Application - Image Restoration

$$Y = X + D$$

X ← image, D ← noise

D - White $N(0, \sigma_d^2)$ noise (e.g. Thermal)

I) Estimation

A) ML estimation

X - deterministic

$$p_y(y|x) = \prod_{s\in S} \frac{1}{\sqrt{2\pi}\,\sigma_d} \exp\left\{-\frac{1}{2\sigma_d^2}(y_s - x_s)^2\right\}$$

x ← parameter

$$\hat{X}_{ML} = \arg\max_{X} \log p_y(y|x)$$

$$= \arg\max_{X} \sum_{s\in S} -\frac{1}{2}\left\{\frac{(y_s - x_s)^2}{\sigma_d^2} + \log(2\pi\sigma_d^2)\right\}$$

$= Y_s$
- Too noisy
- not reasonable.

B) MMSE estimation

X - $\mathbb{R}^N$ GMRF

$$p_x(x) = \frac{1}{(2\pi\sigma_x^2)^{N/2}} |B|^{1/2} \exp\left\{-\frac{1}{2\sigma_x^2} x^T B x\right\}$$

$$B_{ij} = \begin{cases} 1 & i=j \\ -g_{i-j} & \text{o.w.} \end{cases}$$

Lecture 10

$p_x(x)$ is the a priori distribution

$p_{x|y}(x|y) = ?$ a posteriori distribution
Gaussian

$$\log p_{x|y}(x|y) = \log p_{y|x}(y|x) + \log p_x(x) + c(y)$$

$$= -\frac{1}{2\sigma_d^2}\underbrace{(X-Y)^t(X-Y)}_{\sum_{s\in S}(x_s-y_s)^2} + -\frac{1}{2\sigma_x^2}X^t B X + c'(y)$$

$$= -\frac{1}{2}X^t\left(\frac{I}{\sigma_d^2}+\frac{B}{\sigma_x^2}\right)X + \frac{1}{\sigma_d^2}Y^t X + c''(y)$$

$$= -\frac{1}{2}(X-\mu)^t R^{-1}(X-\mu) + c'''(y)$$

$$R^{-1} = \left(\frac{I}{\sigma_d^2}+\frac{B}{\sigma_x^2}\right)$$

$$\mu = R\left(\frac{Y}{\sigma_d^2}\right)$$

$$p_{x|y}(x|y) \sim N(\mu, R)$$

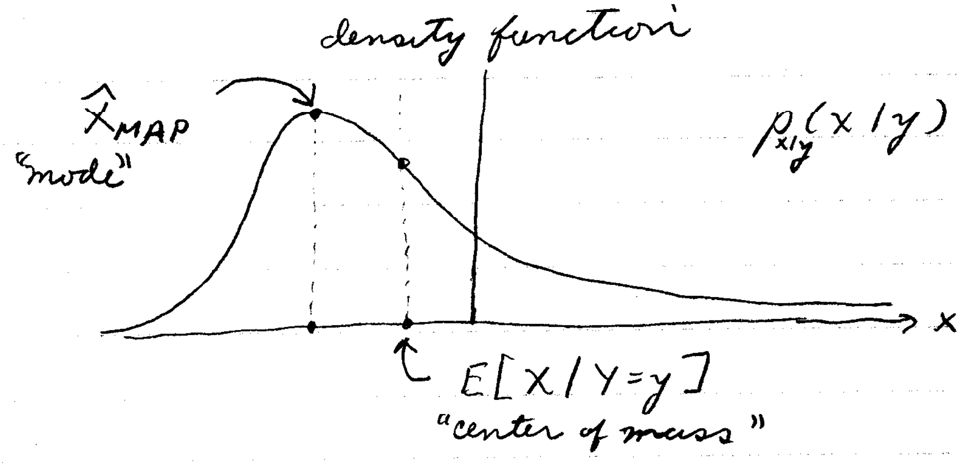
$$\hat{X}_{MMSE} = E[X|Y] = \mu$$
$$= R\left(\frac{Y}{\sigma_d^2}\right) \qquad \text{"Wiener Filter"}$$
$$= \underbrace{\left(\frac{I}{\sigma_d^2} + \frac{B}{\sigma_x^2}\right)^{-1}}_{256{,}000 \times 256{,}000 \text{ matrix}}\left(\frac{Y}{\sigma_d^2}\right)$$

c) Maximum a posteriori (MAP) estimate

$$\hat{X}_{MAP} = \underset{X}{\operatorname{argmax}}\ p_{x|y}(X|Y)$$

"most probable x given y"

