f-17-jupyter-householder-qr

April 6, 2021

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

[2]:
```
def house(x):
    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
```

[3]:
```
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
```

[4]:
```
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
```

[5]:
```
def householder_lsq(a, b):
    data, s = householder_qr_data(a)
    _, k = a.shape
    r = np.triu(data[:k, :k])
    c = np.copy(b)
    for j in range(k):
```

```
        x = data[j+1:, [j]]
        v = np.vstack([[1], x])
        c[j:] -= (s[j] * np.vdot(v, c[j:])) * v
    return np.linalg.solve(r, c[:k])
```

[6]:
```
a = np.array([[ 1.0, 2.0],
              [-1.0, 2.0],
              [ 0.0, 1.0]])
```

[7]:
```
q, r = householder_qr(a)
q, r
```

[7]:
```
(array([[-0.70710678, -0.66666667],
        [ 0.70710678, -0.66666667],
        [ 0.        , -0.33333333]]),
 array([[-1.41421356,  0.        ],
        [ 0.        , -3.        ]]))
```

[8]:
```
gram = q.T @ q
gram
```

[8]:
```
array([[1., 0.],
       [0., 1.]])
```

[10]:
```
gram - np.eye(2)
```

[10]:
```
array([[-2.22044605e-16,  0.00000000e+00],
       [ 0.00000000e+00,  4.44089210e-16]])
```

[11]:
```
a - q @ r
```

[11]:
```
array([[ 2.22044605e-16, -4.44089210e-16],
       [-2.22044605e-16, -4.44089210e-16],
       [ 0.00000000e+00,  0.00000000e+00]])
```

[12]:
```
np.linalg.qr(a)
```

[12]:
```
(array([[-0.70710678, -0.66666667],
        [ 0.70710678, -0.66666667],
        [-0.        , -0.33333333]]),
 array([[-1.41421356e+00,  6.66133815e-16],
        [ 0.00000000e+00, -3.00000000e+00]]))
```

[13]:
```
s = 1e-8
a = np.array([[1.0, 1.0, 1.0],
              [  s, 0.0, 0.0],
              [0.0,   s, 0.0],
              [0.0, 0.0,   s]])
```

[14]:
```
q, r = householder_qr(a)
q, r
```

[14]:
```
(array([[-1.00000000e+00,  7.07106781e-09,  4.08248290e-09],
        [-1.00000000e-08, -7.07106781e-01, -4.08248290e-01],
        [ 0.00000000e+00,  7.07106781e-01, -4.08248290e-01],
        [ 0.00000000e+00,  0.00000000e+00,  8.16496581e-01]]),
 array([[-1.00000000e+00, -1.00000000e+00, -1.00000000e+00],
        [ 0.00000000e+00,  1.41421356e-08,  7.07106781e-09],
        [ 0.00000000e+00,  0.00000000e+00,  1.22474487e-08]]))
```

[15]:
```
gram = q.T @ q
gram
```

[15]:
```
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```

[16]:
```
gram - np.eye(3)
```

[16]:
```
array([[ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
       [ 0.00000000e+00, -2.22044605e-16,  0.00000000e+00],
       [ 0.00000000e+00,  0.00000000e+00,  2.22044605e-16]])
```

[17]:
```
a - q @ r
```

[17]:
```
array([[ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
       [ 0.00000000e+00, -1.65436123e-24, -1.65436123e-24],
       [ 0.00000000e+00,  1.65436123e-24,  8.27180613e-25],
       [ 0.00000000e+00,  0.00000000e+00, -1.65436123e-24]])
```

[18]:
```
qnp, rnp = np.linalg.qr(a)
```

[19]:
```
gram_np = qnp.T @ qnp
gram_np
```

[19]:
```
array([[1.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       [0.00000000e+00, 1.00000000e+00, 5.55111512e-17],
       [0.00000000e+00, 5.55111512e-17, 1.00000000e+00]])
```

[20]:
```
gram_np - np.eye(3)
```

[20]:
```
array([[ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
       [ 0.00000000e+00,  4.44089210e-16,  5.55111512e-17],
       [ 0.00000000e+00,  5.55111512e-17, -2.22044605e-16]])
```

[21]:
```
a - qnp @ rnp
```

[21]:
```
array([[ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
       [ 0.00000000e+00,  3.30872245e-24,  4.13590306e-24],
       [ 0.00000000e+00, -1.65436123e-24, -1.65436123e-24],
       [ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00]])
```

[22]:
```
n = 300
rng = np.random.default_rng()
a = rng.standard_normal((n, n))
```

[23]:
```
qn, rn = householder_qr(a)
```

[24]:
```
signs = np.sign(np.diag(rn))
signs
```

[24]:
```
array([-1.,  1.,  1.,  1., -1.,  1.,  1., -1.,  1., -1.,  1., -1., -1.,
       -1., -1., -1.,  1.,  1., -1.,  1.,  1.,  1., -1., -1., -1.,  1.,
       -1.,  1., -1.,  1., -1.,  1.,  1., -1.,  1., -1.,  1., -1., -1.,
       -1.,  1.,  1.,  1.,  1.,  1.,  1.,  1., -1.,  1., -1., -1.,  1.,
       -1., -1., -1.,  1., -1.,  1.,  1., -1., -1., -1.,  1.,  1., -1.,
       -1.,  1., -1., -1.,  1.,  1., -1., -1., -1., -1., -1., -1., -1.,
        1.,  1.,  1.,  1.,  1., -1., -1., -1.,  1., -1.,  1.,  1.,  1.,
       -1., -1.,  1.,  1.,  1., -1., -1.,  1.,  1., -1., -1., -1., -1.,
       -1., -1.,  1.,  1.,  1., -1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,
       -1., -1., -1.,  1., -1., -1., -1., -1.,  1.,  1., -1., -1.,  1.,
        1., -1.,  1., -1.,  1.,  1., -1., -1., -1., -1.,  1.,  1., -1.,
        1., -1.,  1.,  1., -1.,  1., -1.,  1.,  1.,  1., -1.,  1.,  1.,
       -1., -1.,  1., -1., -1., -1., -1.,  1., -1.,  1., -1., -1., -1.,
        1., -1., -1.,  1., -1.,  1., -1., -1., -1.,  1.,  1., -1., -1.,
        1.,  1., -1., -1., -1.,  1.,  1., -1., -1.,  1., -1.,  1.,  1.,
        1., -1., -1., -1.,  1., -1.,  1.,  1.,  1., -1.,  1.,  1.,  1.,
       -1., -1., -1.,  1.,  1.,  1., -1.,  1.,  1.,  1.,  1.,  1., -1.,
        1.,  1., -1.,  1., -1.,  1., -1.,  1., -1.,  1., -1.,  1.,  1.,
       -1.,  1.,  1., -1., -1., -1., -1.,  1., -1.,  1., -1.,  1., -1.,
       -1.,  1.,  1.,  1.,  1.,  1., -1.,  1., -1.,  1., -1., -1.,  1.,
       -1., -1.,  1.,  1., -1.,  1., -1., -1., -1.,  1., -1.,  1.,  1.,
        1., -1.,  1.,  1.,  1., -1., -1., -1., -1.,  1.,  1.,  1.,  1.,
       -1., -1.,  1., -1.,  1.,  1., -1., -1.,  1.,  1., -1.,  1., -1.,
        1.])
```

[25]:
```
# retter q,r dekomponering så vi få alle r_ii > 0
qm = qn @ np.diag(signs)
rm = np.diag(signs) @ rn
```

[26]:
```
qm @ rm - qn @ rn
```

[26]:
```
array([[0., 0., 0., …, 0., 0., 0.],
       [0., 0., 0., …, 0., 0., 0.],
       [0., 0., 0., …, 0., 0., 0.],
```

```
…,
[0., 0., 0., …, 0., 0., 0.],
[0., 0., 0., …, 0., 0., 0.],
[0., 0., 0., …, 0., 0., 0.]])
```

[ ]: