1.9 Solution

October 6, 2019

[1]:
```
import inspect
import numpy as np
var('t')
def euler(init, fn, step=0.1, end=1):
    res = [init]
    for t in np.arange(init[0] + step, end + step, step):
        if len(inspect.getargspec(fn).args) == 2:
            res.append([round(t * 1e5) / 1e5, res[-1][1] + step *fn(res[-1][1],␣
 ↪res[-1][0])])
        else:
            res.append([round(t * 1e5) / 1e5, res[-1][1] + step␣
 ↪*fn(res[-1][1])])
    return res
```

1 1.8.1

[2]:
```
table(euler([0,0.2], lambda y: y*(1-y), 0.5, 3))
```

[2]:

0	0.200000000000000
0.5	0.280000000000000
1.0	0.380800000000000
1.5	0.498695680000000
2.0	0.623694829374669
2.5	0.741044623967655
3.0	0.836993368595801

2 1.8.2

[3]:
```
table(euler([0,0], lambda y: cos(y), 0.25, 1))
```

[3]:

0	0
0.25	0.250000000000000
0.5	0.492228105427661
0.75	0.712548621735777
1.0	0.901723156014256

3 1.9.1

[4]:
```
f(y) = y
g(t) = exp(t)
```

[19]:
```
actual = [(t, g(t)) for t in np.arange(0, 1.25, 0.25)]
table(actual)
```

[19]:

0.0	1
0.25	1.2840254166877414
0.5	1.6487212707001282
0.75	2.117000016612675
1.0	2.718281828459045

[6]:
```
a = euler([0,1], lambda y: y, 0.25, 1)
table(a)
```

[6]:

0	1
0.25	1.25000000000000
0.5	1.56250000000000
0.75	1.95312500000000
1.0	2.44140625000000

[25]:
```
b = euler([0,1], lambda y:y, 0.05, 1)
table([x for i, x in enumerate(b) if i % 5 == 0])
```

[25]:

0	1
0.25	1.27628156250000
0.5	1.62889462677744
0.75	2.07892817941137
1.0	2.65329770514442

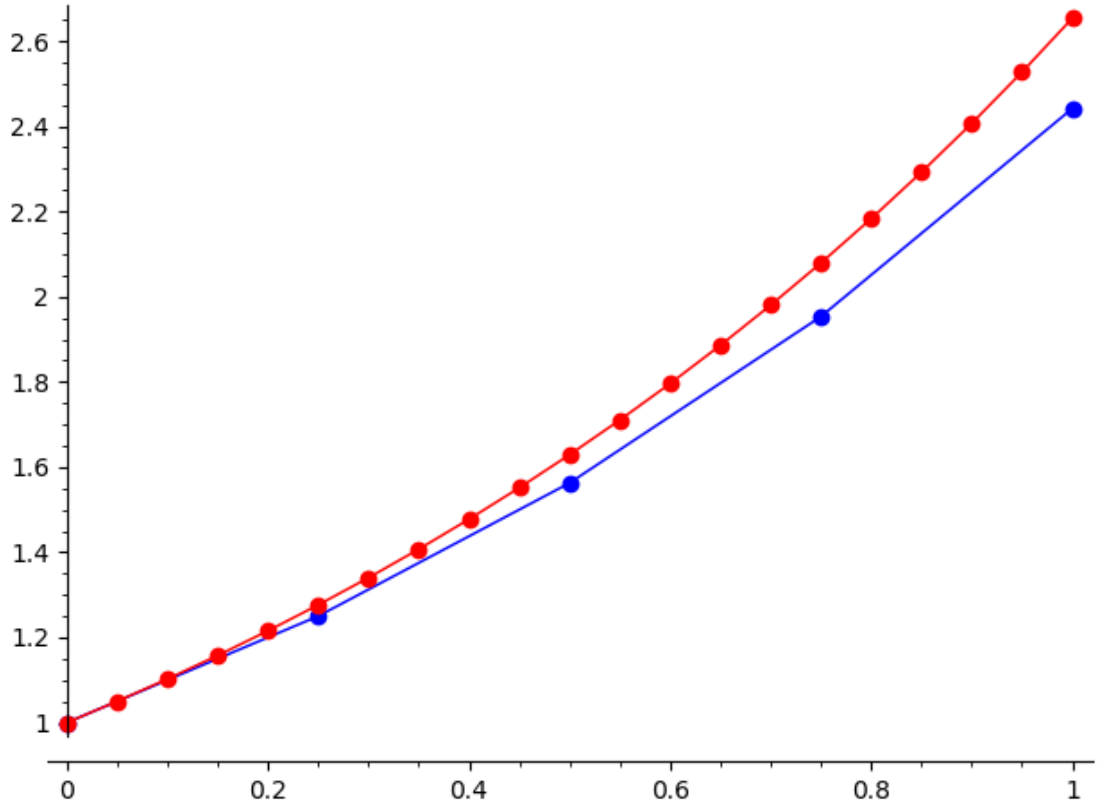
[22]:
```
err = []
for x in actual:
    for y in a:
        for z in b:
            if x[0] == y[0] and x[0] == z[0]:
                err.append([x[0], abs(x[1] - y[1]), abs(x[1] - z[1])])
table(err)
```

[22]:

0.0	0	0
0.25	0.034025416687741394	0.007743854187741039
0.5	0.0862212707001282	0.019826643922686182
0.75	0.16387501661267478	0.03807183720130647
1.0	0.2768755784590451	0.06498412331462378

[9]:
```
list_plot(a, plotjoined=True, marker='o') + list_plot(b,␣
↪plotjoined=True,marker='o',color='red')
```

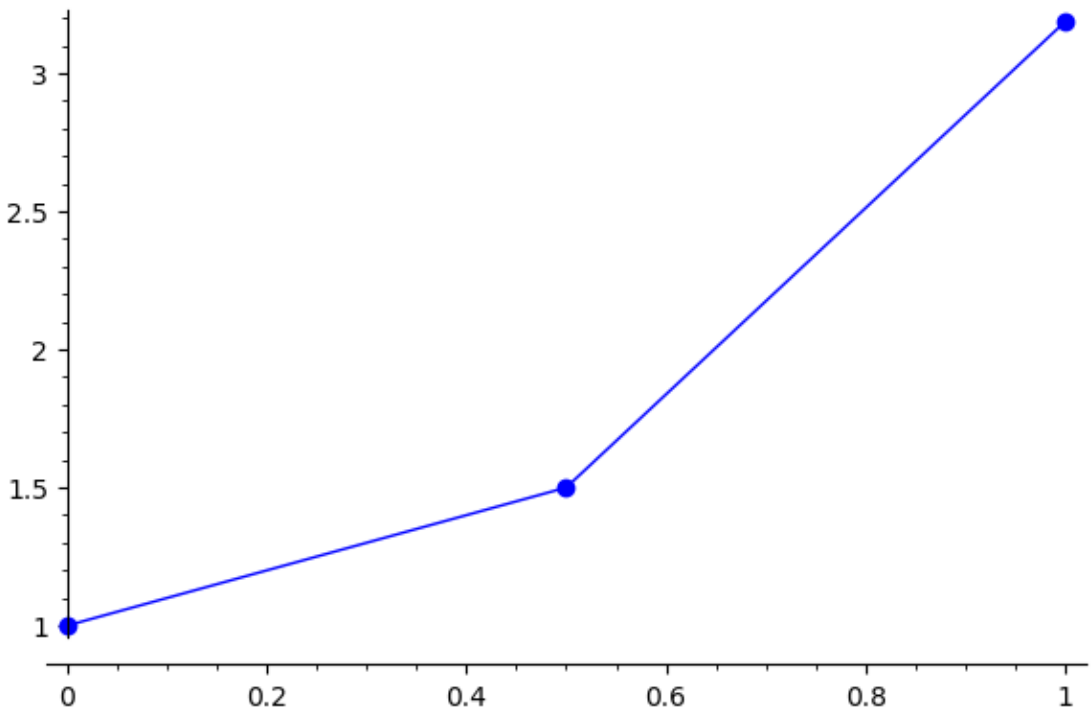
[9]:



4 1.9.2

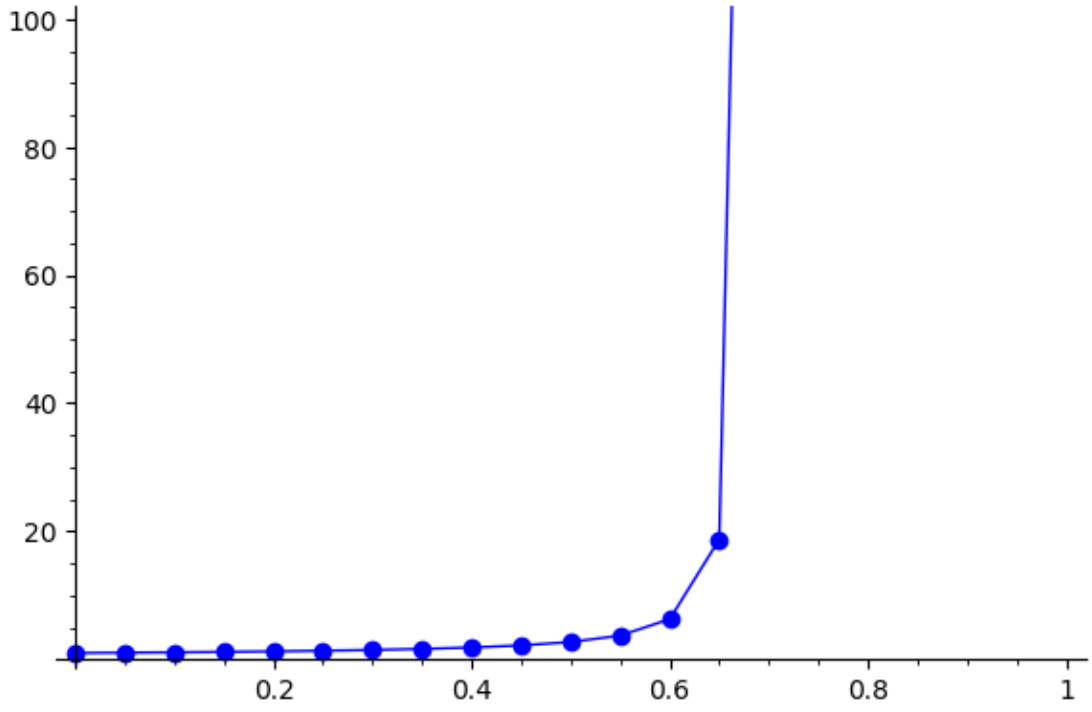
[10]:
```
list_plot(euler([0,1], lambda y: y**3, 0.5, 1), plotjoined=True,marker='o')
```

[10]:



[11]:
```
list_plot(euler([0,1], lambda y: y**3, 0.05, 1),␣
 ↪plotjoined=True,marker='o',ymax=100)
```

[11]:



[12]:
```
soln(t) = 1/sqrt(1-2*t)
derivative(soln(t), t) == soln(t)**3
```

[12]:
```
(-2*t + 1)^(-3/2) == (-2*t + 1)^(-3/2)
```

5 1.9.3

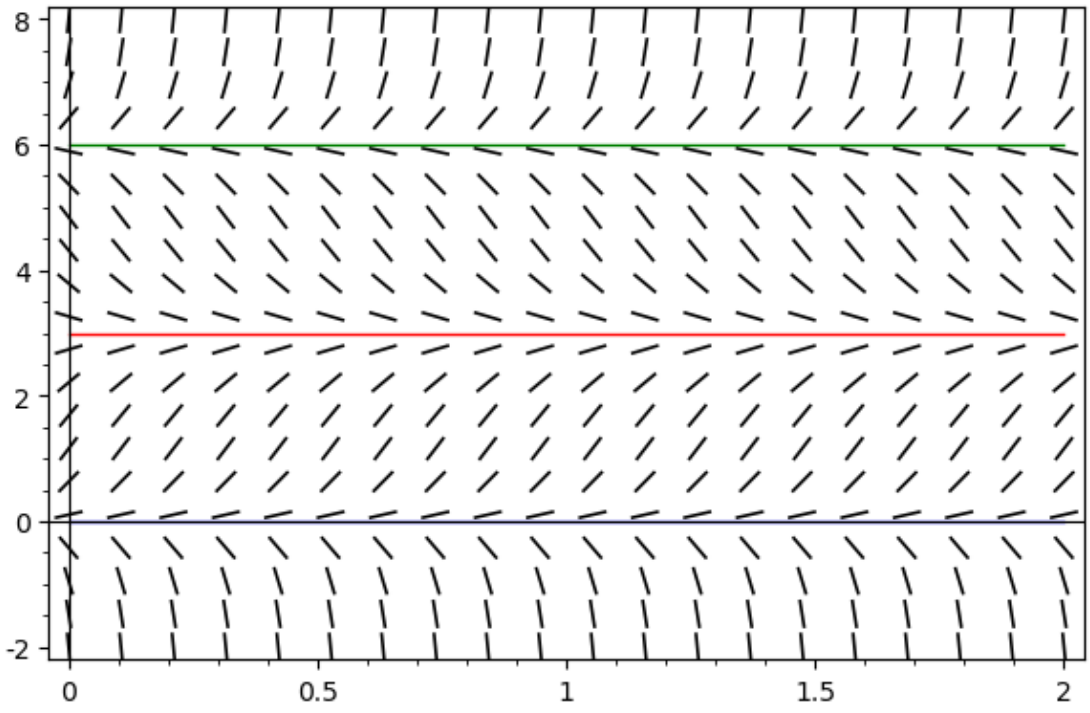
[30]:
```
table(euler([2,2], lambda y,t : -y**2+t, 0.1, 10)[-1])
```

[30]:
```
10.0    3.1368276055319497
```

6 1.9.4

[14]:
```
h(y) = y*(y-3)*(y-6)
plot_slope_field(h, (t,0,2), (y,-2,8)) + plot(0, (t,0,2), color='blue') +␣
↪plot(3, (t,0,2), color='red') + plot(6, (t,0,2), color='green')
```

[14]:



[15]:
```
d = euler([0,1], lambda y: y * (y-3) * (y-6), 0.5, 2)
table(d)
```

[15]:
```
0      1
0.5    6.00000000000000
1.0    6.00000000000000
1.5    6.00000000000000
2.0    6.00000000000000
```

[16]:
```
list_plot(euler([0,1], lambda y: y*(y-3)*(y-6), 0.5, 2), plotjoined=True,␣
↪marker='o') + plot_slope_field(h, (t,0,2), (y,-2,8)) + plot(0, (t,0,2),␣
↪color='blue') + plot(3, (t,0,2), color='red') + plot(6, (t,0,2),␣
↪color='green')
```

[16]: