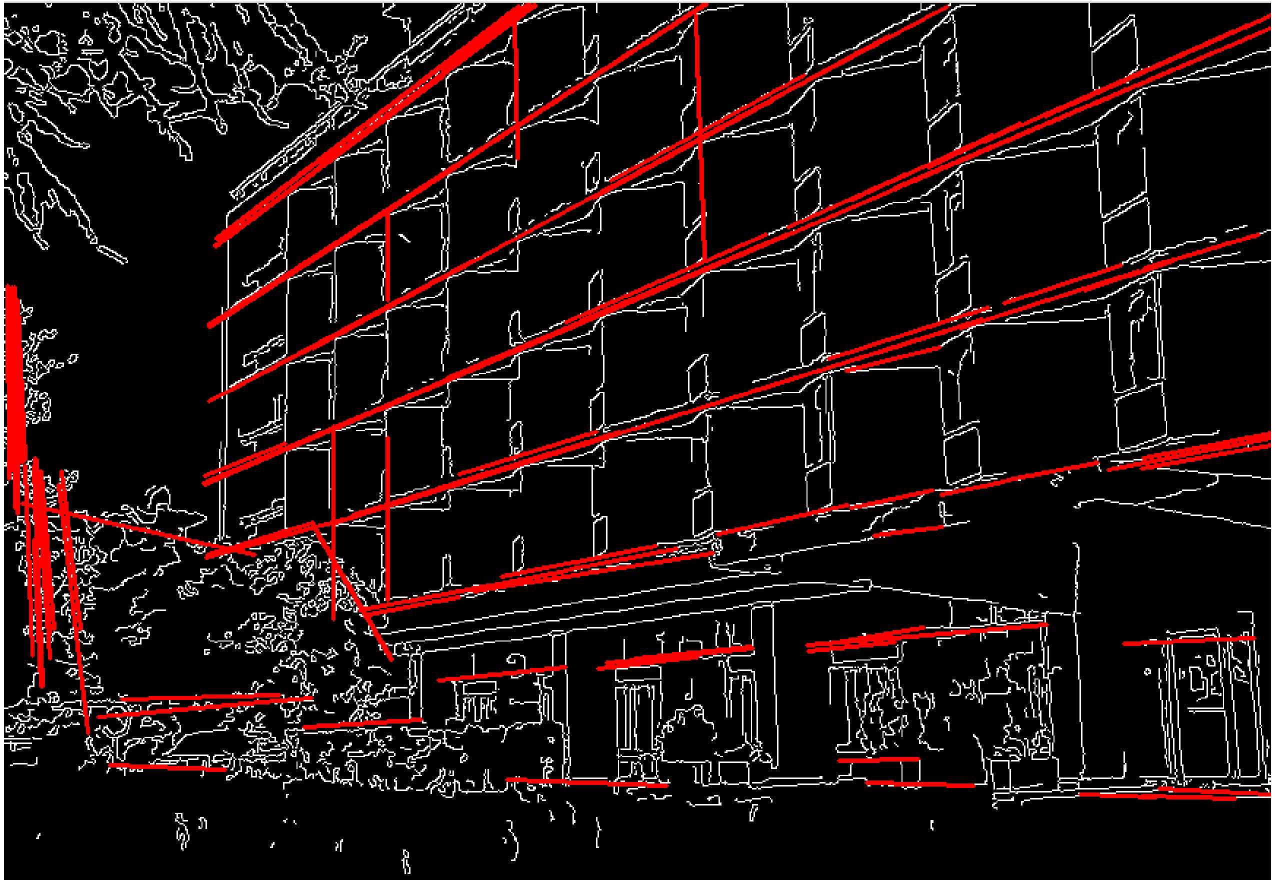
Fitting

and

Non-linear Optimization

Fitting

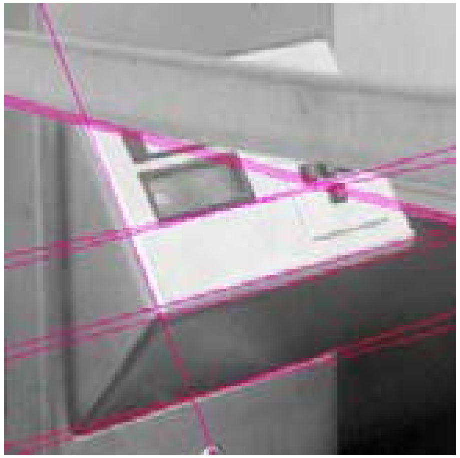


Fitting

- We've learned how to detect edges, corners, blobs. Now what?
- We would like to form a higher-level, more compact representation of the features in the image by grouping multiple features according to a simple model

Fitting

- Choose a parametric model to represent a set of features



simple model: lines

simple model: circles

complicated model: car

Source: K. Grauman

Fitting: Issues

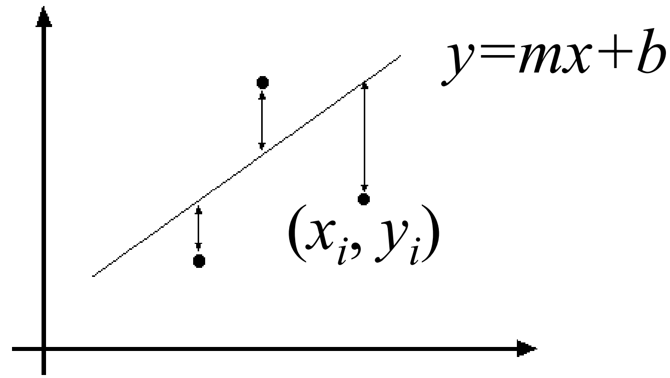
Case study: Line detection

- **Noise** in the measured feature locations
- **Extraneous data:** clutter (outliers), multiple lines
- **Missing data:** occlusions

Least squares line fitting

- Data: $(x_1, y_1), \ldots, (x_n, y_n)$
- Line equation: $y_i = m x_i + b$
- Find (m, b) to minimize

$$\boxed{E = \sum_{i=1}^{n} (y_i - mx_i - b)^2}$$



$$E = \|Y - XB\|^2 \quad \text{where} \quad Y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \qquad X = \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix} \qquad B = \begin{bmatrix} m \\ b \end{bmatrix}$$

$$E = \|Y - XB\|^2 = (Y - XB)^T (Y - XB) = Y^T Y - 2(XB)^T Y + (XB)^T (XB)$$

$$\frac{dE}{dB} = 2X^T XB - 2X^T Y = 0$$

$$\boxed{X^T XB = X^T Y}$$

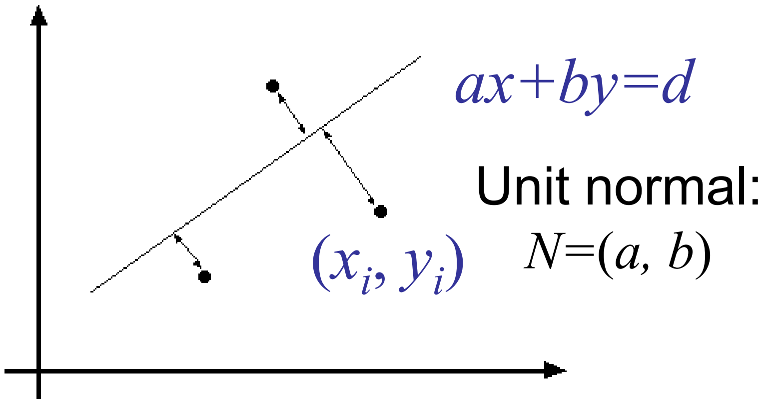
Normal equations: least squares solution to $XB=Y$

