Test #2 – Formulas
STAT 873

## Introduction to R

- None

## Matrix algebra

- Matrix multiplication:

$$\mathbf{AB} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \end{bmatrix} = \begin{bmatrix} a_{11}b_{11} + a_{12}b_{21} + a_{13}b_{31} & a_{11}b_{12} + a_{12}b_{22} + a_{13}b_{32} \\ a_{21}b_{11} + a_{22}b_{21} + a_{23}b_{31} & a_{21}b_{12} + a_{22}b_{22} + a_{23}b_{32} \end{bmatrix}$$

- Inverse: For $\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$, $\mathbf{A}^{-1} = \dfrac{1}{a_{11}a_{22} - a_{12}a_{21}} \begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix}$
- Trace: $\text{tr}(\mathbf{A}) = \sum_{i=1}^{p} a_{ii} = a_{11} + a_{22} + \ldots + a_{pp}$
- Determinant of $2 \times 2$: $\left| \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \right| = a_{11}a_{22} - a_{12}a_{21}$
- Eigenvalues: Roots of the polynomial equation $|\mathbf{A} - \lambda \mathbf{I}| = 0$ where **I** is an identity matrix
- Eigenvectors: Each eigenvalue of **A** has a corresponding nonzero vector **b** that satisfies **Ab** = $\lambda$**b**
- For eigenvalues $\lambda_i$ of **A**: $\text{tr}(\mathbf{A}) = \sum_{i=1}^{p} \lambda_i$ and $|\mathbf{A}| = \prod_{i=1}^{p} \lambda_i = \lambda_1 \lambda_2 \cdots \lambda_p$
- Quadratic formula: The roots of the equation $ax^2 + bx + c = 0$ are $\dfrac{-b \pm \sqrt{b^2 - 4ac}}{2a}$
- Vector length: $\sqrt{\sum_{i=1}^{p} a_i^2}$
- Positive definite matrices have all eigenvalues greater than 0 and positive semidefinite matrices are the same but with at least one eigenvalue equal to 0

## Data, distributions, and correlation

- $\rho_{ij} = \dfrac{\sigma_{ij}}{\sqrt{\sigma_{ii}\sigma_{jj}}} = \dfrac{\text{Cov}(x_i, x_j)}{\sqrt{\text{Var}(x_i)\text{Var}(x_j)}}$
- $\boldsymbol{\mu} = \text{E}(\mathbf{x}) = \begin{bmatrix} \text{E}(x_1) \\ \vdots \\ \text{E}(x_p) \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \vdots \\ \mu_p \end{bmatrix}$
- $\boldsymbol{\Sigma} = \text{Cov}(\mathbf{x}) = \text{E}[(\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})'] = \begin{bmatrix} \sigma_{11} & \sigma_{12} & \cdots & \sigma_{1p} \\ \sigma_{21} & \sigma_{22} & \cdots & \sigma_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{p1} & \sigma_{p2} & \cdots & \sigma_{pp} \end{bmatrix}$
- $\boldsymbol{\Sigma} = \text{E}(\mathbf{x}\mathbf{x}') - \boldsymbol{\mu}\boldsymbol{\mu}'$

- $\mathbf{P} = \text{Corr}(\mathbf{x}) = \begin{bmatrix} 1 & \rho_{12} & \cdots & \rho_{1p} \\ \rho_{21} & 1 & \cdots & \rho_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{p1} & \rho_{p2} & \cdots & 1 \end{bmatrix}$
- Multivariate normal distribution, $\mathbf{x} \sim N_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$: $f(\mathbf{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{p/2} |\boldsymbol{\Sigma}|^{1/2}} e^{-\frac{1}{2}[(\mathbf{x}-\boldsymbol{\mu})' \boldsymbol{\Sigma}^{-1} (\mathbf{x}-\boldsymbol{\mu})]}$ for $-\infty < x_i < \infty$ for i=1,…,p and $|\boldsymbol{\Sigma}| > 0$
- $\hat{\boldsymbol{\mu}} = \frac{1}{N} \sum_{r=1}^{N} \mathbf{x}_r = \frac{1}{N} (\mathbf{x}_1 + \mathbf{x}_2 + \ldots + \mathbf{x}_N)$
- $\hat{\boldsymbol{\Sigma}} = \frac{1}{N-1} \sum_{r=1}^{N} (\mathbf{x}_r - \hat{\boldsymbol{\mu}})(\mathbf{x}_r - \hat{\boldsymbol{\mu}})'$
- $\hat{\sigma}_{ij} = \widehat{\text{Cov}}(x_i, x_j) = \frac{1}{N-1} \sum_{r=1}^{N} (x_{ri} - \bar{x}_i)(x_{rj} - \bar{x}_j)$
- $r_{ij} = \frac{\hat{\sigma}_{ij}}{\sqrt{\hat{\sigma}_{ii} \hat{\sigma}_{jj}}} = \frac{\frac{1}{N-1} \sum_{r=1}^{N} (x_{ri} - \bar{x}_i)(x_{rj} - \bar{x}_j)}{\sqrt{\left[ \frac{1}{N-1} \sum_{r=1}^{N} (x_{ri} - \bar{x}_i)^2 \right] \left[ \frac{1}{N-1} \sum_{r=1}^{N} (x_{rj} - \bar{x}_j)^2 \right]}} = \frac{\sum_{r=1}^{N} (x_{ri} - \bar{x}_i)(x_{rj} - \bar{x}_j)}{\sqrt{\left[ \sum_{r=1}^{N} (x_{ri} - \bar{x}_i)^2 \right] \left[ \sum_{r=1}^{N} (x_{rj} - \bar{x}_j)^2 \right]}}$
- $\mathbf{R} = \begin{bmatrix} 1 & r_{12} & \cdots & r_{1p} \\ r_{21} & 1 & \cdots & r_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ r_{p1} & r_{p2} & \cdots & 1 \end{bmatrix}$
- $z_{rj} = \frac{x_{rj} - \hat{\mu}_j}{\sqrt{\hat{\sigma}_{jj}}}$

