Course Instructor
Dr. Raymond C. Rumpf
Office: A-337
Phone: (915) 747-6958
E-Mail: rcrumpf@utep.edu

Computational Methods in EE

Topic 5 – Curve Fitting & Interpolation

EE 4386/5301 Computational Methods in EE

Outline

Computational Methods in EE

- Introduction
- Statistics of Data Sets
- Best Fit Methods
  - Linear Regression (ugly math)
  - Linear Least Squares (clean math)
  - Nonlinear Regression (moderate math)
- Exact Fit Methods
  - Fitting Polynomials
- Interpolation & Extrapolation

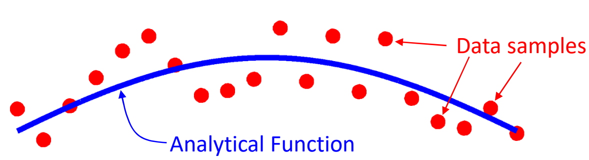
Computational Methods in EE

Introduction

What is Curve Fitting?

Computational Methods in EE

Curve fitting is simply fitting an analytical equation to a set of measured data.



$$f(x) = A + Be^{-\left(\frac{x-C}{D}\right)^2}$$

"Curve fitting" determines the values of A, B, C, and D so that $f(x)$ best represents the given data.

Why Fit Data to a Curve?

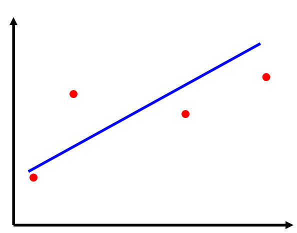
Computational Methods in EE

- We may want to estimate data between the discrete values (interpolation)
- We may want to fit measured data to an analytical equation to extract meaningful parameters.
- We may want to reduce noise
- We may want to observe and quantify a general trend

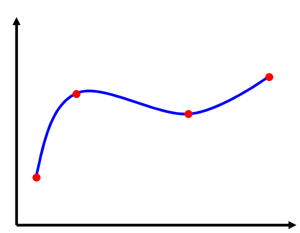
Two Categories of Curve Fitting

Computational Methods in EE

Best Fit – Measured data has noise so the curve does not attempt to intercept every point.



Exact Fit – Data samples are assumed to be exact and the curve is forced to pass through each one.



Computational Methods in EE

Statistics of Data Sets

Arithmetic Mean

Computational Methods in EE

If we had to come up with a single number that be represents an entire set of data, the arithmetic mean would probably be it.

$$f_{\text{avg}} = \frac{f_1 + f_2 + \cdots + f_M}{M} = \frac{1}{M}\sum_{m=1}^{M} f_m$$

Geometric Mean

The geometric mean is defined as

$$f_{\text{gm}} = \sqrt[M]{f_1 f_2 \cdots f_M}$$

The arithmetic mean tends to suppress the significance of outlying data samples. With the geometric mean, even a single small value among many large values can dominate the mean.

This is useful in optimizations where multiple parameters must be maximized at the same time and it is not acceptable to have any one of them low.

Variance & Standard Deviation

Standard Deviation σ_f

The standard deviation is a measure of the "spread" of the data about the mean. It is convenient because it shares the same units as the data.

$$\sigma_f = \sqrt{\frac{1}{M}\sum_{m=1}^{M}\left(f_m - f_{\text{avg}}\right)^2}$$

Variance v_f

Variance is used more commonly in calculations, but carries the same information as the standard deviation.

$$v_f = \sigma_f^2 = \frac{1}{M}\sum_{m=1}^{M}\left(f_m - f_{\text{avg}}\right)^2$$

Coefficient of Variation

Computational Methods in EE

The coefficient of variation (CV) is the standard deviation normalized to the mean.

Think of it as “relative standard deviation.”

$$\text{CV} = \frac{\sigma_f}{f_{\text{avg}}}$$

Computational Methods in EE

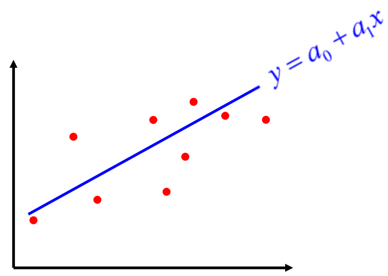
Linear Regression

(Best Fit, Ugly Math)

Goal of Linear Regression

Computational
Methods in EE

The goal of linear regression is to fit a straight line to a set of measured data that has noise.



$$(x_1, y_1),\ (x_2, y_2),\ \ldots,\ (x_M, y_M)$$
$$\downarrow$$
$$y = a_0 + a_1 x$$

Curve Fitting 13

Statement of Problem

Computational
Methods in EE

Given a set of measured data points: (x_1,y_1), (x_2,y_2), …, (x_M,y_M), we write our equation of the line for each point.

$$y_1 = a_0 + a_1 x_1 + e_1$$
$$y_2 = a_0 + a_1 x_2 + e_2$$
$$\vdots$$
$$y_M = a_0 + a_1 x_M + e_M$$

To make these equations correct, we must introduce an error term e called the *residual*.

We wish to determine values of a_0 and a_1 such that the residual terms e_m are as small as possible.

Curve Fitting 14

Criteria for "Best Fit"

Computational Methods in EE

We must define a single quantity that tells us how "good" the line fits our set of data.

Guess #1 – Sum of Residuals

$$E=\sum_{m=1}^{M}e_m$$

This does not work because negative and positive residuals can cancel and mislead the overall criteria to think there is no error.

Guess #2 – Sum of Magnitude of Residuals

$$E=\sum_{m=1}^{M}\left|e_m\right|$$

This does not work because is does not lead to a unique best fit.

Guess #3 – Sum of Squares of Residuals

$$E=\sum_{m=1}^{M}e_m^2$$

This works and leads to a unique solution.

Equation for Criterion

Computational Methods in EE

Our line equation for the mth sample is

$$y_m=a_0+a_1x_m+e_m$$

Solving this for the residual e_m gives

$$e_m=y_m-\left(a_0+a_1x_m\right)$$

This is the value of y of our line at point x_m.

This is our measured value of y.

From this, we can write our criterion as

$$E=\sum_{m=1}^{M}e_m^2=\sum_{m=1}^{M}\left(y_{\text{measured},m}-y_{\text{line},m}\right)^2=\sum_{m=1}^{M}\left(y_m-a_0-a_1x_m\right)^2$$

Least-Squares Fit

Computational Methods in EE

We wish to minimize the error criterion E.

We can identify minimums when the first-order derivative is zero.

$$\frac{\partial E}{\partial a_0} = 0 \quad \text{and} \quad \frac{\partial E}{\partial a_1} = 0$$

We seek the values of a_0 and a_1 that satisfy these equations.

This approach is solving the problem by least-squares (we are minimizing the squares of the residuals).