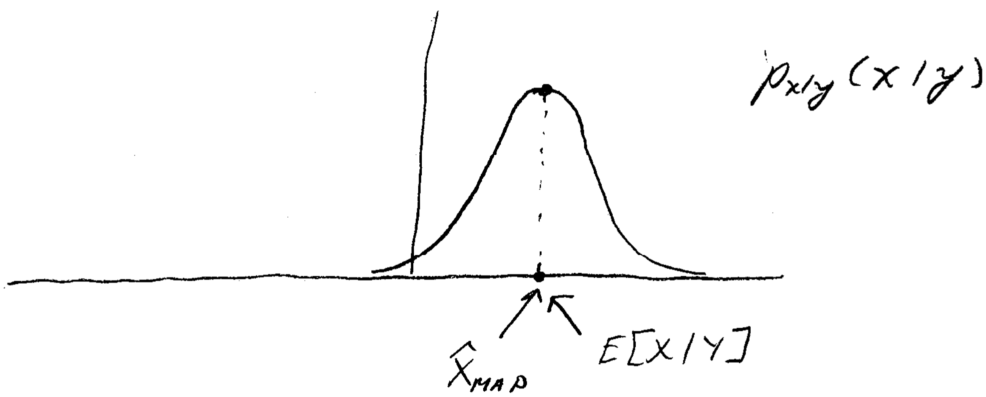
<u>Example</u> x, y scalars



If $p_{x|y}(x|y)$ is a symetric function of x then

$$\hat{X}_{MAP} = E[X|Y]$$

__Fact__: For XY Gaussian, $\hat{X}_{MAP} = \hat{X}_{MMSE}$



- Example

$$\hat{X}_{MAP} = \arg\max_{x}\left\{-\frac{1}{2}(x-\mu)^t \underset{\text{Positive definite}}{R^{-1}}(x-\mu) + c'''(y)\right\}$$

$$= \mu = \hat{X}_{MMSE}$$

$$= \arg\min_{x}\left\{(x-y)^t(x-y) + \underbrace{\frac{\sigma_a^2}{\sigma_x^2}}_{\left(\substack{\text{signal-to-noise}\\ \text{ratio}}\right)^{-1}} x^t B x\right\}$$

$$= \arg\min_{x} f(x)$$

How do we minimize this?

MAP estimation $\Leftrightarrow$ Optimization

II) Optimization of $f(x) = \|x-y\|^2 + \frac{\sigma_d^2}{\sigma_x^2} x^t B x$

- $f(x)$ is strictly convex (why?)
- $f(x)$ takes on its minimum value
- $f(x)$ has a unique global minimum at $\hat{x}_{MAP}$
- $\nabla_x f(x) = 0$ has a unique solution at $\hat{x}_{MAP}$

A) Gradient decent

$$\nabla_x f(x) = \quad (x-y)^t + \frac{\sigma_d^2}{\sigma_x^2} x^t B$$

$$x^{(k+1)} = x^{(k)} - \underset{\text{step size}}{\omega} \left(\nabla_x f(x^{(k)}) \right)^t$$

Since $B_{sr} = \delta_{s-r} - g_{s-r}$:

$$x_s^{(k+1)} = x_s^{(k)} + \omega 2 \left(y_s - x_s^{(k)} + \lambda \left(\sum_{r \neq s} g_r x_{s-r}^{(k)} - x_s^{(k)} \right) \right)$$

$$\lambda = \frac{\sigma_d^2}{\sigma_x^2}$$

- How do we pick ω?
- Poor convergence.

Lecture 11

B) "Coordinate decent" (optimization)
"Iterative Conditional Modes" (estimation)
"Gauss - Seidel" (differential equations)

$$f(x) = f(x_1, \ldots, x_N)$$

$$x_s^{(k+1)} = \begin{cases} \arg\min_{x_s} f(x) & s = k \bmod N \\ x_s^{(k)} & \text{o. w.} \end{cases}$$

$$\arg\min_{x_s} f(x) = ?$$

$$\frac{\partial}{\partial x_s}\left((x-y)^t(x-y) + \lambda\, x^t B x \right) = 0$$

$$= x_s - y_s + \lambda (x^t B)_s$$

$$= x_s - y_s + \lambda \left(x_s - \sum_{r \neq s} g_r x_{s-r} \right) = 0$$

$$(1+\lambda) x_s = y_s + \lambda \sum_{r \neq s} g_r x_{s-r}$$

$$x_s = \frac{y_s + \lambda \sum_{r \neq s} g_r x_{s-r}}{1+\lambda}$$

For any $f(x)$

$$f(x^{(k+1)}) \leq f(x^{(k)})$$

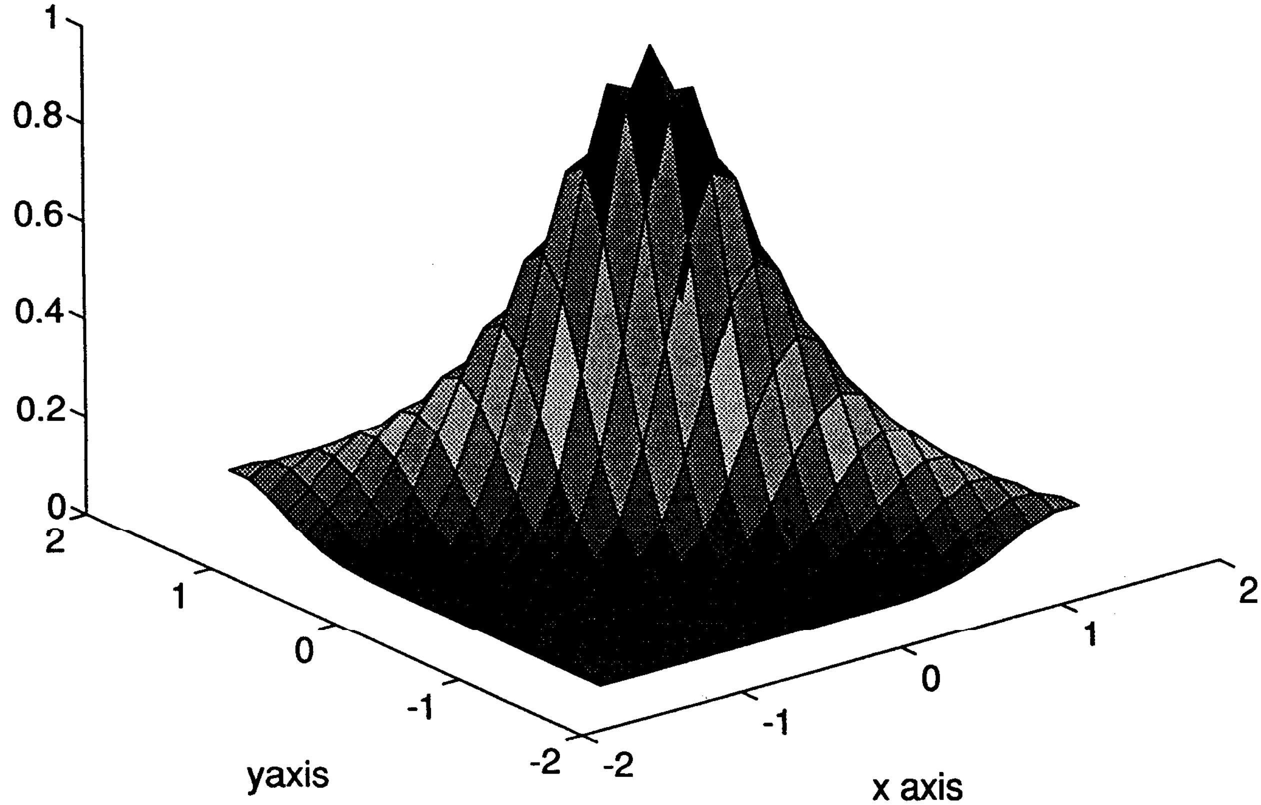
If f is strictly convex

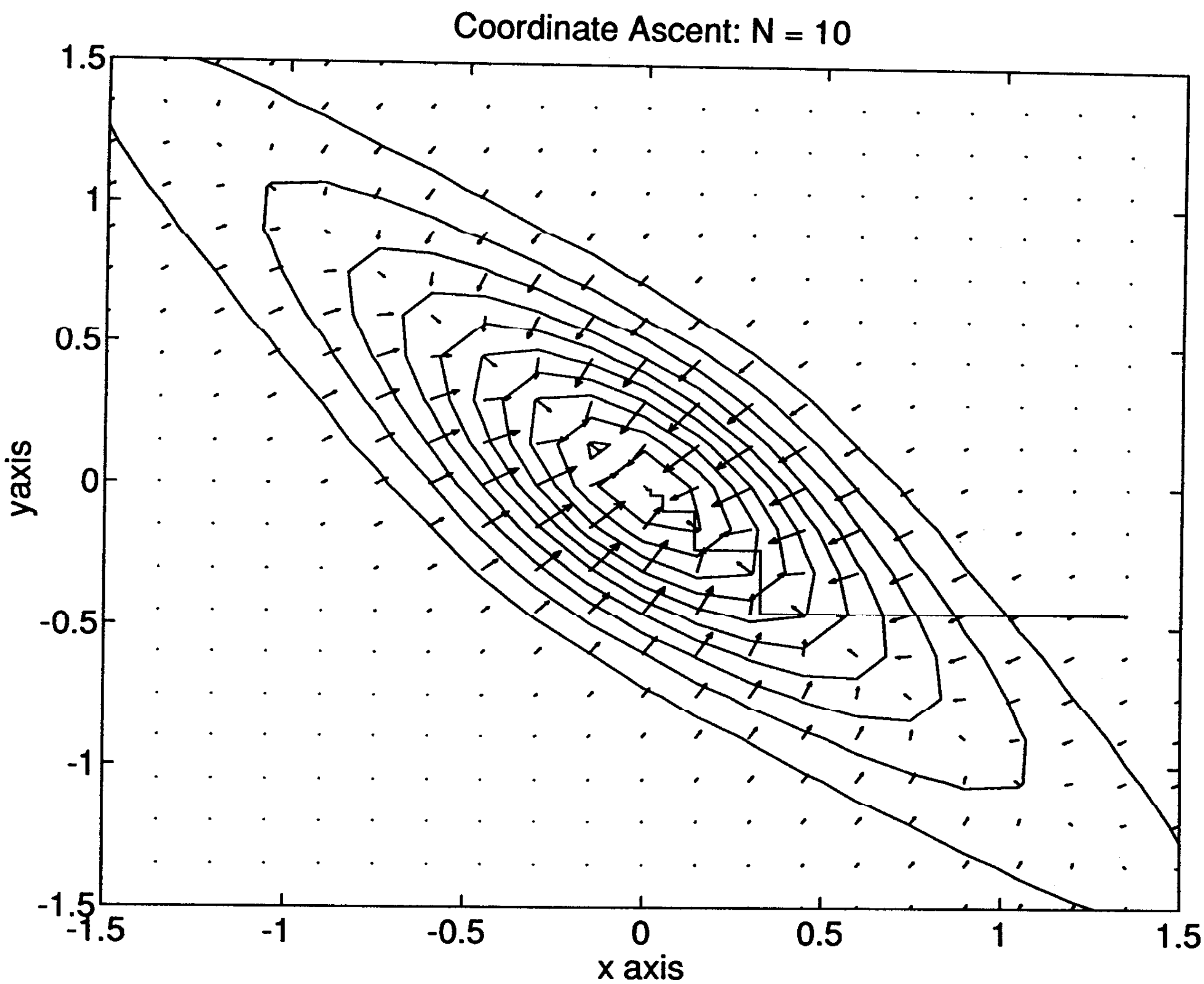
$$f(x^{(k+1)}) < f(x^{(k)})$$

If in addition, f is differentiable and takes on its minimum

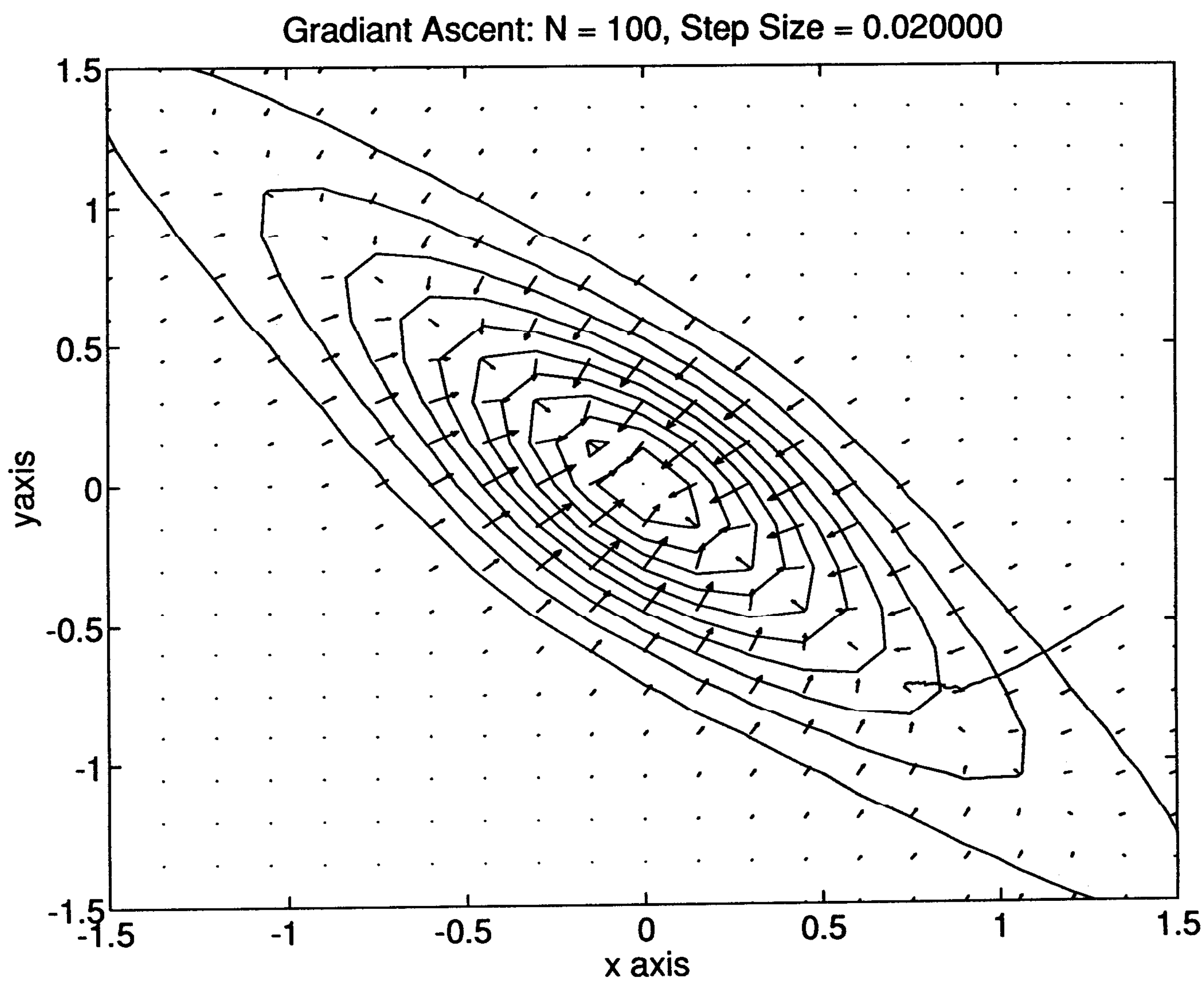
$$\lim_{k\to\infty} X^{(k)} = \hat{X}_{MAP}$$

