Stats recap

Video lecture 1

Sacha Epskamp

SEM 1: Confirmatory Factor Analysis

Scalars, Vectors and Matrices

Normal faced letters indicate scalars (just a single number), **boldfaced** letters indicate *column vectors*, and **UPPERCASE BOLDFACED** letters indicate *matrices*. If a scalar is an element of a matrix, the first subscript will denote the *row* and the second subscript the *column*. If a vector is part of a matrix, the subscript will denote either which row or column (the vector is always transformed to a column-vector).

$$\boldsymbol{Y} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \end{bmatrix}$$

$$\boldsymbol{y}_2 = \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix}$$

$$y_{23} = 6$$

Roman and Greek letters

A Roman letter indicates something that is *observed* (e.g., observed scores $\boldsymbol{y}_i$ of subject i) or *specified* (e.g., sample size n).
A Greek letter indicates something that is *not observed*, e.g., latent variable scores $\boldsymbol{\eta}_i$ or factor loadings matrix $\boldsymbol{\Lambda}$.

GREEK ALPHABET

By Ben Crowder • bencrowder.net • Last modified 2 May 2012

Αα	Ββ	Γγ	Δδ	Εε	Ζζ
ALPHA [a]	**BETA** [b]	**GAMMA** [g]	**DELTA** [d]	**EPSILON** [e]	**ZETA** [dz]
ἄλφα	*βῆτα*	*γάμμα*	*δέλτα*	*ἒ ψιλόν*	*ζῆτα*
Ηη	Θθ	Ιι	Κκ	Λλ	Μμ
ETA [ɛː]	**THETA** [tʰ]	**IOTA** [i]	**KAPPA** [k]	**LAMBDA** [l]	**MU** [m]
ἦτα	*θῆτα*	*ἰῶτα*	*κάππα*	*λάμβδα*	*μῦ*
Νν	Ξξ	Οο	Ππ	Ρρ	Σσς
NU [n]	**XI** [ks]	**OMICRON** [o]	**PI** [p]	**RHO** [r]	**SIGMA** [s]
νῦ	*ξεῖ*	*ὂ μικρόν*	*πεῖ*	*ῥῶ*	*σῖγμα*
Ττ	Υυ	Φφ	Χχ	Ψψ	Ωω
TAU [t]	**UPSILON** [ʉ]	**PHI** [pʰ]	**CHI** [kʰ]	**PSI** [ps]	**OMEGA** [ɔː]
ταῦ	*ὖ ψιλόν*	*φεῖ*	*χεῖ*	*ψεῖ*	*ὦ μέγα*

Transpose

The transpose operator, $\top$, switches rows and columns:

$$\boldsymbol{Y} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \end{bmatrix}$$

$$\boldsymbol{Y}^\top = \begin{bmatrix} 1 & 4 & 7 & 10 \\ 2 & 5 & 8 & 11 \\ 3 & 6 & 9 & 12 \end{bmatrix}$$

$$\boldsymbol{y}_2^\top = \begin{bmatrix} 4 & 5 & 6 \end{bmatrix}$$

Matrix Multiplication

If $\boldsymbol{AB} = \boldsymbol{C}$, with $\boldsymbol{A}$ being a $x \times y$ matrix and $\boldsymbol{B}$ being a $y \times z$ matrix, then $\boldsymbol{C}$ is a $x \times z$ matrix with following elements:

