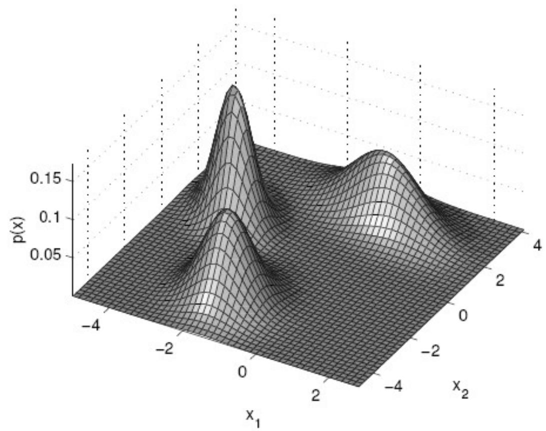
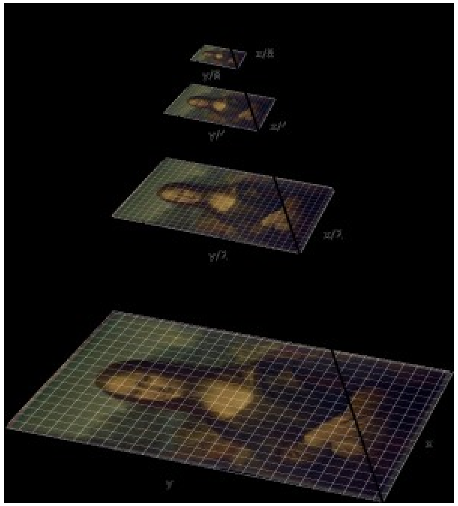
INF555





Fundamentals of 3D

Lecture 9:

Laplacian Image pyramids
Expectation-Maximization

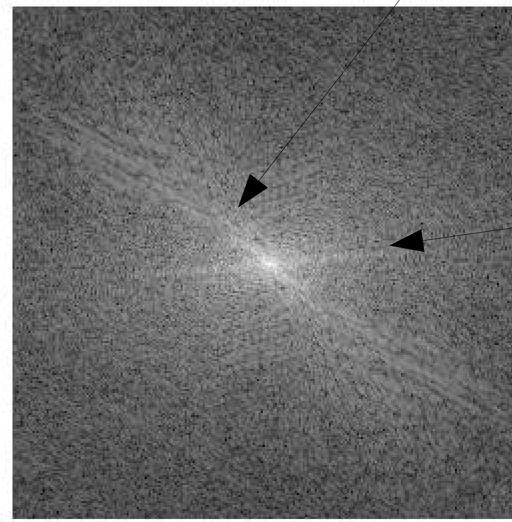
+ Overview of computational photography

Frank Nielsen
nielsen@lix.polytechnique.fr

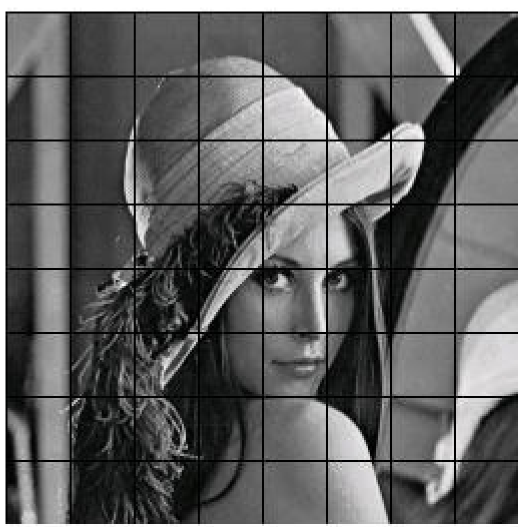
Interpreting Fourier spectra

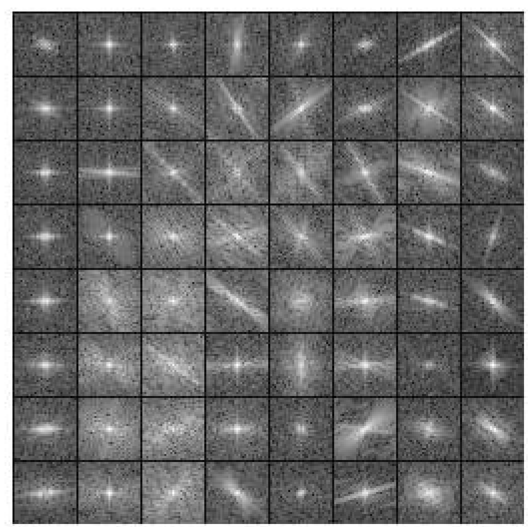
Stripes of the hat

Stripes of the hair

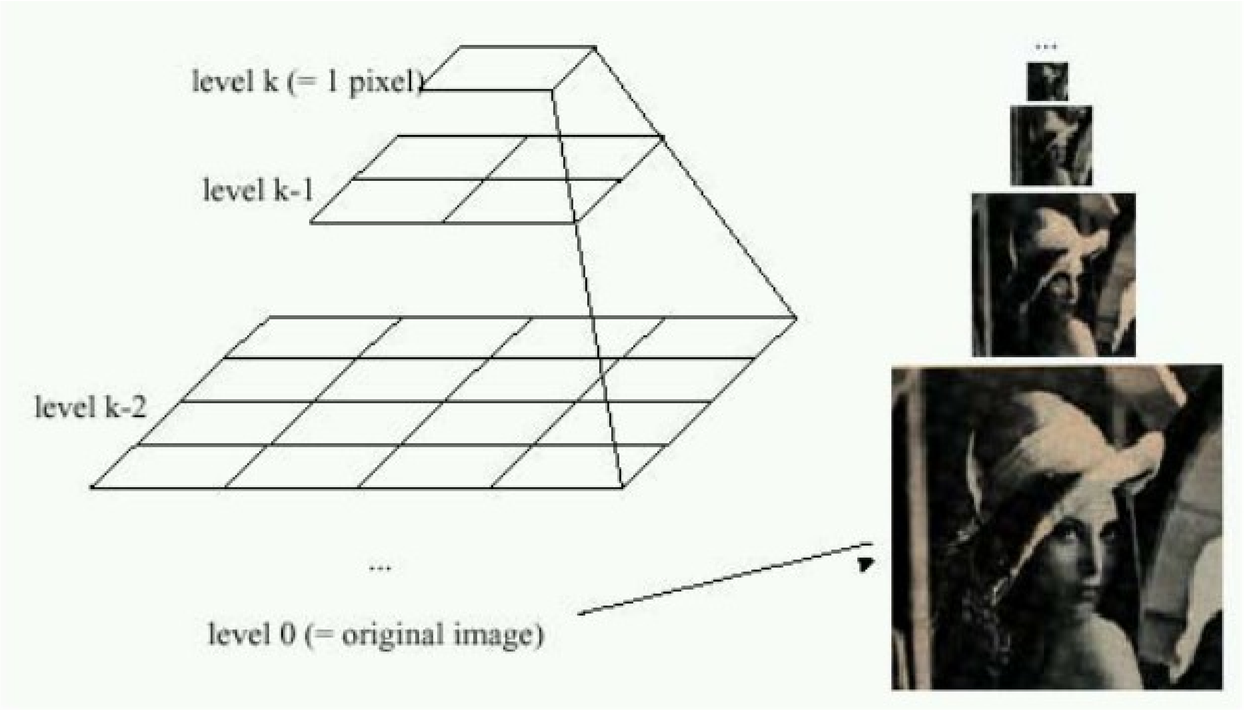


Fourier log power spectrum





Laplacian image pyramids



Used also in graphics for texturing (mipmapping)

