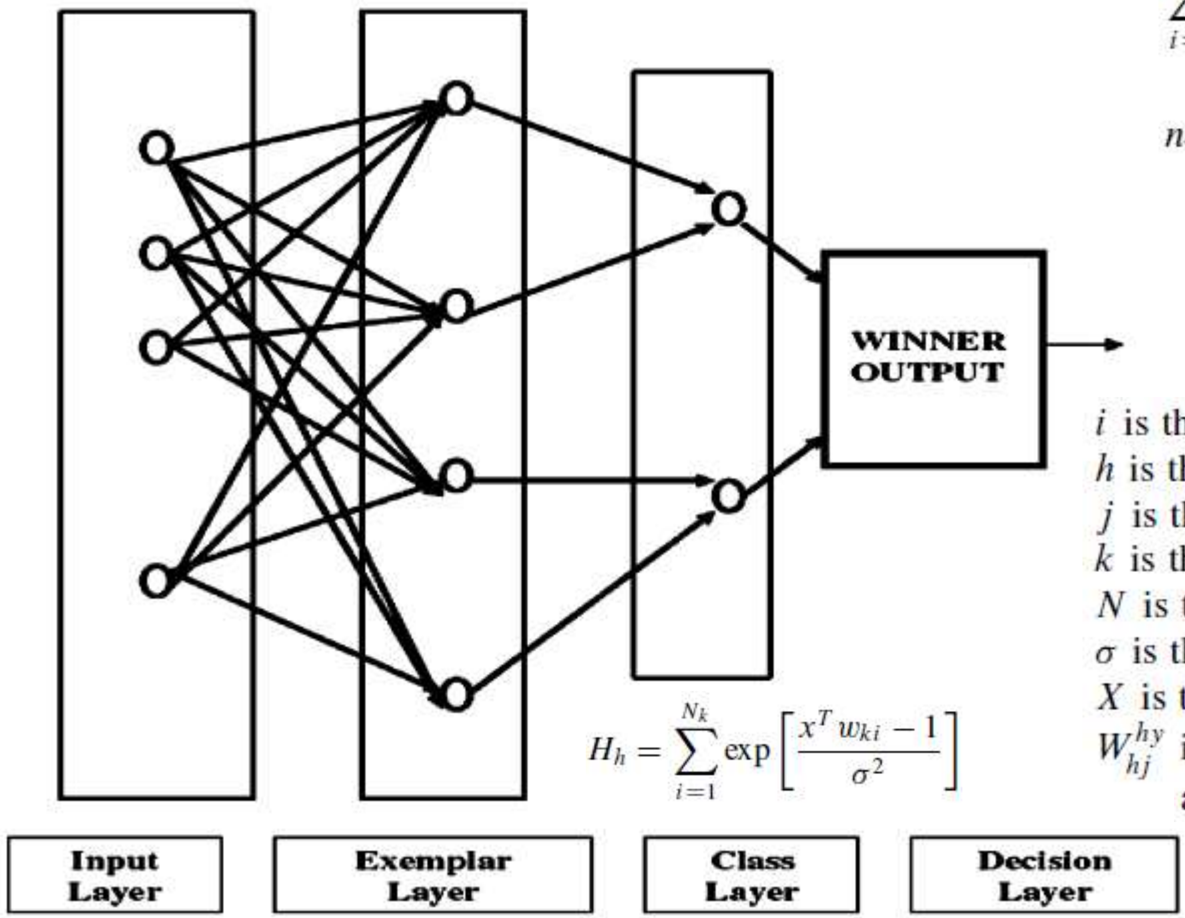
BAYES-PARZEN (PNN)

$$\exp\left[\frac{x^T w_{ki} - 1}{\sigma^2}\right]$$



$$H_h = \sum_{i=1}^{N_k} \exp\left[\frac{x^T w_{ki} - 1}{\sigma^2}\right]$$

$$\sum_{i=1}^{N_k} \exp\left[\frac{x^T w_{ki} - 1}{\sigma^2}\right] > \sum_{i=1}^{N_j} \exp\left[\frac{x^T w_{kj} - 1}{\sigma^2}\right],$$

$$net_j = \frac{1}{N_j} \sum_m W_{hj}^{hy} H_h \quad \text{and} \quad N_j = \sum_h W_{hj}^{hy};$$

$$net_j = \max k(net_k),$$

$$\text{then} \quad y_j = 1, \qquad \text{else} \quad y_j = 0,$$

i is the number of input layers,
h is the number of hidden layers,
j is the number of output layers,
k is the number of training examples,
N is the number of classifications (clusters),
σ is the smoothing parameter (standard deviation),
X is the input vector, and
W_{hj}^{hy} is the connection weight between the hidden layer H and the output layer Y

Consider the two-category situation in which the state of nature θ is known to be either θ_A or θ_B. If it is desired to decide whether $\theta=\theta_A$ or $\theta=\theta_B$ based on a set of measurements represented by the p-dimensional vector $X^t = [\ X_1 \ldots X_i \ldots X_p]$, the Bayes decision rule becomes

$$\begin{aligned} d(X) &= \theta_A \text{ if } h_A l_A f_A(X) > h_B l_B f_B(X) \\ d(X) &= \theta_B \text{ if } h_A l_A f_A(X) < h_B l_B f_B(X) \end{aligned} \qquad (1)$$

where $f_A(X)$ and $f_B(X)$ are the probability density functions for categories θ_A and θ_B respectively, l_A is the loss function associated with the decision $d(X) = \theta_B$ when $\theta=\theta_A$, l_B is the loss associated with the decision $d(X) = \theta_A$ when $\theta=\theta_B$ (the losses associated with correct decisions are taken to be equal to zero), h_A is the a priori probability of occurrence of patterns from category θ_A, and $h_B = 1-h_A$ is the a priori probability that $\theta=\theta_B$. Thus the boundary between the region in which the Bayes decision $d(X)=\theta_A$ and the region in which $d(X)=\theta_B$ is given by the equation

$$f_A(X) = K\ f_B(X) \qquad (2)$$

where

$$K = h_B l_B \ /\ h_A l_A\ . \qquad (3)$$

The key to using eq. 2 is the ability to estimate PDFs based on training patterns. Often the a priori probabilities are known or can be estimated accurately, and the loss functions require subjective evaluation. However, if the probability densities of the patterns in the categories to be separated are unknown, and all that is given is a set of training patterns (training samples), then it is these samples which provide the only clue to the unknown underlying probability densities.

