hw7_code

October 20, 2021

Michael Goforth Homework 7 CAAM 550 10/20/21

[102]:
```
import numpy as np
import pandas as pd
import math
import scipy.linalg
import matplotlib.pyplot as plt
from matplotlib import cm
```

Problem 1

[103]:
```
xex = np.array([1,1,1])
t0vec = [0, 50, 100, 150, 200]

columnnames = ["t0", "Kappa2ATA", "Kappa2A", "||xnorm-xex||2", "||xorth-xex||2"]
data = pd.DataFrame(columns = columnnames)

for t0 in t0vec:
    b = np.zeros([10])
    A = np.zeros([10,3])
    for i in range(1, 11):
        ti = t0 + i
        b[i - 1] = xex[0] / ti + xex[1] / ti**2 + xex[2] / ti**3
        A[i - 1, 0] = 1 / ti
        A[i - 1, 1] = 1 / ti**2
        A[i - 1, 2] = 1 / ti**3
    xorth, res, rank, sv = np.linalg.lstsq(A, b, rcond=None)
    xnorm = np.linalg.solve(np.transpose(A) @ A, np.transpose(A) @ b)
    k2ata = np.linalg.cond(np.transpose(A) @ A, 2)
    k2a = np.linalg.cond(A, 2)
    normdiff = np.linalg.norm(xnorm - xex, 2)
    orthdiff = np.linalg.norm(xorth - xex, 2)
    data = data.append({
        't0': t0, "Kappa2ATA" : k2ata, "Kappa2A" : k2a, "||xnorm-xex||2" :␣
 ↪normdiff,
        "||xorth-xex||2" : orthdiff}, ignore_index=True)
data['t0'] = data['t0'].apply('{:.0f}'.format)
data['Kappa2ATA'] = data['Kappa2ATA'].apply('{:3.2e}'.format)
```

```
data['Kappa2A'] = data['Kappa2A'].apply('{:3.2e}'.format)
data['||xnorm-xex||2'] = data['||xnorm-xex||2'].apply('{:3.2e}'.format)
data['||xorth-xex||2'] = data['||xorth-xex||2'].apply('{:3.2e}'.format)
print(data.to_string(index=False))
```

```
 t0 Kappa2ATA  Kappa2A ||xnorm-xex||2 ||xorth-xex||2
  0  2.10e+03 4.58e+01       6.12e-14       1.08e-15
 50  1.66e+12 1.29e+06       1.24e-07       1.88e-10
100  2.88e+14 1.70e+07       1.21e-05       5.25e-10
150  6.45e+15 8.03e+07       6.00e-05       8.37e-10
200  6.01e+16 2.45e+08       3.69e-04       2.02e-08
```

Problem 3

part iii.

[97]:
```
n = 10
t = np.zeros(n)
A = np.ones([n, 3])
b = np.zeros(n)
for i in range(1, 11):
    t[i - 1] = i / 5
    b[i - 1] = 1 + i / 5 + (i / 5)**2
    A[i - 1, 1] = i / 5
    A[i - 1, 2] = (i / 5)**2

# Householder 1
v1 = A[:, 0] + np.linalg.norm(A[:, 0], 2) * np.array([1, 0, 0, 0, 0, 0, 0, 0,␣
 ↪0, 0])
H1 = np.eye(n) - 2 * np.outer(v1, v1) / np.linalg.norm(v1, 2)**2
A2 = H1 @ A
# Householder 2
A2s = A2[1:, 1:]
v2 = A2s[:, 0] + np.linalg.norm(A2s[:, 0], 2) * np.array([1, 0, 0, 0, 0, 0, 0,␣
 ↪0, 0])
H2 = np.eye(n)
H2[1:, 1:] = np.eye(n-1) - 2 * np.outer(v2, v2) / np.linalg.norm(v2, 2)**2
A3 = H2 @ A2
# Householder 3
A3s = A3[2:, 2:]
v3 = A3s[:, 0] + np.linalg.norm(A3s[:, 0], 2) * np.array([1, 0, 0, 0, 0, 0, 0,␣
 ↪0])
H3 = np.eye(n)
H3[2:, 2:] = np.eye(n-2) - 2 * np.outer(v3, v3) / np.linalg.norm(v3, 2)**2
R = H3 @ A3
Q = np.transpose(H3 @ H2 @ H1)
```

```
c = (np.transpose(Q) @ b)[:3]
xhouse = np.zeros(3)
xhouse[2] = c[2] / R[2, 2]
xhouse[1] = (c[1] - R[1, 2] * xhouse[2]) / R[1, 1]
xhouse[0] = (c[0] - R[0,1] * xhouse[1] - R[0,2] * xhouse[2]) / R[0,0]

xnp, res, rank, sv = np.linalg.lstsq(A, b, rcond=None)
print('||xnp - xhouse||_2 = '+ str(np.linalg.norm(xnp - xhouse, 2)))
print('||xtrue - xhouse||_2 = '+ str(np.linalg.norm(np.array([1, 1, 1]) -␣
 ↪xhouse, 2)))
print('||xtrue - xnp||_2 = '+ str(np.linalg.norm(xnp - np.array([1, 1, 1]), 2)))
```

```
||xnp - xhouse||_2 = 1.8676842118715256e-15
||xtrue - xhouse||_2 = 1.5463617361248926e-15
||xtrue - xnp||_2 = 4.577566798522237e-16
```