Laplacian image pyramids

Gaussian. Blur and sample, and then

Laplacian. Interpolate and estimate

$$\mathbf{G}_1 = \mathbf{I}$$

$$\mathbf{G}_i = \mathrm{EXPAND}(\mathbf{G}_{i+1}) + \mathbf{L}_i.$$ Reconstruction

$$\mathbf{L}_i = \mathbf{G}_i - \mathrm{EXPAND}(\mathbf{G}_{i+1})$$ Residual

$\mathbf{L}_1 = \mathbf{G}_1 - \mathrm{EXPAND}(\mathbf{G}_2)$	$\mathbf{G}_4 = \mathbf{L}_4 + \mathrm{EXPAND}(\mathbf{G}_5)$
$\mathbf{L}_2 = \mathbf{G}_2 - \mathrm{EXPAND}(\mathbf{G}_3)$	$\mathbf{G}_3 = \mathbf{L}_3 + \mathrm{EXPAND}(\mathbf{G}_4)$
$\mathbf{L}_3 = \mathbf{G}_3 - \mathrm{EXPAND}(\mathbf{G}_4)$	$\mathbf{G}_2 = \mathbf{L}_2 + \mathrm{EXPAND}(\mathbf{G}_3)$
$\mathbf{L}_4 = \mathbf{G}_4$	$\mathbf{G}_1 = \mathbf{L}_1 + \mathrm{EXPAND}(\mathbf{G}_2) = \mathbf{I}$

Precursors of wavelets

Laplacian image pyramids

Blurring is efficient for sampling as it removes high-frequency components. (sample at fewer positions.)

Gaussian kernel and resampling at a *quarter* of the image size.
Blurring and resampling is computed using a *single* discrete kernel.

- Central limit theorem:
  (mean of random variables approach Gaussian distribution)

- Infinitely differentiable functions

- Fourier of Gaussians are Gaussians

- Human brain has neuronal regions doing Gaussian filtering

Laplacian image pyramids

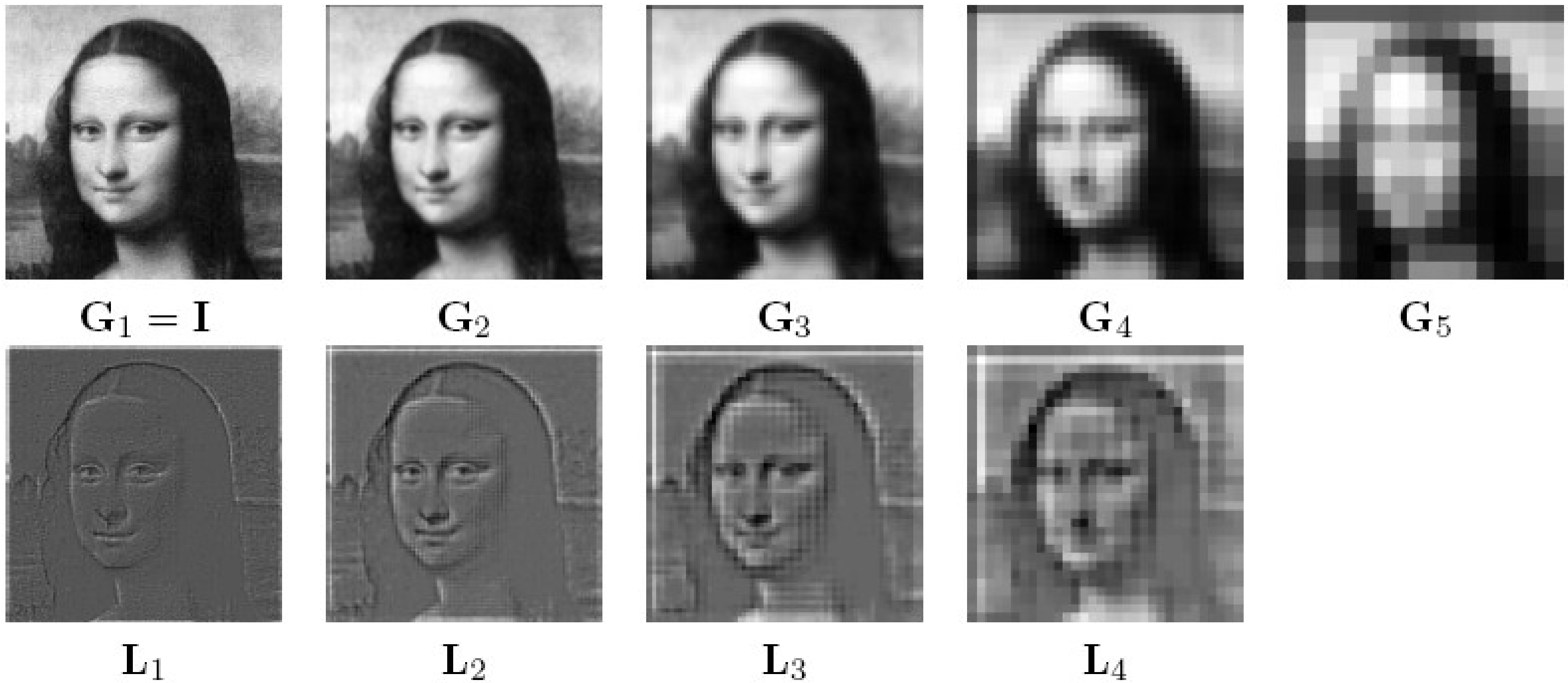
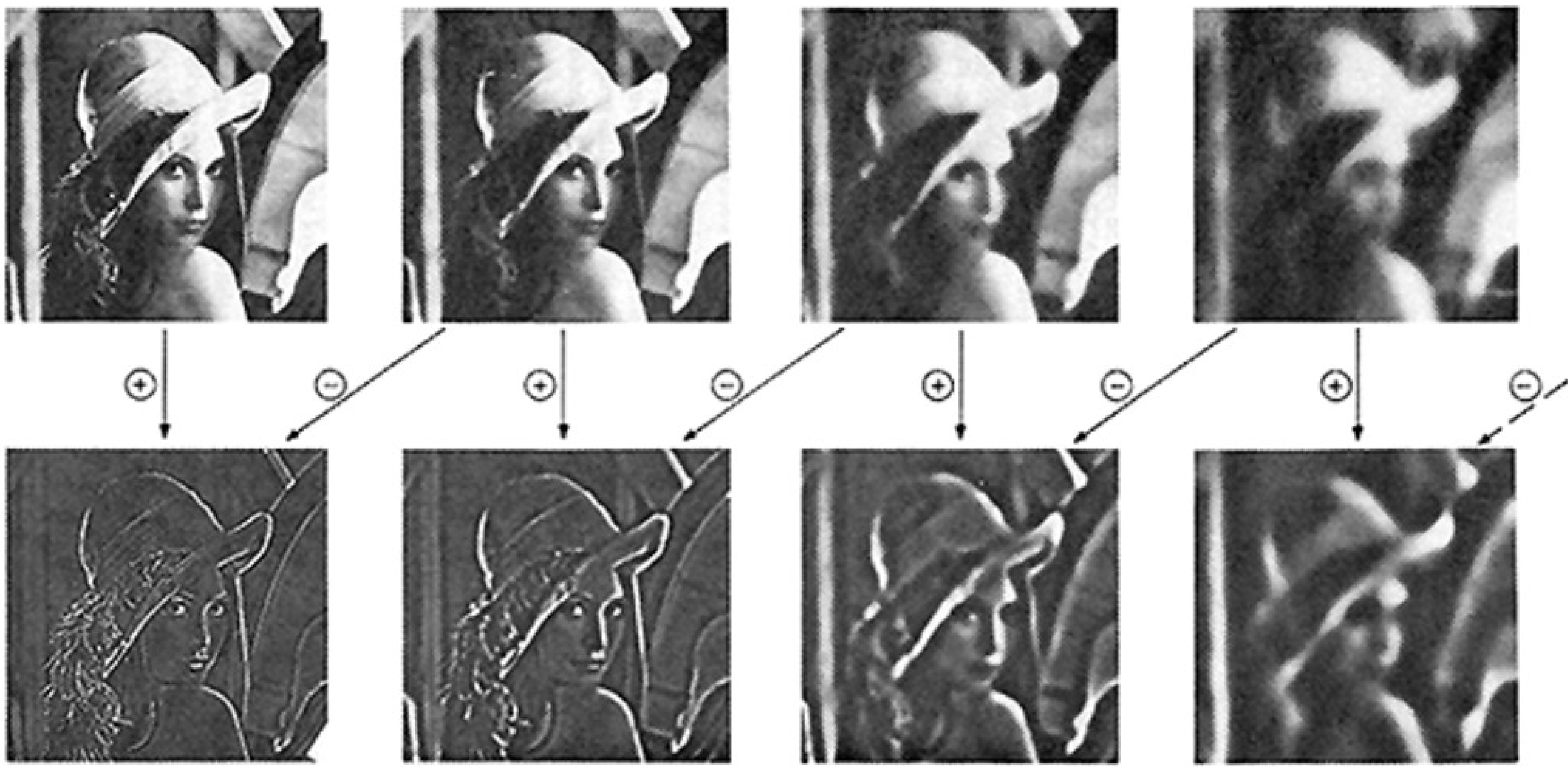