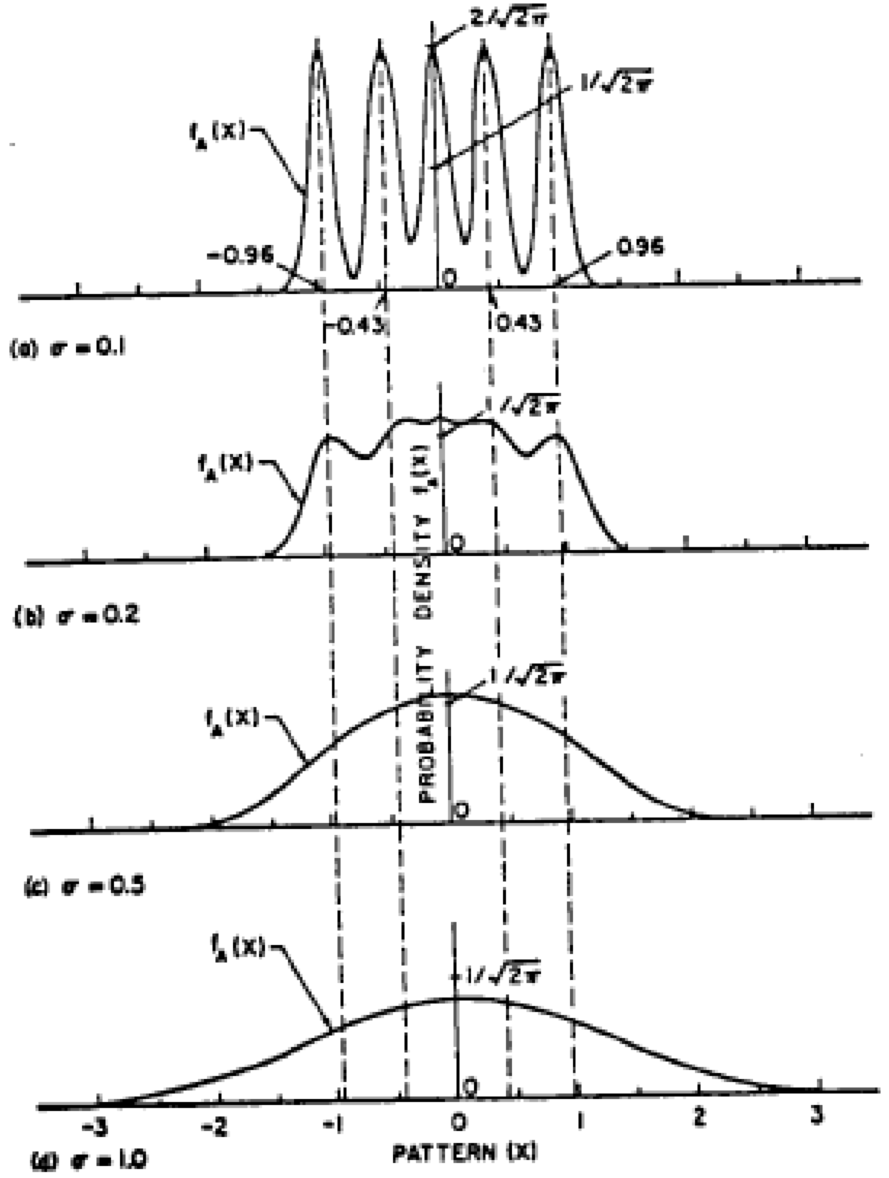
In his classic paper, Parzen [3] showed that a class of PDF estimators asymptotically approach the underlying parent density provided that it is smooth and continuous. The particular estimator used in this study is:

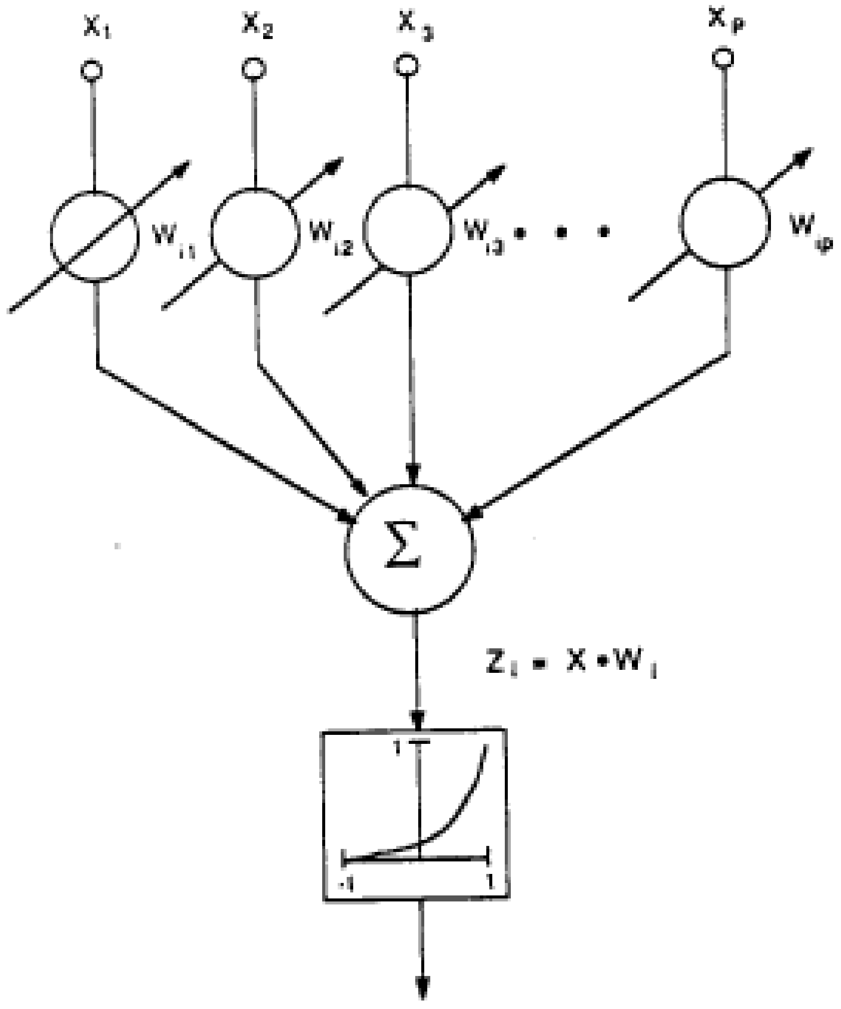
$$f_A(X) = \frac{1}{(2\pi)^{p/2}\sigma^p} \frac{1}{m} \sum_{i=1}^{m} \exp\left[- \frac{(X - X_{ai})^t (X - X_{ai})}{2\sigma^2} \right], \quad (4)$$

where i = pattern number,

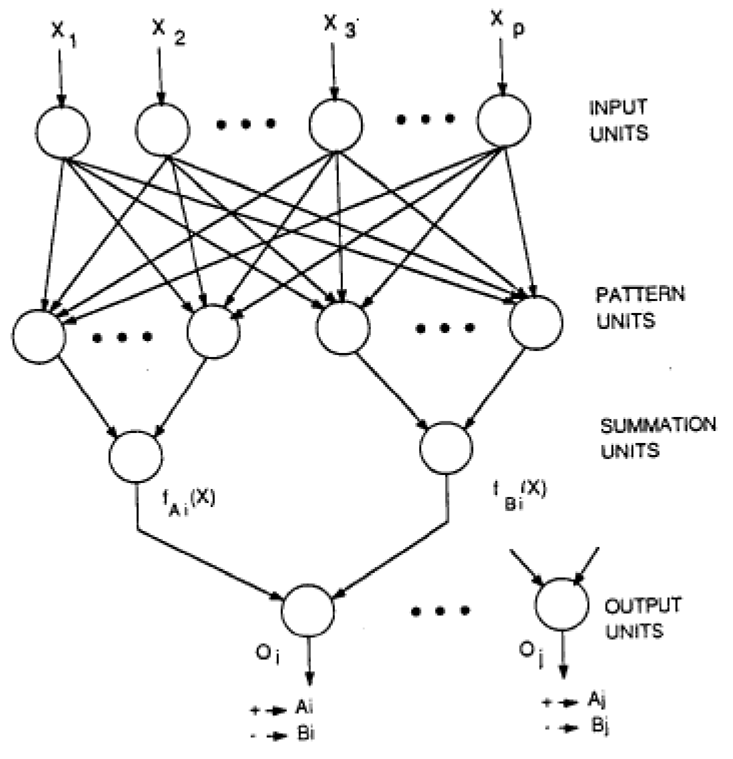
X_{ai} = ith training pattern from category θ_A, and

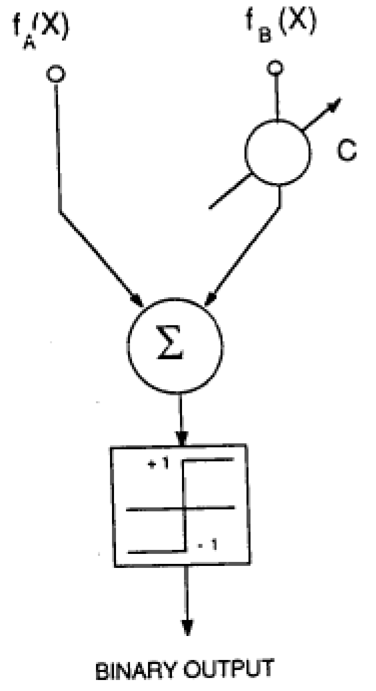
σ is a "smoothing parameter".





$$g(Z_i) = \exp[(Z_i - 1)/\sigma^2]$$





In Figure 2 the input units are merely distribution units which supply the same voltage values to all of the pattern units. The pattern units (shown in more detail in Fig. 3) each form a dot product of the input pattern vector **X** with a weight vector $\mathbf{W}_i$, $Z_i = \mathbf{X}\cdot\mathbf{W}_i$, and then perform a nonlinear operation on Z_i before outputting its activation level to the summation unit. Instead of the Sigmoid Activation Function commonly used for back-propagation [8], the nonlinear operation used here is $\exp[(Z_i - 1)/\sigma^2]$. Assuming that both X and W_i are normalized to unit length, this is equivalent to using

$$\exp\left[-(W_i - X)^t (W_i - X) / 2\sigma^2 \right] .$$

The summation units simply sum the inputs from the pattern units which correspond to the category from which the training pattern was selected.

