4. Maximum Likelihood Estimation

Maximum Likelihood Estimation

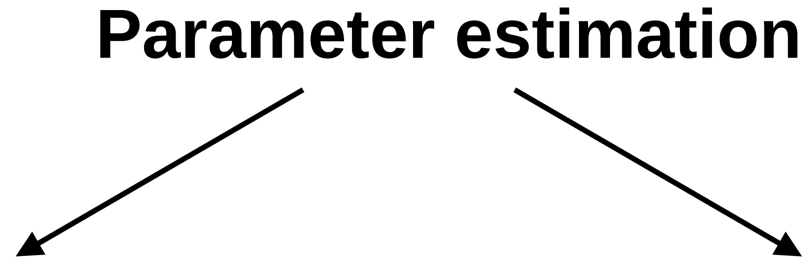
- Data availability in a Bayesian framework
- We could design an optimal classifier if we knew:
    - $P(\omega_i)$ (priors)
    - $P(x \mid \omega_i)$ (class-conditional densities)
    - Unfortunately, we rarely have this complete information.

    - Design a classifier from a training sample
    - No problem with prior estimation
    - Samples are often too small for class-conditional estimation (large dimension of feature space)

Maximum Likelihood Estimation

- A priori information about the problem
  - Normality of $P(x \mid \omega_i)$
  - $P(x \mid \omega_i) \sim N(\mu_i, \Sigma_i)$
  - Characterized by 2 parameters
- Estimation techniques
  - Maximum-Likelihood (ML) and Bayesian estimations
- Results are nearly identical, but the approaches are different

Parameter Estimation



Parameter estimation

Maximum likelihood: values of parameters are fixed but unknown

Bayesian estimation: parameters as random variables having some known a priori distribution

Maximum Likelihood Estimation

- Parameters in ML estimation are fixed but unknown

- Best parameters are obtained by maximizing the probability of obtaining the samples observed

- Here, we use P(ω_i | x) for our classification rule

Maximum Likelihood Estimation

ML Estimation:

- Has good convergence properties as the sample size increases
- Simpler than any other alternative techniques

- General principle in a specific example
- Assume we have c classes and
  - $p(\mathbf{x}|\omega_j) \sim N(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$
  - $p(\mathbf{x}|\omega_j) \equiv p(\mathbf{x}|\omega_j, \boldsymbol{\theta}_j)$ where:

$$\boldsymbol{\theta}_j = (\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) = (\mu_j^1, \mu_j^2, ..., \sigma_j^{11}, \sigma_j^{22}, cov(x_j^m, x_j^n)...)$$

Maximum Likelihood Estimation

•Use the information provided by the training samples to estimate $\theta = (\theta_1, \theta_2, \ldots, \theta_c)$ each θ_i (i = 1, 2, …, c) is associated with each category

• c separate problems: Use a set $\mathcal{D}$ of n training samples $x_1, x_2, \ldots, x_n$ drawn independently from $p(\mathbf{x}|\boldsymbol{\theta})$ to estimate the unknown θ

$$p(\mathcal{D}|\boldsymbol{\theta}) = \prod_{k=1}^{n} p(\mathbf{x}_k|\boldsymbol{\theta})$$

$p(\mathcal{D}|\boldsymbol{\theta})$ is called the likelihood of θ w.r.t. the set of samples

Maximum Likelihood Estimation

$$p(\mathcal{D}|\boldsymbol{\theta}) = \prod_{k=1}^{n} p(\mathbf{x}_k|\boldsymbol{\theta})$$

$p(\mathcal{D}|\boldsymbol{\theta})$ is called the ***likelihood*** of θ w.r.t. the set of samples

- ***ML*** estimate of θ is, by definition the value $\widehat{\theta}$ that maximizes $p(\mathcal{D}|\boldsymbol{\theta})$

- **“It is the value of θ that best agrees with the actually observed training samples”**

Maximum Likelihood Estimation

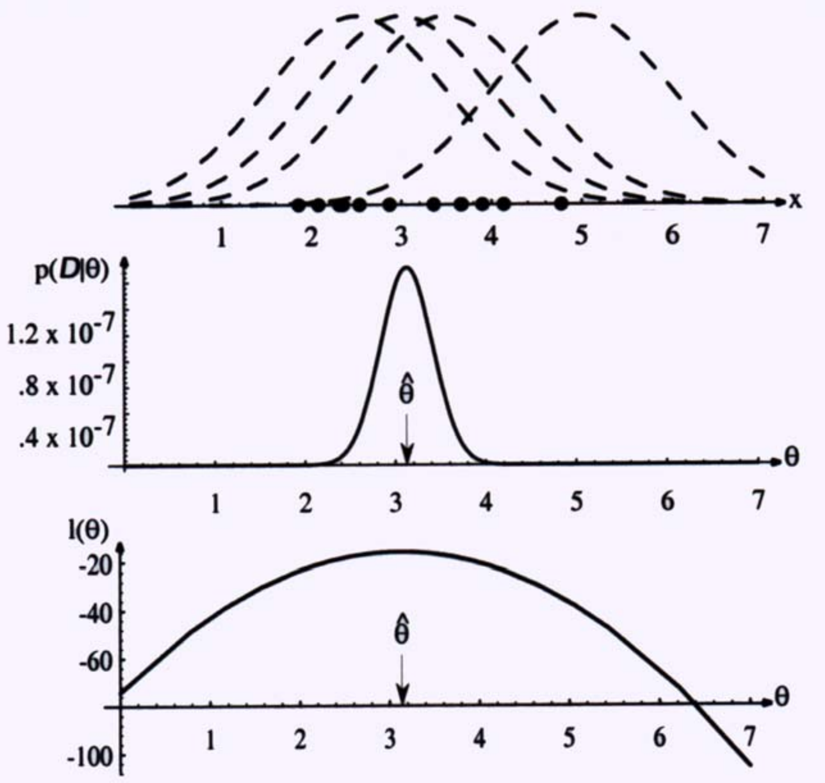


Figure 3.1: The top graph shows several training points in one dimension, known or assumed to be drawn from a Gaussian of a particular variance, but unknown mean. Four of the infinite number of candidate source distributions are shown in dashed lines. The middle figures shows the likelihood $p(\mathcal{D}|\theta)$ as a function of the mean. If we had a very large number of training points, this likelihood would be very narrow. The value that maximizes the likelihood is marked $\hat{\theta}$; it also maximizes the logarithm of the likelihood — i.e., the log-likelihood $l(\theta)$, shown at the bottom. Note especially that the likelihood lies in a different space from $p(x|\hat{\theta})$, and the two can have different functional forms.

Maximum Likelihood Estimation

- Optimal estimation
- Let $\boldsymbol{\theta} = (\theta_1, \theta_2, ..., \theta_p)^T$ and let $\nabla_{\boldsymbol{\theta}}$ be the gradient operator

$$\nabla_{\boldsymbol{\theta}} = \left[\frac{\partial}{\partial\theta_1}, \frac{\partial}{\partial\theta_2}, ..., \frac{\partial}{\partial\theta_p}\right]^T$$

- We define $l(\boldsymbol{\theta})$ as the *log-likelihood* function:

$$l(\boldsymbol{\theta}) \equiv \ln p(\mathcal{D}|\boldsymbol{\theta})$$

Maximum Likelihood Estimation

- New problem statement: determine θ that maximizes the log-likelihood:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} l(\boldsymbol{\theta})$$

- Set of necessary conditions for an optimum is:

$$\nabla_{\boldsymbol{\theta}} l = \sum_{k=1}^{n} \nabla_{\boldsymbol{\theta}} \ln p(\mathbf{x}_k|\boldsymbol{\theta})$$

$$\boxed{\nabla_{\boldsymbol{\theta}} l = 0}$$

Maximum Likelihood Estimation

- Example of a specific case: unknown μ
  - P(x_i | μ) ~ N(μ, Σ) (Samples are drawn from a multivariate normal population)

$$\ln p(\mathbf{x}_k|\boldsymbol{\mu}) = -\frac{1}{2}\ln\left[(2\pi)^d|\boldsymbol{\Sigma}|\right] - \frac{1}{2}(\mathbf{x}_k - \boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}_k - \boldsymbol{\mu})$$

$$\nabla_{\boldsymbol{\theta}} \ln p(\mathbf{x}_k|\boldsymbol{\mu}) = \Sigma^{-1}(\mathbf{x}_k - \boldsymbol{\mu})$$

θ = μ therefore:

- The ML estimate for μ must satisfy:

$$\sum_{k=1}^{n} \boldsymbol{\Sigma}^{-1}(\mathbf{x}_k - \hat{\boldsymbol{\mu}}) = 0$$

Maximum Likelihood Estimation

Multiplying by Σ and rearranging, we obtain:

$$\hat{\mu} = \frac{1}{n} \sum_{k=1}^{n} \mathbf{x}_k$$

(Just the arithmetic average of the samples of the training samples)

Conclusion: "If $p(\mathbf{x}_k | \omega_j)\ (j = 1, 2, ..., c)$ is supposed to be Gaussian in a d dimensional feature space; then we can estimate $\theta = (\theta_1, \theta_2, \ldots, \theta_c)$ and perform an optimal classification"

Maximum Likelihood Estimation

- **Gaussian Case:** ***unknown*** μ ***and*** σ

$$\boldsymbol{\theta} = (\theta_1, \theta_2) = (\mu, \sigma^2)$$

$$l = \ln p(\mathbf{x}_k|\boldsymbol{\theta}) = -\frac{1}{2}\ln 2\pi\theta_2 - \frac{1}{2\theta_2}(x_k - \theta_1)^2$$

$$\nabla_\theta l = \left[\begin{array}{c} \frac{\partial}{\partial\theta_1}(\ln p(x_k|\theta)) \\ \frac{\partial}{\partial\theta_2}(\ln p(x_k|\theta)) \end{array} \right] = 0$$

$$\begin{cases} \sum_{k=1}^{n} \frac{1}{\hat{\theta_2}}(x_k - \hat{\theta_1}) = 0 \\ -\sum_{k=1}^{n} \frac{1}{\hat{\theta_2}} + \sum_{k=1}^{n} \frac{(x_k - \hat{\theta_1})^2}{\hat{\theta_2}^2} = 0 \end{cases}$$

Maximum Likelihood Estimation

$$\begin{cases} \sum_{k=1}^{n} \frac{1}{\widehat{\theta_2}}(x_k - \widehat{\theta_1}) = 0 \\ -\sum_{k=1}^{n} \frac{1}{\widehat{\theta_2}} + \sum_{k=1}^{n} \frac{(x_k - \widehat{\theta_1})^2}{\widehat{\theta_2}^2} = 0 \end{cases}$$

$$\Downarrow$$

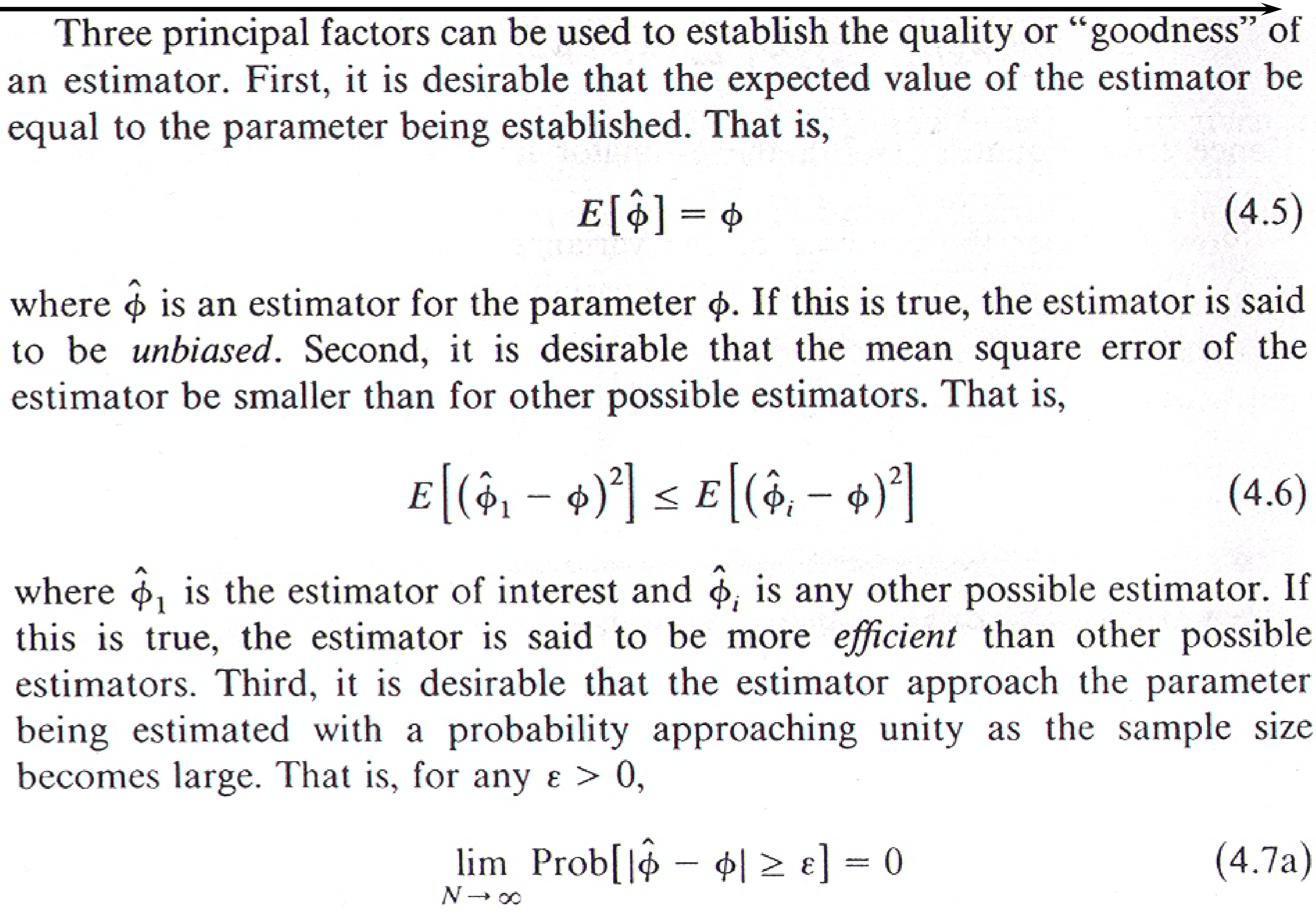
$$\hat{\mu} = \frac{1}{n}\sum_{k=1}^{n} x_k \qquad \hat{\sigma}^2 = \frac{1}{n}\sum_{k=1}^{n} (x_k - \hat{\mu})^2$$

Quality of Estimators

Three Principal Factors can be used to obtain the quality of estimators:

- Bias
- Consistency
- Efficiency

Quality of Estimators



Three principal factors can be used to establish the quality or "goodness" of an estimator. First, it is desirable that the expected value of the estimator be equal to the parameter being established. That is,

$$E[\hat{\phi}] = \phi \qquad (4.5)$$

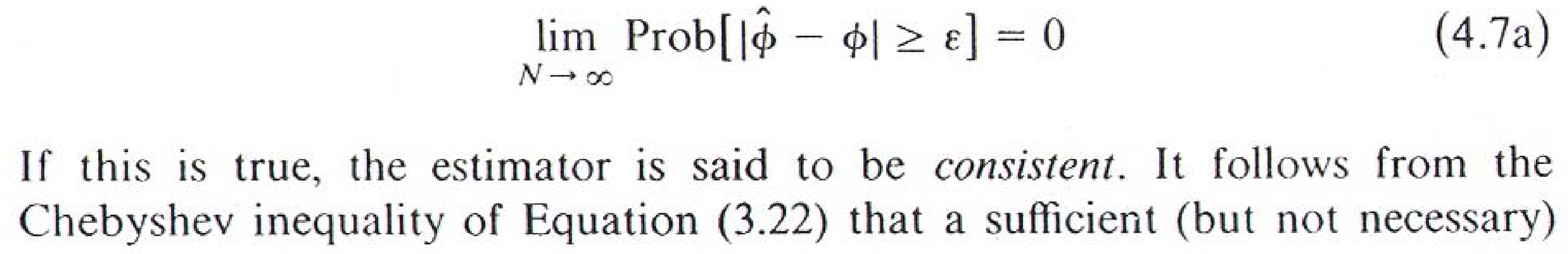
where $\hat{\phi}$ is an estimator for the parameter ϕ. If this is true, the estimator is said to be *unbiased*. Second, it is desirable that the mean square error of the estimator be smaller than for other possible estimators. That is,

$$E\left[(\hat{\phi}_1 - \phi)^2\right] \le E\left[(\hat{\phi}_i - \phi)^2\right] \qquad (4.6)$$

where $\hat{\phi}_1$ is the estimator of interest and $\hat{\phi}_i$ is any other possible estimator. If this is true, the estimator is said to be more *efficient* than other possible estimators. Third, it is desirable that the estimator approach the parameter being estimated with a probability approaching unity as the sample size becomes large. That is, for any $\varepsilon > 0$,

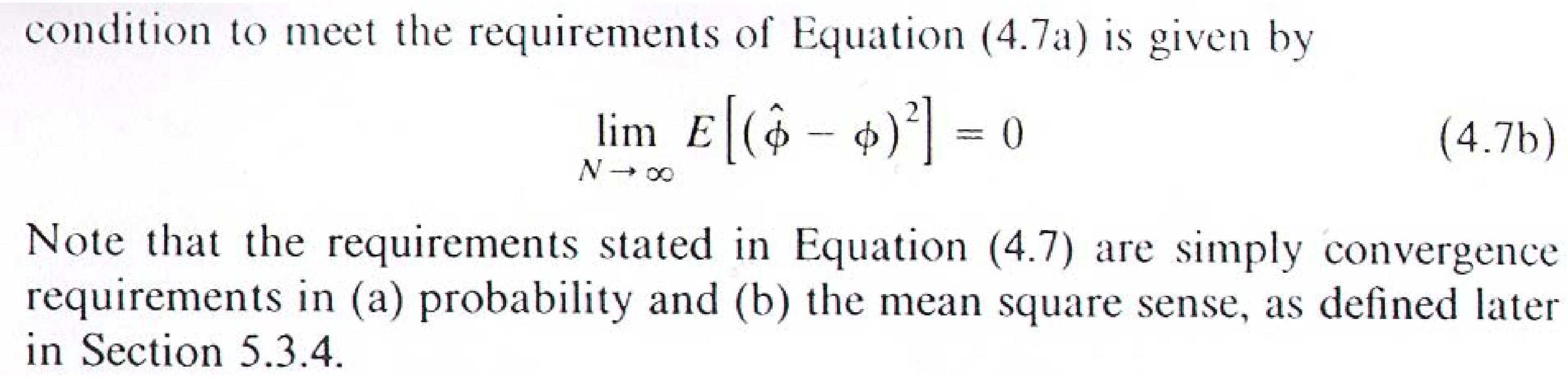
$$\lim_{N \to \infty} \text{Prob}[|\hat{\phi} - \phi| \ge \varepsilon] = 0 \qquad (4.7a)$$

Quality of Estimators



$$\lim_{N\to\infty} \text{Prob}[|\hat{\phi} - \phi| \geq \varepsilon] = 0 \tag{4.7a}$$

If this is true, the estimator is said to be *consistent*. It follows from the Chebyshev inequality of Equation (3.22) that a sufficient (but not necessary)



condition to meet the requirements of Equation (4.7a) is given by

$$\lim_{N\to\infty} E\left[(\hat{\phi} - \phi)^2\right] = 0 \tag{4.7b}$$

Note that the requirements stated in Equation (4.7) are simply convergence requirements in (a) probability and (b) the mean square sense, as defined later in Section 5.3.4.

Maximum Likelihood Estimation (continued)

Bias

- ML estimate for σ^2 is biased

$$E\left[\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^2\right]=\frac{n-1}{n}\sigma^2\neq\sigma^2$$

- An elementary unbiased estimator for σ^2 :

$$E\left[\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\bar{x})^2\right]=\sigma^2$$

Maximum Likelihood Estimation (continued)

- An elementary unbiased estimator for σ^2 :

$$E\left[\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\bar{x})^2\right]=\sigma^2$$

- Sample covariance matrix:

$$\mathbf{C}=\frac{1}{n-1}\sum_{k=1}^{n}(\mathbf{x}_k-\hat{\boldsymbol{\mu}})(\mathbf{x}_k-\hat{\boldsymbol{\mu}})^T$$

Maximum Likelihood Estimation (continued)

Key property of ML:

- If an estimator is unbiased and ML then it is also efficient

Density Function Params. via Sample

Gaussian density function:

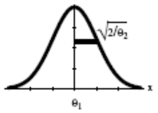
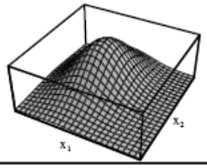
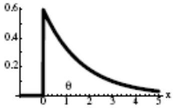
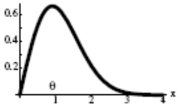
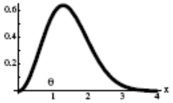
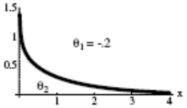
$$\boxed{p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}}$$

where μ and σ² are estimated from sample (via maximum likelihood estimate):