Problem with “vertical” least squares

- Not rotation-invariant
- Fails completely for vertical lines

Total least squares

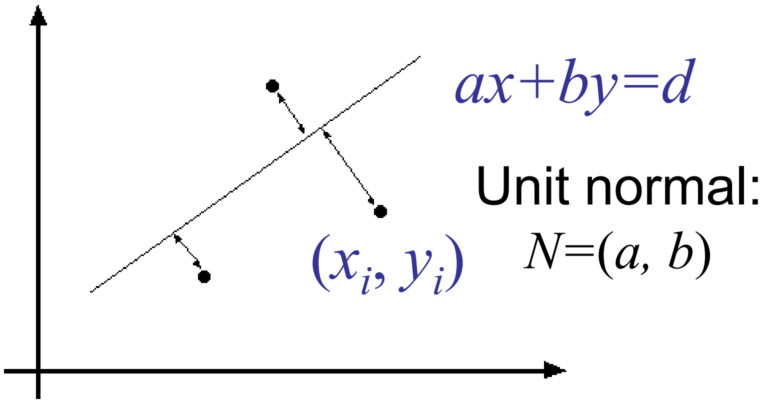
- Distance between point (x_i, y_i) and line $ax+by=d$ $(a^2+b^2=1)$: $|ax_i + by_i - d|$



Total least squares

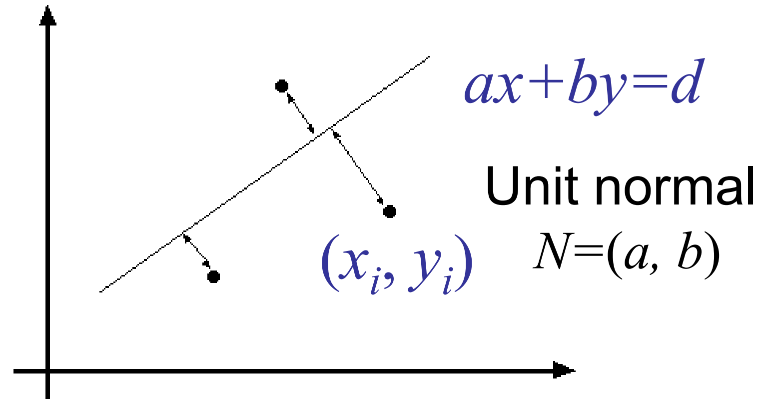
- Distance between point (x_i, y_i) and line $ax+by=d$ $(a^2+b^2=1)$: $|ax_i + by_i - d|$
- Find (a, b, d) to minimize the sum of squared perpendicular distances

$$\boxed{E = \sum_{i=1}^{n} (ax_i + by_i - d)^2}$$



Total least squares

- Distance between point (x_i, y_i) and line $ax+by=d$ $(a^2+b^2=1)$: $|ax_i + by_i - d|$
- Find (a, b, d) to minimize the sum of squared perpendicular distances



$$\boxed{E = \sum_{i=1}^{n} (ax_i + by_i - d)^2}$$

$$\frac{\partial E}{\partial d} = \sum_{i=1}^{n} -2(ax_i + by_i - d) = 0 \qquad d = \frac{a}{n}\sum_{i=1}^{n} x_i + \frac{b}{n}\sum_{i=1}^{n} y_i = a\bar{x} + b\bar{y}$$

$$E = \sum_{i=1}^{n} (a(x_i - \bar{x}) + b(y_i - \bar{y}))^2 = \left\| \begin{bmatrix} x_1 - \bar{x} & y_1 - \bar{y} \\ \vdots & \vdots \\ x_n - \bar{x} & y_n - \bar{y} \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} \right\|^2 = (UN)^T (UN)$$

$$\frac{dE}{dN} = 2(U^T U)N = 0$$

Solution to $(U^T U)N = 0$, subject to $||N||^2 = 1$: eigenvector of $U^T U$ associated with the smallest eigenvalue (least squares solution to *homogeneous linear system* $UN = 0$)

Total least squares

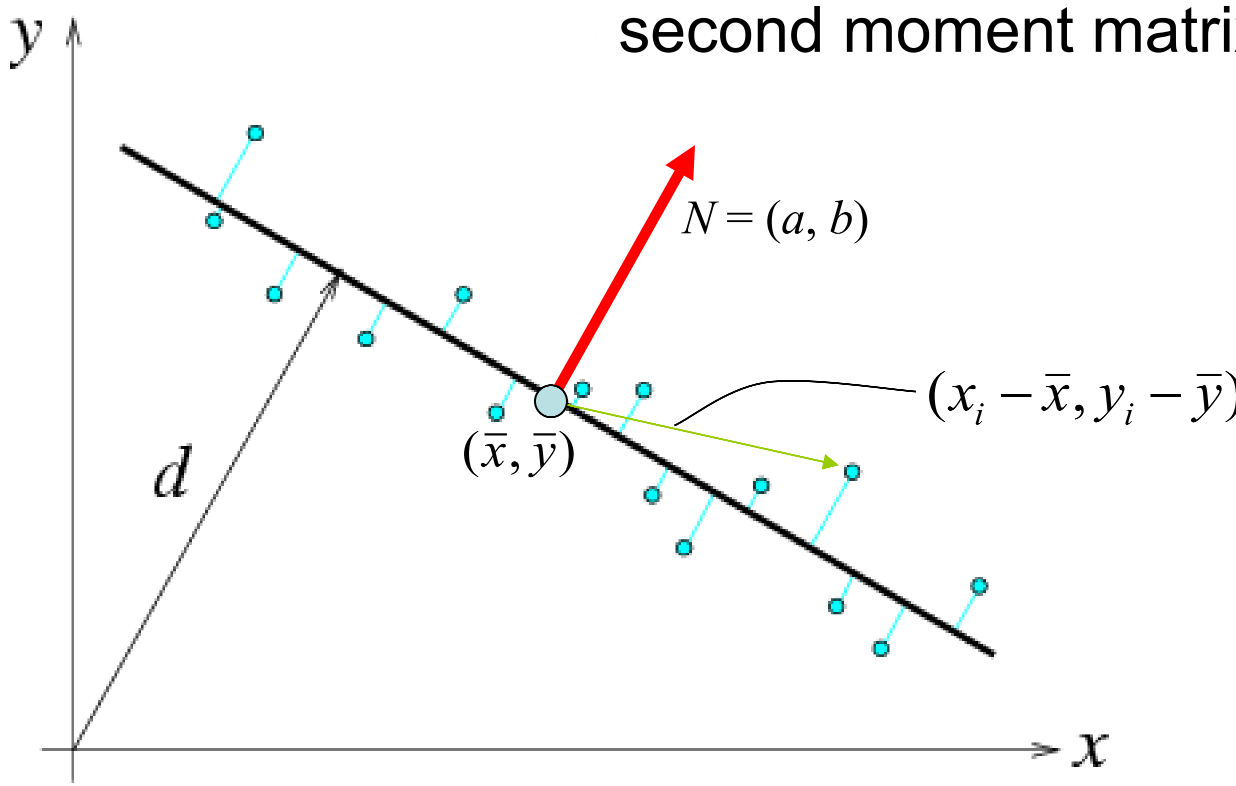
$$U = \begin{bmatrix} x_1 - \bar{x} & y_1 - \bar{y} \\ \vdots & \vdots \\ x_n - \bar{x} & y_n - \bar{y} \end{bmatrix} \qquad U^T U = \begin{bmatrix} \sum_{i=1}^{n} (x_i - \bar{x})^2 & \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) \\ \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) & \sum_{i=1}^{n} (y_i - \bar{y})^2 \end{bmatrix}$$

second moment matrix

Total least squares

$$U = \begin{bmatrix} x_1 - \bar{x} & y_1 - \bar{y} \\ \vdots & \vdots \\ x_n - \bar{x} & y_n - \bar{y} \end{bmatrix} \quad U^T U = \begin{bmatrix} \sum_{i=1}^{n} (x_i - \bar{x})^2 & \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) \\ \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) & \sum_{i=1}^{n} (y_i - \bar{y})^2 \end{bmatrix}$$

second moment matrix



F&P (2nd ed.) sec. 22.1

三維立體視覺(Stereo Vision)

image I(x,y)

Disparity map D(x,y)

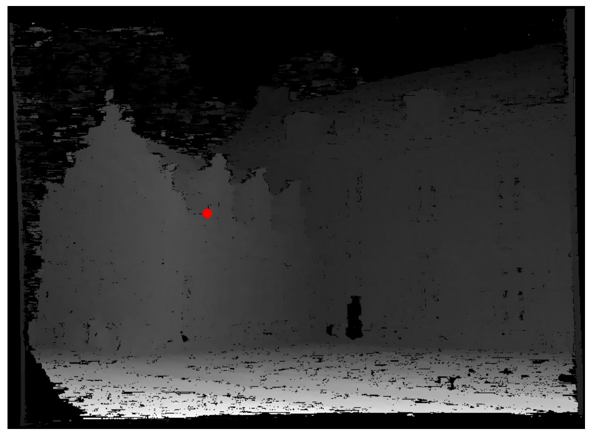
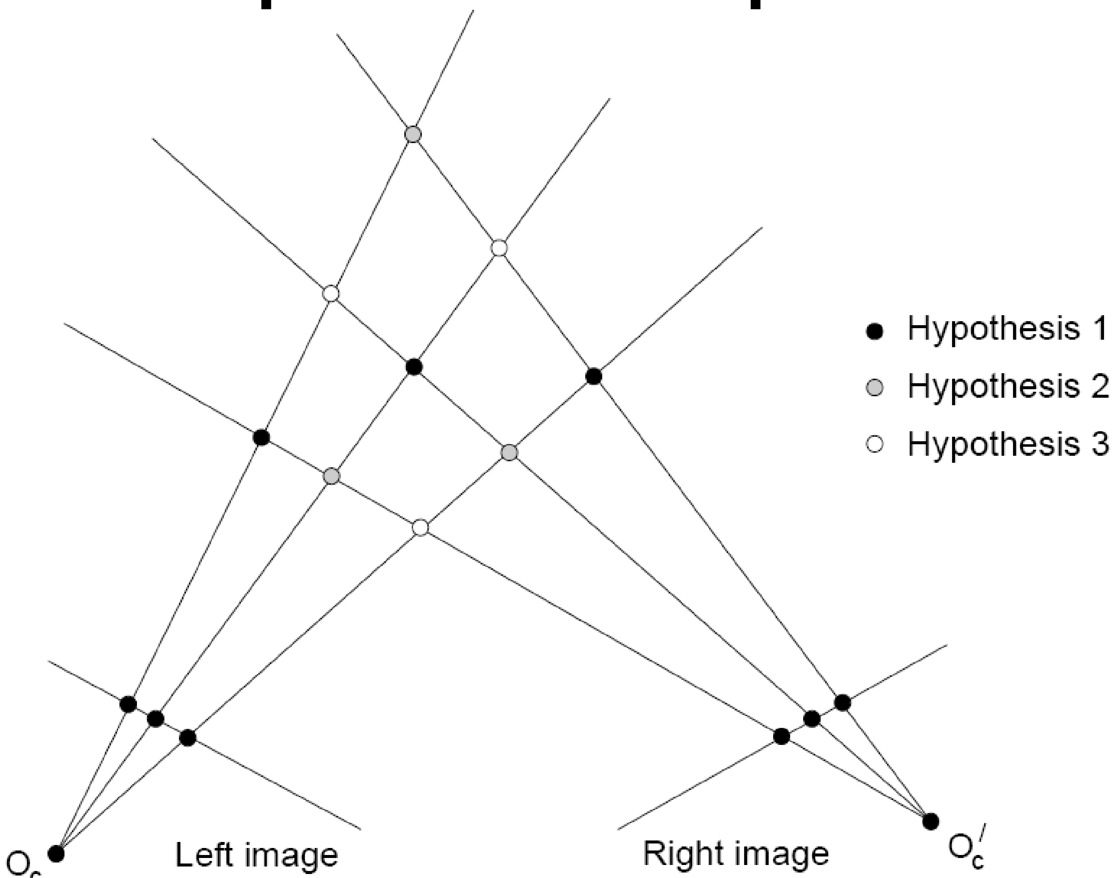


image I´(x´,y´)

$$(x',y')=(x+D(x,y),y)$$

Correspondence problem



$$x' = \frac{m_0 x + m_1 y + m_2}{m_6 x + m_7 y + 1}$$

$$y' = \frac{m_3 x + m_4 y + m_5}{m_6 x + m_7 y + 1}$$

Figure from Gee & Cipolla 1999

Why non-linear?

- The model is non-linear (e.g. joints, position, ..)

$$x' = \frac{m_0 x + m_1 y + m_2}{m_6 x + m_7 y + 1}$$

$$y' = \frac{m_3 x + m_4 y + m_5}{m_6 x + m_7 y + 1}$$

- The error function is non-linear

$$e(p,q,M^i) = \sqrt{\left(q^x - \frac{m_0^i p^x + m_1^i p^y + m_2^i}{m_6^i p^x + m_7^i p^y + 1}\right)^2 + \left(q^y - \frac{m_3^i p^x + m_4^i p^y + m_5^i}{m_6^i p^x + m_7^i p^y + 1}\right)^2}$$

Solve motion parameters

Minimizing the error function:

$$E(M) = \sum_i \left[I_1(x_i', y_i') - I_0(x_i, y_i)\right]^2 = \sum_i e_i^2$$

M can be obtained by the iteration form:

$$M^T \leftarrow M^T + \Delta M^T$$

$$\Delta M^T = (A + \lambda I)^{-1} B$$

$$A = [a_{kn}] = \left[\sum_i \frac{\partial e_i}{\partial m_k} \frac{\partial e_i}{\partial m_n}\right]$$

$$B = [b_k] = \left[\sum_i e_i \frac{\partial e_i}{\partial m_k}\right]$$

最佳化問題的Formulation

- Let f be a function such that

$$\boldsymbol{a} \in R^n \to f(\boldsymbol{a},\boldsymbol{x}) \in R$$

where $\boldsymbol{x}$ is a vector of parameters

$$For\ \text{example,}\ f(\boldsymbol{a},\boldsymbol{x}) = \sum_{k=0}^{n} a_k x^k$$

最佳化問題的Formulation

- Let f be a function such that

$$\boldsymbol{a} \in R^n \rightarrow f(\boldsymbol{a},\boldsymbol{x}) \in R$$

  where $\boldsymbol{x}$ is a vector of parameters

- Let $\{\boldsymbol{a}_k, b_k\}$ be a set of measurements/constraints. We fit f to the data by solving:

$$\min_{\boldsymbol{x}} \frac{1}{2} \sum_k \left(b_k - f\left(\boldsymbol{a}_k, \boldsymbol{x}\right)\right)^2$$

最佳化問題

- Let f be a function such that

$$\boldsymbol{a} \in R^n \to f(\boldsymbol{a},\boldsymbol{x}) \in R$$

where $\boldsymbol{x}$ is a vector of parameters

- Let $\{\boldsymbol{a}_k, b_k\}$ be a set of measurements/constraints. We fit f to the data by solving:

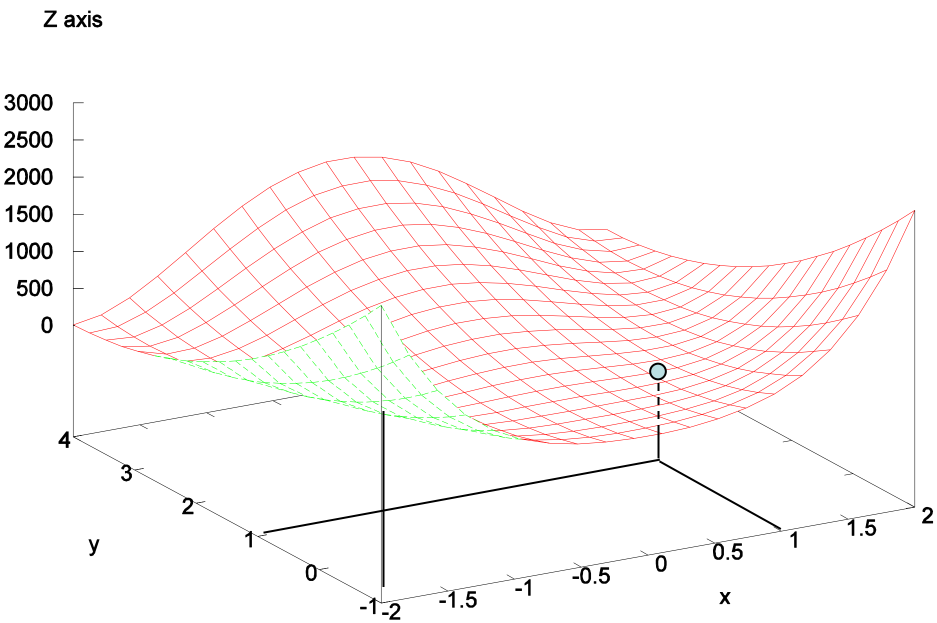
$$\min_{\boldsymbol{x}} \frac{1}{2} \sum_k \left(b_k - f(\boldsymbol{a}_k, \boldsymbol{x})\right)^2 \quad \text{or} \quad \min_{\boldsymbol{x}} \sum_k r_k{}^2$$

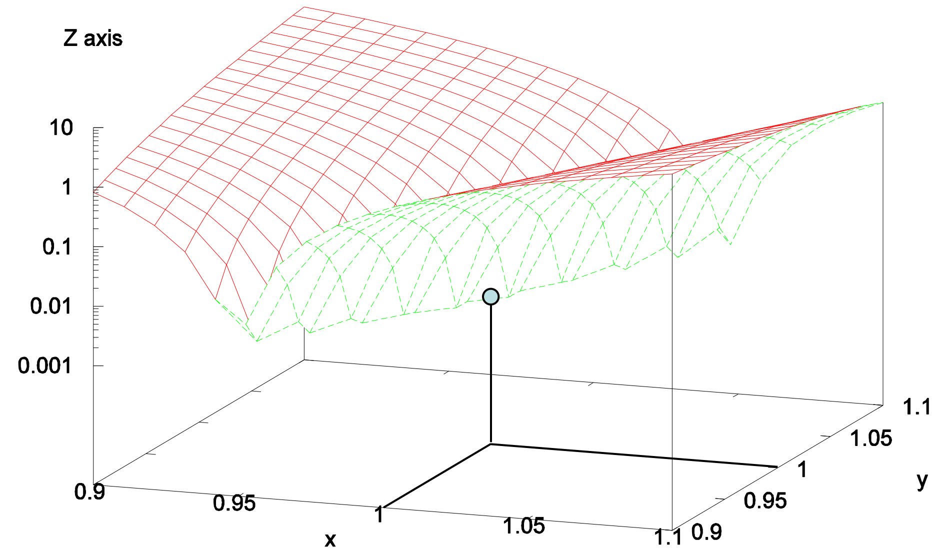
with $r_k = b_k - f(\boldsymbol{a}_k, \boldsymbol{x})$

最佳化問題的各種方法

- Existence and uniqueness of minimum
- Steepest-descent
- Newton’s method
- Gauss-Newton’s method
- Levenberg-Marquardt method
- Conjugate Gradient Method

A non-linear function: the Rosenbrock function





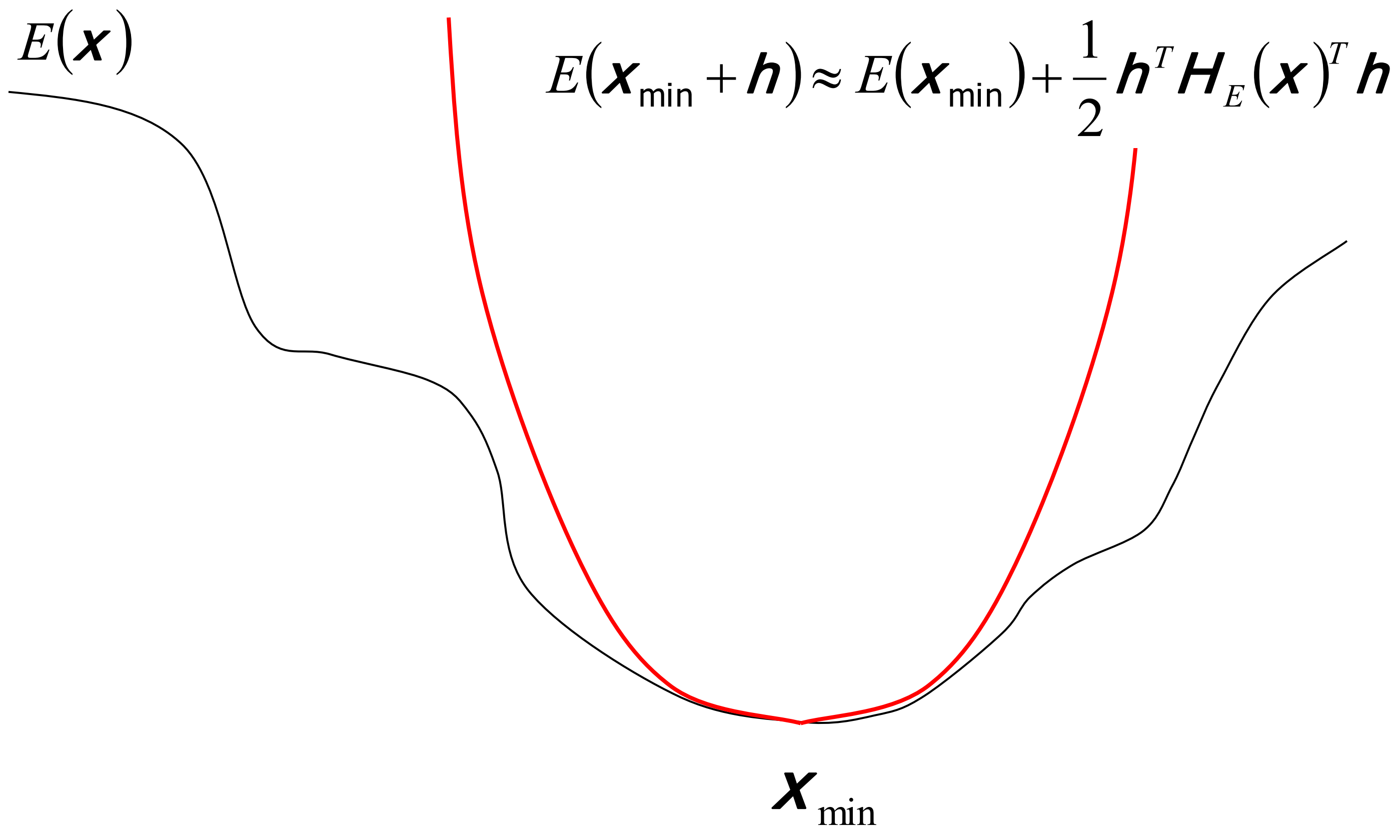
$$z = f(x, y) = (1 - x^2)^2 + 100(y - x^2)^2$$

Global minimum at *(1,1)*

Existence of minimum

$$E(\boldsymbol{x}_{\min} + \boldsymbol{h}) \approx E(\boldsymbol{x}_{\min}) + \boldsymbol{h}^T \Delta E(\boldsymbol{x}_{\min}) + \frac{1}{2}\boldsymbol{h}^T \boldsymbol{H}_E(\boldsymbol{x})^T \boldsymbol{h}$$

$\because \qquad \boldsymbol{x}_{\min}) = 0$ when a minimum is arrived.



Existence of minimum

A local minima is characterized by:

1. $\nabla E(\boldsymbol{x}_{\min}) = \mathbf{0}$

2. $\boldsymbol{h}^T \boldsymbol{H}_E(\boldsymbol{x}_{\min})^T \boldsymbol{h} \geq 0,$ for all $\boldsymbol{h}$ small enough (e.g. $\boldsymbol{H}_E(\boldsymbol{x}_{\min})$ is positive semi-definite)

Descent algorithm

- Start at an initial position $\boldsymbol{x}_0$
- Until convergence
  - Find minimizing step $d\boldsymbol{x}_k$
  - $\boldsymbol{x}_{k+1}=\boldsymbol{x}_\mathrm{k}+ d\boldsymbol{x}_k$

Produce a sequence $\boldsymbol{x}_0$, $\boldsymbol{x}_1$, …, $\boldsymbol{x}_\mathrm{n}$ such that

$$f(\boldsymbol{x}_0) > f(\boldsymbol{x}_1) > \ldots > f(\boldsymbol{x}_\mathrm{n})$$

Descent algorithm

- Start at an initial position $\boldsymbol{x}_0$
- Until convergence
  - Find minimizing step $d\boldsymbol{x}_k$
    using a local approximation of f
  - $\boldsymbol{x}_{k+1}=\boldsymbol{x}_\mathrm{k}+ d\boldsymbol{x}_k$

Produce a sequence $\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_\mathrm{n}$ such that

$$f(\boldsymbol{x}_0) > f(\boldsymbol{x}_1) > \ldots > f(\boldsymbol{x}_\mathrm{n})$$

Approximation using Taylor series

$$E(\boldsymbol{x}+\boldsymbol{h}) = E(\boldsymbol{x}) + \nabla E(\boldsymbol{x})^T \boldsymbol{h} + \frac{1}{2}\boldsymbol{h}^T \boldsymbol{H}_E(\boldsymbol{x})^T \boldsymbol{h} + \boldsymbol{O}\left(\|\boldsymbol{h}\|^2\right)$$

$$H_E(X) = \begin{bmatrix} \frac{\partial^2 E}{\partial x_1^2} & \frac{\partial^2 E}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 E}{\partial x_n} \\ \frac{\partial^2 E}{\partial x_2 \partial x_1} & \frac{\partial^2 E}{\partial x_2 \partial x_2} & & \frac{\partial^2 E}{\partial x_2 \partial x_n} \\ \cdots\cdot & \cdots & \cdots & \cdots \\ \frac{\partial^2 E}{\partial x_n \partial x_1} & \frac{\partial^2 E}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 E}{\partial x_n \partial x_n} \end{bmatrix} (\mathit{Hessian}\ \text{Matrix})$$

Approximation using Taylor series

$$E(\boldsymbol{x}+\boldsymbol{h}) = E(\boldsymbol{x}) + \nabla E(\boldsymbol{x})^T \boldsymbol{h} + \frac{1}{2}\boldsymbol{h}^T \boldsymbol{H}_E(\boldsymbol{x})^T \boldsymbol{h} + \boldsymbol{O}\left(\|\boldsymbol{h}\|^2\right)$$

with $E(\boldsymbol{x}) = \frac{1}{2}\sum_{j=1}^{m} r_j^2$, $\boldsymbol{J} = \left[\frac{\partial r_j}{\partial x_i}\right]$ and $\boldsymbol{H}_{r_j} = \left[\frac{\partial^2 r_j}{\partial x_i}\right]$

Approximation using Taylor series

$$E(\boldsymbol{x}+\boldsymbol{h}) = E(\boldsymbol{x}) + \nabla E(\boldsymbol{x})^T \boldsymbol{h} + \frac{1}{2}\boldsymbol{h}^T \boldsymbol{H}_E(\boldsymbol{x})^T \boldsymbol{h} + \boldsymbol{O}\left(\|\boldsymbol{h}\|^2\right)$$

with $E(\boldsymbol{x}) = \frac{1}{2}\sum_{j=1}^{m} r_j^2$, $\boldsymbol{J} = \left[\frac{\partial r_j}{\partial x_i}\right]$ and $\boldsymbol{H}_{r_j} = \left[\frac{\partial^2 r_j}{\partial x_i}\right]$

$$\nabla E = \sum_{j=1}^{m} r_j \nabla r_j = \boldsymbol{J}^T \boldsymbol{r}$$

$$\boldsymbol{H}_E = \sum_{j=1}^{m} \nabla r_j \nabla r_j^{\,T} + \sum_{j=1}^{m} r_j \boldsymbol{H}_{r_j}^T = \boldsymbol{J}^T \boldsymbol{J} + \sum_{j=1}^{m} r_j \boldsymbol{H}_{r_j}^T$$

Steepest descent

$$f(\boldsymbol{x}+\boldsymbol{h}) \simeq \qquad \boldsymbol{x} \qquad \nabla f(\boldsymbol{x})^{T} \boldsymbol{h}$$

$$\textbf{\textit{if}} \;\; \boldsymbol{h} = -\alpha \nabla f(\boldsymbol{x}), f(\boldsymbol{x}+\boldsymbol{h}) \leq f(\boldsymbol{x})$$