## Graphics

- None

## PCA

- $y_j = \mathbf{a}_j'(\mathbf{x} - \boldsymbol{\mu})$ for j = 1, …, p
- Total variance: $\text{tr}(\boldsymbol{\Sigma}) = \sum_{i=1}^{p} \sigma_{ii} = \sigma_{11} + \sigma_{22} + \ldots + \sigma_{pp}$
- $\hat{y}_j = \hat{\mathbf{a}}_j'(\mathbf{x} - \hat{\boldsymbol{\mu}})$ for j = 1, …, p
- $\hat{y}_{rj}^* = \hat{\mathbf{a}}_j^{*'} \mathbf{z}_r$ and $\hat{y}_{rj} = \hat{\mathbf{a}}_j'(\mathbf{x}_r - \hat{\boldsymbol{\mu}})$ for j = 1, …, p and r = 1, …, N
- $\hat{\mathbf{c}}_j^* = \hat{\lambda}_j^{*1/2} \hat{\mathbf{a}}_j^*$
- $r_{z_i, y_j} = \hat{a}_{ij}^* \sqrt{\hat{\lambda}_j}$
- Orthogonal matrix: Individual columns within a matrix are orthogonal to each other
- $\underset{N \times p}{\tilde{\mathbf{X}}} = \underset{N \times p}{\mathbf{U}} \underset{p \times p}{\hat{\boldsymbol{\Lambda}}} \underset{p \times p}{\hat{\mathbf{A}}'}$ where $\tilde{\mathbf{X}}$ is a matrix with mean adjusted values for each variable, $\hat{\mathbf{A}}$ is a matrix with all of the eigenvectors, $\hat{\boldsymbol{\Lambda}} = \text{Diag}(\hat{\lambda}_1, \ldots, \hat{\lambda}_p)$, $\mathbf{U}$ is an orthogonal matrix, and each row of $\mathbf{U}\hat{\boldsymbol{\Lambda}}$ provides the PC scores

- $x_{rj} - \hat{\mu}_j \approx \hat{y}_{r1}\hat{a}_{j1} + \hat{y}_{r2}\hat{a}_{j2}$