The decision units are 2-input neurons, as shown in Figure 4, which produce a binary output. They have only a single variable weight, C_i, where

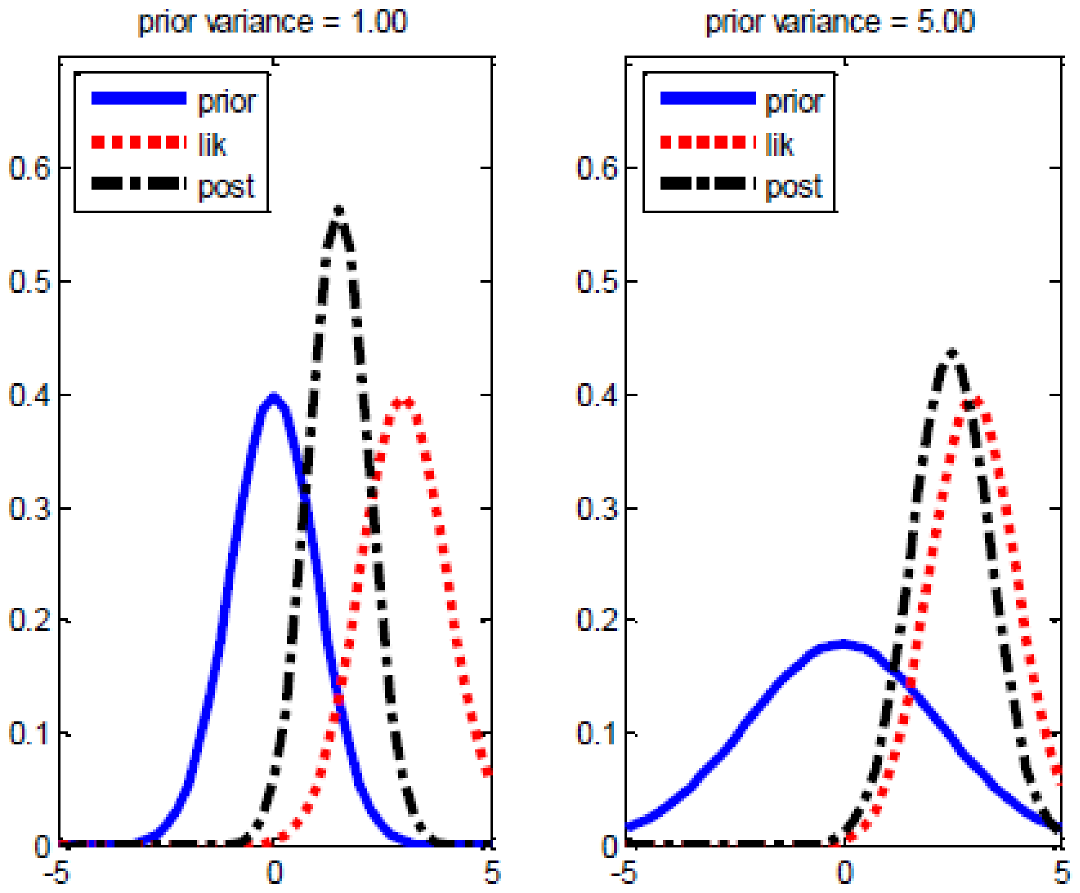
$$C_i = - \frac{h_{Bi} l_{Bi}}{h_{Ai} l_{Ai}} \cdot \frac{n_{Ai}}{n_{Bi}} \qquad (5)$$

and

n_{Ai} = number of training patterns from category A_i ,
n_{Bi} = number of training patterns from category B_i .

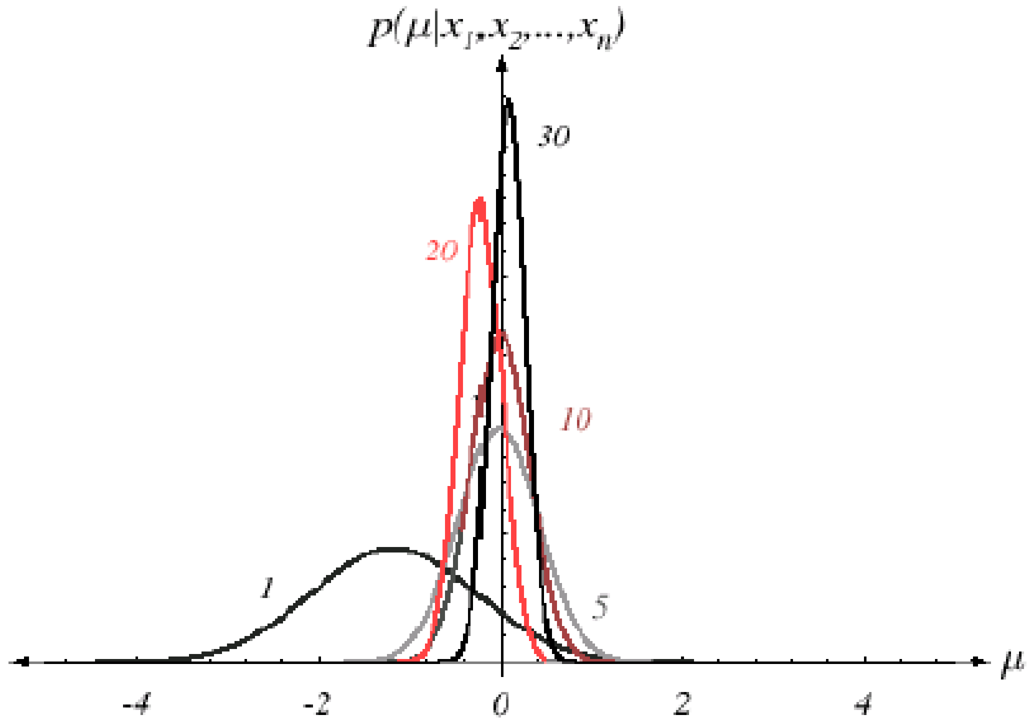
Note that C_i is the ratio of a priori probabilities, divided by the ratio of samples, and multiplied by the ratio of losses. In any problem in which the numbers of training samples from categories A and B are taken in proportion to their a priori probabilities, $C_i = - l_{Bi} / l_{Ai}$. This final ratio cannot be determined from the statistics of the training samples, but only from the significance of the decision. If there is no particular reason for biasing the decision, C_i may simplify to -1 (an inverter).

