1

Introduction to Statistical Methods for Complex Systems

Tristan Mary-Huard[1,2] and Stéphane Robin[1,2]
([1]) AgroParisTech, UMR 518, 16, rue Claude Bernard, F-75005, Paris, FRANCE
([2]) INRA, UMR 518, 16, rue Claude Bernard, F-75005, Paris, FRANCE

This is a Book Title Name of the Author/Editor
© XXXX John Wiley & Sons, Ltd

CONTENTS

1.1 Introduction

The aim of the present chapter is to introduce and illustrate some concepts of statistical inference useful in systems biology. Here we limit ourselves to the classical, so-called ’frequentist’ statistical inference where parameters are fixed quantities that need to be estimated. The Bayesian approach will be presented in chapter A3.

Modelling and inference techniques are illustrated on three recurrent problems in system biology:

Class comparison aims at assessing the effect of some treatment or experimental condition on some biological response. This requires a proper statistical modelling to account for the experimental design, various covariates or stratifications or dependence between the measurements. As systems biology often deals with high-throughput technologies, it also raises multiple testing issues.

Class prediction refers to learning techniques that aim at building a rule to predict the status (e.g. well or ill) of an individual, based on a set of biological descriptors. An exhaustive list of classification algorithms is out of reach, but general techniques such as regularisation or aggregation are of prime interest in systems biology where the number of variables often exceeds the number of observations from far. Evaluating the performances of a classifier also requires relevant tools.

Class discovery aims at uncovering some structure in a set of observations. These techniques include distance-based or model-based clustering methods , and allow to determine distinct groups of individuals, in absence of a prior classification. However, the underlying structure may have more complex forms, each raising specific issues in terms of inference.

This chapter focuses on generic statistical concepts and methods, that can be applied no matter which technology is used for the data acquisition. In practice, applications to any biological problem will necessitate both a relevant strategy for the data collection, and a careful tuning of the methods to obtain meaningful results. These two steps of data collection (or experimental design conception) and adaptation of the generic methods require to take into account the nature of the data. Therefore, they are dependent on the data acquisition technology, and will be discussed in Part B.
In this chapter, the data are assumed to arise from a static process. The analysis of a dynamic biological system would require more sophisticated methods, such as partial differential equations or network modelling. These topics are not discussed here as they will be reviewed in depth in Parts C and D.

Lastly, a basic knowledge in statistics is assumed, covering topics including point estimation (in particular maximum likelihood estimation), hypothesis testing, and a background in regression and linear models.

1.2 Class comparison

We consider here the general problem of assessing the effect of some treatment, experimental condition or covariate on some response. We first address the problem of modelling the data resulting from the experiments, focusing on how to account for the dependency between the observations. We then turn to the problem of multiple testing, which is recurrent in high-throughput data analyses.

1.2.1 *Models for dependent data*

Many biological experiments aim at observing the effects of a given treatment (or combination of treatments) on a given response. 'Treatment' is used here in a very broad sense, including controlled experimental conditions, uncontrolled covariates, time, population structure, etc. In the following n will stand for the total number of experiments.

Linear (Gaussian) models (Searle (1971) or Dobson (1990)) provide a general framework to describe the influence of a set of controlled conditions and/or uncontrolled covariates, summarised in a $n \times p$-dimensional matrix $\mathbf{X}$, on the observed response gathered in a n-dimensional vector $\mathbf{Y}$ as

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\theta} + \mathbf{E} \tag{1.1}$$

where $\boldsymbol{\theta}$ is the p-dimensional vector containing all parameters. In the most classical setting, the response is supposed to be Gaussian, and the dependency structure between the observations is then fully specified by the (co-)variance matrix $\boldsymbol{\Sigma} = \mathbb{V}(\mathbf{Y}) = \mathbb{V}(\mathbf{E})$ which contains the variance of each observation on the diagonal, and the covariances between pairs of observations elsewhere. In the most simple setting, the responses are supposed to be independent with same variance σ^2, that is $\boldsymbol{\Sigma} = \sigma^2\mathbf{I}$.

Writing the right (mixed) model

In more complex experiments, the assumption that observations are independent does not hold and the structure of $\boldsymbol{\Sigma}$ needs to be adapted. Because it contains $n(n+1)/2$ parameters, the shape of $\boldsymbol{\Sigma}$ has to be strongly constrained to allow good inference. We first present here some typical experimental settings, and the associated dependency structures.

Variance components.
Consider the study of the combined effects of the genotype (indexed by i) and of the cell type (k) on some gene expression. Several individuals (j) from each genotype are included and cells from each type are harvested in each of them. In such a setting the expected response is $\mathbb{E}(Y_{ijk}) = \mu_{ik}$, which is often decomposed into a genotype effect, a cell type effect and an interaction as $\mu_{ik} = \mu + \alpha_i + \beta_k + (\alpha\beta)_{ik}$. The most popular way to account for the dependency between measures obtained on a same individual is to add a random term U_{ij} associated with each individual. The complete model can then be written has

$$Y_{ijk} = \mu_{ik} + U_{ij} + E_{ijk} \tag{1.2}$$

where all U_{ij} and E_{ijk} are independent centred Gaussian variables with variance σ_U^2 and σ_E^2, respectively. The variance of one observation is then $\sigma^2 = \sigma_U^2 + \sigma_E^2$, where σ_U^2 is the 'biological' variance and σ_E^2 is the 'technical' one (Kerr and Churchill (2001)). The random effect induces a uniform correlation between observations from the same individual since:

$$\mathbb{C}\text{ov}(Y_{ijk}, Y_{i'j'k'}) = \sigma_U^2 \quad \text{if} \quad (ij) = (i'j'), \quad k \neq k' \tag{1.3}$$

and 0 if $(ij) \neq (i'j')$. The matrix form of this model is a generalisation of (1.1):

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\theta} + \mathbf{Z}\mathbf{U} + \mathbf{E} \tag{1.4}$$

where $\mathbf{Z}$ describes the individual structure: each row corresponds to one measurement and each column to one individual and $\mathbf{Z}$ contains a 1 at the intersection if the measurement has been made on the individual, and a 0 otherwise. The denomination 'mixed' of 'linear mixed models' comes from the simultaneous presence of fixed ($\boldsymbol{\theta}$) and random ($\mathbf{U}$) effects. It corresponds to the simplest form of so-called 'variance components' models. The variance matrix corresponding to (1.3) is $\boldsymbol{\Sigma} = \sigma_U^2\mathbf{Z}\mathbf{Z}' + \sigma_E^2\mathbf{I}$. Application of such model to gene expression data can be found in Wolfinger et al. (2001) or Tempelman (2008).

Repeated measurements.
One considers a similar design where, in place of cell types, we compare successive harvesting times (indexed by t) within each individual. The uniform correlation within each individual given in (1.3) may then seem inappropriate, for it does not account for the delay between times of observation. A common dependency form is then the so-called 'autoregressive', which states that

$$\mathbb{C}\text{ov}(Y_{ijt}, Y_{i'j't'}) = \sigma^2\rho^{-|t-t'|} \quad \text{if} \quad (ij) = (i'j')$$

and 0 otherwise. This is to assume that the correlation decreases (at an exponential rate) with the time delay. Such a variance structure can not be put in a simple matrix form similar to (1.4). Note that Equation (1.1) is still valid, but with non-diagonal variance matrix $\boldsymbol{\Sigma} = \mathbb{V}(\mathbf{E})$.

Spatial dependency.
It is also desirable to account for spatial dependency when observations have some spatial localisation. Suppose one wants to compare treatments (indexed by i), and that replicates (j) have respective localisations ℓ_{ij}. A typical variance structure (Cressie (1993)) is

$$\mathbb{V}(Y_{ij}) = \sigma^2 = \sigma_S^2 + \sigma_E^2,$$
$$\mathbb{C}\text{ov}(Y_{ij}, Y_{i'j'}) = \sigma_S^2\rho^{-\|\ell_{ij}-\ell_{i'j'}\|}$$

where σ_E^2 accounts for the measurement error variability and ρ controls the speed at which the dependency decreases with distance.

The dependency structures described above can of course be combined. Also note that their list is far from exhaustive. The limitations often come from the software at hand or the specific computing developments that can be made. A large catalogue of such structures can be found in softwares like SAS (2002-03) or R (`www.r-project.org`).

Inference

Some problems related to the inference of mixed linear models are still unresolved. We only provide here an introduction to the most popular approaches and emphasise some practical issues that can be faced when using them.

Estimation.
Mixed model inference requires to estimate both $\boldsymbol{\theta}$ and $\mathbf{\Sigma}$. We start with the estimation of $\mathbf{\Sigma}$, which reduces to the estimation of few variance parameters such as $\sigma^2, \sigma_U^2, \sigma_E^2, \sigma_S^2, \rho$ in the examples given above.
Moment estimates can be obtained (see Searle (1971) or Demindenko (2004)), typically for variance components models. Such estimates are often based on sums of squares, that are squared distances between $\mathbf{Y}$ and its projection on various linear spaces, such as span$(\mathbf{X})$, span$(\mathbf{Z})$ or span$(\mathbf{X}, \mathbf{Z})$. The expectation of these sums of squares can hopefully be related to the different variances parameters and the estimation then reduces to solving a set of linear equations.
The maximum likelihood (ML) estimator is defined as

$$\widehat{\mathbf{\Sigma}}_{\text{ML}} = \arg\max_{\mathbf{\Sigma}} \log P(\mathbf{Y}; \boldsymbol{\theta}, \mathbf{\Sigma})$$

and can be used for all models. Unfortunately, ML variance estimates are known to be biased in many (almost all) situations, because both $\boldsymbol{\theta}$ and $\mathbf{\Sigma}$ have to estimated at the same time. The most popular way to circumvent this problem consists in changing to a model where $\boldsymbol{\theta}$ is known (Verbeke and Molenberghs (2000)). Defining some matrix $\mathbf{T}$ such that $\mathbf{TX} = 0$, we may define the Gaussian vector $\widetilde{\mathbf{Y}} = \mathbf{TY}$ which satisfies

$$\mathbb{E}(\widetilde{\mathbf{Y}}) = \mathbf{0} \qquad \text{and} \qquad \mathbb{V}(\widetilde{\mathbf{Y}}) = \mathbf{T\Sigma T}'.$$

The most natural choice for $\mathbf{T}$ is the projector on the linear space orthogonal to span$(\mathbf{X})$. The so-called 'restricted' maximum likelihood ('ReML') of $\mathbf{\Sigma}$ is then defined as

$$\widehat{\mathbf{\Sigma}}_{\text{ReML}} = \arg\max_{\mathbf{\Sigma}} \log P(\widetilde{\mathbf{Y}}; \mathbf{\Sigma}).$$

Note that ReML estimates are maximum likelihood estimates and therefore inherit all their properties (consistency, asymptotic normality, etc.). Also note that ReML provides less biased variance estimates than ML. The ReML estimates can be shown to be unbiased only for some specific cases such as orthogonal designs where they are equivalent to moment estimates.

