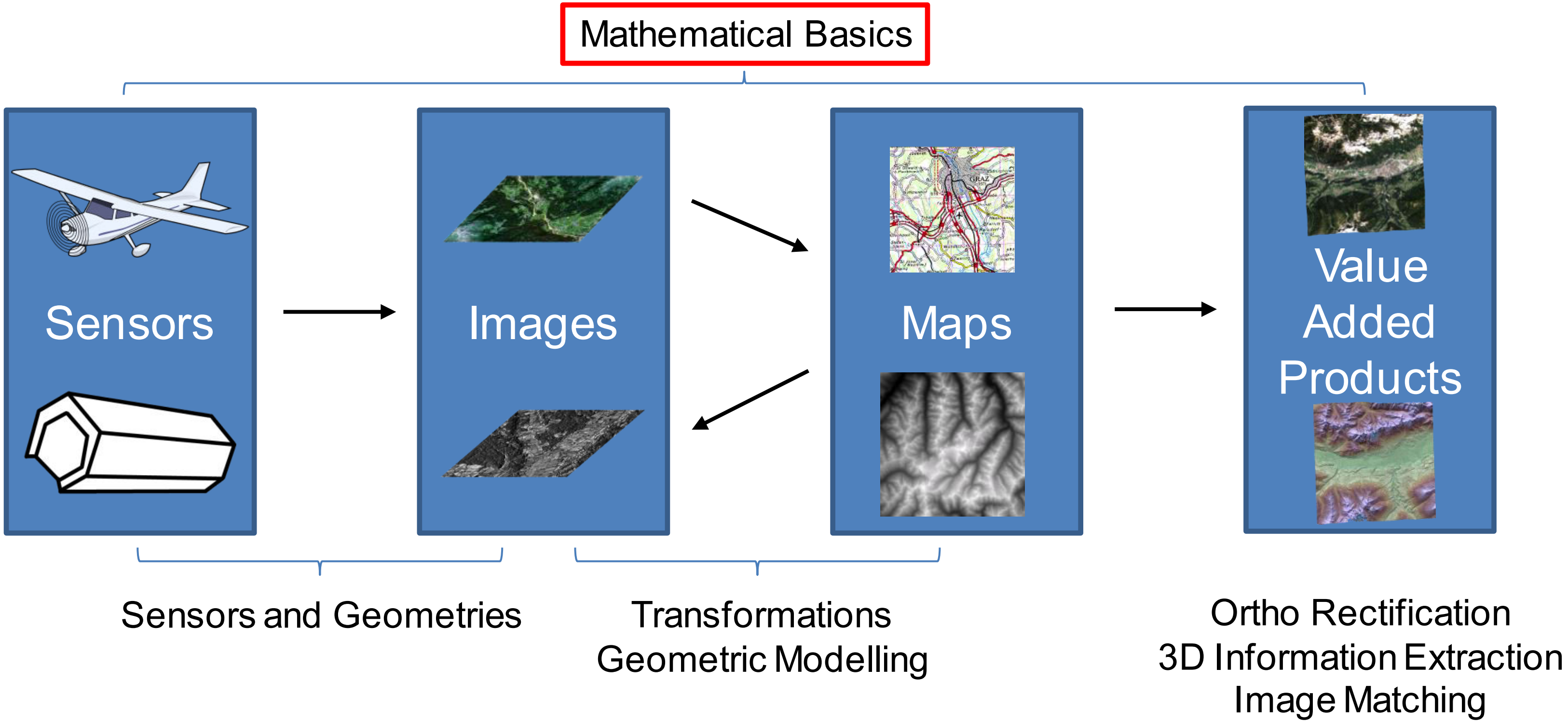
   

Geometric Image Processing in Remote Sensing

Lecture 2 – Mathematical Basics

Dr. Roland Perko
WS 2020/2021

Lecture Overview



Learning Goals

- Ability to describe Newton´s method and its applications
- [Newton Methode und deren Anwendung beschreiben können]

Mathematical Notations

- Domains $\mathbb{N}, \mathbb{Z}, \mathbb{R}, \mathbb{C}, \mathbb{H}$
- Numbers $x \in \mathbb{R}; x = \pi \approx 3.14159$

$$i \in \mathbb{N}_0; i = 42$$

- Vectors $\boldsymbol{v} \in \mathbb{R}^n; \boldsymbol{v} \in \mathbb{R}^2 = \begin{bmatrix} a \\ b \end{bmatrix} = [a, b]^T$
- Matrices $\boldsymbol{M} \in \mathbb{R}^{m \times n}; \boldsymbol{M} \in \mathbb{R}^{2 \times 3} = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix}$

Mathematical Notations

- Functions $f(\boldsymbol{x}) : \mathbb{R}^n \rightarrow \mathbb{R}$

$$f : \mathbb{R}^2 \rightarrow \mathbb{R} \quad \text{with} \quad f(x, y) = xy^2 + x - 1$$

- Equation systems

$$F(\boldsymbol{x}) : \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^n$$

$$F : \mathbb{R}^{2 \times 3} \rightarrow \mathbb{R}^3 \quad \text{with}$$

$$F_1(x, y) = xy^2 + 1$$

$$F_2(x, y) = x + y - 3$$

$$F_3(x, y) = x^2 + y - 2$$

Matrix Notation

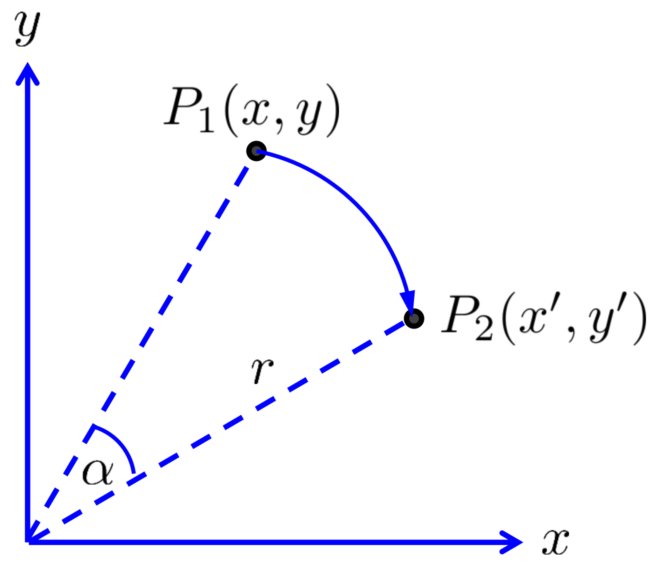
- Example – Rotation in 2D

$$x' = x\cos\alpha + y\sin\alpha$$

$$y' = -x\sin\alpha + y\cos\alpha$$

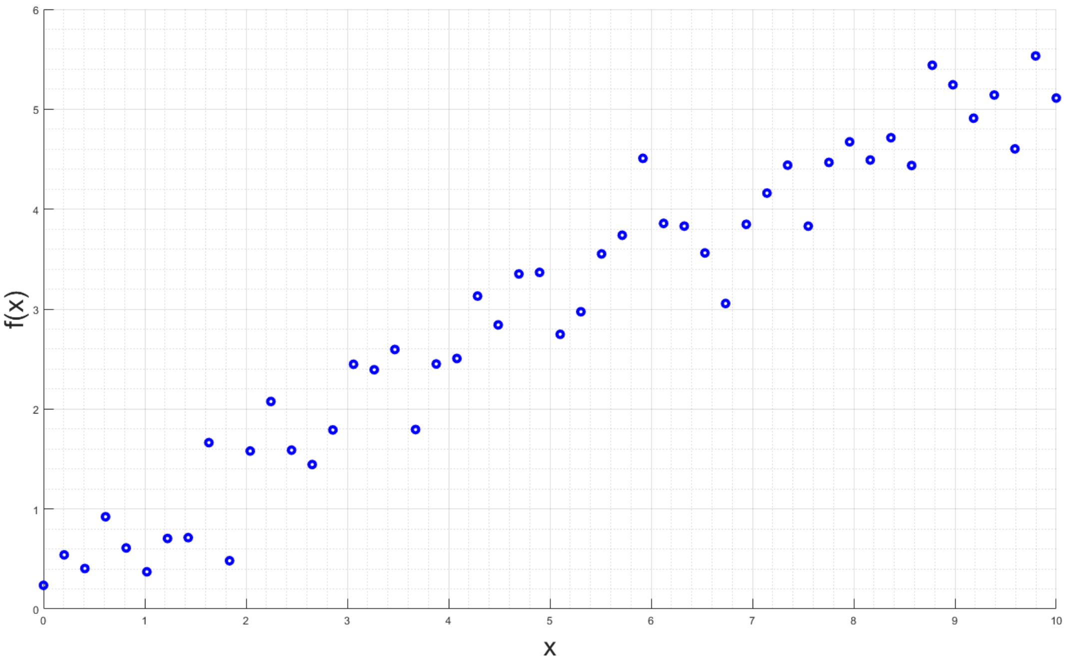
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\boldsymbol{x}' = \boldsymbol{R}(\alpha)\boldsymbol{x}$$



Parameter Adjustment – Example

- Find a line that optimally fits the measured points



One equation per point

$$y_1 = a + bx_1 + r_1$$

$$y_2 = a + bx_2 + r_2$$

$$\vdots$$

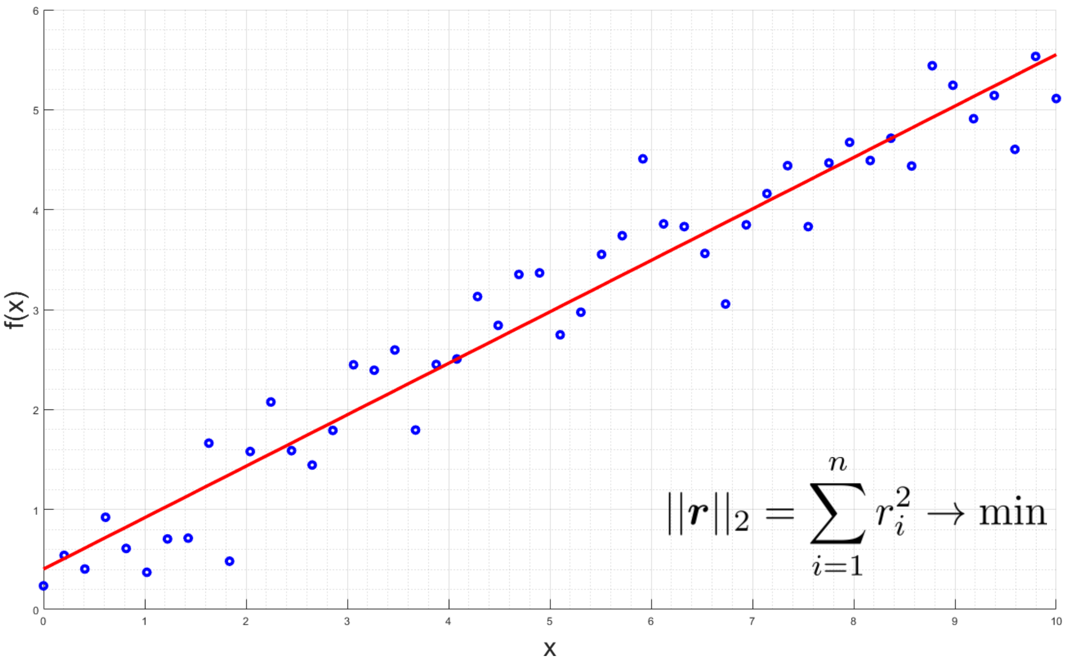
$$y_n = a + bx_n + r_n$$

$$\boldsymbol{l} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{r}$$

$$\boldsymbol{x} = \begin{bmatrix} a \\ b \end{bmatrix}$$

Parameter Adjustment – Example

- Find a line that optimally fits the measured points



$$||\boldsymbol{r}||_2 = \sum_{i=1}^{n} r_i^2 \rightarrow \min$$

One equation per point

$$y_1 = a + bx_1 + r_1$$

$$y_2 = a + bx_2 + r_2$$

$$\vdots$$

$$y_n = a + bx_n + r_n$$

$$\boldsymbol{l} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{r}$$

$$\boldsymbol{x} = \begin{bmatrix} a \\ b \end{bmatrix}$$

Least Squares Adjustment

- Least squares adjustment (over-determined system)

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$$

equation system does not have a solution

$$\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} = \boldsymbol{0}$$

reformulate

$$\boldsymbol{r} = \min_{\boldsymbol{x}} ||\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}||_2$$

find best „solution“ via least squares with residuals $\boldsymbol{r}$

$$\widetilde{\boldsymbol{x}} = \boldsymbol{A}^{+}\boldsymbol{b}$$

$$\boldsymbol{A}^{+} = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T$$

pseudo inverse of $\boldsymbol{A}$

$$\widetilde{\boldsymbol{x}} = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{b}$$

- Could also be solved via Singular Value Decomposition (SVD)

Least Squares – Example

- Find an affine transformation between two 2D point sets

$$x' = ax + by + e$$

$$y' = cx + dy + f$$

affine transformation

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & e \\ c & d & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

matrix notation

How to convert to form $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$

Two rows for each points (two equations)

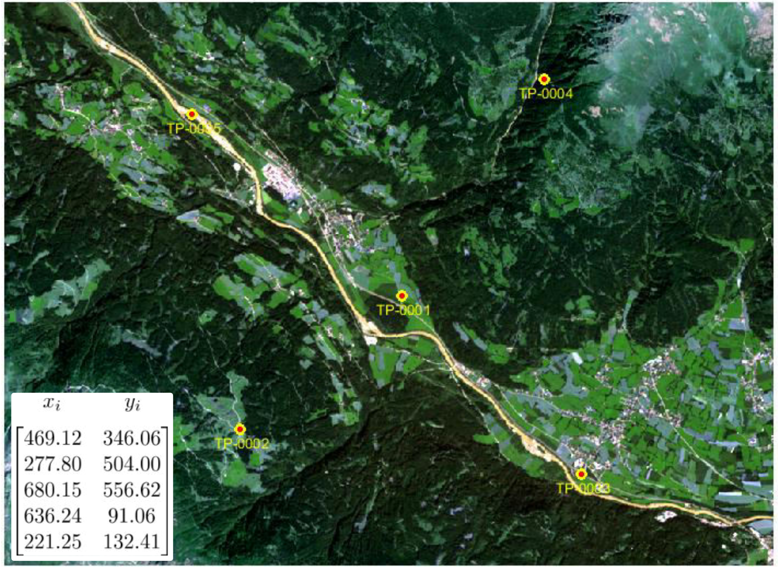
$$\begin{bmatrix} x & y & 0 & 0 & 1 & 0 \\ 0 & 0 & x & y & 0 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix} = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

$$\tilde{\boldsymbol{x}} = (\boldsymbol{A}^T \boldsymbol{A})^{-1} \boldsymbol{A}^T \boldsymbol{b}$$

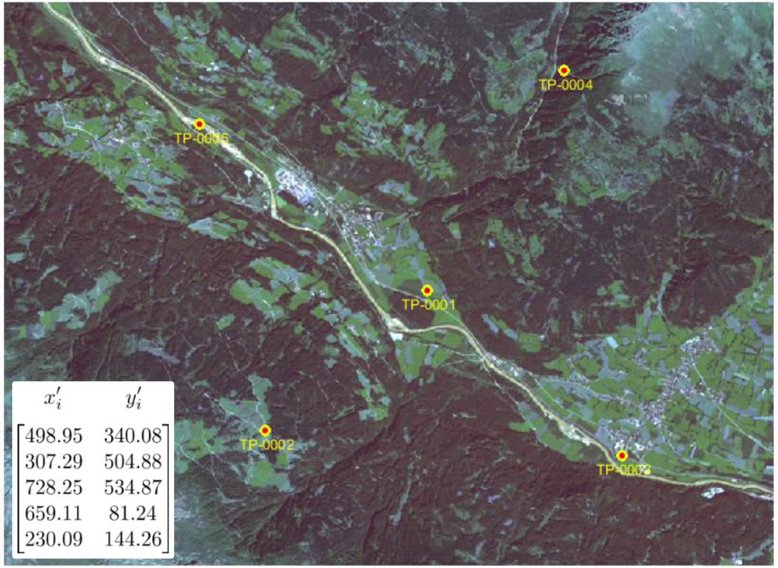
least squares solution

Least Squares – Example

reference image



search image



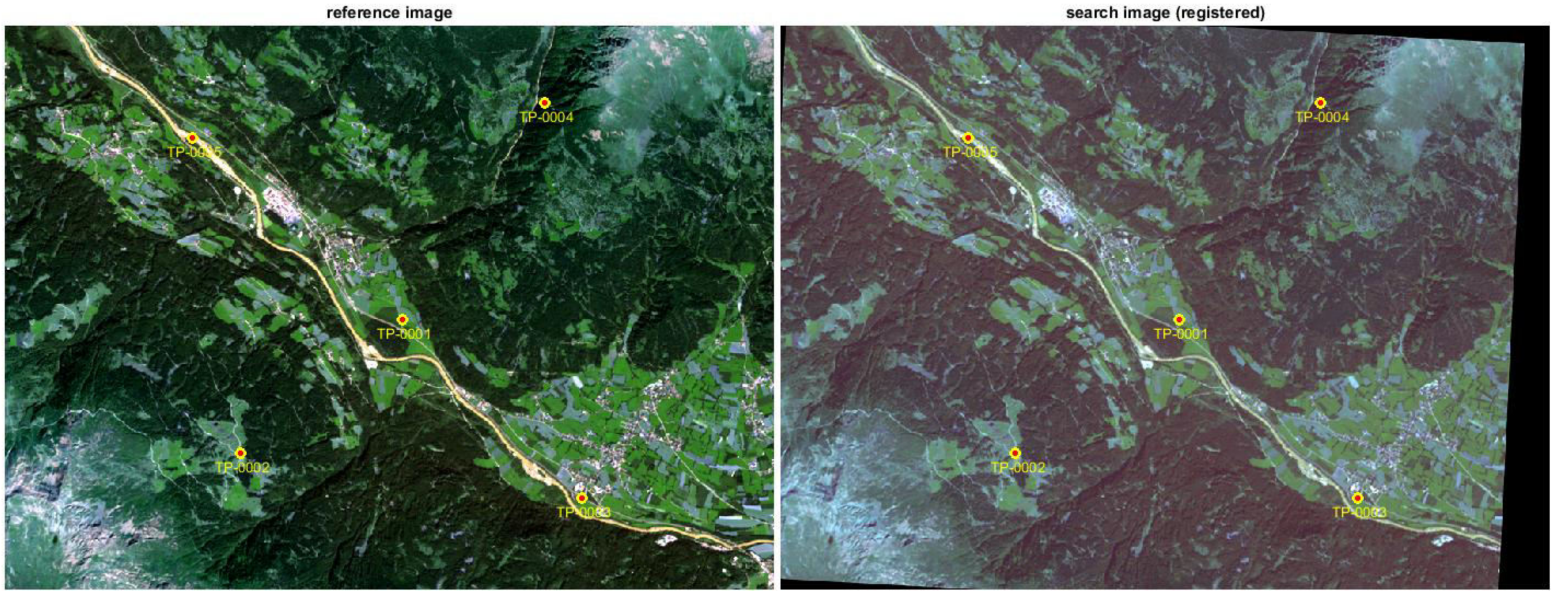
$$\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} = \boldsymbol{0}$$