FIGURE 4.46 *Gaussian and Laplacian image pyramids: the original image* $\mathbf{I}$ *can be reconstructed without any error from the smallest image of the Gaussian pyramid* ($\mathbf{G}_5$) *and the Laplacian image pyramid* $\mathcal{L} = \{\mathbf{L}_i\}_i$.



Laplacian image pyramids: Application to blending

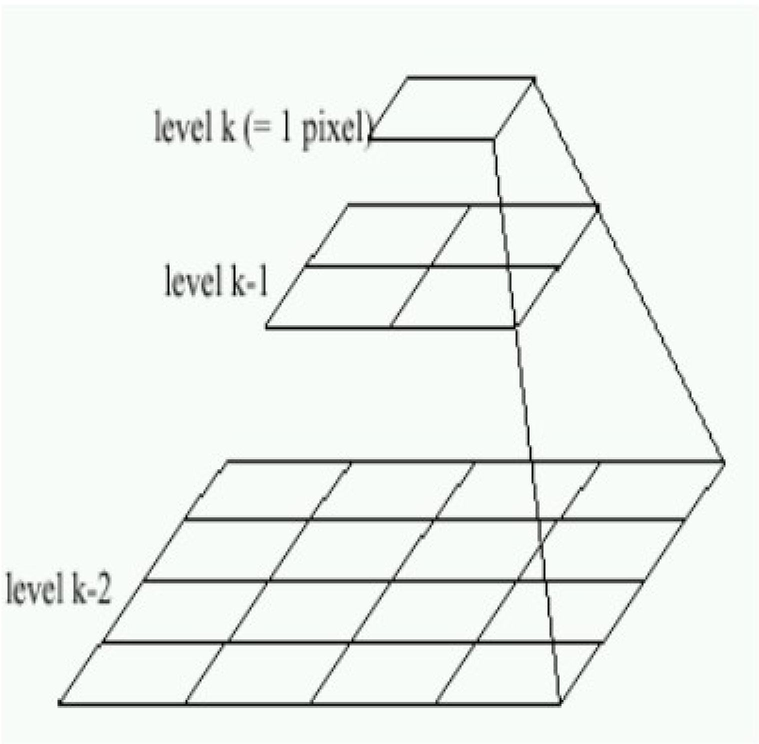
Multiband blending.

Blending two overlapping images using their pyramids

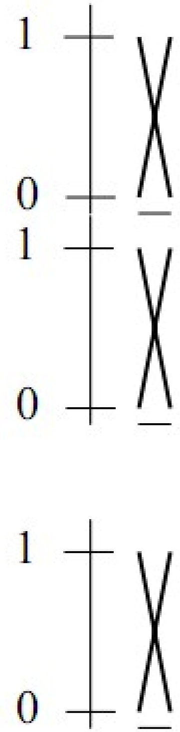
- Compute Laplacian pyramids $L(\mathbf{I}1)$ and $L(\mathbf{I}2)$ of $\mathbf{I}1$ and $\mathbf{I}2$.
- Generate a hybrid Laplacian pyramid L_r by creating for each image of the   pyramid a 50%/50% mix of images, obtained by selecting the leftmost half of $L(\mathbf{I}1)$ with the rightmost half of $L(\mathbf{I}2)$.

- Reconstruct blended images from the Laplacian pyramid L_r.

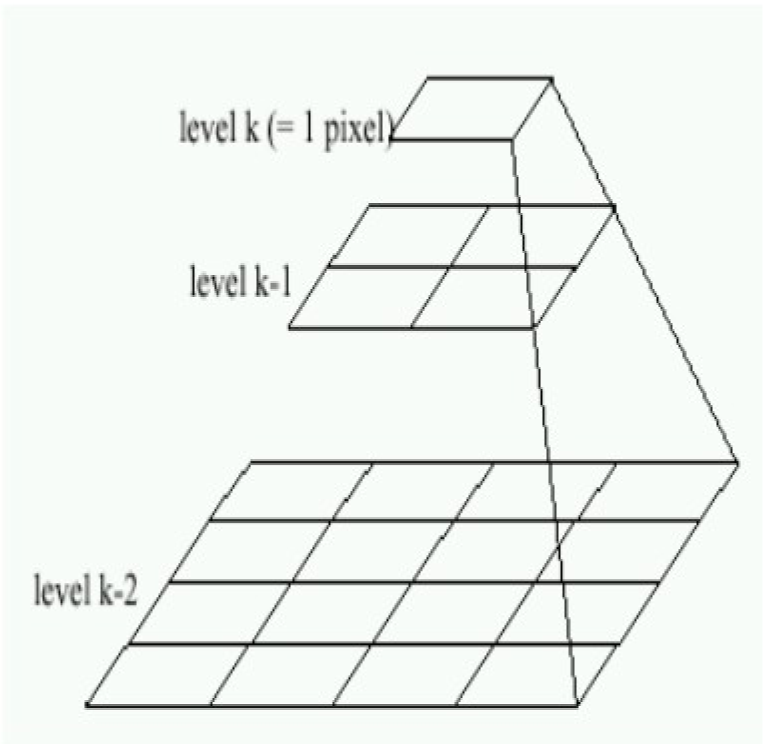
Laplacian image pyramids: Application to blending