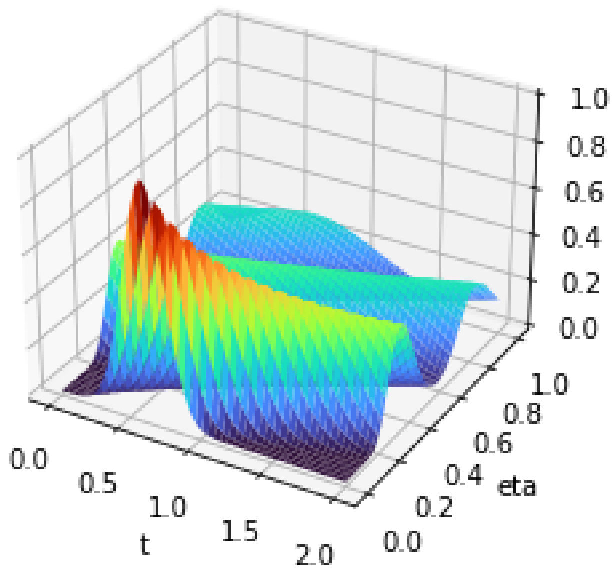
Problem 4

part i.

[104]:
```
# Constants
n = 100
beta = 1
alpha = .01
h = 1 / (n + 1)

# Build x0 and B
x0 = np.zeros(n)
B = np.zeros([n, n])
for j in range(n):
    psij = j * h
    x0[j] = math.exp(-(psij - .3)**2 / .01)
    B[j, j] = 2 * alpha + h * beta
    if j < n - 1:
        B[j, j + 1] = -alpha
    else:
        B[j, 0] = -alpha
    B[j, j - 1] = - (alpha + h * beta)
B = -h**-2 * B

x = np.zeros([n, n])
t = np.linspace(0, 2, n)
etagrid = np.linspace(0, 1, n)

X, Y = np.meshgrid(t, etagrid)

for i in range(t.size):
    x[i, :] = scipy.linalg.expm(B * t[i]) @ x0
```

```
fig, ax = plt.subplots(subplot_kw={"projection": "3d"})

surf = ax.plot_surface(X, Y, x, cmap=cm.turbo)

ax.set_zlim(0, 1)
ax.set_xlabel('t')
ax.set_ylabel('eta')
```

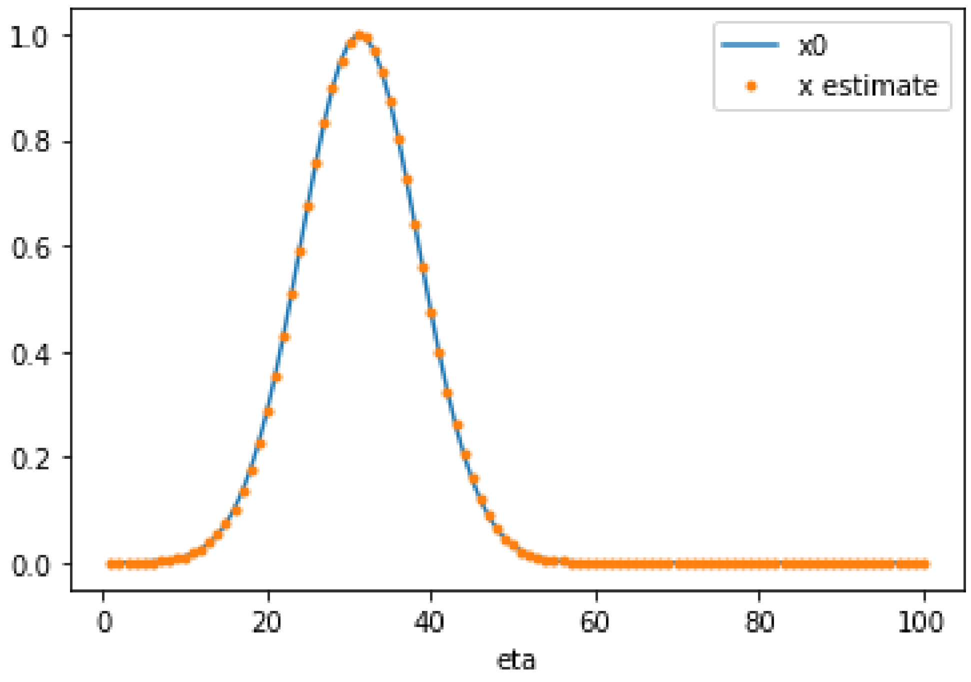
[104]:
```
Text(0.5, 0, 'eta')
```



part iii.

