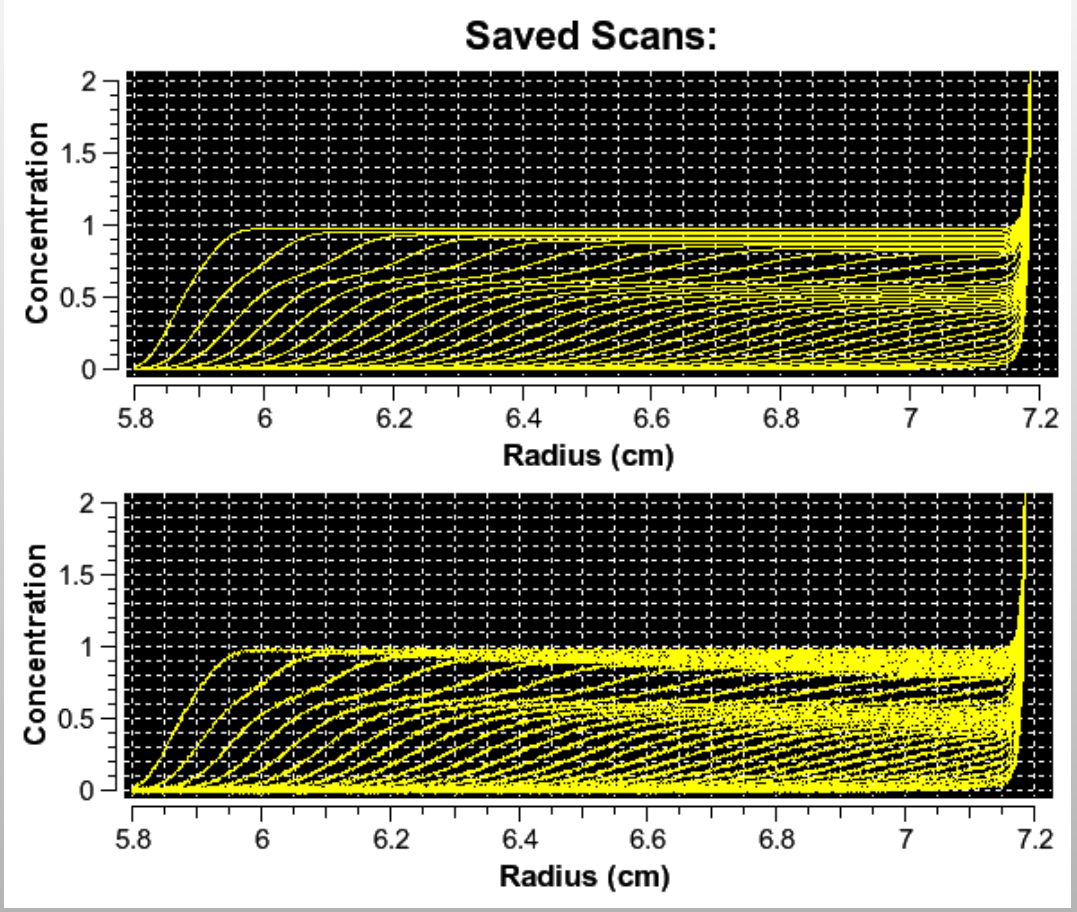
Different Types of Noise - Systematic

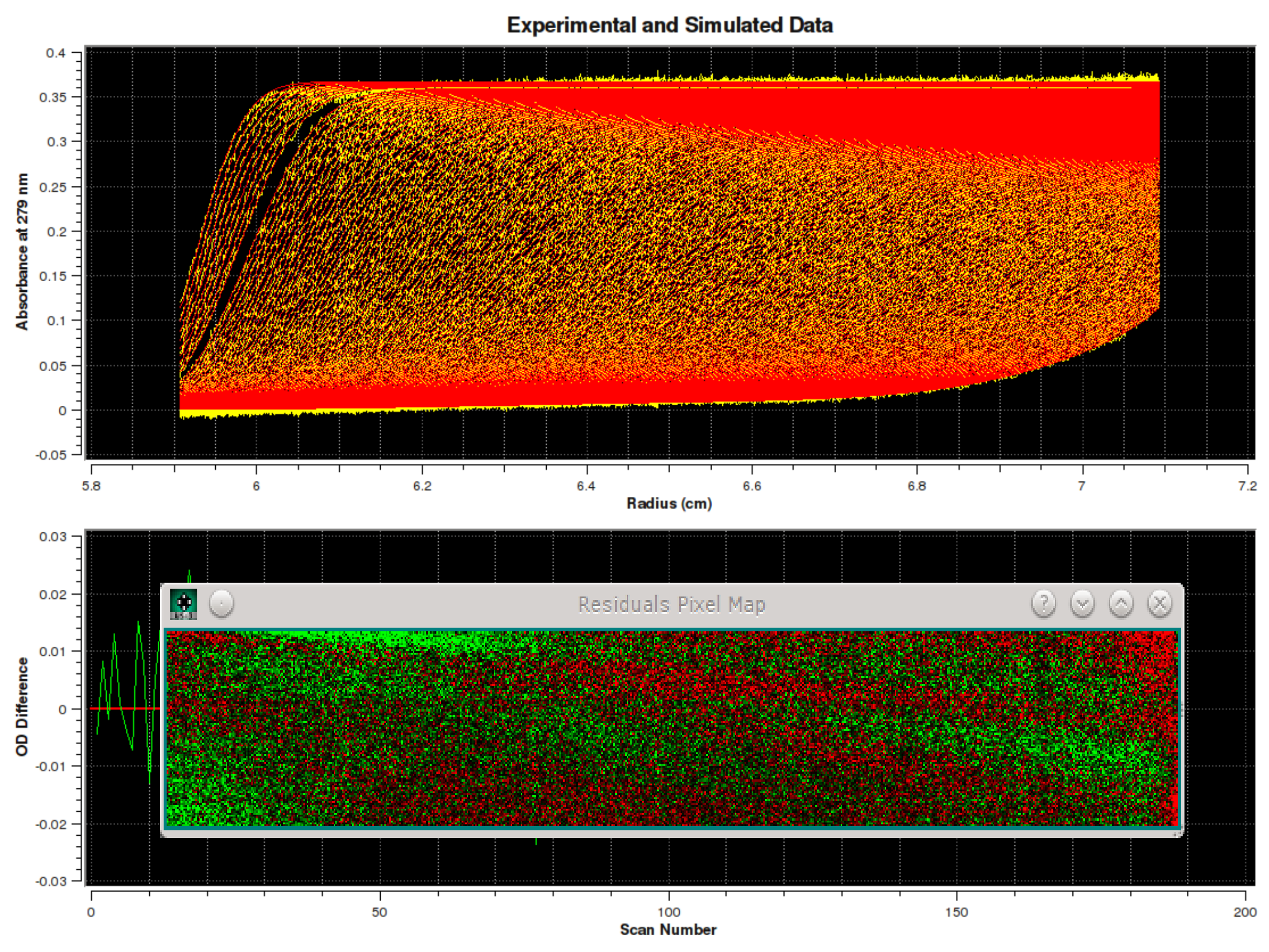
Radially Invariant noise: **Noise is different for each scan, but each radial position is offset by the same amount throughout the scan**