Left pyramid

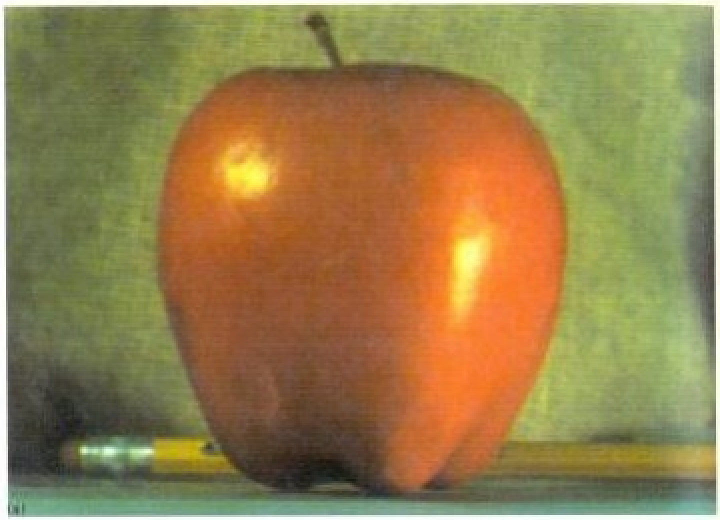


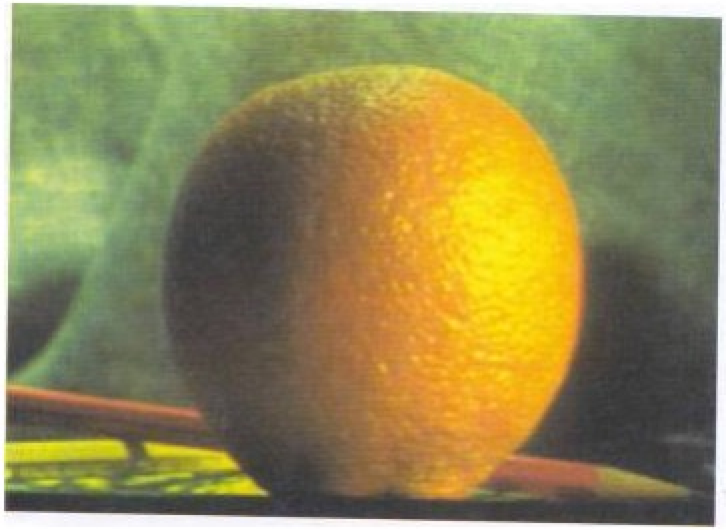
blend

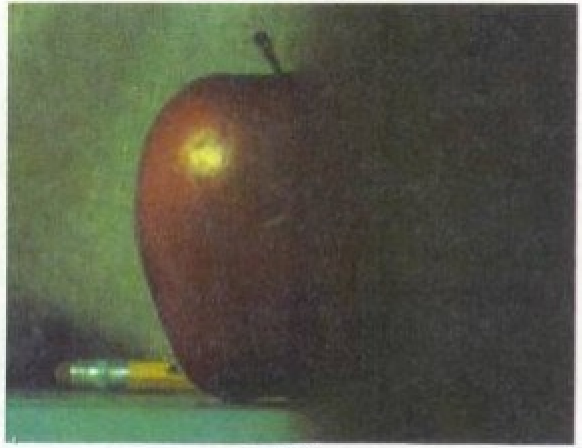


Right pyramid

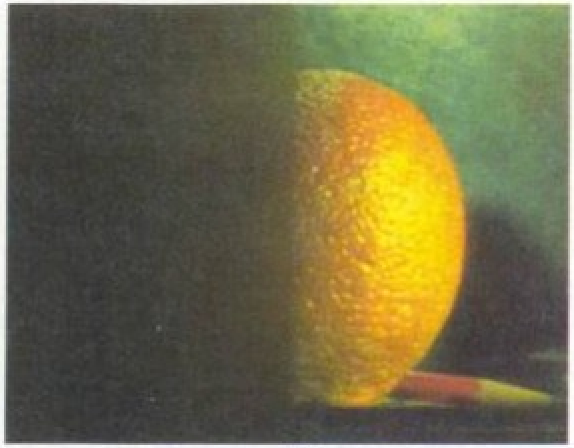
Laplacian image pyramids: Application to blending



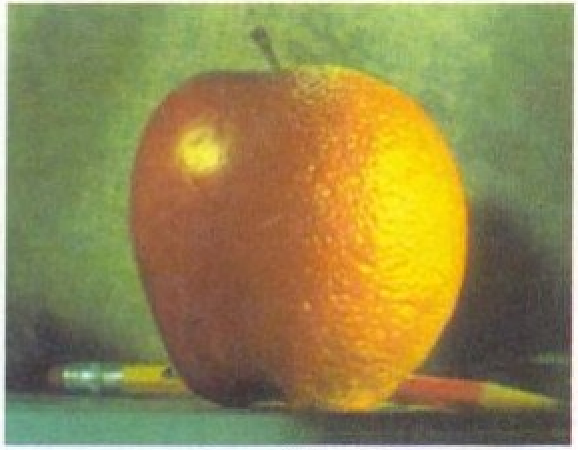




(d)