The Fun Math

Computational Methods in EE

Step 1 – Differential E with respect to each of the unknowns.

$$\begin{aligned}\frac{\partial E}{\partial a_0} &= \frac{\partial}{\partial a_0}\sum_{m=1}^{M}\left(y_m - a_0 - a_1 x_m\right)^2 \\ &= \sum_{m=1}^{M}\frac{\partial}{\partial a_0}\left(y_m - a_0 - a_1 x_m\right)^2 \\ &= \sum_{m=1}^{M}2\left(y_m - a_0 - a_1 x_m\right)\left(-1\right) \\ &= -2\sum_{m=1}^{M}\left(y_m - a_0 - a_1 x_m\right)\end{aligned}$$

$$\begin{aligned}\frac{\partial E}{\partial a_1} &= \frac{\partial}{\partial a_1}\sum_{m=1}^{M}\left(y_m - a_0 - a_1 x_m\right)^2 \\ &= \sum_{m=1}^{M}\frac{\partial}{\partial a_1}\left(y_m - a_0 - a_1 x_m\right)^2 \\ &= \sum_{m=1}^{M}2\left(y_m - a_0 - a_1 x_m\right)\left(-x_m\right) \\ &= -2\sum_{m=1}^{M}\left(y_m - a_0 - a_1 x_m\right)x_m\end{aligned}$$

The Fun Math

Computational Methods in EE

Step 2 – We set the derivatives to zero to find the minimum of E.

$$
\begin{aligned}
0 &= \frac{\partial E}{\partial a_0} \\
&= -2\sum_{m=1}^{M}\left(y_m - a_0 - a_1 x_m\right) \\
&= \sum_{m=1}^{M} y_m - \sum_{m=1}^{M} a_0 - \sum_{m=1}^{M} a_1 x_m \\
&= \sum_{m=1}^{M} y_m - Ma_0 - \sum_{m=1}^{M} a_1 x_m
\end{aligned}
$$

$$
\begin{aligned}
0 &= \frac{\partial E}{\partial a_1} \\
&= -2\sum_{m=1}^{M}\left(y_m - a_0 - a_1 x_m\right) x_m \\
&= \sum_{m=1}^{M}\left(y_m - a_0 - a_1 x_m\right) x_m \\
&= \sum_{m=1}^{M} y_m x_m - a_0 \sum_{m=1}^{M} x_m - \sum_{m=1}^{M} a_1 x_m^2
\end{aligned}
$$

The Fun Math

Computational Methods in EE

Step 3 – We write these as two simultaneous equations. These are called the *normal equations*.

$$
\begin{aligned}
0 &= \frac{\partial E}{\partial a_0} \\
&= -2\sum_{m=1}^{M}\left(y_m - a_0 - a_1 x_m\right) \\
&= \sum_{m=1}^{M} y_m - \sum_{m=1}^{M} a_0 - \sum_{m=1}^{M} a_1 x_m \\
&= \sum_{m=1}^{M} y_m - Ma_0 - a_1 \sum_{m=1}^{M} x_m
\end{aligned}
$$

$$\downarrow$$

$$Ma_0 + a_1 \sum_{m=1}^{M} x_m = \sum_{m=1}^{M} y_m$$

$$
\begin{aligned}
0 &= \frac{\partial E}{\partial a_1} \\
&= -2\sum_{m=1}^{M}\left(y_m - a_0 - a_1 x_m\right) x_m \\
&= \sum_{m=1}^{M}\left(y_m - a_0 - a_1 x_m\right) x_m \\
&= \sum_{m=1}^{M} y_m x_m - a_0 \sum_{m=1}^{M} x_m - \sum_{m=1}^{M} a_1 x_m^2
\end{aligned}
$$

$$\downarrow$$

$$a_0 \sum_{m=1}^{M} x_m + a_1 \sum_{m=1}^{M} x_m^2 = \sum_{m=1}^{M} y_m x_m$$

The Fun Math

Computational Methods in EE

Step 4 – The normal equations are solved simultaneously and the solution is

$$a_0 = y_{\text{avg}} - a_1 x_{\text{avg}}$$

$$a_1 = \frac{M\sum_{m=1}^{M} x_m y_m - \sum_{m=1}^{M} x_m \sum_{m=1}^{M} y_m}{M\sum_{m=1}^{M} x_m^2 - \left(\sum_{m=1}^{M} x_m\right)^2}$$

$$x_{\text{avg}} = \frac{1}{M}\sum_{m=1}^{M} x_m$$

$$y_{\text{avg}} = \frac{1}{M}\sum_{m=1}^{M} y_m$$

Yikes!
There has to be an easier way!

Computational Methods in EE

Linear Least-Squares

(Best Fit, Clean Math)

Statement of Problem

Computational Methods in EE

We wish to fit a set of M measured data points to a curve containing $N+1$ terms:

$$f = a_0 z_0 + a_1 z_1 + a_2 z_2 + \cdots + a_N z_N$$

$$\begin{aligned} f &\equiv \text{measured value} \\ z_n &\equiv \text{parameters from which } f \text{ is evaluated} \\ a_n &\equiv \text{coefficients for the curve fit} \end{aligned}$$

Curve Fitting 23

Formulation of Matrix Equation

Computational Methods in EE

We start by writing the function f for each of our M measurements. We also include the residual term.

$$\begin{aligned} f_1 &= a_0 z_{0,1} + a_1 z_{1,1} + \cdots + a_N z_{N,1} + e_1 \\ f_2 &= a_0 z_{0,2} + a_1 z_{1,2} + \cdots + a_N z_{N,2} + e_2 \\ &\vdots \\ f_M &= a_0 z_{0,M} + a_1 z_{1,M} + \cdots + a_N z_{N,M} + e_M \end{aligned}$$

This large set of equations is put into matrix form.

$$\begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_{M-1} \\ f_M \end{bmatrix} = \begin{bmatrix} z_{0,1} & z_{1,1} & \cdots & z_{N,1} \\ z_{0,2} & z_{1,2} & \cdots & z_{N,2} \\ z_{0,3} & z_{1,3} & \vdots & z_{N,3} \\ \vdots & \vdots & & \vdots \\ z_{0,M-1} & z_{1,M-1} & \cdots & z_{N,M-1} \\ z_{0,M} & z_{1,M} & \cdots & z_{N,M} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ \vdots \\ e_{M-1} \\ e_M \end{bmatrix} \Rightarrow \begin{gathered} [f] = [Z][a] + [e] \\ \text{or} \\ \mathbf{f} = \mathbf{Z}\mathbf{a} + \mathbf{e} \end{gathered}$$

Curve Fitting 24

Formulation of Solution by Least-Squares (1 of 4)

Computational Methods in EE

Step 1 – Solve matrix equation for $\mathbf{e}$.

$$\mathbf{f} = \mathbf{Z}\mathbf{a} + \mathbf{e} \quad \rightarrow \quad \mathbf{e} = \mathbf{f} - \mathbf{Z}\mathbf{a}$$

Step 2 – Calculate the error criterion E from $\mathbf{e}$.

$$E = \sum_{m=1}^{M} e_m^2 = \begin{bmatrix} e_1 & e_2 & e_3 & \cdots & e_{M-1} & e_M \end{bmatrix} \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ \vdots \\ e_{M-1} \\ e_M \end{bmatrix} = \mathbf{e}^T \mathbf{e}$$

Step 3 – We substitute our equation for $\mathbf{e}$ from Step 1 into our equation for E from Step 2.

$$E = \mathbf{e}^T \mathbf{e} = (\mathbf{f} - \mathbf{Z}\mathbf{a})^T (\mathbf{f} - \mathbf{Z}\mathbf{a})$$

Formulation of Solution by Least-Squares (2 of 4)

Computational Methods in EE

Step 4 – Our new matrix equation is algebraically manipulated as follows in order to make it easier to find its first-order derivative.

$$E = (\mathbf{f} - \mathbf{Z}\mathbf{a})^T (\mathbf{f} - \mathbf{Z}\mathbf{a}) \qquad \text{original equation}$$

$$= (\mathbf{f}^T - \mathbf{a}^T \mathbf{Z}^T)(\mathbf{f} - \mathbf{Z}\mathbf{a}) \qquad \text{distribute the transpose}$$

$$= \mathbf{f}^T \mathbf{f} \underbrace{- \mathbf{f}^T \mathbf{Z}\mathbf{a} - \mathbf{a}^T \mathbf{Z}^T \mathbf{f}}_{\text{These are scalars and transposes of each other so they are equal.}} + \mathbf{a}^T \mathbf{Z}^T \mathbf{Z}\mathbf{a} \qquad \text{expand equation}$$

$$= \mathbf{f}^T \mathbf{f} - 2\mathbf{a}^T \mathbf{Z}^T \mathbf{f} + \mathbf{a}^T \mathbf{Z}^T \mathbf{Z}\mathbf{a} \qquad \text{combine terms}$$

Formulation of Solution by Least-Squares (3 of 4)

Computational Methods in EE

Step 5 – We differential E with respect to $\mathbf{a}$.

We wish to determine $\mathbf{a}$ that minimizes E.
We can do this using the first-derivative rule.

$$E = \mathbf{f}^T\mathbf{f} - 2\mathbf{a}^T\mathbf{Z}^T\mathbf{f} + \mathbf{a}^T\mathbf{Z}^T\mathbf{Z}\mathbf{a}$$

$$\frac{\partial E}{\partial \mathbf{a}} = \frac{\partial}{\partial \mathbf{a}}\left(\mathbf{f}^T\mathbf{f} - 2\mathbf{a}^T\mathbf{Z}^T\mathbf{f} + \mathbf{a}^T\mathbf{Z}^T\mathbf{Z}\mathbf{a}\right) \qquad \text{substitute in expression for } E$$

$$= \frac{\partial}{\partial \mathbf{a}}\mathbf{f}^T\mathbf{f} - 2\frac{\partial}{\partial \mathbf{a}}\mathbf{a}^T\mathbf{Z}^T\mathbf{f} + \frac{\partial}{\partial \mathbf{a}}\mathbf{a}^T\mathbf{Z}^T\mathbf{Z}\mathbf{a} \qquad \mathbf{f} \text{ is not a function of } \mathbf{a}$$

$$= -2\mathbf{Z}^T\mathbf{f} + 2\mathbf{Z}^T\mathbf{Z}\mathbf{a} \qquad \text{finish differentiation}$$

Formulation of Solution by Least-Squares (4 of 4)

Computational Methods in EE

Step 6 – We find the value of $\mathbf{a}$ that makes the derivative equal to zero.

$$\frac{\partial E}{\partial \mathbf{a}} = -2\mathbf{Z}^T\mathbf{f} + 2\mathbf{Z}^T\mathbf{Z}\mathbf{a} = 0$$

$$-2\mathbf{Z}^T\mathbf{f} + 2\mathbf{Z}^T\mathbf{Z}\mathbf{a} = 0$$

$$2\mathbf{Z}^T\mathbf{Z}\mathbf{a} = 2\mathbf{Z}^T\mathbf{f}$$

$$\mathbf{Z}^T\mathbf{Z}\mathbf{a} = \mathbf{Z}^T\mathbf{f}$$

This is the original equation premultiplied by $\mathbf{Z}^T$.

$$\boxed{\mathbf{a} = \left(\mathbf{Z}^T\mathbf{Z}\right)^{-1}\mathbf{Z}^T\mathbf{f}}$$

DO NOT SIMPLIFY FURTHER!

Computational Methods in EE

If we were to simplify our least-squares equation, we would get

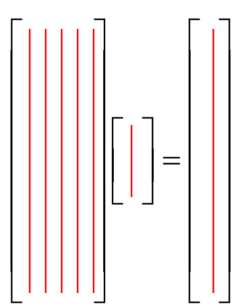
$$\begin{aligned}\mathbf{a} &= \left(\mathbf{Z}^T\mathbf{Z}\right)^{-1}\mathbf{Z}^T\mathbf{f}\\ &= \mathbf{Z}^{-1}\underbrace{\left(\mathbf{Z}^T\right)^{-1}\mathbf{Z}^T}_{\mathbf{I}}\mathbf{f}\\ &= \mathbf{Z}^{-1}\mathbf{f}\end{aligned}$$

This is just our original equation again ($\mathbf{f} = \mathbf{Za}$) without the least-squares approach incorporated.

Visualizing Least-Squares (1 of 3)

Computational Methods in EE

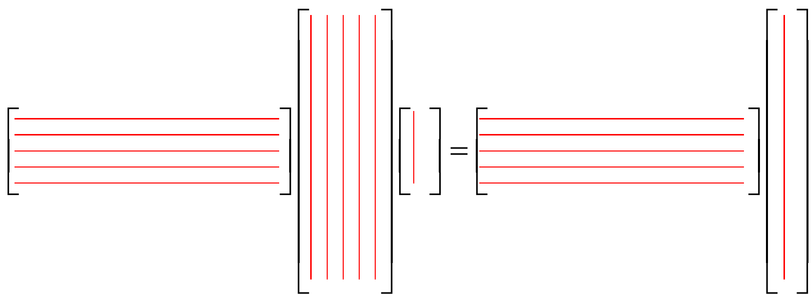
We are initially given a matrix equation with more equations than unknowns.



Visualizing Least-Squares (2 of 3)

Computational Methods in EE

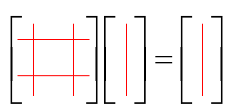
We premultiply by the transpose of $\mathbf{A}$.



Visualizing Least-Squares (3 of 3)

Computational Methods in EE

The matrix equations reduces to the same number of equations as unknowns, which is solvable by many standard algorithms.



Least-Squares Algorithm

Computational Methods in EE

Step 1 – Construct matrices. $\mathbf{Z}$ is essentially just a matrix of the coordinates of the data points. $\mathbf{f}$ is a column vector of the measurements.

$$\mathbf{f} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_{M-1} \\ f_M \end{bmatrix} \qquad \mathbf{Z} = \begin{bmatrix} z_{0,1} & z_{1,1} & \cdots & z_{N,1} \\ z_{0,2} & z_{1,2} & \cdots & z_{N,2} \\ z_{0,3} & z_{1,3} & \cdots & z_{N,3} \\ \vdots & \vdots & & \vdots \\ z_{0,M-1} & z_{1,M-1} & & z_{N,M-1} \\ z_{0,M} & z_{1,M} & & z_{N,M} \end{bmatrix}$$

Step 2 – Solve for the unknown coefficients $\mathbf{a}$.

$$\mathbf{a} = \left(\mathbf{Z}^T\mathbf{Z}\right)^{-1}\mathbf{Z}^T\mathbf{f}$$

Step 3 – Extract the coefficients from $\mathbf{a}$.

$$\mathbf{a} = \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_N \end{bmatrix}$$

Least-Squares for Solving $\mathbf{Ax} = \mathbf{b}$

Computational Methods in EE

Suppose we wish to solve $\mathbf{Ax} = \mathbf{b}$, but we have more equations than we have unknowns.

We must solve this as a “best fit” because a perfect fit is impossible in the presence of noise.

We apply least-squares by premultipling by $\mathbf{A}^T$.

$$\mathbf{Ax} = \mathbf{b} \quad \rightarrow \quad \mathbf{A}'\mathbf{x} = \mathbf{b}'$$

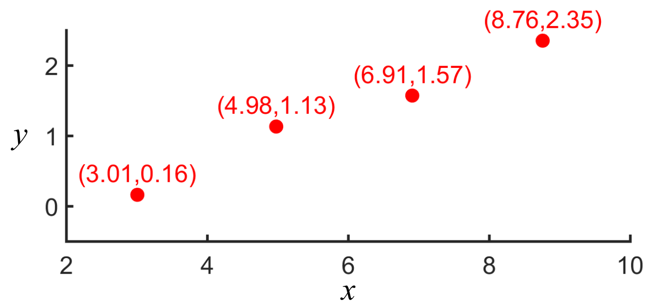
$$\mathbf{A}' = \mathbf{A}^T\mathbf{A}$$

$$\mathbf{b}' = \mathbf{A}^T\mathbf{b}$$

Example 1 (1 of 3)

Computational Methods in EE

Let's fit a line to the following set of points.



$$y = mx + b$$

Example 1 (2 of 3)

Computational Methods in EE

Step 1 – Build matrices

$$\begin{array}{l} y_1 = mx_1 + b \\ y_2 = mx_2 + b \\ y_3 = mx_3 + b \\ y_4 = mx_4 + b \end{array} \rightarrow \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ x_3 & 1 \\ x_4 & 1 \end{bmatrix} \begin{bmatrix} m \\ b \end{bmatrix} \rightarrow \mathbf{f} = \begin{bmatrix} 0.16 \\ 1.13 \\ 1.57 \\ 2.35 \end{bmatrix} \quad \mathbf{Z} = \begin{bmatrix} 3.01 & 1 \\ 4.98 & 1 \\ 6.91 & 1 \\ 8.76 & 1 \end{bmatrix}$$

With some practice, you will be able to write matrices directly from measured data.

Step 2 – Solve by least squares.

$$\mathbf{Z}^T\mathbf{Z}\mathbf{x} = \mathbf{Z}^T\mathbf{f} \rightarrow \begin{bmatrix} 3.01 & 4.98 & 6.91 & 8.76 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 3.01 & 1 \\ 4.98 & 1 \\ 6.91 & 1 \\ 8.76 & 1 \end{bmatrix} \begin{bmatrix} m \\ b \end{bmatrix} = \begin{bmatrix} 3.01 & 4.98 & 6.91 & 8.76 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0.16 \\ 1.13 \\ 1.57 \\ 2.35 \end{bmatrix}$$

$$\rightarrow \begin{bmatrix} 158.3462 & 23.6600 \\ 23.6600 & 4.0000 \end{bmatrix} \begin{bmatrix} m \\ b \end{bmatrix} = \begin{bmatrix} 37.5437 \\ 5.2100 \end{bmatrix}$$

$$\rightarrow \begin{bmatrix} m \\ b \end{bmatrix} = \begin{bmatrix} 0.3656 \\ -0.8602 \end{bmatrix}$$

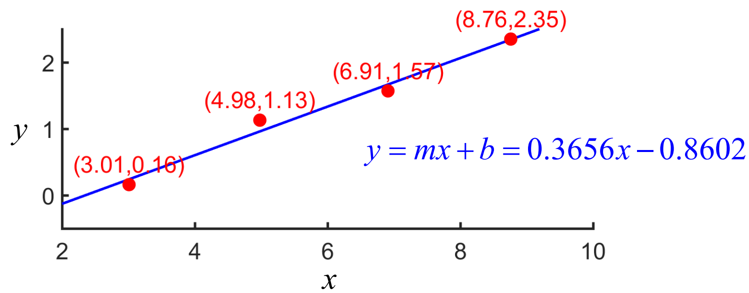
Example 1 (3 of 3)

Computational Methods in EE

Step 3 – Extract coefficients

$$\begin{bmatrix} m \\ b \end{bmatrix} = \begin{bmatrix} 0.3656 \\ -0.8602 \end{bmatrix} \rightarrow \begin{aligned} m &= 0.3656 \\ b &= -0.8602 \end{aligned}$$

Step 4 – Plot the result



Computational Methods in EE

Nonlinear Regression

(Best Fit, Moderate Math)

Statement of Problem

Computational Methods in EE

We wish to fit a set of M measured data points to a nonlinear function $f(x)$.

$$y = f\left(x; a_0, a_1, \ldots, a_N\right)$$

$$y \equiv \text{measured value}$$
$$x \equiv \text{parameter from which } f \text{ is evaluated}$$
$$a_n \equiv \text{coefficients for the function fit}$$

The function can be anything like sine's, logarithms, exponentials…

$$f(x) = A + B\sin(Cx) \qquad f(x) = A + Be^{-Cx^2} \qquad f(x) = A + B\ln(Cx)$$

Curve Fitting 39

Multiple-Parameter Taylor Series

Computational Methods in EE

Recall the Taylor series for a single parameter…

$$f(x) = f(\tilde{x}) + \frac{f'(\tilde{x})}{1!}(x-\tilde{x}) + \frac{f''(\tilde{x})}{2!}(x-\tilde{x})^2 + \frac{f'''(\tilde{x})}{3!}(x-\tilde{x})^3 + \cdots$$

The two-parameter Taylor series is

$$f(x,y) = f(\tilde{x},\tilde{y}) + \frac{1}{1!}\left[\frac{\partial f(\tilde{x},\tilde{y})}{\partial x}(x-\tilde{x}) + \frac{\partial f(\tilde{x},\tilde{y})}{\partial y}(y-\tilde{y})\right]$$
$$+ \frac{1}{2!}\left[\frac{\partial^2 f(\tilde{x},\tilde{y})}{\partial x^2}(x-\tilde{x})^2 + 2\frac{\partial^2 f(\tilde{x},\tilde{y})}{\partial x \partial y}(x-\tilde{x})(y-\tilde{y}) + \frac{\partial^2 f(\tilde{x},\tilde{y})}{\partial y^2}(y-\tilde{y})^2\right]$$
$$\vdots$$

We will ignore the higher-order terms.

The N-parameter Taylor series using only first-order derivatives is

$$f(x_1, x_2, \ldots, x_N) \approx f(\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_N) + \sum_{n=1}^{N} \frac{\partial f}{\partial x_n}\Delta x_n \qquad \Delta x_n = x_n - \tilde{x}_n$$

Curve Fitting 40

Solution Using Gauss-Newton Method (1 of 5)

Computational Methods in EE

We write our equation for each measured sample.

$$\begin{aligned} y_1 &= f\left(x_1; a_0, a_1, \ldots, a_N\right) + e_1 \\ y_2 &= f\left(x_2; a_0, a_1, \ldots, a_N\right) + e_2 \\ &\vdots \\ y_M &= f\left(x_M; a_0, a_1, \ldots, a_N\right) + e_M \end{aligned} \quad \rightarrow \quad \underbrace{\begin{aligned} y_1 &= f\left(x_1\right) + e_1 \\ y_2 &= f\left(x_2\right) + e_2 \\ &\vdots \\ y_M &= f\left(x_M\right) + e_M \end{aligned}}_{\text{Shorthand notation}}$$

Solution Using Gauss-Newton Method (2 of 5)

Computational Methods in EE

We can convert the nonlinear equations into linear equations by expanding them into multi-parameter Taylor series.

$$\begin{aligned} y_1 &= f\left(x_1\right) + \frac{\partial f\left(x_1\right)}{\partial a_0}\Delta a_0 + \frac{\partial f\left(x_1\right)}{\partial a_1}\Delta a_1 + \cdots + \frac{\partial f\left(x_1\right)}{\partial a_N}\Delta a_N + e_1 \\ y_2 &= f\left(x_2\right) + \frac{\partial f\left(x_2\right)}{\partial a_0}\Delta a_0 + \frac{\partial f\left(x_2\right)}{\partial a_1}\Delta a_1 + \cdots + \frac{\partial f\left(x_2\right)}{\partial a_N}\Delta a_N + e_2 \\ &\vdots \\ y_M &= f\left(x_M\right) + \frac{\partial f\left(x_M\right)}{\partial a_0}\Delta a_0 + \frac{\partial f\left(x_M\right)}{\partial a_1}\Delta a_1 + \cdots + \frac{\partial f\left(x_M\right)}{\partial a_N}\Delta a_N + e_M \end{aligned}$$

Solution Using Gauss-Newton Method (3 of 5)

Computational Methods in EE

Now that our equations are linear, we can put them in matrix form.

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_{M-1} \\ y_M \end{bmatrix} = \begin{bmatrix} f(x_1) \\ f(x_2) \\ f(x_3) \\ \vdots \\ f(x_{M-1}) \\ f(x_M) \end{bmatrix} + \begin{bmatrix} \frac{\partial f(x_1)}{\partial a_0} & \frac{\partial f(x_1)}{\partial a_1} & \cdots & \frac{\partial f(x_1)}{\partial a_N} \\ \frac{\partial f(x_2)}{\partial a_0} & \frac{\partial f(x_2)}{\partial a_1} & \cdots & \frac{\partial f(x_2)}{\partial a_N} \\ \frac{\partial f(x_3)}{\partial a_0} & \frac{\partial f(x_3)}{\partial a_1} & \cdots & \frac{\partial f(x_3)}{\partial a_N} \\ \vdots & \vdots & & \\ \frac{\partial f(x_{M-1})}{\partial a_0} & \frac{\partial f(x_{M-1})}{\partial a_1} & \cdots & \frac{\partial f(x_{M-1})}{\partial a_N} \\ \frac{\partial f(x_M)}{\partial a_0} & \frac{\partial f(x_M)}{\partial a_1} & \cdots & \frac{\partial f(x_M)}{\partial a_N} \end{bmatrix} \begin{bmatrix} \Delta a_0 \\ \Delta a_1 \\ \vdots \\ \Delta a_N \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ \vdots \\ e_{M-1} \\ e_M \end{bmatrix}$$

$$[y] = [f] + [Z][\Delta a] + [e]$$

Curve Fitting 43

Solution Using Gauss-Newton Method (4 of 5)

Computational Methods in EE

We solve for $[\Delta a]$ using least-squares.

$$[y] = [f] + [Z][\Delta a] + \cancel{[e]}$$

$$[y] - [f] = [Z][\Delta a]$$

$$[d] = [Z][\Delta a]$$

$$[Z]^T [d] = [Z]^T [Z][\Delta a]$$

$$[\Delta a] = \left([Z]^T [Z]\right)^{-1} [Z]^T [d]$$

Let $[d] = [y] - [f]$

$[d]$ is a new error function.
$[y]$ contains the measured values and $[f]$ contains the calculated values from the curve fit $f(x)$.

Curve Fitting 44

Solution Using Gauss-Newton Method (5 of 5)

Computational Methods in EE

We now have

$$[\Delta a] = \left([Z]^T [Z]\right)^{-1} [Z]^T [d]$$

This will only tell us how to change the coefficients $[a]$ given an initial guess $[a]_0$.

We will have to iterate to find the actual coefficients of $[a]$.

Algorithm for Nonlinear Regression (1 of 2)

Computational Methods in EE

Step 0 – Derive analytical expressions for partial derivatives of $f(x)$.

Step 1 – Make an initial guess at your coefficients.

$$[a]_0 = \begin{bmatrix} a_{0,0} & a_{1,0} & \cdots & a_{N,0} \end{bmatrix}^T$$

Make an intelligent guess!

Step 2 – Calculate the function $[f]_i$ at all measured points given the current value of the coefficients $[a]_i$.

$$[f]_i = \begin{bmatrix} f_i(x_1) & f_i(x_2) & f_i(x_3) & \cdots & f_i(x_{M-1}) & f_i(x_M) \end{bmatrix}^T$$

Step 3 – Calculate the error $[d]_i$ in the estimate of $[f]_i$.

$$[d]_i = [y] - [f]_i$$

```
% CALCULATE d
d = fm(:) - f;
```

Algorithm for Nonlinear Regression (2 of 2)

Computational Methods in EE

Step 4 – Construct the $[Z]_i$ matrix given the current coefficients $[a]_i$.

$$[Z]_i = \begin{bmatrix} \frac{\partial f_i(x_1)}{\partial a_0} & \frac{\partial f_i(x_1)}{\partial a_1} & \cdots & \frac{\partial f_i(x_1)}{\partial a_N} \\ \frac{\partial f_i(x_2)}{\partial a_0} & \frac{\partial f_i(x_2)}{\partial a_1} & \cdots & \frac{\partial f_i(x_2)}{\partial a_N} \\ \vdots & \vdots & \ddots & \\ \frac{\partial f_i(x_M)}{\partial a_0} & \frac{\partial f_i(x_M)}{\partial a_1} & & \frac{\partial f_i(x_M)}{\partial a_N} \end{bmatrix}$$

We will need to derive $N+1$ derivatives, one for each coefficient a_n.

We will need to evaluate each of these $N+1$ derivatives at all M points.

This is a lot of work!

Step 5 – Solve for $[\Delta a]_i$ using least squares.

$$[\Delta a]_i = \left([Z]_i^T [Z]_i\right)^{-1} \left([Z]_i^T [d]_i\right)$$

Algorithm for Nonlinear Regression (3 of 3)

Computational Methods in EE

Step 6 – Adjust $[a]_i$ using $[\Delta a]_i$.

$$[a]_{i+1} = [a]_i + [\Delta a]_i$$

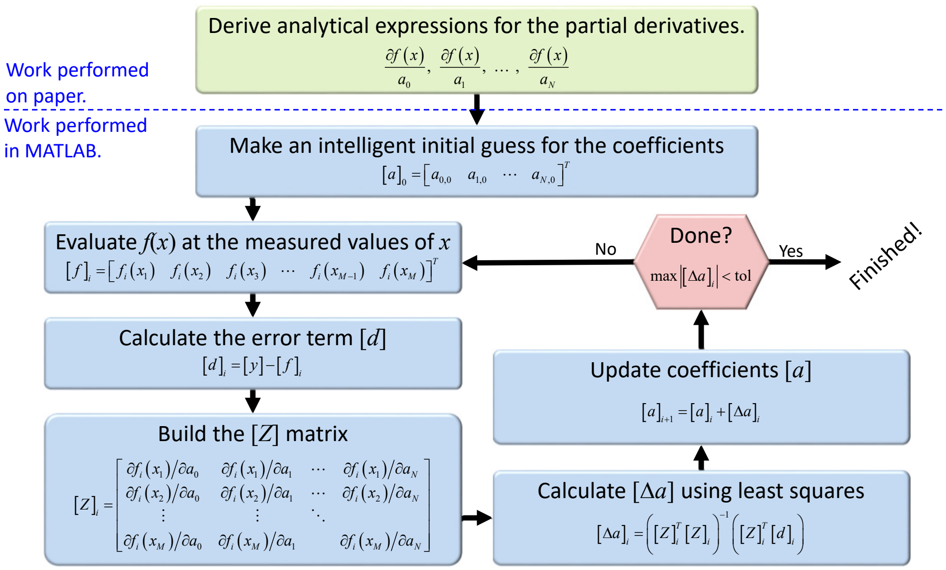
Step 7 – Go back to Step 2 until converged.

Convergence happens when the change in coefficient values $[\Delta a]_i$ falls below some tolerance.

$$\left| \frac{a_{n,i+1} - a_{n,i}}{a_{n,i+1}} \right| \cdot 100\% \le \text{tol} \qquad \text{for all } a_n$$

Block Diagram

Computational Methods in EE



Notes for Nonlinear Regression

Computational Methods in EE

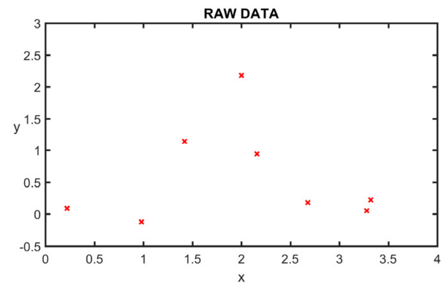
- Method can be used to fit any set of measured data to any function who's first-derivative exists.
- This method does not always converge. Try different initial guesses.
- Use the most intelligent initial guess as possible.
- Linear regression is a special case of nonlinear regression.

