System of linear equations:

$$
\begin{aligned}
a_{00}x_0 + a_{01}x_1 + a_{02}x_2 + \cdots + a_{0,N-1}x_{N-1} &= b_0 \\
a_{10}x_0 + a_{11}x_1 + a_{12}x_2 + \cdots + a_{1,N-1}x_{N-1} &= b_1 \\
a_{20}x_0 + a_{21}x_1 + a_{22}x_2 + \cdots + a_{2,N-1}x_{N-1} &= b_2 \\
\cdots \qquad\qquad\qquad\qquad &\quad \cdots \\
a_{M-1,0}x_0 + a_{M-1,1}x_1 + \cdots + a_{M-1,N-1}x_{N-1} &= b_{M-1}
\end{aligned}
$$

Here the N unknowns x_j, $j = 0, 1, \ldots, N-1$ are related by M equations. The coefficients a_{ij} with $i = 0, 1, \ldots, M-1$ and $j = 0, 1, \ldots, N-1$ are known numbers, as are the *right-hand side* quantities b_i, $i = 0, 1, \ldots, M-1$.

Matrix-vector notation:

$$
\begin{aligned}
a_{00}x_0 + a_{01}x_1 + a_{02}x_2 + \cdots + a_{0,N-1}x_{N-1} &= b_0 \\
a_{10}x_0 + a_{11}x_1 + a_{12}x_2 + \cdots + a_{1,N-1}x_{N-1} &= b_1 \\
a_{20}x_0 + a_{21}x_1 + a_{22}x_2 + \cdots + a_{2,N-1}x_{N-1} &= b_2 \\
\cdots \qquad\qquad\qquad\qquad &\quad \cdots \\
a_{M-1,0}x_0 + a_{M-1,1}x_1 + \cdots + a_{M-1,N-1}x_{N-1} &= b_{M-1}
\end{aligned}
$$

$$
\mathbf{A} = \begin{bmatrix} a_{00} & a_{01} & \cdots & a_{0,N-1} \\ a_{10} & a_{11} & \cdots & a_{1,N-1} \\ & \cdots & & \\ a_{M-1,0} & a_{M-1,1} & \cdots & a_{M-1,N-1} \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} b_0 \\ b_1 \\ \cdots \\ b_{M-1} \end{bmatrix}
$$

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$$

We start with: $N = M$

LU decomposition:

Suppose we are able to write the matrix $\mathbf{A}$ as a product of two matrices,

$$\mathbf{L} \cdot \mathbf{U} = \mathbf{A} \tag{2.3.1}$$

where $\mathbf{L}$ is *lower triangular* (has elements only on the diagonal and below) and $\mathbf{U}$ is *upper triangular* (has elements only on the diagonal and above). For the case of a 4×4 matrix $\mathbf{A}$, for example, equation (2.3.1) would look like this:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ \alpha_{10} & 1 & 0 & 0 \\ \alpha_{20} & \alpha_{21} & 1 & 0 \\ \alpha_{30} & \alpha_{31} & \alpha_{32} & 1 \end{bmatrix} \cdot \begin{bmatrix} \beta_{00} & \beta_{01} & \beta_{02} & \beta_{03} \\ 0 & \beta_{11} & \beta_{12} & \beta_{13} \\ 0 & 0 & \beta_{22} & \beta_{23} \\ 0 & 0 & 0 & \beta_{33} \end{bmatrix} = \begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{bmatrix}$$

$$\mathbf{L} \cdot \mathbf{y} = \mathbf{b}$$

$$\mathbf{A} \cdot \mathbf{x} = (\mathbf{L} \cdot \mathbf{U}) \cdot \mathbf{x} = \mathbf{L} \cdot (\mathbf{U} \cdot \mathbf{x}) = \mathbf{b}$$

$$\mathbf{U} \cdot \mathbf{x} = \mathbf{y}$$

LU decomposition including complexity (O())

Gaussian elimination is also $O(N^3)$

Advantage of LU decomposition when comparing to Gaussian elimination ?

Assume we have to solve a system of linear equations with the **same matrix** and a **new right hand side.**

Then we only need to do forward and back substitutions (assuming we stored L and U). Hence $O(N^2)$ rather than $O(N^3)$.