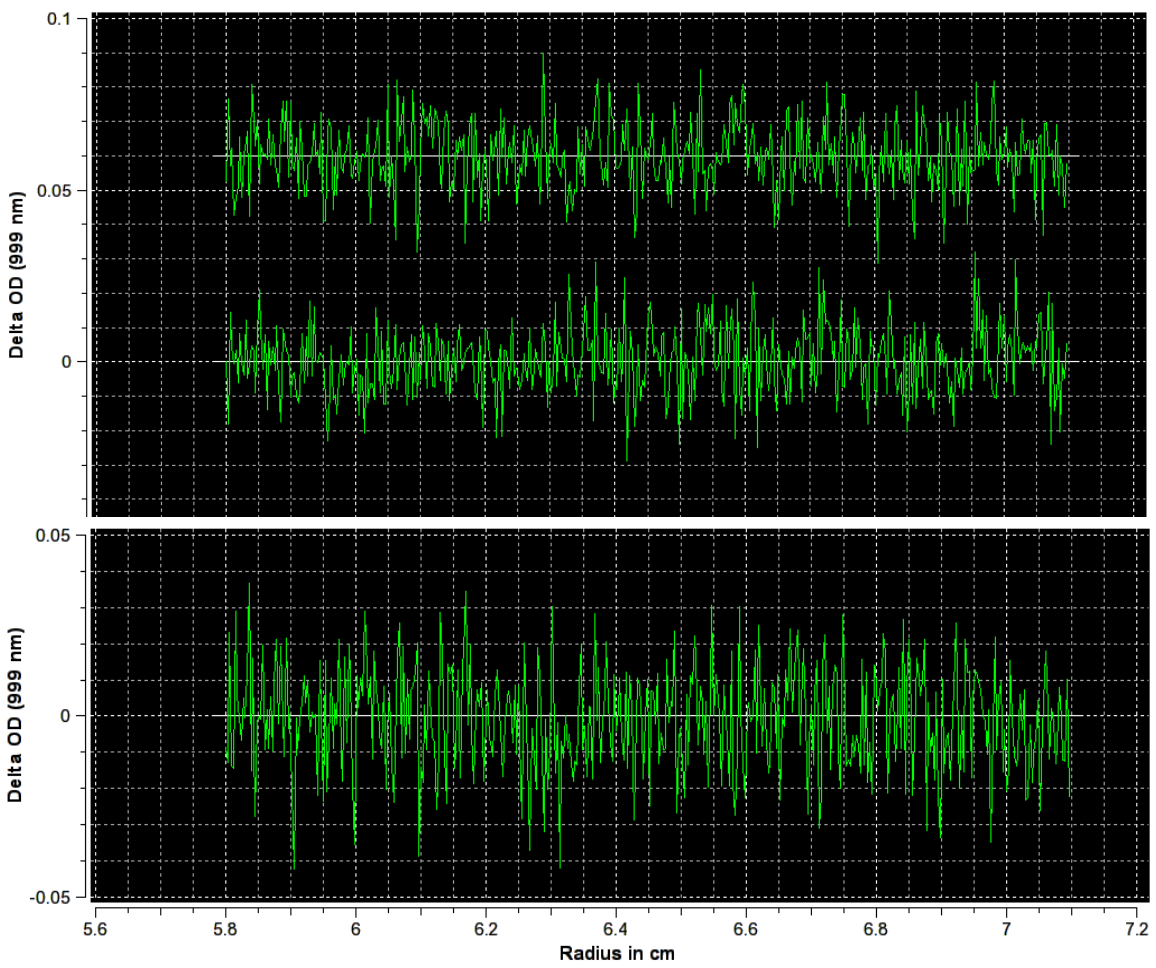
Different Types of Noise - Stochastic

Stochastic (random noise): **Noise is different for each radial and time point and it is (hopefully) Gaussian in distribution:**