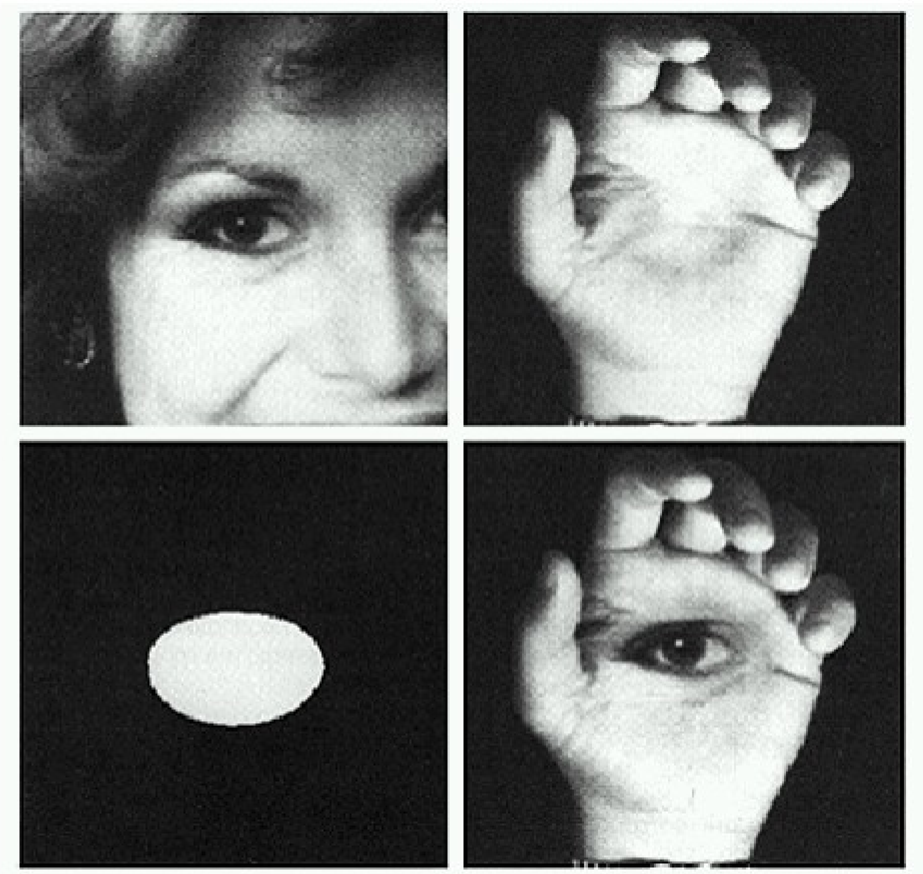


(h)

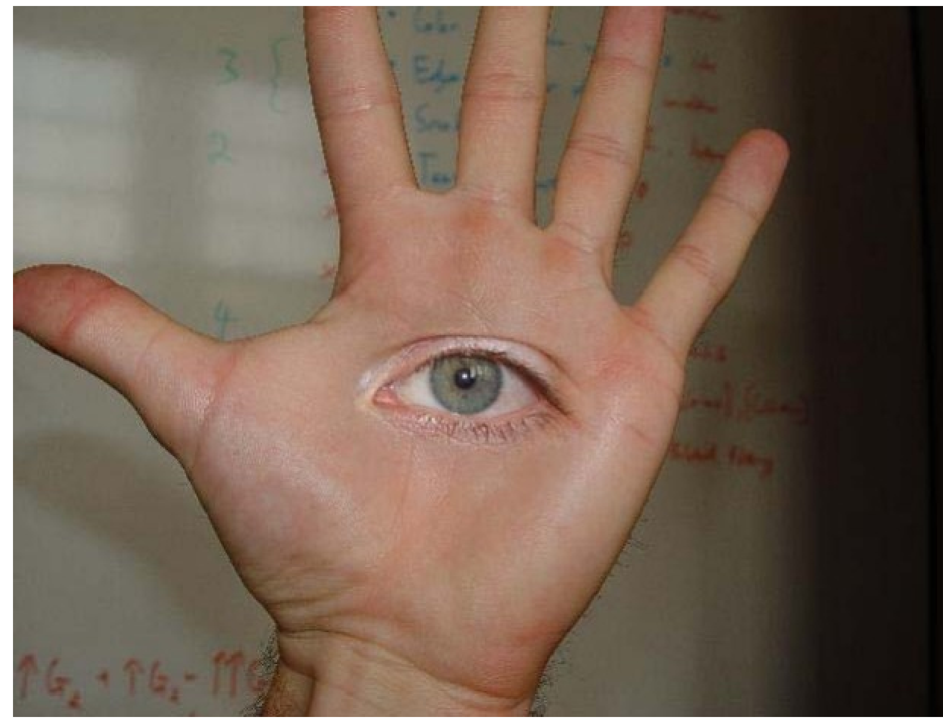


(l)

Laplacian image pyramids: Application to blending



Using a region mask



Nowadays, we better use **Poisson image editing** and
gradient/image reconstruction

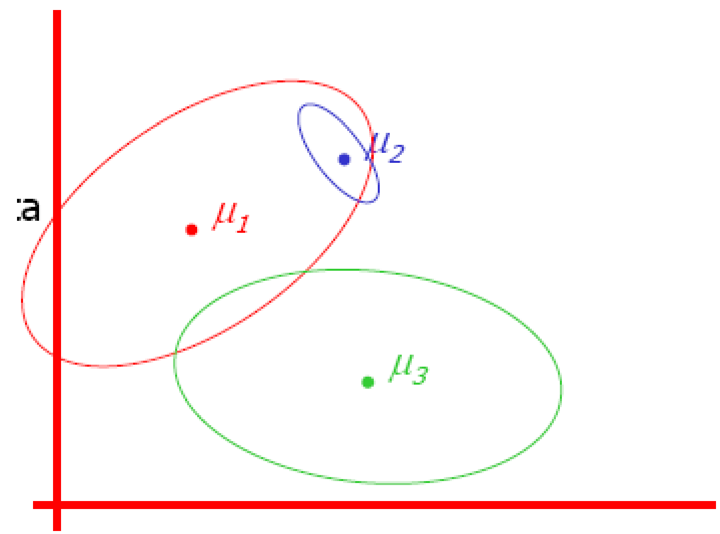
Expectation-Maximization (EM)

Generative statistical models

$$\mathbf{x};\boldsymbol{\theta}_m \sim \mathcal{N}\left(\mathbf{x};\boldsymbol{\mu}_m,\boldsymbol{\Sigma}_m\right)$$

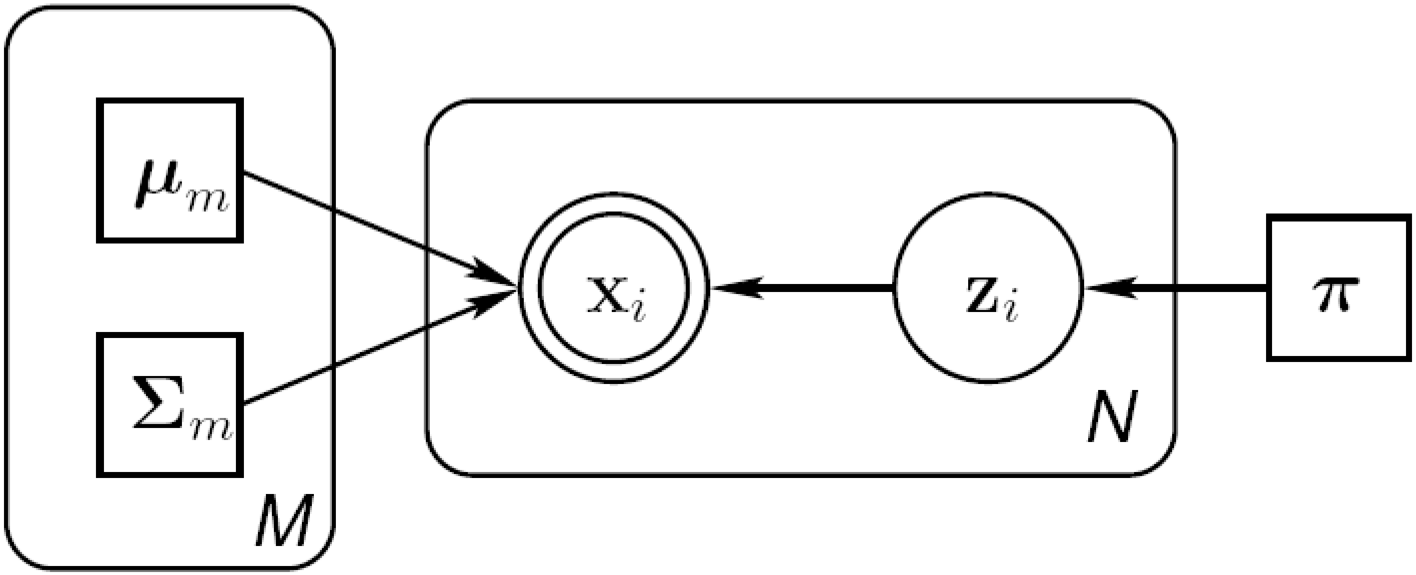
$$\mathcal{N}\left(\mathbf{x};\boldsymbol{\mu},\boldsymbol{\Sigma}\right) = (2\pi)^{-d/2}\left|\boldsymbol{\Sigma}\right|^{-1/2}\exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}$$