LU decomposition:

For each $j = 0, 1, 2, \ldots, N-1$ do these two procedures:

$$\text{for } i = 0, 1, \ldots, j \qquad \beta_{ij} = a_{ij} - \sum_{k=0}^{i-1} \alpha_{ik}\beta_{kj}$$

$O(N^3)$

$$\text{for } i = j+1, j+2, \ldots, N-1 \qquad \alpha_{ij} = \frac{1}{\beta_{jj}}\left(a_{ij} - \sum_{k=0}^{j-1} \alpha_{ik}\beta_{kj}\right)$$

Forward substitution:

$$\mathbf{L} \cdot \mathbf{y} = \mathbf{b}$$

$$y_0 = \frac{b_0}{\cancel{\alpha_{00}}}$$

$$y_i = \cancel{\frac{1}{\alpha_{ii}}}\left[b_i - \sum_{j=0}^{i-1} \alpha_{ij} y_j\right] \qquad i = 1, 2, \ldots, N-1$$

$O(N^2)$

Back substitution:

$$\mathbf{U} \cdot \mathbf{x} = \mathbf{y}$$

$$x_{N-1} = \frac{y_{N-1}}{\beta_{N-1,N-1}}$$

$O(N^2)$

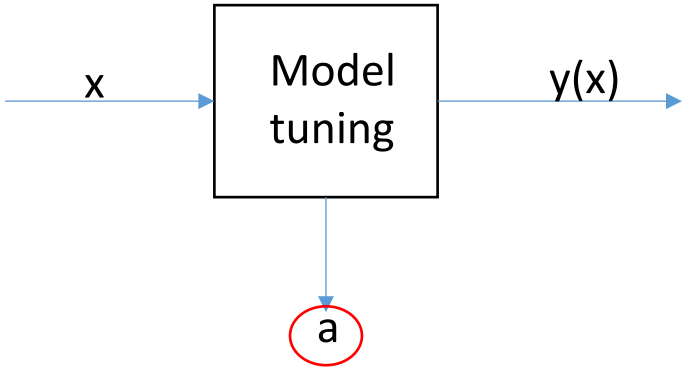
$$x_i = \frac{1}{\beta_{ii}}\left[y_i - \sum_{j=i+1}^{N-1} \beta_{ij} x_j\right] \qquad i = N-2, N-3, \ldots, 0$$

Application:Linear least squares problems:

Input-output relation

N data points (x_i, y_i), $i = 0, \ldots, N-1$

Model calibration



Model: $$y(x) = \sum_{k=0}^{M-1} a_k X_k(x)$$

$X_0(x), \ldots, X_{M-1}(x)$ are arbitrary fixed functions of x, called the *basis functions.*

M adjustable parameters a_j, $j = 0, \ldots, M-1$

Example:

Data generated by introducing fluctuations to ground truth parabola: $a_0 = 1; \quad a_1 = 1; \quad a_2 = 0.5$

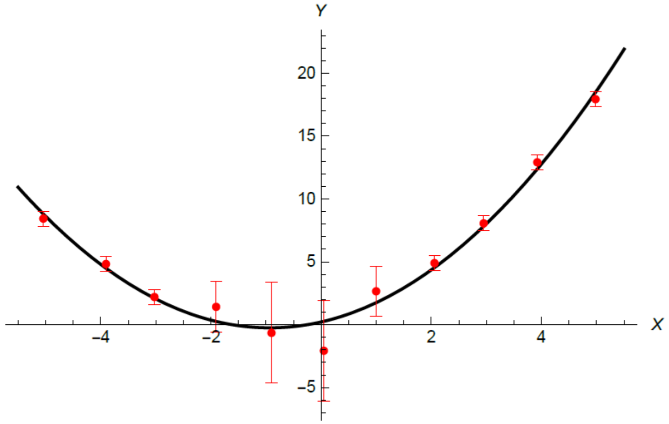
Example:

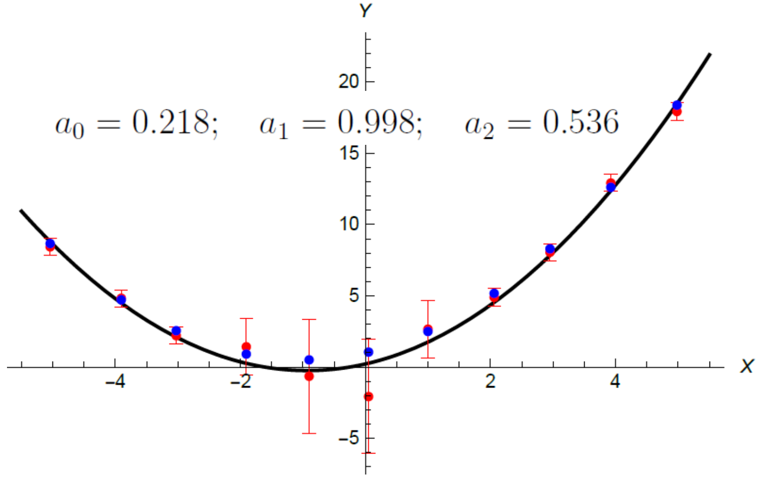
Fitting a parabola to a set of data points

$y(x) = a_0 + a_1 x + a_2 x^2$

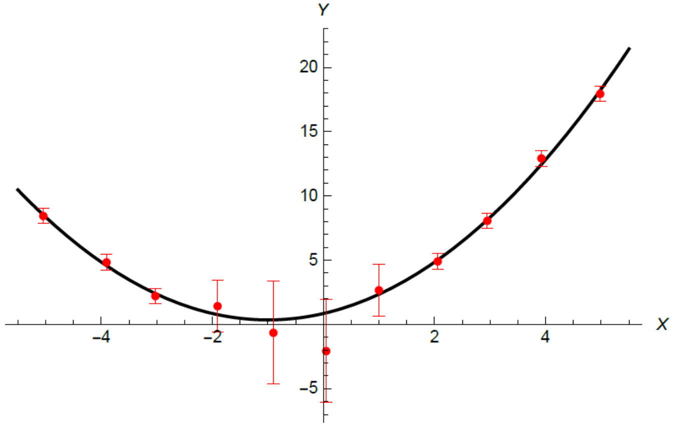
Btw: Data points must have accuracy info.

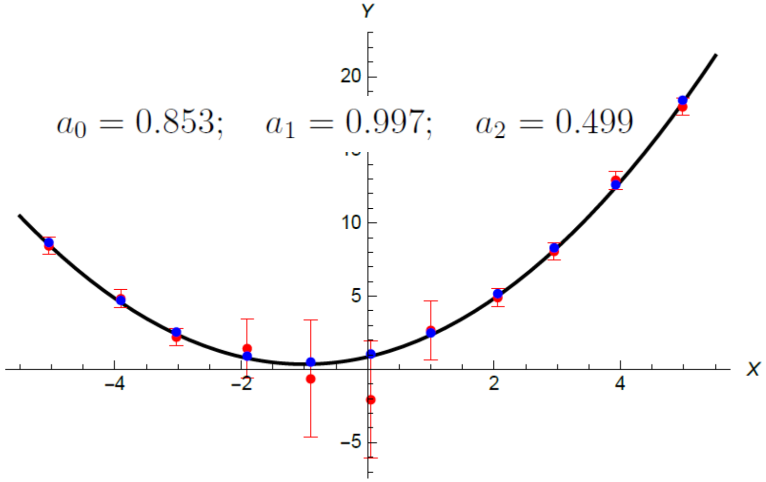
$$\min \sum_{i=0}^{N-1} \left[a_0 + a_1 x_i + a_2 x_i^2 - y_i\right]^2$$





$$\min \sum_{i=0}^{N-1} \left[\frac{a_0 + a_1 x_i + a_2 x_i^2 - y_i}{\sigma_i}\right]^2$$





From statistics, we hence know that we must **minimize**:

$$\chi^2 = \sum_{i=0}^{N-1} \left[\frac{y_i - \sum_{k=0}^{M-1} a_k X_k(x_i)}{\sigma_i} \right]^2$$

σ_i is the measurement error (standard deviation) of the ith data point

First order conditions are

$$\frac{\partial \chi^2}{\partial a_k} = 0 \quad k = 0, \ldots, M-1$$

from which we get

$$0 = \sum_{i=0}^{N-1} \frac{1}{\sigma_i^2} \left[y_i - \sum_{j=0}^{M-1} a_j X_j(x_i) \right] X_k(x_i) \qquad k = 0, \ldots, M-1 \qquad (15.4.6)$$

$$0 = \sum_{i=0}^{N-1} \frac{1}{\sigma_i^2} \left[y_i - \sum_{j=0}^{M-1} a_j X_j(x_i) \right] X_k(x_i) \qquad k = 0, \ldots, M-1 \qquad (15.4.6)$$