[105]:
```
# Variables from part i are reused here

k = 20
m = 25

H = np.zeros([k, n])
for i in range(n):
    for j in range(k):
        if j == (n / k) * i:
            H[i, j] = 1

tvec = np.linspace(.02, .5, m)
```

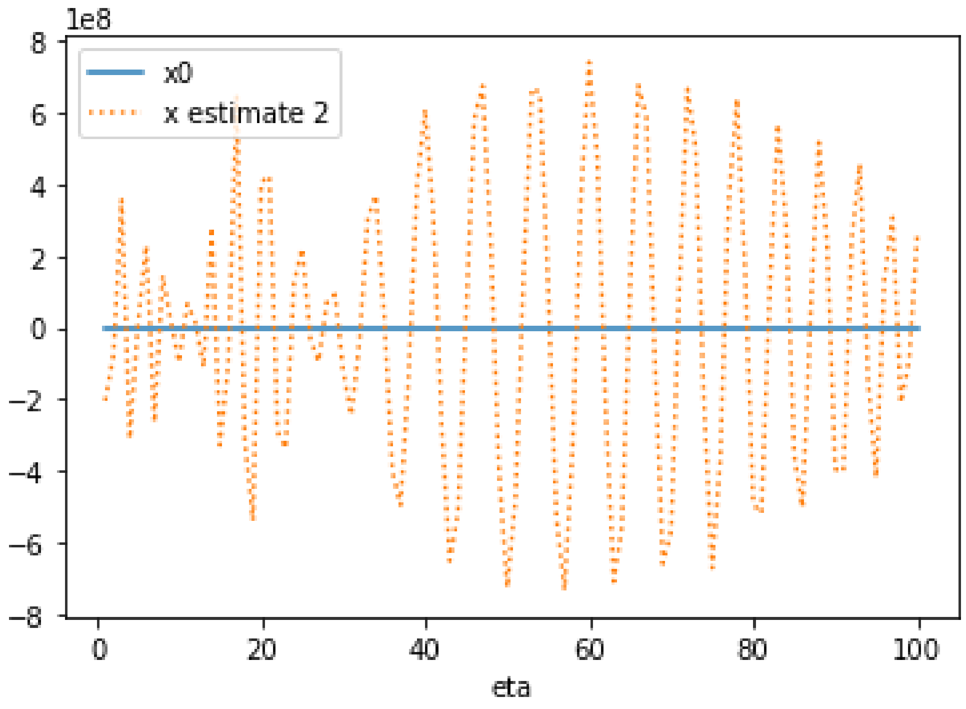
```
A = np.zeros([m * k, n])
b = np.zeros([k * m])

for i in range(m):
    A[k*(i):k*(i+1), :] = H @ scipy.linalg.expm(B * tvec[i])
    b[k*(i):k*(i+1)] = H @ scipy.linalg.expm(B * tvec[i]) @ x0

xest, res, rank, sv = np.linalg.lstsq(A, b, rcond=None)

xax = range(1, n+1)
plt.plot(xax, x0)
plt.plot(xax, xest, '.')
plt.legend(['x0', 'x estimate'])
plt.xlabel('eta')
plt.show()
```



part iv.

[106]:
```
# Variables from parts i and iii are reused here
zj = np.zeros([k * m])
for i in range(m):
    zj[k*(i):k*(i+1)] = (1 + .1 * np.random.randn(k, 1) ) * H @ scipy.linalg.
 ↪expm(B * tvec[i]) @ x0

xest2, res2, rank2, sv2 = np.linalg.lstsq(A, zj, rcond=None)
```

```
plt.plot(xax, x0)
plt.plot(xax, xest2, ':')
plt.legend(['x0', 'x estimate 2'])
plt.xlabel('eta')
plt.show()
```



part v.

[107]:
```
# Variables from parts i, iii, and iv are reused here

eps = np.finfo(np.float64).eps
effmax = max(k*m, n) * eps * abs(sv[0])
effrank = -1
for i in range(len(sv)):
    if sv[i] > effmax:
        effrank = i + 1
    else:
        break

print('Effective Rank of A: ' + str(effrank))
print('Sigma_1: ' + str(sv[0]))
print('Smallest non-zero sigma: ' + str(sv[effrank - 1]))
print('Conditioning number kappa_2: ' + str(sv[0] / sv[effrank - 1]))
```

```
Effective Rank of A: 44
Sigma_1: 1.2888887479272817
Smallest non-zero sigma: 1.8426742277136827e-13
Conditioning number kappa_2: 6994664214338.548
```

```
[ ]:
```