UNIVERSITY OF OSLO

MAT1120 - Assignment 2

Ivar Haugaløkken Stangeby

October 28, 2014

PROBLEM 1

Assume that $U = [\mathbf{u}_1, \ldots, \mathbf{u}_n]$ is an $(n \times n)$-matrix. Verify that U is ***semi-orthogonal*** if and only if $\mathbf{u}_j \neq \mathbf{0}$ for $j = 1, \ldots, n$ and all the $\mathbf{u}_j$'s are orthogonal to each other. In other words, $\mathbf{u}_j \cdot \mathbf{u}_i = 0$ when $j \neq i$.

Definition. A quadratic matrix U is called **semi-orthogonal** if $U^T U$ is a **diagonal** matrix with only positive entries along the main diagonal.

Definition. A matrix U is called **diagonal** if U has non-zero entries only along the main diagonal.

We now want to show that the condition is both sufficient and neccessary for the matrix to be ***semi-orthogonal***.
Let us first form the matrix $B = U^T U$.

$$B = \begin{pmatrix} \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_n \end{pmatrix} \begin{bmatrix} \mathbf{u}_1 & \ldots & \mathbf{u}_n \end{bmatrix} = \begin{bmatrix} \mathbf{u}_1 \cdot \mathbf{u}_1 & \ldots & \mathbf{u}_1 \cdot \mathbf{u}_n \\ \vdots & \ddots & \vdots \\ \mathbf{u}_n \cdot \mathbf{u}_1 & \ldots & \mathbf{u}_n \cdot \mathbf{u}_n \end{bmatrix}.$$

Assume now that each column in U is orthogonal to all the other columns. We can then directly see that all the entries in B that are not along the main diagonal must be zero. It is now sufficient to show that the entries along the main diagonal must be strictly positive. Let $\mathbf{u} \in U$ be an arbitrary vector from U.

$$\mathbf{u} \cdot \mathbf{u} = u_1^2 + \cdots + u_n^2.$$

Each term is squared, thus $\mathbf{u} \cdot \mathbf{u}$ is strictly positive.

If we instead were to assume that each column in U was **not** orthogonal to the other columns, we can directly from the matrix B see that we get non-zero entries that are not along the main diagonal, which contradict the condition that B must be a diagonal matrix. We can then conclude that U is ***semi-orthogonal*** if and only if the columns of U are orthogonal, and not the zero vector.
Further assume that U is ***semi-orthogonal*** and let

$$\mathbf{u}'_j = \frac{1}{\mathbf{u}_j \cdot \mathbf{u}_j}\mathbf{u}_j, \qquad j = 1, \ldots, n.$$

Argue that U is invertible and that

$$U^{-1} = \left[\mathbf{u}'_1 \ldots \mathbf{u}'_n\right]^T.$$

Let us look at the columns of $U, \{\mathbf{u}_1, \ldots, \mathbf{u}_n\}$. If the set of columns is an orthogonal set of non-zero vectors then the set is linearly independent. Since we assumed that U is ***semi-orthogonal***, then it follows that the set of columns is linearly independent.
By the Invertible Matrix Theorem, U is invertible.
We now want to show that the inverse of U is as given above. Let us check if $UU^{-1} = I$.

$$UU^{-1} = \begin{bmatrix}\mathbf{u}_1 & \ldots & \mathbf{u}_n\end{bmatrix}\begin{pmatrix}\mathbf{u}'_1 \\ \vdots \\ \mathbf{u}'_n\end{pmatrix} = \begin{bmatrix}\mathbf{u}_1 \cdot \mathbf{u}'_1 & \ldots & \mathbf{u}_1 \cdot \mathbf{u}'_n \\ \vdots & \ddots & \vdots \\ \mathbf{u}_n \cdot \mathbf{u}'_1 & \ldots & \mathbf{u}_n \cdot \mathbf{u}'_n\end{bmatrix}$$

Let $\mathbf{x}, \mathbf{y}' \in \{\mathbf{u}_1, \ldots, \mathbf{u}_n\}$.

- $\mathbf{x} = \mathbf{y} \Rightarrow \mathbf{x} \cdot \mathbf{y}' = 1$
- $\mathbf{x} \neq \mathbf{y} \Rightarrow \mathbf{x} \cdot \mathbf{y}' = 0$

We have that $\mathbf{x} = \mathbf{y}$ only along the main diagonal, therefore $UU^{-1} = I$.

PROBLEM 2

Divide $[0, \pi]$ into 3 intervals of length $\frac{\pi}{3}$ and let t_1, t_2 and t_3 be the midpoints of these intervals.
Let

$$C_3 = \begin{bmatrix}1 & \cos(t_1) & \cos(2t_1) \\ 1 & \cos(t_2) & \cos(2t_2) \\ 1 & \cos(t_3) & \cos(2t_3)\end{bmatrix} \qquad S_3 = \begin{bmatrix}\sin(t_1) & \sin(2t_1) & \sin(3t_1) \\ \sin(t_2) & \sin(2t_2) & \sin(3t_2) \\ \sin(t_3) & \sin(2t_3) & \sin(3t_3)\end{bmatrix}$$

Plugging in the values of t_1, t_2 and t_3 gives:

$$C_3 = \begin{bmatrix}1 & \frac{\sqrt{3}}{2} & \frac{1}{2} \\ 1 & 0 & -1 \\ 1 & -\frac{\sqrt{3}}{2} & \frac{1}{2}\end{bmatrix} \qquad S_3 = \begin{bmatrix}\frac{1}{2} & \frac{\sqrt{3}}{2} & 1 \\ 1 & 0 & -1 \\ \frac{1}{2} & -\frac{\sqrt{3}}{2} & 1\end{bmatrix}$$

These two matrices are ***semi-orthogonal*** because $C_3^T C_3$ and $S_3^T S_3$ are diagonal matrices with only positive entries along the main diagonal.

$$C_3^T C_3 = \begin{bmatrix} 3 & 0 & 0 \\ 0 & \frac{3}{2} & 0 \\ 0 & 0 & \frac{3}{2} \end{bmatrix} \qquad S_3^T S_3 = \begin{bmatrix} \frac{3}{2} & 0 & 0 \\ 0 & \frac{3}{2} & 0 \\ 0 & 0 & 3 \end{bmatrix}$$

Based on this we know that the inverses of these matrices are given by $U^{-1} = \begin{bmatrix} \mathbf{u}_1' & \dots & \mathbf{u}_n' \end{bmatrix}^T$ This gives us the inverses:

$$C_3^{-1} = \begin{bmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{\sqrt{3}}{3} & 0 & -\frac{\sqrt{3}}{3} \\ \frac{1}{3} & -\frac{2}{3} & \frac{1}{3} \end{bmatrix} \qquad S_3^{-1} = \begin{bmatrix} \frac{1}{3} & \frac{2}{3} & \frac{1}{3} \\ \frac{\sqrt{3}}{3} & 0 & -\frac{\sqrt{3}}{3} \\ \frac{1}{3} & -\frac{1}{3} & \frac{1}{3} \end{bmatrix}$$

PROBLEM 3

According to my python program both the matrices C and S are ***semi-orthogonal***, with the following inverses:

$$C^{-1} = \begin{bmatrix} 0.125 & 0.125 & 0.125 & 0.125 & 0.125 & 0.125 & 0.125 & 0.125 \\ 0.245 & 0.208 & 0.139 & 0.049 & -0.049 & -0.139 & -0.208 & -0.245 \\ 0.231 & 0.096 & -0.096 & -0.231 & -0.231 & -0.096 & 0.096 & 0.231 \\ 0.208 & -0.049 & -0.245 & -0.139 & 0.139 & 0.245 & 0.049 & -0.208 \\ 0.177 & -0.177 & -0.177 & 0.177 & 0.177 & -0.177 & -0.177 & 0.177 \\ 0.139 & -0.245 & 0.049 & 0.208 & -0.208 & -0.049 & 0.245 & -0.139 \\ 0.096 & -0.231 & 0.231 & -0.096 & -0.096 & 0.231 & -0.231 & 0.096 \\ 0.049 & -0.139 & 0.208 & -0.245 & 0.245 & -0.208 & 0.139 & -0.049 \end{bmatrix}$$

