7.a) $\frac{dv_x}{dt} = 0$

EF

v_x: $\frac{dv_x}{dt} = \frac{v_x(t+\Delta t) - v_x(t)}{\Delta t} = 0$

$v_x(t+\Delta t) = v_x(t)$

v_z: $\frac{dv_z}{dt} = \frac{v_z(t+\Delta t) - v_z(t)}{\Delta t} = -g$

$v_z(t+\Delta t) = v_z(t) - g\Delta t$

c) $\frac{dx}{dt} = v_x \quad \wedge \quad \frac{dz}{dt} = v_z$

$\frac{dx}{dt} = \frac{x(t+\Delta t) - x(t)}{\Delta t} = v_x$

$\boxed{x(t+\Delta t) = v_x \Delta t + x(t)}$

same for z!

$\boxed{z(t+\Delta t) = v_z \Delta t + z(t)}$

since $v_x = \text{const} = v_0$

$\boxed{x(t+\Delta t) = v_0 \Delta t + x(t)}$

d) ~~$\frac{dv_x}{dt}$~~

$$\frac{dv_x}{dt} = -C v_x \sqrt{v_x^2 + v_z^2}$$

$$\frac{v_x(t+\Delta t) - v_x(t)}{\Delta t} = -C v_x \sqrt{v_x^2 + v_z^2}$$

$$\boxed{v_x(t+\Delta t) = -C v_x(t) \Delta t \sqrt{v_x(t)^2 + v_z(t)^2} + v_x(t)}$$

$$\frac{v_z(t+\Delta t) - v_z(t)}{\Delta t} = -g - C v_z(t) \sqrt{v_x^2 + v_z^2}$$

$$v_z(t+\Delta t) = \Delta t \left(-g - C v_z(t) \sqrt{v_x^2 + v_z^2}\right) + v_z(t)$$

e) $\frac{dx}{dt} = v_x \wedge \frac{dz}{dt} = v_z$

$dx = v_x(t)\, dt \quad (*)$

$\frac{dv_x}{dt} = 0$

$\int_0^t dv_x = 0$

$v_x(t) - v_x(0) = 0$

$v_x(t) = v_x(0) \quad (**)$

$(**)$ in $(*)$

$\int_0^t dx = [v_0 t]_0^t$

$x(t) - x(0) = v_0 \rightarrow \boxed{x(t) = v_0 t + x_0}$

$dz = v_z(t)\, dt$

$\int_0^t v_z = [-gt]_0^t$

$v_z(t) - v_z(0) = -gt$

$v_z(t) = v_z(0) - gt$

$\int_0^t dz = \int_0^t (v_z(0) - gt)\, dt$

$z(t) - z(0) = \left[v_z(0)t - \frac{1}{2}gt^2\right]_0^t$

$\rightarrow z(t) = v_z(0)t + z(0) - \frac{1}{2}gt^2$

t_1 & t_2 given $\phi = 45° \wedge v_0 = 24\,\frac{m}{s}$

t_1: $v_z(t_1) = 0$

$v_z(t) = v_z(0) - gt \rightarrow t = -\left(\frac{v_z(t) - v_z(0)}{g}\right)$

$t = \frac{v_z(0) - v_z(t)}{g}$

$t_1 = \frac{v_z(0)}{g} = \frac{24\,\frac{m}{s} \cdot \sin(45°)}{9{,}81\,\frac{m}{s^2}} = \underline{1{,}73\,s}$