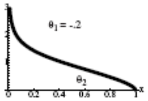
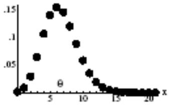
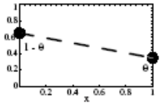
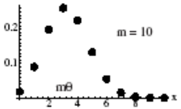
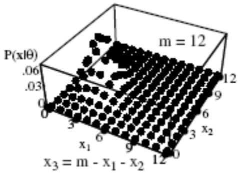
$$\boxed{\mu = \frac{1}{n}\sum_i x_i}$$

$$\boxed{\sigma^2 = \frac{1}{n}\sum_i (x_i - \mu)^2}$$

Common Exponential Distributions

Name	Distribution	Domain		s
Normal	$p(x\|\boldsymbol{\theta}) = \sqrt{\frac{\theta_2}{2\pi}} e^{-(1/2)\theta_2(x-\theta_1)^2}$	$\theta_2 > 0$		$\begin{bmatrix} \frac{1}{n}\sum_{k=1}^{n} x_k \\ \frac{1}{n}\sum_{k=1}^{n} x_k^2 \end{bmatrix}$
Multi-variate Normal	$p(\mathbf{x}\|\boldsymbol{\theta}) = \frac{\|\boldsymbol{\Theta}_2\|^{1/2}}{(2\pi)^{d/2}} e^{-(1/2)(\mathbf{x}-\boldsymbol{\theta}_1)^t \boldsymbol{\Theta}_2 (\mathbf{x}-\boldsymbol{\theta}_1)}$	$\boldsymbol{\Theta}_2$ positive definite		$\begin{bmatrix} \frac{1}{n}\sum_{k=1}^{n} \mathbf{x}_k \\ \frac{1}{n}\sum_{k=1}^{n} \mathbf{x}_k\mathbf{x}_k^t \end{bmatrix}$
Exponential	$p(x\|\theta) = \begin{cases} \theta e^{-\theta x} & x \geq 0 \\ 0 & \text{otherwise} \end{cases}$	$\theta > 0$		$\frac{1}{n}\sum_{k=1}^{n} x_k$
Rayleigh	$p(x\|\theta) = \begin{cases} 2\theta x e^{-\theta x^2} & x \geq 0 \\ 0 & \text{otherwise} \end{cases}$	$\theta > 0$		$\frac{1}{n}\sum_{k=1}^{n} x_k^2$
Maxwell	$p(x\|\theta) = \begin{cases} \frac{4}{\sqrt{\pi}}\theta^{3/2}x^2 e^{-\theta x^2} & x \geq 0 \\ 0 & \text{otherwise} \end{cases}$	$\theta > 0$		$\frac{1}{n}\sum_{k=1}^{n} x_k^2$
Gamma	$p(x\|\boldsymbol{\theta}) = \begin{cases} \frac{\theta_2^{\theta_1+1}}{\Gamma(\theta_1+1)} x^{\theta_1} e^{-\theta_2 x} & x \geq 0 \\ 0 & \text{otherwise} \end{cases}$	$\theta_1 > -1$ $\theta_2 > 0$		$\begin{bmatrix} \left(\prod_{k=1}^{n} x_k\right)^{1/n} \\ \frac{1}{n}\sum_{k=1}^{n} x_k \end{bmatrix}$

Common Exponential Distributions

Beta	$p(x\|\boldsymbol{\theta}) = \begin{cases} \frac{\Gamma(\theta_1+\theta_2+2)}{\Gamma(\theta_1+1)\Gamma(\theta_2+1)} x^{\theta_1}(1-x)^{\theta_2} \\ \qquad 0 \le x \le 1 \\ 0 \qquad \text{otherwise} \end{cases}$	$\theta_1 > -1$ $\theta_2 > -1$		$\begin{bmatrix} \left(\prod_{k=1}^{n} x_k\right)^{1/n} \\ \left(\prod_{k=1}^{n} (1-x_k)\right)^{1/n} \end{bmatrix}$
Poisson	$P(x\|\theta) = \frac{\theta^x}{x!}e^{-\theta} \quad x = 0, 1, 2, \ldots$	$\theta > 0$		$\frac{1}{n}\sum_{k=1}^{n} x_k$
Bernoulli	$P(x\|\theta) = \theta^x(1-\theta)^{1-x} \quad x = 0, 1$	$0 < \theta < 1$		$\frac{1}{n}\sum_{k=1}^{n} x_k$
Binomial	$P(x\|\theta) = \frac{m!}{x!(m-x)!}\theta^x(1-\theta)^{m-x}$ $x = 0, 1, \ldots, m$	$0 < \theta < 1$		$\frac{1}{n}\sum_{k=1}^{n} x_k$
Multinomial	$P(\mathbf{x}\|\boldsymbol{\theta}) = \frac{m! \prod_{i=1}^{d} \theta_i^{x_i}}{\prod_{i=1}^{d} x_i!}$	$x_i = 0, 1, \ldots, m$ $\sum_{i=1}^{d} x_i = m$	$0 < \theta_i < 1$ $\sum_{i=1}^{d} \theta_i = 1$		$\frac{1}{n}\sum_{k=1}^{n} \mathbf{x}_k$

Example with real world data

- **Classification of remote sensing hyperspectral image using maximum likelihood technique**

Maximum Likelihood Classification

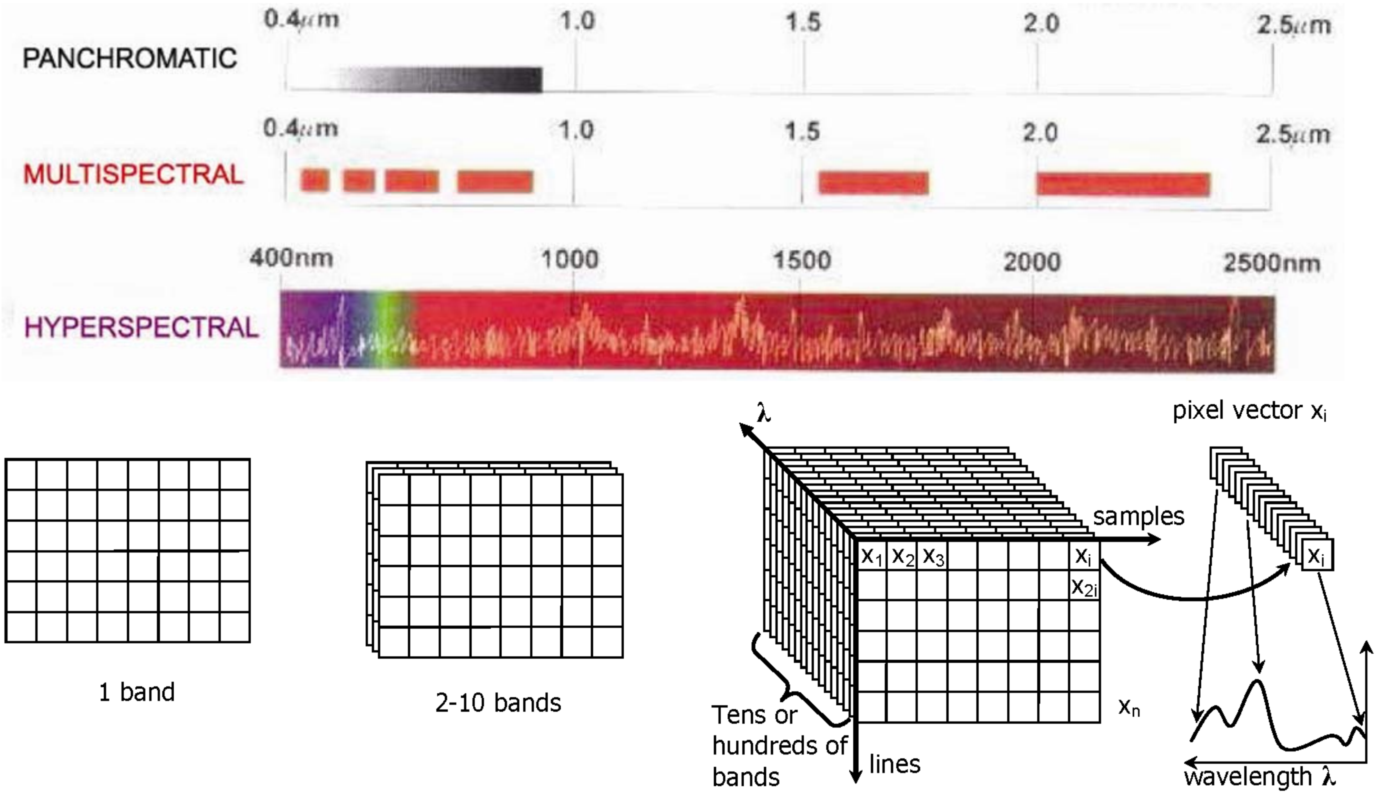
- Image is acquired by the ROSIS-03 optical sensor over the University of Pavia, Italy

