Unit 3

Linear Systems & Root Finding

we want to find…

solving linear systems

1. **…the solution to a system of linear equations**

$$A_1 x + A_2 y = c_1$$
$$A_3 x + A_4 y = c_2$$

- here x and y are the unknowns and A_1, A_2, A_3, A_4, c_1, and c_2 need to be known
- the system is best described in matrix form:

$$\begin{pmatrix} A_1 & A_2 \\ A_3 & A_4 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}$$

root finding

2. **…the roots (=points where the function crosses the zero axis) of a given function $f(x)$**

$$f(x) = 0$$

- root finding can also be used for finding other special points...

$$g(x) = 5 \quad \Leftrightarrow \quad g(x)\text{-}5 = 0 \quad \Leftrightarrow \quad f(x) = 0 \ \text{ with } f(x) = g(x)\text{-}5$$

- ...as well as the intersection point of two functions

$$g(x) = h(x) \quad \Leftrightarrow \quad g(x)\text{-}h(x) = 0 \quad \Leftrightarrow \quad f(x) = 0 \ \text{ with } f(x) = g(x)\text{-}h(x)$$

- **Note**: these functions are arbitrary and there are no restrictions to them.

Linear Systems *definitions*

- a system of linear equations is a set of M equations…

$$\left.\begin{aligned} A_{11}x_1 + A_{12}x_2 + \ldots + A_{1N}x_N &= b_1 \\ A_{21}x_1 + A_{22}x_2 + \ldots + A_{2N}x_N &= b_2 \\ &\ldots \\ A_{M1}x_1 + A_{M2}x_2 + \ldots + A_{MN}x_N &= b_M \end{aligned}\right\} \text{M equations}$$

N unknown variables

…for N unknown variables x_i

$$\begin{pmatrix} x_1 \\ x_2 \\ \ldots \\ x_N \end{pmatrix}$$

- a linear system can be written as a matrix equation...

$$A\,\vec{x} = \vec{b}$$

…with:

$$A = \underbrace{\begin{pmatrix} A_{11} & \ldots & A_{1N} \\ \ldots & \ldots & \ldots \\ A_{M1} & \ldots & A_{MN} \end{pmatrix}}_{\text{MxN matrix}}, \vec{x} = \begin{pmatrix} x_1 \\ \ldots \\ x_N \end{pmatrix}, \vec{b} = \begin{pmatrix} b_1 \\ \ldots \\ b_M \end{pmatrix}$$

- **Note**:

  - $A_{11}x_1 + \ldots + A_{1N}x_N = b_1$ describes a hyper-plane in the N-dimensional space ($x_1, \ldots, x_N$)
  - the solution to a linear system is the intersection of hyper-planes.
  - linear systems also work for non-linear function if the functions have the same structure, e.g.

$$\begin{aligned} y &= m_1 \sin(x) + c_1 \\ y &= m_2 \sin(x) + c_2 \end{aligned} \quad \Leftrightarrow \quad \begin{aligned} m_1 \sin(x) - y &= -c_1 \\ m_2 \sin(x) - y &= -c_2 \end{aligned} \quad \Leftrightarrow \quad \begin{pmatrix} m_1 & -1 \\ m_2 & -1 \end{pmatrix} \begin{pmatrix} \sin(x) \\ y \end{pmatrix} = \begin{pmatrix} -c_1 \\ -c_2 \end{pmatrix}$$

(one then certainly does not solve for x but for $\sin(x)$ in the end...)

Linear Systems

solving linear systems

- a system of linear equations is a set of M equations for N unknown variables x_i :

$$A\,\vec{x} = \vec{b}$$

$$A = \begin{pmatrix} A_{11} & \dots & A_{1N} \\ \dots & & \dots \\ A_{M1} & \dots & A_{MN} \end{pmatrix}, \quad \vec{x} = \begin{pmatrix} x_1 \\ \dots \\ x_N \end{pmatrix}, \quad \vec{b} = \begin{pmatrix} b_1 \\ \dots \\ b_N \end{pmatrix}$$

- the solution to the system is given by…

$$\vec{x} = A^{-1}\vec{b}$$

…with A^{-1} being the inverse matrix of A defined via

$$AA^{-1} = 1$$

- Note:
  - the inverse matrix is **not** given by

$$A^{-1} \neq \begin{pmatrix} A_{11}^{-1} & \dots & A_{1N}^{-1} \\ \dots & & \dots \\ A_{M1}^{-1} & \dots & A_{MN}^{-1} \end{pmatrix}$$