The estimates of $\boldsymbol{\theta}$ (and the variance of this estimator) are the same for maximum likelihood and generalised least squares (Searle (1971)):

$$\widehat{\boldsymbol{\theta}} = (\mathbf{X}'\widehat{\boldsymbol{\Sigma}}^{-1}\mathbf{X})^{-1}\mathbf{X}'\widehat{\boldsymbol{\Sigma}}^{-1}\mathbf{Y}, \qquad \widehat{\mathbb{V}}(\widehat{\boldsymbol{\theta}}) = (\mathbf{X}'\widehat{\boldsymbol{\Sigma}}^{-1}\mathbf{X})^{-1}.$$

These estimates are often calculated with a simple plug-in of one of the estimates $\widehat{\boldsymbol{\Sigma}}$ described above. However, as it relies on the shape of $\boldsymbol{\Sigma}$ specified in the model, such an estimator of $\mathbb{V}(\widehat{\boldsymbol{\theta}})$ is not robust to a misspecification of the dependency structure. Alternative estimators such as the 'sandwich' estimator can be defined, that are both consistent and robust (Diggle et al. (2002)).

Tests.
As many experiments aim to assess the significance of a given effect, we are often interested in testing the hypothesis H_0 stating that the corresponding contrast between the elements of $\boldsymbol{\theta}$ is null. The global procedure is similar to that of model (1.1): writing the contrast under study as a linear combination of the parameters $\mathbf{c}'\boldsymbol{\theta}$, we get the usual test statistic:

$$T = \mathbf{c}'\widehat{\boldsymbol{\theta}} \Big/ \sqrt{\mathbf{c}'\widehat{\mathbb{V}}(\widehat{\boldsymbol{\theta}})\mathbf{c}} \underset{H_0}{\approx} \mathcal{T}_\nu. \tag{1.5}$$

A central practical issue is the determination of the (approximate) degree of freedom of the Student's t distribution as it depends on both the experimental design and the contrast itself.

As an example, in model (1.2), we first consider the average difference between genotype i and i': $\mu_{i\bullet} - \mu_{i'\bullet}$ that is estimated by $Y_{i\bullet\bullet} - Y_{i'\bullet\bullet}$ (the notation '$\bullet$' means that the variables are averaged over the index it replaces (Searle (1971))). We then consider the difference between two cell types k and k' is $\mu_{\bullet k} - \mu_{\bullet k'}$ and its estimate $Y_{\bullet\bullet k} - Y_{\bullet\bullet k'}$. If the design is completely balanced with I genotypes, J individuals per genotypes and K cell types within each individual, the respective variances of these estimates are

$$\mathbb{V}(\widehat{\mu}_{i\bullet} - \widehat{\mu}_{i'\bullet}) = (2\sigma_U^2/J) + (2\sigma_E^2/JK),$$
$$\mathbb{V}(\widehat{\mu}_{\bullet i} - \widehat{\mu}_{\bullet k'}) = 2\sigma_E^2/IJ.$$

We see here that the biological variance σ_U^2 does not affect the contrast on the cell types (that are harvested *within* each individual) whereas it is predominant for the genotype contrast, which refers to differences *between* the individuals. The distribution of the test statistics is Student in both case, but with different degrees of freedom: $IJ - I$ for the genotype and $IJK - IK$ for the cell types. The intuition is that the number of observations is the number of individuals IJ for the genotype, whereas it is the total number of measures IJK for the cell type. Hence, in such a design the power will be (much) greater for distinguishing cell types than genotypes.

For more complex dependency structures such as repeated measurements or spatial data, the distribution of (1.5) is only approximated and the degrees of freedom ν need to be approximated (see Fai and Cornelius (1996), Kenward and Roger (1997) or Khuri et al. (1998)).

Generalised Linear Mixed Models

Observations can not always be modelled with a Gaussian distribution. The linear model can however easily be generalised to most usual distributions such as binomial, Poisson, Gamma, etc., giving raise to the 'generalised' linear model (Dobson (1990), Demindenko (2004), Zuur et al. (2009)). A key ingredient is the introduction of a 'link function' g between the expectation of the responses and the linear model as $g(\mathbb{E}\mathbf{Y}) = \mathbf{X}\boldsymbol{\theta}$.
For non Gaussian models, a canonical parametrisation of the correlation between the observations does not always exist. A specific modelling of the dependency is then required. As an example, in the Poisson case, the variance components model (1.4) will typically be rewritten as

$$g[\mathbb{E}(\mathbf{Y}|\mathbf{U})] = \mathbf{X}\boldsymbol{\theta} + \mathbf{Z}\mathbf{U}$$

where g is the $\log$ function and the coordinates of $\mathbf{U}$ are independent Gamma distributed variables. However the Gaussian modelling can be re-used in all generalised linear models, stating that

$$g[\mathbb{E}(\mathbf{Y}|\mathbf{V})] = \mathbf{X}\boldsymbol{\theta} + \mathbf{V},$$

where $\mathbf{V}$ is a centred Gaussian vector with variance matrix $\boldsymbol{\Sigma}$ similar to those seen above in the Gaussian context.

1.2.2 *Multiple testing*

Models such as those described in the preceeding section allow to assess or reject the existence of some effect on a given response. They typically allow to assess whether a given gene is differentially expressed between several conditions in microarray experiments, accounting for possible covariates and dependency between the replicates. Because of the large number of genes spotted on each microarray, we are then faced with a multiple testing since we get one test statistic T_i similar to (1.5) for each of the genes. This example has become a common place (see Dudoit et al. (2003)), but similar issues are encountered in SNP studies, QTL detection, mass-spectrometry, that is in all statistical analyses dealing with high throughput technologies.
Multiple testing issues can be addressed in a Bayesian setting, as described in Chapter XXX, Section 3.6.

The basic problem

One test setting.
We first consider hypothesis H_i that states that the expression level of gene i is not affected by the treatment under study. To assess H_i, a test statistic T_i is calculated from the observations such that a large value of T_i will lead to reject H_i (gene i is

then said to be ’differentially expressed’). More precisely, denoting Φ the cumulative distribution function (cdf) of T_i if H_i holds, we calculate a p-value defined as

$$P_i = 1 - \Phi(T_i)$$

and reject H_i if P_i is below a pre-specified level t. P_i measures the significance of the test and t is the level of the test, that is the risk (probability) to reject H_i whereas it is true. When testing one hypothesis at a time, t is generally set to 1% or 5%.

Multiple testing.
Now consider m null hypotheses $H_1, \dots H_m$. Suppose that among these m hypotheses, m_0 actually hold. For a given level t, denote $FP(t)$ the number of false positives, that is the number of tests i for which H_i actually holds, whereas P_i falls below t. The expected number of false positives is

$$\mathbb{E}[FP(t)] = m_0 t.$$

This number can exceed several hundreds when m_0 reaches 10^4 or 10^6, as in microarray or SNP analyses. The primary goal of a multiple testing procedure (MTP) is to keep this number of false positives small, by tuning the threshold t.

Distribution of the p-value.
Most MTP apply to the set of p-values $P_1, \dots P_m$ and the control of $FP(t)$ relies essentially on their distribution. If H_i is true, we have

$$T_i \sim \Phi \qquad \Rightarrow \qquad P_i \sim \mathcal{U}_{[0;1]},$$

which means that, when H_i holds, P_i is uniformly distributed over the interval $[0; 1]$. As we shall see, this property is one of the key ingredients of all MTP. It is therefore highly recommended to make a graphical check of it by simply plotting the histogram of the P_is: it should show a peak close to 0 (corresponding to truly differentially expressed genes associated with small P_is) and a uniform repartition along the rest of the interval $[0, 1]$ (see Figure 1.1, left).

Global risk

Because $FP(t)$ is random, an MTP aims at controlling some characteristic of its distribution. Such a characteristic can be view as a global risk since it considers all the tests at the same time. Our goal is to maintain it at some pre-specified value α. We assume that m_0 is known; if not, it can be either estimated or replaced by a surrogate (see Paragraph ‘Estimation of π_0’).

Family-wise Error Rate (FWER).
The most drastic approach is to keep $FP(t)$ close to 0, that is to make

$$FWER = \Pr\{FP > 0\}$$

small. The goal is then to find the right level t to guarantee a targeted FWER α. When all tests are assumed to be independent, $FP(t)$ has a binomial $\mathcal{B}(m_0, t)$ distribution so

$$FWER = 1 - (1-t)^{m_0} = \alpha \qquad \Leftrightarrow \qquad t = 1 - (1-\alpha)^{1/m_0}.$$

This MTP is known as the Sidak procedure. When independence is not assumed, $FWER$ can be upper bounded via the Bonferroni inequality which leads to

$$FWER \leq m_0 t = \alpha \qquad \Leftrightarrow \qquad t = \alpha/m_0.$$

For a small α, we have

$$1 - (1-\alpha)^{1/m_0} \approx 1 - (1 - \alpha/m_0) = \alpha/m_0,$$

so the two MTP lead to similar thresholds t.

False Discovery Rate (FDR).
Because it basically aims at maintaining FP at 0, FWER control often leads to too stringent thresholds, especially when m_0 is large. Benjamini and Hochberg (1995) suggest to rather control the False Discovery Rate (FDR), that is the expected proportion of false positives among all positives:

$$FDR(t) = \mathbb{E}[FP(t)/P(t)]$$

where $P(t)$ denotes the number of P_is smaller than t (the precise definition of the FDR is slightly different to account for the case where $P(t) = 0$.). These authors propose the following MTP: denoting $P_{(1)} \leq \cdots \leq P_{(i)} \leq \cdots \leq P_{(n)}$ the ordered p-values, define

$$i^* = \max\{i : m_0 P_{(i)}/i \leq \alpha\}$$

and reject all hypotheses H_i such that $P_i \leq P_{(i^*)}$, then $\mathbb{E}[FDR(P_{(i^*)})] \leq \alpha$. The intuition behind this is that if t is chosen as $t = P_{(i)}$ then $P(t) = i$ and $\mathbb{E}[FP(t)] = m_0 P_{(i)}$, so $\mathbb{E}[FP(t)/P(t)|P_{(i)}] = m_0 P_{(i)}/i$.

Some extensions

Estimation of π_0**.**
Most MTP involve the number of true hypotheses m_0, which is unknown, in general. A conservative strategy consists in replacing m_0 with m, but this leads to a (sometimes dramatic) loss of power. A reliable estimation of m_0, or equivalently $\pi_0 = m_0/m$, strongly improves the MTP. Thanks to the large number of tested hypotheses, the proportion π_0 of true ones can be accurately estimated and several estimates have been proposed (see Langaas et al. (2005), Blanchard and Roquain (2009) or Celisse and Robin (2010) and references therein). The basic idea traces back to Storey (2002) who suggests to consider the number $N(\lambda)$ of p-values

exceeding a given threshold λ, above which only true null hypotheses should be found. π_0 can then be estimated by

$$\widehat{\pi}_0 = N(\lambda)/(1-\lambda).$$

$\widehat{m}_0$ can then be plugged into the MTP. The properties of the resulting procedure can be assessed via a careful study of the behaviour of the empirical cdf of the p-value and associated False Positive and Negative empirical processes (Genovese and Wasserman (2002), Genovese and Wasserman (2004)). The choice of λ can also be made in an adaptive way (Celisse and Robin (2010)).

Dependency.
The independence hypothesis stated up to now does not hold in most real situations: gene expressions are typically correlated and the test statistics associated to each gene are therefore not independent. A strong departure from this hypothesis may have strong consequences (Kim and van de Wiel (2008)). When permutation tests are used, the dependency structure can be embedded in the permutation algorithm so no specific adjustment is required (Westfall and Young (1993)). In a more general context, the MTP can be adapted to still control FDR under dependency. This field is active and we only provide here few references: Sun and Cai (2008) consider a Markovian dependency, Friguet et al. (2009) propose a general approach to model the dependency and Blanchard and Roquain (2008) consider a more general context and define adaptive MTP's.

Local FDR.
FDR only informs us about the proportion of false positive within a set of rejected tests. Some information about the probability for each of them to be a false positive is clearly desirable in practice. This is the aim of the local FDR (ℓFDR) defined by Efron et al. (2001) who rephrase the multiple testing problem as a mixture model (see Section 1.4). By construction, p-values associated with true null hypotheses have a uniform distribution over $[0, 1]$, so the distribution of all p-values displayed in Figure 1.1 (left) can be written as

$$g(p) = \pi_0 + (1-\pi_0)f(p)$$

where f is some density concentrated toward 0. This modelling is consistent with most estimates of π_0 presented above. The local FDR of hypothesis H_i is then defined as in (1.19):

$$\ell\text{FDR}_i = \pi_0 \, / g(P_i) \ .$$

Various estimates of f and π_0 have been proposed in this context. Efron et al. (2001) considered a empirical Bayes approach, Allison et al. (2002) proposed a parametric mixture of Beta distribution, while McLachlan et al. (2006) applied a Gaussian mixture to probit-tranformed p-values. This last transformation turns out to be very efficient in practice, as it zooms into the region where true and false hypotheses are mixed, making the mixture more identifiable. More flexible modellings have also

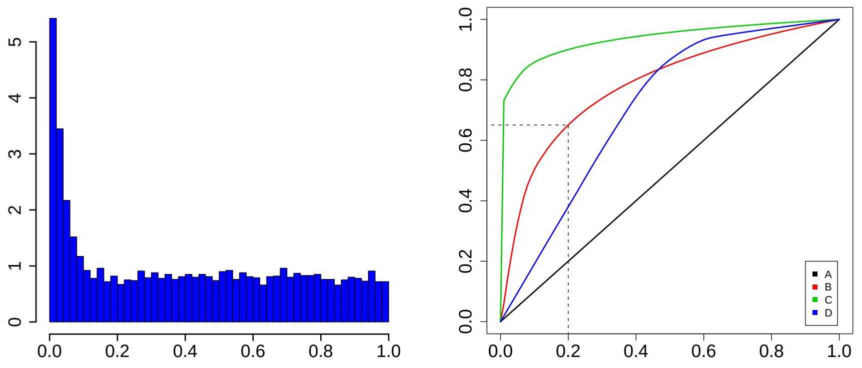

Figure 1.1 **Left:** Typical distribution of the p-values in a multiple setting. **Right:** ROC curves for 4 different classifiers A, B, C and D. Sensitivity is given in ordinates, and $1-$Specificity in abscisses.

been proposed, including a semi-parametric estimate of f (Robin et al. (2007)), or estimation under constraint of convexity (Strimmer (2008)). R packages are associated with most of these procedures, the ’Mutoss’ R-project aim at gathering them (`r-forge.r-project.org/projects/mutoss/`).
Again, local FDR has a natural sense in a Bayesian setting in terms of posterior probability, as described in Chapter XXX, Section 3.6.

1.3 Class Prediction

1.3.1 *Building a classifier*

Classification problem

In classification, the goal is to predict the class or label Y of an observation, according to some information X collected on this observation. Here are two examples of classification problems in computational biology:

- Cancer diagnosis: Based on information X that consists in gene activity measured via microarray technology on a given tissue sample collected from an individual, one wants to determine the disease status Y of these individual that can be 0 if the individual is healthy, 1 if a moderate form of cancer is diagnosed or 2 if the cancer is invasive (Golub et al. (1999)).
- Protein-protein interaction: the problem is to predict whether a protein interacts (i.e. binds) with a protein of interest. In this problem, each candidate protein can be described by its amino-acid sequence and/or its surfacic

geometrical elements that constitute the available information X, and the label Y is $+1$ if the candidate protein interacts, -1 otherwise (Hue et al. (2010)).

In these examples information X is of various forms, while Y is a qualitative variable with usually few possibles values. In the following we only consider binary classification problems where $Y = -1$ or $+1$.

In order to predict Y from X, one has to build a *classifier*, i.e. a function $f : \mathcal{X} \rightarrow \{-1, +1\}$, whose predictions are as accurate as possible. The prediction performance of a classifier can be quantified by its classification error rate

$$L(f) = \Pr\{f(X) \neq Y\} \tag{1.6}$$

Theoretically, the problem is simple: define the Bayes classifier as

$$f_B(x) = \begin{cases} +1 & \text{if } p_{+1}(x) > p_{-1}(x) \\ -1 & \text{otherwise} \ , \end{cases} = \begin{cases} +1 & \text{if } p_{+1}(x) > 0.5 \\ -1 & \text{otherwise} \ , \end{cases} \tag{1.7}$$

where $p_\ell(x) = \Pr\{Y = \ell | X = x\}$ is the posterior probability for point x to belong to class ℓ (to have label ℓ). This classifier is known to be optimal, in the sense that it minimises the classification error rate. However in practice this classifier is not available, because the joint distribution $P(X, Y)$ is usually unknown. In fact, the only information at our disposal to built a classifier is a training set, i.e. a sample of n independent observations $(X_1, Y_1), ..., (X_n, Y_n)$ for which both informations and labels are known. Still, the Bayes classifier gives important clues about how to build a classifier: the decision to classify an observation as $+1$ is based on the posterior probabilities $p_{+1}(x)$ and $p_{-1}(x)$. One then needs to either estimate this probabilities, or at least to identify the set of points x such that $p_{+1}(x) > p_{-1}(x)$.

Many *classification algorithms* have been proposed to build a classifier, 3 of the most popular ones are presented in the next paragraph.

Some classification algorithms

For sake of simplicity, we will assume here that X is a collection of p quantitative descriptors $X^1, ..., X^p$, such as the expression measurements of genes in a given tissue.

Logistic Regression.
The logistic regression algorithm is based on the following hypothesis: for any point x, we have:

$$\log\left(\frac{p_{+1}(x)}{p_{-1}(x)}\right) = \alpha_0 + \sum_{j=1}^{p} \alpha_j x^j = g(x) \ . \tag{1.8}$$

We can infer the unknown parameters $\alpha_j, \ j = 0, ..., p$ via maximum likelihood estimation, and then estimate the ratio of posterior probabilities by

$$\widehat{R} = \widehat{\left(\frac{p_{+1}(x)}{p_{-1}(x)}\right)} = \exp\left(\widehat{\alpha}_0 + \sum_{j=1}^{p} \widehat{\alpha}_j x^j\right) \ .$$

The logistic regression classifier is then

$$f_{LReg}(x) = \begin{cases} +1 & \text{if } \widehat{R} > 1 \\ -1 & \text{otherwise} \end{cases}.$$

A consequence of hypothesis (1.8) is that the boundary between the two classes, i.e. the points such that $p_{-1}(x) = p_{+1}(x)$, is linear. Therefore the logistic regression classifier results in a hyperplane that splits space $\mathcal{X}$ in two. Note that the model can be extended to quadratic or higher polynomial decompositions. One difficulty is then to make a relevant choice of the polynomial order. A low order may result in a too simplistic regression function and a poor fit to the training data. A high order will result in a very good fit to the traning data, but will poorly generalize to new observations. This last phenomenon is often referred to as 'over-fitting'.

Nearest Neighbour Algorithm.
The posterior probabilities $p_{-1}(x_0)$ and $p_{+1}(x_0)$ of any new observation x_0 can be locally estimated by looking at points that are close to x_0 in the training set. This is the principle of the kNN algorithm (Fix and Hodges (1951, 1952)): first, for a given observation x_0 to classify, find $X_{(1)}, ...X_{(k)}$ the k points closest to x_0 in the training set, then classify x_0 according to a majority vote decision rule among these k neighbours. We have:

$$f_{kNN}(x) = \begin{cases} +1 & \text{if } \widehat{p}_{+1}(x_0) > \widehat{p}_{-1}(x_0) \\ -1 & \text{otherwise} \end{cases}.$$

$$\text{where } \widehat{p}_\ell(x_0) = \frac{1}{k}\sum_{i=1}^{k} \mathbb{I}\{Y_{(i)} = \ell\}$$

A variant of this simple algorithm is the Weighted kNN algorithm (WkNN), where each neighbour has a specific weight $w_{(i)}^{x_0}$ in the majority vote decision rule, according to its proximity to x_0 (the closer to x_0, the higher the weight). When elaborating a kNN classifier, the difficulty lies in the choice of the distance measure between points and the number of neighbours k to consider.

Classification Tree.
A classification tree is a classifier that predicts the label of an observation by verifying a sequence of conditions, of the form "is measurement X^j higher than threshold s_j for this observation ?". Since the prediction is the result of the sequential checking process, such classifiers can be graphically represented as binary trees, where each node corresponds to a condition, except for the terminal nodes that correspond to the predictions. Assuming that the size of the tree (its number of non-terminal nodes) is K, then there are $K+1$ profiles of answers leading to one of the $K+1$ terminal nodes $N_d^1, ..., N_d^{K+1}$. Assuming without loss of generality that nodes $N_d^1, ..., N_d^k$ and $N_d^{k+1}, ..., N_d^{K+1}$ are associated to labels -1 and $+1$,

respectively, we have:

$$f_{tree}(x) = \begin{cases} +1 & \text{if } x_0 \in N_d^\ell,\ \ell \in \{1, ..., k\} \\ -1 & \text{otherwise} \end{cases} .$$

When elaborating a tree classifier, the difficulty lies in the choice of the number K of conditions to check, their ordering, the variable X^j and threshold s_j that should be used at each condition. Many algorithms have been proposed to build tree classifiers, CART (Breiman et al. (1984)) and C4.5 (Quinlan (1993)) being the most popular ones. Compared with the kNN algorithm, a major specificity of tree classifiers lies in their embedded variable selection process: at most K of the p variables will be used for the classification of any observation. This may be of importance when the number of variables is large, with most of them being irrelevant for the prediction purpose and only a small subset being informative.

These examples illustrate the great diversity of existing classification algorithms, many more being detailed in Hastie et al. (2001). One may wonder how these algorithms would perform in high dimensional settings where the number of descriptors p in X is much larger than the number of observations (often referred to as the '$p \gg n$' paradigm), a classical situation in computational biology. It appears that all will deteriorate in performance as the dimension increases, but for different reasons. Due to their internal variable selection process, classification trees may be unstable: small changes in the training set will yield different classifiers with different sets of selected variables, and large discrepancies in prediction. Nearest neighbours will be much more stable since no variable selection is performed, but the pertinent information may be diluted if most variables are uninformative, leading to poor prediction performance. Lastly, logistic regression may be inefficient too, since it requires to estimate one parameter per variable, each estimation being vitiated by noise.

We therefore need efficient strategies to deal with high dimensional data. Two of them, aggregation and regularisation, are presented in sections 1.3.2 and 1.3.3.

1.3.2 Aggregation

Aggregation may be motivated by different purposes: to ease the interpretation of the classification rule, or to improve the predictions of a given classification algorithm. When interpretation is at stake, aggregation is usually performed at the variable level. If the goal is to improve the prediction performance, one will rather aggregate classifiers. We briefly review variable aggregation and then focus on the two classical strategies to aggregate classifiers, bagging and boosting.

Variable aggregation

High dimensional data are usually characterised by a high level of redundancy: different variables or sets of variables may share the same information about the label. When classification is based on a subset of the p variables at hand, like in classification tree, different (and sometimes non-overlapping) subsets may be used to achieve the same prediction performance. The choice between the equivalent subsets will arbitrarily depend on the training sample, and small changes of the training sample may lead to drastic changes in subset selection, hence to non-interpretable classifiers (Michiels et al. (2005)). To deal with the redundancy problem, several authors proposed to first aggregate the variables that share the same information into clusters, and then to build a classifier based on these clusters. Several strategies have been proposed (Hastie et al. (2000), Dettling and Buhlmann (2002), Mary-Huard and Robin (2009)), that differ from each others on mainly two points:

- supervised/unsupervised: the aggregation method can take into account the label or not,
- summarising: once the clustering of redundant variables is done, the classifier can be built on the complete clusters, on some of the variables of each cluster, or on a synthesised variable that sums up the cluster information (for instance a linear combination of the cluster variables).

Variable aggregation should be distinguished from variable compression, where one tries to summarise the information shared by the whole set of variables. While variable compression often improves the classification error rate, it usually does not lead to an interpretable classifier.

Classifier aggregation

Bagging.
Bagging can be understood as an application of the classical bootstrap (Efron and Tibshirani (1993)) to classification algorithms. In the classical setting where a parameter θ has to be estimated from a sample, bootstrap consists in re-sampling the sample at hand B times, obtaining an estimation of θ from each sample and then averaging the B estimates. This averaging process results in a bootstrap estimator whose variance is lower than the initial estimator, i.e. the one obtained by estimating θ from the initial sample. The same idea can be exploited in the classification context to improve the performance of an unstable classification algorithm A. Given a training sample, B bootstrap samples are drawn. A classifier f_A^b is obtained by applying algorithm A to each bootstrap sample, and the bagging classifier is then defined as

$$f_A^{bag}(x_0) = \begin{cases} +1 & \text{if } \widehat{p}_{+1}(x_0) > \widehat{p}_{-1}(x_0) \\ -1 & \text{otherwise} \end{cases}, \tag{1.9}$$

$$\text{where } \widehat{p}_\ell(x_0) = \frac{1}{B}\sum_{b=1}^{B} \mathbb{I}\{f_A^b(x_0) = \ell\} \ .$$

While bagging can be applied to any classification algorithm, one may wonder in which circumstances it is fruitful to use it. This can be theoretically answered by looking at the regression case. Assume for this section that the variable Y is quantitative, and that an algorithm A is available to predict Y from X. Let $\mu_A(x_0) = E_P[f_A(x_0)]$ denote the average prediction at a fixed point x_0 (with unknown label Y_0) over all the possible classifiers obtained by applying A to a sample drawn from the joint distribution P of (X, Y). We have (see Breiman (1996)):

$$E_P\left[(Y_0 - f_A(x))^2\right] = E_P\left[(Y_0 - \mu_A(x_0))^2\right] + V_P(f_A(x_0)) \ ,$$

where $V_P(f_A(x))$ is the variance of the prediction at point x_0. From this equation we deduce two important results:

- The average discrepancy between the true label of x_0 and the prediction given by $f_A(x)$ is always higher than the discrepancy between the true label of x_0 and the average prediction $\mu_A(x_0)$.
- The difference between $E_P\left[(Y_0 - f_A(x))^2\right]$ and $E_P\left[(Y_0 - \mu_A(x_0))^2\right]$ is proportional to variance $V_P(f_A(x_0))$, that directly measures the stability of the prediction of algorithm A at point x_0.

The first point means that it is always better (equivalent at worst) to use $\mu_A(x_0)$ rather than $f_A(x)$. Since $\mu_A(x_0)$ cannot be computed in practice (since P is unknown), we can use its bagging version (1.9) as a surrogate. The second point shows that the relative improvement due to bagging will be greater as the variance (instability) of algorithm A increases. As an example, the CART algorithm described in section 1.3.1 is a classical example of unstable classification algorithm (because of its internal variable selection process), and many articles demonstrated the efficiency of bagging when applied to CART. To the contrary, the kNN algorithm is a much stable algorithm, and application of bagging to kNN yields little improvement of the classification performance (Breiman (1996)).

Choosing the number of bootstrap samples is an open question. Selecting a small B can lead to sub-optimality of the resulting classifier in term of classification performance, but increasing B comes to a cost in term of computational burden that is not worthwhile if the gain in performance is marginal. A discussion on this topic can be found in Sutton (2005).

Boosting.

In Schapire (2003), it is argued that building a classifier with good prediction performance is difficult, but building moderately efficient classifiers is often simple. The idea is then to combine such classifiers to build a prediction rule that will be more accurate than any one of the basic classifiers. The boosting algorithm builds the combination iteratively: at each step a new basic classifier is added to the combination. This classifier is trained on a weighted version of the original training set, where increased weights are given to observations that were frequently misclassified during the previous iterations. The final combination then votes to predict the label of any new information.

Many boosting algorithms have been proposed, here we briefly introduce the Adaboost algorithm of Freund and Schapire (1997). For a given classification algorithm A, starting with initial weights $w_i^1 = 1/n$ for observations $i = 1, ..., n$, Adaboost runs B times the following program:

1. Build classifier f_A^b on the training set with weights w_i^b
2. Compute $err_b = \dfrac{\sum_{i=1}^n w_i^b \, \mathbb{I}\left\{f_A^b(x_i) \neq y_i\right\}}{\sum_{i=1}^n w_i^b}$
3. Compute $\alpha_b = \log\left(\dfrac{1 - err_b}{err_b}\right)$
4. Set $w_i^{b+1} \propto w_i^b \exp\left(\alpha_b \, \mathbb{I}\left\{f_A^b(x_i) \neq y_i\right\}\right)$

The final classifier is then:

$$f_A^{boost}(x_0) = \begin{cases} 1 & \text{if } \sum_{b=1}^B \alpha_b f_A^b(x_0) > 0 \\ -1 & \text{otherwise} \end{cases}. \tag{1.10}$$

Equation (1.10) and point 3 show that the weight α_b given to classifier f_A^b in the final voting rule depends on its performance err_b: efficient classifiers receive more weights in the final decision. Point 4 of the program shows that misclassified points during step b will have their weights inflated by a factor $\exp(\alpha_b)$ at step $b + 1$.

While many applications showed the efficiency of boosting in practice (Dettling and Buhlmann (2003)), one may wonder why boosting works. Compared with bagging, boosting does not rely on the fact that many classifiers are built on samples drawn from the *same* empirical distribution, since the distribution (i.e. the weights) change at each boosting iteration. In fact, theoretical arguments have been advanced that show that the way boosting concentrates on observations that are hard to classify is a key of its accuracy (Schapire et al. (1998)). Hard-to-classify points often lie on the boundary between the two classes in $\mathcal{X}$, i.e. in regions where $p_{+1}(x) \approx p_{-1}(x)$, hence the need to focus on these points to sharply evaluate these two quantities at stake in (1.7).

As bagging, boosting can be applied to any classification algorithm, and in practice is often (but not only) applied to trees of moderate size or neural networks (Dietterich (2000); Schwenk and Bengio (1998)). The choice of the number of iterations is also critical. In bagging, increasing the number of iterations makes the resulting classification rule more robust. In boosting, increasing the number of observation may lead to an over-fit of the data. Over-fitting refers to classifiers that exhibit optimistic performance on the training set that are not reproducible on future data to classify. This usually happens when the classification algorithm is free to adapt to the training data without restriction. In boosting, the higher the number of iterations, the higher the risk to over-fit. However, some authors observed that in practice boosting was robust to over-fitting, even for large number of iterations. This behaviour is still under theoretical investigation (Friedman et al. (1998)).

1.3.3 Regularisation

Once a classification algorithm is chosen and the training data collected, obtaining a classifier usually requires the solution of an optimisation problem. The function to optimise can be directly the classification error rate (or one of its surrogates, such as the resubstitution error rate, see section 1.3.4), or some other objective function that depends on the training sample.
In large dimension problems, a direct resolution of the optimisation problem may lead to over-fitting. Regularisation refers to the strategy that consists in adding some constraints to the optimisation process to avoid the aforementioned problems.

L_2 regularisation

Penalised Logistic regression.
To build a LR classifier, one has to estimate parameters $\alpha_j, j = 0, ..., p$. This is done by minimising the negative log-likelihood of the training data

$$\sum_{i=1}^{n} \log\left\{1 + \exp[-y_i g(x_i)]\right\} \quad ,$$

with notations of Equation (1.8). In large dimension problems, applying minimisation without any constraint on the norm of the parameters will lead to inconsistent parameter estimates, with possibly infinite values. To avoid this problem, one can consider the quadratically-regularised log-likelihood (Zhu and Hastie (2004)):

$$\sum_{i=1}^{n} \log\left(1 + \exp[-y_i g(x_i)]\right) + \lambda \|g\|_2^2 \quad , \tag{1.11}$$

where the last term is a *quadratic penalisation function* defined as $\|g\|_2^2 = \sum_{j=1}^{p} \alpha_j^2$.

Support Vector Machine.
SVM is another popular classification algorithm, that can be understood as a classification method with an embedded regularisation process. Indeed, the SVM algorithm (Vapnik (1995)) fits a function $g(x) = \alpha_0 + \sum_{j=1}^{p} \alpha_j x^j$ that minimises

$$\sum_{i=1}^{n} \left(1 - y_i g(x_i)\right)_+ + \lambda \|g\|_2^2 \quad , \tag{1.12}$$

where $(u)_+ = u$ if u is positive, 0 otherwise. The resulting SVM classifier is

$$f_{SVM}(x) = \begin{cases} +1 & \text{if } g(x) > 0 \\ -1 & \text{otherwise} \end{cases} .$$

Interpretation.
It is worth noticing that expressions (1.11) and (1.12) are similar, and lead to a general interpretation of regularised methods. The regularised optimisation problem can be rewritten as:

$$\sum_{i=1}^{n} \ell\left(g,(x_i,y_i)\right) + \lambda R(g) \enspace .$$

The loss function $\ell(.,.)$ evaluates the quality of adjustment of function g to the training data: if $g(x_i)$ and y_i have the same sign, then the classifier associated to g makes the right prediction for x_i., and the loss is low. If the signs of $g(x_i)$ and y_i differ, the loss is high. Note that the loss increases according to $-y_i g(x_i)$: severe classification errors make the loss higher.
The penalisation function $R(.)$ takes into account the statistical cost of function g. For example, in (1.11) and (1.12), the regularisation function is the Euclidian norm of the parameters (also called the L_2-norm). This term implies that given two functions g_1 and g_2 with identical fits to the training set, the simplest one (in term of norm) should be preferred.
Regularisation amounts to finding a tradeoff between adjustment and cost. Depending on parameter λ, emphasis will be put either on fit or on cost. Tuning parameter λ to achieve optimal performance is a difficult task. In practice, λ can be chosen by cross-validation (see Section 1.3.4), .

Other regularisation functions

We observed that penalised LR and SVM correspond to the same regularisation strategy, applied with different loss functions. It is also possible to consider alternative regularisation functions, that are more adapted to the nature of the data, or to the classification problem at hand.

L_1 ***regularisation.***
In large dimension problems, it is sometimes assumed that only a small subset of variables is informative to predict the label. In this situation, a relevant classifier should only depend on this (unknown) subset. While L_2 regularisation globally shrinks coefficients, in practice, even for a large value of λ, no or only a few of them will be shrunk to 0. Conversely, the use of the L_1 regularisation function $R(g) = \sum_{j=1}^{p} |\alpha_j|$ will tend to sparse solutions where only a few parameters will be non-zeros (Lee et al. (2006); Tibshirani (1996)). Therefore, L_1 regularisation (also called Lasso regularisation) is commonly used as a variable selection strategy.

Fused lasso regularisation.
In some situations, a natural ordering of the predictor variables can be defined. Consider for instance Comparative Genomic Hybridisation (CGH) array experiments where variables correspond to probe measurements. Since two adjacent probes on a given chromosome are likely to share the same copy number, their

associated weights in the decision rule should be similar. A relevant classifier should be sparse (only some chromosomal regions are informative), and parameter estimates corresponding to adjacent variables (i.e. adjacent probes) should be identical (Rapaport et al. (2008); Tibshirani and Wang (2008)).
To this purpose, the following regularisation function can be considered:

$$R(g) = \lambda_1 \sum_{j=1}^{p} |\alpha_j| + \lambda_2 \sum_{j_1 \neq j_2} |\alpha_{j_1} - \alpha_{j_2}| \ .$$

The first term corresponds to the Lasso penalty that guarantees sparsity, and the second term causes the classification algorithm to produce similar weights for adjacent variables.

These two examples illustrate the flexibility of regularisation methods, and shed light on their effectiveness to benefit from prior information about the data structure, and produce classifiers with good prediction performance. In practice, the implementation of regularisation methods may raise difficult optimisation problems, that have to be taken into account when elaborating a new regularisation function.

1.3.4 Performance assessment

In previous sections we saw many algorithms to build or combine classifiers. While some algorithms may be preferred for some specific applications, we usually have little clue about how to select the classification algorithm to use. In practice, the choice will often be based on the respective performance of each classifier. It is therefore critical to have at our disposal efficient tools to assess the performance of any classifier, both for evaluation and comparison.

An intuitive strategy would be to evaluate the classifier $\widehat{f}$ on the training data, and compare the prediction obtained to the true labels The "$\widehat{\ }$" notation in $\widehat{f}$ underlines the fact that $\widehat{f}$ was already built using the training data. More precisely, we can compute the empirical error rate (also called resubstitution error rate)

$$EER(\widehat{f}) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}\left\{\widehat{f}(x_i) \neq y_i\right\} \ .$$

Unfortunately, this strategy leads to a biased estimation of the true classification error rate of the classifier, as defined in (1.6). Indeed, since the data are used both for building and evaluating the classifier, EER will lead to an optimistic estimation of the true error rate. Therefore no fair evaluation or comparison between classifiers is possible using EER. From this, we conclude that an unbiased evaluation necessitates to split the data into two distinct samples, one for construction and one for evaluation. The next section presents different strategies of splitting that are commonly used.

Hold-out and cross-validation

Hold-out (HO).
Usually, a single sample is collected at once, and then split into a *train* and a *test* sample, for construction and evaluation respectively. This strategy is called Hold Out. Defining $e = (X_1, Y_1), ..., (X_{n-m}, Y_{n-m})$ and $\bar{e} = (X_{n-m+1}, Y_{n-m+1})$, ..., (X_n, Y_n), the hold-out estimator is

$$HO(\widehat{f^e}) = \frac{1}{m} \sum_{j=n-m+1}^{n} \mathbb{I}\left\{\widehat{f^e}(X_j) \neq Y_j\right\} = \frac{1}{m} \sum_{j \in \bar{e}} \mathbb{I}\left\{\widehat{f^e}(X_j) \neq Y_j\right\} \ ,$$

where $\widehat{f^e}$ is the classifier built on sample e. Although HO is unbiased, the splitting of the data into train/test induces some randomness due to the arbitrary choice of the withdrawn data points.

Leave-m-out (LmO).
To avoid an arbitrary choice between classifiers based on a particular splitting of the data, every possible subset of m observations is successively left out of the initial sample and used as test set. The algorithm with the smallest averaged classification error rate is then selected. The LmO estimator is then

$$LmO(\widehat{f}) = \binom{n}{m}^{-1} \sum_{e} \left[\frac{1}{m} \sum_{j \in \bar{e}} \mathbb{I}\left\{\widehat{f^e}(X_j) \neq Y_j\right\}\right] = \binom{n}{m}^{-1} \sum_{e} HO(\widehat{f^e}) \ ,$$

where the sum is over all the possible subsets e of size $n - m$. While LmO solves the randomness problem of HO, its computational cost is prohibitive for large values of m, and in practice only the leave-one-out (LOO) procedure is routinely used.

K-fold (KF).
An intermediate strategy consists in dividing the complete dataset into K subsamples $\bar{e}_1, ..., \bar{e}_K$ with roughly equal size n/K, each subsample being successively used for validation. This leads to the following estimator:

$$KF(\widehat{f}) = \frac{1}{K} \sum_{e_k} HO(\widehat{f^{e_k}}) \ .$$

KF with $m = n/K$ is a compromise between HO and LmO: it is more stable than HO since it averages the results of different samplings, and its computational cost is much lower than the one of LmO.

Choosing the train/test proportions.
All these procedures request the tuning of a parameter that balances the sizes of the training and test sets: m for HO and LmO, K for KF. Choosing different values for this parameter yields different evaluations of the performance, and sometimes different conclusions about which classifier performs best. The optimal tuning of this parameter is still an open question, a review can be found in Arlot and Celisse (2010).

About the good use of cross-validation.

We emphasise again that all the resampling methods presented here assume that the test set is never used at any step of the training. In particular, selecting a subset of variables, or tuning the inner parameters of a classification algorithm are part of training process, and therefore should not involve the test set. In Ambroise and McLachlan (2002) for instance, the authors showed that the performance estimation could be severely biased when the variable selection step is performed external to rather than within the cross-validation loop, an error that still occurs in publications.

ROC curves, AUC

So far, we worked under the implicit hypothesis that all types of classification error have the same cost: whether an observation is misclassified as $+1$ or misclassified as -1 is equivalent. In many applications this assumption does not hold: for instance, in cancer diagnosis, to misclassify an individual as ”healthy” or ”ill” do not have the same consequences. It such cases, one needs to focus either on the sensitivity or the specificity of the classification rule, where

$$\begin{aligned} & Sensitivity = \Pr\{f(X)=1|Y=1\} \\ \text{and} \qquad & Specificity = \Pr\{f(X)=-1|Y=-1\}\ . \end{aligned}$$

In section 1.3.1, we defined the optimal classifier as

$$f_B^c(x) = \begin{cases} +1 & \text{if } p_{+1}(x) > t \\ -1 & \text{otherwise} \end{cases},$$

where $t = 0.5$. This particular choice of threshold t assumes balance between sensitivity and specificity. It is possible to change the tradeoff between these two quantities by tuning the threshold: the higher t, the higher the specificity (and the lower the sensitivity).

The evolution of the sensitivity/specificity tradeoff can be investigated for a given classifier via the Receiver Operating Characteristic curve, or ROC curve, that plots sensitivity versus $1-$specificity as t varies from 0 to 1. Figure 1.1 (right) exhibits 4 ROC curves obtained from classifiers A, B, C and D. The bottom-left corner corresponds to high values of t, for which sensitivity is high and specificity low. As t increases, sensitivity decreases whereas specificity increases. For instance, if ones requests a specificity of at least 80% then classifier B will score a sensitivity of 65%.

ROC curves may also be useful to compare classifiers. For instance, in Figure 1.1 classifier C should be preferred to the other ones since its ROC curve is uniformly higher. In practice though, different classifiers may be optimal for different sets of values of t, resulting in no obvious leader: in Figure 1.1, if a high specificity is requested, one should prefer B to D. To the contrary, if the goal is to achieve high sensitivity, D should be preferred to D.

The Area Under the Curve (AUC) provides a summary of the classifier performance over all values of t. A perfect classifier with high sensitivity and specificity at the same time would score an AUC of 1. Classifiers can then be compared using their AUC scores, according to the rule ”the higher the better”. According to this criterion, classifier B should be preferred to classifier D since their respective AUC score are 0.78 and 0.71, respectively.

1.4 Class discovery

Clustering is one of the most used techniques to find some structure within a dataset of entities. This problem is also known as ’unsupervised classification’ for the classes are to be discovered based on a set of unlabelled observations. This makes a strong distinction with the previous section where classes were ’learned’, based on a set of labelled observations. Clustering aims at finding an underlying structure (i.e. clusters) that is *a priori* unknown. Although other descriptive techniques, such as principal component analysis (PCA), may sometimes reveal the existence of clusters, this is not their primary goal. They are therefore not considered as clustering methods..
Gene clustering based on expression data has become a common place in the last decades and tens (hundreds?) of strategies have been proposed to this end. Two main approaches can be distinguished: geometric (or distance-based) techniques and probabilistic (or model-based) techniques. We will briefly introduce the former and concentrate on the latter. As for the notations, we consider a dataset of n observations ($i = 1 \dots n$), each characterised by a vector X_i, that we aim at clustering into K distinct clusters. Depending on the application, K may be known *a priori* or not.

1.4.1 *Geometric methods*

Hierarchical algorithms

Distance-based clustering traces back to early ages of taxonomy (Sokal and Sneath (1963)). The most typical algorithms provide a hierarchical structure and work as follows. Start with n clusters each made of one observation. At each step the two closest individuals (or groups of individuals) are merged to make a new cluster (Hastie et al. (2001)). The number of clusters is hence reduced by 1 at each step. The clustering process is often depicted as a tree, also called ’dendrogram’, with n leaves corresponding to each individual, and one root corresponding to the final cluster that includes them all. Such algorithms are based on three key ingredients (a more detailed presentation can be found in Mary-Huard et al. (2006)):

Distance between observations. We first need a distance (or similarity, or dissimilarity) measure $d(i, j)$ between observations, to decide which ones have to be first clustered. It is not possible to summarise the huge variety of distances that can be considered. It goes from simple euclidian distances $d(i, j) = \|X_i - X_j\|$ when observations X_i are vectors of real numbers (e.g. gene expression levels, see D’haeseleer (2005) for some examples) to so-called kernels when observations are made of complex heterogenous descriptors (Schölkopf and Smola (2002)).

Aggregation criterion. Once the hierarchical clustering process has started, some observations are gathered into clusters that are themselves to be clustered. This requires to define a second distance $D(A, B)$, where A and B are clusters. Again, there exists an impressive list of such distances (simple linkage, diameter, UPGMA, etc), which often give their name to the clustering algorithm itself.

Stopping rule. Because clustering aims at finding a 'reasonable' number of groups, the process has to be stopped before it ends with 1 single cluster. In absence of modelling, the choice of the number of clusters often relies on some heuristics, such as finding long branches in the dendrogram, or finding change-points in the behaviour of the variance between clusters along the clustering process.

K means

Non-hierarchical methods where the number of groups is known *a priori* also exist. The most emblematic is probably the K means algorithm where the K clusters are associated with K centroids (Hastie et al. (2001)). The algorithm starts with K randomly chosen centroids ands then alternates 2 steps:

1. Each observation is clustered with its closest centroid;
2. Centroids are updated as the mean of the observations clustered together.

This kind of algorithms are known to converge very quickly and to be very sensitive to the choice of the initial centroids ('seeds').

1.4.2 (Discrete) latent variable models

The clustering problem can be set in a model-based setting via the definition of latent random variables $Z_i \in \{1 \dots K\}$ corresponding to the labels of each observation (Equivalently, we define the binary variable Z_{ik} which is 1 if $Z_i = k$ and 0 otherwise). In this setting, the observations X_i are often supposed to be independent conditionally to the Z_i, with label-dependent distribution:

$$\{X_i\}_{i=1\dots n} \text{ independent } |\{Z_i\}_{i=1\dots n}, \qquad (X_i|Z_{ik}=1) \sim f(\cdot;\theta_k). \tag{1.13}$$

f is often chosen within a family of parametric distribution (with parameter θ_k) that seem relevant for the observed data X_i. Note that f can be any distribution, up to non-parametric regression (Luan and Li (2003)) or even more complex structures. Denoting $X = \{X_i\}$ the observed data and $Z = \{Z_i\}$ the unobserved labels, the conditional log-likelihood of X under (1.13) is

$$\log P(X|Z) = \sum_i \sum_k Z_{ik} \log f(X_i;\theta_k) \tag{1.14}$$

Such models mostly differ for the distribution of the unobserved labels Z. We give here a quick overview of the most classical models:

Mixture: The simplest model assumes that the labels are independent and identically distributed (i.i.d.) with multinomial distribution $\mathcal{M}(1; \boldsymbol{\pi})$ where $\boldsymbol{\pi}$ is the vector of proportions of each group (McLachlan and Peel (2000)), so we have

$$\log P(Z) = \sum_{i,k} Z_{ik} \log \pi_k. \tag{1.15}$$

Hidden Markov Model (HMM): When data are observed along a single direction (such as time or genomic position), it may be relevant to introduce a dependency between the labels of adjacent positions. So-called HMM often actually refer to hidden Markov chain models where Z is assumed to be a Markov chain with transition matrix $\boldsymbol{\pi}$:

$$\log P(Z) \simeq \sum_{i,k,\ell} Z_{i-1,k} Z_{i\ell} \log \pi_{k\ell}. \tag{1.16}$$

Such models are intensively used in sequence analysis (Durbin et al. (1999)).

Hidden Markov Random Field (HMRF): When the data have a more general spatial organisation (such as pixels in an image), $\{Z_i\}$ can be assumed to be a Markov random field where each Z_i only depends on its neighbours and is conditionally independent from the rest (Besag (1974)).

Stochastic Block Model (SBM): This last example refers to interaction graphs where nodes can be proteins or species linked to each other by edges revealing possible physical interactions or some similarity in terms of phenotype. The observed data are then the (possibly valued) edges $\{X_{ij}\}$ between the n nodes. SBM is a mixture model stating that the unknown labels are i.i.d. $\mathcal{M}(1; \boldsymbol{\pi})$ and that the edges values are conditionally independent with distribution depending on the labels of both nodes (Nowicki and Snijders (2001)): $X_{ij} | (Z_i = k, Z_j = \ell) \sim f(\cdot, \theta_{k\ell})$. The log-likelihood of Z is hence the same as in (1.15) but, due to the network structure, the conditional log-likelihood of X differs from (1.14):

$$\log P(X|Z) = \sum_{i,j} \sum_{k,\ell} Z_{ik} Z_{j\ell} \log f(X_{ij}; \theta_{k\ell}). \tag{1.17}$$

1.4.3 Inference

Maximum likelihood inference

As for all incomplete data models, the main issue for the inference of such models comes from the unobserved labels Z_i. Although several approaches do exist, maximum likelihood is the most commonly used. In order to maximise the likelihood of the observed data $\log P(X)$, most inference algorithms include a 'retrieval' step where the conditional distribution of the unobserved labels given the observations $P(Z|X)$ is calculated or approximated.

Likelihoods.
The likelihood of the observed data $\log P(X)$ is often difficult (or even impossible) to handle, whereas the likelihood of the complete dataset

$$\log P(X, Z) = \log P(Z) + \log P(X|Z)$$

generally has a much more convenient form, as seen in Section 1.4.2. A connexion between these likelihoods can be made (Dempster et al. (1977)) as

$$\log P(X) = \mathbb{E}[\log P(X, Z)|X] + \mathcal{H}[P(Z|X)] \tag{1.18}$$

where $\mathcal{H}(F)$ stands for the entropy of distribution F defined as $\mathcal{H}(F) = -\int F(z) \log F(z) \mathrm{d}z$. The calculation of the two elements in the right-hand-side requires the evaluation of the conditional distribution $P(Z|X)$, or at least some of its moments. All expectation-maximisation (EM) like algorithms alternate

1. E-step: evaluation of $P(Z|X)$;
2. M-step: maximisation of $\mathbb{E}[\log P(X, Z)|X]$ w.r.t. the parameters.

The M-step step does, in general, not raise more difficulties than classical maximum-likelihood inference.

