f-15-jupyter-GS-problem

March 23, 2021

$$u_0 = \begin{bmatrix} 1 \\ s \\ 0 \\ 0 \end{bmatrix}, \quad u_1 = \begin{bmatrix} 1 \\ 0 \\ s \\ 0 \end{bmatrix}, \quad u_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ s \end{bmatrix}.$$

Klassisk Gram-Schmidt

$w_0 = u_0, \qquad v_0 = w_0/\|w_0\|$

$w_1 = u_1 - \mathrm{pr}_{v_0}(u_1), \qquad v_1 = w_1/\|w_1\|$

$w_2 = u_2 - \mathrm{pr}_{v_0} u_2 - \mathrm{pr}_{v_1} u_2, \qquad v_2 = w_2/\|w_2\|$

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

[2]:
```
s = 1e-8
u0 = np.array([1.0,   s, 0.0, 0.0])[:, np.newaxis]
u1 = np.array([1.0, 0.0,   s, 0.0])[:, np.newaxis]
u2 = np.array([1.0, 0.0, 0.0,   s])[:, np.newaxis]
```

[3]:
```
a = np.hstack([u0, u1, u2])
a
```

[3]:
```
array([[1.e+00, 1.e+00, 1.e+00],
       [1.e-08, 0.e+00, 0.e+00],
       [0.e+00, 1.e-08, 0.e+00],
       [0.e+00, 0.e+00, 1.e-08]])
```

[4]:
```
def proj_på(v, u):
    return np.vdot(v, u)/np.vdot(v, v) * v
```

[5]:
```
v0 = u0 / np.linalg.norm(u0)
w1 = u1 - proj_på(v0, u1)
v1 = w1 / np.linalg.norm(w1)
w2 = u2 - proj_på(v0, u2) - proj_på(v1, u2)
v2 = w2 / np.linalg.norm(w2)
q = np.hstack([v0, v1, v2])
q
```

```
[5]: array([[ 1.00000000e+00,  0.00000000e+00,  0.00000000e+00],
       [ 1.00000000e-08, -7.07106781e-01, -7.07106781e-01],
       [ 0.00000000e+00,  7.07106781e-01,  0.00000000e+00],
       [ 0.00000000e+00,  0.00000000e+00,  7.07106781e-01]])
```

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

```
[6]: array([[ 1.00000000e+00, -7.07106781e-09, -7.07106781e-09],
       [-7.07106781e-09,  1.00000000e+00,  5.00000000e-01],
       [-7.07106781e-09,  5.00000000e-01,  1.00000000e+00]])
```

```
[7]: np.arccos(np.vdot(v1, v2)/(np.linalg.norm(v1)*np.linalg.norm(v2))) * 360 /(2*np.
     ↪pi)
```

```
[7]: 59.99999999999999
```

```
[8]: v0 = u0 / np.linalg.norm(u0)
     w1 = u1 - proj_på(v0, u1)
     x2 = u2 - proj_på(v0, u2)
     v1 = w1 / np.linalg.norm(w1)
     w2 = x2 - proj_på(v1, x2)
```

```
[9]: v2 = w2 / np.linalg.norm(w2)
     q = np.hstack([v0, v1, v2])
     q
```

```
[9]: array([[ 1.00000000e+00,  0.00000000e+00,  0.00000000e+00],
       [ 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]])
```

```
[10]: q.T @ q
```

```
[10]: array([[ 1.00000000e+00, -7.07106781e-09, -4.08248290e-09],
       [-7.07106781e-09,  1.00000000e+00, -1.11022302e-16],
       [-4.08248290e-09, -1.11022302e-16,  1.00000000e+00]])
```

```
[11]: np.arccos(np.vdot(v1, v2)/(np.linalg.norm(v1)*np.linalg.norm(v2))) * 360 /(2*np.
      ↪pi)
```

```
[11]: 90.0
```

```
[12]: np.arccos(np.vdot(v0, v2)/(np.linalg.norm(v0)*np.linalg.norm(v2))) * 360 /(2*np.
      ↪pi)
```

```
[12]: 90.00000023390903
```

[13]:
```
1 + s**2
```

[13]: 1.0

[ ]:
```

```