t_2: $v_z(t_2) = -v_0 \sin\phi$

$t_2 = \frac{v_z(0) + v_z(0)}{g} = \underline{\underline{3{,}46\,s}}$

sheet5

September 28, 2017

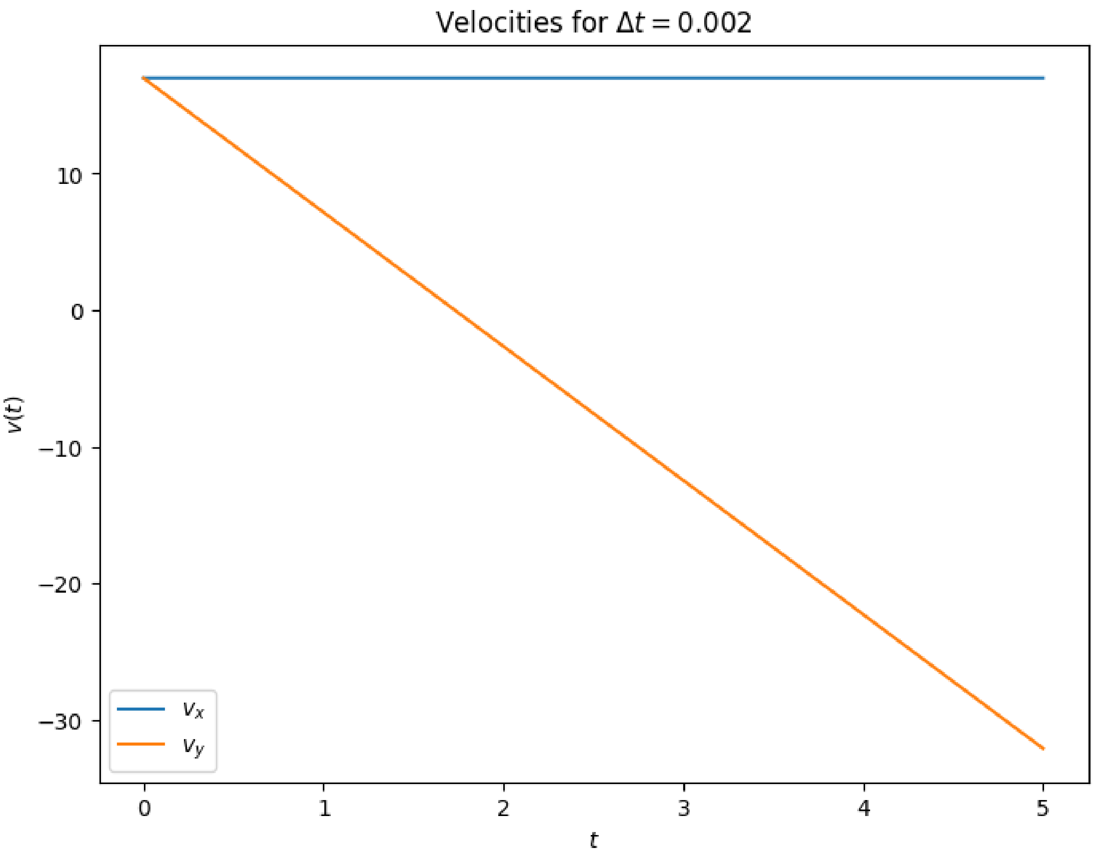
1 7. Motion of a ball in Earth's gravitational field

1.1 (a)

```
In [1]: using PyPlot
```

```
In [2]: #parameters
        g=9.81
        v0=24
        #grad in radiant
        phi=45*pi/180

        dt=0.002
        #endtime in [s]
        tend=5

        #the velocities
        vx=Float64[]
        vy=Float64[]
        push!(vx,v0*cos(phi))
        push!(vy,v0*sin(phi))

        t=Float64[]
        push!(t,0)

        #i-iteration,tval-value in the interval
        for (i,tval) in enumerate(dt:dt:tend)
            #the derived formulas for the ef-scheme
            push!(vx,vx[i])
            push!(vy,vy[i]-g*dt)
            push!(t,tval)
        end

        #just the plot
        figure(1,figsize=(8,6))
        plot(t,vx,label=L"$v_x$")
        plot(t,vy,label=L"$v_y$")
        title(L"Velocities for $\Delta t=0.002$")
```

```
ylabel(L"$v(t)$")
xlabel(L"$t$")
legend()
```

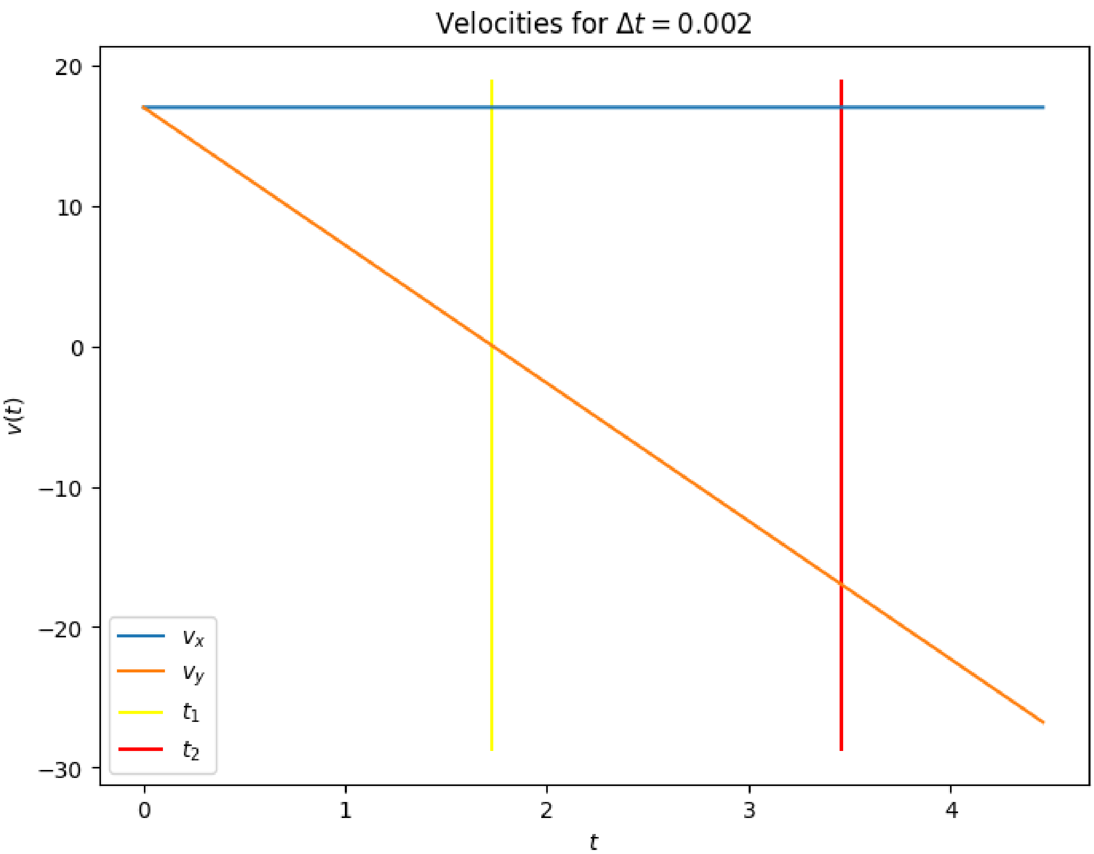


Out[2]: PyObject <matplotlib.legend.Legend object at 0x7fc5f8742890>

1.2 (b)

In [7]:
```
#parameters
g=9.81
v0=24
#grad in radiant
phi=45*pi/180

dt=0.002

#wished vy values
vt=[0.,-v0*sin(phi)]

#the velocities
```

```
vx=Float64[]
vy=Float64[]
#time
t=Float64[]
#tvals pos[1]-t1 & [2]-t2
tvals=Float64[]

#for both wished velocities
#since I do a "push" before the routine, the first array entry in julia is at "1" and
for val in vt
    push!(vy,v0*sin(phi))
    i=0
    #while the error is to high, do
    while(abs(vy[i+1]-val)>10e-3)
        push!(vy,vy[i+1]-g*dt)
        i+=1
    end
    push!(tvals,i*dt)
    vy=[]
end

push!(vx,v0*cos(phi))
push!(vy,v0*sin(phi))
push!(t,0)

#i-iteration,tval-value in the interval
#here i starts at 1, so the value t-1 is at i, not i+1
for (i,tval) in enumerate(dt:dt:tvals[2]+1)
    #the derived formulas for teh ef-scheme
    push!(vx,vx[i])
    push!(vy,vy[i]-g*dt)
    push!(t,tval)
end

#just the plot
figure(2,figsize=(8,6))
plot(t,vx,label=L"$v_x$")
plot(t,vy,label=L"$v_y$")
title(L"Velocities for $\Delta t=0.002$")
ylabel(L"$v(t)$")
xlabel(L"$t$")
vlines(tvals[1],vy[1]+2,vy[length(vy)]-2,label=L"$t_1$",color="yellow")
vlines(tvals[2],vy[1]+2,vy[length(vy)]-2,label=L"$t_2$",color="red")
legend()

println("t1= ",tvals[1])
println("t2= ",tvals[2])
```




```
t1= 1.73
t2= 3.46
```

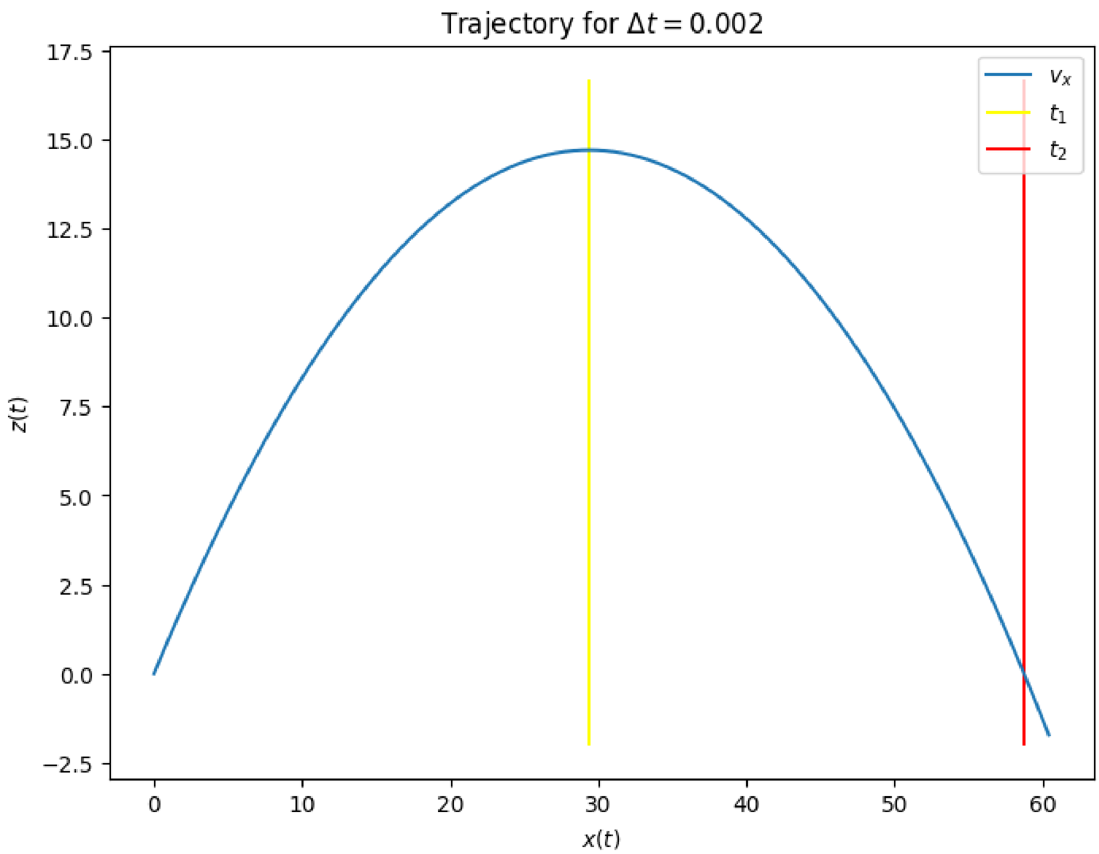
t1 is the turning point of the ball, t2 is the point where the ball hits the ground again

1.3 (c)

```
In [9]: #parameters
        g=9.81
        v0=24
        #grad in radiant
        phi=45*pi/180

        dt=0.002

        #wished vy values
        vt=[0.,-v0*sin(phi)]

        #the velocities
```

```
vx=Float64[]
vy=Float64[]

#the trajectories
x=Float64[]
y=Float64[]

#time
t=Float64[]

push!(vx,v0*cos(phi))
push!(vy,v0*sin(phi))
push!(x,0)
push!(y,0)

#I want the y values for t1 and the x-coordinate
maxima=Float64[]
maximax=Float64[]

#wished vy values
vt=[0.,-v0*sin(phi)]

#i-iteration,tval-value in the interval
#for the explanation of i look in b)
for (i,tval) in enumerate(dt:dt:tvals[2]+0.1)
    #the derived formulas for the ef-scheme
    push!(x,vx[i]*dt+x[i])
    push!(y,vy[i]*dt+y[i])
    push!(vx,vx[i])
    push!(vy,vy[i]-g*dt)
    if abs(vy[i+1]-vt[1])<10e-3 || abs(vy[i+1]-vt[2])<10e-3
        #current calue
        push!(maxima,y[i+1])
        push!(maximax,x[i+1])
    end
end

#just the plot
figure(2,figsize=(8,6))
plot(x,y,label=L"$v_x$")
#plot(t,vy,label=L"$v_y$")
title(L"Trajectory for $\Delta t=0.002$")
ylabel(L"$z(t)$")
xlabel(L"$x(t)$")
vlines(maximax[1],maxima[1]+2.,-2.,label=L"$t_1$",color="yellow")
vlines(maximax[2],maxima[1]+2.,-2.,label=L"$t_2$",color="red")

legend()
```

```
println("x-Coordinates calculated by EF-scheme for x(t):")
println("x1= ",maximax[1])
println("x2= ",maximax[2])
```



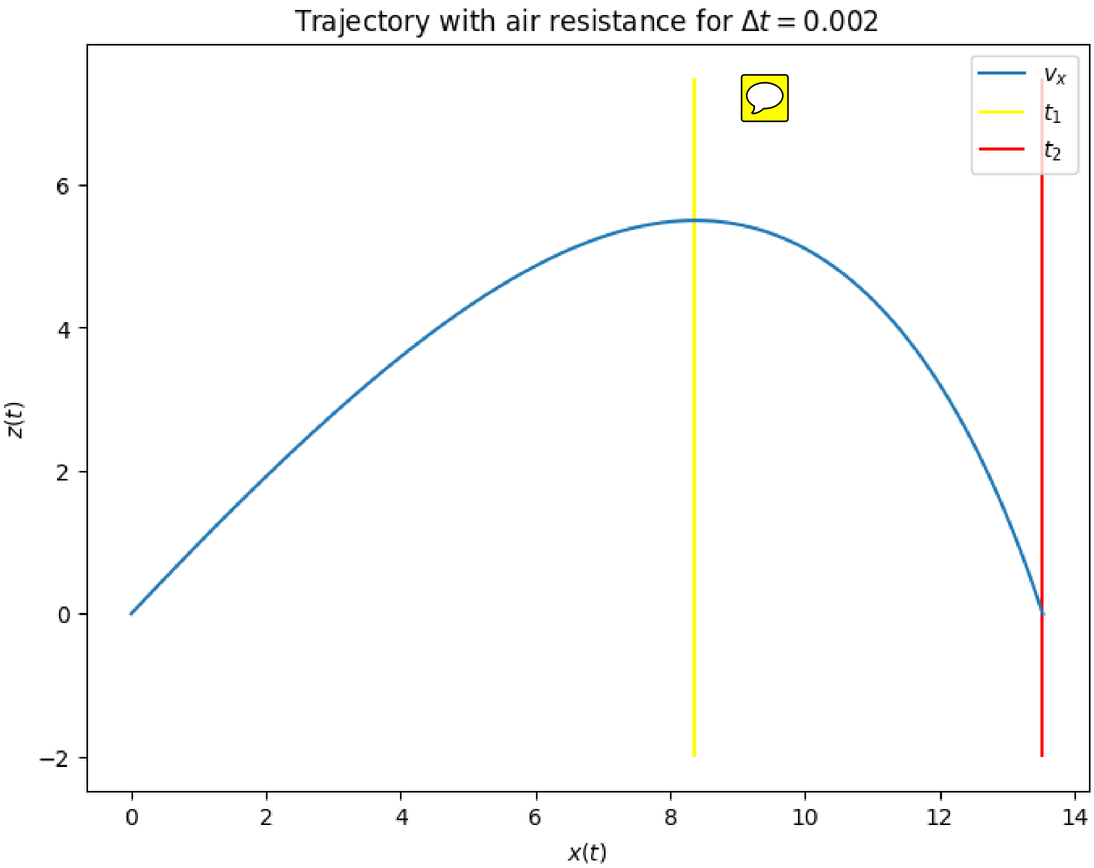

```
x-Coordinates calculated by EF-scheme for x(t):
x1= 29.35907355486583
x2= 58.718147109729536
```