$$S^{-1} = \begin{bmatrix} 0.049 & 0.139 & 0.208 & 0.245 & 0.245 & 0.208 & 0.139 & 0.049 \\ 0.096 & 0.231 & 0.231 & 0.096 & -0.096 & -0.231 & -0.231 & -0.096 \\ 0.139 & 0.245 & 0.049 & -0.208 & -0.208 & 0.049 & 0.245 & 0.139 \\ 0.177 & 0.177 & -0.177 & -0.177 & 0.177 & 0.177 & -0.177 & -0.177 \\ 0.208 & 0.049 & -0.245 & 0.139 & 0.139 & -0.245 & 0.049 & 0.208 \\ 0.231 & -0.096 & -0.096 & 0.231 & -0.231 & 0.096 & 0.096 & -0.231 \\ 0.245 & -0.208 & 0.139 & -0.049 & -0.049 & 0.139 & -0.208 & 0.245 \\ 0.125 & -0.125 & 0.125 & -0.125 & 0.125 & -0.125 & 0.125 & -0.125 \end{bmatrix}$$

Multiplying the matrices with the respective inverses yields the identity matrix in both cases.

PROBLEM 4

Let $\mathscr{F}([0,2\pi],\mathbb{R})$ be the vector space consisting of all real functions defined on $[0,\pi]$ and let $\mathscr{W}_c$ be its subspace that is spanned by the set

$$\mathscr{C} = \{1, \cos(t), \cos(2t), \dots, \cos(7t)\}$$

We showed in the previous problem that C is semi-orthogonal. It then follows that C is invertible, which means that the columns of C neccessarily has to be linearly independent. This, in turn, means that $\mathscr{C}$ is a basis for $\mathscr{W}_C$. There is a result regarding signal spaces that tells ut that it is sufficient to show that this holds for the discrete values $t = t_1, \dots, t_8$

PROBLEM 5

Let $T_c : \mathscr{W}_c \to \mathbb{R}^8$ be defined by:

$$T_c(h) = (h(t_1), \dots, h(t_8))$$

for $h \in \mathscr{W}_c$, and let $\epsilon = \{\mathbf{e}_1, \dots, \mathbf{e}_8\}$.

Verify that C is the matrix of T_c with respect to the bases $\mathscr{C}$ and ϵ, and show that T_c is an isomorphism.
C is a matrix with those properties if $T_c(\mathbf{x})$ maps $[\mathbf{x}]_{\mathscr{C}}$ to $[T_c(\mathbf{x})]_\epsilon$.
Let h be any arbitrary vector in $\mathscr{W}_c$ with respect to the basis $\mathscr{C}$

$$h = \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_8 \end{bmatrix}$$

.
Multiplying from the left by the matrix C, we get

$$Ch = \begin{bmatrix} 1 & \dots & \cos(7t_1) \\ \vdots & \dots & \vdots \\ 1 & \dots & \cos(7t_8) \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_8 \end{bmatrix} = [(h(t_1), \dots, h(t_8))]_\epsilon$$

Since h was any arbitrary vector from $\mathscr{W}_c$, this means that C is the matrix of T_c with respect to the bases $\mathscr{C}$ and ϵ.
Since $\mathscr{C}$ is a basis for $\mathscr{W}_c$, and ϵ is a basis for $\mathbb{R}^8$ it follows that T_c is a bijective transformation. Since T_c is linear, this means T_c is an isomorph transformation. The matrix for T_c^{-1} is C^{-1}.

PROBLEM 6

Let $g \in \mathscr{F}([0,\pi], \mathbb{R})$ and set $\mathbf{y} = \big(g(t_1), \dots, g(t_8)\big) \in \mathbb{R}^8$. In addition, let

$$g^c = T_c^{-1}(\mathbf{y}) \in \mathscr{W}_c$$

Show that g^c is an 8-midpoint interpolation of g on the interval $[0,\pi]$ that satisfy $\left[g^c\right]_C = C^{-1}\mathbf{y}$.
g^c is a function in $\mathscr{W}_c$ that is defined in such a way that g^c interpolate g in the midpoints $t_1, \dots, t_8$. This means that g^c is an 8-midpoint interpolation of g on $[0,\pi]$ since,

$$
\begin{aligned}
Cg^c &= CT_c^{-1}(\mathbf{y}) = CC^{-1}\mathbf{y} = \mathbf{y} \\
T_c(g^c) &= C\left[g^c\right]_C = y \Leftrightarrow \left[g^c\right]_C = C^{-1}\mathbf{y}
\end{aligned}
$$

.