$$f(Y=y|\theta) = \sum_{j=1}^{k}\alpha_j\frac{1}{\sqrt{(2\pi)^d|\Sigma_j|}}\exp\{-\frac{1}{2}(y-\mu_j)^T\Sigma_j^{-1}(y-\mu_j)\}$$



http://www.neurosci.aist.go.jp/~akaho/MixtureEM.html

Indicator variables z



$$\mathbf{z}_i = \underbrace{\begin{bmatrix} 0 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 & 0 \end{bmatrix}^T}_{M \text{ elements}}$$

Multinomially distributed π_m

$$p(\mathbf{x}_i | z_{im} = 1; \boldsymbol{\theta}) = \frac{1}{(2\pi)^{d/2} |\boldsymbol{\Sigma}_m|^{1/2}} \exp\left\{ -\frac{1}{2} (\mathbf{x}_i - \boldsymbol{\mu}_m)^T \boldsymbol{\Sigma}_m^{-1} (\mathbf{x}_i - \boldsymbol{\mu}_m) \right\}$$

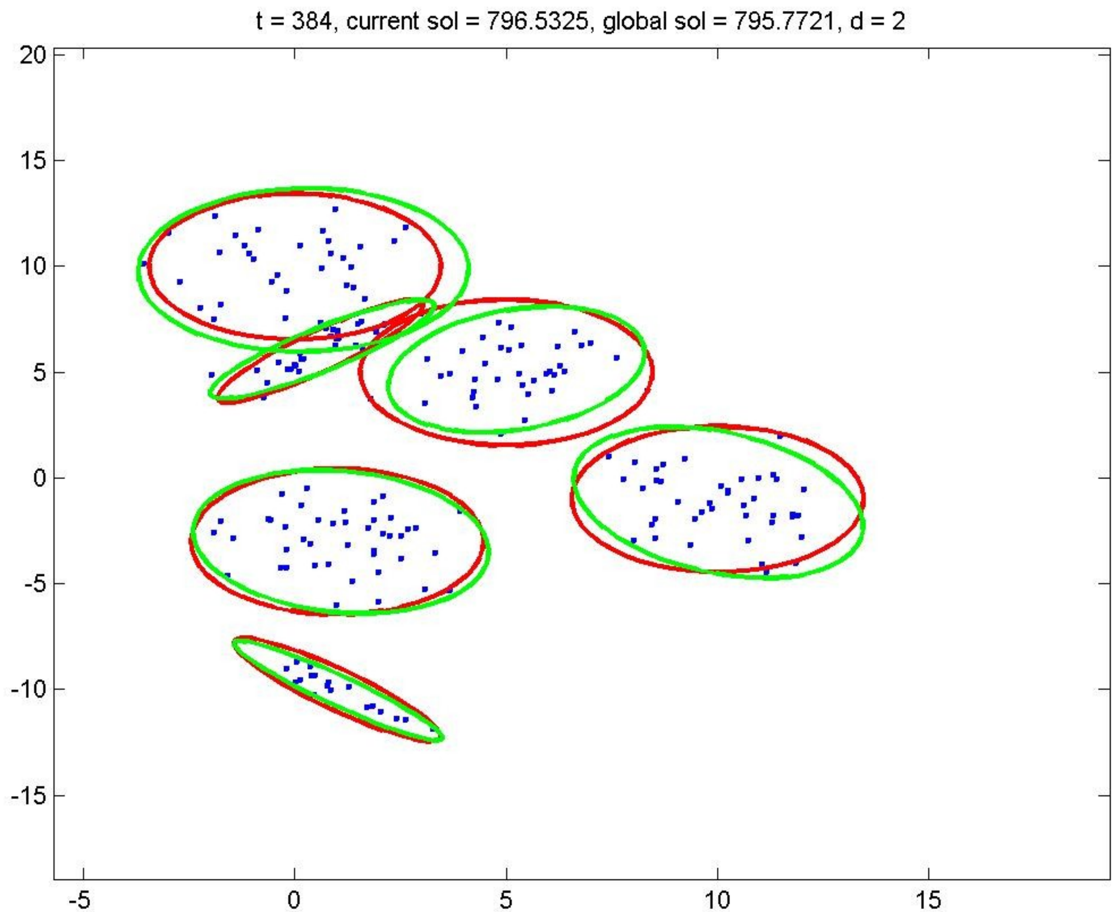
Generating samples from Gaussian Mixture Models (GMMs)

1: **for** $i = 1$ to N **do**

2: $\quad m \leftarrow$ index of one of the M models randomly selected according to the prior probability vector $\boldsymbol{\pi}$

3: $\quad$ Randomly generate $\mathbf{x}_i$ according to the distribution $\mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$

4: **end for**



Maximize the likelihood (incomplete) $\mathcal{L}(\boldsymbol{\theta}) = p(\mathbf{X}; \boldsymbol{\theta})$

$$\boldsymbol{\theta} = \{\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m, \pi_m\}_1^M$$

Maximize the likelihood (complete likelihood) $p(\mathbf{X}, \mathbf{Z}; \boldsymbol{\theta})$

joint distribution of $\mathbf{X}$ and $\mathbf{Z} = \{\mathbf{z}_i\}_1^N$

Expectation-Maximization algorithm: Iteration

EM iteration:

- Expectation step :

$$w_{tj} = p(x_t = j | y_t) = \frac{\alpha_j f(y_t | \mu_j, \Sigma_j)}{\sum_{i=1}^{k} \alpha_i f(y_t | \mu_i, \Sigma_i)}$$