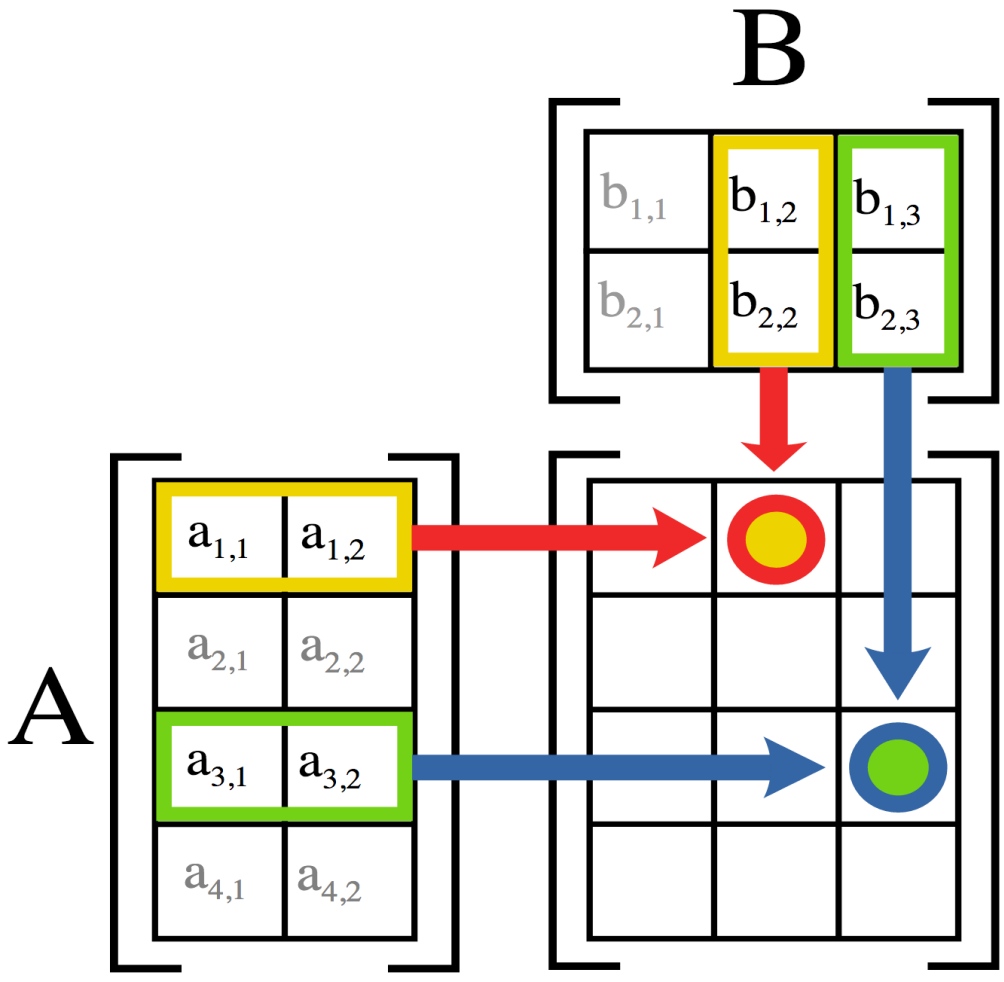
$$c_{ij} = \sum_{k=1}^{y} a_{ik} b_{kj}$$

For example:

$$\begin{bmatrix} 4 & 5 & 8 \\ 2 & 1 & 7 \\ 3 & 6 & 9 \end{bmatrix} \begin{bmatrix} 5 & 1 & 9 \\ 4 & 3 & 6 \\ 2 & 8 & 7 \end{bmatrix} = \begin{bmatrix} 56 & 83 & 122 \\ 28 & 61 & 73 \\ 57 & 93 & 126 \end{bmatrix}$$

In R, use %*% to compute this!