Experimental and Simulated Data
Absorbance at 279 nm
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
-0.05
5.8
6
6.2
6.4
6.6
6.8
7
7.2
Radius (cm)
OD Difference
0.03
0.02
0.01
0
-0.01
-0.02
-0.03
Residuals Pixel Map
0
50
100
150
200
Scan Number

Tale of 2 noisy vectors



Noise of Scan 1

+

Noise of Scan 2

=

Noise * $\sqrt{2}$

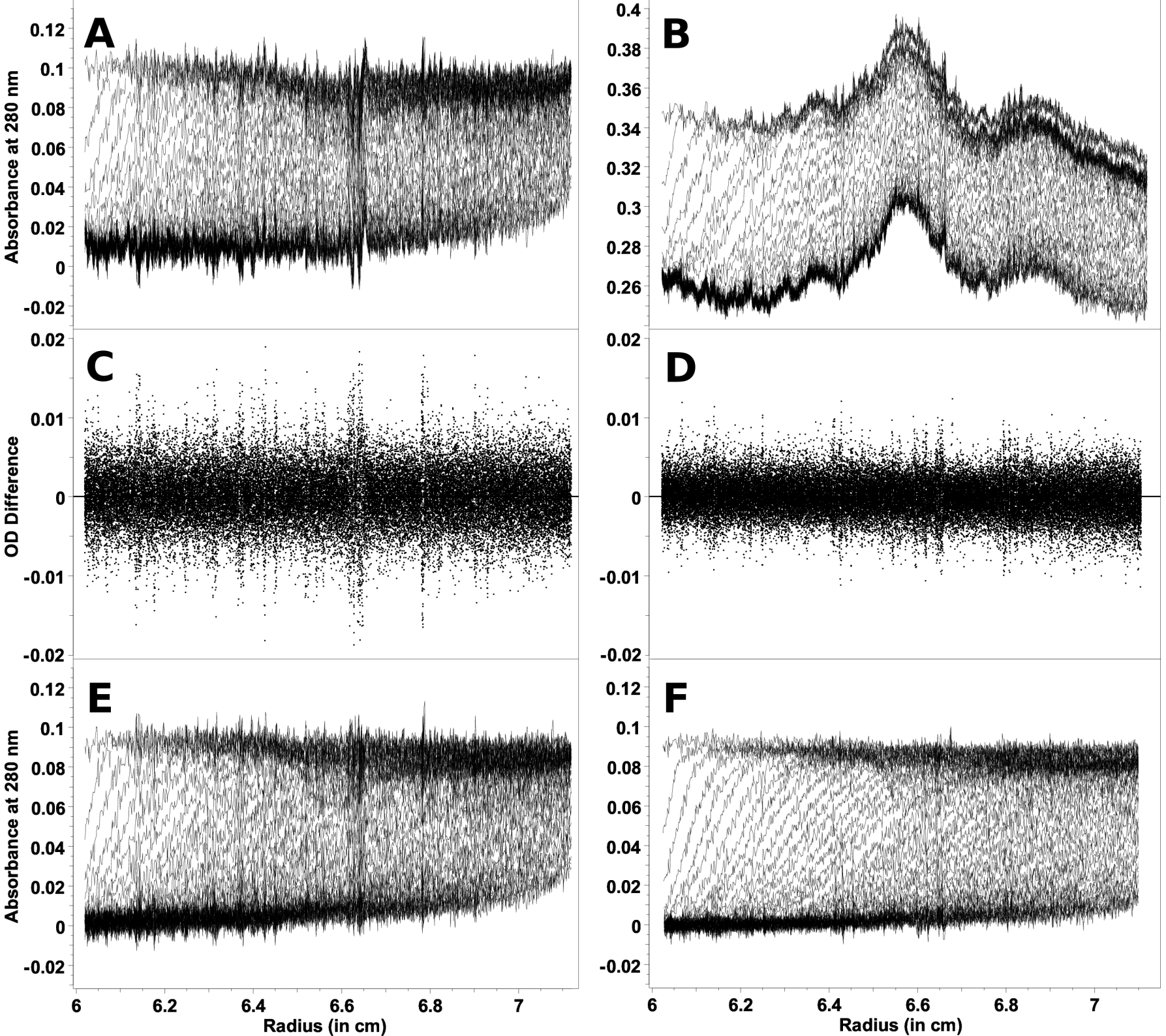
Intensity vs. Absorbance

Optical system considerations

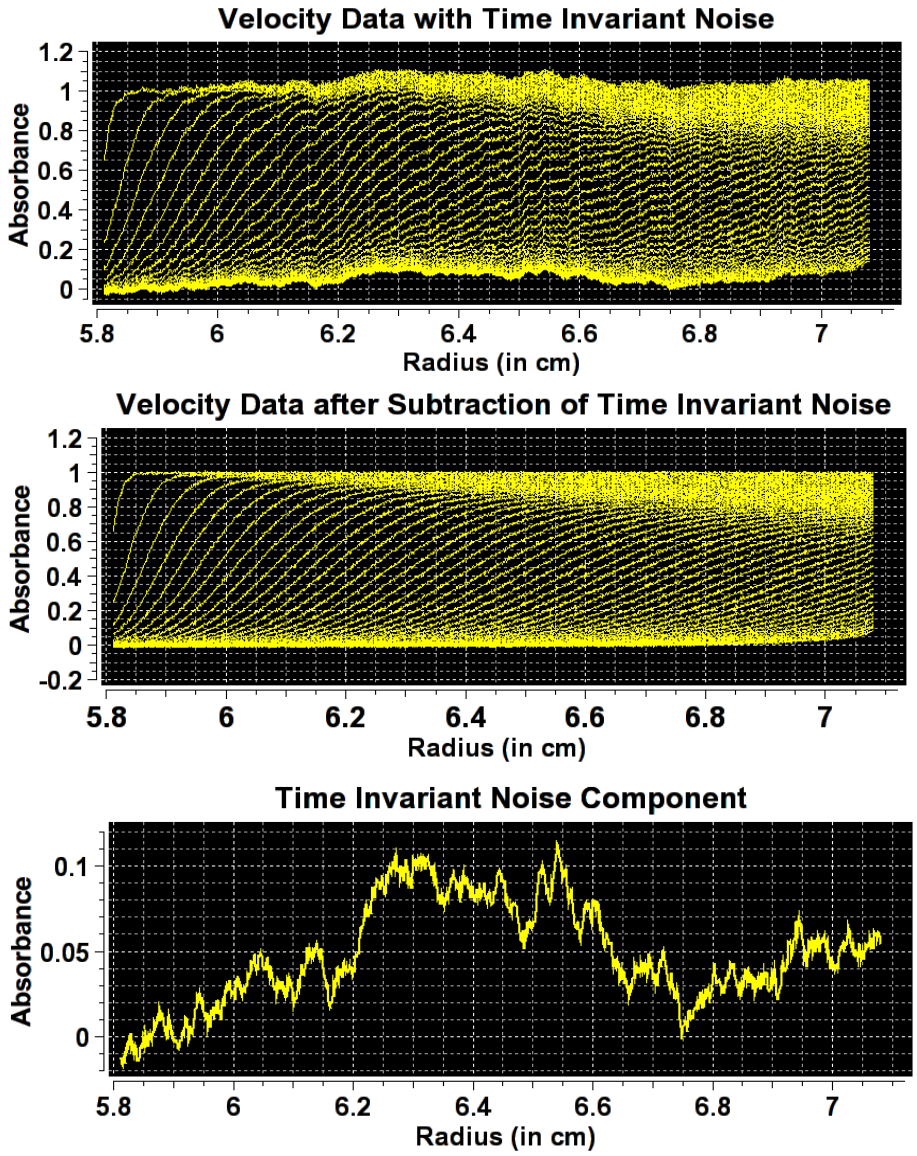