- Solvability of Linear Systems:

$$\begin{pmatrix} A_{11} & \dots & A_{1N} \\ \dots & & \dots \\ A_{M1} & \dots & A_{MN} \end{pmatrix} \begin{pmatrix} x_1 \\ \dots \\ x_N \end{pmatrix} = \begin{pmatrix} b_1 \\ \dots \\ b_N \end{pmatrix}$$

  - M<N: underdetermined system, i.e. you cannot find a unique solution
  - M=N: there exists a unique solution if det(A)≠0
  - M>N: overdetermined system, i.e. you may find a solution by requiring $r = Ax - b$ to be minimal

Linear Systems *solving linear systems*

- MATLAB has built-in commands to solve linear systems and calculate inverse matrices, respectively:

```
>> x = mldivide(A,b)
>> x = A\b
>> x = inv(A)*b
```

- Note:
    - the multiplication for the last option is “*” and not “.*”!

➢ **exercise:**

- calculate the intersection point of the two lines

$$y = 5x - 3$$
$$y = -0.3x + 7$$

…by solving the linear system

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

- use all three methods to solve the system and compare the results
- generate a figure that plots the intersection point as a cross as well as the two lines

➢ **exercise:**

- you plan to buy a new suitcase for your flight to Melbourne, Australia, that complies with the airline regulations and the DIN norm:
    - airline regulation says that the sum of all three lengths is limited by

$$x+y+z=158\text{cm}$$

    - DIN norm says that the ratio of two lengths has to be √2

$$x = \sqrt{2}\ y$$
$$y = \sqrt{2}\ z$$

- calculate the dimensions x, y, and z of the allowed suitcase by solving the linear system

$$\begin{pmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

- **hint**: you must bring each of the three equations into the form $A_{i1}x + A_{i2}y + A_{i3}z = b_i$

Linear Systems

solving linear systems

➢ **exercise:**

- remember the cannonball exercise from Unit 1, using now the following constraints:
  - assume a starting point of x_0=0m, y_0=0m
  - after T=3sec the cannonball has reached position x=18m, y=2m
- calculate the initial velocity v_{0x}, v_{0y} by solving the linear system

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}\begin{pmatrix} v_{0x} \\ v_{0y} \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

- what was the angle at which the cannonball was shot?

➢ **exercise:**

- you and a horse are having a race:
  - you can run 0.2km per minute, and
  - the horse can run 0.5km per minute, but it takes 6min. to saddle the horse.
- how far (and how long) can you run before the horse catches up with you?

➢ **exercise:**

- consider the following linear system

$$\begin{pmatrix} 1 & -2 \\ 2 & -4 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 5 \\ 6 \end{pmatrix}$$

- what do you get when trying to solve this system using `inv(A)` or `mldivide(A,b)`
- does this system have a solution? (check `det(A)`)

Root Finding *motivation*

1. we either want to find the root of any arbitrary function $f(x) = 0$

- **Note**: finding points $f(x) = b$ require finding the root of $f(x)-b = 0$

2. or we want to find the intersection point of two arbitrary functions

$$y = g(x)$$
$$y = h(x)$$

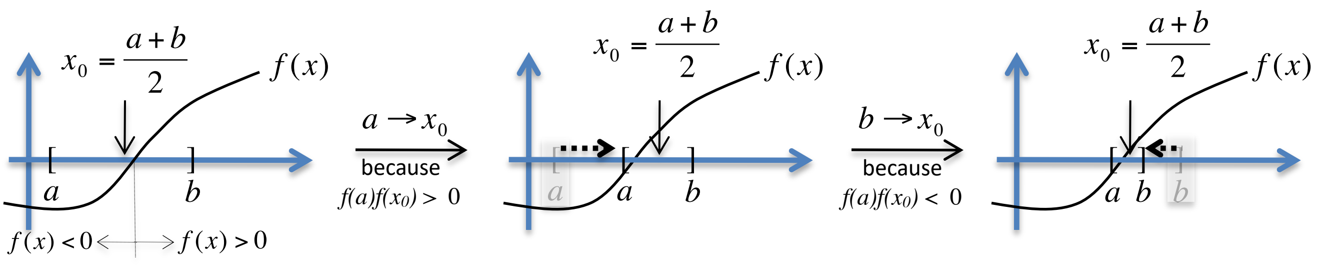
- **Note**:
  - in the case of linear functions $f(x)$ and $h(x)$ we are able to calculate the intersection by solving the corresponding linear system $A\boldsymbol{x}=\boldsymbol{b}$ as discussed previously!
  - the problem of finding the intersection of 2 functions $g(x)$ and $h(x)$ can be solved by "Root Finding",