Problem 7

Argue that V_l and V_o are subspaces of V.
We want to show that both V_l and V_o are closed under addition, multiplication by a scalar, and also that the zero-function is both spaces. Ill show the calculations for V_l, the same is proved for V_o similarly. Let $f_1, f_2 \in V_l$, and set $f_3 = f_1 + f_2$. We now want to show that $f_3 \in V_l$.

$$
f_3(-t) = (f_1 + f_2)(-t) = f_1(-t) + f_2(-t) = f_1(t) = f_2(t) = f_3(t)
$$

Thus V_l is closed under addition. Now, let $f_3 = cf_1, c \in \mathbb{R}$. We want to show that $f_3 \in V_l$

$$
f_3(-t) = (cf_1)(t) = cf_1(-t) = cf_1(t) = f_3(t) \in V_l.
$$

.

We now want to show that if $f \in V$, then $f_l \in V_l$ and $f_o \in V_o$ as well as $f = f_l + f_o$.
Assume $f \in V$. We then have:

$$
\begin{aligned}
f_l &= \frac{1}{2}\left(f + T\left(f\right)\right) \\
f_o &= \frac{1}{2}\left(f - T(f)\right) \\
f_l + f_o &= \frac{1}{2}2f = f
\end{aligned}
$$

To show $f_l \in V_l$ and $f_o \in V_o$.

$$
\begin{aligned}
\left[T(f_l)\right](t) &= \frac{1}{2}\left(\left[T(f)\right](t) + \left[T(T(f))\right](t)\right) \\
&= \frac{1}{2}\left(f(t) + f(-t)\right) = f_l(t)
\end{aligned}
$$

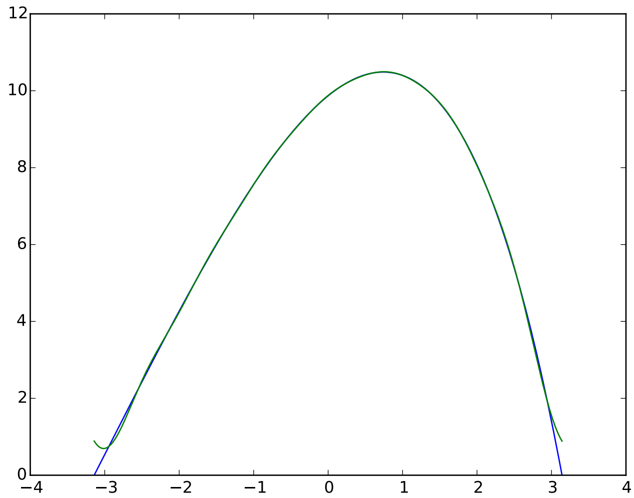
Thus $f_l \in V_l$.

$$
\begin{aligned}
\left[T(f_o)\right](t) &= \frac{1}{2}(f(-t) - f(t)) \\
&= f_o(-t) = -f_o(t).
\end{aligned}
$$

Thus $f_o \in V_o$.

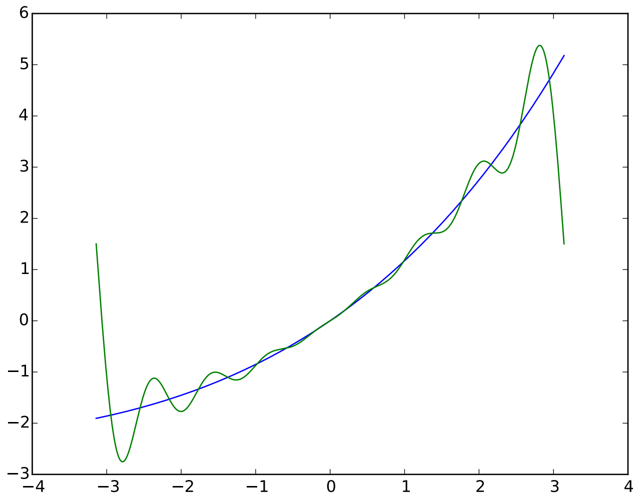
PROBLEM 8

This is a 16-midpoint interpolation of the function $f(t) = \left(\pi^2 - t^2\right) e^{t/2\pi}$, $t \in [-\pi, \pi]$