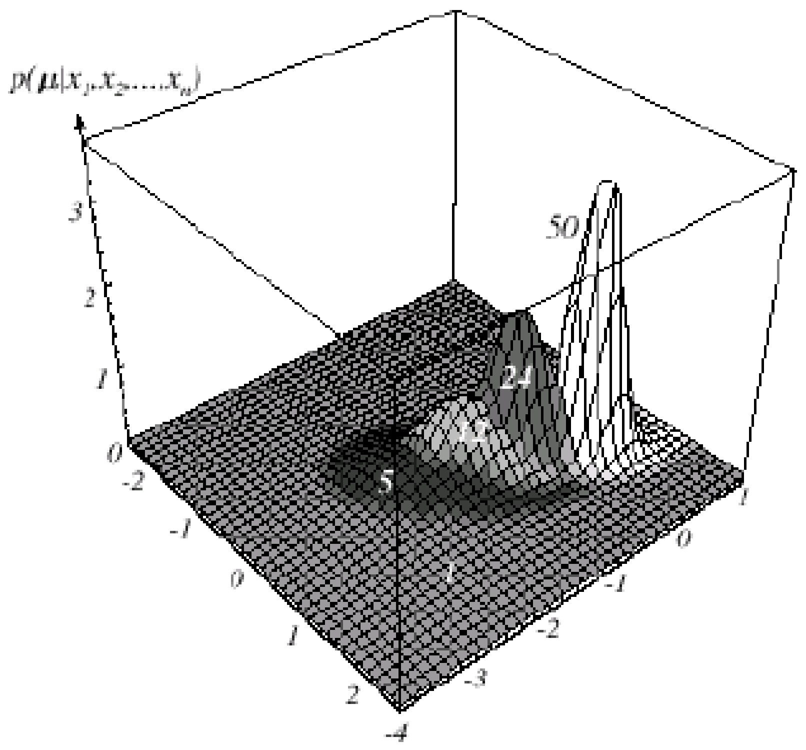
Training of the network is accomplished by setting each X pattern in the training set equal to the W_i weight vector in one of the pattern units, and then connecting the pattern unit's output to the appropriate summation unit. A separate neuron is required for every training pattern. As is indicated in Fig. 2, the same pattern units can be grouped by different summation units to provide additional pairs of categories and additional bits of information in the output vector.



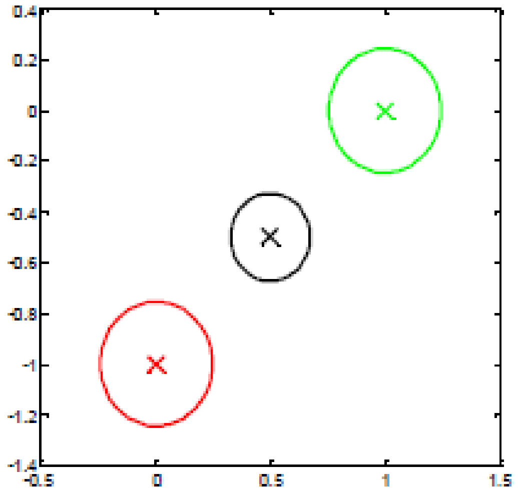
gaussInferParamsMean1d from Murphy, Page 121

Bayesian learning of the mean of Gaussian distributions in one dimension.
Strong prior (small variance) $\Rightarrow$ posterior mean "shrinks" towards the prior mean.
Weak prior (large variance) $\Rightarrow$ posterior mean is similar to the MLE

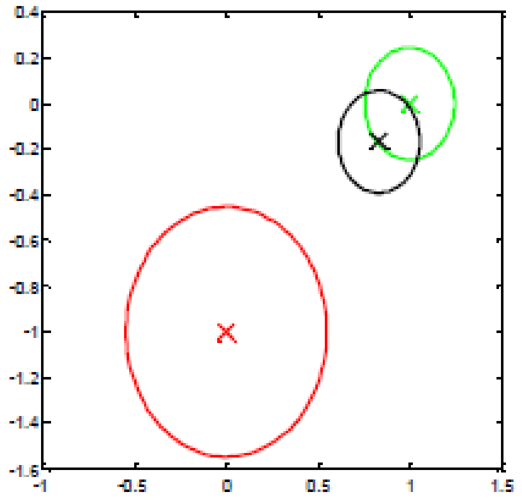




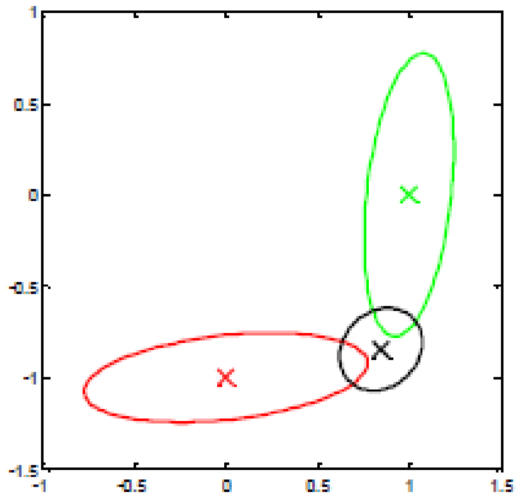
Bayesian learning of the mean of Gaussian distributions in one and two dimensions. As the number of samples increase, the posterior density peaks at the true value.



Equally reliable Sensors (R,G)
Fused Estimate: Black



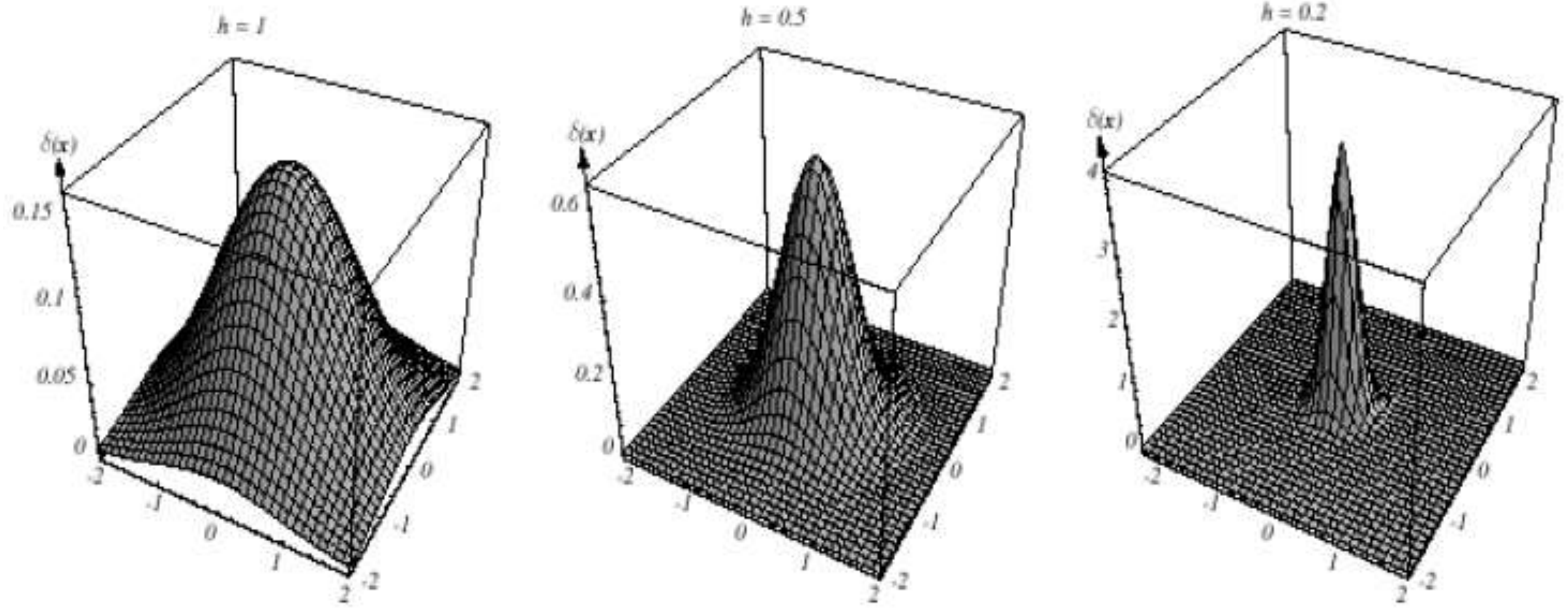
G is more reliable than R
Fused Estimate: Black



R is more reliable in y
G is more reliable in x
Fused Estimate: Black

sensorFusion2d from Murphy, Page 123

Bayesian sensor fusion appropriately combines measurements from multiple sensors based on uncertainty of the sensor measurements. Larger uncertainty $\Rightarrow$ less weight.