- Spatial dimension: 610 x 340 pixels

- Spatial resolution: 1.3m per pixel

- Spectral dimension: 103 spectral channels (0.43-0.86 μm)

Spectral context

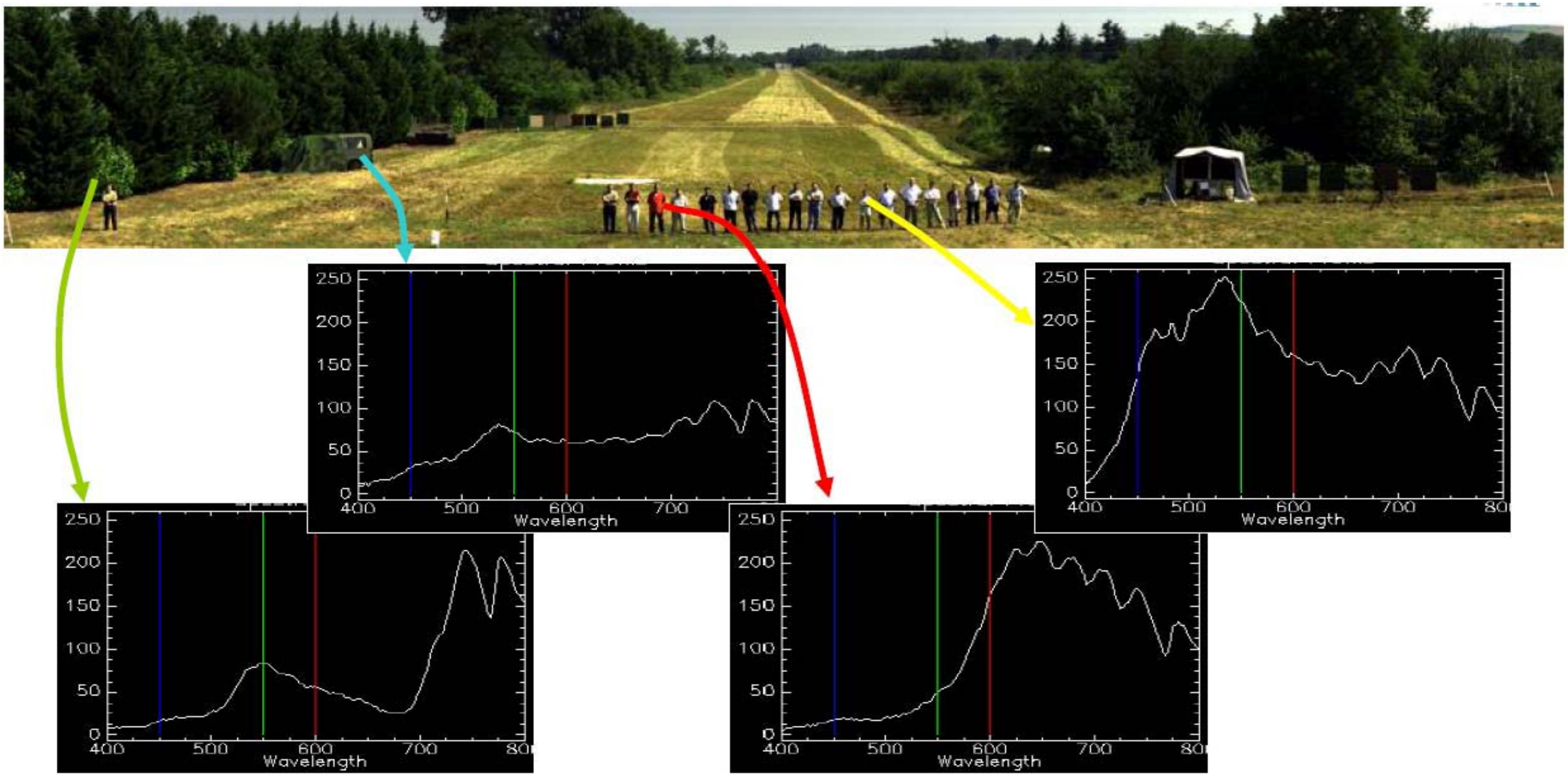


Panchromatic: one grey level value per pixel

Multispectral: limited spectral info

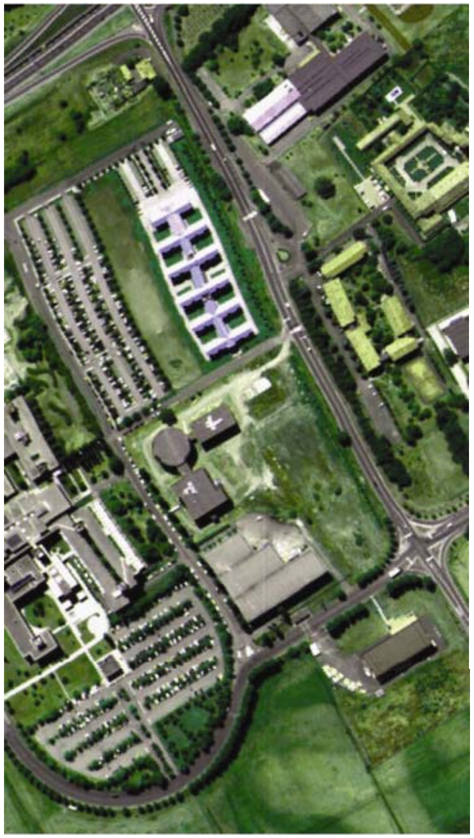
Hyperspectral: detailed spectral info

Spectral context



Maximum Likelihood Classification

Input image (103 spectral channels)



Task:

Assign *every* pixel to *one* of the *nine* classes:

alphalt

meadows

gravel

trees

metal sheets

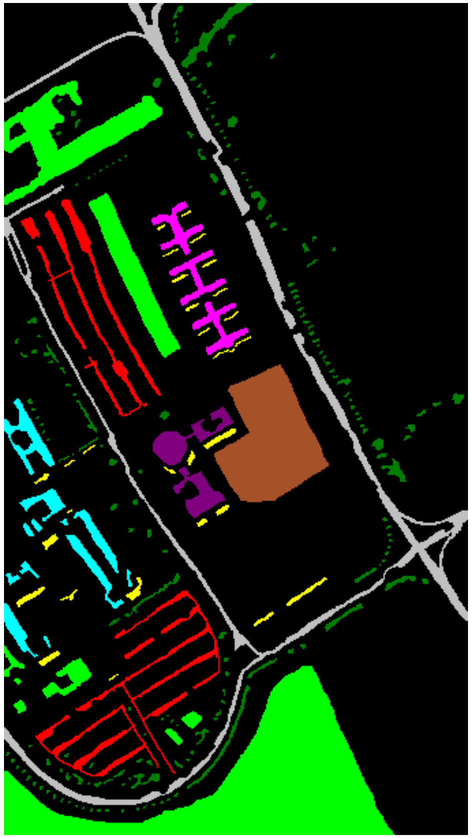
bare soil

bitumen

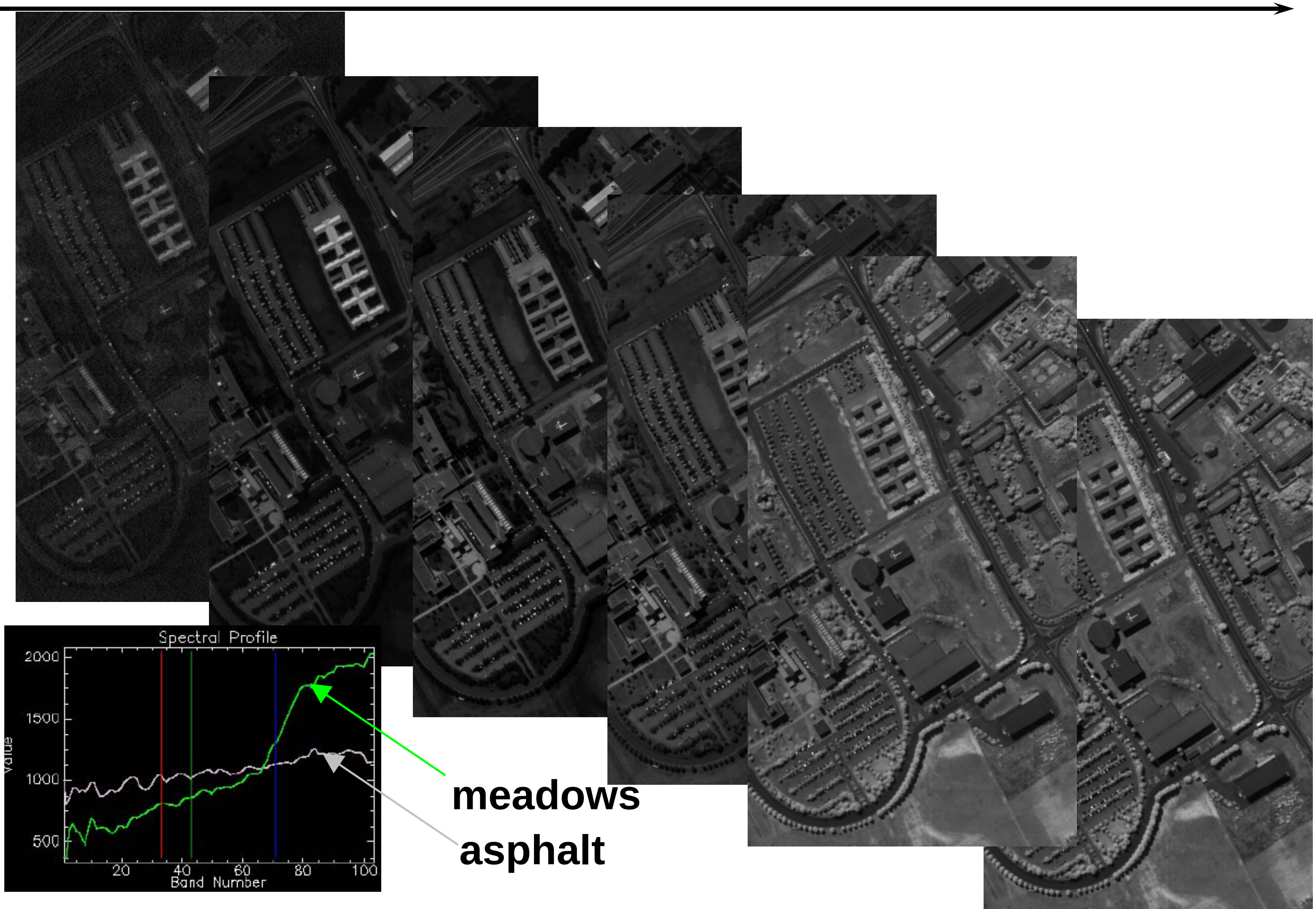
bricks

shadows

Reference data



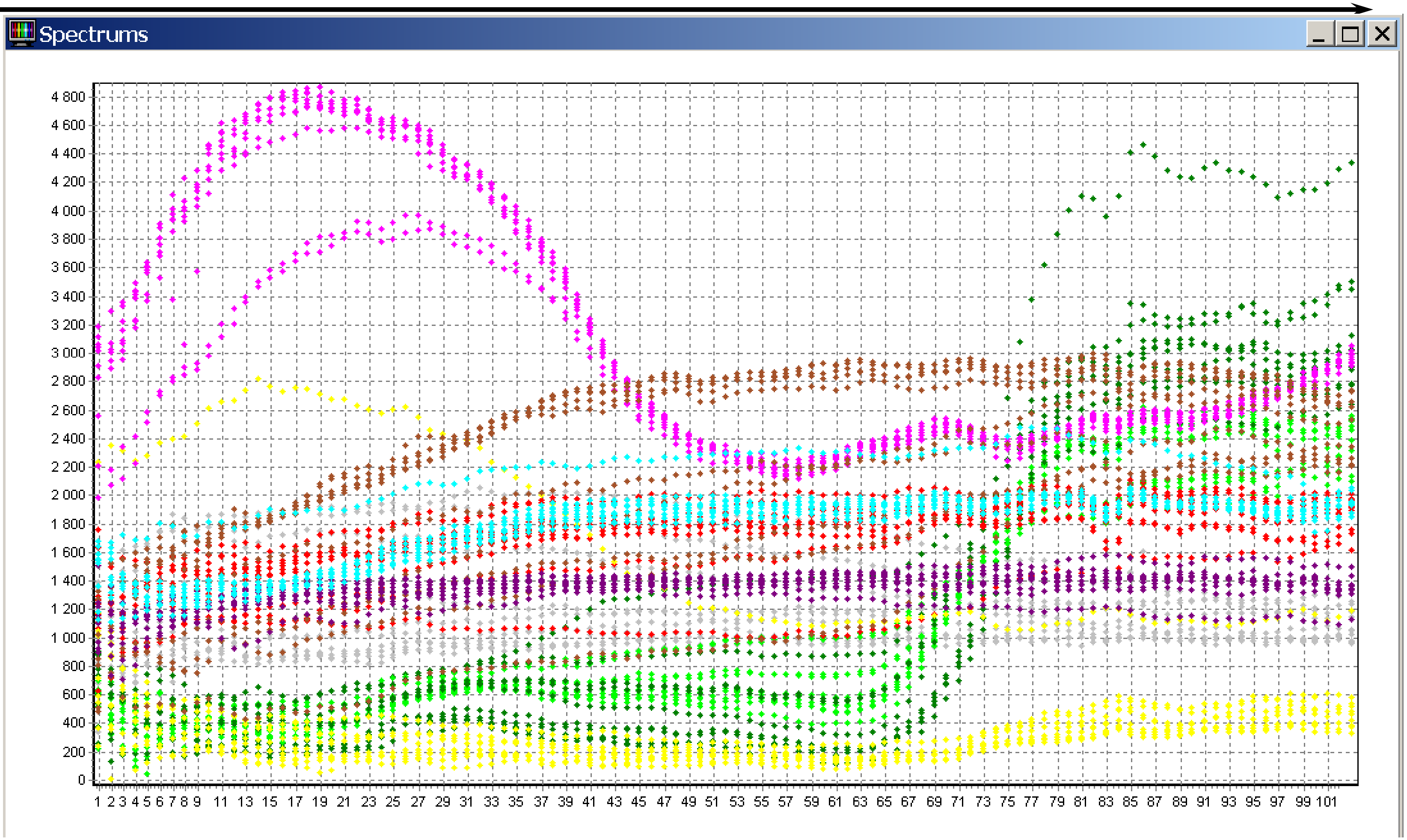
Spectral Context for HS Image



meadows

asphalt

Spectral Context for HS Image



alphalt, meadows, gravel, trees, metal sheets,
bare soil, bitumen, bricks, shadows