Example 1 – Fit to a Gaussian

Computational
Methods in EE

We wish to fit a set of measured data to a standard Gaussian function.

$$f(x) = A \exp\left[-\left(\frac{x - x_0}{\sigma}\right)^2\right]$$




```
% MEASURED DATA
xm = [ -0.14 ; 0.22 ;  0.98 ; 1.42 ; 2.00 ; 2.16 ; 2.68 ; 3.28 ; 3.32  ];
fm = [  0.01 ; 0.09 ; -0.12 ; 1.14 ; 2.18 ; 0.94 ; 0.18 ; 0.05 ; 0.22  ];
```


Example 1 – Fit to a Gaussian

Computational
Methods in EE

Formulation Step 1 – Identify the unknown parameters.

$$[a] = \begin{bmatrix} A \\ x_0 \\ \sigma \end{bmatrix}$$

In this case, we have 3 unknown parameters: A, x_0, and σ.

Example 1 – Fit to a Gaussian

Computational Methods in EE

Formulation Step 2 – Derive terms in $[Z]$ matrix.

$$\frac{\partial f(x)}{\partial A} \quad \frac{\partial f(x)}{\partial x_0} \quad \frac{\partial f(x)}{\partial \sigma}$$

$$[Z]=\begin{bmatrix} \frac{\partial f(x_1)}{\partial A} & \frac{\partial f(x_1)}{\partial x_0} & \frac{\partial f(x_1)}{\partial \sigma} \\ \frac{\partial f(x_2)}{\partial A} & \frac{\partial f(x_2)}{\partial x_0} & \frac{\partial f(x_2)}{\partial \sigma} \\ & \vdots & \\ \frac{\partial f(x_M)}{\partial A} & \frac{\partial f(x_M)}{\partial x_0} & \frac{\partial f(x_M)}{\partial \sigma} \end{bmatrix}$$

$$\frac{\partial f(x)}{\partial A} = \exp\left[-\left(\frac{x-x_0}{\sigma}\right)^2\right] = \frac{f(x)}{A}$$

$$\frac{\partial f(x)}{\partial x_0} = A\exp\left[-\left(\frac{x-x_0}{\sigma}\right)^2\right]\cdot\left\{-\frac{2}{\sigma^2}(x-x_0)\cdot(-1)\right\} = \frac{2(x-x_0)}{\sigma^2}f(x)$$

$$\frac{\partial f(x)}{\partial \sigma} = A\exp\left[-\left(\frac{x-x_0}{\sigma}\right)^2\right]\cdot\left\{-(x-x_0)^2\cdot\left(-\frac{2}{\sigma^3}\right)\right\} = \frac{2(x-x_0)^2}{\sigma^3}f(x)$$

Example 1 – Fit to a Gaussian

Computational Methods in EE

The $[Z]$ matrix is

$$[Z]=\begin{bmatrix} \frac{f(x_1)}{A} & \frac{2(x_1-x_0)}{\sigma^2}f(x_1) & \frac{2(x_1-x_0)^2}{\sigma^3}f(x_1) \\ \frac{f(x_2)}{A} & \frac{2(x_2-x_0)}{\sigma^2}f(x_2) & \frac{2(x_2-x_0)^2}{\sigma^3}f(x_2) \\ & \vdots & \\ \frac{f(x_M)}{A} & \frac{2(x_M-x_0)}{\sigma^2}f(x_M) & \frac{2(x_M-x_0)^2}{\sigma^3}f(x_M) \end{bmatrix}$$

$$=\begin{bmatrix} \frac{1}{A}f(\mathbf{x}) & \frac{2(\mathbf{x}-x_0)}{\sigma^2}f(\mathbf{x}) & \frac{2(\mathbf{x}-x_0)^2}{\sigma^3}f(\mathbf{x}) \end{bmatrix}$$

← We will calculate $[Z]$ this way.

Example 1 – Fit to a Gaussian

Computational Methods in EE

An example of a more intelligent initial guess…

$$f(x) = A\exp\left[-\left(\frac{x-x_0}{\sigma}\right)^2\right]$$

$$A_1 \leftarrow \max\left[f_i\right]$$

Consider using the maximum value of f_i in your measured points as your initial guess for A.

$$x_{0,1} \leftarrow \text{average}\left[x_i\right]$$

Consider using the average value of x_i as your initial guess for x_0.

$$\sigma \leftarrow s\left(\max\left[x_i\right] - \min\left[x_i\right]\right)$$

The standard deviation will likely be on the same order of magnitude as the range of values of x_i. Maybe choose $s = 0.5$?

```
% SMART INITIAL GUESS
A  = max(fm);
x0 = mean(xm);
s  = 0.5*(max(xm) - min(xm));
```


Example 1 – Fit to a Gaussian

Computational Methods in EE

The main loop…

Step a – Calculate $[f_i]$

$$[f]_i = \left[f_i(x_1) \quad f_i(x_2) \quad f_i(x_3) \quad \cdots \quad f_i(x_{M-1}) \quad f_i(x_M)\right]^T$$

```
% COMPUTE COLUMN VECTOR F
f = A*exp(-((xm - x0)/s).^2);
```

Here is $[f]$ over ten iterations…

0.6452	0.0393	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.9780	0.1218	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
1.7707	0.6366	0.0160	0.2244	0.0088	0.0234	0.0131	0.0141	0.0141	0.0141
2.0931	1.0530	0.4043	1.7341	0.7211	1.2323	1.1202	1.1358	1.1357	1.1357
2.1414	1.2289	1.5594	1.3029	2.3184	2.0714	2.1838	2.1829	2.1833	2.1833
2.0714	1.1582	1.2653	0.6630	1.2660	0.9323	0.9277	0.9363	0.9361	0.9362
1.6520	0.7047	0.1130	0.0124	0.0111	0.0042	0.0026	0.0028	0.0028	0.0028
1.0165	0.2228	0.0003	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.9757	0.2018	0.0002	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

Example 1 – Fit to a Gaussian

Computational Methods in EE

The main loop…

Step b – Calculate $[d_i]$

$$[d]_i = [y] - [f]_i$$

```
% CALCULATE d
d = fm(:) - f;
```

Here is $[d]$ over ten iterations…

-0.6352	-0.0293	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100	0.0100
-0.8880	-0.0318	0.0900	0.0899	0.0900	0.0900	0.0900	0.0900	0.0900	0.0900
-1.8907	-0.7566	-0.1360	-0.3444	-0.1288	-0.1434	-0.1331	-0.1341	-0.1341	-0.1341
-0.9531	0.0870	0.7357	-0.5941	0.4189	-0.0923	0.0198	0.0042	0.0043	0.0043
0.0386	0.9511	0.6206	0.8771	-0.1384	0.1086	-0.0038	-0.0029	-0.0033	-0.0033
-1.1314	-0.2182	-0.3253	0.2770	-0.3260	0.0077	0.0123	0.0037	0.0039	0.0038
-1.4720	-0.5247	0.0670	0.1676	0.1689	0.1758	0.1774	0.1772	0.1772	0.1772
-0.9665	-0.1728	0.0497	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500	0.0500
-0.7557	0.0182	0.2198	0.2200	0.2200	0.2200	0.2200	0.2200	0.2200	0.2200