Example of two dimensional circularly symmetric normal Parzen windows for three different values of *h*

$$\boxed{H\left(\frac{\underline{x}}{h}\right) = \frac{1}{(2\pi)^{p/2} h^{p}} \exp\left[-\frac{1}{2} \left\| \frac{\underline{x}}{h} \right\|^{2}\right]}$$

$$\hat{p}(\underline{x}) = \frac{k}{NV} = \frac{1}{Nh^p}\sum_{j=1}^{N} H\left(\frac{\underline{x}-\underline{x}^j}{h}\right)$$

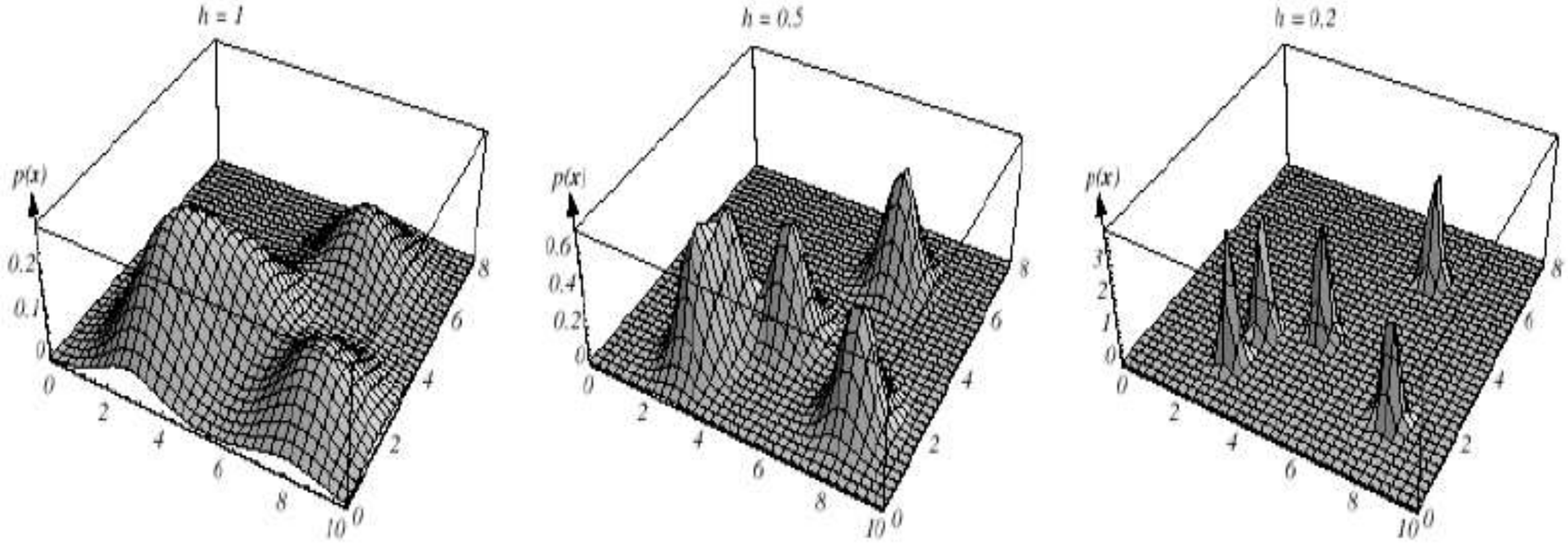
$\Rightarrow \hat{p}(\underline{x}) =$ superposition of N cubes of side h, with each cube centered on one of the data points

$\Rightarrow$ We can smooth out this estimate by choosing different forms for the kernel function $H(\underline{u})$. *Example: Gaussian Parzen Windows*

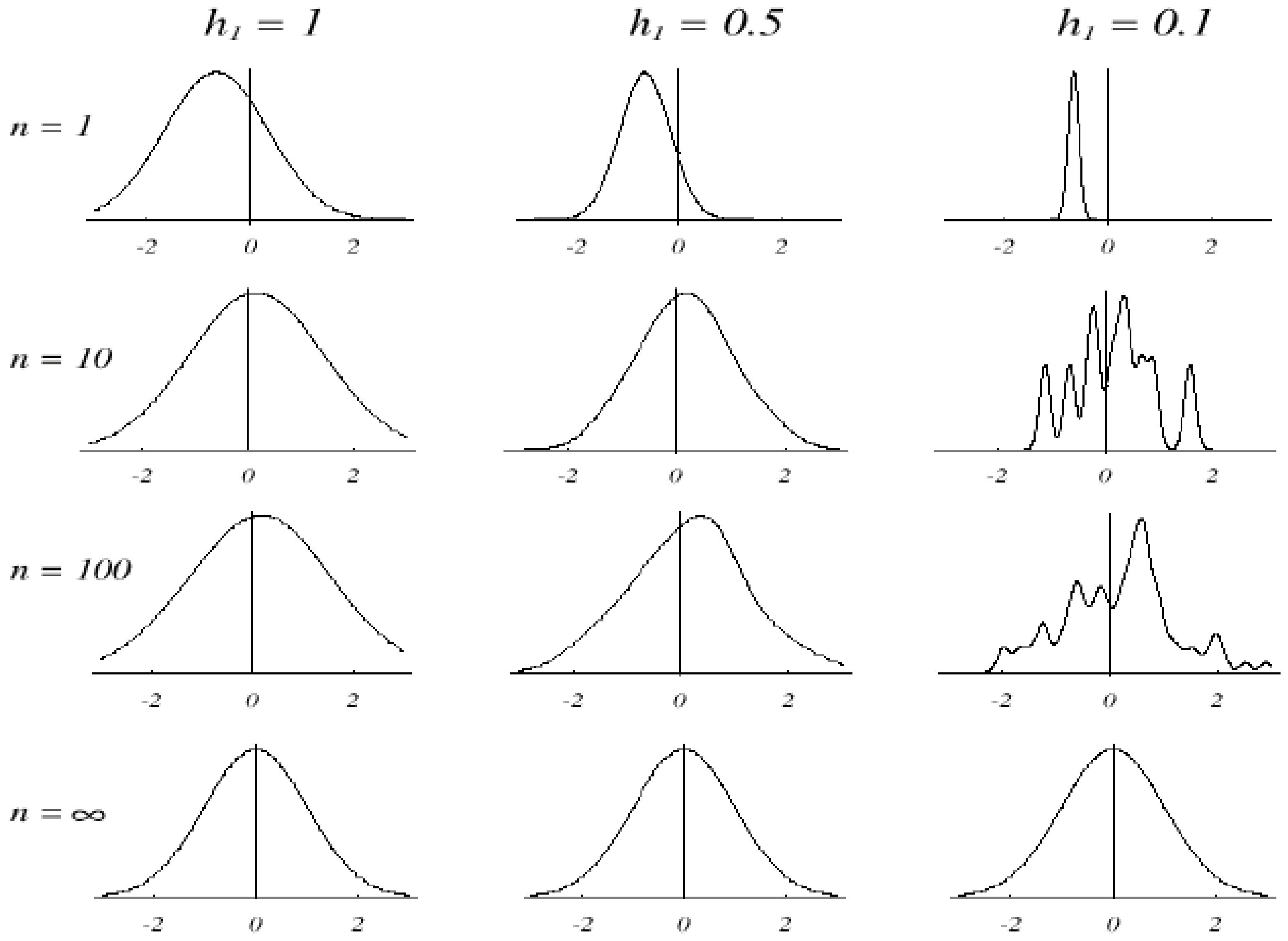
$$E[\hat{p}(\underline{x})] = E\left[\frac{1}{N}\frac{1}{h^p}\sum_{j=1}^{N} H\left(\frac{\underline{x}-\underline{x}^j}{h}\right)\right] = E\left[\frac{1}{h^p} H\left(\frac{\underline{x}-\underline{v}}{h}\right)\right]$$

$$= \frac{1}{h^p}\int_{\underline{v}} H(\underline{x}-\underline{v})\,p(\underline{v})\,d\underline{v} \longleftarrow \boxed{\begin{array}{l}\text{Convolution of } H \text{ and } p.\\ \text{Blurred version of } p(\underline{x}) \text{ as}\\ \text{seen through the window}\end{array}}$$