Matrix Multiplication



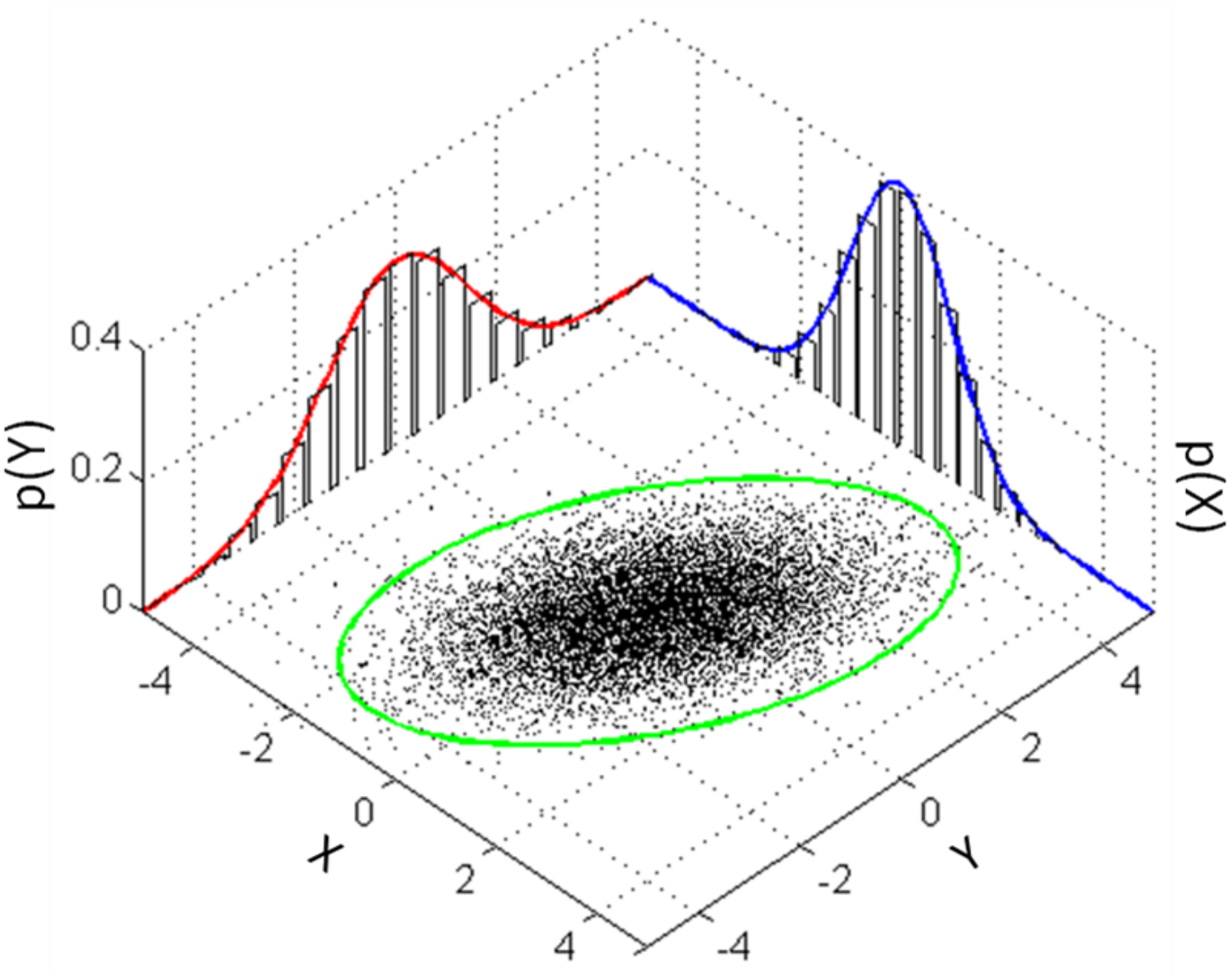
Source: Wikipedia

Multivariate Normal

We assume data are multivariate normally distributed:

$$\boldsymbol{y} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

- $\boldsymbol{y}$ is a random vector of item responses (e.g., answers to "are you fatigued?", "are you concentrating well?")
- $\boldsymbol{\mu}$ is a vector of means
- $\boldsymbol{\Sigma}$ is a variance–covariance matrix, standardizable to a correlation matrix.