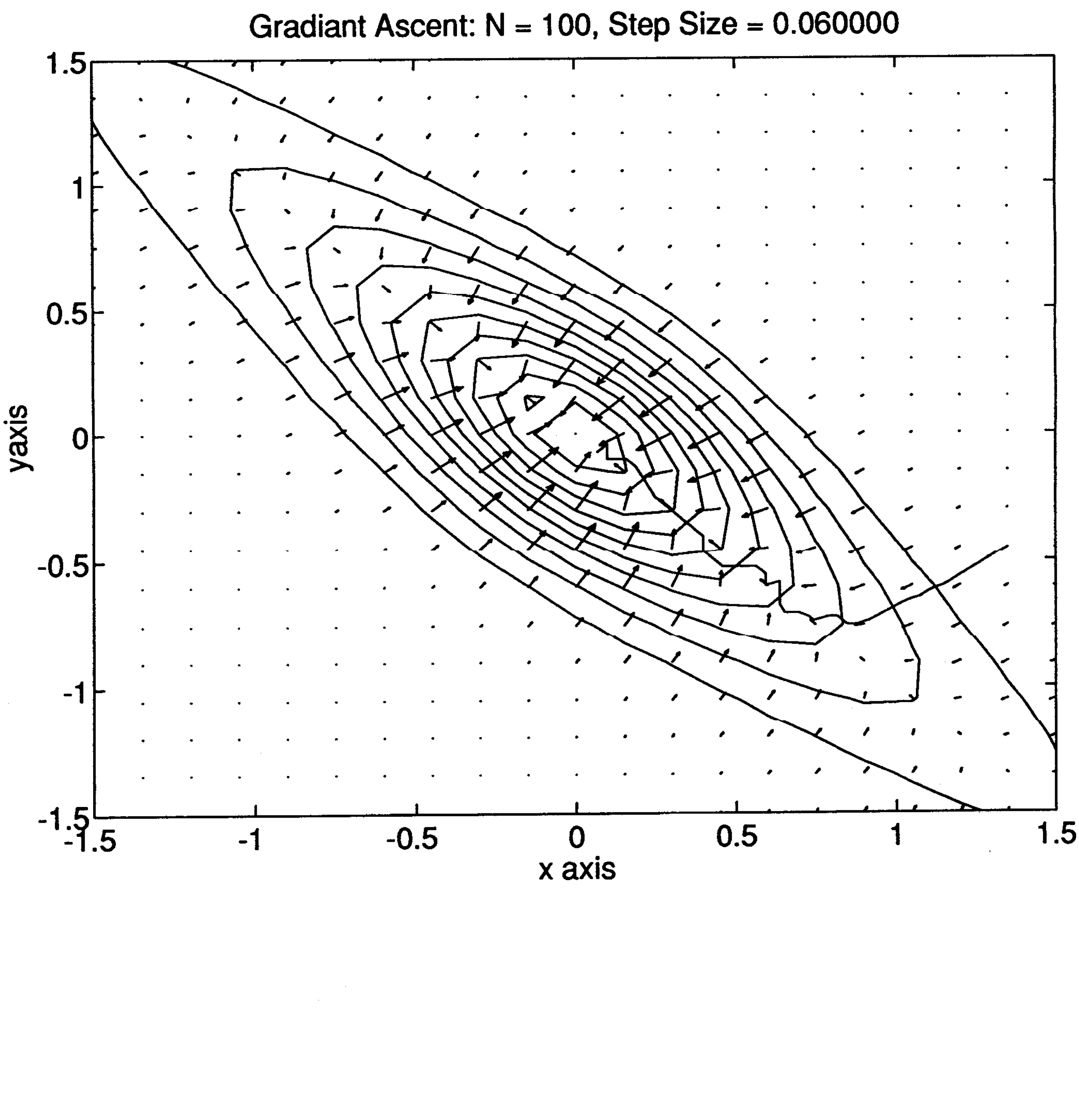
- Simple local iteration
- guaranteed convergence


1
0.8
0.6
0.4
0.2
0
2
1
0
-1
-2
yaxis
-2
-1
0
1
2
x axis

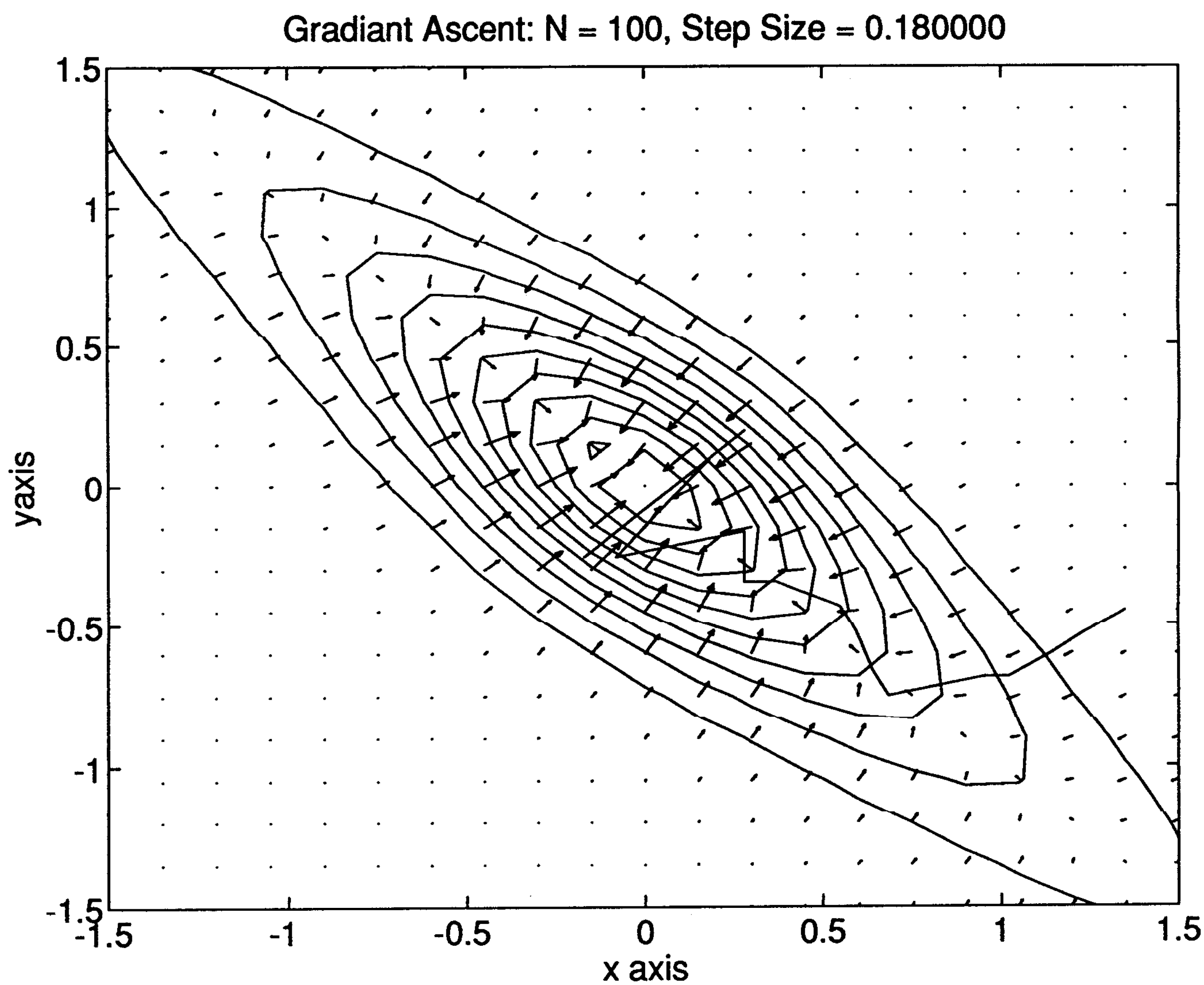


Coordinate Ascent: N = 10
1.5
1
0.5
0
-0.5
-1
-1.5
yaxis
-1.5
-1
-0.5
0
0.5
1
1.5
x axis


Gradiant Ascent: N = 100, Step Size = 0.020000
yaxis
x axis
1.5
1
0.5
0
-0.5
-1
-1.5
-1.5
-1
-0.5
0
0.5
1
1.5



Gradiant Ascent: N = 100, Step Size = 0.060000
1.5
1
0.5
0
-0.5
-1
-1.5
yaxis
-1.5
-1
-0.5
0
0.5
1
1.5
x axis


Gradiant Ascent: N = 100, Step Size = 0.180000
1.5
1
0.5
0
-0.5
-1
-1.5
yaxis
-1.5
-1
-0.5
0
0.5
1
1.5
x axis

Steepest Ascent: N = 10