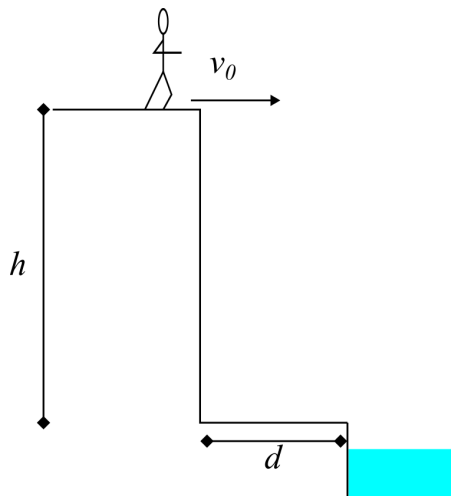


## Øving 2

### Oppgave 1

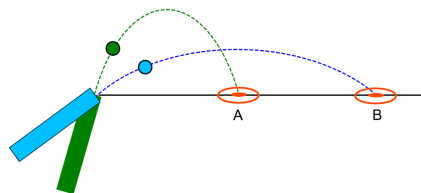


En modig svømmer stuper fra en klippe som har et horisontalt utspring som vist i figuren ovenfor. Utspringet har bredde  $d = 1,75$  m bredt og befinner seg en høyde  $h = 9,00$  m nedenfor toppen av klippen.

Hvor høy horisontal hastighet  $v_0$  må svømmeren minst ha på toppen av klippen for å akkurat unngå utspringet?

### Oppgave 2

a) To fjærdrevne kanoner med munninger i samme punkt avfyres samtidig. De skyter kuler med samme startfart, men i hver sin vinkel, som illustrert i figuren under. Begge kanoner har munning på bakkenivå, samme som blinkene A og B. Den grønne kanonen skyter mer vertikalt mot blink A, mens den blå skyter mer horisontalt mot blink B. Vi ser bort i fra luftmotstand.

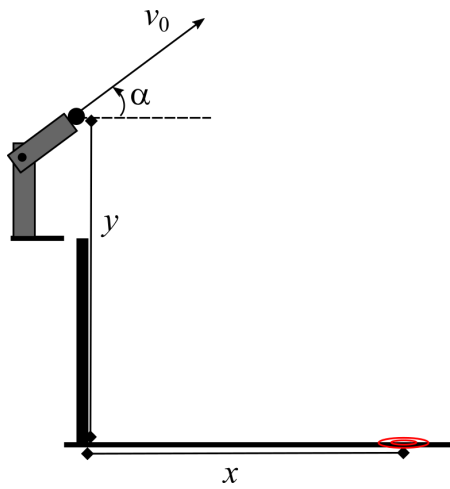


Hvilke av følgende påstander er riktige?

- A. Den grønne kanonen treffer blink A først.
- B. Den blå kanonen treffer blink B først.
- C. Blinkene treffes samtidig.
- D. Den blå kula treffer blink med større fart enn den grønne.
- E. Kulene treffer blinkene med samme fart.

### Oppgave 3

En fjærkanon skyter en kule med startfart  $v_0$  mot en blink som ligger i en horisontal avstand  $x$  og vertikal avstand  $y$  fra munningen, som illustrert i figuren under.



a) Sett opp en trigonometrisk likning for utskytingsvinkelen,  $\alpha$ , uttrykt ved  $v_0$ ,  $x$  og  $y$ .

b) Vi skal løse likninga numerisk i Python ved hjelp av funksjonen `fsolve` fra pakken `scipy.optimize`. Denne forutsetter at likninga som skal løses, skrives på formen  $f(x) = 0$ , dvs.  $f(x)$  er venstresiden i likninga når alle ledd er flyttet over slik at høyresiden er null.

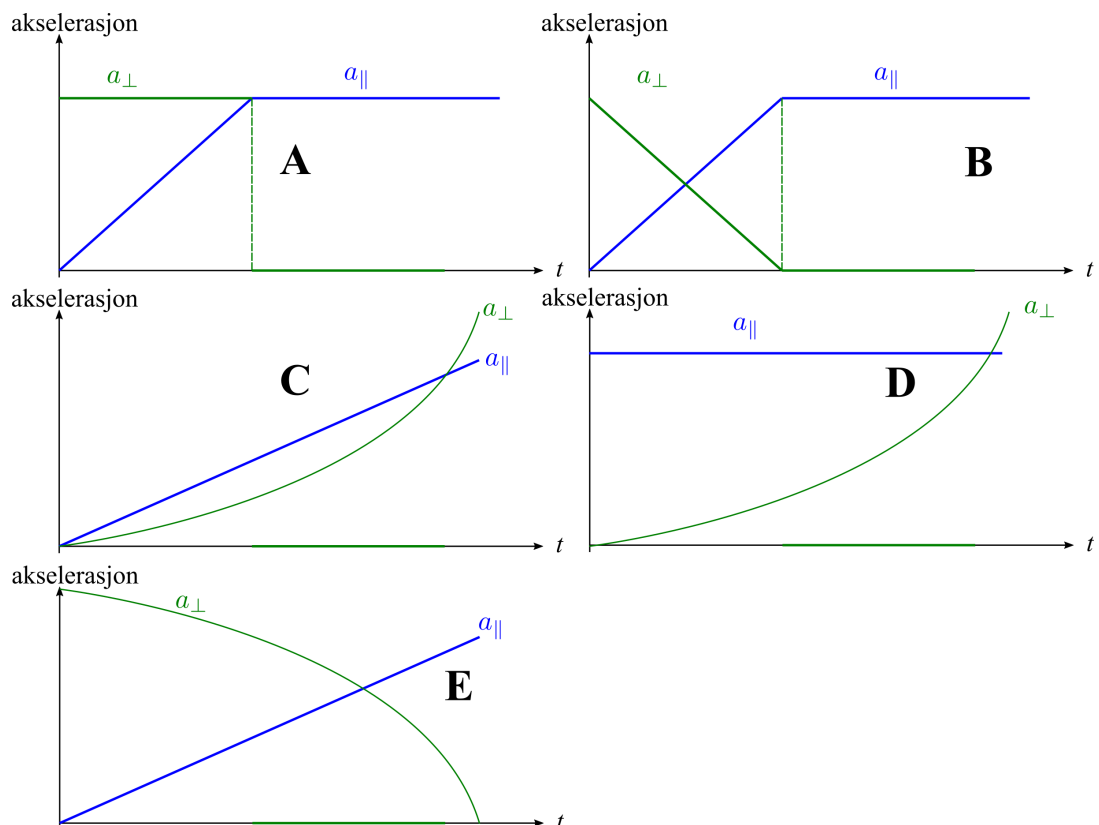
Bestem funksjonen  $f(\alpha)$  for å kunne løse likninga i a) ved hjelp av `fsolve`.

c) Hva må utskytingsvinkelen  $\alpha$  være for at kula skal treffe midt i blinken dersom startfarten  $v_0 = 4,0 \text{ m/s}$ ,  $x = 1,5 \text{ m}$  og  $y = 0,40 \text{ m}$ ? NB! Det kan finnes flere gyldige vinkler. Eksempelkoden nederst viser et eksempel på bruken av `fsolve`.

## Oppgave 4

a) En bil starter med null startfart og kjører med jevnt økende banefart i en sirkelformet rundkjøring. En passasjer i bilen bruker akselerometrene i mobiltelefonen til å måle baneakselerasjonen  $a_{\parallel}$  og sentripetalakselerasjonen  $a_{\perp}$  til bilen som funksjon av tiden.

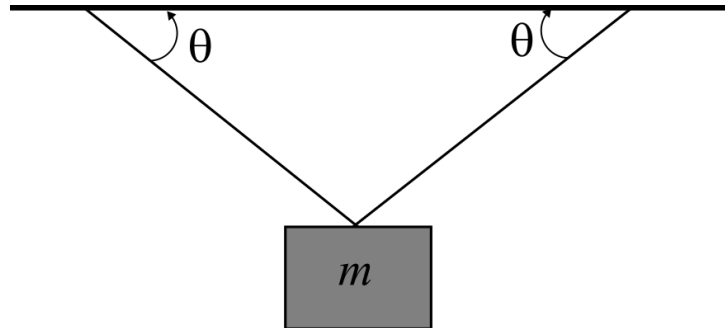
Hvilken av grafene under viser  $a_{\parallel}(t)$  og  $a_{\perp}(t)$  for bilen?



b) Hvor stor er baneakselerasjonen  $a_{\parallel}$ , sentripetalakselerasjonen  $a_{\perp}$  og den totale akselerasjonen  $a = |\vec{a}| = |\vec{a}_{\parallel} + \vec{a}_{\perp}|$  idet banefarten er 60 km/h dersom sirkelradien er 60 m og bilens banefart øker jevnt fra null til 60 km/h?

## Oppgave 5

Et trafikkllys med masse  $m$  henger i to identiske tau som danner en vinkel  $\theta$  med horisontalplanet, som vist på figuren under.



a) Vinkelen  $\theta$  kan justeres ved å gjøre tauene kortere/lengre, dvs. strammere/slakkere. Finn snordraget som funksjon av vinkelen  $\theta$ . Hva skjer med snordraget når  $\theta \rightarrow 0$ ?

b) Bestem draget i hvert tau dersom  $m = 50$  kg og  $\theta = 30^\circ$ .

## Eksempelkode: Løse likninger i Python

Funksjonen `fsolve` fra Python-pakken `scipy.optimize` løser likninger ved å finne nullpunkter til en funksjon. Likninga må skrives som  $f(x) = 0$ , der venstresiden i likninga utgjør  $f(x)$ . Ettersom `fsolve` bruker Newtons metode, må vi oppgi et startpunkt som 'gjetning' på hvor nullpunktet er. Dersom funksjonen har flere nullpunkter, må vi angi forskjellige startpunkter i nærheten av de ulike nullpunktene for å beregne disse.

```
In [ ]: #Importerer nødvendige pakker
from scipy.optimize import fsolve
import numpy as np
import matplotlib.pyplot as plt

#Definerer funksjonen som angir venstresiden i likninga f(x)=0 som skal løses; her 5cos(x)+x = 0
def f(x):
    return 5*np.cos(x) + x

#Tegner funksjonen for å få et bilde av løsningene:
t=np.linspace(-5,5)
plt.axis([-5,5,-10,10])
plt.grid()
plt.axhline(color='black', lw=0.5)
plt.plot(t,f(t))

#Ser at en løsning ligger i nærheten av x=-1; en annen i nærheten av x=2
start = -1
sol = fsolve(f, start)
print("Løsning i nærheten av x=-1: ",sol[0])

start = 2
sol = fsolve(f, start)
print("Løsning i nærheten av x=2: ",sol[0])
```

```
Løsning i nærheten av x=-1: -1.306440008369511
Løsning i nærheten av x=2: 1.977383029328841
```