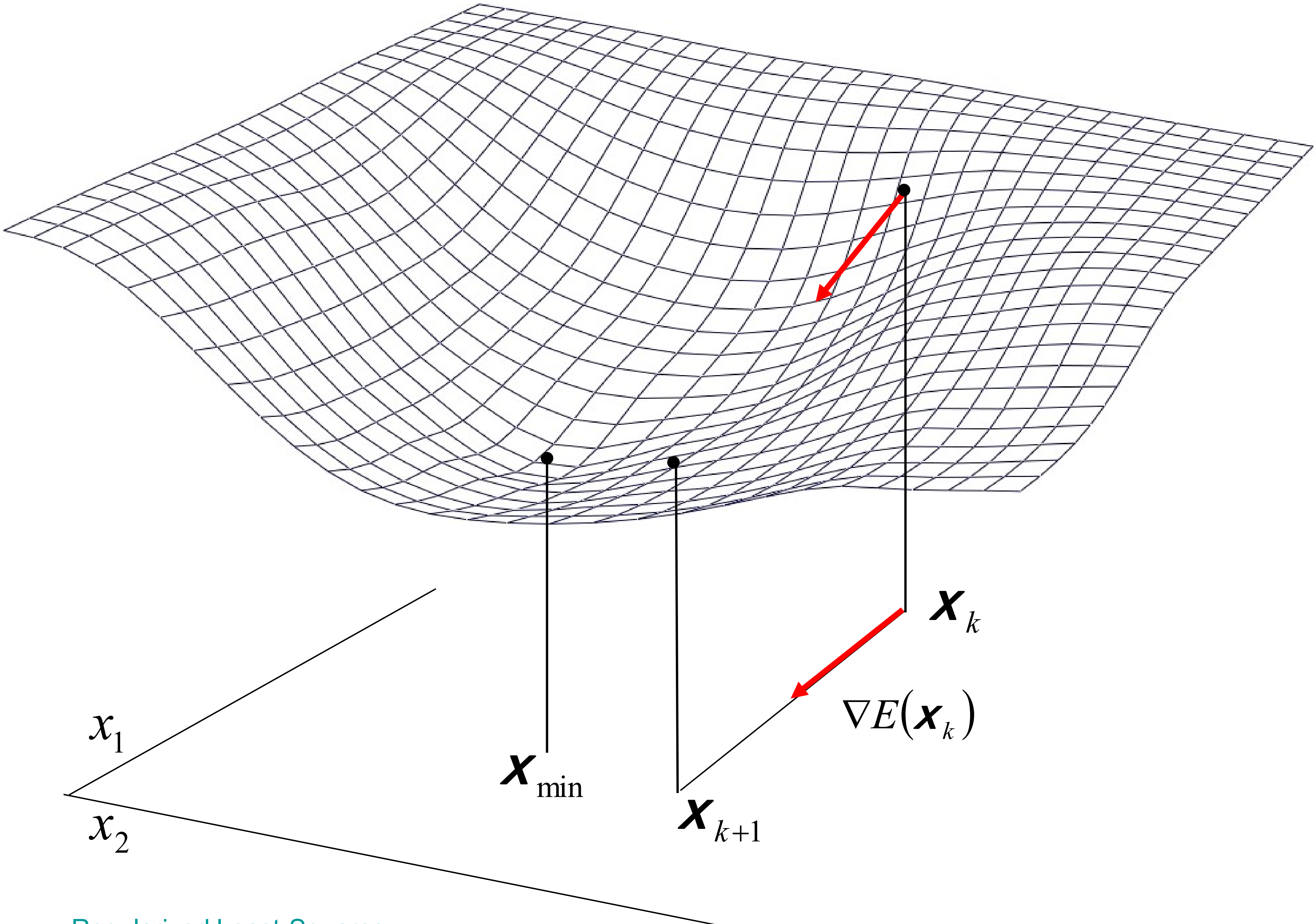
Thus, we have the steepest descent

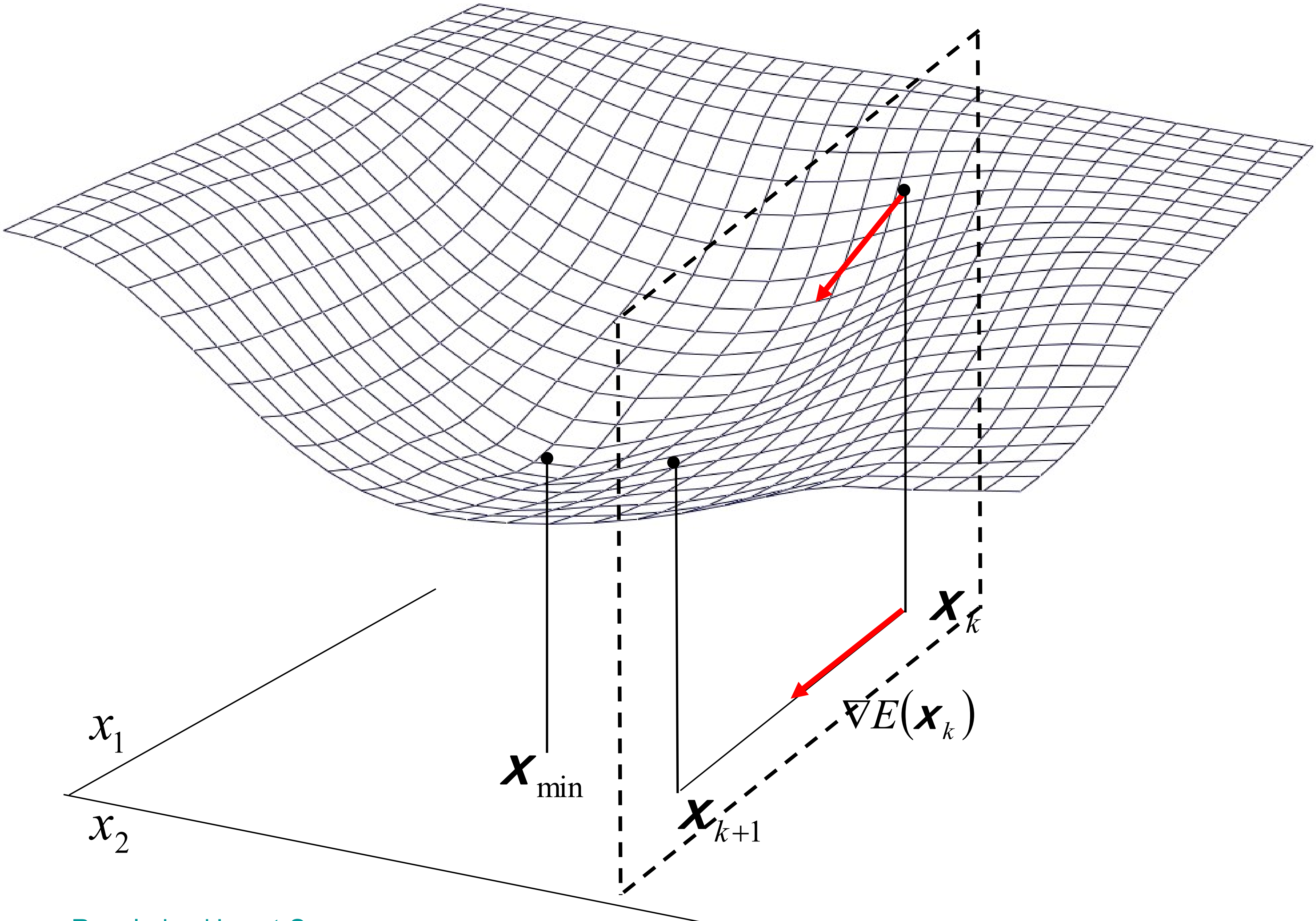
Step $\quad \boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \alpha \nabla E(\boldsymbol{x}_k)$

where α is chosen such that: $\boldsymbol{f}(\boldsymbol{x}_{k+1}) < \boldsymbol{f}(\boldsymbol{x}_k)$

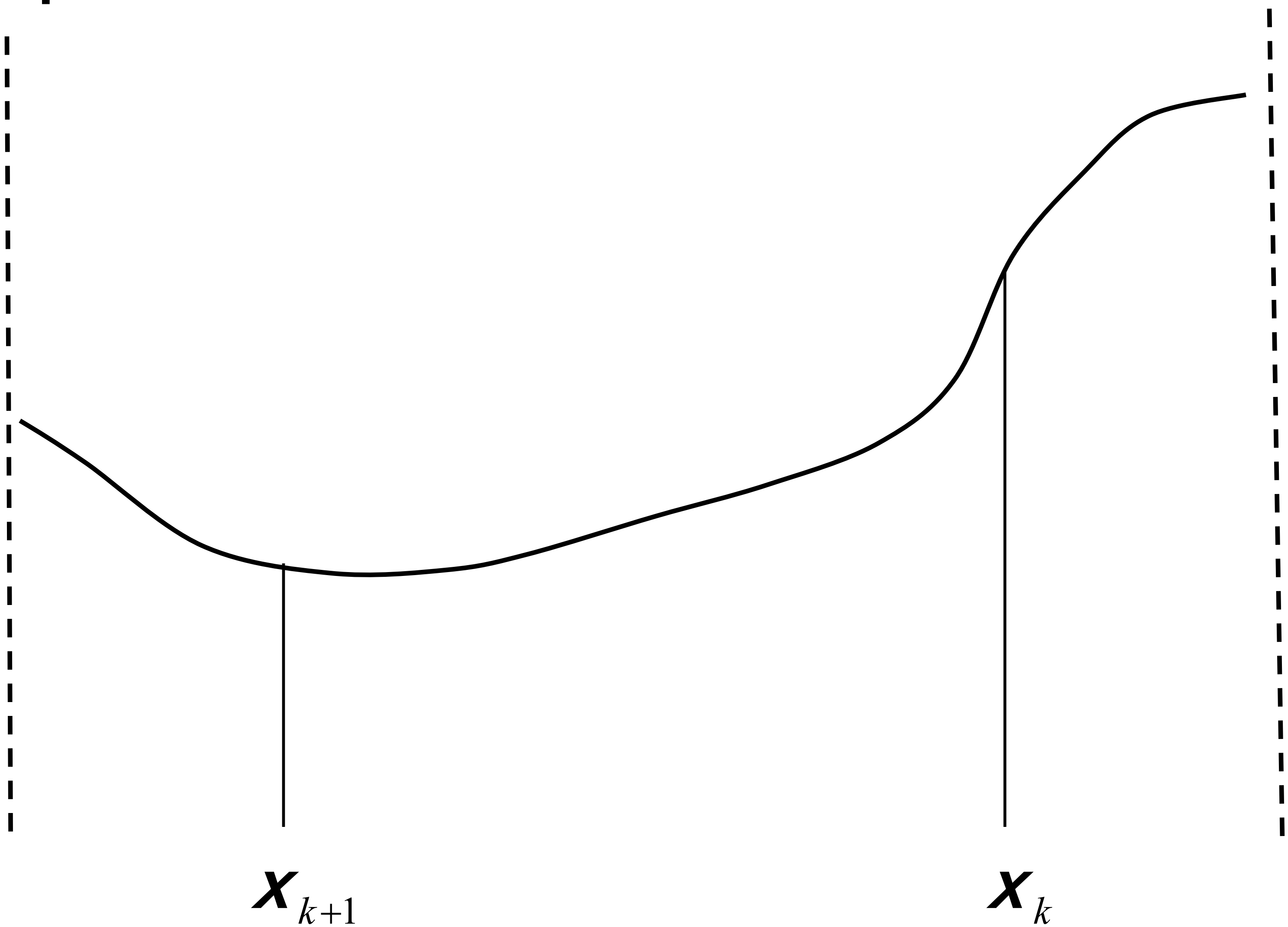
using a line search algorithm:

$$\min_{\alpha} f\left({}_k\boldsymbol{x}_k - \alpha \nabla E(\boldsymbol{x}_k)\right)$$


$\boldsymbol{x}_k$
$\nabla E(\boldsymbol{x}_k)$
x_1
x_2
$\boldsymbol{x}_{\min}$
$\boldsymbol{x}_{k+1}$


x_1
x_2
$\boldsymbol{x}_{\min}$
$\boldsymbol{x}_{k+1}$
$\boldsymbol{x}_k$
$\nabla E(\boldsymbol{x}_k)$

In the plane of the steepest descent direction



Newton's method

- Step: second order approximation

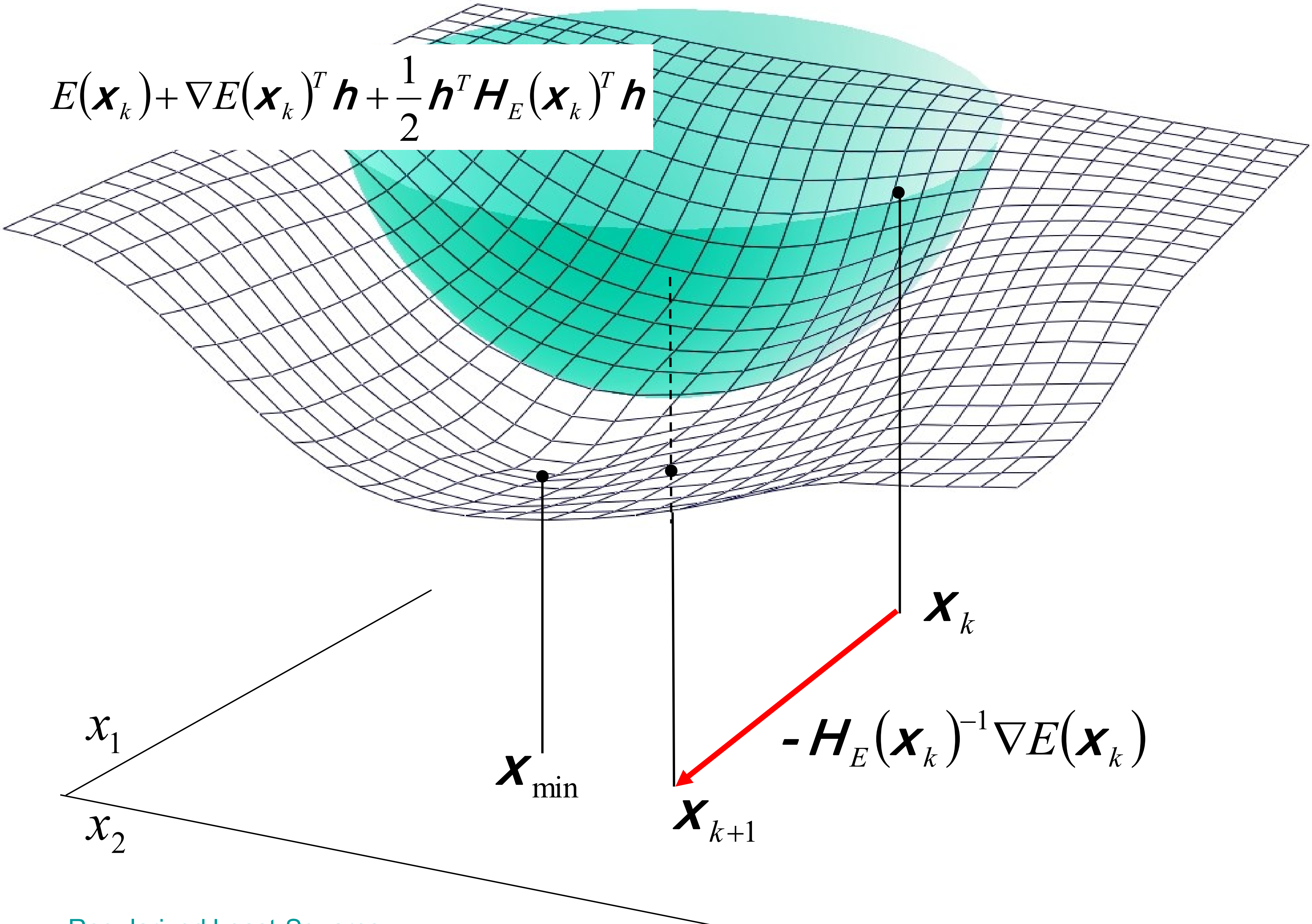
$$E(\boldsymbol{x}_k + \boldsymbol{h}) \approx E(\boldsymbol{x}_k) + \nabla E(\boldsymbol{x}_k)^T \boldsymbol{h} + \frac{1}{2}\boldsymbol{h}^T \boldsymbol{H}_E(\boldsymbol{x}_k)^T \boldsymbol{h}$$

