Advanced Topics in Computational Statistics

Lecture Summary
(Autumn Semester 2015)

Giuseppe Accaputo
g@accaputo.ch

February 9, 2016

Disclaimer

This is a summary of the *Advanced Topics in Computational Statistics* lecture [10] taught by Prof. Maathius and Prof. Mächler during the autumn semester 2015 at the ETH Zürich and was written by me as a preparation for the oral exam. The equations displayed in boxes in this summary have been presented during the lecture; equations taken from other sources are appropriately referenced in the text.

Lecture 1 (Week 38)

Bayes Risk and Bayes Estimator (7)

- Let Y be a categorical variable and $\mathcal{Y}$ be the set of possible classes, with $Y \in \mathcal{Y}$; an estimate $\hat{Y}$ will also assume values in $\mathcal{Y}$. Further, let $L(k, l)$ be a loss function which characterizes the price to pay for classifying an observation belonging to class $\mathcal{Y}_k$ as $\mathcal{Y}_l$. The expected prediction error EPE is

$$\boxed{\text{EPE} = \mathbb{E}[L(Y, \hat{Y}(X))]} \tag{1}$$

  By minimizing the EPE pointwise, we get

$$\boxed{\hat{Y}(x) = \text{argmin}_{l \in \mathcal{Y}} \sum_{k=1}^{K} L(\mathcal{Y}_k, l) \mathbb{P}[\mathcal{Y}_k | X = x]} \tag{2}$$

Using a $0-1$ loss function, where all the misclassifications are charged a single unit, Eq. (2) simplifies to

$$\boxed{\hat{Y}(x) = \text{argmin}_{l \in \mathcal{Y}}[1 - \mathbb{P}[l|X = x]]} \tag{3}$$

or simply

$$\boxed{\hat{Y}(x) = \mathcal{Y}_k \quad \text{if} \quad \mathbb{P}[\mathcal{Y}_k|X = x] = \max_{l \in \mathcal{Y}} \mathbb{P}[l|X = x]} \tag{4}$$

Eq. (4) is known as the *Bayes classifier*, and says that we classify to the most probable class, using the conditional distribution $\mathbb{P}[Y|X]$. The error rate of the Bayes classifier is called the *Bayes rate*

Definition of k-nearest neighbours (k-NN) (5)

- Given a positive integer k and a test observation x, the k-NN classifier first identifies the k points in the training data that are closest to x, represented by N_k. Let Y be a quantitive output (regression), which we want to accurately approximate through the estimation $\hat{Y}$. The k-nearest neighbor fit for $\hat{Y}$ is then defined as

$$\boxed{\hat{Y}(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i} \tag{5}$$

  - If $\hat{Y}$ should accurately approximate a qualitative output $Y = 0, \ldots, l-1$ (classification), then by using k-NN $\hat{Y}$ can be calculated with

$$\boxed{\hat{Y} = \frac{1}{k} \sum_{i \in N_k} I(y_i = j)} \tag{6}$$

    where $j \in \{0, \ldots, l-1\}$ is the class label and $I(y_i = j)$ is an indicator function, i.e.,

$$\boxed{I(y_i = j) = \begin{cases} 1 & \text{if } y_i = j, \\ 0 & \text{else.} \end{cases}} \tag{7}$$

- k-NN does not appear to rely on any assumptions about the underlying data, and can adapt to any situation. However, any particular subregion of the decision boundary depends on a handful of input points and their particular position, and is thus wiggly and unstable, resulting in *high variance and low bias*
  - The linear decision boundary from least squares is very smooth, and apperently stable to fit, but does rely heavily on the assumption that a linear decision boundary is appropriate, thus resulting in *low variance and high bias*

Lecture 2 (Week 39)

Bias-Variance Tradeoff (7)

- The expected test MSE is defined as

$$\boxed{\mathbb{E}[Y - \hat{Y}(X)]^2 = \mathrm{Var}(\hat{Y}(X)) + [\mathrm{Bias}(\hat{Y}(X))]^2 + \mathrm{Var}(\epsilon) \quad ,} \tag{8}$$

  where ϵ are the error terms. In order to minimize the expected test MSE in Eq. (8), we need to select a statistical learning method that simultaneously achieves *low variance* and *low bias*

  - *Variance* refers to the amount by which $\hat{Y}$ would change if we estimated it using a different training data set. Ideally, the estimate for Y should not vary too much between training sets, meaning if a method has high variance then small changes in the training data set can result in large changes in $\hat{Y}$
  - *Bias* refers to the error that is introduced by approximating a real-life problem — which may be extremely complicated — by a much simpler model
    - For example, linear regression assumes that there is a linear relationship between Y and $X_1, X_2, \ldots, X_p$, but it is unlikely that any real-life problem truly has such a simple linear relationship, and so performing linear regression will undoubtedly result in some bias in the estimate of Y.

- For k-NN, choosing the optimal k can be achieved by using cross validation

The Different Types of Errors

- We have a target variable Y, a vector of inputs $\mathbf{X}$, and a prediction model $\hat{f}(\mathbf{X})$ that has been estimated from a training set $\mathcal{T}$. The loss function for measuring errors between Y and $\hat{f}(\mathbf{X})$ is denoted by $L(Y, \hat{f}(\mathbf{X}))$

- The *training error* is the average loss over the training samples, i.e.,

$$\boxed{\overline{\mathrm{err}} = \frac{1}{n}\sum_{i=1}^{n} L(Y_i, \hat{Y}) \quad ,} \tag{9}$$

  where $L(\cdot, \cdot)$ is the loss function and Y_i is the true class label and $\hat{Y}$ the predicted class label.

  - For 1-NN, $\overline{\mathrm{err}} = 0$, since the nearest neighbor of X_i is X_i itself, thus giving $Y_i = \hat{Y}_i$
  - For k-NN with $k > 1$, $\overline{\mathrm{err}}$ goes up given a low model complexity. $\overline{\mathrm{err}}$ decreases in general if one increases the model complexity

- The *test error*, also referred to as *generalization error*, is the prediction error over an independent test sample drawn from the distribution of (X, Y) and is given by

$$\boxed{\mathrm{Err}_{\mathcal{T}} = \mathbb{E}[L(Y, \hat{f}(\mathbf{X}))|\mathcal{T}] \quad ,} \tag{10}$$

  where the training data set $\mathcal{T}$ is fixed, and test error refers to the error for this specific training set

- The *expected test/prediction error* is defined as

$$\boxed{\mathrm{Err} = \mathbb{E}[\mathrm{Err}_{\tau}] \quad ,} \tag{11}$$

  where Err_{τ} is defined in Eq. (10)

K-Fold Cross-Validation (5)

- K-fold cross-validation (CV) uses part of the available data to fit the model, and a different part to test it
- K-fold CV works as follows:
  1. Split the data into K roughly equal-sized parts
  2. For the kth part, fit the model to the other $K - 1$ parts of the data
  3. Calculate the prediction error of the fitted model when predicting th ekth part of the data
  4. Do steps (1) to (3) for $k = 1, 2, \ldots, K$ and combine the K estimates of prediction error
- The cross-validation estimate of the prediction error is

$$\boxed{\mathrm{CV}(\hat{f}) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, \hat{f}^{-\kappa(i)}(x_i)) \quad ,} \tag{12}$$

  where $\kappa : \{1, \ldots, N\} \to \{1, \ldots, K\}$ is an indexing function that indicates the partition to which observation i is allocated by the randomization and $\hat{f}^{-k}(x)$ the fitted function, computed with the kth part of the data removed.
- The expected test error Err can be approximated using cross-validation.
  - Often $K = 5, 10$ is used.

Curse of Dimensionality

- With a large data set, k-NN seems a good method, as we can always find k-nearest neighbour points close to x. This is true for small dimensions. In higher dimensions we run into *the curse of dimensionality*, which has the following implications:

1. Consider the nearest-neighbor procedure for inputs uniformly distributed in a p-dimensional hypercube. Suppose we want to capture a fraction r of the observations by a smaller hypercube. Since this corresponds to a fraction r of the unit hypercube, the edge length of the smaller hypercube will be $r^{1/p}$. For example, to capture 1% of the data in dimension $p = 10$, you need to capture $0.01^{1/10} = 0.63 = 63\%$ of each axis / input variable.
2. Another consequence of the sparse sampling in high dimensions is that all sample points are close to an edge of the sample. Consider N data points uniformly distributed in a p-dimensional unit ball centered at the origin. The median from the origin to the closest data point is given by the expression

$$\boxed{d(p, N) = \left(1 - \frac{1^{1/N}}{2}\right)^{1/p}} \tag{13}$$

   e.g., with $N = 500$ and $p = 10$ Eq. (13) evaluates to approximately 0.52, resulting in many points being closer to the boundary than to any other data point
3. Another manifestation of the curse is that the sampling density is proportional to $N^{1/p}$.This means that if $N_1 = 100$ represents a dense sample for a single input problem, then $N_10 = 100^10$ is the sample size required for the same sampling density with 10 inputs, meaning that in high dimensions all feasible training samples sparsely populate the input space.

Lecture 3 (Week 40)

Weighted k-Nearest Neighbours

- In plain k-NN, all neighbours are weighted equally, independent from the distance
- Idea: data points that are closer to the target point x should get higher weights

Issues with k-NN

- Not invariant to monotone transormations of the variables
- Not easy to chose k (tuning parameter)
- Curse of dimensionality

Layered Nearest Neighbours (1)

- *Definition*: An observation $\mathbf{X}_i = (x_{i1}, \ldots, x_{ip}) \in \mathbb{R}^p$ is a layered nearest neighbour (LNN) of a target point x if the hyperrectangle defined by $\mathbf{x}$ and $\mathbf{X}_i$ does not contain any other data points

Estimation

- Assume that we are given a sequence $(\mathbf{X}_1, Y), \ldots, (\mathbf{X}_n, Y_n)$ of i.i.d. $\mathbb{R}^p \times \mathbb{R}$-valued random variables. Further, let $\mathcal{L}_n(\mathbf{x})$ be the set of LNNs of $\mathbf{x}$ with $L_n(\mathbf{x}) = |\mathcal{L}_n(\mathbf{x})|$ being the number of LNNs of $\mathbf{x}$. Then, the regression function $r(\mathbf{x}) = \mathbb{E}[Y|\mathbf{X} = \mathbf{x}]$ may be estimated by

$$\boxed{r_n(\mathbf{x}) = \frac{1}{L_n(\mathbf{x})} \sum_{\mathbf{X}_i \in \mathcal{L}_n(\mathbf{x})} Y_i} \tag{14}$$

  - Note: $L_n(\mathbf{x}) \geq 1$, so the division makes sense

Classification

- For classification use

$$\boxed{\text{argmax}_{l \in \{0,\ldots,k-1\}} \frac{1}{L_n(\mathbf{x})} \sum_{\mathbf{X}_i \in \mathcal{L}_n(\mathbf{x})} I(Y_i = l)} \tag{15}$$

  where I is the indicator function defined in Eq. (7)

Properties of $L_n(\mathbf{x})$

- $L_n(\mathbf{x}) \to \infty$ as $n \to \infty$
  - For k-NN we choose $k \to \infty$ as $n \to \infty$, but $k/n \to 0$ as $n \to \infty$
- $\mathbb{E}[L_n(\mathbf{x})] \approx \frac{2^p (\log n)^{p-1}}{(p-1)!}$, i.e., $L_n(\mathbf{x})$ goes slower to infinity as n

Why Is LNN Interesting?

1. We do not need a tuning parameter
2. It is scale-invariant, which is clearly a desirable feature when the components of the vector represent physically different quantities
3. There is a close relation with random forests; the latter suffers less from the curse of dimensionality

Random Forests (1)

1. Take the data $(\mathbf{X}_1, Y), \ldots, (\mathbf{X}_n, Y_n)$ and partition $\mathbf{R}^d$ randomly into pure rectangles, i.e., rectangles that each contain one data point
   - If $A(\mathbf{X})$ is the rectangle to which $\mathbf{X}$ belongs, then $\mathbf{X}$ votes "Y_i", where $\mathbf{X}_i$ is the unique data point in $A(\mathbf{X})$. This means that each voting $\mathbf{X}_i$ is a LNN of $\mathbf{X}$

2. Repeat step 1) many times, resulting in a random forest. Since each voting $\mathbf{X}_i$ is a LNN of $\mathbf{X}$, it means that a random forests leads to a weighted LNN estimate
   - More formally, assume that $\theta_1, \ldots, \theta_m$ are i.i.d. draws of some randomising variable θ, independent of the sample. A random forest is a collection of m randomised trees $t_1(\mathbf{x}, \theta_1, \mathcal{D}_n), \ldots, t_m(\mathbf{x}, \theta_m, \mathcal{D}_n)$ which is constructed by repeatedly bagging, i.e., selecting a random sample with replacement of the training set and fit trees t_j to these m samples, with $\mathcal{D}_n = \{\mathbf{X}_1, \ldots, \mathbf{X}_n\}$ being a sample of i.i.d. random vectors in $\mathbb{R}^d$, $d \geq 2$
     - Regression:
     $$\boxed{r_n = \frac{1}{m} \sum_{j=1}^{m} t_j(\mathbf{x}, \theta_j, \mathcal{D}_n)} \tag{16}$$
     - Classification: Majority vote among $t_j(\mathbf{x}, \theta_j, \mathcal{D}_n)$, with $j = 1, \ldots, m$ for classification

Lecture 4 (Week 41)

- Assume we have points $\mathbf{X}_1, \ldots, \mathbf{X}_n \in \mathbb{R}^p$ with p very large. Storing and computing is rather expensive
- *Idea:* Consider a mapping
$$\boxed{\Phi : \mathbb{R}^p \to \mathbb{R}^m, \quad m \leq p} \tag{17}$$
such that important properties of the points $\mathbf{X}_1, \ldots, \mathbf{X}_n$ are preserved in $\Phi(\mathbf{X}_1), \ldots, \Phi(\mathbf{X}_n)$
  - For example, for k-NN methods it is important to preserve pairwise distances
- *Definition:* Given a tolerance parameter $\delta \in (0, 1)$, a dimension reduction mapping Φ is called *δ-faithful* if for all $j, k \in \{1, \ldots, n\}$
$$\boxed{(1 - \delta) \leq \frac{\|\Phi(\mathbf{X}_j) - \Phi(\mathbf{X}_k)\|}{\|\mathbf{X}_j - \mathbf{X}_k\|} \leq (1 + \delta)} \tag{18}$$
  - Multidimensional scaling (MDS) and principle component analysis (PCA) are two linear mappings that generate faithful low dimensional representations

Johnson-Lindenstrauss Lemma

- Assume the random linear map
$$\boxed{\Phi(\mathbf{x}) = \frac{\mathbf{S} \cdot \mathbf{x}}{\sqrt{m}}} \tag{19}$$
is given, where $\mathbf{S} \in \mathbb{R}^{m \times p}$ is the standard Gaussian matrix with entries i.i.d. $\mathcal{N}(0, 1)$

- Given a projection dimension

$$\boxed{m > \frac{32}{\delta^2} \log n} \tag{20}$$

the random linear map in Eq. (19) is δ-faithful with probability

$$\boxed{\mathbb{P}[\delta\text{-faithful}] \geq 1 - \exp\{-m\delta^2/16\}} \tag{21}$$

Lecture 5 (Week 42)

The Expectation-Maximization (EM) Algorithm

- The likelihood function $L(a)$ is the probability for the occurence of a sample configuration $x_1, \ldots, x_n$ given that the probability density $f(x; a)$ with parameter a known,

$$L(a) = f(x_1; a) \cdots f(x_n; a) \tag{22}$$

- The EM algorithm is a very general iterative algorithm for maximum likelihood estimation in incomplete data-problems

ien

Mixture of Two Univariate Gaussians

- If the histogram of data shows bi-modality, i.e., there seems to be two separate underlying regimes, then it is advisable to model the data Y as a mixture of two normal distributions, since a single Gaussian distribution would not be appropriate
- Model Y as a mixture of two normal distributions:

$$\boxed{\begin{aligned} U &\sim \mathcal{N}(\mu_1, \sigma_1^2) \\ V &\sim \mathcal{N}(\mu_2, \sigma_2^2) \\ Z &\sim \text{Bernoulli}(\pi) \\ Y &= (1 - Z) \cdot U + Z \cdot V \end{aligned}} \tag{23}$$

  - The density of Y is

$$\boxed{f_\theta(y) = (1 - \pi)\, g_{\theta_1}(y) + \pi\, g_{\theta_2}(y)} \quad , \tag{24}$$

    where g_{θ_j} is the normal density, $\pi \in [0, 1]$, $\theta_j = (\mu_j, \sigma_j^2)$ and $\theta = (\pi, \theta_1, \theta_2)$
  - $Z \in 0, 1$ with probability π, i.e, $\Pr(Z = 1) = \pi$

  * The θ in $f_\theta(y)$ describes the parameters of the density family used; in this case, the density family is the normal distribution and the density function f_θ is parameterized by $\theta = (\pi, \theta_1, \theta_2)$

- The log-likelihood is

$$\boxed{l(\theta; y_1, \ldots, y_n) = \sum_{i=1}^{n} \log((1-\pi)) g_{\theta_1}(y_i) + \pi g_{\theta_2}(y_i)} \tag{25}$$

  – For $\sigma_i \to 0$ and $\mu_j = y_i$ for some i's,

$$\boxed{g_{\theta_j}(y) = \frac{1}{\sqrt{2\pi}\sigma_j} \exp\{-\frac{1}{2}\left(\frac{y_i - \mu_j}{\sigma_j}\right)^2\}} \tag{26}$$

  tends to go to ∞, meaning that the maximum likelihood estimate only *exists* when $\sigma_j > 0$

Working with *latent* Z_i

- If $Z_i = 1$, then Y_i comes from model 2, i.e. $V \sim \mathcal{N}\theta_2$, otherwise it comes from model 1, i.e., $U \sim \mathcal{N}\theta_1$ (see Eq. (23))

- The log-likelihood in this case is given by

$$\boxed{\begin{aligned} l_c(\theta; y_1, \ldots, y_n; z_1, \ldots, z_n) = &\sum_{i=1}^{N} [(1-z_i) \log g_{\theta_1}(y_i) + z_i \log g_{\theta_2}(y_i)] \\ &+ \sum_{i=1}^{N} [(1-z_i)\log(1-\pi) + z_i \log \pi] \end{aligned}} \tag{27}$$

  – The joint density $f(y, z)$ is given by

$$[(1-\pi) g_{\theta_1}(y)]^{1-z} \cdot [\pi\, g_{\theta_2}(y)]^z \tag{28}$$

  and can be derived using the fact that $f(y, z) \propto f(y|z) \cdot f(z)$ with

$$f(y|z) = g_{\theta_1}(y)^{1-z}\; g_{\theta_2}(y)^z \tag{29}$$

$$f(z) = (1-\pi)^{1-z}\; \pi^z \tag{30}$$

- Maximization can happen termwise:

1. For θ_1 (model 1, for which $z_i = 0$), minimize
$$\sum_{i:z_i=0} -\log g_{\theta_i}(y_i) \tag{31}$$
2. Same as 1) for θ_2 (with $z_i = 1$ respectively)
3. π can be determined by maximizing the 3rd and 4th term in l_c

Defining a Value for the Z_i's

E-Step: Since we do not know the values of the z_i's, the idea is to replace them by their expected values. Given the parameters θ and the observed data Y_i, we compute

$$\begin{aligned}\gamma_i &= \mathrm{E}[Z_i|\theta; Y_1, \ldots, Y_n] \\ &= \Pr[Z_i = 1|\theta; Y_i] \\ &= \frac{f_\theta(Y_i|Z_i = 1) \cdot \Pr[Z_i = 1]}{f_\theta(Y_i|Z_i = 0) \cdot \Pr[Z_i = 0] + f_\theta(Y_i|Z_i = 1) \cdot \Pr[Z_i = 1]} \\ &= \frac{g_{\theta_1}(Y_i) \cdot \pi}{g_{\theta_1}(Y_i) \cdot (1 - \pi) + g_{\theta_2}(Y_i) \cdot \pi}\end{aligned} \tag{32}$$

- Setting Z_i to its expected value γ_i is called a *soft assignment*

The EM Algorithm for the Two Component Mixture

1. Take initial guesses for the parameters $\hat{\mu}_1, \hat{\sigma}_1^2, \hat{\mu}_2, \hat{\sigma}_2^2, \hat{\pi}$
   - Construct the initial guess for $\hat{\mu}_1, \hat{\mu}_2$ by simply choosing two of the y_i at random
   - Set both $\hat{\sigma}_1^2, \hat{\sigma}_2^2$ to the overall sample variance, i.e.,
   $$\hat{\sigma}_1^2 = \hat{\sigma}_2^2 = \sum_{i=1}^{N} (y_i - \overline{y}_i)^2 / N \tag{33}$$
   - Set the starting mixing proportion to $\hat{\pi} = 0.5$
2. *Expectation Step*: compute the responsabilities
$$\gamma_i = \frac{g_{\theta_1}(Y_i) \cdot \pi}{g_{\theta_1}(Y_i) \cdot (1 - \pi) + g_{\theta_2}(Y_i) \cdot \pi}, \quad i = 1, 2, \ldots, N$$
3. *Maximation Step*: compute the weighted means and variances:
$$\begin{aligned}\hat{\mu}_1 &= \frac{\sum_{i=1}^N (1 - \hat{\gamma}_i) y_i}{\sum_{i=1}^N (1 - \hat{\gamma}_i)}, & \hat{\sigma}_1^2 &= \frac{\sum_{i=1}^N (1 - \hat{\gamma}_i)(y_i - \hat{\mu}_1)^2}{\sum_{i=1}^N (1 - \hat{\gamma}_i)} \\ \hat{\mu}_2 &= \frac{\sum_{i=1}^N \hat{\gamma}_i y_i}{\sum_{i=1}^N \hat{\gamma}_i}, & \hat{\sigma}_2^2 &= \frac{\sum_{i=1}^N \hat{\gamma}_i (y_i - \hat{\mu}_1)^2}{\sum_{i=1}^N (1 - \hat{\gamma}_i)}\end{aligned}$$

The EM Algorithm for the k Component Mixture

- $\mathbf{Y}_i \in \mathbb{R}^p$ instead of being one-dimensional
- Mixture of k components instead of just 2
- The density $f_\theta(\mathbf{y})$ is now defined as

$$\boxed{f_\theta(\mathbf{y}) = \sum_{j=1}^{k} \pi_j g_j(\mathbf{y}; \theta)} \quad , \tag{34}$$

  with $\theta_j = (\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j), j = 1, \ldots, k$ and $\theta = (\pi_1, \ldots, \pi_k; \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k; \boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_k)$, $\boldsymbol{\Sigma}_j$ being the $p \times p$ positive-definite covariance matrix
- The indicator matrix of latent variables $\mathbf{Z}$ is introduced, with $Z_{ij} \in \{0, 1\}$ for an observation $i \in \{1, \ldots, n\}$ and component $j \in \{1, \ldots, k\}$ with the meaning that $Z_{ij} = 1$ if the observation i is in *group* j
- The augmented data is given by $(\mathbf{Y}_i, \mathbf{Z}_i) \in \mathbb{R}^p \times \{0, 1\}^k$
- The complete likelihood is defined as

$$\boxed{L_c(\theta; \mathbf{Y}_1, \ldots, \mathbf{Y}_n; z_1, \ldots, z_n) = \prod_{i=1}^{n} \prod_{j=1}^{k} [\pi_j \, g_{\theta_j}(\mathbf{Y}_i)]^{Z_{ij}}} \tag{35}$$

- The responsability γ_{ij} is given by

$$\boxed{\gamma_{ij} = \frac{\pi_j \cdot g_{\theta_j}(\mathbf{Y}_i)}{\sum_{r=1}^{k} \pi_r \cdot g_{\theta_r}(\mathbf{Y}_i)}} \tag{36}$$

Lecture 6 (Week 43)

Intro to Missing Data (9)

- $Y = (y_{ij})$ is the $(n \times K)$ rectangular data set without missing values, with the ith row $y_i = (y_{i1}, \ldots, y_{iK})$ where y_{ij} is the value of the variable Y_j for subject i
- With missing data, $M = (M_{ij})$ is the missing-data indicator matrix, such that $M_{ij} = 1$ if y_{ij} is missing and $M_{ij} = 0$ if y_{ij} is present
  - $Y = (Y_{\text{obs}}, Y_{\text{mis}})$
- Data is called missing completely at random (MCAR) if

$$\boxed{f(M|Y, \phi) = f(M|\phi) \quad \forall Y, \phi \quad ,} \tag{37}$$

  i.e., missingness does not depend on the values of the data Y (missing or observed)

  - The probability of being missing is the same for all cases
  - *Example*: Take a random sample of a population, where each member has the same chance of being included in the sample. The (unobserved) data of members in the population that were not included in the sample are MCAR

- Data is called missing at random (MAR) if

$$\boxed{f(M|Y,\phi) = f(M|Y_{\text{obs}},\phi) \quad \forall Y_{\text{mis}},\phi \quad ,} \tag{38}$$

  i.e., missingnes depends only on the components Y_{obs} of Y that are observed, and not on the components that are missing
  - The probability of being missing is the same only within groups defined by the observed data
  - *Example:* We take a sample from a population, where the probability to be included depends on some known property

- If neither MCAR nor MAR holds, then we speak of missing not at random (MNAR)
  - MNAR means that the probability of being missing varies for reasons that are unknown to us, i.e., it could depend on the value of the missing data itself

The EM Algorithm (DLR77 Notation (3))

- Two sample spaces exist:
  1. $\mathcal{Y}$, the sample space of observed (incomplete) data $\mathbf{y}$
  2. $\mathcal{X}$, the sample space of the unobserved (complete) data $\mathbf{x}$
- A many-to-one mapping from $\mathcal{X}$ to $\mathcal{Y}$ exists
- $\mathbf{x}$ may only be observed indirectly through $\mathbf{y}$, i.e.,

$$\mathbf{x} \to \mathbf{y}(\mathbf{x}) \tag{39}$$

  - It follows that $\mathbf{x} \in \mathcal{X}(\mathbf{y})$, where $\mathcal{X}(\mathbf{y})$ is determined by the equation $\mathbf{y} = \mathbf{y}(\mathbf{x})$ and $\mathbf{y}$ is the observed data
- The existance of a family of sampling densities $f(x|\theta)$, for the complete data, and $g(\mathbf{y}|\theta)$ for the incomplete data is assumed
- Both density are related through

$$g(\mathbf{y}|\theta) = \int_{\mathcal{X}(\mathbf{y})} f(w|\theta)\mathrm{d}\mathbf{x} \tag{40}$$

  - The incomplete density $g(\cdot)$ is obtained from the complete density $f(\theta)$ by *integrating out* the unobserved data x over its sample space $\mathcal{X}(y)$

  - $f(\cdot|\theta)$ is the joint density of the observable and unobservable data and $g(\cdot|\theta)$ the corresponding marginal density of the observable data
  - For a given incomplete data specification $g(\cdot|\theta)$, many possible complete data specifications $f(\cdot|\theta)$ may be defined

- Define

$$L(\mathbf{\Phi}) = \log g(\mathbf{y}|\mathbf{\Phi}) \tag{41}$$

- Further, define $k(\mathbf{x}|\mathbf{y}, \mathbf{\Phi}) = f(\mathbf{x}|\mathbf{\Phi})/g(\mathbf{y}|\mathbf{\Phi})$ such that L can be rewritten to

$$L(\mathbf{\Phi}) = \log f(\mathbf{x}|\mathbf{\Phi}) - \log k(\mathbf{x}|\mathbf{y}, \mathbf{\Phi}) \tag{42}$$

- Define the *complete log likelihood function* as

$$\boxed{Q(\mathbf{\Phi}'|\mathbf{\Phi}) = E(\log f(\mathbf{x}|\mathbf{\Phi}')|\mathbf{y}, \mathbf{\Phi}) \quad ,} \tag{43}$$

  where $f(\mathbf{x}|\mathbf{\Phi}')$ is the likelihood function (or family of sampling densities) defined as

$$f(\mathbf{x}|\mathbf{\Phi}') = \prod_{i=1}^{N} f(\mathbf{x}_i|\mathbf{\Phi}') \tag{44}$$

  with $\mathbf{x}_i$ being the ith column of the complete data matrix $\mathbf{x}$
  - $\mathbf{\Phi}$ is a provisional guess for the parameter vector $\mathbf{\Phi}'$

Algorithm

E-Step: Calculate the conditional expectation of the complete log likelihood given the observations $\mathbf{y}$ and the current guess of the parameter vector $\mathbf{\Phi}'$, i.e., $Q(\mathbf{\Phi}'|\mathbf{\Phi})$

M-Step: Choose $\mathbf{\Phi}$ as the value of $\mathbf{\Phi}'$ which maximizes $Q(\mathbf{\Phi}'|\mathbf{\Phi})$, i.e.,

$$\mathbf{\Phi} = \max_{\mathbf{\Phi}'} Q(\mathbf{\Phi}'|\mathbf{\Phi}) \tag{45}$$

General Properties of the EM Algorithm

- Define for convenience

$$H(\mathbf{\Phi}'|\mathbf{\Phi}) = E(\log k(\mathbf{x}|\mathbf{y}, \mathbf{\Phi}')|\mathbf{y}, \mathbf{\Phi}) \tag{46}$$

- Rewrite Q as

$$\boxed{Q(\mathbf{\Phi}'|\mathbf{\Phi}) = L(\mathbf{\Phi}') + H(\mathbf{\Phi}'|\mathbf{\Phi})} \tag{47}$$

- The term "iterative algorithm" means a rule applicable to any starting point, i.e., a mapping $\boldsymbol{\Phi} \to \mathbf{M}(\boldsymbol{\Phi})$ from Ω to Ω such that each step $\boldsymbol{\Phi}^{(p)} \to \boldsymbol{\Phi}^{(p+1)}$ is defined by

$$\boldsymbol{\Phi}^{(p+1)} = \mathbf{M}(\boldsymbol{\Phi}^{(p)}) \tag{48}$$

  - An iterative algorithm with mapping $\mathbf{M}(\boldsymbol{\Phi})$ is a generalized EM (GEM) algorithm if

$$Q(\mathbf{M}(\boldsymbol{\Phi})|\boldsymbol{\Phi}) \geq Q(\boldsymbol{\Phi}|\boldsymbol{\Phi}) \quad \forall \boldsymbol{\Phi} \in \Omega \tag{49}$$

Theorem 1: For every GEM algorithm

$$\boxed{L(\mathbf{M}(\boldsymbol{\Phi})) \geq L(\boldsymbol{\Phi}) \quad \forall \boldsymbol{\Phi} \in \Omega \quad ,} \tag{50}$$

where equality holds if and only if both

$$Q(\mathbf{M}(\boldsymbol{\Phi})|\boldsymbol{\Phi}) = Q(\boldsymbol{\Phi}|\boldsymbol{\Phi}) \tag{51}$$

and

$$k(\mathbf{x}|\mathbf{y}, \mathbf{M}(\boldsymbol{\Phi})) = k(\mathbf{x}|\mathbf{y}, \boldsymbol{\Phi}) \tag{52}$$

almost everywhere.

Proof:

$$\boxed{L(\mathbf{M}(\boldsymbol{\Phi})) - L(\boldsymbol{\Phi}) = \underbrace{[Q(\mathbf{M}(\boldsymbol{\Phi})|\boldsymbol{\Phi}) - Q(\boldsymbol{\Phi}|\boldsymbol{\Phi})]}_{\geq 0, \text{ because Eq. (49)}} + \underbrace{[H(\boldsymbol{\Phi}|\boldsymbol{\Phi}) - H(\mathbf{M}(\boldsymbol{\Phi})|\boldsymbol{\Phi})]}_{\geq 0, \text{ and } = 0 \text{ iff } k(\mathbf{x}|\mathbf{y},\mathbf{M}(\boldsymbol{\Phi}))=k(\mathbf{x}|\mathbf{y},\boldsymbol{\Phi})}}$$

Lecture 7 (Week 44)

Listwise Deletion (Complete Case Analysis) (15)

- With listwise deletion, an entire record is excluded from analysis if any single value is missing
  - *Example:* See Table (1)
- *Advantages under MCAR:*
  1. Produces unbiased estimates of means, variances and regression weights
  2. Produces standard errors and significance levels that are correct for the reduced subset of data
- *Disadvantages if not MCAR:*
  1. Can severely bias estimates of means, regression coefficients and correlations

Table 1: Sample data with missing values. With listwise deletion, rows 1 and 4 would be excluded from the analysis

Subject	Age	Gender
1	28	NA
2	30	m
3	26	f
4	NA	f

Pairwise Deletion (Available-Case Analysis) (15)

- Mean of each variable X_i is based on all cases with observed data on X_i
- For the correlation and covariance, all data are taken on which both X_i and X_j ($i \neq j$) have non-missing scores
- *Advantages under MCAR:*
  1. The method produces consistent estimates of mean, correlations and covariances
- *Disadvantages if not MCAR:*
  1. Estimates can be biased
  2. The correlation matrix may not be positive-definite, which is a requirement for most multivariate procedures
     - Among other things, a positive-definite matrix has positive eigenvalues and a unique Cholesky decomposition
  3. Correlations outside of the range $[-1, 1]$ can occur, since the method works with different subsets for the covariances and variances
  4. Due to the different subset sizes, it is not clear which sample size should be used for calculating standard errors

Mean Imputation (15)

- A quick fix for the missing data is to replace them by the mean
- *Disadvantages:*
  1. Underestimates the variance
  2. Disturbs the relation between variables
  3. Biases almost any estimate other than the mean
  4. If the data are not MCAR, it biases the estimate of the mean

Regression Imputation (15)

- First build a model from the observed data
- Next, predictions for the incomplete cases are calculated under the fitted model, and serve as replacements for the missing data. In other words, RI performs

$$\dot{y} = \hat{\beta}_0 + X_{\text{mis}}\hat{\beta}_1 \quad , \tag{53}$$

  where $\dot{y}$ contains the imputed values in y and $\hat{\beta}_0, \hat{\beta}_1$ are least squares estimates calculated from the observed data
- *Advantages under MCAR:*
  1. Yields unbiased estimates of the means and the regression weights if the explanatory variables are complete

Stochastic Regression Imputation (15)

- SRI is a refinement of regression imputation that adds noise to the predictions. In other words, SRI performs

$$\dot{y} = \hat{\beta}_0 + X_{\text{mis}}\hat{\beta}_1 + \dot{\epsilon} \quad , \tag{54}$$

  where $\dot{y}$ contains the imputed values in y, $\hat{\beta}_0, \hat{\beta}_1$ are least squares estimates calculated from the observed data and $\dot{\epsilon}$ is randomly drawn from the normal distribution as $\dot{\epsilon} \sim \mathcal{N}(0, \hat{\sigma}^2)$
- *Disadvantages:*
  - In the example shown in the book, the method produces negative values for the Ozone concentrations, which of course are implausible

Last Observation Carried Forward (LOCF) and Baseline Observation Carried Forward (BOCF) (15)

- The idea is to take the last observed value as a replacement for the missing data

Indicator Method (15)

- The indicator method replaces each missing value by a zero and extends the regression model by the response indicator
  - The procedure is applied to each incomplete variable
  - The user analyzes the extended model
- *Disadvantages:*

1. The method can yield severly biased regression estimates, even under MCAR and for low amounts of missing data
2. The conditions under which the indicator method works are often difficult to achieve in practice
3. The method also does not allow for missing data in the outcomes

Lecture 8 (Week 45)

MCAR and MAR (9)

- Rubin's theory formalized the concept of analyzing data with missing values by treating the missing-data indicators as random variables and assigning them a distribution
- M is the missing-data indicator, with

$$M_{ij} = \begin{cases} 1 & y_{ij} \text{ missing} \\ 0 & y_{ij} \text{ not missing} \end{cases} \tag{55}$$

  - The probability that M takes a value $m = (m_1, \ldots, m_n)$ given that Y takes the value $y = (y_1, \ldots, y_n)$ is $f(M|Y, \theta)$
  - $f(M|Y, \theta)$ corresponds to the process that causes missing data
- R is the response indicator, with $R = 1 - M$ and

$$R_{ij} = \begin{cases} 1 & y_{ij} \text{ observed} \\ 0 & y_{ij} \text{ not observed} \end{cases} \tag{56}$$

  - M is the missing-data indicator
- In general, we would not expect the distribution of R to be unrelated to Y, so a probability model for R is proposed. The distribution of R may depend on $Y = (Y_{\text{obs}}, Y_{\text{mis}})$ and this relation is described by the missing data model $\Pr(R|Y_{\text{obs}}, Y_{\text{mis}}, \psi)$, where ψ contains the parameters for the missing data model
- The data are said to be MCAR if

$$\boxed{\Pr(R = 0|Y_{\text{obs}}, Y_{\text{mis}}, \psi) = \Pr(R = 0|\psi)} \tag{57}$$

  so the probability of being missing depends only on some parameters ψ, the overall probability of being missing
- The data are said to be MAR if

$$\boxed{\Pr(R = 0|Y_{\text{obs}}, Y_{\text{mis}}, \psi) = \Pr(R = 0|Y_{\text{obs}}, \psi)} \tag{58}$$

  so the missingness probability may depend on observed information, including any design factors

Ignorability (9)

- The assumption of ignorability is essentially the belief on the part of the user that the available data are sufficient to correct for the effects of the missing data
- The missing data mechanism is ignorable for likelihood inference if:
  1. The missing data are MAR (see Eq. (58)), which implies that the distribution of the missing data mechanism does not depend on the missing data Y_{mis}
  2. The parameters θ of the data model (for the full data Y) and the parameters ψ of the missingness mechanism (that relates Y to R) are distinct
     - The full model specifies the joint distribution of M and Y as

$$\boxed{f(Y, M|\theta, \psi) = f(Y|\theta)\, f(M|Y, \psi), \quad (\theta, \psi) \in \Omega_{\theta,\psi}} \tag{59}$$

     where $\Omega_{\theta,\psi}$ is the parameter space of (θ, ψ)

Implications of Ignorability (15)

- In imputation, we want to draw synthetic observations from the posterior distribution of the missing data, given the observed data and given the process that generated the missing data; the distribution is denoted as $f(Y_{\text{mis}}|Y_{\text{obs}}, R)$
- If the nonresponse is ignorable, then this distribution does not depend on R, i.e.,

$$\boxed{f(Y_{\text{mis}}|Y_{\text{obs}}, R) = f(Y_{\text{mis}}|Y_{\text{obs}})} \tag{60}$$

  which implies

$$\boxed{f(Y_{\text{mis}}|Y_{\text{obs}}, R = 1) = f(Y_{\text{mis}}|Y_{\text{obs}}, R = 0)} \tag{61}$$

  so the distribution of the data Y is the same in the response and nonresponse groups. Imputation thus makes sense.
  - It follows that if the missing data model is ignorable we can model the posterior distribution $f(Y|Y_{\text{obs}}, R = 1)$ from the observed data, and use this model to create imputations for the missing data
  - Under MNAR, if the nonresponse is nonignorable, we have

$$f(Y_{\text{mis}}|Y_{\text{obs}}, R = 1) \neq f(Y_{\text{mis}}|Y_{\text{obs}}, R = 0) \tag{62}$$

Multiple Imputation (15)

- Let Q be a scientific estimand, e.g., the population mean
  - We can only calculate Q if the population data are fully known

- Goal of multiple imputation is to find an estimate $\hat{Q}$ that is unbiased and confidence valid
  - Unbiasedness means that the average $\hat{Q}$ over all possible samples Y from the population is equal to Q, i.e.,

$$\mathbb{E}[\hat{Q}|Y] = Q \tag{63}$$

  - Let U be the estimated variance-covariance matrix of $\hat{Q}$; then, this estimate is confidence valid if

$$\mathbb{E}[U|Y] \geq \mathrm{Var}(\hat{Q}|Y) \quad , \tag{64}$$

    where $\mathrm{Var}(\hat{Q}|Y)$ is the variance caused by the sampling process

Sources of Variation (15)

- The actual value of Q is unknown if some of the population data are unknown
- The possible values of Q given our knowledge of the data Y_{obs} are captured by the posterior distribution $f(Q|Y_{\text{obs}})$
- Combining the results of m repeated imputations results in the combined estimate defined as

$$\overline{Q} = \frac{1}{m}\sum_{l=1}^{m}\hat{Q}_l \quad , \tag{65}$$

  where $\hat{Q}_l$ is the estimate of the lth repeated imputation

- The posterior variance of $f(Q|Y_{\text{obs}})$ is given as

$$\mathrm{Var}(Q|Y_{\text{obs}}) = \overbrace{\mathbb{E}[\mathrm{Var}(Q|Y_{\text{obs}}, Y_{\text{mis}})|Y_{\text{obs}}]}^{\text{within-variance}} \tag{66}$$

$$+ \underbrace{\mathrm{Var}(\mathbb{E}[Q|Y_{\text{obs}}, Y_{\text{mis}}]|Y_{\text{obs}})}_{\text{between-variance}} \tag{67}$$

  - Within-variance: average of the repeated complete data posterior variances of Q
  - Between-variance: variance between the complete data posterior means of Q

- The posterior variance $\mathrm{Var}(Q|Y_{\text{obs}})$ can be defined as

$$T = \overline{U} + B + B/m \quad , \tag{68}$$

where

$$B = \frac{1}{m}\sum_{l=1}^{m}(\hat{Q}_l - \overline{Q})(\hat{Q}_l - \overline{Q})' \tag{69}$$

is the standard unbiased estimate of the variance between the m complete data estimates and

$$\overline{U} = \frac{1}{m}\sum_{l=1}^{m}\overline{U}_l \tag{70}$$

is the average of the complete-data variance where the term $\overline{U}_l$ is the variance-covariance matrix of $\hat{Q}_l$ obtained from the lth imputation

- The term B/m is the extra simulation variance caused by the fact that $\overline{Q}$ itself is estimated for finite m

Excursion: Likelihood-Based Inference

Maximum Likelihood Inference (9) (5)

- Y denotes the data, where Y may be scalar, vector-valued or matrix-valued (according to context)
- The data are assumed to be generated by a model described by a probability or density function $f(Y|\theta)$, indexed by a scalar or vector parameter θ, where θ lies in the parameter space Ω_θ, i.e.,

$$y_i \sim f(y_i|\theta) \tag{71}$$

  - Given the model and parameter θ, $f(Y|\theta)$ is a function of Y that gives the probabilities or densities of various Y values

- Given the data value Y, the *likelihood function* $L(\theta|Y)$ is any function of $\theta \in \Omega_\theta$ proportional to $f(Y|\theta)$, i.e., $L(\theta|Y) \propto f(Y|\theta)$ and is defined by

$$L(\theta|Y) = \prod_{i=1}^{N} f(y_i|\theta) \tag{72}$$

  - The likelihood function describes the probability of the observed data Y under the model $f(Y|\theta)$
    - $L(\theta|Y)$ is a function of the parameter θ for fixed Y
    - $f(Y|\theta)$ is a function of Y for fixed θ
  - By definition, $L(\theta|Y) = 0$ for any $\theta \notin \Omega_\theta$

- The *maximum likelihood estimate* of θ is a value of $\theta \in \Omega_\theta$ that maximizes the likelihood $L(\theta|Y)$, or equivalently, the loglikelihood $l(\theta|Y) = \ln L(\theta|Y)$

  - Suppose that for fixed data Y, two possible values of θ are being considered, θ' and θ''. Suppose further, that $L(\theta'|Y) = 2L(\theta''|Y)$. It is now reasonable to say, that the observed outcome Y is twice as likely under $\theta = \theta'$ as under $\theta = \theta''$
  - More generally, consider a value of θ, say $\hat{\theta}$, such that $L(\hat{\theta}|Y) \geq L(\theta|Y)$ for all other possible θ; the observed outcome Y is then at least as likely under $\hat{\theta}$ as under any other value of θ
    * The value $\theta = \hat{\theta}$ is thus best supported by the data
  - The value of θ that maximizes the likelihood function is of interest

- The *likelihood equation* is defined as

$$\frac{\partial l(\theta|Y)}{\partial \theta} = 0, \tag{73}$$

  and the ML estimate can be found by solving this equation for θ

Likelihood-Based Inference with Incomplete Data (9)

- Let Y again denote the data, with $Y = (Y_{\text{obs}}, Y_{\text{mis}})$, where Y_{obs} denotes the observed values and Y_{mis} denotes the missing values
- Let $f(Y|\theta) \equiv f(Y_{\text{obs}}, Y_{\text{mis}}|\theta)$ be the joint distribution
- The marginal probability density of Y_{obs} is defined as

$$f(Y_{\text{obs}}|\theta) = \int f(Y_{\text{obs}}, Y_{\text{mis}}|\theta)\, \mathrm{d}Y_{\text{mis}} \tag{74}$$

- The likelihood of θ based on data Y_{obs} ignoring the missing-data mechanism is proportional to $f(Y_{\text{obs}}|\theta)$, i.e.,

$$L_{\text{ign}}(\theta|Y_{\text{obs}}) \propto f(Y_{\text{obs}}|\theta), \quad \theta \in \Omega_\theta \tag{75}$$

  - Inferences about θ can be based on this likelihood, $L_{\text{ign}}(\theta|Y_{\text{obs}})$, providing the mechanism leading to incomplete data can be ignored (see section *Ignorability*

Lecture 9 (Week 46)

The Need for Imputation

- Let $\mathbf{x} \sim f_\theta(\mathbf{x})\mathrm{d}x \quad \mathbf{x} \in \mathbb{R}^p$. Assume regression models for $x_j|\{x_k, k \neq j\} \quad \forall j \in \{1, \ldots, p\}$, resulting in the imputation

$$\boxed{\hat{x}_j = f_{\hat{\theta}}(\mathbf{x}_{k \neq j})} \tag{76}$$

  - Each regression is $O(p^2)$, resulting in an overall complexity of $O(p^3)$
  - Large percentage of entries are missing, e.g., $\sum M_{ij}/(np) = 0.988$ for the Netflix dataset

- Good methods are characterized by a small value for

$$\boxed{\frac{\sum_{M_{ij}=1} \|X_{ij} - \hat{X}_{ij}\|}{\|X_{ij}\|}} \tag{77}$$

  where $\mathbf{X}$ are the true data and $\hat{\mathbf{X}}$ contains the imputed values
  - Remove around 10% of the available data and use the true data and the imputed values to calculate the error

Imputation using Singular Value Decomposition (6)

- Expression matrix $\mathbf{X} \in \mathbb{R}^{N \times p}$
  - Rows are genes
  - Columns are observations (DNA arrays)
  - $\mathbf{X} = (\mathbf{X}^c, \mathbf{X}^m)$, where $\mathbf{X}^c \in \mathbb{R}^{c \times p}$ is the subset of complete genes
- The truncated singular value decomposition (SVD) of $\mathbf{X}^c$ is given by

$$\boxed{\underbrace{\hat{\mathbf{X}}^c_J}_{c \times p} = \underbrace{\mathbf{U}_J}_{c \times J} \underbrace{\mathbf{D}_J}_{J \times J} \underbrace{\mathbf{V}^T_J}_{J \times p}} \tag{78}$$

  - $\mathbf{D}_J$ is a diagonal matrix containing the leading $J \leq \min(p, N)$ singular values of $\mathbf{X}^c$; we now assume that $p < N$, thus resulting in $J \leq p$
    - Singular values are the square roots of the eigenvalues of $\mathbf{D}^H\mathbf{D}$, where $\mathbf{D}^H$ is the conjugate transpose of $\mathbf{D}$
  - $\mathbf{V}_J, \mathbf{U}_J$ are the corresponding orthogonal matrices of J right and left singular vectors with

$$\mathbf{U}^T_J \mathbf{U}_J = I \tag{79}$$

$$\mathbf{V}^T_J \mathbf{V}_J = I \tag{80}$$

SVD Imputation Using a Clean Training Set

- The basic paradigm is:
  1. Generate eigen-genes from the complete data
  2. Impute the missing cells for a gene by regressing its non-missing entries on the eigen-genes, and use the regression function to predict the expression values at the missing locations

- Rank-J SVD: provide the best rank-J matrix approximation to $\mathbf{X}^c$, i.e., it solves the problem

$$\boxed{\min_{\mathbf{M}\,\text{rank}\,J} \|\mathbf{X}^c - \mathbf{M}\|_F^2} \tag{81}$$

  where $\|\cdot\|_F$ is the Frobenius norm

- Let $\mathbf{x}$ be any row of $\mathbf{X}^c$, and consider the least squares regression of the p values in $\mathbf{x}$ on the eigen-genes $v_1, \ldots, v_J \in \mathbb{R}^p$. This regression solves the least squares approximation problem

$$\boxed{\min_{\beta} \|\mathbf{x} - \mathbf{V}_J\boldsymbol{\beta}\|^2 = \min_{\boldsymbol{\beta}} \sum_{l=1}^{p} \left(x_l - \sum_{j=1}^{J} v_{lj}\beta_j \right)^2} \tag{82}$$

  with solution

$$\boxed{\hat{\boldsymbol{\beta}} = (\mathbf{V}_J^T\mathbf{V}_J)^{-1}\mathbf{V}_J^T\mathbf{x} \overset{(\mathbf{V}_J\,\text{orthogonal})}{=} \mathbf{V}_J^T\mathbf{x}} \tag{83}$$

  and fitted values

$$\boxed{\hat{\mathbf{X}} = \mathbf{V}_J\hat{\boldsymbol{\beta}}} \tag{84}$$

  - Once the $\mathbf{V}_J$ are found, the SVD approximates each row of $\mathbf{X}^c$ by its fitted vector obtained by regression on $\mathbf{V}_J$
  - $\mathbf{X}^c\mathbf{V}_J = \mathbf{U}_J\mathbf{D}_J$ gives all the regression coefficients $\hat{\boldsymbol{\beta}}$
  - $\hat{\mathbf{X}}^c = \mathbf{U}_J\mathbf{D}_J\mathbf{V}_J^T$ gives all the fitted values

- Impute the missing values of $\mathbf{X}^m$ by the regression

$$\min_{\boldsymbol{\beta}} \sum_{l\,\text{non-missing}}^{p} \left(x_l - \sum_{j=1}^{J} v_{lj}\beta_j \right)^2 \tag{85}$$

SVD Imputation Using All the Data

- Previous approach implies the availability of a reasonable set of complete genes
- Eqs. (82) and (85) do not include intercepts, but it is customary to center the data before computing the SVD
  - The intercept amounts to subtracting the ith row-mean

$$m_i^c = 1/p \sum_{l=1}^{p} X_{il}^c \tag{86}$$

    from each element in row i

- Include all the data, and solve

$$\boxed{\min_{\mathbf{U}_J,\mathbf{V}_J,\mathbf{D}_J} \|\mathbf{X} - mI^T - \mathbf{U}_J\mathbf{D}_J\mathbf{V}_J^T\|^* \quad ,} \tag{87}$$

  where $\|\cdot\|^*$ is a squared matrix norm, which sums the squares of all the elements, ignoring those entries where $\mathbf{X}$ has missing data, and where $m = (m_1^c, m_2^c, \ldots, m_N^c)$ is a vector of means, one element per row of $\mathbf{X}$ (see Eq. (86))

Iterative Algorithm:

1. Set missing entries to the mean of the non-missing entries for each row, producing a complete matrix $\mathbf{X}^0$
2. Compute SVD solution to Eq. (87) for the complete matrix $\mathbf{X}^i$, and produce $\mathbf{X}^{i+1}$ by replacing the missing values in $\mathbf{X}$ by the fitted values from this solution, i.e., by regressing on V_j (see Eqs. (82), (83) and (84). Note that in Eq. (82) the intercept is missing and should be modified to take it into account)
3. Set $i \leftarrow i + 1$ and repeat step 2 until

$$\|\mathbf{M}^i - \mathbf{M}^{i+1}\|/\|\mathbf{M}^i\| < \epsilon \quad , \tag{88}$$

   where ϵ is some threshold (e.g. 0.01) and $\mathbf{M}^i$ is the entire fitted matrix

Choice of Best Rank J by Cross-Validation

- See chapter "Choice of Tuning Parameter k by Cross-Validation", and substitute k with the rank J; the cross-validation procedure is basically the same

k-Nearest Neighbour Imputation (14)

- k-NN assumes MAR
- NN approaches are useful in high dimensional problems in which multiple imputation cannot be applied
- Drawback of k-NN is that its performance depends on the tuning parameter k
- k-NN is a localized approach that uses a weighted average of NNs based on L_q distances. For the high-dimensional case, a new distance that explicitly uses the correlation among variables is considered

Distances and Computation of Neareast Neighbours

- Let $\mathbf{X} = (x_{is}) \in \mathbb{R}^{N \times p}$
- Let $\mathbf{R} = (r_{is})$ be defined as

$$r_{is} = \begin{cases} 1 & x_{is} \text{ observed} \\ 0 & x_{is} \text{ not observed} \end{cases} \tag{89}$$

- Distances between two observations $\mathbf{x}_i$ and $\mathbf{x}_j$ (rows in the data matrix $\mathbf{X}$) can be computed using the metric given by

$$\boxed{d_q(\mathbf{x}_i, \mathbf{x}_j) = \left[\frac{1}{m_{ij}} \sum_{s=1}^{p} |x_{is} - x_{js}|^q 1(r_{is} = 1)\, I(r_{js} = 1) \right]^{1/q} \quad ,} \tag{90}$$

  where $I(x)$ is the indicator function defined in Eq. (7) and

$$\boxed{m_{ij} = \sum_{s=1}^{p} I(r_{is} = 1)\, I(r_{js} = 1) \quad ,} \tag{91}$$

  i.e., the L_q distance only uses the components of the vectors for which observations in both vectors are available

Imputation Procedure

- Consider the imputation for $\mathbf{x}_i$ in component s, i.e., $r_{is} = 0$ (component is missing)
- The k NNs used for the imputation estimate for $\mathbf{x}_i$ are determined from the corresponding $(c \times p)$-dimensional reduced data set $\mathbf{X}^c = (x_{ij}, r_{is} = 1)$ to obtain

$$\boxed{\mathbf{x}_{(1)}, \ldots, \mathbf{x}_{(k)} \quad \text{with } d(\mathbf{x}_i, \mathbf{x}_{(1)}) \leq \cdots \leq d(\mathbf{x}_i, \mathbf{x}_{(k)}) \quad ,} \tag{92}$$

  where $\mathbf{x}_{(j)}^T = (x_{(j)1}, \ldots, x_{(j)p})$ denotes the jth NNs

- The imputation value for a fixed k is then given by

$$\boxed{\hat{x}_{is} = \frac{1}{k} \sum_{j=1}^{k} x_{(j)s}} \tag{93}$$

  – The missing value in the sth component of observation vector $\mathbf{x}_i$ is replaced by the average of the corresponding values of the k NNs

Choice of Tuning Parameter k by Cross-Validation

1. Generate completely at random (MCAR) m^* artificially missing values from the available data $\{x_{is} : r_{is} = 1\}$

2. Let $\mathbf{X}^m$ denote denote the $n \times p$ data matrix that contains the originally missing values ($\{x_{is} : r_{is} = 0\}$) and the artificially missing values ($\{x^*_{is} : r^*_{is} = 0\}$). The mean absolute imputation error (MAIE) is defined as

$$\boxed{\text{MAIE}(\mathbf{X}^*) = \frac{1}{m^*} \sum_{x_{is}: r^*_{is}=0} |x^*_{is} - x^*_{is\,\text{(imputed)}}|} \tag{94}$$

3. Step 2. is repeated C times, yielding the averaged value

$$\boxed{\text{MAIE}_{\text{CV}} = \frac{1}{R} \sum_{r=1}^{R} \text{MAIE}(\mathbf{X}^*_r) \quad ,} \tag{95}$$

   replication

4. The cross-validated mean squared imputation error (MSIE) is defined as

$$\boxed{\text{MSIE}(\mathbf{X}^*) = \frac{1}{m^*} \sum_{x_{is}: r^*_{is}=0} \left(x^*_{is} - x^*_{is\,\text{(imputed)}}\right)^2} \tag{96}$$

Algorithm:

1. For a specific value of k:
   a) Artificially delete m^* value/s in the data matrix (MCAR)
   b) Impute these missing values and calculate the MSIE or MAIE
   c) Repeat a) and b) C times to obtain MSIE_{CV} or MAIE_{CV}

2. Repeat steps a)-c) for all values of k and choose the parameter with the minimum value of MSIE_{CV} or MAIE_{CV} as the optimal k

k-NN-Based Imputation vs. SVD-Based Imputation (13)

- k-NN imputation is accurate in the esimation of missing values for genes that are expressed in small clusters. Further, k-NN-based exhibits higher performance for both noisy time series and non-time series data

- SVD imputation yields best results on time-series data with low noise level. Under such conditions the method performs better than k-NN imputation if the right number of eigengenes is used for estimation

Lecture 10 (Week 47)

Soft-Impute (11)

- The goal is to find the lowest rank matrix $\mathbf{Z}$ which matches the matrix $\mathbf{X}$ containing the observed values. The optimization problem is thus stated as

$$\begin{aligned} &\text{minimize} && \operatorname{rank}(\mathbf{Z}) \\ &\text{subject to} && \sum_{(i,j)\in\Omega} (X_{ij} - Z_{ij})^2 \leq \delta \quad , \end{aligned} \tag{97}$$

  where $\delta \geq 0$ is a regularization parameter controlling the tolerance in training error and $\Omega = \{(i,j) : X_{ij} \text{ observed}\}$ denotes the indices of observed entries

- *Problem:* The rank constraint in ((97)) makes the problem for general Ω combinatorially hard. *Solution:* Reformulate ((97)) to a convex problem by introducing the nuclear norm $\|\mathbf{Z}\|_*$, or the sum of the singular values of $\mathbf{Z}$, thus resulting in the definition

$$\boxed{\begin{aligned} &\text{minimize} && \|\mathbf{Z}\|_* \\ &\text{subject to} && \sum_{(i,j)\in\Omega} (X_{ij} - Z_{ij})^2 \leq \delta \end{aligned}} \tag{98}$$

  – The nuclear norm is under many situations an effective convex relaxation to the rank constraint

- *Problem:* ((98)) can be solved efficiently for small problems using modern convex optimization software, but since these algorithms are based on second order methods, they can be quite expensive if the dimensions of the matrix get large. *Solution:* Reformulate ((98)) in Lagrange form

$$\min_{\mathbf{Z}} \frac{1}{2} \sum_{(i,j)\in\Omega} (X_{ij} - Z_{ij})^2 + \lambda \|\mathbf{Z}\|_* \quad , \tag{99}$$

  where $\lambda \geq 0$ is a regularization parameter controlling the nuclear norm of the minimizer $\hat{\mathbf{Z}}_\lambda$

- Soft-impute iteratively replaces the missing elements with those obtained from a soft-thresholded SVD

- Define the projection $P_\Omega(\mathbf{X})$ to be the matrix with the observed elements of $\mathbf{X}$ preserved, and the missing entries replaced by 0, i.e.,

$$\boxed{P_\Omega(\mathbf{X})(i,j) = \begin{cases} X_{ij} & \text{if } (i,j) \in \Omega \\ 0 & \text{if } (i,j) \notin \Omega \end{cases}} \tag{100}$$

– The complementary projection $P_\Omega^\perp$ is defined via

$$P_\Omega^\perp(\mathbf{X}) + P_\Omega(\mathbf{X}) = \mathbf{X} \tag{101}$$

- Let $\hat{\mathbf{X}}$ be the complete matrix containing all the entries of $\mathbf{X}$ and the imputed entries, i.e., $\hat{\mathbf{X}}$ has no missing values; then it follows from Eq. (101) that

$$\boxed{\mathbf{Z} = P_\Omega(\mathbf{X}) + P_\Omega^\perp(\hat{\mathbf{X}})} \tag{102}$$

- Using Eq. (100) it follows that

$$\sum_{(i,j)\in\Omega} (X_{ij} - Z_{ij})^2 \Leftrightarrow \|P_\Omega(\mathbf{X}) - P_\Omega(\mathbf{Z})\|_F^2 \quad , \tag{103}$$

where $\|\cdot\|_F$ is the Frobenius norm, and thus ((99)) can be reformulated to

$$\boxed{\min_{\mathbf{Z}} \frac{1}{2}\|P_\Omega(\mathbf{X}) - P_\Omega(\mathbf{Z})\|_F^2 + \lambda\|\mathbf{Z}\|_*} \tag{104}$$

- The solution to the optimization problem shown in ((104)) is given by

$$\boxed{\hat{\mathbf{Z}} = \mathcal{S}_\lambda(\mathbf{X})} \tag{105}$$

where

$$\boxed{\mathcal{S}_\lambda(\mathbf{X}) \equiv \mathbf{U}\mathbf{D}_\lambda\mathbf{V}^T \quad \text{with}\, \mathbf{D}_\lambda = \text{diag}[(d_1 - \lambda)_+, \ldots, (d_r - \lambda)_+]} \tag{106}$$

is the soft-thresholding operator and $\mathbf{U}\mathbf{D}\mathbf{V}^T$ is the SVD of $\mathbf{X}$ with $\mathbf{D} = \text{diag}[d_1, \ldots, d_r]$

Algorithm

$$\min_{\mathbf{Z}} \frac{1}{2}\|P_\Omega(\mathbf{X}) - P_\Omega(\mathbf{Z})\|_F^2 + \lambda\|\mathbf{Z}\|_* \tag{107}$$

1. Replace the missing entries in $\mathbf{X}$ with the corresponding entries from the current estimate $\hat{\mathbf{Z}} = \mathcal{S}_\lambda(\mathbf{X})$, i.e.,

$$\boxed{\hat{\mathbf{X}} \leftarrow P_\Omega(\mathbf{X}) + P_\Omega^\perp(\hat{\mathbf{Z}})} \tag{108}$$

2. Update $\hat{\mathbf{Z}}$ by computing the soft-thresholded SVD of $\hat{\mathbf{X}}$

$$\boxed{\begin{aligned} \hat{\mathbf{X}} &= \mathbf{U}\mathbf{D}\mathbf{V}^T && (109)\\ \hat{\mathbf{M}} &\leftarrow \mathbf{U}\mathbf{S}_\lambda(\mathbf{D})\mathbf{V}^T && (110) \end{aligned}}$$

where the soft-thresholding operator $\mathcal{S}_\lambda$ operates element-wise on the diagonal matrix $\mathbf{D}$, and replaces D_{ii} with $(D_{ii} - \lambda)_+$. With large λ many of the diagonal elements will be set to zero, leading to a low-solution for ((107))

Weighted k-Nearest Neighbour Imputation (14)

- *Problem:* In imputation based on the k NNs, the value of the first NN has the same importance as the kth NN. *Solution:* Use weights that account for the distance of the observations. This distances are determined by kernel functions, e.g. Gaussian or tricube
- The weighted imputation estimate is defined as

$$\boxed{\hat{x}_{is} = \sum_{j=1}^{k} w(\mathbf{x}_i, \mathbf{x}_{(j)}) x_{(j)s}} \tag{111}$$

  where $\mathbf{x}_{(j)}$ is the jth neighbour and the weights are given by

$$w(\mathbf{x}_i, \mathbf{x}_{(j)}) = K(d(\mathbf{x}_i, \mathbf{x}_{(j)})/\lambda) \Big/ \sum_{l=1}^{k} K(d(\mathbf{x}_i, \mathbf{x}_{(l)}) \quad , \tag{112}$$

  with $K(\cdot)$ being the kernel function and λ the tuning parameter
  - For small λ the weights decrease very strongly with distance
  - For $\lambda \to \infty$ all neighbours are of equal weight
  - λ can be choosen by cross-validation; see chapter "Choice of Tuning Parameter k by Cross-Validation" for a description of the algorithm (replace k with λ)

Lecture 11 (Week 48)

Directed Acyclic Graph (DAG) Models

Graph Terminology

- A graph $G = (\mathbf{V}, \mathbf{E})$ consists of vertices $\mathbf{V}$ and edges $\mathbf{E}$
- A directed acyclic graph (DAG) is a directed graph without directed cycles
- $i \to j \iff i$ is parent of j, i.e., $\mathrm{pa}(j) = \{i\}$
- $i \to j \to k \to l \iff i, j, k, l$ are all ancestors of l and descendants of i, i.e. $\mathrm{an}(l) = \{i, j, k, l\}$ and $\mathrm{desc}(i) = \{i, j, k, l\}$ respectively

DAGs and Random Variables

- A DAG model is a combination (G, f), where G is a DAG and f is a distribution that factorizes according to G
- Each node i in the DAG corresponds to a random variable X_i

- Chain rule for the joint probability distribution:

$$\boxed{f(x_1, \ldots, x_p) = f(x_1) f(x_2|x_1) \ldots f(x_p|x_1, \ldots, x_{p-1})} \tag{113}$$

  – *Problem:* It's rather expensive to store the complete joint distribution, since it requires a probability table of size 2^p, with p being the number of random variables

- The set of variables $\mathrm{pa}(j)$ is said to be *Markovian parents* of X_j if

$$\boxed{f(x_j|x_1, \ldots, x_{j-1}) = f(x_j|\mathrm{pa}(j))} \tag{114}$$

- From Eqs. (113) and (114) it follows that the underlying distribution is composed via

$$\boxed{f(x_1, \ldots, x_p) = \prod_{j=1}^{p} f(x_j|\mathrm{pa}(j))} \tag{115}$$

  i.e., we can draw a DAG accordingly, and the distribution is said to *factorize according to this DAG*

- $X \perp Y \iff X, Y$ are independent
  – *Example:* From $X_1 \perp X_3|X_2$ it follows that

$$f(x_1|x_2, x_3) = f(x_1|x_2), \tag{116}$$
$$f(x_3|x_1, x_2) = f(x_3|x_2) \tag{117}$$

  and for the joint distribution we thus have

$$\begin{aligned} f(x_1, x_2, x_3) &= f(x_1) f(x_2|x_1) f(x_3|x_1, x_2) \\ &= f(x_1) f(x_2|x_1) f(x_3|x_2) \end{aligned} \tag{118}$$

  resulting in the DAG $1 \to 2 \to 3$, which is only one of the $p!$ possibilities to construct the DAG for the given conditional independence, i.e., a distribution can factorize according to several DAGs

Uses of DAG Models

Estimating the joint density from low order conditional densities:

- Estimating the joint density is an expensive task
- If you know that the distribution factorizes according to a DAG, one only needs to estimate $f(x_i|\mathrm{pa}(i))$ for $i = 1, \ldots, p$; if the parent sets are small, only low order conditional densities need to be estimated

Reading off conditional independencies from the DAG:

- Markov models: the future is independent of the past given the present, i.e.,

$$1 \to 2 \to \cdots \to (t-1) \to t \to (t+1) \tag{119}$$

or

$$X_{t+1} \perp (X_{t-1}, X_{t-2}, \ldots, X_1) | X_t \tag{120}$$

- In DAG models, the Markov property can be expressed as

$$\boxed{S \perp \{\text{nondesc}(\mathbf{S}) \setminus \text{pa}(\mathbf{S})\} \,|\, \text{pa}(\mathbf{S})} \tag{121}$$

where $\mathbf{S}$ is any collection of nodes. This means that S is independent of its non-descendants given its parents

- Use d-separation to read off arbitrary conditional (in)dependencies [12]. If every path from node i to node i is d-separated by a set of nodes $\mathbf{S}$, then the random variables X_i and X_j are conditionally independent given $\mathbf{S}$, i.e. $X_i \perp X_j \mid \mathbf{S}$

  Rule 1 (Unconditional Separation) i and j are d-connected if there is an unblocked path between them, i.e., a path that can be traced without traversing a collider. A non-endpoint node k is a collider on a path if the path contains $\to k \leftarrow$, i.e., the arrows collide at k

  Rule 2 (Blocking by Conditioning) i and j are d-connected, conditioned on a set $\mathbf{S}$ of nodes, if there is a collider-free path between i and j that traverses no member of $\mathbf{S}$. If no such path exists, we say that i and j are d-separated by $\mathbf{S}$

  – *Example:* The following graph is given

$$x \quad \to \quad \boxed{r} \quad \to \quad s \quad \to \quad t \quad \leftarrow \quad u \quad \leftarrow \quad \boxed{v} \quad \to \quad y$$

  with $\boxed{\cdot}$ denoting a node in the conditioning set $\mathbf{S}$, i.e., $\mathbf{S} = \{r, v\}$. Rule 2 tells us that x and y are d-separated by $\mathbf{S}$. The only pairs of unmeasured nodes that remain d-connected in this example, conditioned on $\mathbf{S}$, are s and t and u and t

  Rule 3 (Conditioning on Colliders) If a collider is a member of the conditioning set $\mathbf{S}$, or has a descendant in $\mathbf{S}$, then it no longer blocks any path that traces this collider

  – *Example:* The following graph is given

$$\begin{array}{ccccccccccccc} x & \to & \boxed{r} & \to & s & \to & t & \leftarrow & u & \leftarrow & v & \to & y \\ & & \downarrow & & & & \downarrow & & & & \downarrow & & \\ & & r & & & & \boxed{p} & & & & q & & \end{array}$$

with $\boxed{\cdot}$ denoting a node in the conditioning set $\mathbf{S}$, i.e., $\mathbf{S} = \{r, q\}$. Rule 3 tells us that s and y are d-connected by $\mathbf{S}$, because the collider at t has a descendant in $\mathbf{S}$, which unblocks the path $s - t - u - v - y$

- In any distribution that factorizes according to a DAG: if i and j are d-separated by $\mathbf{S}$ in the DAG, then X_i and X_j are conditionally independent given $\mathbf{S}$ in the distribution
- *Example:* The following DAG is given:

$$\text{yellow teeth} \leftarrow \text{smoking} \rightarrow \text{tar in lungs} \rightarrow \text{cancer} \leftarrow \text{asbestos}. \tag{122}$$

Denote d-separation by $\perp$; then

yellow teeth $\perp$ cancer $\mid$ smoking	✓
tar $\perp$ asbestos	✓
tar $\perp$ asbestos $\mid$ cancer	×
yellow teeth $\perp$ asbestos $\mid$ cancer	×

Lecture 12 (Week 49)

Constraint-Based Structure Learning (2)

- Given all conditional independence relationships in the observational distribution, we should be able to find the DAG
- Several DAGs can encode the same conditional independence information. Such DAGs are called *Markov equivalent* and form a *Markov equivalence class*
- Markov equivalent DAGs have the same skeleton and the same v-structures
- Markov equivalence class can be described uniquely by a completed partially directed acyclic graph (CPDAG). A CPDAG has the following properties:
  1. Every directed edge exists in every DAG in the Markov equivalence class
  2. For every undirected edge $X_i - X_j$ there exists a DAG with $X_i \rightarrow X_j$ and a DAG with $X_i \leftarrow X_j$ in the Markov equivalence class
  3. A CPDAG $\mathcal{C}$ is said to represent a DAG $\mathcal{G}$ if $\mathcal{G}$ belongs to the Markov equivalence class described by $\mathcal{C}$
- Constraint-based methods require a Markov and faithfulness assumption

  Causal Markov Condition Once we know all direct causes of an event, the event is probabilistically independent of its causal non-descendants [4]

  - *Example:* Suppose we see a broken glass bottle on the bicycle path with small pieces of glass lying around. Learning the cause of this broken bottle or that a piece from the bottle hurt a passing dog, does not change our expectation of a flat tire caused by the pieces of glass on the road

Faithfulness Assumption All interdependencies observed in the data are structural, resulting from the structure of the causal graph, and not accidental, e.g., by some particular combination of parameter values that result in causal effects canceling out [4]

- Under both assumptions: there is an edge between X_i and X_j in the DAG if and only if X_i and X_j are dependent given every subset of the remaining variables
  - The skeleton of a DAG is determined uniquely by conditional independence relationships
- Assuming faithfulness, a CPDAG can be estimated by the PC-algorithm

The PC-Algorithm: Oracle Version

- The oracle version of the PC-algorithm works under the assumption that we have perfect conditional independence information between all variables [2]

1. No edge between X_i and X_j

$$\Longleftrightarrow$$

$X_i \perp X_j \mid \mathbf{S}$ for some subset $\mathbf{S}$ of the remaining variables

$$\Longleftrightarrow$$

$X_i \perp X_j \mid \mathbf{S}'$ for some subset $\mathbf{S}'$ of $\mathrm{adj}(X_i)\backslash\{X_j\}$ or of $\mathrm{adj}(X_j)\backslash\{X_i\}$

2. Start with the complete graph
3. For $k = 0, 1, \ldots$ do:
   - Consider all pairs of adjacent vertices (X_i, X_j) and remove edge if they are conditionally independent given some subset of size k of $\mathrm{adj}(X_i)\backslash\{X_j\}$ or of $\mathrm{adj}(X_j)\backslash\{X_i\}$

   Until $k > \max(|\mathrm{adj}(X_i)\backslash\{X_j\}|, |\mathrm{adj}(X_j)\backslash\{X_i\}|)$

The PC Algorithm: Sample Version (8)

- In the sample version of the PC-algorithm, the conditional independence relationships have to be estimated from the data [2]
- In the multivariate Gaussian setting, this is equivalent to testing for zero partial correlation, i.e.,

$$\boxed{H_0 : \rho_{i,j|\mathbf{S}} = 0 \qquad H_a : \rho_{ij|\mathbf{S}} \neq 0} \tag{123}$$

where $\rho_{i,j|\mathbf{S}}$ denotes the partial correlation between $\mathbf{X}_i$ and $\mathbf{X}_j$ given $\mathbf{S}$ and can be computed via regression, inversion of parts of the covariance matrix, or a recursive formula

  - Assuming that the distribution f of the random vector $\mathbf{X}$ is mutlvariate normal. Then, $\rho_{i,j|\mathbf{S}} = 0$ if and only if $\mathbf{X}_i$ and $\mathbf{X}_j$ are conditionally independent given $\mathbf{S}$

- Apply Fisher's Z-transform for testing whether a partial correlation is zero or not:

$$\boxed{Z(i,j|\mathbf{S}) = \frac{1}{2}\log\left(\frac{1+\hat{\rho}_{i,j|\mathbf{S}}}{1-\hat{\rho}_{i,j|\mathbf{S}}}\right)} \tag{124}$$

- Reject the null-hypothesis $H_0 : \rho_{i,j|\mathbf{S}} = 0$ against the two-sided alternative $H_a : \rho_{ij|\mathbf{S}} \neq 0$ if

$$\boxed{\sqrt{n-|\mathbf{S}|-3}\; Z(i,j|\mathbf{S}) > \Phi^{-1}(1-\alpha/2)} \tag{125}$$

where $\Phi(\cdot)$ denotes the cumulative distribution function of $\mathcal{N}(0,1)$ and α is the significance level and serves as a tuning parameter

Consistency for High-Dimensional Data (8)

- Most of the time, data sets contain many more variables than observations, i.e., $p \gg n$
- Consider a framework in which the graph is allowed to grow with the sample size n
  - DAG: G_n
  - CPDAG: C_n
  - Number of variables: p_n
  - Variables: $\mathbf{X}_{n1}, \ldots, \mathbf{X}_{n\,p_n}$
  - Distribution: P_n
- The following assumptions are made:
  1. The dimension $p_n = O(n^a)$, for some $0 \leq a < \infty$
  2. The maximal number of neighbours in the DAG G_n is $q_n = O(n^{1-b})$ for some $0 < b \leq 1$
  3. The absolute values of the partial correlations $\rho_{i,j|\mathbf{S}}$ are bounded from below and above:

$$\boxed{\begin{aligned} &\inf\{|\rho_{i,j|\mathbf{S}}| : i,j,\mathbf{S} \text{ with } \rho_{i,j|\mathbf{S}} \neq 0\} \geq c_n \quad , && (126)\\ &\sup_{n;i,j,\mathbf{S}} |\rho_{i,j|\mathbf{S}}| \leq M < 1 && (127)\end{aligned}}$$

  with $c_n^{-1} = O(n^d)$ for some $0 < d < b/2$

- Denote the estimated CPDAG by $\hat{C}_n(\alpha_n)$ and the true CPDAG by C_n from the DAG G_n. Then, there exists a sequence $\alpha_n \to 0$ such that

$$\boxed{\mathbb{P}[\hat{C}_n(\alpha_n) = C_n] = 1 - O(\exp\{-Kn^{1-2d}\})} \tag{128}$$

  for some $0 < K < \infty$ and $0 < d < b/2$

References

[1] G. Biau and L. Devroye, On the layered nearest neighbour estimate, the bagged nearest neighbour estimate and the random forest method in regression and classification, Journal of Multivariate Analysis, 101 (2010), pp. 2499–2518.

[2] D. Colombo and M. H. Maathuis, Order-independent constraint-based causal structure learning, arXiv preprint arXiv:1211.3295, (2012).

[3] A. P. Dempster, N. M. Laird, and D. B. Rubin, Maximum likelihood from incomplete data via the em algorithm, Journal of the royal statistical society. Series B (methodological), (1977), pp. 1–38.

[4] M. J. Druzdzel, The role of assumptions in causal discovery, UNCERTAINTY PROCESSING, (2009), p. 57.

[5] T. Hastie, R. Tibshirani, J. Friedman, and J. Franklin, The elements of statistical learning: data mining, inference and prediction, The Mathematical Intelligencer, 27 (2005), pp. 83–85.

[6] T. Hastie, R. Tibshirani, G. Sherlock, M. Eisen, P. Brown, and D. Botstein, Imputing missing data for gene expression arrays, 1999.

[7] G. James, D. Witten, T. Hastie, and R. Tibshirani, An introduction to statistical learning, Springer, 2013.

[8] M. Kalisch and P. Bühlmann, Estimating high-dimensional directed acyclic graphs with the pc-algorithm, The Journal of Machine Learning Research, 8 (2007), pp. 613–636.

[9] R. J. Little and D. B. Rubin, Statistical analysis with missing data, John Wiley & Sons, 2014.

[10] M. H. Maathuis and M. Mächler, 401-3611-00l advanced topics in computational statistics. http://www.vvz.ethz.ch/Vorlesungsverzeichnis/lerneinheitPre.do?lerneinheitId=99921&semkez=2015W&lang=de.

[11] R. Mazumder, T. Hastie, and R. Tibshirani, Spectral regularization algorithms for learning large incomplete matrices, The Journal of Machine Learning Research, 11 (2010), pp. 2287–2322.

[12] J. Pearl, Causality, Cambridge university press, 2009.

[13] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. B. Altman, Missing value estimation methods for dna microarrays, Bioinformatics, 17 (2001), pp. 520–525.

[14] G. TUTZ AND S. RAMZAN, Improved methods for the imputation of missing data by nearest neighbor methods, Computational Statistics & Data Analysis, 90 (2015), pp. 84–99.

[15] S. VAN BUUREN, Flexible imputation of missing data, CRC press, 2012.