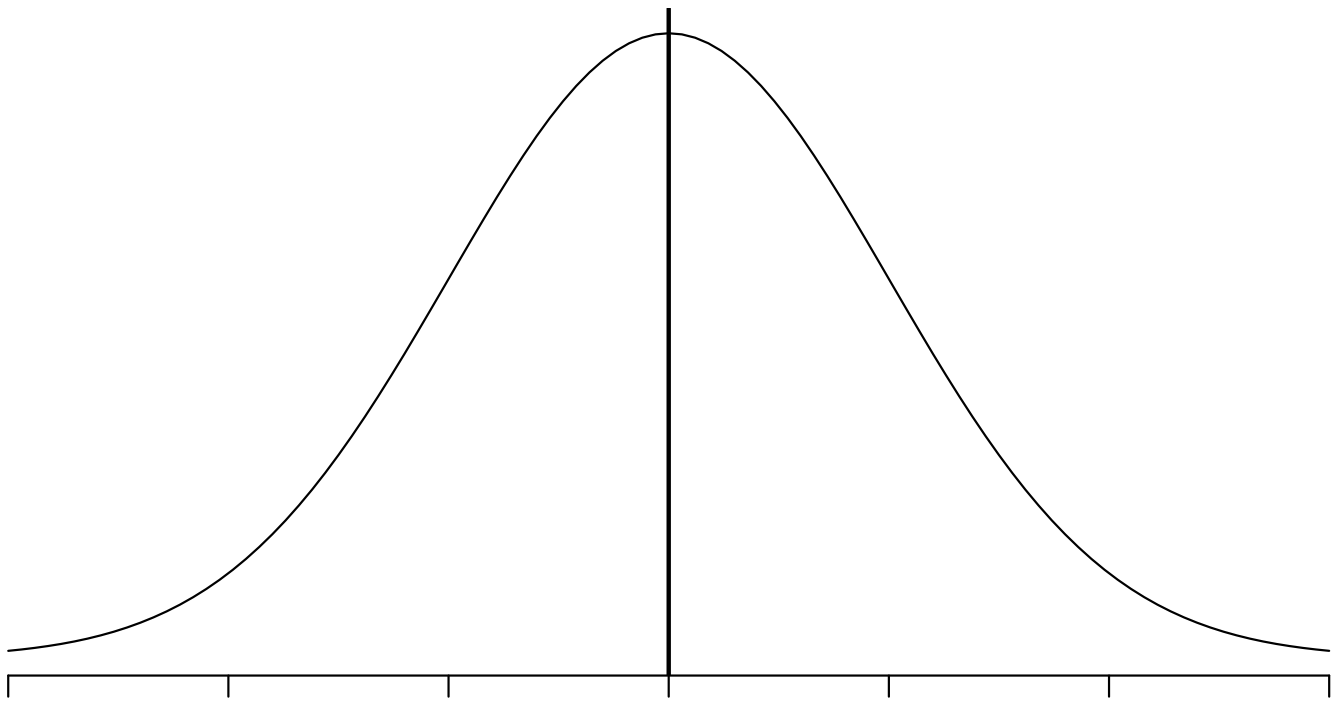
source: Wikipedia

Example data of $n = 10$ subjects:

$$\boldsymbol{Y} = \begin{bmatrix} \boldsymbol{y}_1^\top \\ \boldsymbol{y}_2^\top \\ \boldsymbol{y}_3^\top \\ \boldsymbol{y}_4^\top \\ \boldsymbol{y}_5^\top \\ \boldsymbol{y}_6^\top \\ \boldsymbol{y}_7^\top \\ \boldsymbol{y}_8^\top \\ \boldsymbol{y}_9^\top \\ \boldsymbol{y}_{10}^\top \end{bmatrix} = \begin{bmatrix} 4.09 & 4.94 & 4.58 \\ 2.48 & 4.27 & 5.44 \\ 4.93 & 4.24 & 6.24 \\ 2.17 & 5.36 & 5.79 \\ 3.39 & 4.62 & 4.81 \\ 1.41 & 3.74 & 5.21 \\ 2.87 & 2.60 & 5.17 \\ 2.62 & 2.37 & 4.55 \\ 3.49 & 3.74 & 6.75 \\ 4.10 & 5.69 & 3.00 \end{bmatrix}$$

