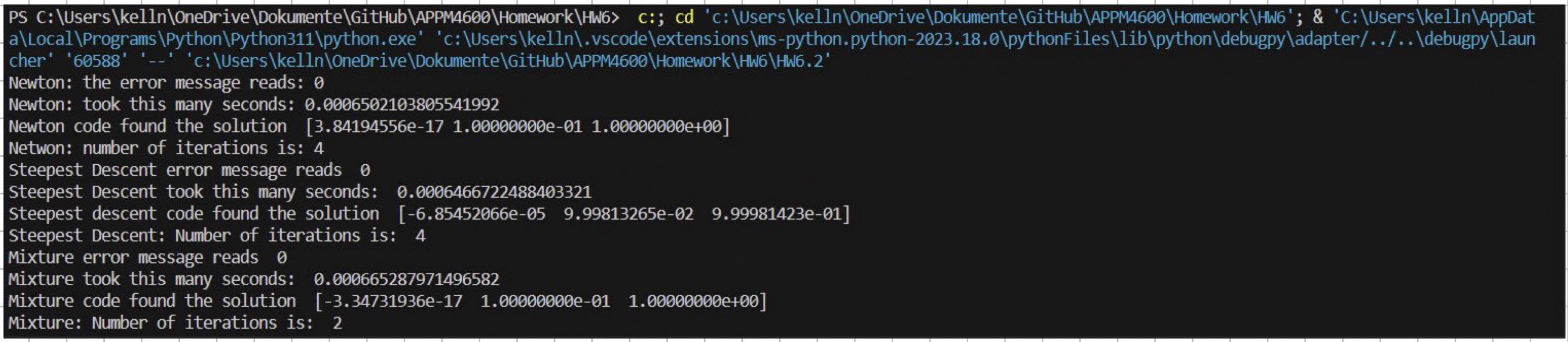
HW 6, APPM 4600, Lena Kellner   due 13th Oct

② Newton needs 4 iterations

Steepest Descent needs 4 iterations

The Mixture needs 2 iterations

$\Rightarrow$ The Mixture is the fastest because it combines the advantages of both methods: By using the steepest descent to find an initial guess for Newton's method, we avoid a poor chosen guess which would be challenging for the sensitive Newton method. By switching to Newton as soon as we have a good choice, we can use Newton's advantage of rapid convergence.



```
PS C:\Users\kelln\OneDrive\Dokumente\GitHub\APPM4600\Homework\HW6>  c:; cd 'c:\Users\kelln\OneDrive\Dokumente\GitHub\APPM4600\Homework\HW6'; & 'C:\Users\kelln\AppDat
a\Local\Programs\Python\Python311\python.exe' 'c:\Users\kelln\.vscode\extensions\ms-python.python-2023.18.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\laun
cher' '60588' '--' 'c:\Users\kelln\OneDrive\Dokumente\GitHub\APPM4600\Homework\HW6\HW6.2'
Newton: the error message reads: 0
Newton: took this many seconds: 0.0006502103805541992
Newton code found the solution  [3.84194556e-17 1.00000000e-01 1.00000000e+00]
Netwon: number of iterations is: 4
Steepest Descent error message reads  0
Steepest Descent took this many seconds:  0.0006466722488403321
Steepest descent code found the solution  [-6.85452066e-05  9.99813265e-02  9.99981423e-01]
Steepest Descent: Number of iterations is:  4
Mixture error message reads  0
Mixture took this many seconds:  0.000665287971496582
Mixture code found the solution  [-3.34731936e-17  1.00000000e-01  1.00000000e+00]
Mixture: Number of iterations is:  2
```

① i) Broyden is faster than Newton

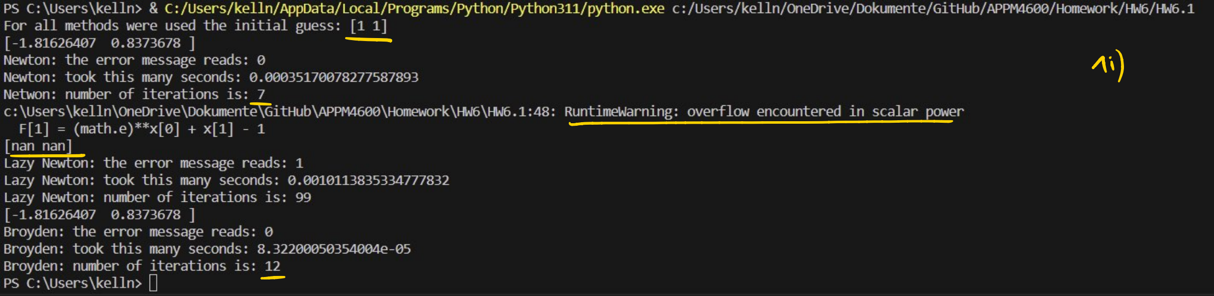
Lazy-Newton get's too big numbers because having to go through too much iterations

ii) The time of Broyden and Newton are not measurable different (both zero). But Newton needs one iteration less (5 vs. 6)

Lazy-Newton needs more time and way more iterations (36)

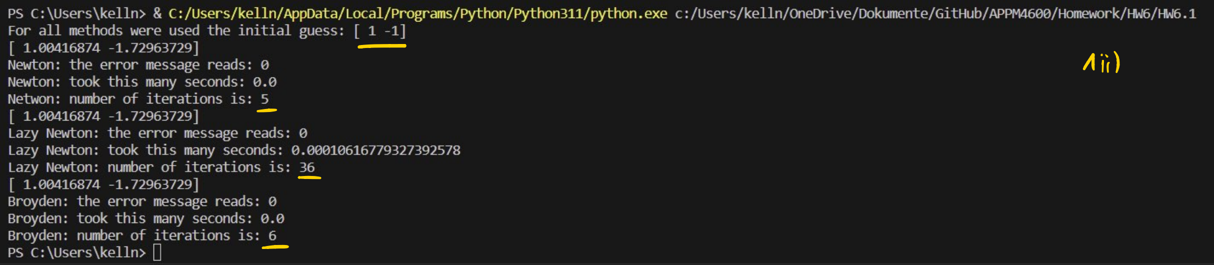
iii) None of the methods work because of having to invert a matrix with all entries = 0.

$\Rightarrow$ Conclusion: i's winner is Broyden, ii's winner is Newton, overall loser is Lazy-Newton and there are cases (f.e. iii) where none of the methods is successful!



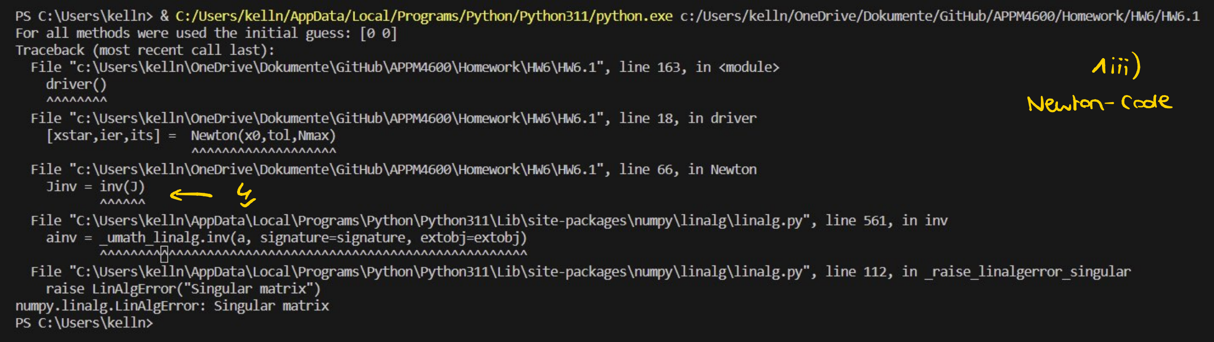
1i)

```
PS C:\Users\kelln> & C:/Users/kelln/AppData/Local/Programs/Python/Python311/python.exe c:/Users/kelln/OneDrive/Dokumente/GitHub/APPM4600/Homework/HW6/HW6.1
For all methods were used the initial guess: [1 1]
[-1.81626407  0.8373678 ]
Newton: the error message reads: 0
Newton: took this many seconds: 0.0003517007827758789З
Netwon: number of iterations is: 7
c:\Users\kelln\OneDrive\Dokumente\GitHub\APPM4600\Homework\HW6\HW6.1:48: RuntimeWarning: overflow encountered in scalar power
  F[1] = (math.e)**x[0] + x[1] - 1
[nan nan]
Lazy Newton: the error message reads: 1
Lazy Newton: took this many seconds: 0.0010113835334777832
Lazy Newton: number of iterations is: 99
[-1.81626407  0.8373678 ]
Broyden: the error message reads: 0
Broyden: took this many seconds: 8.32200050354004e-05
Broyden: number of iterations is: 12
PS C:\Users\kelln> 
```



1ii)

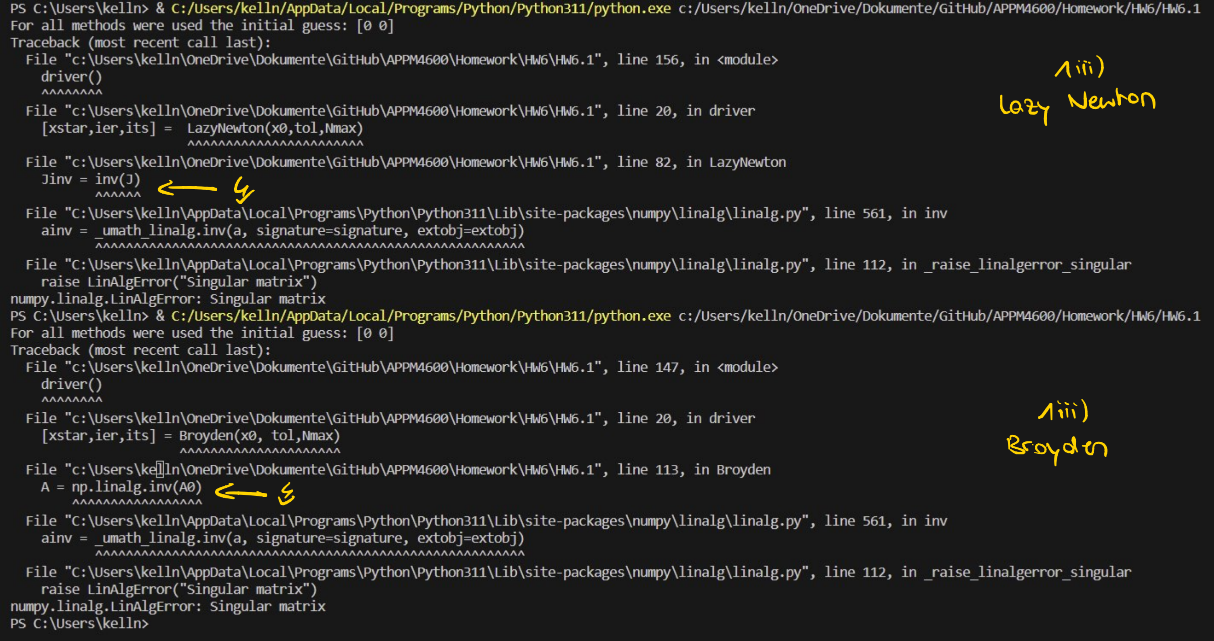
```
PS C:\Users\kelln> & C:/Users/kelln/AppData/Local/Programs/Python/Python311/python.exe c:/Users/kelln/OneDrive/Dokumente/GitHub/APPM4600/Homework/HW6/HW6.1
For all methods were used the initial guess: [ 1 -1]
[ 1.00416874 -1.72963729]
Newton: the error message reads: 0
Newton: took this many seconds: 0.0
Netwon: number of iterations is: 5
[ 1.00416874 -1.72963729]
Lazy Newton: the error message reads: 0
Lazy Newton: took this many seconds: 0.00010616779327392578
Lazy Newton: number of iterations is: 36
[ 1.00416874 -1.72963729]
Broyden: the error message reads: 0
Broyden: took this many seconds: 0.0
Broyden: number of iterations is: 6
PS C:\Users\kelln> 
```



1iii)

Newton-Code

```
PS C:\Users\kelln> & C:/Users/kelln/AppData/Local/Programs/Python/Python311/python.exe c:/Users/kelln/OneDrive/Dokumente/GitHub/APPM4600/Homework/HW6/HW6.1
For all methods were used the initial guess: [0 0]
Traceback (most recent call last):
  File "c:\Users\kelln\OneDrive\Dokumente\GitHub\APPM4600\Homework\HW6\HW6.1", line 163, in <module>
    driver()
    ^^^^^^^^
  File "c:\Users\kelln\OneDrive\Dokumente\GitHub\APPM4600\Homework\HW6\HW6.1", line 18, in driver
    [xstar,ier,its] =  Newton(x0,tol,Nmax)
                       ^^^^^^^^^^^^^^^^^^^
  File "c:\Users\kelln\OneDrive\Dokumente\GitHub\APPM4600\Homework\HW6\HW6.1", line 66, in Newton
    Jinv = inv(J)      ←    ↯
           ^^^^^^
  File "C:\Users\kelln\AppData\Local\Programs\Python\Python311\Lib\site-packages\numpy\linalg\linalg.py", line 561, in inv
    ainv = _umath_linalg.inv(a, signature=signature, extobj=extobj)
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\kelln\AppData\Local\Programs\Python\Python311\Lib\site-packages\numpy\linalg\linalg.py", line 112, in _raise_linalgerror_singular
    raise LinAlgError("Singular matrix")
numpy.linalg.LinAlgError: Singular matrix
PS C:\Users\kelln>
```



1 iii)
Lazy Newton

```
PS C:\Users\kelln> & C:/Users/kelln/AppData/Local/Programs/Python/Python311/python.exe c:/Users/kelln/OneDrive/Dokumente/GitHub/APPM4600/Homework/HW6/HW6.1
For all methods were used the initial guess: [0 0]
Traceback (most recent call last):
  File "c:\Users\kelln\OneDrive\Dokumente\GitHub\APPM4600\Homework\HW6\HW6.1", line 156, in <module>
    driver()
    ^^^^^^^^
  File "c:\Users\kelln\OneDrive\Dokumente\GitHub\APPM4600\Homework\HW6\HW6.1", line 20, in driver
    [xstar,ier,its] =  LazyNewton(x0,tol,Nmax)
                       ^^^^^^^^^^^^^^^^^^^^^^^
  File "c:\Users\kelln\OneDrive\Dokumente\GitHub\APPM4600\Homework\HW6\HW6.1", line 82, in LazyNewton
    Jinv = inv(J)
           ^^^^^^
  File "C:\Users\kelln\AppData\Local\Programs\Python\Python311\Lib\site-packages\numpy\linalg\linalg.py", line 561, in inv
    ainv = _umath_linalg.inv(a, signature=signature, extobj=extobj)
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\kelln\AppData\Local\Programs\Python\Python311\Lib\site-packages\numpy\linalg\linalg.py", line 112, in _raise_linalgerror_singular
    raise LinAlgError("Singular matrix")
numpy.linalg.LinAlgError: Singular matrix
PS C:\Users\kelln> & C:/Users/kelln/AppData/Local/Programs/Python/Python311/python.exe c:/Users/kelln/OneDrive/Dokumente/GitHub/APPM4600/Homework/HW6/HW6.1
For all methods were used the initial guess: [0 0]
Traceback (most recent call last):
  File "c:\Users\kelln\OneDrive\Dokumente\GitHub\APPM4600\Homework\HW6\HW6.1", line 147, in <module>
    driver()
    ^^^^^^^^
  File "c:\Users\kelln\OneDrive\Dokumente\GitHub\APPM4600\Homework\HW6\HW6.1", line 20, in driver
    [xstar,ier,its] = Broyden(x0, tol,Nmax)
                      ^^^^^^^^^^^^^^^^^^^^^
  File "c:\Users\kelln\OneDrive\Dokumente\GitHub\APPM4600\Homework\HW6\HW6.1", line 113, in Broyden
    A = np.linalg.inv(A0)
        ^^^^^^^^^^^^^^^^^
  File "C:\Users\kelln\AppData\Local\Programs\Python\Python311\Lib\site-packages\numpy\linalg\linalg.py", line 561, in inv
    ainv = _umath_linalg.inv(a, signature=signature, extobj=extobj)
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\kelln\AppData\Local\Programs\Python\Python311\Lib\site-packages\numpy\linalg\linalg.py", line 112, in _raise_linalgerror_singular
    raise LinAlgError("Singular matrix")
numpy.linalg.LinAlgError: Singular matrix
PS C:\Users\kelln>
```

$\leftarrow$ ↯

1 iii)
Broyden

$\leftarrow$ ↯