Intensity measurements record the intensity of light passing through one channel. Absorbance measurements record the intensity of light passing through one channel, then record the intensity of the light passing through the reference channel, and subtract it from the first channel. Each channel recording contains the (nearly) same amount of time invariant noise, but different amount of stochastic noise. Subtraction of time invariant noise will eliminate it:

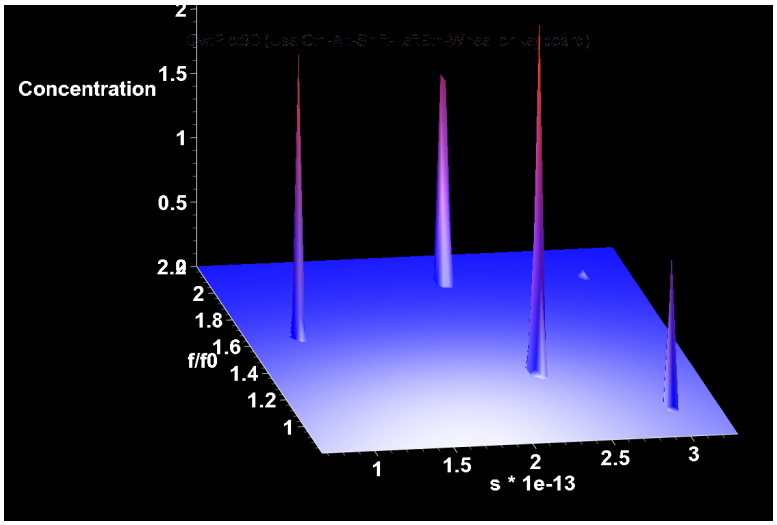
$$scan_1 = signal_1 + N_{s1} + N_{ti}$$
$$scan_2 = signal_2 + N_{s2} + N_{ti}$$
$$scan_1 - scan_2 = signal_1 - signal_2 + N_{s1} + N_{s2}$$