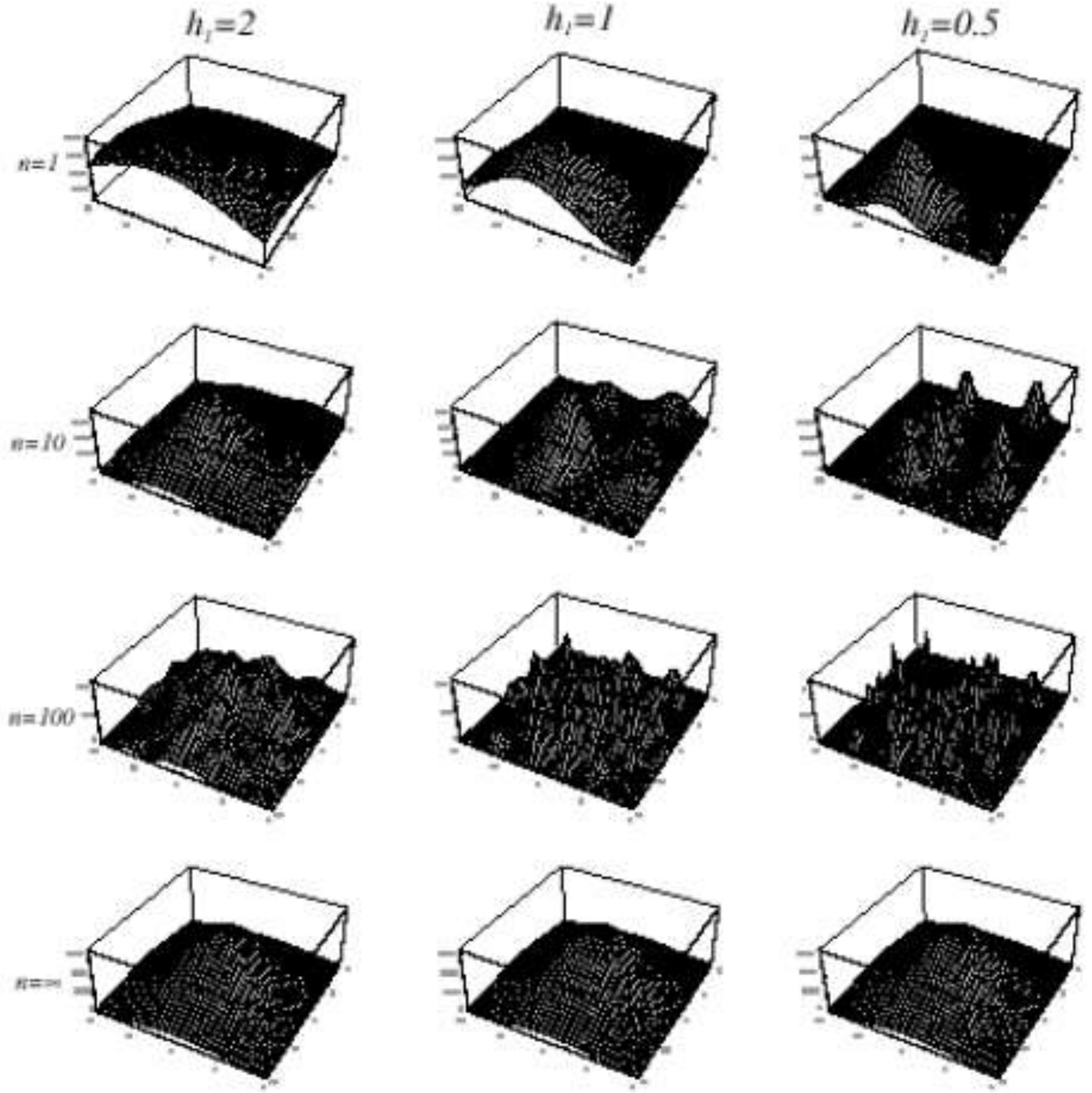
Large $N \Rightarrow$ Good estimate for $h \downarrow 0$
Small $N \Rightarrow$ Need to select h properly
For small N, h small $\Rightarrow$ noisy estimate



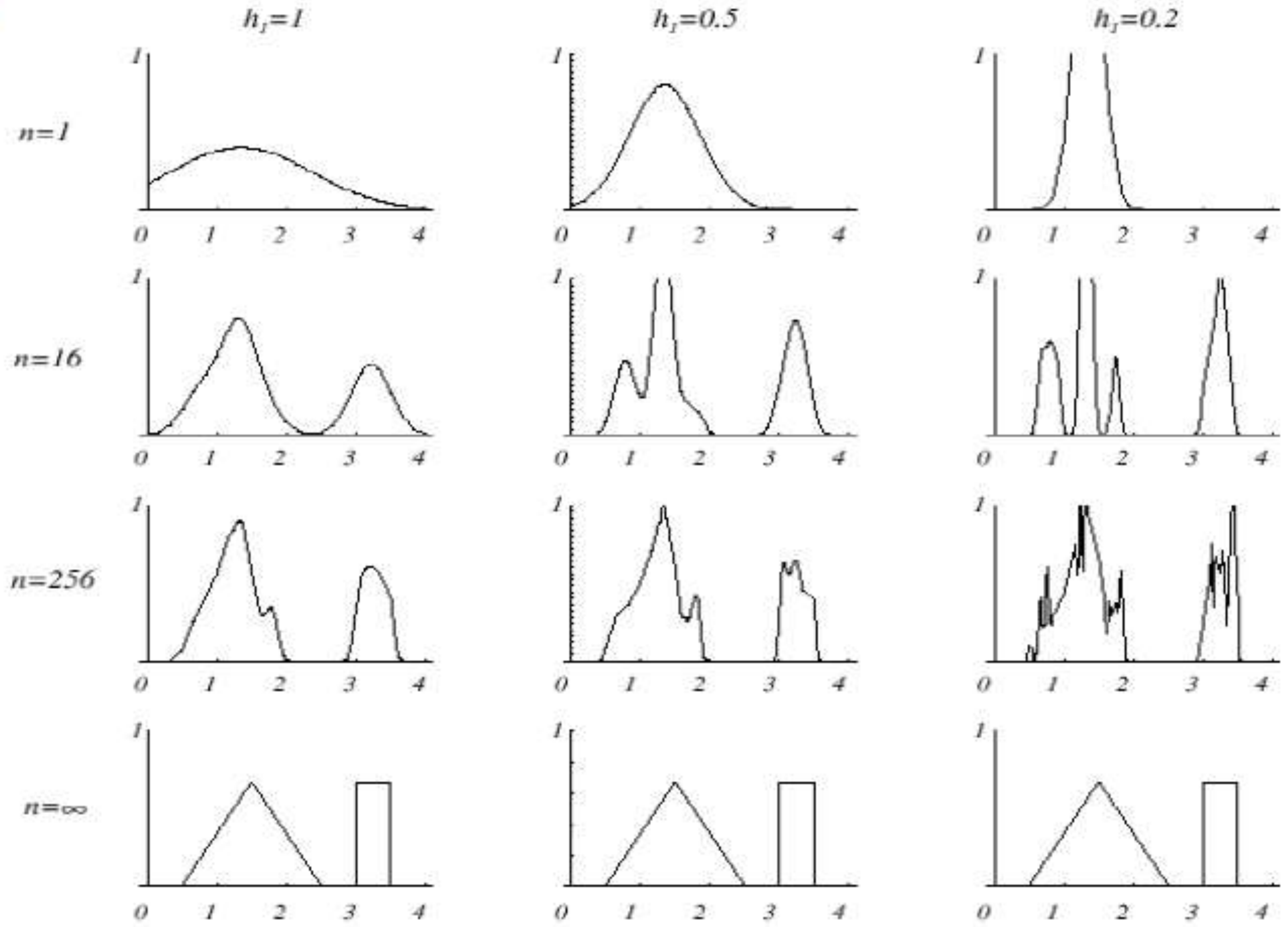
Three Parzen-window density estimates based on the same set of five samples, using the Gaussian Parzen window functions



Parzen-window estimates of a univariate normal density using different window widths and numbers of samples



Parzen-window estimates of a bivariate normal density using different window widths and numbers of samples



Parzen-window estimates of a bimodal distribution using different window widths and numbers of samples. Note that when $n \rightarrow \infty$, estimates are the same and match the true distribution.