$$\left.\begin{array}{l} y = g(x) \\ y = h(x) \end{array}\right\} \Rightarrow \quad g(x) = h(x) \quad \Rightarrow \quad 0 = f(x) = g(x) - h(x)$$

i.e. we define a function $f(x)=g(x)-h(x)$ and determine the points x_0 where $f(x_0) = 0$.

bi-section method

- the bi-section method successively divides an interval $[a,b]$ bracketing a root of $f(x)$ until the difference between the left and right edge of the interval is smaller than a pre-selected accuracy threshold,

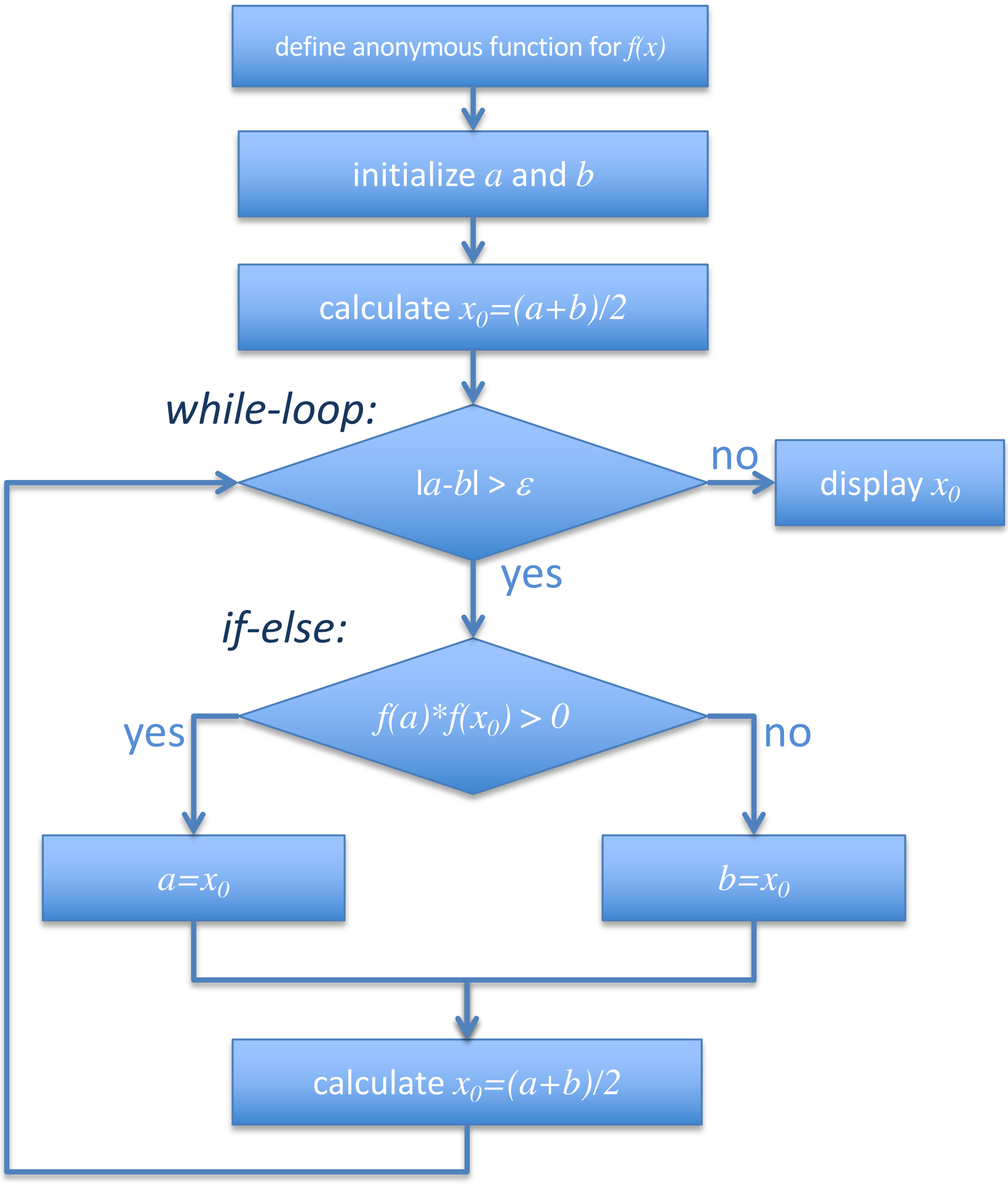


i.e. we are constantly shifting either a or b towards the actual root depending on the position of the root with respects to the mid-point $(a+b)/2$ of the current interval!

- necessary requirements to program this algorithm:
  - a kind of loop that loops until a certain condition is no longer valid: while-loop
  - a criterion to decide whether to shift a or to shift b to the midpoint: if-then-else

Root Finding *bi-section method*

- flowchart representation of the bi-section algorithm:



- Note:
  - the "small number" ε should be larger than MATLAB's `eps` (`help eps`)

- Advanced Tips:
  - the calculation of $f(x)$ is best done by using a my_function.m script (cf. Unit 2)
  - to avoid writing an extra script my_function.m that contains only a single function $f(x)$ you can define an "anonymous function" in MATLAB (cf. Unit 2):

```
>> f = @(x) (expression defining function of x);
```

Root Finding *bi-section method*

➢ **exercise:**