$$At \text{ the minimum}, \frac{\partial\{E(\boldsymbol{x}_k + \boldsymbol{h}) - E(\boldsymbol{x}_k)\}}{\partial h} = \mathbf{0}$$

$$\Rightarrow \nabla E(\boldsymbol{x}_k) + \boldsymbol{H}_E(\boldsymbol{x}_k)\boldsymbol{h} = \boldsymbol{0}$$

$$\Rightarrow \boldsymbol{h} = -\boldsymbol{H}_E(\boldsymbol{x}_k)^{-1} \nabla E(\boldsymbol{x}_k)$$

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \boldsymbol{H}_E(\boldsymbol{x}_k)^{-1} \nabla E(\boldsymbol{x}_k)$$


$E(\boldsymbol{x}_k)+\nabla E(\boldsymbol{x}_k)^T\boldsymbol{h}+\frac{1}{2}\boldsymbol{h}^T\boldsymbol{H}_E(\boldsymbol{x}_k)^T\boldsymbol{h}$
$\boldsymbol{x}_k$
$-\boldsymbol{H}_E(\boldsymbol{x}_k)^{-1}\nabla E(\boldsymbol{x}_k)$
x_1
x_2
$\boldsymbol{x}_{\min}$
$\boldsymbol{x}_{k+1}$

Problem

- If $\boldsymbol{H}_E(\boldsymbol{x}_k)$ is not positive semi-definite, then $-\boldsymbol{H}_E(\boldsymbol{x}_k)^{-1}\nabla E(\boldsymbol{x}_k)$ is not a descent direction: the step increases the error function

- Uses positive semi-definite approximation of Hessian based on the jacobian
  (quasi-Newton methods)

Gauss-Newton method

Step: use

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \boldsymbol{H}_E(\boldsymbol{x}_k)^{-1} \nabla E(\boldsymbol{x}_k)$$

with the approximate hessian

$$\boldsymbol{H}_E = \boldsymbol{J}^T \boldsymbol{J} + \sum_{j=1}^{m} r_j \boldsymbol{H}_{r_j}^T \approx \boldsymbol{J}^T \boldsymbol{J}$$

Advantages:

- No second order derivatives
- $\boldsymbol{J}^T \boldsymbol{J}$ is positive semi-definite

Levenberg-Marquardt algorithm

- Blends Steepest descent and Gauss-Newton
- At each step solve, for the descent direction $\boldsymbol{h}$

$$\left(\boldsymbol{J}^T\boldsymbol{J} + \lambda \boldsymbol{I}\right)\boldsymbol{h} = -\nabla E\left(\boldsymbol{x}_k\right)$$

if λ large

$$\boldsymbol{h} \propto -\nabla E\left(\boldsymbol{x}_k\right) \qquad \text{(steepest descent)}$$

if λ small

$$\boldsymbol{h} \propto -\left(\boldsymbol{J}^T\boldsymbol{J}\right)^{-1}\nabla E\left(\boldsymbol{x}_k\right) \qquad \text{(Gauss-Newton)}$$

Managing the damping parameter λ

- General approach:
  - If step fails, increase damping until step is successful
  - If step succeeds, decrease damping to take larger step

- Improved damping

$$\left(\boldsymbol{J}^T\boldsymbol{J} + \lambda \mathrm{diag}(\boldsymbol{J}^T\boldsymbol{J})\right)\boldsymbol{h} = -\nabla E\left(\boldsymbol{x}_k\right)$$

Conjugate Gradient Method

- Suppose we want to solve the following system:

$$Ax = b.$$

$$where \text{ the } n \times n \text{ matrix } A \text{ is symmtric if } A^t = A,$$
$$A \text{ is positive definite if } x^t Ax > 0.$$

Conjugate Gradient Method

- Two non-zero vector *u* and *v* are said to be conjugate (with respect to A) if

$$u^t A v = 0.$$

Since A is symmetric and positive definite,

we have the inner product

$$< u, v >_A = < Au, v > = < u, Av > = u^t Av.$$

Conjugate Gradient Method

- Suppose that $\{p_k\}$ is a sequence of n mutually conjugate directions.
- The p_k form a basis of $\mathbf{R}_n$. Then, the solution x^* of $Ax = b$ can be expanded in this basis:

$$Ax^* = A(\alpha_1 p_1 + \ldots + \alpha_n p_n) = b.$$

The coefficients $\{\alpha_1, \ldots, \alpha_n\}$ are give by

$$p_k^t A x^* = p_k^t A(\alpha_1 p_1 + \ldots + \alpha_n p_n) = \alpha_k p_k^t A p_k = p_k^t b.$$

$$\alpha_k = \frac{p_k^t b}{p_k^t A p_k}$$

Conjugate Gradient Method

- Given the system $Ax = b$, a sequence of n conjugate directions p_k are first derived and then the coefficients α_k are obtained.
- How to get p_k ?

  The first guest for x^* is denoted by x_0.

  x^* is also *the unique minimizer of the quadratic form:*

$$f(x) = \frac{1}{2} x^t A x - b^t x$$

Conjugate Gradient Method

- Let r_k be the residual at the kth step:

$$r_k = b - Ax_k.$$

- Note that r_k is the negative gradient of f at $x = x_k$.
- So the gradient descent method would be to move in the direction r_k.
- Thus, we have the algorithm:

$$p_0 = r_0, p_{k+1} = r_{k+1} - \frac{p_k^t A r_{k+1}}{p_k^t A p_k} p_k$$ (線性代數裡的正交化過程）

$$\alpha_{k+1} = \frac{p_{k+1}^t b}{p_{k+1}^t A p_{k+1}}$$