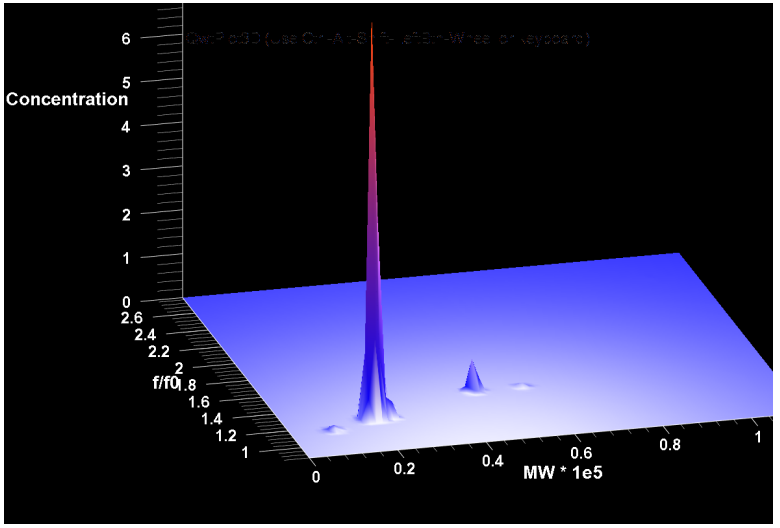
$$N_{s1} + N_{s2} \approx N_{s1}\sqrt{(2)}$$


A
B
C
D
E
F
Absorbance at 280 nm
OD Difference
Absorbance at 280 nm
0.12
0.1
0.08
0.06
0.04
0.02
0
-0.02
0.4
0.38
0.36
0.34
0.32
0.3
0.28
0.26
0.02
0.01
0
-0.01
-0.02
6
6.2
6.4
6.6
6.8
7
Radius (in cm)
Radius (in cm)

Factors that affect Accuracy – Time-invariant Noise







How can we deal with uncertainty?
A Recipe for Optimal Resolution:

Our solution is affected by random noise, time-invariant noise, the available signal, and its ratio to the random noise.

Improve signal to noise by:

- **Obtain lots of high quality data**
- **Exploit the entire dynamic range of the acquisition system**
- **Reduce noise and only use a well-functioning instrument**
- **Remove systematic noise from experimental data**

Replace fitting parameters with experimentally determined values from separate experiments

Explore the parameter space with a grid method, then parsimoniously regularize solution with GA, and use Monte Carlo to explore confidence regions

Perform global fits for multiple experimental conditions to improve signal

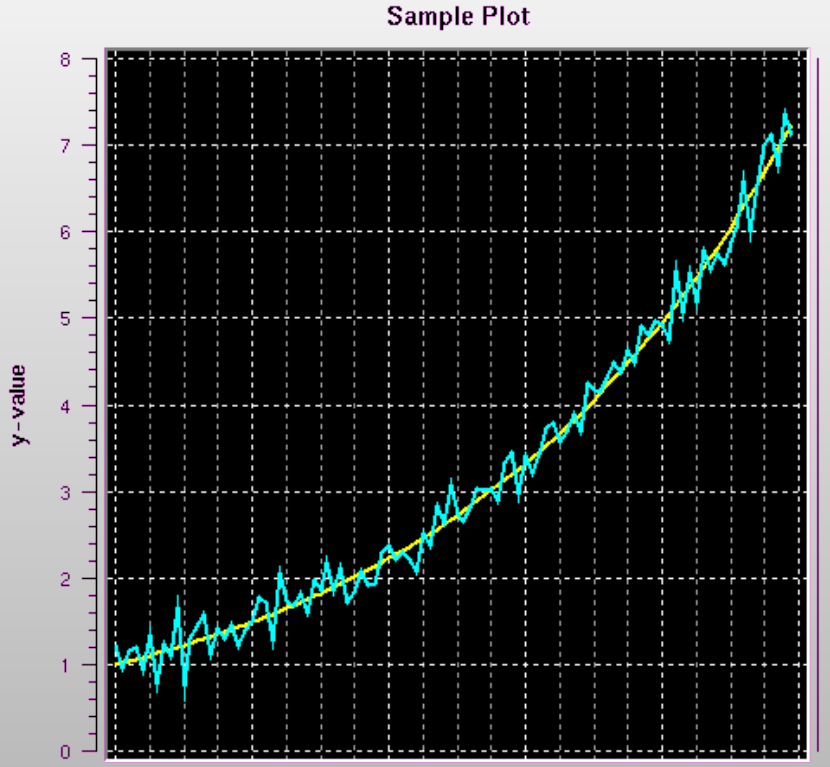
Summary:

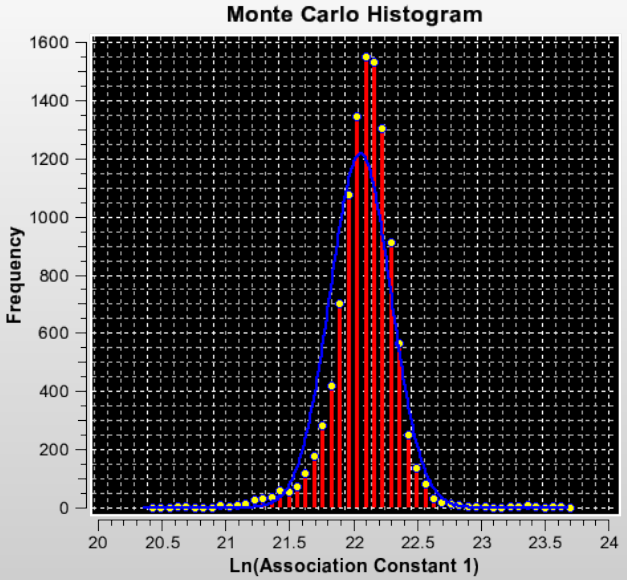
Remember:

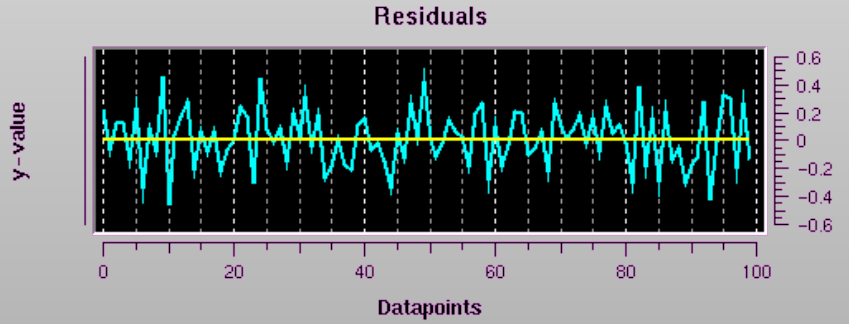
you cannot get reliable answers if you start with low quality input data!

Statistical Verification

Use Monte Carlo to determine the reliability of the fitted data:







Implementation of Monte Carlo Method

The Monte Carlo method is a stochastic approach that can be used to identify the effect noise has on the reliability of determined parameters. With the Monte Carlo approach the statistical confidence limits of each measured parameter can be determined.

Recipe for Monte Carlo:

Obtain a best-fit solution from model function fit and confirm that the residuals are random and without systematic deviation

Generate new synthetic Gaussian noise with the same quality as was observed in the original experiment and add it to the best-fit solution

Re-fit the solution

Repeat (2-4) at least 100 times and collect all parameter values

Calculate statistics from Monte Carlo distribution for each parameter

Implementation of Monte Carlo Method

Generation of synthetic noise:

Method 1 – use Bootstrapping:

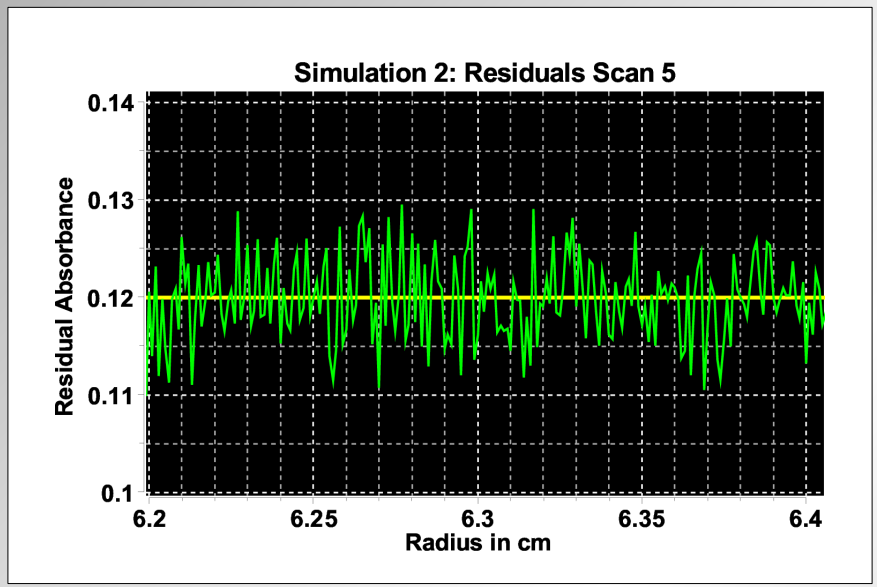
Permute a percentage of residuals. Take any residual, positive or negative, and place it elsewhere in the data. Assumption: The likelihood of a residual's magnitude is the same anywhere in the dataset.

Method 2 – generate Gaussian noise:

Run a 5-7 point Gaussian kernel over the residual vector from the best fit and use the Box-Müller algorithm to generate random new noise with the same quality at that position based on the variance obtained from the 5-point kernel.

Implementation of Monte Carlo Method

Generation of synthetic Gaussian noise with Method 2:



Take the absolute value of the residual values from 5-7 points, and smooth them with a Gaussian kernel (most weight on the center point). The average residual value is fed into the Box – Müller function to generate a new random residual that has a Gaussian probability distribution with a mean of the average residual value.

If there is a lot of noise in some area of the scan, the new data will have noise locally equivalent to the original data.