Conditional distribution.
The evaluation of $P(Z|X)$ is hence the crucial step for the inference of such models. We go back to some models presented above to illustrate how this can be achieved.

Mixture: Because all couples (X_i, Z_i) are independent, the Z_i are conditionally independent and the distribution if each of them is given by the Bayes formula:

$$P(Z_i = k|X) = P(Z_i = k|X_i) = \pi_k f(X_i; \theta_k) \bigg/ \sum_\ell \pi_\ell f(X_i; \theta_\ell) \tag{1.19}$$

HMM: Due to the underlying Markov structure, the Z_i are not conditionally independent. However, adding (1.16) and (1.14) shows that only single labels Z_{ik} and *couples of adjacent labels* $Z_{i-1,k} Z_{i,\ell}$ are involved in $\log P(X, Z)$. This proves that the Z_i still have a Markovian dependency structure. The conditional distribution $P(Z|X)$ can be obtained via the so-called 'forward-backward' recursion, with a linear complexity (Cappé et al. (2005)). The forward step iteratively calculates the conditional distribution of Z_i given the past observations: $P(Z_i|X_1 \dots X_i)$. The backward step completes the calculation of all $P(Z_i|X)$, starting from $P(Z_n|X_1 \dots X_n) = P(Z_n|X)$, and going back to $P(Z_1|X)$.

HMRF and SBM: The conditional distribution of the labels displays a more intricate dependency structure. In HRMF it is still a Markovian structure but, in absence of natural order in dimension 2 or more, no efficient recursion such as 'forward-backward' can be derived. In SBM, the sum of (1.15) and (1.17) involves both single labels Z_{ik} and *all couples of labels* $Z_{i,k}Z_{j,\ell}$. This proves that their conditional dependency structure is a clique (Daudin et al. (2008)) and no simplification can be expected. In such models, the conditional distribution $P(Z|X)$ can only be either estimated via Monte-Carlo techniques (giving raise to stochastic versions of EM such as Stochastic EM (SEM), Celeux and Diebolt (1992), Stochastic Approximation EM (SAEM), Delyon et al. (1999), *etc.*) or approximated with mean-field (Jaakkola (2000)) or message-passing (Winn and Bishop (2005)) techniques.

Variational point of view.
From a more general point of view, these inference strategies can be considered as variational techniques which aim at finding the distribution $Q(Z)$ that best approximates $P(Z|X)$. The most popular strategy consists in minimising the Küllback-Leibler divergence $KL[Q(Z); P(Z|X)]$, where

$$KL[F(Z); G(Z)] = \int F(z) \log[F(z)/G(z)]\mathrm{d}z.$$

As the Küllback-Leibler divergence is always positive, for any distribution $Q(Z)$, $\log P(X)$ can be lower-bounded by $\log P(X) - KL[Q(Z); P(Z|X)]$, that can be reformulated as (Jaakkola (2000))

$$\log P(X) \geq \int Q(z) \log P(X, z)\mathrm{d}z + \mathcal{H}(Q). \tag{1.20}$$

This lower bound is similar to (1.18), except that $P(Z|X)$ is replaced by $Q(Z)$. Regular EM algorithms correspond to cases (such as mixtures or HMM) where $P(Z|X)$ can actually be computed, so $Q(Z)$ is chosen as $Q(Z) = P(Z|X)$. The divergence is hence set to zero, so (1.20) equals (1.18) and maximum likelihood inference is achieved.
In more complex cases (HMRF, SBM), $Q(Z)$ is chosen as the best possible approximation (in terms of Küllback-Leibler divergence) of $P(Z|X)$ among a certain class of manageable distributions. It results in the maximisation of a lower bound of the likelihood, which is not equivalent to maximum likelihood inference (Gunawardana and Byrne (2005)). Such strategies can be generalised to many graphical models (Jordan et al. (1999)) that will be introduced later in this book.
Note that the class of incomplete data models is not limited to those described here, but also includes mixed models described in Section 1.2. Another interesting example of such a modelling can be found in Beal et al. (2005). Note also that a Bayesian version of variational inference does exist in the context of exponential family distributions (Beal and Ghahramani (2003)).

Additional issues

Model selection.
The determination of the number of groups K is a problem per se. Although in some situations it may be made based on some prior knowledge, K generally needs to be estimated as well as the other parameters. Because K class mixtures are part of the set of $K+1$ class mixtures, the maximised likelihood $\log P(X; \widehat{\pi}_K, \widehat{\theta}_K)$ can not be used to choose K as it always increases when K increases. In the context of maximum likelihood inference, most criteria consist in adding a penalty to the likelihood, which account for the complexity of the model (Burnham and Anderson (1998)). The Akaike's criterion, known as AIC and the Bayesian Information Criterion (BIC) are the two most famous examples of that kind. The latter is an approximation of the probability $P(K, X)$ based on a Laplace approximation. It is hence only valid for reasonably large datasets. BIC is clearly most popular in unsupervised classification (McLachlan and Peel (2000)) and is defined as

$$\text{BIC}(K) = \log P(X; \widehat{\boldsymbol{\pi}}_K, \widehat{\boldsymbol{\theta}}_K) - .5(\text{\# parameters}) \times \log n$$

where $\widehat{\boldsymbol{\pi}}$ and $\widehat{\boldsymbol{\theta}}$ are the maximum likelihood estimates of $\boldsymbol{\pi}$ and $\boldsymbol{\theta}$. In the example of a HMM where each parameter θ_k has dimension D, there are $K(K-1)$ independent transition probabilities, so the total number of parameters is $K(D+K-1)$.
In the context of classification, the fit of the model may not be the only desirable property to achieve. It is also important to account for the reliability of the classification which can be measured by the entropy of the conditional distribution $\mathcal{H}[P(Z|X)]$. A large value of this entropy reveals that the classification is uncertain for a large part of the observation. Biernacki et al. (2000) defined the Integrated Completed Likelihood (ICL) criterion, which consists in adding this entropy to the penalty. Combined with the decomposition (1.18), is reduces to

$$\text{ICL}(K) = \mathbb{E}[\log P(X, Z; \widehat{\boldsymbol{\pi}}, \widehat{\boldsymbol{\theta}})|X] - .5(\text{\# parameters}) \times \log n.$$

Variable selection.
When the clustering is based on a large set of descriptors (e.g. $X_i \in \mathbb{R}^d$, $d \gg 1$), the question of selecting the relevant descriptors raises naturally. The distribution of relevant descriptors is supposed to change from one group to another, whereas irrelevant descriptors keep a constant distribution for all observations. If the labels where known, the problem would reduce to a class comparison problem (see Section 1.2). In the unsupervised context, it must be combined with the determination of the classes and this raises both theoretical and algorithmic issues (Maugis et al. (2009); Raftery and Dean (2006)).

Classification.
As explained above, the inference of latent-variable models provides the (approximate) conditional distribution of the labels $P(Z|X)$. The posterior probability $P(Z_i = k|X)$ may seem sufficient as they provide a 'fuzzy' classification. This also informs us about the reliability of the classification: classification is clearly risky when all conditional probabilities are similar.

When a formal classification is needed, the maximum a posteriori (MAP) strategy consists in classifying each observation to the class with highest conditional probability:

$$\widehat{Z}_i = \arg\max_k P(Z_i = k|X).$$

It is worth to note that, in general, the set of most probable labels $\{\widehat{Z}_i\}$ is not equal to the most probable set of labels $\widehat{Z}$. As an example, in an HMM, the most probable hidden path

$$\widehat{Z} = \arg\max_z P(Z = z|X)$$

can be retrieved via the Viterbi algorithm (Cappé et al. (2005)). It is not made of the succession of MAP predictions $\widehat{Z}_i$ at each position. The difference is due to the distinction between the marginal classification of one observation and the joint classification of all observations. The latter provides a path which is typically more consistent with the estimated transition probabilities than the former, which concentrates on the separate classification of each observation.

References

Allison DB, Gadbury G, Heo M, Fernandez J, Lee CK, Prolla TA and Weindruch RA 2002 Mixture model approach for the analysis of microarray gene expression data. *Comput. Statist. and Data Analysis* **39**, 1–20.

Ambroise C and McLachlan GJ 2002 Selection bias in gene extraction on the basis of microarray gene-expression data. *Proc. Natl. Acad. Sci. U.S.A.* **99**, 6562–6566.

Arlot S and Celisse A 2010 A survey of cross-validation procedures for model selection. *Statist. Surv.* **4**, 40–79.

Beal, M. J and Ghahramani Z 2003 The variational Bayesian EM algorithm for incomplete data: with application to scoring graphical model structures. *Bayes. Statist.* **7**, 543–52.

Beal MJ, Falciani F, Ghahramani Z, Rangel C and Wild D 2005 A bayesian approach to reconstructing genetic regulatory networks with hidden factors. *Bioinformatics* **21**(3), 349–56.

Benjamini Y and Hochberg Y 1995 Controlling the false discovery rate: a practical and powerfull approach to multiple testing. *JRSSB* **57**(1), 289–300.

Besag J 1974 Spatial interaction and the statistical analysis of lattice systems (with discussion). *J. of Royal Statist. Soc., series B* **36**(2), 192–326.

Biernacki C, Celeux G and Govaert G 2000 Assessing a mixture model for clustering with the integrated completed likelihood. *IEEE Trans. Pattern Anal. Machine Intel.* **22**(7), 719–725.

Blanchard G and Roquain E 2008 Two simple sufficient conditions for FDR control. *Electron. J. Statist.* **2**, 963–92.

Blanchard G and Roquain E 2009 Adaptive false discovery rate control under independence and dependence. *J. Machine Learn. Res.* **10**, 2837–71.

Breiman L 1996 Bagging predictors *Machine Learning*, vol. 24, pp. 123–140.

Breiman L, Friedman J, Olshen R and Stone C 1984 *Classification and regression trees*. Wadsworth International, Belmont, CA.

Burnham KP and Anderson RA 1998 *Model Selection and Inference: A Practical Information-Theoretic Approach*. Wiley: New-York.

Cappé O, Moulines E and Rydén T 2005 *Inference in Hidden Markov Models*. Springer:Berlin.

Celeux G and Diebolt J 1992 A stochastic approximation type EM algorithm for the mixture problem. *Stochastics* **1-2**, 119–34.

Celisse A and Robin S 2010 A cross-validation based estimation of the proportion of true null hypotheses. *J. Statist. Planing Infer.* **140**, 3132–47.