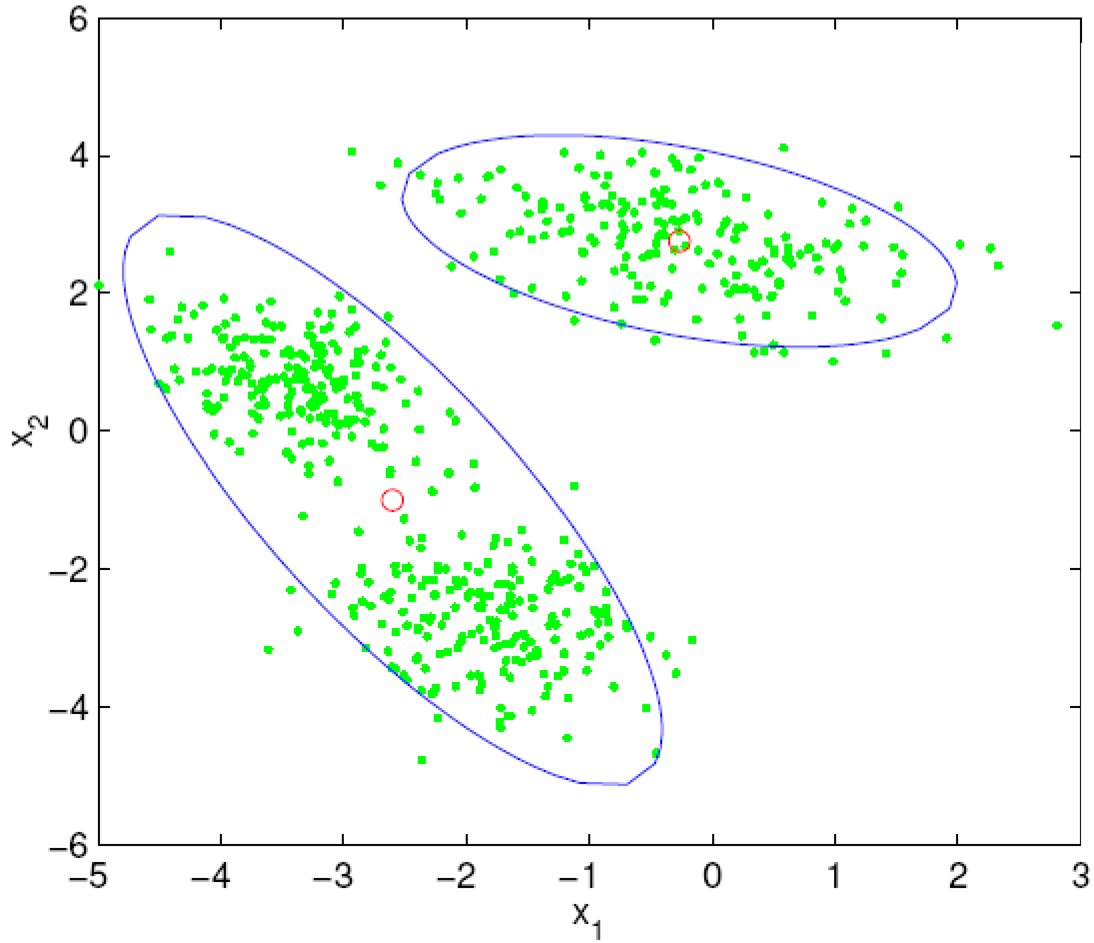
- Maximization step :

$$\hat{\alpha}_j \leftarrow \frac{1}{n} \sum_{t=1}^{n} w_{tj}$$

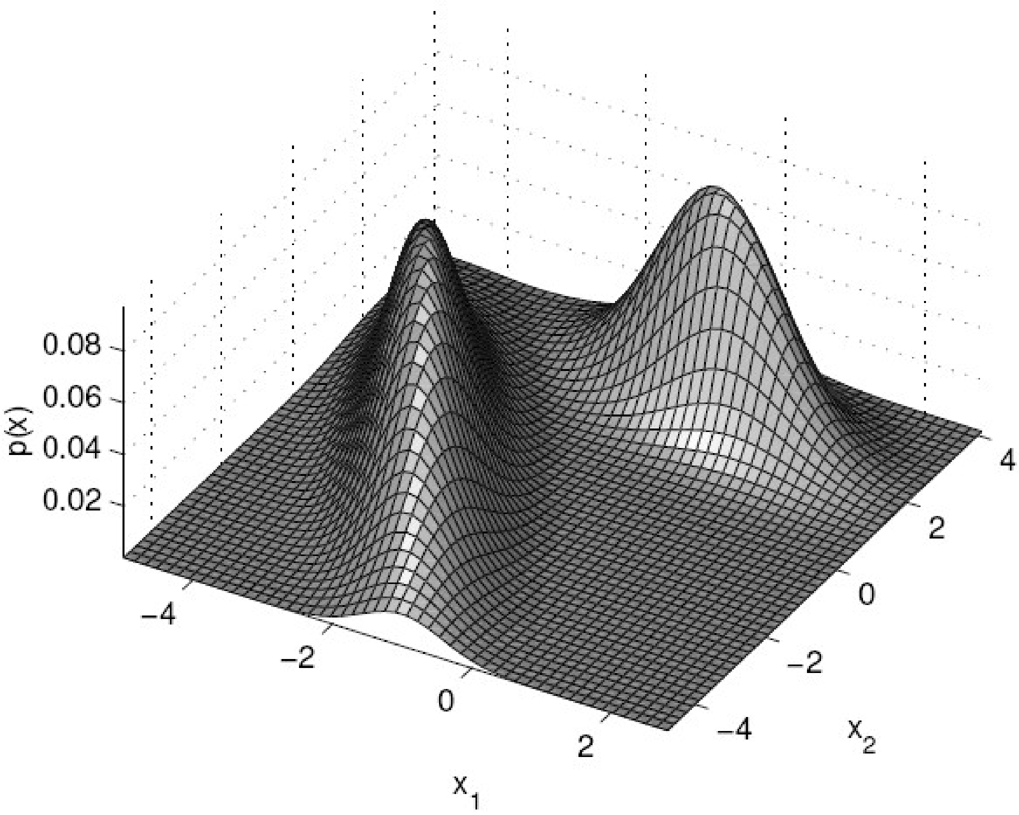
$$\hat{\mu}_j \leftarrow \frac{\sum_{t=1}^{n} w_{tj} y_t}{\sum_{t=1}^{n} w_{tj}}$$

$$\hat{\Sigma}_j \leftarrow \frac{\sum_{t=1}^{n} w_{tj} (y_t - \hat{\mu}_j)(y_t - \hat{\mu}_j)^T}{\sum_{t=1}^{n} w_{tj}}$$