- find the root of the function $\Phi(r) = \frac{2}{r^2} - \frac{5}{r}$ on the interval [0.2,2] by bi-section.
- plot the function $\Phi(r)$ and mark the root r_0
- **Note**:
  - Φ is the *effective* gravitational potential in spherical coordinates where
    …the first term is the centrifugal potential and
    …the second term the gravitational potential

➢ **exercise**:

- find the root of the function $f(x) = x^2$ on the interval [-2,2] by bi-section.
- do you find a solution?
- what happens when you change the initial bi-section interval (e.g. [-3,1], [-1,2], …)?

➢ **exercise**:

- write a function

```
function [root] = bisec(f, a, b)
```

  that takes as input arguments an anonymous function f=@(x)(…) and the intervall [a,b] and returns the root of $f(x)$ on that interval
- repeat the previous exercises using your `bisec.m` script

➢ **exercise**:

- with the definition of an anonymous function you can use MATLAB's `fzero()` to find roots!
- adjust your root finding script with the anonymous function to use `fzero()`
- **hint**: `help fzero()`

voluntary exercise!

Root Finding *Newton-Raphson method*

▪ the convergence of the root-finding can be increased by not only using the actual values of the function but also including its derivatives in the root-finding process:

• we are Taylor-expanding the function $f(x)$ about a point x_0 close to the root up to the first order term:

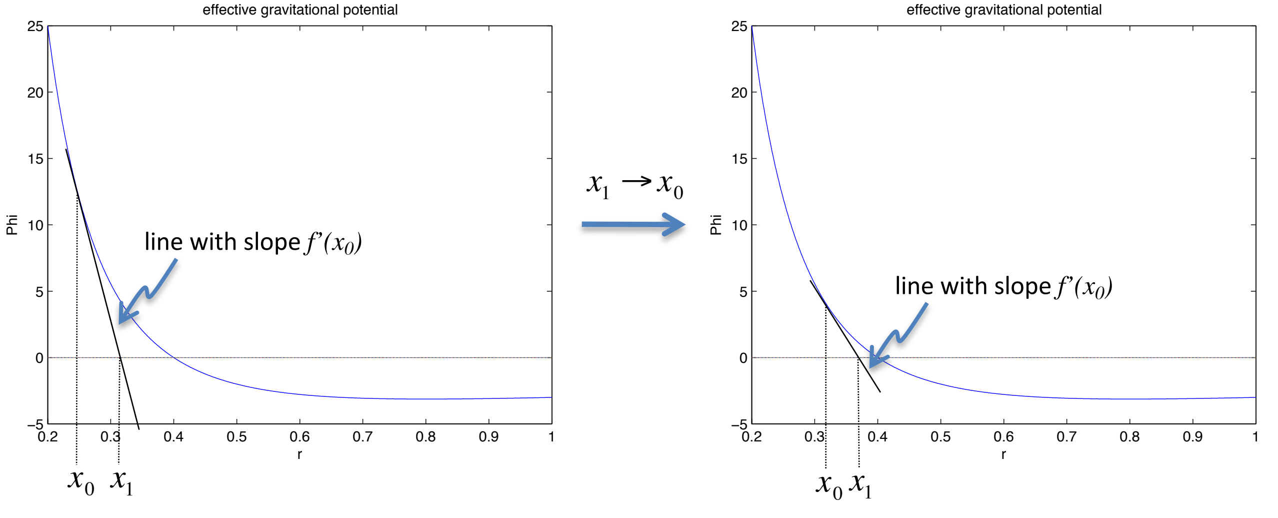
$$f(x) = f(x_0) + f'(x_0)(x - x_0) \quad + O(x^2)$$