1.4 (d)

```
In [5]: #parameters
        c=10e-2
        g=9.81
        v0=24
        #grad in radiant
        phi=45*pi/180

        dt=0.002
```

```
#wished vy values
vt=[0.,-v0*sin(phi)]

#the velocities
vx=Float64[]
vy=Float64[]

#the trajectories
x=Float64[]
y=Float64[]

#time
t=Float64[]

push!(vx,v0*cos(phi))
push!(vy,v0*sin(phi))
push!(x,0)
push!(y,0)

#I want the y values for t1 and the x-coordinate
maxima=Float64[]
maximax=Float64[]

#wished vy values
vt=[0.,-v0*sin(phi)]

i=1
while(y[i]>=0)
    #the derived formulas for the ef-scheme
    push!(x,vx[i]*dt+x[i])
    push!(y,vy[i]*dt+y[i])
    push!(vx,-c*vx[i]*dt*sqrt(vx[i]^2+vy[i]^2)+vx[i])
    push!(vy,dt*(-g-c*vy[i]*sqrt(vx[i]^2+vy[i]^2))+vy[i])
    if abs(vy[i+1]-vt[1])<10e-3
        #current calue
        push!(maxima,y[i+1])
        push!(maximax,x[i+1])
    end
    i+=1
end

#just the plot
figure(2,figsize=(8,6))
plot(x,y,label=L"$v_x$")
#plot(t,vy,label=L"$v_y$")
title(L"Trajectory with air resistance for $\Delta t=0.002$")
ylabel(L"$z(t)$")
```

```
xlabel(L"$x(t)$")
vlines(maximax[1],maxima[1]+2.,-2.,label=L"$t_1$",color="yellow")
vlines(x[length(x)-1],maxima[1]+2.,-2.,label=L"$t_2$",color="red")

legend()


println("x-Coordinates:")
println("x1= ",maximax[1])
println("x2= ",x[length(x)-1])
```

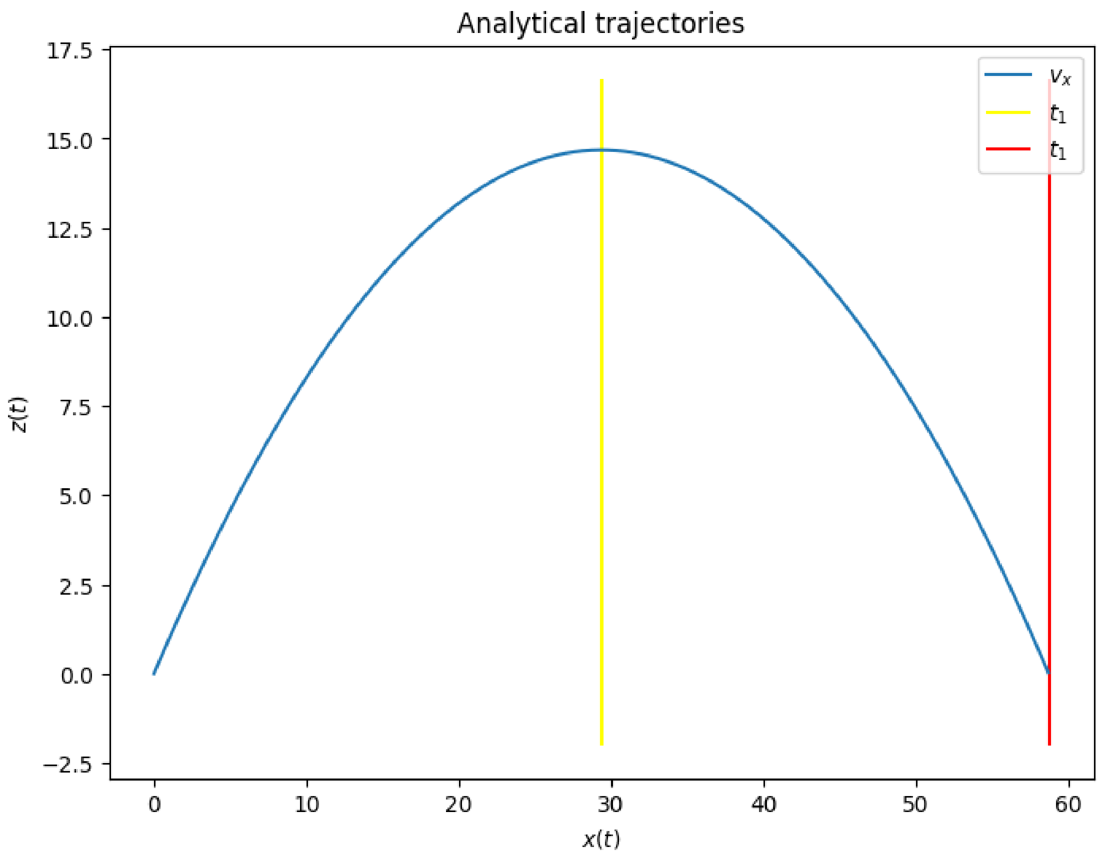



```
x-Coordinates:
x1= 8.363602196159704
x2= 13.524332727167792
```

1.5 (e)

```
In [6]: #parameters
        g=9.81
```

```
v0=24
#grad in radiant
phi=45*pi/180

dt=0.002

#wished vy values
vt=[0.,-v0*sin(phi)]

#the velocities
vx=Float64[]
vy=Float64[]

#wished vy values
vt=[0.,-v0*sin(phi)]

push!(vx,v0*cos(phi))
push!(vy,v0*sin(phi))


#analytical t1 & t2
tvals[1]=vy[1]/g
tvals[2]=2*vy[1]/g

#analytical trajectories
fx(t,v0,x0)=v0*t+x0
fy(t,v0,x0,g)=v0*t+x0-.5*g*t.^2.0

#the interval
t=0:dt:tvals[2]
#just the plot
figure(2,figsize=(8,6))
plot(fx(t,vx[1],0),fy(t,vy[1],0,g),label=L"$v_x$")
title("Analytical trajectories")
ylabel(L"$z(t)$")
xlabel(L"$x(t)$")
vlines(tvals[1]*vx[1],fy(tvals[1],vy[1],0,g)+2.,-2.,label=L"$t_1$",color="yellow")
vlines(tvals[2]*vx[1],fy(tvals[1],vy[1],0,g)+2.,-2.,label=L"$t_1$",color="red")

legend()

println("t-values:")
println("t1= ",tvals[1])
println("t2= ",tvals[2])
println("")
println("x-Coordinates:")
println("x1= ",tvals[1]*vx[1])
println("x2= ",tvals[2]*vx[1])
```




```
t-values:
t1= 1.7299248469395656
t2= 3.459849693879131

x-Coordinates:
x1= 29.357798165137616
x2= 58.71559633027523
```

As we see, the values are pretty much the same

```
In [ ]:
```