Maximum Likelihood Classification

- *Feature vector:* a vector of radiance values x for each pixel

103 spectral bands → dimensionality of the feature vector d=103

Maximum Likelihood Classification

- Samples of each class *k* are assumed to have a Gaussian distribution
- Parameters of distributions for each class are estimated from the training samples, using the maximum likelihood estimates:

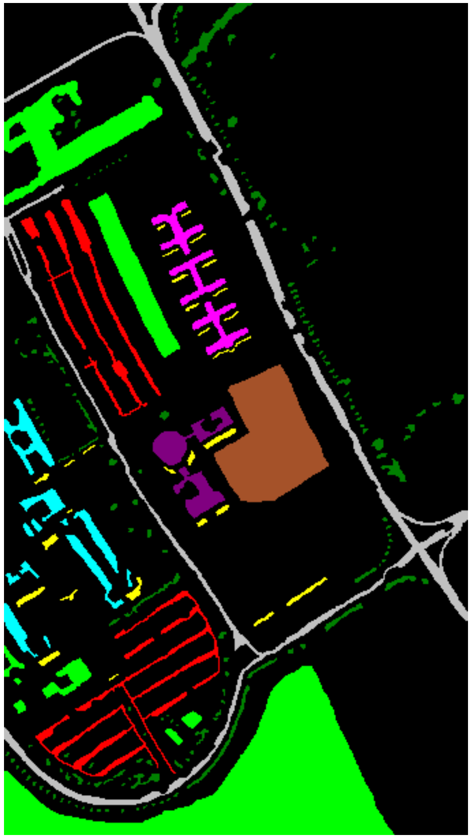
$$\boldsymbol{\mu}_k = \frac{1}{m_k} \sum_{j=1}^{m_k} \mathbf{x}_{j,k}$$

$$\Sigma_k = \frac{1}{m_k} \sum_{j=1}^{m_k} (\mathbf{x}_{j,k} - \boldsymbol{\mu}_k)(\mathbf{x}_{j,k} - \boldsymbol{\mu}_k)^T,$$

where $\mathbf{x}_{j,k}, j = 1, ..., m_k$ - training samples for class k.

Maximum Likelihood Classification

- We split reference data into sets of training and test samples:



Class	Training samples	Test samples
Asphalt	**548**	**6304**
Meadows	**540**	**18146**
Gravel	**392**	**1815**
Trees	**524**	**2912**
Metal sheets	**265**	**1113**
Bare soil	**532**	**4572**
Bitumen	**375**	**981**
Bricks	**514**	**3364**
Shadows	**231**	**795**

Maximum Likelihood Classification

- For each class *k*, P = [d(d+1)/2 + d] parameters have to be estimated
- If d = 103, P = 5459!
- We have only from 231 to 548 training samples per class
- To avoid a significant parameter estimation error: P << m_k (m_k – number of training samples for class k)

Maximum Likelihood Classification

- **Dimensionality reduction must be performed first, to reduce the dimensionality d**
  - **The first 3 bands on the 103-band image are omitted**
  - **A 10-band image is obtained by averaging over every 10 bands (new d = 10)**

Maximum Likelihood Classification

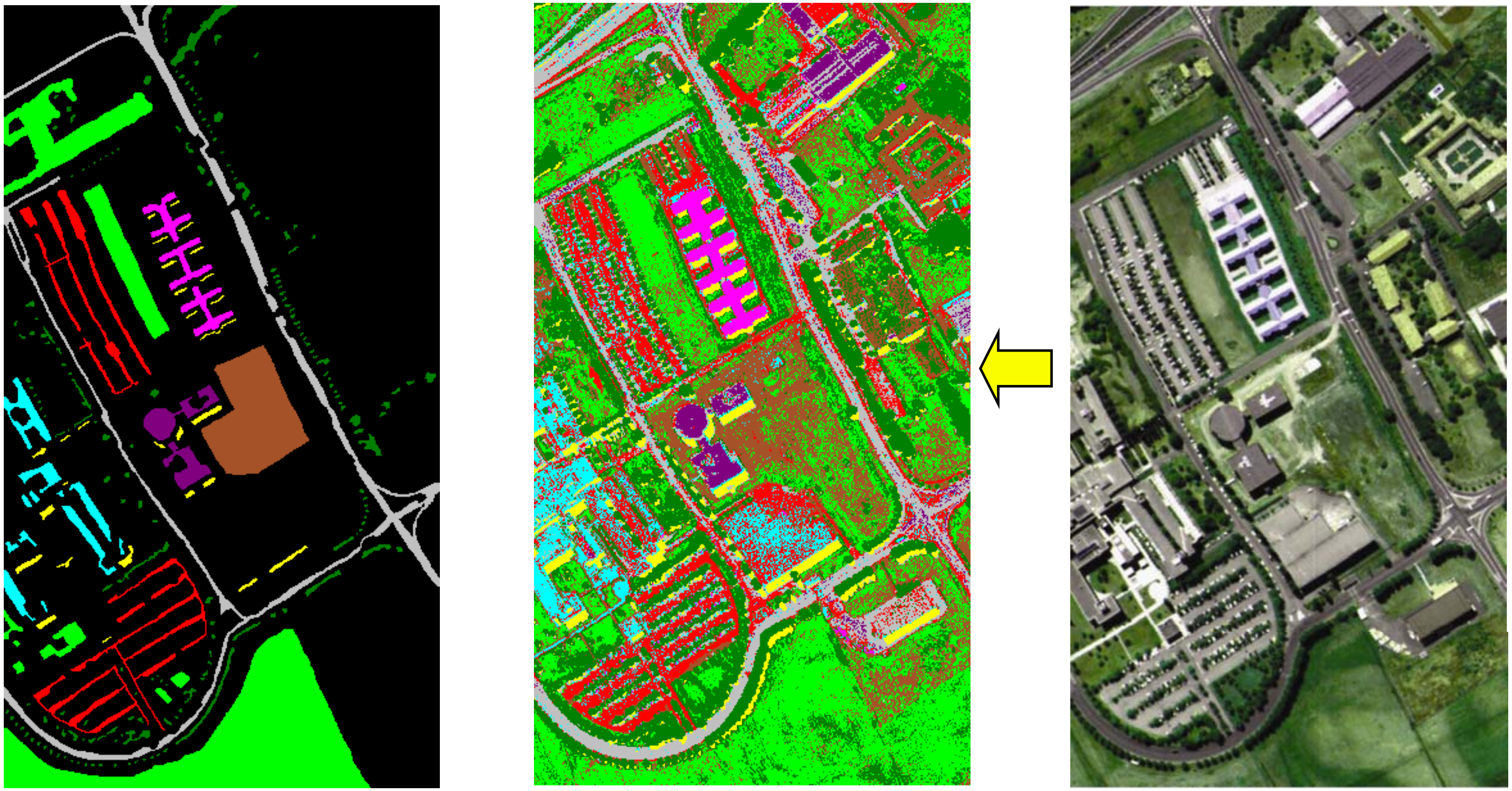
1) Parameters of Gaussian distributions for each class are estimated