Example 1 – Fit to a Gaussian

Computational Methods in EE

The main loop…

Step c – Build $[Z_i]$

$$\mathbf{z}_1 = \frac{1}{A} f(\mathbf{x}) \qquad \mathbf{z}_2 = \frac{2(\mathbf{x} - x_0)}{\sigma^2} f(\mathbf{x})$$

$$\mathbf{z}_3 = \frac{2(\mathbf{x} - x_0)^2}{\sigma^3} f(\mathbf{x})$$

$$\mathbf{Z}_i = [\mathbf{z}_1 \quad \mathbf{z}_2 \quad \mathbf{z}_3]$$

```
% FORM Z MATRIX
z1 = f/A;
z2 = 2*f.*(xm - x0)/s^2;
z3 = 2*f.*(xm - x0).^2/s^3;
Z  = [ z1 z2 z3 ];
```

$$\mathbf{Z}_1 = \begin{bmatrix} 0.2960 & -0.8230 & 0.9081 \\ 0.4486 & -1.0123 & 0.9063 \\ 0.8123 & -0.9335 & 0.4257 \\ 0.9601 & -0.4880 & 0.0984 \\ 0.9823 & 0.3307 & 0.0442 \\ 0.9502 & 0.5414 & 0.1224 \\ 0.7578 & 1.0058 & 0.5297 \\ 0.4663 & 1.0265 & 0.8966 \\ 0.4476 & 1.0114 & 0.9068 \end{bmatrix} \qquad \mathbf{Z}_{10} = \begin{bmatrix} 0.0000 & -0.0000 & 0.0000 \\ 0.0000 & -0.0000 & 0.0000 \\ 0.0042 & -0.1944 & 0.4552 \\ 0.3352 & -6.9938 & 7.3115 \\ 0.6445 & 8.5245 & 5.6503 \\ 0.2763 & 6.2539 & 7.0924 \\ 0.0008 & 0.0436 & 0.1163 \\ 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 \end{bmatrix}$$

Example 1 – Fit to a Gaussian

Computational Methods in EE

The main loop…

Step d – Solve for $[\Delta a]_i$

$$[\Delta a]_i = \left([Z]_i^T [Z]_i\right)^{-1} \left([Z]_i^T [d]_i\right)$$

```
% CALCULATE [Da]
da = (Z'*Z)\(Z'*d);
```

Here is $[\Delta a]$ over ten iterations…

-0.9316	0.3327	0.7433	0.5188	0.3549	0.2022	-0.0135	0.0010	-0.0000	0.0000
0.0958	0.0823	-0.2859	0.1775	-0.0723	0.0092	-0.0006	0.0000	-0.0000	0.0000
-0.6519	-0.6268	-0.0058	-0.0881	-0.0027	-0.0172	0.0022	-0.0001	0.0000	-0.0000

Example 1 – Fit to a Gaussian

Computational Methods in EE

The main loop…

Step e – Update coefficients $[a]_i$

$$[a]_{i+1} = [a]_i + [\Delta a]_i$$

```
% UPDATE COEFFICIENTS [a]
A  = A + da(1);
x0 = x0 + da(2);
s  = s + da(3);
```

Here is $[a] = [A \;\; x_0 \;\; \sigma]^T$ over ten iterations…

1.2484	1.5810	2.3244	2.8432	3.1981	3.4003	3.3868	3.3878	3.3878	3.3878
1.8647	1.9470	1.6611	1.8386	1.7663	1.7755	1.7749	1.7750	1.7750	1.7750
1.0781	0.4513	0.4454	0.3574	0.3546	0.3374	0.3396	0.3395	0.3395	0.3395

Example 1 – Fit to a Gaussian

Computational Methods in EE

The main loop…

Step f – Calculate error and check for convergence

$$\varepsilon = \max\left[\left|\frac{a_{n,i+1} - a_{n,i}}{a_{n,i+1}}\right|\right]$$

```
% CALCULATE MAXIMUM ERROR
err = max(abs(da./[A;x0;s]));
```

Here is `err` over ten iterations…

```
0.7463   1.3889   0.3198   0.2465   0.1110   0.0595   0.0065   0.0003   0.0000   0.0000
```

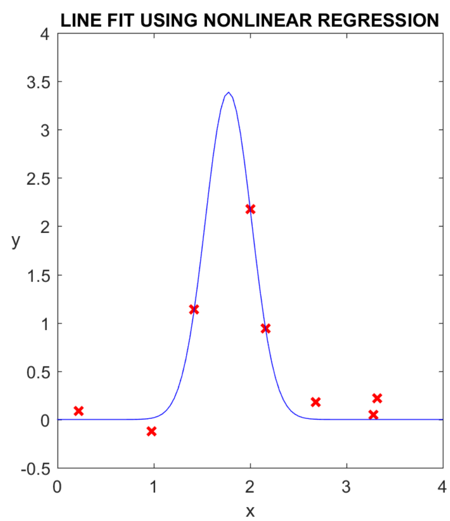

Example 1 – Fit to a Gaussian

Computational Methods in EE

The final answer is…

$$A = 3.3878$$
$$x_0 = 1.7750$$
$$\sigma = 0.3395$$

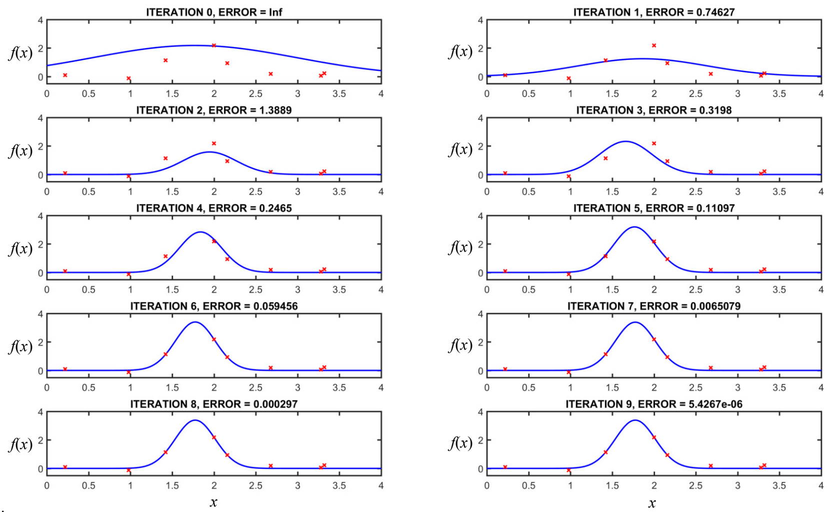
$$f(x) = A\exp\left[-\left(\frac{x - x_0}{\sigma}\right)^2\right]$$
$$= 3.3878\exp\left[-\left(\frac{x - 1.7750}{0.3395}\right)^2\right]$$



Example 1 – Fit to a Gaussian

Computational Methods in EE

Here are the first 10 iterations of the algorithm visualized…



Computational Methods in EE

Fitting Polynomials (Exact Fit, Clean Math)

Statement of the Problem

Computational Methods in EE

Suppose we wish to fit the following Nth-order polynomial to a set of points.

$$f(x) \approx a_0 + a_1 x + a_2 x^2 + \cdots + a_N x^N$$

We have $N+1$ unknown coefficients so we need $N+1$ points to calculate the coefficients exactly.

Calculating the Unknown Polynomial Coefficients

Computational Methods in EE

First, we write the polynomial at $N+1$ points.

$$\begin{aligned} f(x_1) &= a_0 + a_1 x_1 + a_2 x_1^2 + \cdots + a_N x_1^N \\ f(x_2) &= a_0 + a_1 x_2 + a_2 x_2^2 + \cdots + a_N x_2^N \\ &\vdots \\ f(x_{N+1}) &= a_0 + a_1 x_{N+1} + a_2 x_{N+1}^2 + \cdots + a_N x_{N+1}^N \end{aligned}$$

Since we are doing an exact fit, there are no residual terms.

