f-24-jupyter-inverspotents-og-rayleigh

April 29, 2021

[1]:
```python
import numpy as np
```

[2]:
```python
a = np.array([[2., 1., 1.],
              [1., 3., 1.],
              [1., 1., 4.]])
```

[3]:
```python
# inverspotensmetod

rng = np.random.default_rng()

mu = 1

w = rng.standard_normal((a.shape[0], 1))
w /= np.linalg.norm(w)

n = 15
lambda_out = np.empty(n)

for i in range(n):
    v = np.linalg.solve(a - mu * np.eye(a.shape[0]), w)
    w = v / np.linalg.norm(v)
    lambda_out[i] = w.T @ (a @ w)

print(lambda_out)
```

```
[1.39521076 1.32767566 1.3250032  1.32487573 1.32486946 1.32486915
 1.32486913 1.32486913 1.32486913 1.32486913 1.32486913 1.32486913
 1.32486913 1.32486913 1.32486913]
```

[4]:
```python
a @ w - lambda_out[-1]*w
```

[4]:
```
array([[ 4.20081747e-11],
       [ 1.33169697e-10],
       [-1.13811766e-10]])
```

[5]:
```python
# inverspotensmetoden med Householder QR

def house(x):
```

```python
    u = x / np.linalg.norm(x)
    eps = -1 if u[0] >= 0 else +1
    s = 1 + np.abs(u[0])
    v = - eps * u
    v[0] += 1
    v /= s
    return v, s

def householder_qr_data(a):
    data = np.copy(a)
    _, k = a.shape
    s = np.empty(k)
    for j in range(k):
        v, s[j] = house(data[j:, [j]])
        data[j:, j:] -= (s[j] * v) @ (v.T @ data[j:, j:])
        data[j+1:, [j]] = v[1:]
    return data, s

def householder_qr(a):
    data, s = householder_qr_data(a)
    n, k = a.shape
    r = np.triu(data[:k, :k])
    q = np.eye(n, k)
    for j in reversed(range(k)):
        x = data[j+1:, [j]]
        v = np.vstack([[1], x])
        q[j:, j:] -= (s[j] * v) @ (v.T @ q[j:, j:])
    return q, r
```

[6]:
```python
mu = 1

w = rng.standard_normal((a.shape[0], 1))
w /= np.linalg.norm(w)

q, r = householder_qr(a - mu * np.eye(a.shape[0]))

n = 15
lambda_out = np.empty(n)

for i in range(n):
    v = np.linalg.solve(r, q.T @ w)
    w = v / np.linalg.norm(v)
    lambda_out[i] = w.T @ (a @ w)

print(lambda_out)
```

```
[1.64829078 1.3415406  1.32568054 1.32490914 1.32487111 1.32486923
 1.32486913 1.32486913 1.32486913 1.32486913 1.32486913 1.32486913
```

```
 1.32486913 1.32486913 1.32486913]
```

[11]:

```
# Rayleighkvotientmetoden

w = rng.standard_normal((a.shape[0], 1))
w /= np.linalg.norm(w)

i = 0
lambda_est = w.T @ (a @ w)

while True:
    i += 1
    b = a - lambda_est * np.eye(a.shape[0])
    if np.linalg.det(b) == 0:    # ikke så god en test
        print('singulær')
        break
    v = np.linalg.solve(b, w)
    w = v / np.linalg.norm(v)
    lambda_est = w.T @ (a @ w)
    print(lambda_est)
    if i > 20:
        break
```

```
[[4.09903185]]
[[4.65645978]]
[[5.17274351]]
[[5.21431006]]
[[5.21431974]]
[[5.21431974]]
[[5.21431974]]
[[5.21431974]]
[[5.21431974]]
[[5.21431974]]
[[5.21431974]]
[[5.21431974]]
[[5.21431974]]
[[5.21431974]]
[[5.21431974]]
[[5.21431974]]
[[5.21431974]]
[[5.21431974]]
[[5.21431974]]
[[5.21431974]]
[[5.21431974]]
```

[10]:

```
a @ w - lambda_est * w
```

[10]: array([[-2.22044605e-16],
       [1.11022302e-16],
       [1.11022302e-16]])

[15]:
```
# Rayleighkvotientmetoden

w = rng.standard_normal((a.shape[0], 1))
w /= np.linalg.norm(w)

i = 0
lambda_est = w.T @ (a @ w)

while True:
    i += 1
    b = a - lambda_est * np.eye(a.shape[0])
    if np.allclose(a @ w, lambda_est * w,
                   atol = np.finfo(float).eps):
        print('færdig')
        break
    v = np.linalg.solve(b, w)
    w = v / np.linalg.norm(v)
    lambda_est = w.T @ (a @ w)
    print(lambda_est)
    if i > 20:
        break
```

```
[[3.29307874]]
[[2.69680962]]
[[2.4625968]]
[[2.46081113]]
[[2.46081113]]
færdig
```

[20]:
```
# Rayleighkvotientmetoden

w = rng.standard_normal((a.shape[0], 1))
w /= np.linalg.norm(w)

i = 0
lambda_est = 1

while True:
    i += 1
    b = a - lambda_est * np.eye(a.shape[0])
    if np.allclose(a @ w, lambda_est * w,
                   atol = np.finfo(float).eps):
        print('færdig')
        break
```

```
    v = np.linalg.solve(b, w)
    w = v / np.linalg.norm(v)
    lambda_est = w.T @ (a @ w)
    print(lambda_est)
    if i > 20:
        break
```

```
[[2.27266755]]
[[2.44918156]]
[[2.46080985]]
[[2.46081113]]
færdig
```

[ ]:

```

```