• as we are interested in the root we request $f(x) = 0$ leading to (ignoring higher-order terms):

$$x = x_0 - \frac{f(x_0)}{f'(x_0)}$$

• x will not be the root (as we truncated the Taylor-expansion), but it will be closer to the root than x_0

→ we need to determine the root by applying the “formula” iteratively :

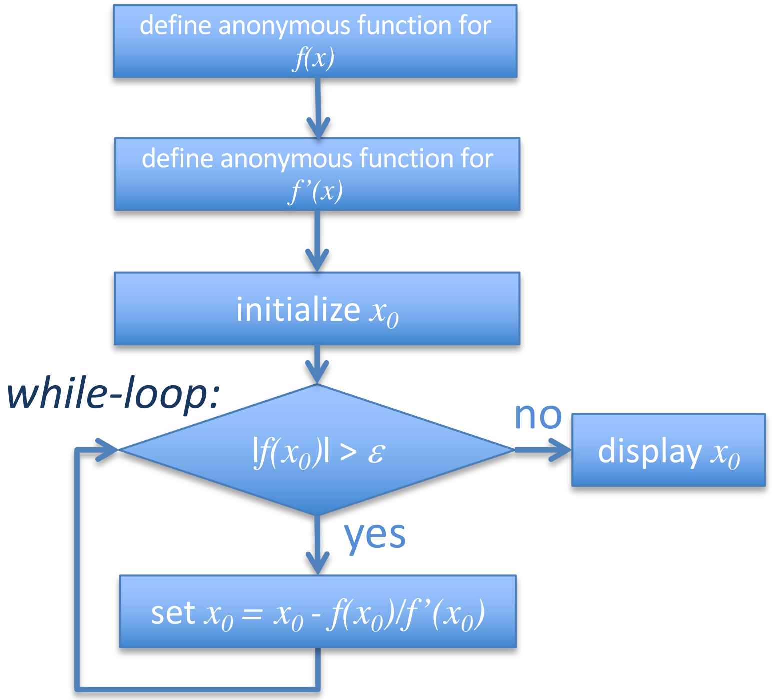


▪ Note:

• this method only convergences when choosing a starting value x_0 sufficiently close to the actual root
• the derivative of $f(x)$ should neither be zero nor infinite in the region of interest
• this method convergences faster than the bi-section (i.e. fewer iterations)
• you ***must*** have an analytical formula for the derivative $f'(x)$

Root Finding *Newton-Raphson method*

- flowchart representation of Newton-Raphson root finding procedure:



➢ **exercise:**

- find the root of the function $\Phi(r) = \frac{2}{r^2} - \frac{5}{r}$ on the interval [0.2,2] by Newton-Raphson.
- plot the function $\Phi(r)$ and mark the root r_0
- **hints**:
    - use as initial guess for the root r_0=0.3
    - use the analytical expression for the derivative to calculate $f'(r_0)$
    - $|f(x_0)|$ can be calculated using MATLAB's `abs()` function (`help abs`)

➢ **exercise**:

- what happens when you start the iterations with r_0>0.6 ?
- what is special about the point r_0=0.6 and why does the method fail for r_0>0.6?

➢ **exercise**:

- calculate the minimum of $\Phi(r)$ by finding the root of $\Phi'(r_{min})$=0?
- hint: you obviously require the second derivative of Φ for the Newton-Raphson method

➢ **exercise**:

- write a function

```
function [root] = NewRaph(f, fder, x0)
```

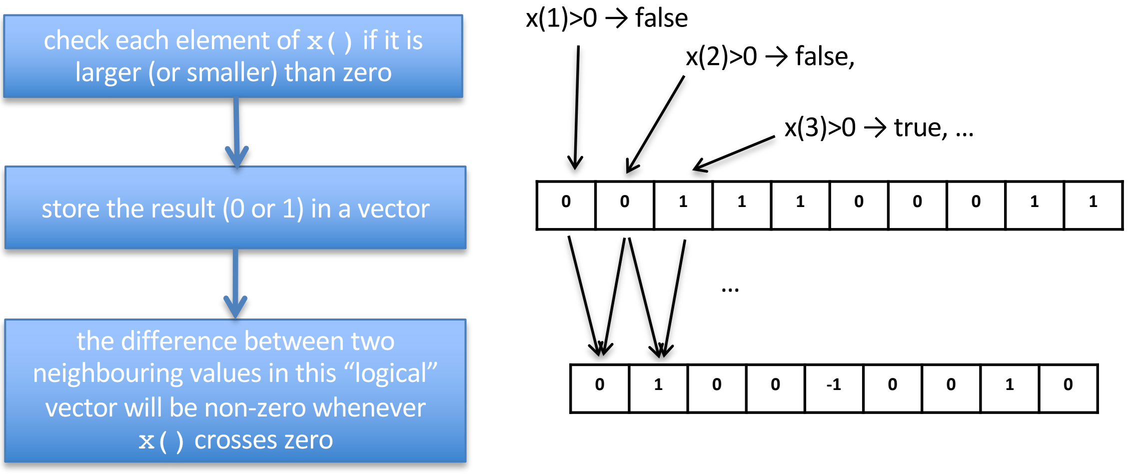
that takes as input arguments two anonymous functions for $f(x)$ and $f'(x)$ and the first guess x_0

Root Finding *approximate method for vectors*

- we want to find the zero-crossings of a given vector, e.g.

x =

-0.5	-0.2	1.2	3.4	0.3	-0.1	-0.5	-0.3	0.3	0.9



➢ **exercise:**

- write a

  `function [roots] = veczeros(x)`

  that returns all the (approximate) positions where `x()` crosses zero

- **hints**:
  - use the logical operator '>' for the vector `x()`
  - use `diff()` to calculate the difference between neighbouring elements in a vector
  - use `find()` to find the non-zero positions in a vector
  - the whole calculation could (and should!) be done in one single line!

Examples *application - intersection of circle and exponential*

▪ **circle:**

- a circle can be described in two different ways in Cartesian coordinates

$$x^2 + y^2 + Ax + By + C = 0 \qquad \text{Eq.(1)}$$

or

$$(x - x_0)^2 + (y - y_0)^2 = R^2 \qquad \text{Eq.(2)}$$

where x_0 and y_0 are the centre and R its radius.

➢ **exercise:**

- determine x_0, y_0, and R of the circle crossing the 3 points

$$(3,-1)$$
$$(-2,4)$$
$$(6,8)$$

by solving the linear system for A, B, and C first.

- hint: to obtain x_0, y_0, and R from A, B, and C you need to convert Eq.(1) into something similar to Eq.(2) by using the binomial rules.

➢ **exercise:**

- plot the circle marking the three points with crosses
- hint: plot two functions $y=f(x)$ where one is the positive and the other one the negative root of y^2

➢ **exercise:**

- find the two intersection points of the circle with $f(x)=e^{-x/4}$
- plot the function $f(x)$ onto the same figure as the circle
- mark the two intersection points wit the circle with a cross

- hint: you need to use a root-finding technique

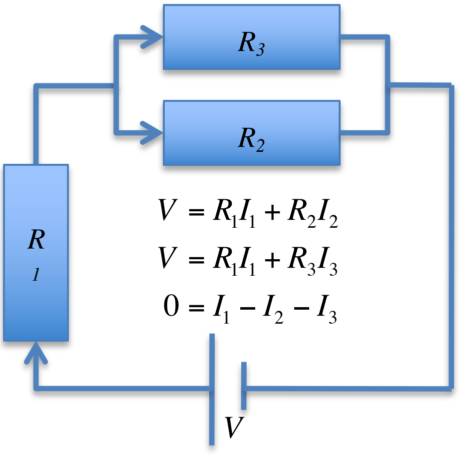
Examples *application - electrical circuits*

- **exercise:**

• solve for the currents I_1, I_2, and I_3 using

- V = 5 V
- R_1 = 4 Ω
- R_2 = 4 Ω
- $R_3 \in$ [0,100] Ω

• graphically represent the currents as a function of R_3

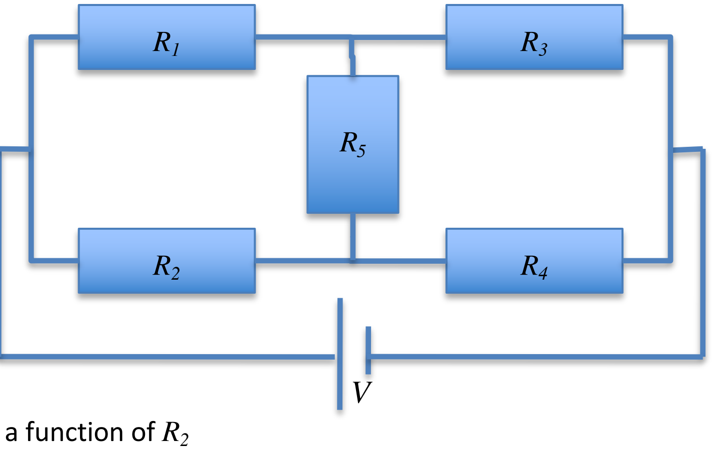


$$V = R_1 I_1 + R_2 I_2$$
$$V = R_1 I_1 + R_3 I_3$$
$$0 = I_1 - I_2 - I_3$$

- **exercise ("Wheatstone Bridge"):**

• solve for the currents I_1, ..., I_5 using

- V = 5 V
- R_1 = 10 Ω
- $R_2 \in$ [0,50] Ω
- R_3 = 15 Ω
- R_4 = 20 Ω
- R_5 = 3 Ω



• graphically represent the currents as a function of R_2

• graphically show that $I_5 = 0$ for $R_2 = \frac{R_1 R_4}{R_3}$

• ***hint***: the application of Kirchoff's rules to the Wheatstone Bridge gives:

currents:
$$I_1 = I_3 - I_5$$
$$I_2 = I_4 + I_5$$

voltages:
$$V_1 = V_2 + V_5$$
$$V_3 = V_4 - V_5$$
$$V = V_1 + V_3$$

=>

$$0 = -I_1 + I_3 - I_5$$
$$0 = -I_2 + I_4 + I_5$$
$$0 = -I_1 R_1 + I_2 R_2 + I_5 R_5$$
$$0 = -I_3 R_3 + I_4 R_4 - I_5 R_5$$
$$V = I_1 R_1 + I_3 R_3$$

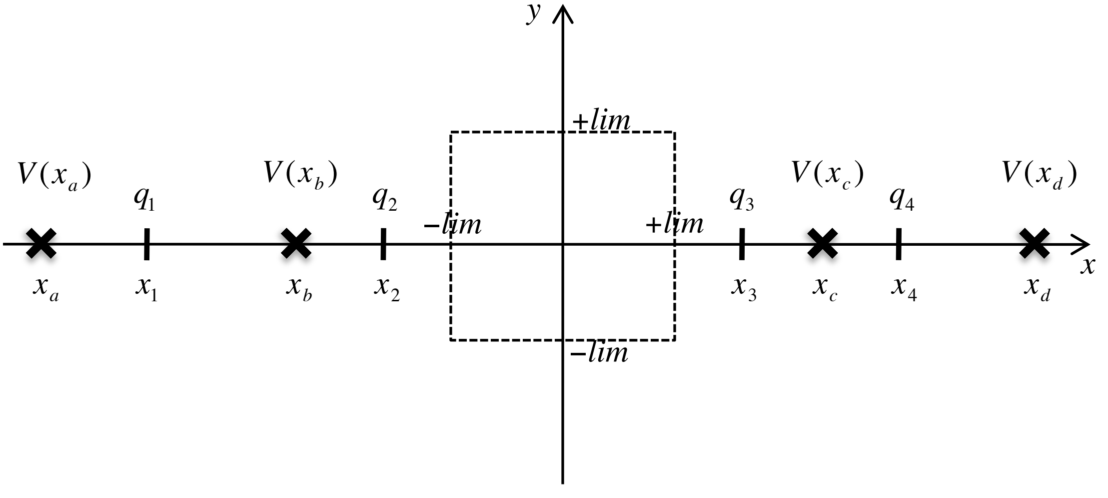
Examples *application – unknown charges*

➢ **exercise:**

Four charges q_1, q_2, q_3 y q_4 of unknown value are placed along the x-axis. The electrical potential

$$V(x)=\sum_i \frac{q_i}{x_i - x}$$

has been measured experimentally at the four positions x_a, x_b, x_c, x_d



- determine the charge values q_i by solving the system of equations

$$V(x_j)=\frac{1}{4\pi\varepsilon_0}\sum_i \frac{q_i}{x_i - x_j} \qquad j=(a,b,c,d)$$

- visualize the electric potential generated by the four charges in the region x = [*-lim*, *lim*] and y = [*-lim*, *lim*].

(the relevant values are

x_1=−4.3·10^{-2} m, x_2=−1.5·10^{-2} m, x_3=1.29·10^{-2} m, x_4=4.7·10^{-2} m,

x_a=−5.1·10^{-2} m, x_b=−2.7·10^{-2} m, x_c=3.0·10^{-2} m, x_d=7.0·10^{-2} m,

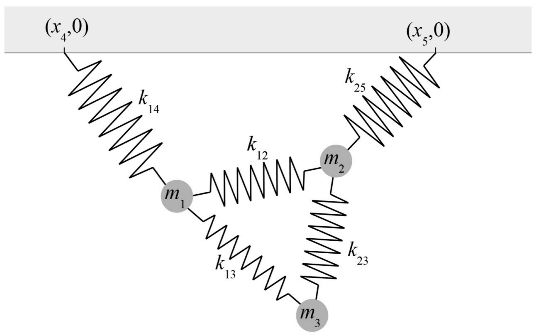
V_a=−1.3 V, V_b=3.33 V, V_c=−0.77 V, V_d=3.0 V,

lim=0.98·10^{-2} m)

Examples *application – equilibrium springs*

➢ **exercise:**

Three masses are connected via a system of springs:



The corresponding equations for the equilibrium state are

$$0 = F_{ix} = \sum_{j=1}^{5} k_{ij}(x_j - x_i)$$

$$0 = F_{iy} = \sum_{j=1}^{5} k_{ij}(y_j - y_i) - m_i g$$

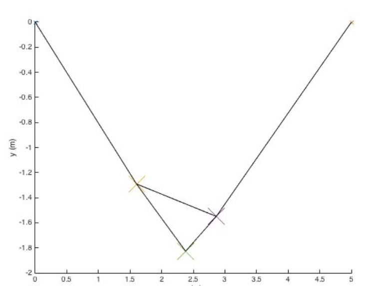
where $k_{ij}=k_{ji}$ for all existing springs and $k_{nm}=0$ for all non-existing springs. These equations form a system of 6 equations for the 6 unknowns $x_1, x_2, x_3, y_1, y_2, y_3$.

- Write a function `equilib3m.m` that calculates the equilibrium positions of the masses m_1, m_2, and m_3, the matrix of the spring constants k_{ij}, and the fixpoints x_4, x_5. This function should work like this:

```
function [r] = equilib3m(m, k, p)
% Equilibrium points of 3 hanging masses connected by springs and
% under the gravitational force as shown in the Figure above
%   Input:
%       m(3)    :   particle masses (kg)
%       k(5)    :   spring constants (N/m)
%       p(2)    :   [x4, x5] coordinates (m)
%  Output:
%       r(2,3)  :   equilibrium points of each mass (m)
```

- Using your script function determine the equilibrium points for the following setup:
  x_4=0m, x_5=5m,
  m_1=2kg, m_2=3kg, m_3=5kg,
  k_{12}=20N/m, k_{23}= 80N/m, k_{14}=40N/m, k_{13}=50N/m, k_{25}=30N/m

- Create a plot like this:



MATLAB *summary*

- besides of all the new commands and functions, you need to know how to…
  - define and solve systems of linear equations
  - find the root of functions of one variable
  - find the zero crossings of vectors