Interchanging the order of summations, we can write (15.4.6) as the matrix equation

$$\sum_{j=0}^{M-1} \alpha_{kj} a_j = \beta_k \qquad (15.4.7)$$

where $\alpha_{kj} = \sum_{i=0}^{N-1} \frac{X_j(x_i) X_k(x_i)}{\sigma_i^2}$ and $\beta_k = \sum_{i=0}^{N-1} \frac{y_i X_k(x_i)}{\sigma_i^2}$

We can now formulate this with a ”design matrix” A and a right hand side b

$$A_{ij} = \frac{X_j(x_i)}{\sigma_i} \qquad b_i = \frac{y_i}{\sigma_i} \qquad \mathbf{a} \equiv (a_0, \ldots, a_{M-1})$$

$$\alpha_{kj} = \sum_{i=0}^{N-1} [A^T]_{ki} [A]_{ij} \qquad \beta_k = \sum_{i=0}^{N-1} [A^T]_{ki} b_i$$

$$(\mathbf{A}^T \cdot \mathbf{A}) \cdot \mathbf{a} = \mathbf{A}^T \cdot \mathbf{b}$$

”Normal equations”

Design matrix A (and right hand side b):

$$A_{ij} = \frac{X_j(x_i)}{\sigma_i}$$

$$b_i = \frac{y_i}{\sigma_i}$$

$$\begin{array}{cc}
 & \begin{matrix} X_0(\quad) & X_1(\quad) & \cdots & X_{M-1}(\quad) \end{matrix} \\
 & \longleftarrow \text{Basis functions} \longrightarrow \\
\text{data points} \updownarrow \begin{matrix} x_0 \\[2ex] x_1 \\[2ex] \vdots \\[2ex] x_{N-1} \end{matrix} &
\begin{pmatrix}
\frac{X_0(x_0)}{\sigma_0} & \frac{X_1(x_0)}{\sigma_0} & \cdots & \frac{X_{M-1}(x_0)}{\sigma_0} \\
\frac{X_0(x_1)}{\sigma_1} & \frac{X_1(x_1)}{\sigma_1} & \cdots & \frac{X_{M-1}(x_1)}{\sigma_1} \\
\vdots & & & \vdots \\
\frac{X_0(x_{N-1})}{\sigma_{N-1}} & \frac{X_1(x_{N-1})}{\sigma_{N-1}} & \cdots & \frac{X_{M-1}(x_{N-1})}{\sigma_{N-1}}
\end{pmatrix}
\end{array}$$

Algorithm for Linear Least Square Problems (Normal Equations):

N data points $(x_i, y_i), i = 0, \ldots, N-1$

Model: $$y(x) = \sum_{k=0}^{M-1} a_k X_k(x)$$

$$A_{ij} = \frac{X_j(x_i)}{\sigma_i} \qquad b_i = \frac{y_i}{\sigma_i} \qquad \mathbf{a} \equiv (a_0, \ldots, a_{M-1})$$

$$\mathbf{C} \equiv \mathbf{A}^T \cdot \mathbf{A}$$

$$\mathbf{c} \equiv \mathbf{A}^T \cdot \mathrm{b}$$

Solve $\mathbf{C} \cdot \mathbf{a} = \mathbf{c}$

Symmetric and positive (semi-)definite matrices

An $N \times N$ matrix $\mathbf{B}$ is called *symmetric* if $B_{ij} = B_{ji}$ for all i, j.

A symmetric $N \times N$ matrix $\mathbf{B}$ is called *positive definite* if $\mathbf{v} \cdot \mathbf{B} \cdot \mathbf{v} > 0$ for all **nonzero** vectors $\mathbf{v} \in \mathbb{R}^N$. (It is called *positive semi-definite* if $\mathbf{v} \cdot \mathbf{B} \cdot \mathbf{v} \geq 0$ for all **nonzero** vectors $\mathbf{v} \in \mathbb{R}^N$.

Consider now the matrix $\mathbf{C} = \mathbf{A}^T \mathbf{A}$ from the Normal equations. We now get that

$$C_{kj} = \sum_{i=0}^{M-1} [A^T]_{ki} [A]_{ij} = \sum_{i=0}^{M-1} [A]_{ik} [A]_{ij}$$

$$\begin{aligned}
\mathbf{v} \cdot \mathbf{C} \cdot \mathbf{v} &= \sum_{k=0}^{N-1} v_k \left[\sum_{j=0}^{N-1} C_{kj} v_j \right] \\
&= \sum_{k=0}^{N-1} v_k \left[\sum_{j=0}^{N-1} \left(\sum_{i=0}^{M-1} [A]_{ik} [A]_{ij} \right) v_j \right] \\
&= \sum_{i=0}^{M-1} \left[\sum_{k=0}^{N-1} A_{ik} v_k \right] \left[\sum_{j=0}^{N-1} A_{ij} v_j \right] \\
&= \sum_{i=0}^{M-1} \left[\sum_{k=0}^{N-1} A_{ik} v_k \right]^2 \geq 0
\end{aligned}$$

Hence $\mathbf{C}$ is at least positive semi-definite. Mostly, it will be positive definite, and you may assume this to begin with.

Cholesky decomposition (for symmetric and pos. def. matrices):

Assume that $\mathbf{A}$ is symmetric and positive definite. We now first write the LU decomposition of A:

$$\mathbf{A} = \mathbf{L}^*\mathbf{U}^* \equiv \mathbf{L}^*\mathbf{D}\mathbf{D}\mathbf{V}$$

where $\mathbf{D} = \text{diag}\{\sqrt{U^*_{ii}}\}$ contains the square roots of the diagonal elements of $\mathbf{U}^*$ (when $\mathbf{A}$ is positive definite, it can be shown (we skip that) that all diagonal elements in $\mathbf{U}^*$ are positive). Hence, V is an upper triangular matrix with $V_{ii} = 1$ for all i.

Since the LU-matrices are unique $\mathbf{L}^*, \mathbf{D}, \mathbf{V}$ are also unique. We now use that $\mathbf{A}$ is symmetric to obtain

$$\mathbf{A} = \mathbf{A}^T = \mathbf{V}^\mathbf{T}\mathbf{D}\mathbf{D}(\mathbf{L}^*)^\mathbf{T}$$

As $\mathbf{L}^*, \mathbf{D}, \mathbf{V}$ are unique, we get $\mathbf{V} = (\mathbf{L}^*)^\mathbf{T}$ and hence

$$\mathbf{A} = \mathbf{L}^*\mathbf{D}\mathbf{D}(\mathbf{L}^*)^T$$

Hence, we can uniquely write

$$\mathbf{A} = \mathbf{L}^*\mathbf{D}(\mathbf{L}^*\mathbf{D})^T \equiv \mathbf{L}\mathbf{L}^T$$

where $\mathbf{L} = \mathbf{L}^*\mathbf{D}$ is lower triangular, but *no longer* with $L_{ii} = 1$

Cholesky decomposition (for symmetric and pos. def. matrices):

Similar to LU decomposition:

$$\mathbf{L} \cdot \mathbf{L}^T = \mathbf{A}$$

For i=0 to N-1

$$L_{ii} = \left(a_{ii} - \sum_{k=0}^{i-1} L_{ik}^2 \right)^{1/2}$$

$$L_{ji} = \frac{1}{L_{ii}} \left(a_{ij} - \sum_{k=0}^{i-1} L_{ik} L_{jk} \right) \qquad j = i+1, i+2, \ldots, N-1$$

Cholesky decomposition vs. LU decomposition:

- Cholesky decomposition can only be used for symmetric and positive definite matrices.
- Compared to LU decomposition, Cholesky decomposition is approximately twice as fast and requires only half the storage
- Pivoting is unnecessary and should NEVER be carried out
- Cholesky decomposition is numerically more stable than LU decomposition (reason is subtle and beyond the scope of NR)

We must still discuss the following:

- How do we compute the accuracy on a solution x to the linear equation Ax=b ?

(We define the accuracy on the solution x as ||x-x_True|| where x_True is the true solution. Unfortunately, we don't know x_True)

Exercise:

Two data sets from real applications will be provided by Jens

Establish the design matrix $A_{ij} = \frac{X_j(x_i)}{\sigma_i}$ and right hand side $b_i = \frac{y_i}{\sigma_i}$ for each of the two applications

Establish the Normal equations and solve these with Cholesky decomposition.

Compare your results to the results and error bounds stated on the web pages.