Means

The center of the normal distribution of variable y_j is controlled by its *mean* μ_j:



Means

Means per column:

$$\bar{y}_j = \frac{1}{n} \sum_{p=1}^{n} y_{ij},$$

in which y_{ij} indicates row i, column j of data matrix $\boldsymbol{Y}$. Combining these values in a vector $\bar{\boldsymbol{y}}$ we get:

$$\bar{\boldsymbol{y}}^\top = \begin{bmatrix} 3.16 & 4.16 & 5.16 \end{bmatrix}$$

In R: `colMeans()`. This is an unbiased estimator of $\boldsymbol{\mu}$:

$$\mathcal{E}\left(\bar{\boldsymbol{y}}\right) = \boldsymbol{\mu}$$

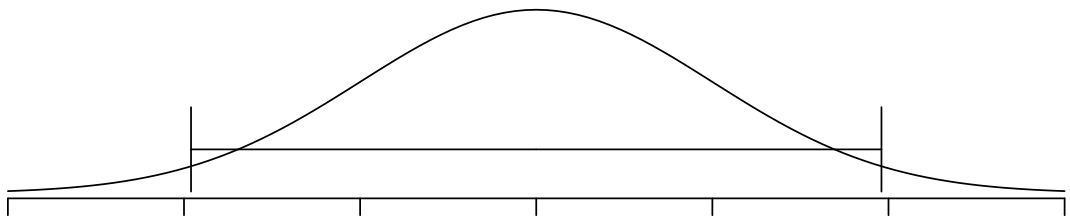
Diagonal elements of $\mathbf{\Sigma}$ correspond to variances:

$$\mathrm{Var}(y_j) = \sigma_j^2 = \sigma_{jj}$$

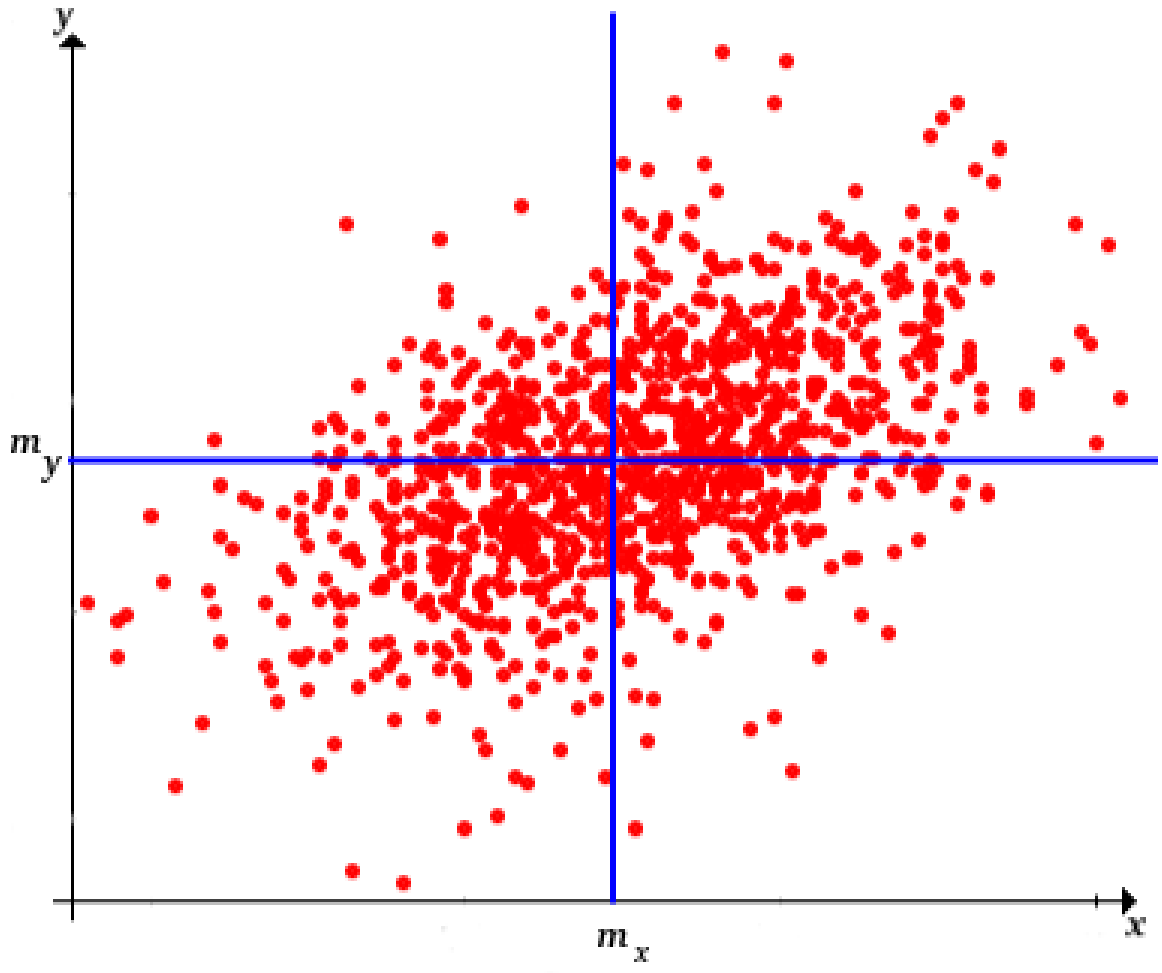
It's square root equals the standard deviation, which usually is interpreted:

$$\mathrm{SD}(y_j) = \sqrt{\mathrm{Var}(y_j)} = \sigma_j$$

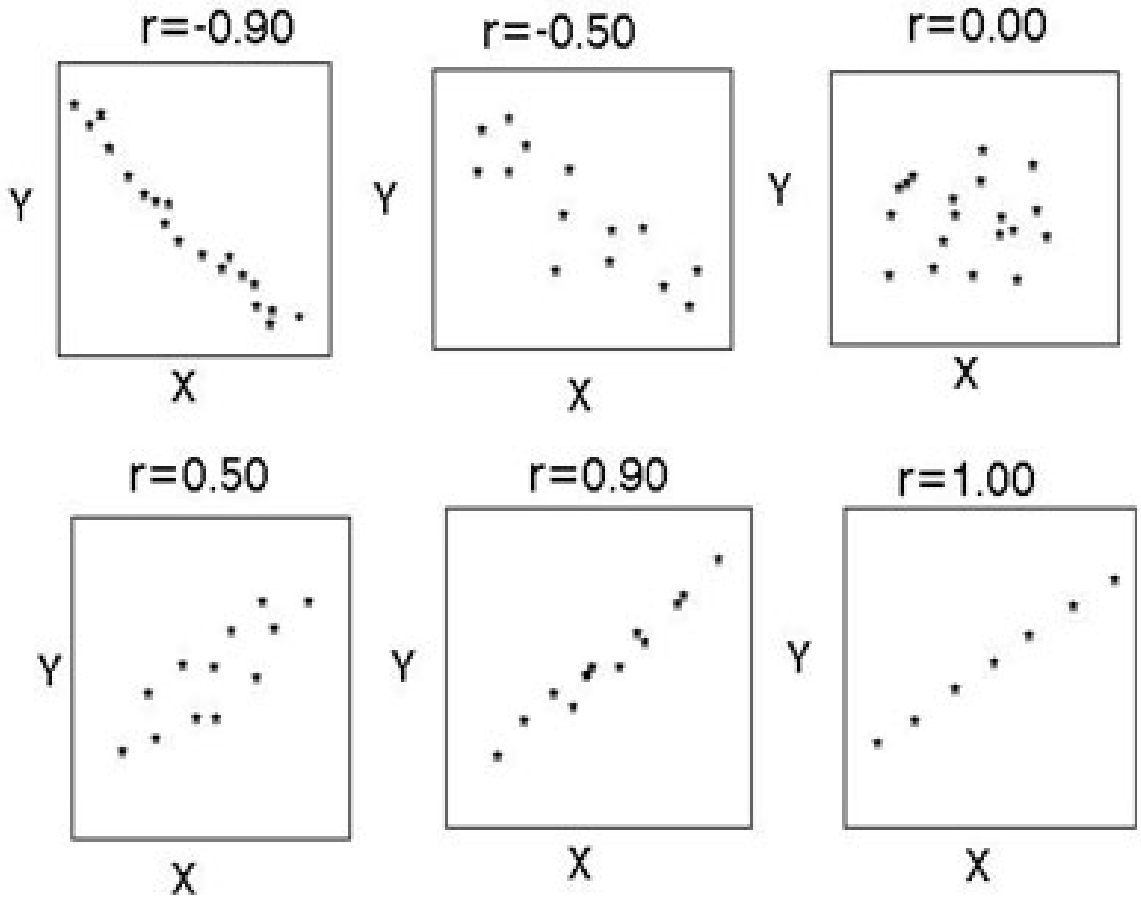
This parameter controls the *width* of the normal distribution. E.g., 95% of the probability mass is contained in $(\mu_j - 1.96\sigma_j, \mu_j + 1.96\sigma_j)$



Covariances encode association, a positive covariance encodes that when one variable is above (below) its mean, the other variable probably also is above (below) its mean. Negative covariances encode that whenever one variable is above (below) its mean, the other variable is likely below (above) its mean:



Covariances can be standardized to correlations, which are always between -1 and 1.



Variance–covariance matrix

Computing the variance–covariance matrix:

$$s^*_{jk} = \frac{\sum_{i=1}^{n}(y_{ij} - \bar{y}_j)(y_{ik} - \bar{y}_k)}{n-1},$$

Which becomes:

$$\boldsymbol{S}^* = \begin{bmatrix} 1.11 & 0.35 & -0.09 \\ 0.35 & 1.18 & -0.29 \\ -0.09 & -0.29 & 1.07 \end{bmatrix}$$

In R: `cov()`. This is an unbiased estimator of $\boldsymbol{\Sigma}$:

$$\mathcal{E}\left(\boldsymbol{S}^*\right) = \Sigma$$

Variance–covariance matrix

Dividing by $n-1$ leads to an unbiased estimator, but not the maximum likelihood estimate! This is obtained by dividing by n:

$$s_{jk} = \frac{\sum_{i=1}^{n}(y_{ij} - \bar{y}_j)(y_{ik} - \bar{y}_k)}{n},$$

The R function `cov()` returns the unbiased estimator (divided by $n-1$), to obtain the ML solution:

$$\boldsymbol{S} = \frac{n-1}{n}\boldsymbol{S}^*$$

Lavaan, the software we use, automatically applies this correction! So you use $\boldsymbol{S}^*$ as input, but $\boldsymbol{S}$ in computations.

Correlation matrix

Standardizing to correlations:

$$r_{jk} = \frac{s_{jk}}{\sqrt{s_{jj}}\sqrt{s_{kk}}},$$

which becomes

$$\boldsymbol{R} = \begin{bmatrix} 1.00 & 0.31 & -0.08 \\ 0.31 & 1.00 & -0.26 \\ -0.08 & -0.26 & 1.00 \end{bmatrix}.$$

In R: `cov2cor()` (or simply use `cor()` on data)