$$\boldsymbol{A}\tilde{\boldsymbol{x}} - \boldsymbol{b} = \boldsymbol{r}$$

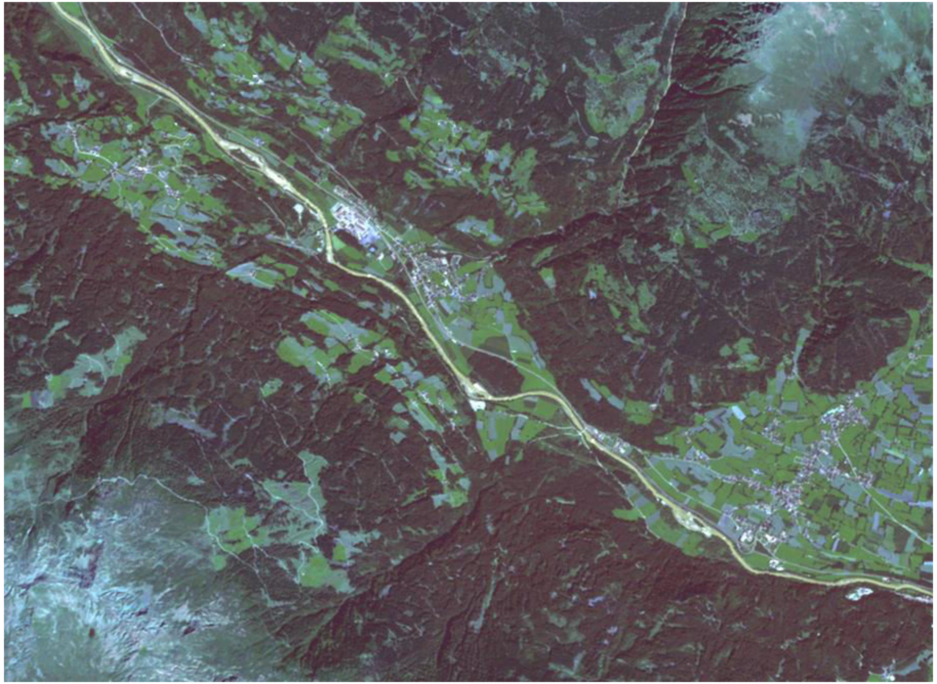
$$\boldsymbol{A} = \begin{bmatrix} 469.12 & 346.06 & 0.00 & 0.00 & 1.00 & 0.00 \\ 0.00 & 0.00 & 469.12 & 346.06 & 0.00 & 1.00 \\ 277.80 & 504.00 & 0.00 & 0.00 & 1.00 & 0.00 \\ 0.00 & 0.00 & 277.80 & 504.00 & 0.00 & 1.00 \\ 680.15 & 556.62 & 0.00 & 0.00 & 1.00 & 0.00 \\ 0.00 & 0.00 & 680.15 & 556.62 & 0.00 & 1.00 \\ 636.24 & 91.06 & 0.00 & 0.00 & 1.00 & 0.00 \\ 0.00 & 0.00 & 636.24 & 91.06 & 0.00 & 1.00 \\ 221.25 & 132.41 & 0.00 & 0.00 & 1.00 & 0.00 \\ 0.00 & 0.00 & 221.25 & 132.41 & 0.00 & 1.00 \end{bmatrix} \quad \tilde{\boldsymbol{x}} = \begin{bmatrix} 1.04 \\ 0.05 \\ -0.05 \\ 0.98 \\ -6.49 \\ 26.46 \end{bmatrix} \quad \boldsymbol{b} = \begin{bmatrix} 498.95 \\ 340.08 \\ 307.29 \\ 504.88 \\ 728.25 \\ 534.87 \\ 659.11 \\ 81.24 \\ 230.09 \\ 144.26 \end{bmatrix} \quad \boldsymbol{r} = \begin{bmatrix} -0.52 \\ -0.06 \\ 0.24 \\ 0.11 \\ 0.07 \\ -0.05 \\ 0.20 \\ 0.08 \\ 0.01 \\ -0.07 \end{bmatrix}$$

affine transformation parameters

Least Squares – Example



Least Squares – Example



search image

Least Squares – Example

reference image

Least Squares – Example

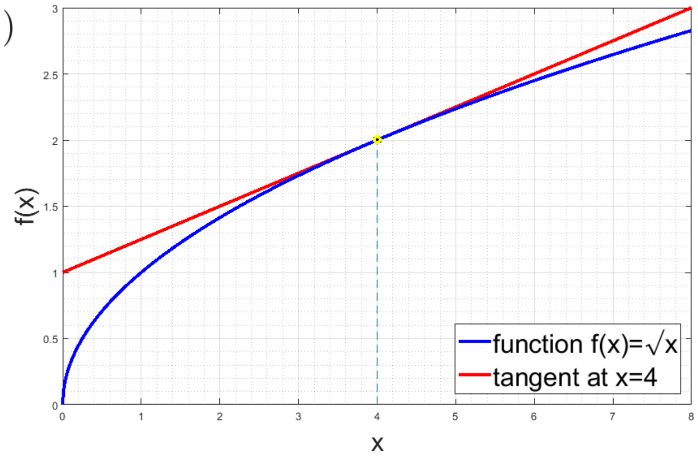
registered search image

Linearization

- Linearization of a function $f(x)$ is the linear approximation of $f(x)$ at a given point x_0
  - Taylor expansion at x_0

$$T_n(x) = \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!}(x-x_0)^k = f(x_0) + \frac{f'(x_0)}{1!}(x-x_0) + \frac{f''(x_0)}{2!}(x-x_0)^2 + \frac{f'''(x_0)}{3!}(x-x_0)^3 + \cdots$$

$$f(x) \approx f(x_0) + f'(x_0)(x-x_0)$$

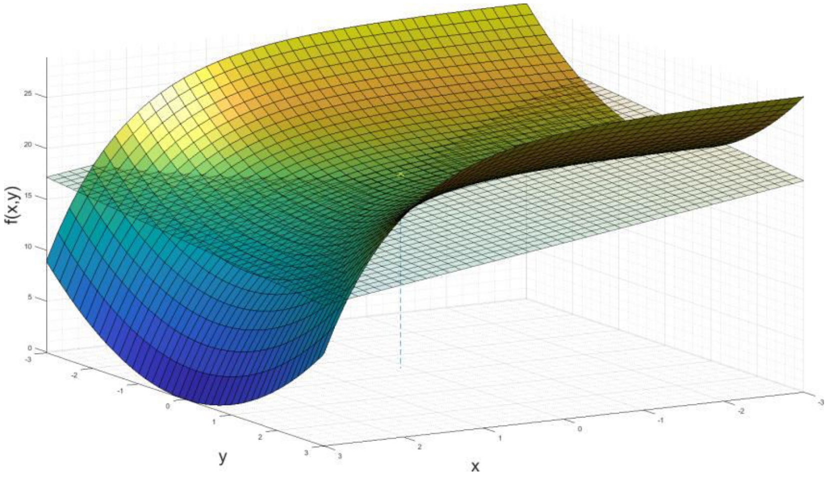


Linearization

- Linearization of a multivariable function $f(x,y) = f(\boldsymbol{x})$

$$f(x,y) \approx f(x_0,y_0) + \left.\frac{\partial f(x,y)}{\partial x}\right|_{x_0,y_0}(x-x_0) + \left.\frac{\partial f(x,y)}{\partial y}\right|_{x_0,y_0}(y-y_0)$$

$$f(\boldsymbol{x}) \approx f(\boldsymbol{x}_0) + \nabla f|_{\boldsymbol{x}_0}(\boldsymbol{x}-\boldsymbol{x}_0) \quad \text{with} \quad \nabla \ldots \text{ nabla operator} \qquad \nabla = \left(\frac{\partial}{\partial x_1}, \cdots, \frac{\partial}{\partial x_n}\right)$$



$$f(x,y) = 20 - e^x + y^2$$

tangent plane

Linearization – Example

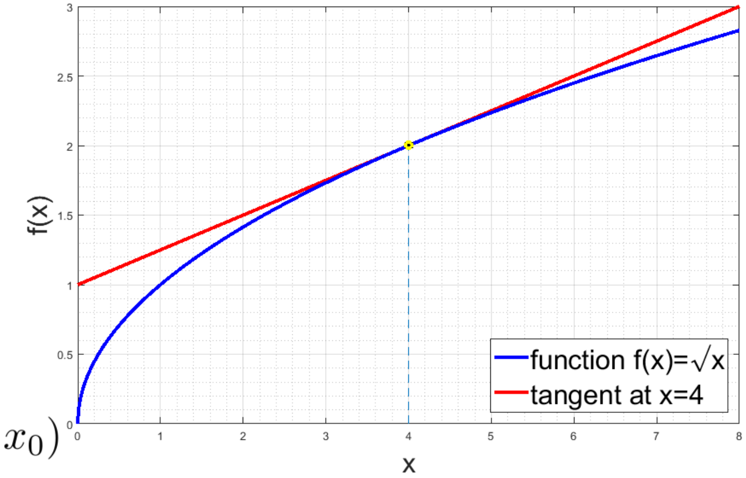
- Approximation near $x = x_0$

$$f(x) = \sqrt{x} \quad \text{and} \quad \sqrt{4} = 2$$

$$\sqrt{4.001} = ?$$

$$\text{linearization of } f(x) \text{ at } x = x_0 \text{ yields}$$

$$y(x) = f(x_0) + f'(x_0)(x - x_0) = \sqrt{x_0} + \frac{1}{2\sqrt{x_0}}(x - x_0)$$

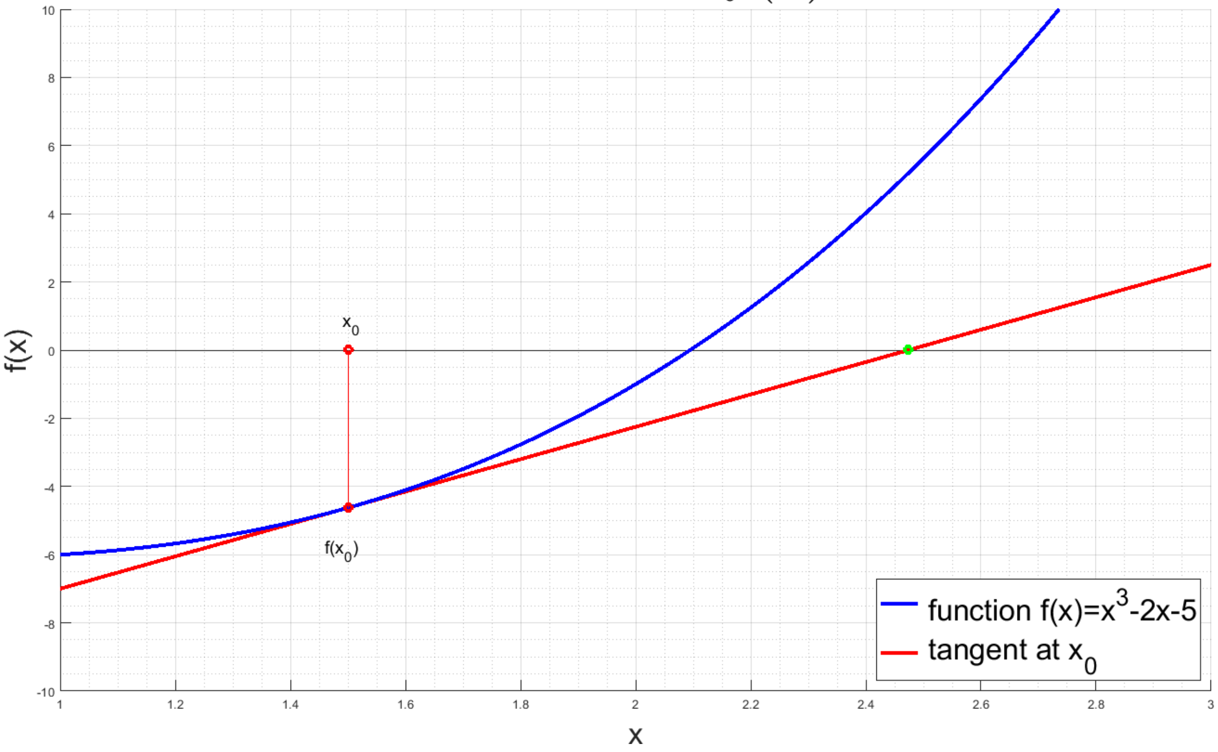


$$x_0 = 4 \quad \text{and} \quad y(x) = 2 + \frac{x-4}{4} \quad \text{and} \quad y(4.001) = 2.00025$$

$$\text{is very close to the real value} \quad \sqrt{4.001} \approx 2.000249984$$

Newton´s Method (from 1669)

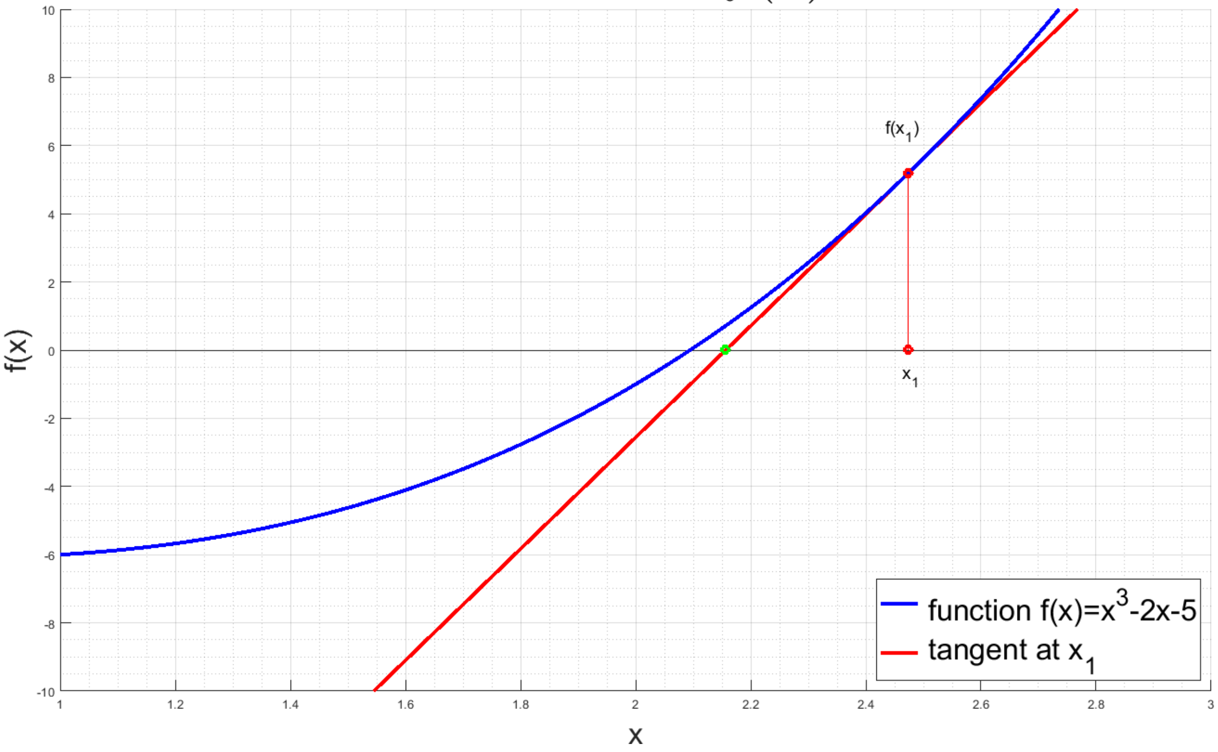
find a solution for $f(x) = 0$



$x_0 = 1.500000$

Newton´s Method (from 1669)

find a solution for $f(x) = 0$

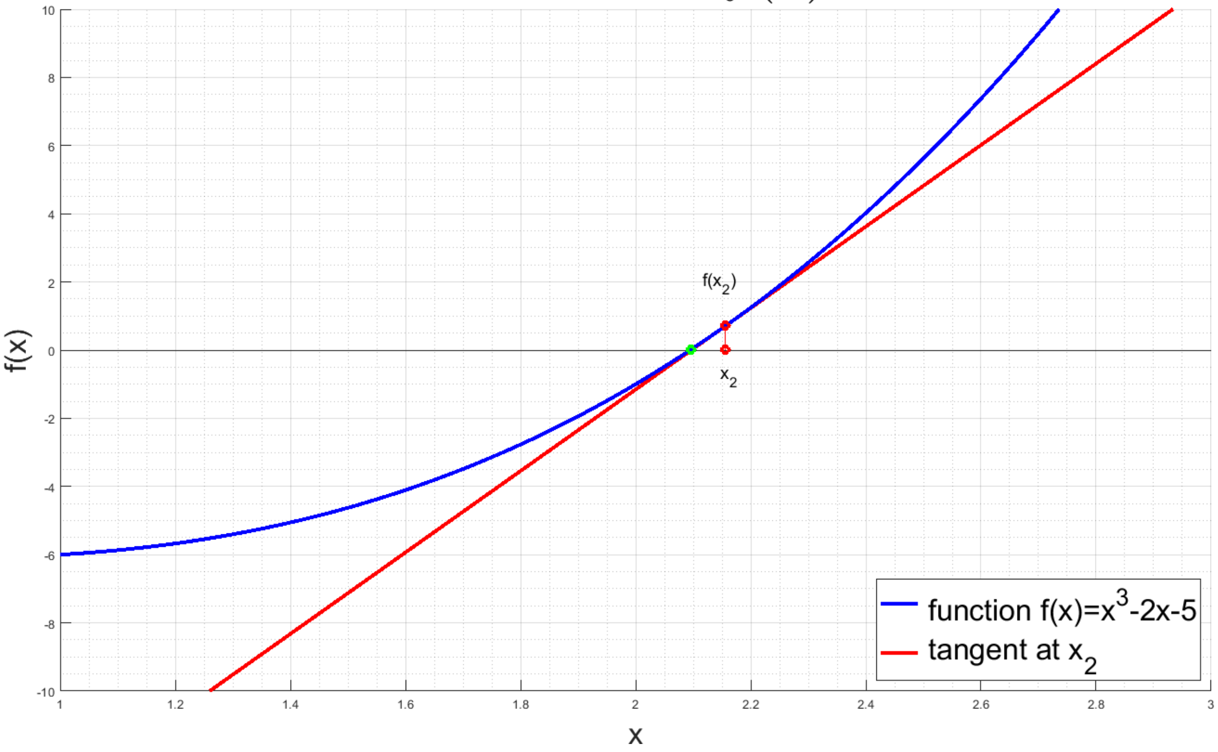


$$x_0 = 1.500000$$

$$x_1 = 2.473684$$

Newton´s Method (from 1669)

find a solution for $f(x) = 0$



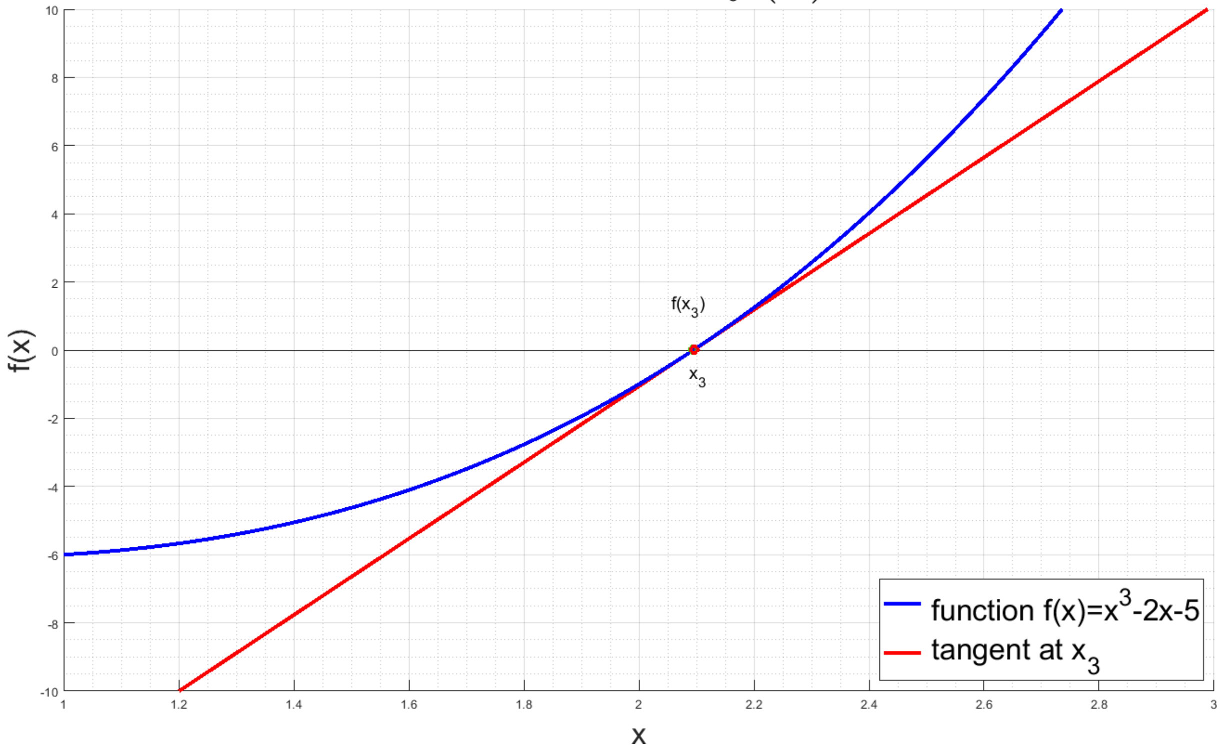
$$x_0 = 1.500000$$

$$x_1 = 2.473684$$

$$x_2 = 2.156433$$

Newton´s Method (from 1669)

find a solution for $f(x) = 0$



$$x_0 = 1.500000$$

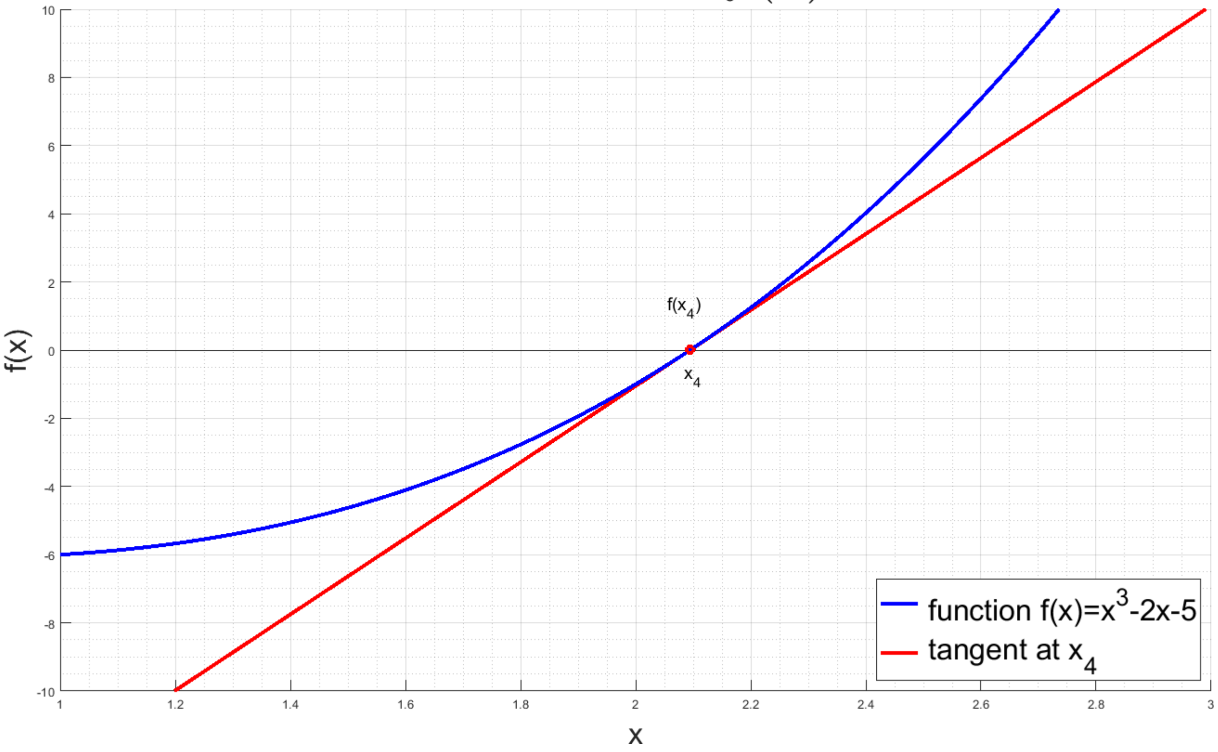
$$x_1 = 2.473684$$

$$x_2 = 2.156433$$

$$x_3 = 2.096605$$

Newton´s Method (from 1669)

find a solution for $f(x) = 0$



$$x_0 = 1.500000$$

$$x_1 = 2.473684$$

$$x_2 = 2.156433$$

$$x_3 = 2.096605$$

$$x_4 = 2.094554$$

$$x \approx 2.094551$$

Newton´s Method

- Linearization and Least Squares Adjustment
  - Allows to solve non-linear equation systems

$$F(\boldsymbol{x}) = \mathbf{0} \;\ldots\; \text{non linear multivariable equation system}$$

$$\boldsymbol{x}_0 \;\ldots\; \text{starting point}$$

- Linearization

$$F(\boldsymbol{x} + \Delta\boldsymbol{x}) \approx F(\boldsymbol{x}) + \boldsymbol{J}_F(\boldsymbol{x})\Delta\boldsymbol{x}$$

- Iterate

$$\boldsymbol{J}_F(\boldsymbol{x}_n)\Delta\boldsymbol{x}_n + F(\boldsymbol{x}_n) = \mathbf{0}$$

→ least squares solution for $\Delta\boldsymbol{x}_n$

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n + \Delta\boldsymbol{x}_n$$

Jacobian matrix

$$\boldsymbol{J}_F(\boldsymbol{a}) := \frac{\partial F}{\partial \boldsymbol{x}}(\boldsymbol{a}) = \left(\frac{\partial F_i}{\partial x_j}(\boldsymbol{a})\right)_{i,j} = \begin{bmatrix} \frac{\partial F_1}{\partial x_1}(\boldsymbol{a}) & \frac{\partial F_1}{\partial x_2}(\boldsymbol{a}) & \cdots & \frac{\partial F_1}{\partial x_n}(\boldsymbol{a}) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial F_m}{\partial x_1}(\boldsymbol{a}) & \frac{\partial F_m}{\partial x_2}(\boldsymbol{a}) & \cdots & \frac{\partial F_m}{\partial x_n}(\boldsymbol{a}) \end{bmatrix}$$

Newton´s Method – Algorithm

Algorithm 4.1: Solving non-linear equation systems with Newton's method.

Input:

- non-linear equation system of form $F(\boldsymbol{x}) = \boldsymbol{0}$
- and its Jacobian matrix J_F
- starting vector $\boldsymbol{x}_0$
- maximal iterations — `// E.g., set to 20.`
- tolerance — `// E.g., set to 1e−7.`

Output:

- solution vector $\boldsymbol{x}_{n+1}$

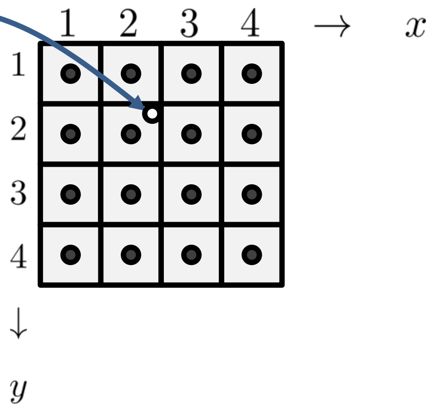
Function `NewtonsMethod`(F, J_F, $\boldsymbol{x}_0$, iterations, tolerance)**:**

- **for** $n = 0 : \textit{iterations}$ **do**
  - $J_F(\boldsymbol{x}_n)\Delta\boldsymbol{x}_n + F(\boldsymbol{x}_n) = \boldsymbol{0}$ — `// Solve for` $\Delta\boldsymbol{x}_n$ `via least squares.`
  - $\boldsymbol{x}_{n+1} = \boldsymbol{x}_n + \Delta\boldsymbol{x}_n$ — `// Get next approximation.`
  - **if** $(|\Delta\boldsymbol{x}_n| \leq \textit{tolerance} \cdot |\boldsymbol{x}_n|)$ **then**
    - `break` — `// Solution found within given tolerance.`
  - **end**
- **end**
- **return** $\boldsymbol{x}_{n+1}$ — `// Return solution vector.`

Interpolation of Pixel Values

- Get the pixel value at a subpixel location
  - Get value from given image at location with subpixel coordinate
  - Pixel with center •

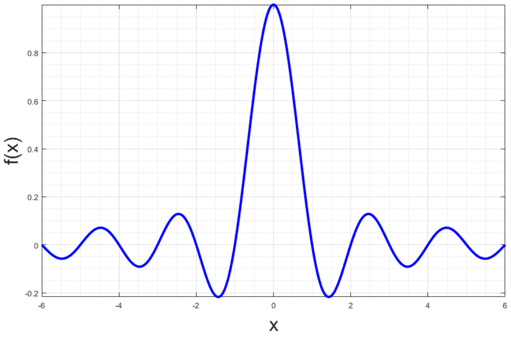
$$\boldsymbol{P}(x, y) = \boldsymbol{P}(2.3, 1.8)$$



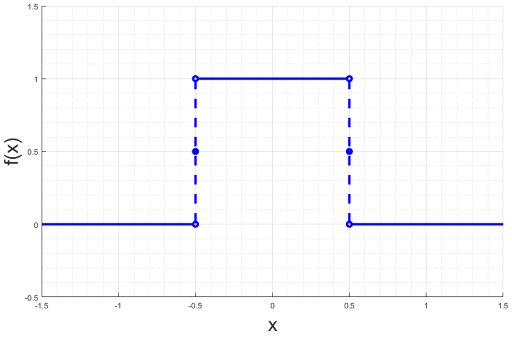
- Also called resampling

Interpolation of Pixel Values

- Interpolation of different order
  - Use neighboring pixel values to interpolate the new value
  - Nearest, Linear, Cubic, Quintic, Windowed Sinc
  - The sinc function is the Fourier transform of the rectangular function



$$\mathrm{sinc}(x) = \frac{\sin(\pi x)}{\pi x}$$



$$\mathrm{rect}(x) = \begin{cases} 0 & \text{if } |x| > \frac{1}{2} \\ \frac{1}{2} & \text{if } |x| = \frac{1}{2} \\ 1 & \text{if } |x| < \frac{1}{2} \end{cases}$$

Interpolation of Pixel Values – Example

input

nearest

linear

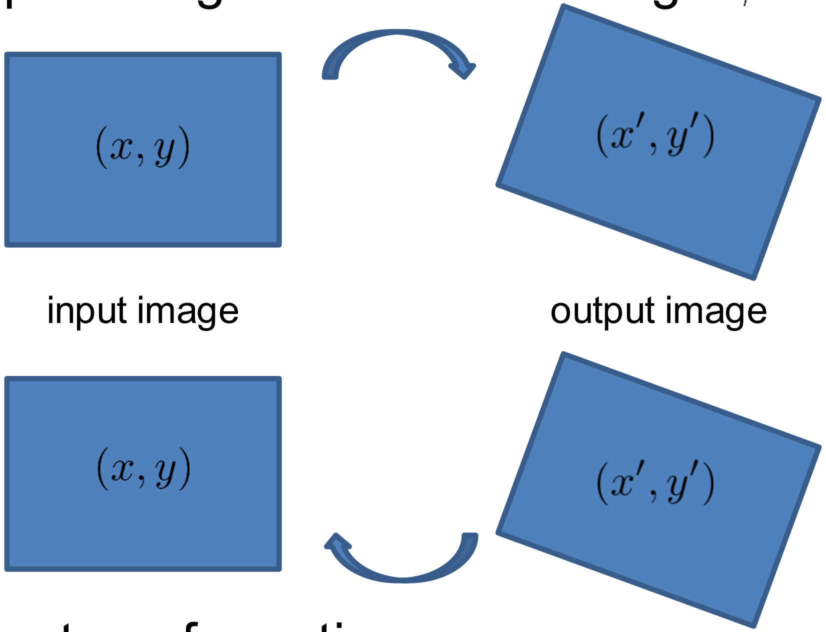
reduced by
factor 4

cubic

sinc 16

Interpolation of Pixel Values

- Example: Image Rotation with angle ϕ



$$\boldsymbol{x}' = \boldsymbol{R}(\phi)\boldsymbol{x}$$

$$\boldsymbol{x} = \boldsymbol{R}^{-1}(\phi)\boldsymbol{x}'$$

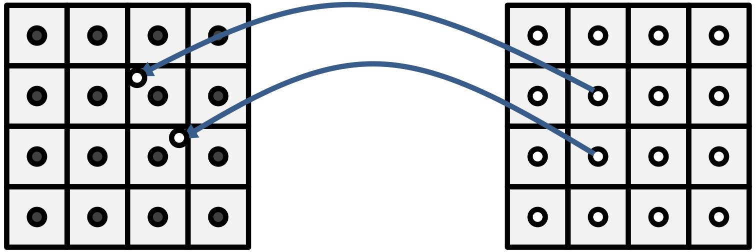
- Inverse transformation

Interpolation of Pixel Values

- Example: Image Rotation with angle ϕ

input image

output image



$$\boldsymbol{x}' = \boldsymbol{R}(\phi)\boldsymbol{x}$$

direct mapping

$$\boldsymbol{x} = \boldsymbol{R}^{-1}(\phi)\boldsymbol{x}'$$

indirect mapping

License Statement

- Unless otherwise noted this work is licensed by **Roland Perko** under a [Creative Commons Attribution 4.0 International License]

- Attributions
  - Image by TU Graz, http://cd.tugraz.at
  - Image by TU Graz, https://www.tugraz.at/institute/ifg/home/