2) The whole image is classified using K = 9 (number of classes) discriminant functions (MAP classification):

$$g_k(\mathbf{x}) = -\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_k)^T\Sigma_k^{-1}(\mathbf{x}-\boldsymbol{\mu}_k)-\frac{1}{2}\ln|\Sigma_k|+\ln P(\omega_k)$$

$P(\omega_k) = \frac{m_k}{m}, m$ - total number of training samples

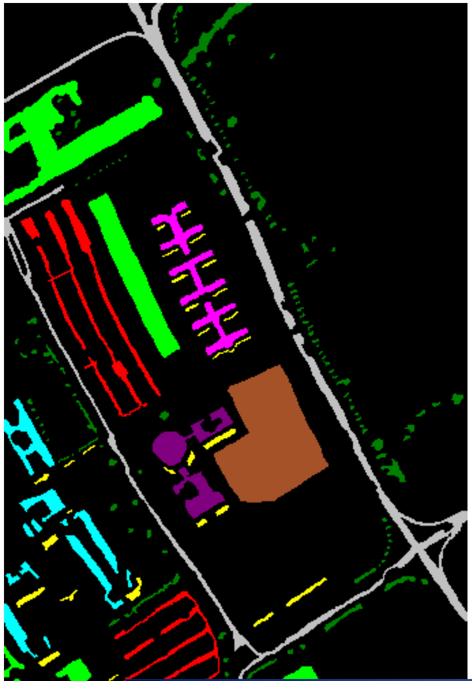
Maximum Likelihood Classification

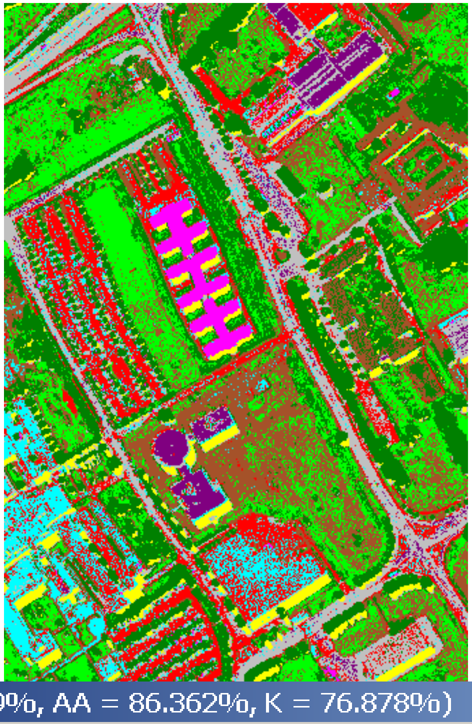


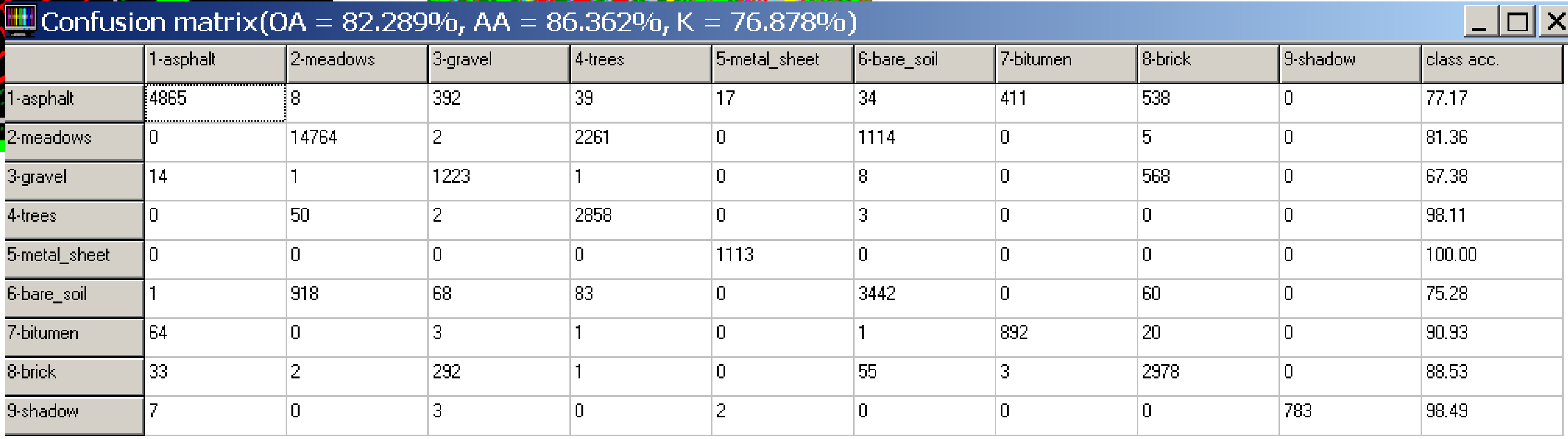
alphalt, meadows, gravel, trees, metal sheets,
bare soil, bitumen, bricks, shadows

Overall accuracy = 82.29%

Maximum Likelihood Classification







Confusion matrix(OA = 82.289%, AA = 86.362%, K = 76.878%)

	1-asphalt	2-meadows	3-gravel	4-trees	5-metal_sheet	6-bare_soil	7-bitumen	8-brick	9-shadow	class acc.
1-asphalt	4865	8	392	39	17	34	411	538	0	77.17
2-meadows	0	14764	2	2261	0	1114	0	5	0	81.36
3-gravel	14	1	1223	1	0	8	0	568	0	67.38
4-trees	0	50	2	2858	0	3	0	0	0	98.11
5-metal_sheet	0	0	0	0	1113	0	0	0	0	100.00
6-bare_soil	1	918	68	83	0	3442	0	60	0	75.28
7-bitumen	64	0	3	1	0	1	892	20	0	90.93
8-brick	33	2	292	1	0	55	3	2978	0	88.53
9-shadow	7	0	3	0	2	0	0	0	783	98.49

Maximum Likelihood Classification

***Conclusions* for the classification example:**

- **Classification accuracies are high for most of the classes**

- **Other feature extraction (dimensionality reduction) method can be used → accuracies can be further improved**

Computational complexity

- *Example:* complexity of a ML estimation of the parameters in a classifier for Gaussian priors in d dimension, with n training samples for each of c categories

$$g(\mathbf{x}) = -\frac{1}{2}(\mathbf{x} - \overset{O(dn)}{\overset{\uparrow}{\hat{\mu}}})^t \overbrace{\widehat{\Sigma}^{-1}}^{O(nd^2)} (\mathbf{x} - \hat{\mu}) - \overbrace{\frac{d}{2} \ln 2\pi}^{O(1)} - \overbrace{\frac{1}{2} \ln |\widehat{\Sigma}|}^{O(d^2 n)} + \overbrace{\ln P(\omega)}^{O(n)}$$

- Overall *computational complexity* (CC) for learning is *O(cd²n)*
- CC for classificaiton of one sample is *O(cd²)*

Computational complexity

- **Parallel implementations**
  - **Space complexity**
  - **Time complexity**

- **Example: Estimation of the sample mean using d processors, each adding n values**
  - **Space complexity:** ***O(d)***
  - **Time complexity:** ***O(n)***