**FA**

- $x_j = \mu_j + \lambda_{j1}f_1 + \lambda_{j2}f_2 + \ldots + \lambda_{jm}f_m + \eta_j$ for j = 1, …, p
- $\tilde{x}_j = \lambda_{j1}f_1 + \lambda_{j2}f_2 + \ldots + \lambda_{jm}f_m + \eta_j$ for j = 1, …, p; $\underset{p\times 1}{\tilde{\mathbf{x}}} = \underset{p\times m}{\boldsymbol{\Lambda}}\,\underset{m\times 1}{\mathbf{f}} + \underset{p\times 1}{\boldsymbol{\eta}}$
- $z_j = \lambda_{j1}f_1 + \lambda_{j2}f_2 + \ldots + \lambda_{jm}f_m + \eta_j$ for j = 1, …, p; $\underset{p\times 1}{\mathbf{z}} = \underset{p\times m}{\boldsymbol{\Lambda}}\,\underset{m\times 1}{\mathbf{f}} + \underset{p\times 1}{\boldsymbol{\eta}}$
- $\text{Var}(ay_1+by_2) = a^2\text{Var}(y_1) + b^2\text{Var}(y_2) + 2ab\text{Cov}(y_1,y_2)$
- $\boldsymbol{\Sigma} = \boldsymbol{\Lambda}\boldsymbol{\Lambda}' + \boldsymbol{\psi}$; $\text{Var}(x_j) = \sum_{k=1}^{m}\lambda_{jk}^2 + \psi_j$ and $\text{Cov}(x_j, x_{j'}) = \sum_{k=1}^{m}\lambda_{jk}\lambda_{j'k}$
- With standardized variables, $\mathbf{P} = \boldsymbol{\Lambda}\boldsymbol{\Lambda}' + \boldsymbol{\psi}$, $\sum_{k=1}^{m}\lambda_{jk}^2 + \psi_j = 1$, and $\text{Corr}(z_j, f_k) = \lambda_{jk}$
- LRT: $A = (N-1-(2p+4m+5)/6)\log\left(\dfrac{|\hat{\boldsymbol{\Lambda}}\hat{\boldsymbol{\Lambda}}' + \hat{\boldsymbol{\Psi}}|}{|[(N-1)/N]\hat{\boldsymbol{\Sigma}}|}\right)$ can be approximated by $\chi^2_{[(p-m)^2-p-m]/2}$
- AIC: $-2\log(L(\tilde{\mathbf{x}}\,|\,\hat{\boldsymbol{\Lambda}}, \hat{\boldsymbol{\Psi}})) + 2$(degrees of freedom for model)
- $\underset{p\times m}{\mathbf{B}} = \underset{p\times m}{\boldsymbol{\Lambda}}\,\underset{m\times m}{\mathbf{T}}$
- $V = \dfrac{1}{p^2}\sum_{q=1}^{m}\left(p\sum_{j=1}^{p}\dfrac{b_{jq}^4}{h_j^4} - \left(\sum_{j=1}^{p}\dfrac{b_{jq}^2}{h_j^2}\right)^2\right)$ where $h_j^2 = \sum_{k=1}^{m}\lambda_{jk}^2$
- Bartlett's method: $\hat{\mathbf{f}}_r = \left(\hat{\boldsymbol{\Lambda}}'\hat{\boldsymbol{\Psi}}^{-1}\hat{\boldsymbol{\Lambda}}\right)^{-1}\hat{\boldsymbol{\Lambda}}'\hat{\boldsymbol{\Psi}}^{-1}\mathbf{z}_r$
- Thompson's method: $\hat{\mathbf{f}}_r = \hat{\boldsymbol{\Lambda}}'(\hat{\boldsymbol{\Lambda}}\hat{\boldsymbol{\Lambda}}' + \hat{\boldsymbol{\Psi}})^{-1}\mathbf{z}_r$

**R functions** – These functions are listed mostly in the order they were introduced in the notes

**Introduction to R**

| Function | Description |
|---|---|
| `pnorm()` | Finds a cumulative probability from a univariate normal distribution |
| `qnorm()` | Finds a quantile from a univariate normal distribution |
| `ls() and objects()` | List items in R's database |
| `c()` | Combine items into a vector |
| `sd()` | Calculate a standard deviation |
| `var()` | Calculate a variance |
| `sqrt()` | Calculate a square root |
| `pt()` | Finds a cumulative probability from a univariate t distribution |
| `qt()` | Finds a quantile from a univariate t distribution |
| `t.test()` | Calculates quantities associated with a t-test |
| `read.table(file = "c:\\chris\\datafile.txt", header = TRUE, sep = "")` | Read in a text data file with variable names in the first row and spaces separating the variable names and their values. Comma delimited data files can be read in using the `sep=","` option. If the first row does not contain the variable names, use `header = FALSE` and `col.names = c("var1", …, "var.c")` to name the variables yourself. The `read.csv()` function provides a shortcut way to read in comma delimited data files. |
| `summary()` | Summarize information in a data frame or list |
| `head()` | Print the first few rows of a data frame |
| `write.table(x = set1, file = "C:\\out_file.csv", quote = FALSE, row.names = FALSE, sep=",")` | Save data in a data frame to a file on the hard drive. The data was in the data frame `set1` and it will be written as a comma delimited file named `out_file.csv`. |
| `library(RODBC)`<br>`z<-odbcConnectExcel("C:\\gpa.xls")`<br>`gpa.excel<-sqlFetch(z, "sheet1")`<br>`close(z)` | Read in data from an Excel file called `gpa.xls`. The data is located on sheet1 of the Excel file. |
| `plot(x = x, y = y)` | Plots y on the y-axis and x on the x-axis |
| `lm(formula = y ~ x, data = set1)` | Find the sample regression model (and various other measures) with the response variable y and predictor variable x within `set1` |
| `names()` | Provide the names of items in a list |
| `class()` | State the class of an object |
| `win.graph(width = 6, height = 6, pointsize = 10)` | Opens a new graphics window that is 6"×6" with font size of 10 |
| `segments()` | Draw a line segment on a plot |
| `curve()` | Plot a function of x, like f(x) = $x^2$ |

| Function | Description |
|---|---|
| `expression()` | Can be used to put Greek letters and mathematical symbols on a plot |
| `axis()` | Allows for finer control of an x or y-axis in a plot |
| `methods()` | List the method or generic functions |

**Matrix algebra**

| Function | Description |
|---|---|
| `matrix(data = c(1, 2, 3, 4, 5, 6), nrow = 2, ncol = 3, byrow = TRUE)` | Create a matrix of size 2×3 by row |
| `t()` | Transpose a matrix |
| `A+B` | Matrix addition for matrices **A** and **B** |
| `A%*%B` | Matrix multiplication for **A** and **B** |
| `A*B` | Elementwise multiplication for **A** and **B** |
| `cbind()` | Combine elements by column |
| `solve(A)` | Find the inverse of **A** |
| `diag(A)` | Extract the diagonal elements of **A** |
| `sum(A)` | Sum the elements of **A** |
| `det(A)` | Determinant of **A** |
| `eigen(A)` | Find the eigenvalues and eigenvectors of **A** |
| `abline(h = y)` | Plots a horizontal line at y. A vertical line is plotted with the argument v. |
| `arrows()` | Draw an arrow on a plot |

**Data, distributions, and correlation**

| Function | Description |
|---|---|
| `cov2cor()` | Calculate a correlation matrix from a covariance matrix |
| `dmvnorm()` | f(**x**) for a multivariate normal distribution; this is in the mvtnorm package |
| `seq()` | Create a sequence of numbers |
| `persp3d()` | 3D surface plot; this function is in the rgl package |
| `contour()` | Contour plot |
| `cov()` | Calculate estimated covariance matrix |
| `cor()` | Calculate estimated correlation matrix |
| `colMeans()` | Find the means of each column in a matrix |
| `apply()` | Apply a function to every row or column of a matrix |
| `set.seed()` | Set a seed number |
| `rmvnorm()` | Simulate random vectors from a multivariate normal distribution; this function is in the mvtnorm package |
| `points()` | Add points to a plot |
| `scale()` | Standardize columns of data |
| `expand.grid()` | Create all possible combinations of items within separate vectors |

| | |
|---|---|
| `par()` | Graphics parameters; `pty = "s"` creates a square plot, `mfrow = c(2,2)` creates a 2×2 matrix of plots |

**Graphics**

| Function | Description |
|---|---|
| `pairs()` | Side-by-side scatter plots |
| `scatterplotMatrix()` | Side-by-side scatter plots |
| `symbols()` | Bubble plot; `circles` argument specifies the third variable; `inches` argument controls the maximum size of the bubble |
| `identify()` | Interactively identifies points on a plot |
| `text()` | Puts text on a plot |
| `plot3d()` | 3D scatter plot; this function is within the rgl package |
| `stars()` | Star plot |
| `parcoord()` | Parallel coordinate plot; this function is within the MASS package |
| `ipcp()` | This function is within the iplots package |
| `reshape()` | Changes a data frame from a wide to long format and vice versa |
| `histogram()` | Trellis histogram; this function is within the lattice package |
| `xyplot()` | Trellis scatter plot; this function is within the lattice package |
| `cloud()` | Trellis 3D scatter plot; this function is within the lattice package |
| `equal.count()` | Creates shingles for a trellis plot |

**PCA**

| Function | Description |
|---|---|
| `princomp()` | Performs PCA; `cor` argument specifies whether to use the covariance (`FALSE`) or correlation (`TRUE`) matrix |
| `summary()` | This function can be used to summarize the information with an object created by `princomp()`; the argument values of `loadings = TRUE` and `cutoff = 0.0` will lead to the printing of all eigenvectors |
| `predict()` | Computes PC scores when using an object created by `princomp()`; see my programs for how to calculate the scores correctly |

**FA**

| Function | Description |
|---|---|
| `factanal()` | Performs FA; the `rotation = "varimax"` argument specifies the varimax rotation method; the `scores` argument can be used to specify the type of scores (`"regression"` or `"Bartlett"`) to be calculated |
| `print()` | This function can be used to summarize the information with an object created by `factanal()`; the argument value of `cutoff = 0.0` will lead to the printing of all common factor loadings |