Volume of a hypersphere in p dimensions of radius $\sigma = \dfrac{2(\pi)^{p/2}}{\Gamma(p/2)} \dfrac{\sigma^p}{p}$

$$\Gamma(a) = \int_0^\infty u^{a-1} e^{-u}\, du$$

$$\hat{p}(\underline{x}) = \frac{1}{(2\pi)^{p/2}\sigma^p} \frac{1}{N} \sum_{j=1}^{N} \exp\left\{ -\frac{(\underline{x}-\underline{x}^j)^T(\underline{x}-\underline{x}^j)}{2\sigma^2} \right\} \qquad \sigma = h$$

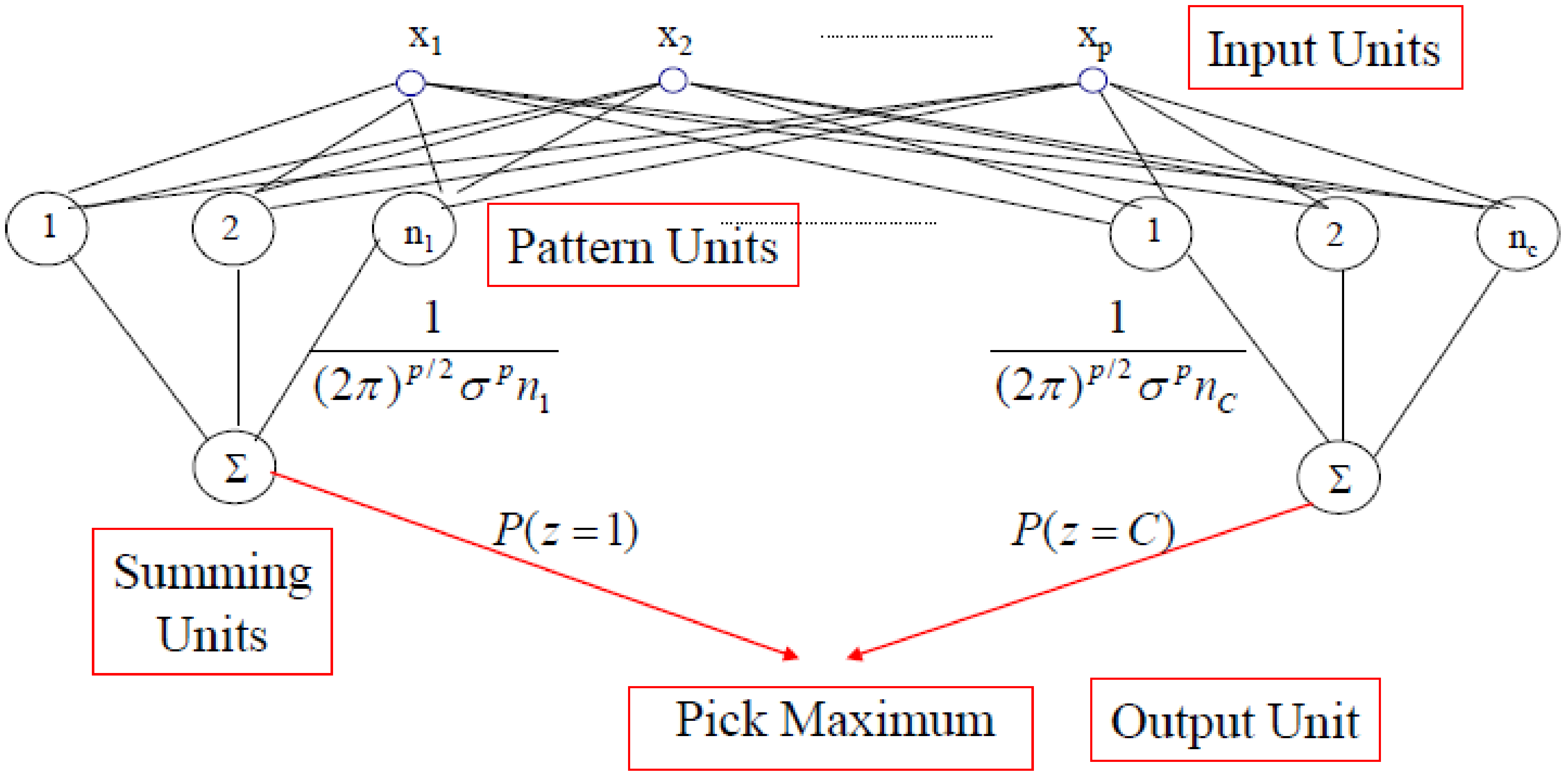
= sum of multivariate Gaussian distributions centered at each training sample.

(can also do for each class $\hat{p}(\underline{x} \mid z = k) \Rightarrow \begin{matrix} N \to n_k \\ \underline{x}^j \to \underline{x}_k^j \end{matrix}$)

Also called "Probabilistic Neural Network (PNN)"

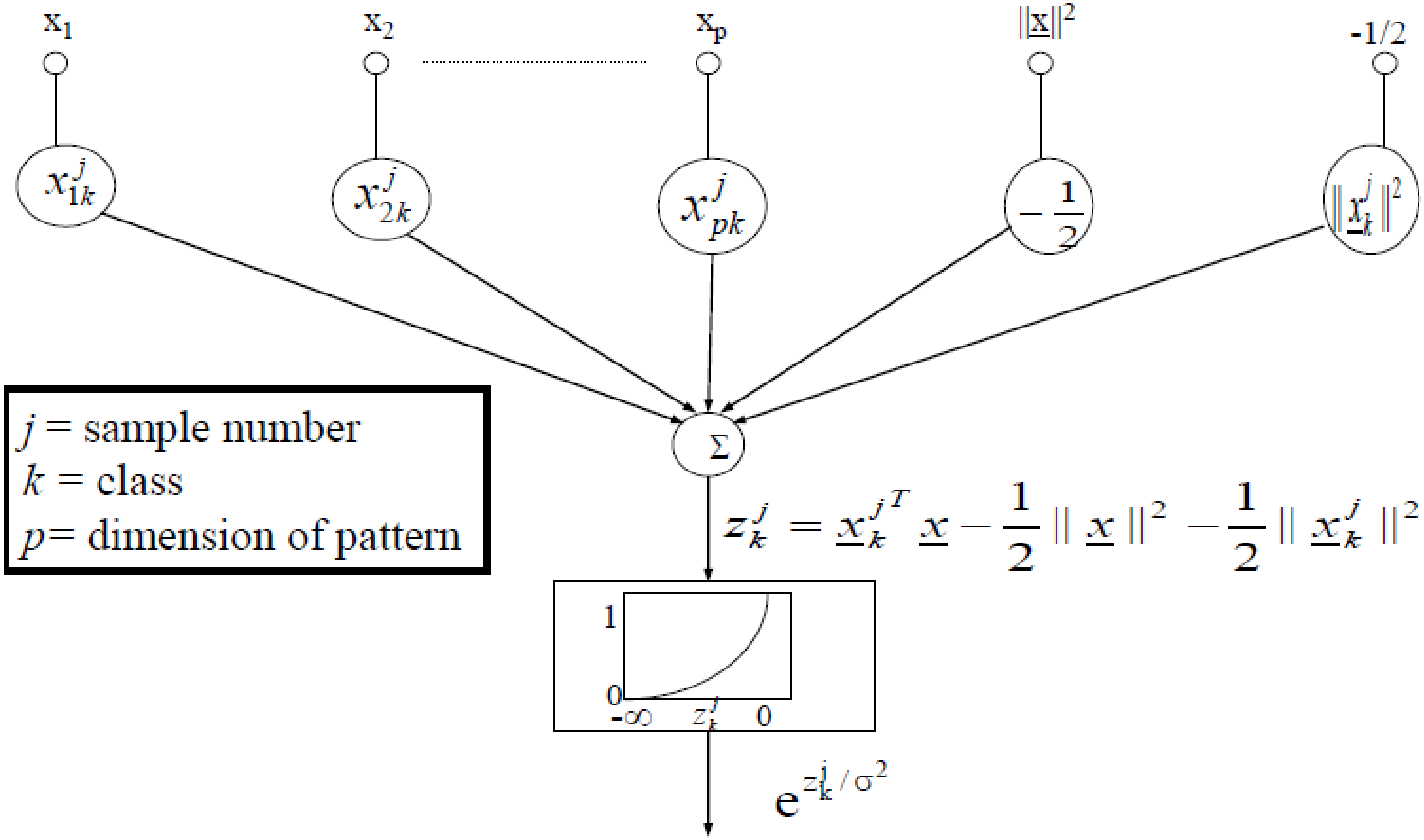
Small σ ⇒ density estimation will have discontinuities
Larger σ ⇒ causes greater degree of interpolation (smoothness)

PNN structure



- Speed up PNN using cluster centers as representative patterns
  - Cluster using *K*-means, LVQ,….