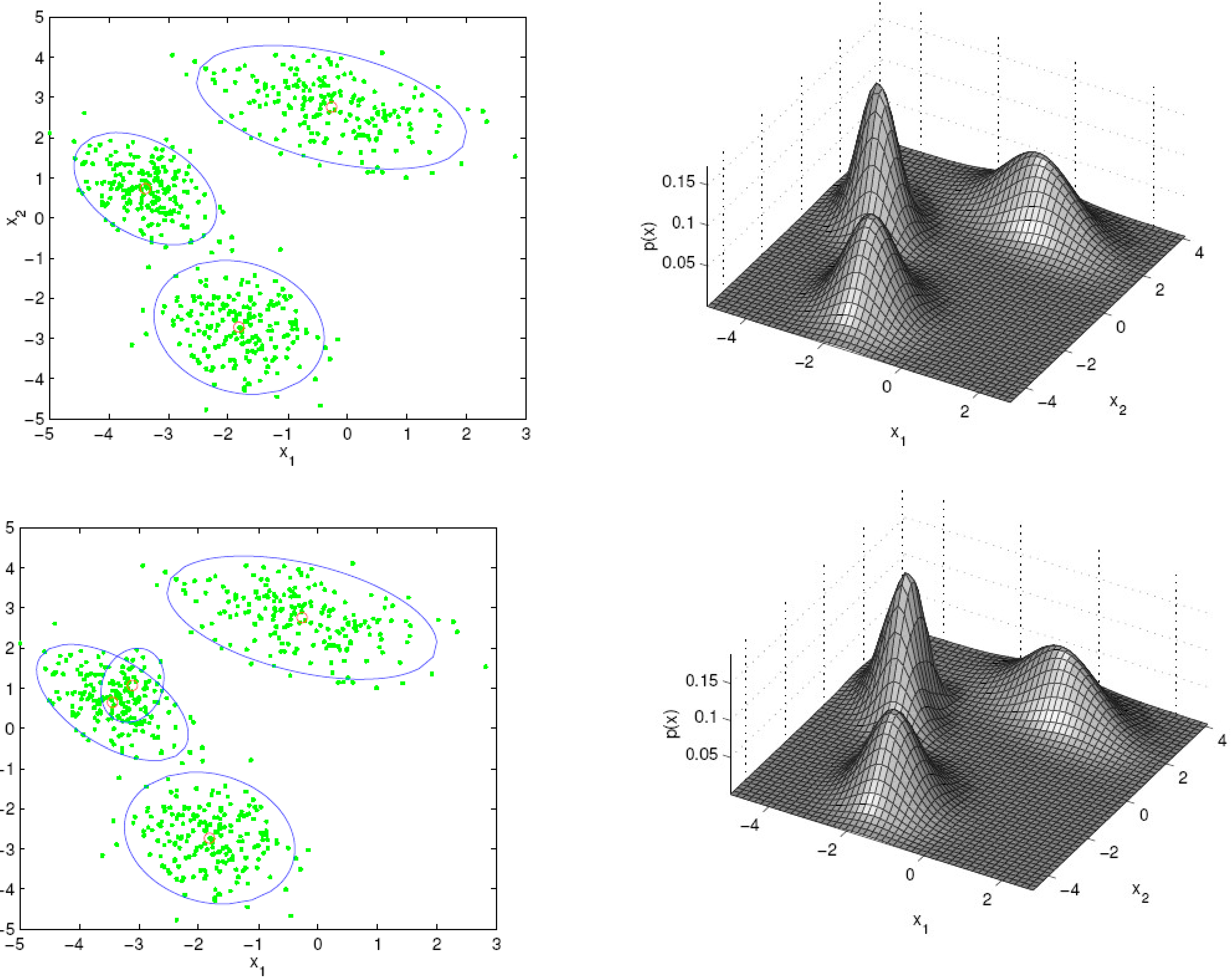
Initialize with k-means (or k-means++)

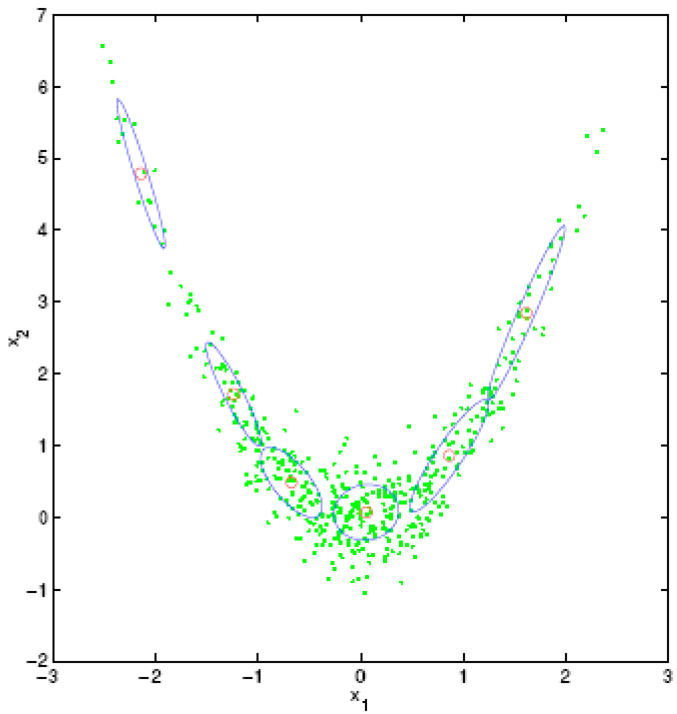


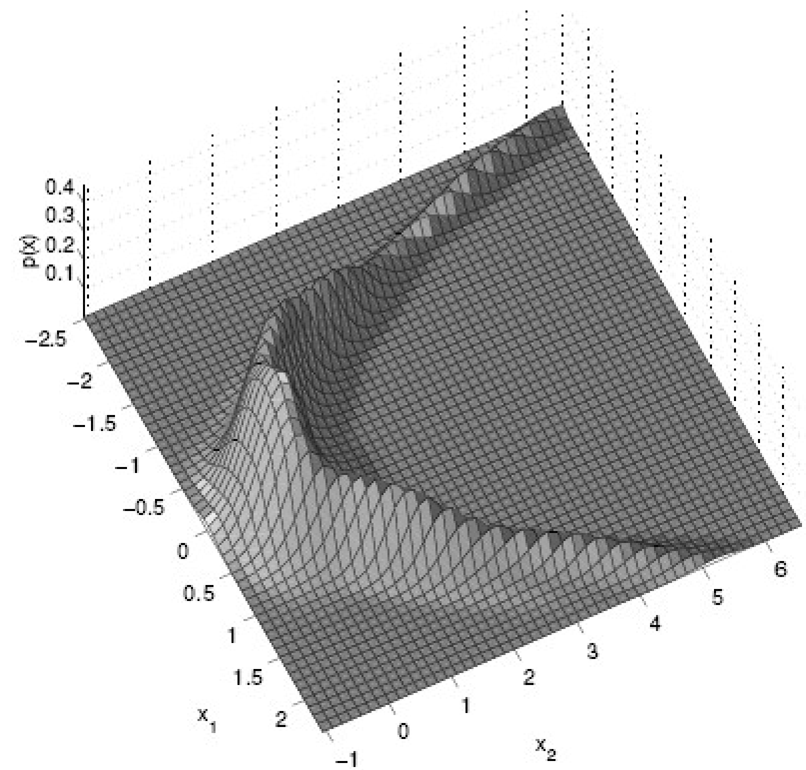
6
4
2
0
−2
−4
−6
−5
−4
−3
−2
−1
0
1
2
3
x1
x2



0.08
0.06
0.04
0.02
p(x)
−4
−2
0
2
x1
4
2
0
−2
−4
x2


x1
x2
p(x)


x1
x2
−3
−2
−1
0
1
2
3
7
6
5
4
3
2
1
0
−1
−2


p(x)
0.4
0.3
0.2
0.1
−2.5
−2
−1.5
−1
−0.5
0
0.5
1
1.5
2
x1
−1
0
1
2
3
4
5
6
x2