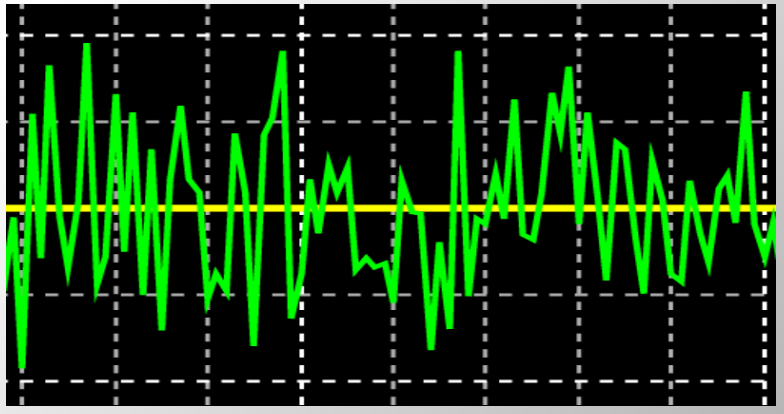
Generation of Gaussian noise is preferred because the quality of the noise varies at different points in the cell and it also varies with absorbance (more absorbance = more noise).

Implementation of Monte Carlo Method

Box – Müller function:

```
float box_muller(float m, float s)      /* normal random variate generator */
{                                       /* mean m, standard deviation s    */
     float x1, x2, w = 2.0;
     while (w >= 1.0)
     {
          x1 = 2.0 * ranf() - 1.0;
          x2 = 2.0 * ranf() - 1.0;
          w = x1 * x1 + x2 * x2;
     }
     w = sqrt((-2.0 * log(w)) / w);
     return( m + x1 * w * s );
}

float ranf()
{
     int N = 1;
     float temp = 0.0;
     temp = (((float) rand() / ((float) RAND_MAX + 1) * N));
     return(temp);
}
```

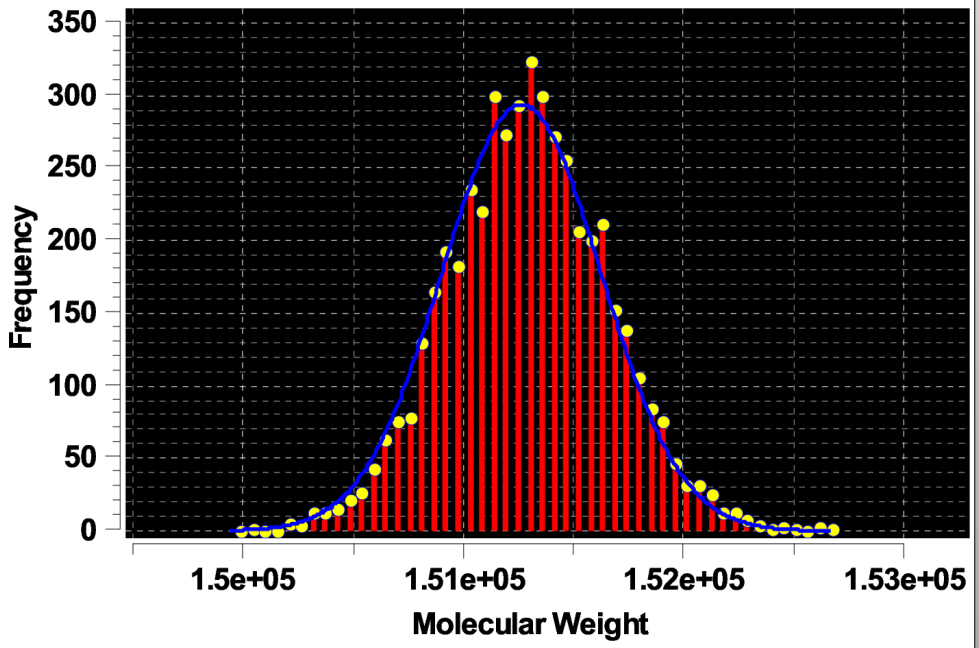


Obtaining Confidence Intervals

Once a Monte Carlo analysis is completed, a frequency distribution of parameter values for each parameter is obtained, and statistics for the distribution can be calculated:

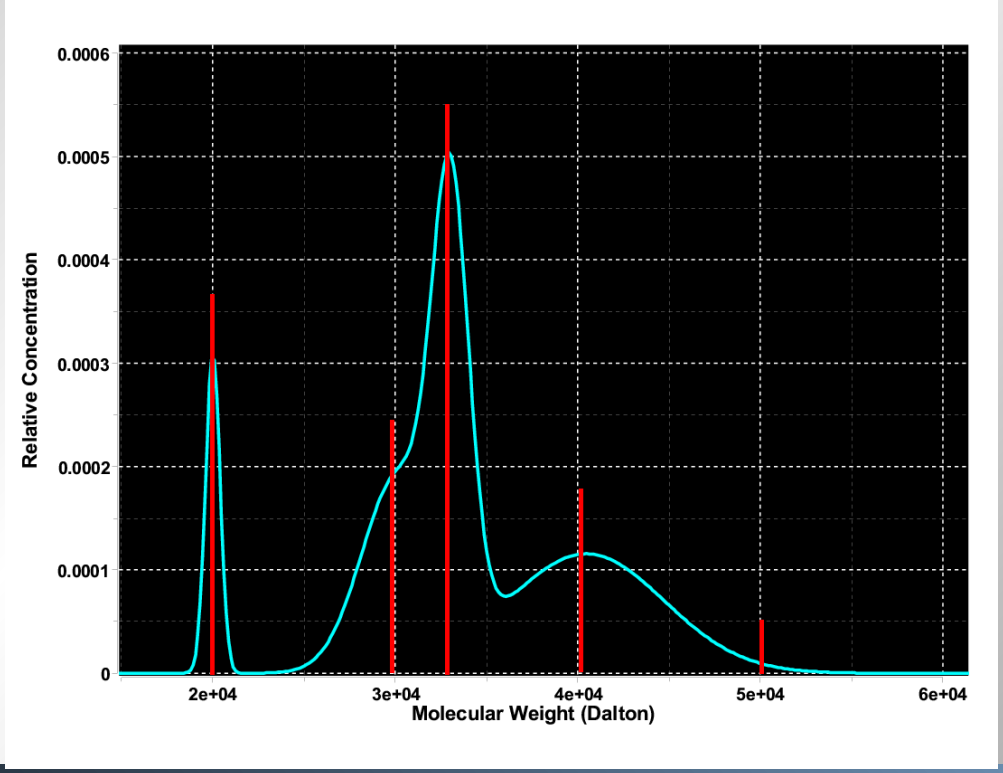
```
Molecular Weight

Maximum Value:              1.526728e+05
Minimum Value:              1.499355e+05
Mean Value:                 1.512681e+05
Median Value:               1.513042e+05
Skew Value:                 2.345094e-02
Kurtosis Value:             4.796727e-02
Lower Mode Limit:           1.513042e+05
Upper Mode Limit:           1.513589e+05
Mode Center:                1.513315e+05
95% Confidence Limits:      +6.445781e+02
                            -7.715156e+02
99% Confidence Limits:      +8.671094e+02
                            -9.940469e+02
Standard Deviation:         3.612517e+02
Standard Error:             3.572899e+00
Variance:                   1.305028e+05
Correlation Coefficient:    1.387418e-02
Gaussian Area:              5.592846e+05
95 % Confidence Interval:   1.505600e+05 (low),  1.519761e+05 (high)
99 % Confidence Interval:   1.503375e+05 (low),  1.521986e+05 (high)
```



Regularization:

Occam's Razor: The simplest solution describing the data is the preferred solution. Instead of regularizing the solution and introducing infinitely many solutes with different probability, we want to *REDUCE* the solution space to find the solution that has the smallest number of possible solutes. For probabilities of solutes inspect the GA evolution profile which provides MC statistics.



Solving the Least Squares Equation

Homework 1: For the dataset shown on the right calculate the equations for a and b for a straight line fit using Cramer's rule ($y = c_1 + c_2 x$). Assume a standard deviation of 1 for each measurement. Show your work. Compare your answer by fitting with a plotting program. Show results.

x	y	σ
2	10.6	1
4	12.1	1
6	14.5	1
8	20.8	1
10	17.3	1
12	24.7	1
14	29.1	1

$$A = \begin{bmatrix} \sum_{i=1}^{n} \frac{1}{\sigma_i^2} & \sum_{i=1}^{n} \frac{x_i}{\sigma_i^2} \\ \sum_{i=1}^{n} \frac{x_i}{\sigma_i^2} & \sum_{i=1}^{n} \frac{x_i^2}{\sigma_i^2} \end{bmatrix}, \quad X = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}, \quad \text{and } B = \begin{bmatrix} \sum_{i=1}^{n} \frac{D_i}{\sigma_i^2} \\ \sum_{i=1}^{n} \frac{D_i x_i}{\sigma_i^2} \end{bmatrix}$$

$$c_1 = \frac{b_1 a_{22} - b_2 a_{12}}{a_{11} a_{22} - a_{12} a_{21}}, \quad c_2 = \frac{b_2 a_{11} - b_1 a_{21}}{a_{11} a_{22} - a_{12} a_{21}}$$

Homework assignment: For each of the following situations, rank the appropriateness of each fitting approach from best to worst or not applicable: non-parametric, gradient descent, grid method, or stochastic, specify whether each model is linear or nonlinear in its fitting parameters, propose a function if non-parametric, and justify your answer for your ranking.

1. Smoothing a set of noisy data with a recognizable wave pattern

2. Fitting parameters a and b from function:

$$F(x) = a_0 + a_1 \log(a\,x) + a_2 \sin\left(b - x^2\right) + c$$

3. Fitting parameters a_i from function:

$$F(x) = \sum_{i=0}^{100} a_i e^{b_i\left(x^2 - c^2\right)}, \quad where\ b_i = \frac{i}{100}$$

4. Fitting parameters a_i from function:

$$F(x) = a_0 + a_1 \log(b\,x) + a_2 \sin\left(a_3 - x^2\right) + c$$

5. Fitting parameters *a, b* and *c* from function:

$$F(x) = \log(a\,x) + \sin\left(bx^2\right) + c$$

6. Fitting parameters a_1 - a_8:

$$F(x) = a_1 \log(a_{2x}) + \frac{a_3 \sin\left(\sqrt{2}\,a_4\,x^2\right)}{a_7\,x^3} + a_5 e^{\left(a_6 \frac{x^2}{2}\right)} + a_8$$