Calculating the Unknown Polynomial Coefficients

Computational Methods in EE

Second, we put the equations into matrix form.

$$f(x_1) = a_0 + a_1 x_1 + a_2 x_1^2 + \cdots + a_N x_1^N$$
$$f(x_2) = a_0 + a_1 x_2 + a_2 x_2^2 + \cdots + a_N x_2^N$$
$$\vdots$$
$$f(x_{N+1}) = a_0 + a_1 x_{N+1} + a_2 x_{N+1}^2 + \cdots + a_N x_{N+1}^N$$

$$\Rightarrow \begin{bmatrix} f(x_1) \\ f(x_2) \\ f(x_3) \\ \vdots \\ f(x_{N+1}) \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 & & x_1^N \\ 1 & x_2 & x_2^2 & \cdots & x_2^N \\ 1 & x_3 & x_3^2 & & x_3^N \\ & \vdots & & \ddots & \\ 1 & x_{N+1} & x_{N+1}^2 & & x_{N+1}^N \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_N \end{bmatrix}$$

This can be written more compactly as

$$[f] = [X][a]$$

Calculating the Unknown Polynomial Coefficients

Computational Methods in EE

Last, the matrix equation is solved for $[a]$ to find the polynomial coefficients.

$$[a] = [X]^{-1}[f]$$

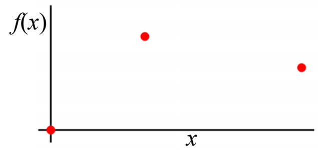
$$[a] = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_N \end{bmatrix}$$

Example 1 (1 of 2)

Computational Methods in EE

Fit the following points to a polynomial..

$$f(0)=0$$
$$f(1.5)=1.5$$
$$f(4.0)=1.0$$



Step 1 – Determine order of polynomial

Since we have three points, we can fit a quadratic polynomial.

$$f(x)\approx a_0+a_1x+a_2x^2$$

Example 1 (2 of 2)

Computational Methods in EE

Step 2 – Our matrix equation is

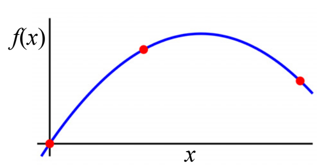
$$\begin{aligned} f_1 &= a_0+a_1x_1+a_2x_1^2 \\ f_2 &= a_0+a_1x_2+a_2x_2^2 \\ f_3 &= a_0+a_1x_3+a_2x_3^2 \end{aligned} \rightarrow \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} \rightarrow \begin{bmatrix} 0 \\ 1.5 \\ 1.0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1.5 & 2.25 \\ 1 & 4 & 16 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix}$$

Step 3 – Calculate unknown polynomial coefficients.

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1.5 & 2.25 \\ 1 & 4 & 16 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 1.5 \\ 1.0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1.45 \\ -0.3 \end{bmatrix}$$

Step 4 – Write the final polynomial.

$$f(x)\approx 1.45x-0.3x^2$$



Computational Methods in EE

Interpolation & Extrapolation

What is Interpolation & Extrapolation?

Computational Methods in EE

Sometimes we have some several precisely known values and want to know an intermediate value (interpolation) or a value outside of the range of the known values (extrapolation).

Measuring the new value may be difficult, expensive, or impossible.

Interpolation and extrapolation can be thought of as two steps:

Step 1 – Fit data to a curve.
Step 2 – Use the curve fit to calculate the new value(s).

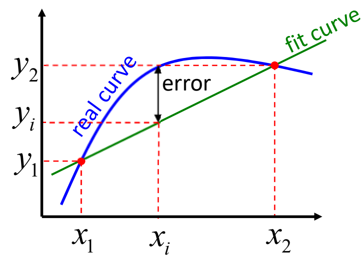
Linear Interpolation

Computational Methods in EE

Given two data points, we can fit a line from which we can interpolate or extrapolate anything else.

Given two data points (x_1, y_1) and (x_2, y_2), the equation for the line is

$$y_i = y_1 + \underbrace{\frac{y_2 - y_1}{x_2 - x_1}}_{\text{slope}}(x_i - x_1)$$

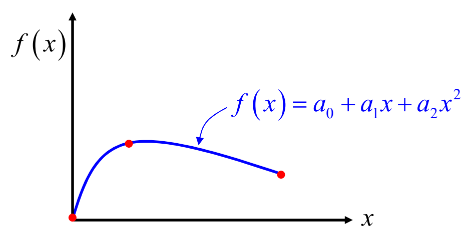


Polynomial Interpolation (1 of 3)

Computational Methods in EE

Interpolation using a quadratic polynomial is likely the most common method of interpolation.

This requires three points.



Polynomial Interpolation (2 of 3)

Computational Methods in EE

Write form a matrix equation.

$$\begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix}$$

Solve for unknown coefficients.

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \end{bmatrix}^{-1} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} = \begin{bmatrix} \frac{x_2 x_3}{(x_1 - x_2)(x_1 - x_3)} & -\frac{x_1 x_3}{(x_1 - x_2)(x_2 - x_3)} & \frac{x_1 x_2}{(x_1 - x_3)(x_2 - x_3)} \\ -\frac{x_2 + x_3}{(x_1 - x_2)(x_1 - x_3)} & \frac{x_1 + x_3}{(x_1 - x_2)(x_2 - x_3)} & -\frac{x_1 + x_2}{(x_1 - x_3)(x_2 - x_3)} \\ \frac{1}{(x_1 - x_2)(x_1 - x_3)} & -\frac{1}{(x_1 - x_2)(x_2 - x_3)} & \frac{1}{(x_1 - x_3)(x_2 - x_3)} \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix}$$

Polynomial Interpolation (3 of 3)

Computational Methods in EE

The final equations are

$$a_0 = \frac{x_2 x_3}{(x_1 - x_2)(x_1 - x_3)} f_1 - \frac{x_1 x_3}{(x_1 - x_2)(x_2 - x_3)} f_2 + \frac{x_1 x_2}{(x_1 - x_3)(x_2 - x_3)} f_3$$

$$a_1 = -\frac{x_2 + x_3}{(x_1 - x_2)(x_1 - x_3)} f_1 + \frac{x_1 + x_3}{(x_1 - x_2)(x_2 - x_3)} f_2 - \frac{x_1 + x_2}{(x_1 - x_3)(x_2 - x_3)} f_3$$

$$a_2 = \frac{1}{(x_1 - x_2)(x_1 - x_3)} f_1 - \frac{1}{(x_1 - x_2)(x_2 - x_3)} f_2 + \frac{1}{(x_1 - x_3)(x_2 - x_3)} f_3$$

Typically, these are not calculated using these equations. Instead, the matrix inversion is performed numerically.

Example 1 – Quadratic Interpolation

Computational Methods in EE

Given the following points, interpolate the value at x = 3.

$$f(0)=0$$
$$f(1.5)=1.5$$
$$f(4.0)=1.0$$

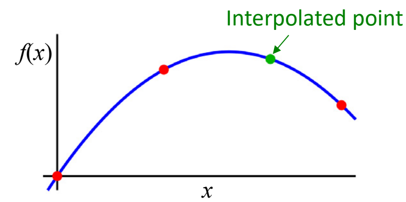
Solution

Step 1 – We already fit these points to a quadratic polynomial. The answer was

$$f(x)=1.45x-0.3x^2$$

Step 2 – We evaluate this at x = 3.

$$f(3)=1.45(3)-0.3(3)^2=\boxed{1.65}$$



Example 2 – Quadratic Extrapolation

Computational Methods in EE

Given the following points, extrapolate the value at x = -0.5.

$$f(0)=0$$
$$f(1.5)=1.5$$
$$f(4.0)=1.0$$

Solution

Step 1 – We already fit these points to a quadratic polynomial. The answer was

$$f(x)=1.45x-0.3x^2$$

Step 2 – We evaluate this at x = -0.5.

$$f(3)=1.45(-0.5)-0.3(-0.5)^2=\boxed{-0.8}$$