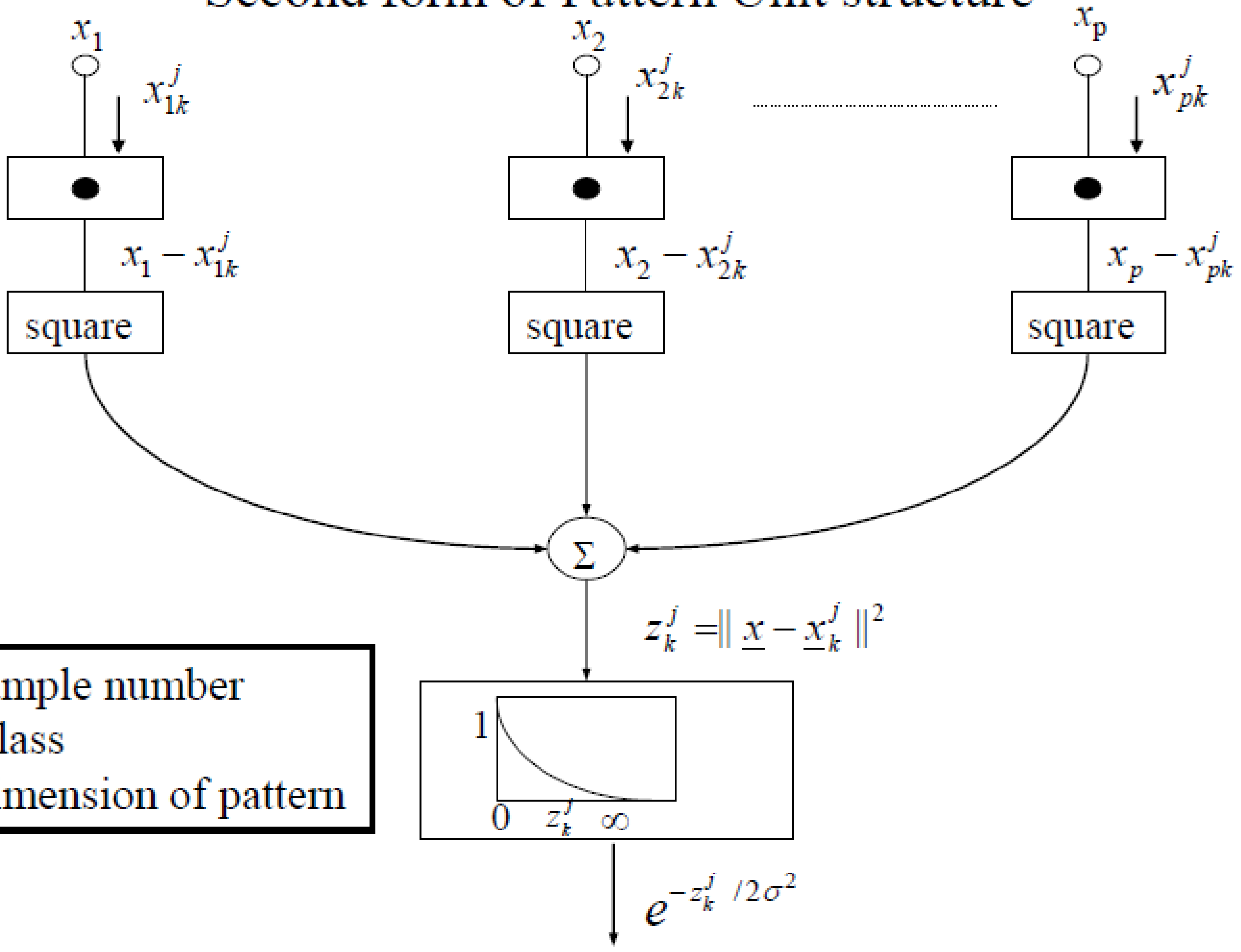
One form of Pattern Unit structure



$$z_k^j = \underline{x}_k^{j\,T}\underline{x} - \frac{1}{2}\|\underline{x}\|^2 - \frac{1}{2}\|\underline{x}_k^j\|^2$$

j = sample number
k = class
p = dimension of pattern

Second form of Pattern Unit structure



j = sample number
k = class
p= dimension of pattern

❑ Alternate forms of Kernels:

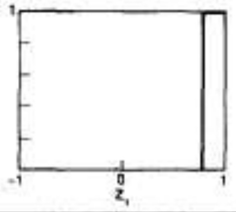
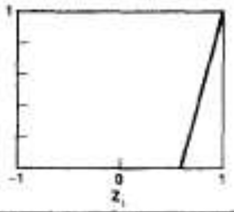
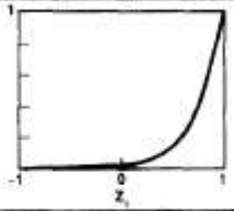
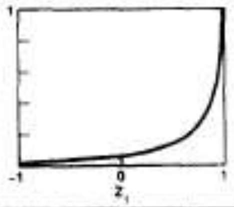
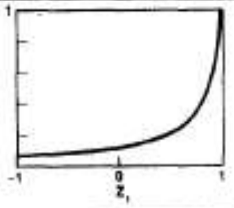
$$p(x) = \frac{1}{Nh^p}\sum_{j=1}^{N}\exp\left(\underbrace{-\frac{2}{h}\sum_{i=1}^{p}|x_i - x_i^j|}_{\text{Manhattan Distance, 1-norm}}\right)$$

$$p(\underline{x}) = \frac{1}{N(\pi h)^p}\sum_{j=1}^{N}\prod_{i=1}^{p}\left[1 + \frac{(x_i - x_i^j)^2}{h^2}\right]^{-1} \quad \text{Cauchy Distribution}$$

$$p(x) = \frac{1}{N(2\pi\sigma)^p}\sum_{j=1}^{N}\prod_{i=1}^{p}\left[\sin c\left(\frac{(x_i - x_i^j)}{\sigma}\right)\right]^2 \quad \text{where} \quad \sin c(x) = \frac{\sin x}{x}$$

$$p(\underline{x}) = \frac{1}{N}\left(\frac{3}{4}\right)^p\sum_{j=1}^{N}\prod_{i=1}^{p}\left(1 - x_i^{\,2}\right); |x_i| \le 1 \quad \textit{Epanechnikov Kernel}$$

$$p(\underline{x}) = \frac{1}{N}\left(\frac{70}{81}\right)^p\sum_{j=1}^{N}\prod_{i=1}^{p}\left(1 - |x_i|^3\right); |x_i| \le 1 \quad \textit{tri-cube Kernel}$$

W (y)	ACTIVATION FUNCTION
1, $y \leq 1$ 0, $y \geq 1$	
$1 - y$, $y \leq 1$ 0, $y \geq 1$	
$e^{-1/2\, y^2}$	
$e^{-\|y\|}$	
$\frac{1}{1+y^2}$	
$\left(\frac{\sin(y/2)}{y/2}\right)^2$