Cressie N 1993 *Statistics for Spatial Data*. Wiley, NY.

Daudin JJ, Picard F and Robin S 2008 A mixture model for random graphs. *Stat. Comput.* **18**(2), 173–83.

Delyon B, Lavielle M and Moulines E 1999 Convergence of a stochastic approximation version of the EM algorithm. *Ann. Statist.* **27**(1), 94–128.

Demindenko E 2004 *Mixed Models: Theory and Applications*. Wiley:Hoboken.

Dempster AP, Laird NM and Rubin DB 1977 Maximum likelihood from incomplete data via the EM algorithm. *J. R. Statist. Soc. B* **39**, 1–38.

Dettling M and Buhlmann P 2002 Supervised clustering of genes. *Genome Biology* **3**(12), 1–15.

Dettling M and Buhlmann P 2003 Boosting for tumor classification with gene expression data. *Bioinformatics* **19**(9), 1061–1069.

D'haeseleer P 2005 How does gene expression clustering work?. *Nat. Biotechnol.* **23**, 1499–1501.

Dietterich T 2000 An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning* **40**(2), 139–157.

Diggle P, Liang K and Zeger S 2002 *Analysis of Longitudinal Data* 2nd edn. London: Oxford Science.

Dobson AJ 1990 *An introduction to generalized linear models*. Chapman & Hall:London.

Dudoit S, Shaffer JP and Boldrick JC 2003 Multiple hypothesis testing in microarray experiments. *Statistical Science* **18**(1), 71–103.

Durbin R, Eddy S, Krogh A and Mitchison G 1999 *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press.

Efron B and Tibshirani R 1993 *An Introduction to the Bootstrap.*. FL: Chapman & Hall/CRC.

Efron B, Tibshirani R, Storey JD and Tusher V 2001 Empirical bayes analysis of a microarray experiment. *J. Amer. Statist. Assoc.* **96**, 1151–1160.

Fai A and Cornelius P 1996 Approximate F-tests of multiple degree of freedom hypotheses in generalized least squares analyses of unbalanced split-plot experiments. *J. Statis. Comput. Simul.* **54**, 363–78.

Fix E and Hodges J 1951 Discriminatory analysis, nonparametric discrimination: Consistency properties. Technical Report 4, U.S. Air Force, School of Aviation Medicine.

Fix E and Hodges J 1952 Nonparametric discrimination: small sample performance. Technical Report 11, U.S. Air Force, School of Aviation Medicine.

Freund Y and Schapire R 1997 A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences* **55**, 119–139.

Friedman J, Hastie T and Tibshirani R 1998 Additive logistic regression: a statistical view of boosting. *Annals of Statistics* **28**, 2000.

Friguet C, Kloareg M and Causeur D 2009 A factor model approach to multiple testing under dependence. *J. Amer. Statist. Assoc.* **104**(488), 1406–15.

Genovese C and Wasserman L 2002 Operating characteristics and extensions of the fdr procedure. *J. R. Statist. Soc. B* **64**(3), 499–517.

Genovese C and Wasserman L 2004 A stochastic process approach to false discovery control. *Ann. Statist.* **32**(3), 499–518.

Golub TR, Slonim DK, Tamayo P, Huard C, Gaasenbeek M, Mesirov J, Coller H, Loh M, Downing J, Caligiuri M, Bloomfield C and Lander E 1999 Class prediction and discovery using gene expression data. *Science* **286**, 531–537.

Gunawardana A and Byrne W 2005 Convergence theorems for generalized alternating minimization procedures. *J. Mach. Learn. Res.* **6**, 2049–73.

Hastie T, Tibshirani R and Friedman J 2001 *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.*. Springer, New York.

Hastie T, Tibshirani R, Eisen MB, Alizadeh A, Levy R, Staudt L, Chan WC, Botstein D and Brown P 2000 'Gene shaving' as a method for identifying distinct sets of genes with similar expression patterns. *Genome Biol.* **1**(RESEARCH0003), 1–21.

Hue M, Riffle M, Vert J and Noble W 2010 Large-scale prediction of protein-protein interactions from structures. *BMC Bioinformatics*.

Jaakkola TS 2000 Tutorial on variational approximation methods In *In Advanced Mean Field Methods: Theory and Practice* (ed. Opper M and Saad D), pp. 129–159. MIT Press.

Jordan MI, Ghahramani Z, Jaakkola T and Saul LK 1999 An introduction to variational methods for graphical models. *Machine Learning* **37**(2), 183–233.

Kenward M and Roger J 1997 Small sample inference for fixed effects from restricted maximum likelihood. *Biometrics* **53**, 983–97.

Kerr MK and Churchill G 2001 Experimental design for gene expression microarrays. *Biostatistics* **2**, 183–201.

Khuri A, Mathew T and Sinha B 1998 *Statistical tests for mixed linear models*. Wiley:New-York.

Kim KI and van de Wiel MA 2008 Effects of dependence in high-dimensional multiple testing problems. *BMC Bioinformatics* **9**, 114.

Langaas M, Lindqvist B and Ferkingstad E 2005 Estimating the proportion of true null hypotheses, with applications to DNA microarray data. *J. R. Statist. Soc. B* **67**(4), 555–572.

Lee SI, Lee H, Abbeel P and Ng AY 2006 Efficient l1 regularized logistic regression *AAAI*. AAAI Press.

Luan Y and Li H 2003 Clustering of time-course gene expression data using a mixed-effects model with b-splines. *Bioinformatics* **19**(4), 474–482.

Mary-Huard T and Robin S 2009 Tailored aggregation for classification. *IEEE Trans Pattern Anal Mach Intell* **31**, 2098–2105.

Mary-Huard T, Picard F and Robin S 2006 Introduction to statistical methods for microarray data analysis In *Mathematical and Computational Methods in Biology* (ed. Maass A, Martinez S and Pecou E), pp. 56–126.

Maugis C, Celeux G and Martin-Magniette ML 2009 Variable selection in model-based clustering: A general variable role modeling. *Comput. Stat. Data Anal.* **53**(11), 3872–3882.

McLachlan G and Peel D 2000 *Finite Mixture Models*. Wiley:New-York.

McLachlan G, Bean R and Ben-Tovim Jones L 2006 A simple implementation of a normal mixture approach to differential gene expression in multiclass microarrays. *Bioinformatics* **22**, 1608–1615.

Michiels S, Koscielny S and Hill C 2005 Prediction of cancer outcome with microarrays: a multiple random validation strategy. *Lancet* **365**, 488–492.

Nowicki K and Snijders T 2001 Estimation and prediction for stochastic block-structures. *J. Amer. Statist. Assoc.* **96**, 1077–87.

Quinlan J 1993 *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc.

Raftery A and Dean N 2006 Variable selection for model-based clustering. *J. Amer. Statist. Assoc.* **101**(473), 168–78. DOI: 10.1198/016214506000000113.

Rapaport F, Barillot E and Vert JP 2008 Classification of arrayCGH data using fused SVM. *Bioinformatics* **24**, i375–82.

Robin S, Bar-Hen A, Daudin JJ and Pierre L 2007 A semi-parametric approach for mixture models: Application to local false discovery rate estimation. *Comput. Statist. and Data Analysis* **51**(12), 5483–93.

SAS 2002-03 *SAS OnlineDoc 9.1.3*. `support.sas.com/onlinedoc/913/`.

Schapire R 2003 The boosting approach to machine learning: An overview In *Nonlinear Estimation and Classification* (ed. Denison DD, Hansen MH, Holmes C, Mallick B and Yu B), pp. 149–72. Springer:Berlin.

Schapire R, Freund Y, Bartlett P and Lee W 1998 Boosting the margin: a new explanation for the effectiveness of voting methods. *Ann. Statist.* **26**(5), 1651–1686.

Schölkopf B and Smola A 2002 *Learning with kernels*. MIT Press.

Schwenk H and Bengio Y 1998 Training methods for adaptive boosting of neural networks In *Conference on Advances in Neural Information* (ed. Cohn DA), pp. 647–653.

Searle S 1971 *Linear models* 1st edn. Wiley:New-York.

Sokal RR and Sneath PHA 1963 *Principles of numerical taxonomy*. Freeman.

Storey JD 2002 A direct approach to false discovery rate. *J. R. Statist. Soc. B* **64**(3), 479–498.

Strimmer K 2008 fdrtool: a versatile R package for estimating local and tail area-based false discovery rates. *Bioinformatics* **24**(12), 1461–62.

Sun W and Cai TT 2008 Large-scale multiple testing under dependence. *J. R. Statist. Soc. B* **71**(2), 393–424.

Sutton CD 2005 Classification and regression trees, bagging, and boosting In *Handbook of statistics: Data mining and data visualization* (ed. Rao C, Wegman E and Solka J), pp. 303–329. Elsevier/North-Holland:Amsterdam.

Tempelman RJ 2008 Statistical analysis of efficient unbalanced factorial designs for two-color microarray experiments. *Int J Plant Genomics* **2008**, 584360.

Tibshirani R 1996 Regression shrinkage and selection via the lasso.. *J. Royal. Statist. Soc B.* **58**(1), 267–288.

Tibshirani R and Wang P 2008 Spatial smoothing and hot spot detection for CGH data using the fused lasso. *Biostatistics* **9**, 18–29.

Vapnik V 1995 *The Nature of Statistical Learning Theory*. Springer Verlag, NY.

Verbeke G and Molenberghs G 2000 *Linear Mixed Models for Longitudinal Data*. Springer.

Westfall PH and Young SS 1993 *Resampling-Based Multiple Testing: Examples and Methods for P-value Adjustment*. Wiley.

Winn JM and Bishop C 2005 Variational message passing. *J. Mach. Learn. Res.* (6), 661–694.

Wolfinger RD, Gibson G, Wolfinger ED, Bennett L, Hamadeh H, Bushel P, Afshari C and Paules RS 2001 Assessing gene significance from cDNA microarray expression data via mixed models. *J. Comput. Biol.* **8**(6), 625–637.

Zhu J and Hastie T 2004 Classification of gene microarrays by penalized logistic regression.. *Biostatistics* **5**(3), 427–443.

Zuur A, Ieno E, Walker N, Saveliev A and Smith G 2009 *Mixed Effects Models and Extensions in Ecology with R*. Springer.