PROBLEM 9

This is a 16-midpoint interpolation of the function $f(t) = te^{t/2\pi}$, $t \in [-\pi, \pi]$



Appendix

```
# This is the code used for the second mandatory assignment in
# MAT1120 - Linear Algebra
# at the University of Oslo.

import numpy as np

def is_semi_orthogonal(matrix):
    """ Tests whether a matrix is semi-orthogonal
        based on the definition given in the
        assignment text

        Assumes a square matrix (n x n),
        raises an exception if not square.
    """
    if matrix.shape[0] != matrix.shape[1]:
        raise Exception('Matrix is not square')

    B = matrix.T.dot(matrix) # Forms the matrix to be tested
    eps = 1.0e-12
    B[np.abs(B) < eps] = 0 # Makes all entries below a certain threshold zero
    try:
        B = B.getA()
    except:
        pass

    for i in range(matrix.shape[0]):
        for j in range(matrix.shape[0]):
            if i == j:
                if  B[i][j] <= 0:
                    return False
            else:
                if abs(B[i][j]) > eps:
                    return False
    return True

def find_inverse(matrix):
    """ Finds and returns the inverse of a matrix with the properties
    given in the assignment text.

    Assumes a semi-orthagonal matrix,
    raises an exception if the matrix
    is not orthogonal.
    """
    if not is_semi_orthogonal(matrix):
        raise Exception("Matrix is not orthogonal")


    matrix_inv = np.zeros((matrix.shape[0], matrix.shape[0]))
    index = 0
    for i in matrix.T: # Iterating over columns instead of rows, hence the transpose
        matrix_inv[index] = transform_vector(i)
```

```
        index += 1

    eps = 1.0e-12
    matrix_inv[np.abs(matrix_inv) < eps] = 0
    return matrix_inv

def transform_vector(vec):
    """ Transforms the given vector u_j into
    u_j' as given in problem 1
    """
    try:
        vec = vec.getA()
    except:
        pass
    return (1./vec.dot(vec.T))*vec

def print_matrix(matrix):
    try:
        matrix = matrix.getA()
    except:
        pass
    for i in matrix:
        for j in i:
            print("%7.3f" % (j), end="")
        print()

def matrix_to_tex(matrix, filename):
    outfile = open(filename+".tex", 'w')
    outfile.write("\\begin{bmatrix}" + "\n")
    body = ""
    for i in matrix:
        for j in i:
            body += "%7.3f" % (j) + "&"
        body = body[:-1]
        body += " \\\\ " + "\n"
    outfile.write(body)
    outfile.write("\end{bmatrix}")

def calc_vector_y(function, n):
    """ Calculates the vector y_o/y_l using a function f_o/f_l, and n midpoints
        This is used for exercise 8 and 9
    """
    t = np.pi/16 + (np.linspace(1, 8, 8)-1)*np.pi/8
    return function(t)

if __name__ == "__main__":

    t1 = np.pi / 6; t2 = np.pi / 2; t3 = 5 * np.pi / 6
    A = np.matrix([[1, np.cos(t1), np.cos(2*t1)],[1,np.cos(t2), np.cos(2*t2)], [1, np.
    cos(t3), np.cos(2*t3)]])
    B = np.matrix([[1, 2, 3],[4, 5, 6], [7, 8, 9]])
    print_matrix(find_inverse(A))

    def test_function(t):
```

```
        return 2*t

    calc_yo(test_function, 8)
```

```python
# The code used for problem 3
# in the second mandatory assignment
# in MAT1120 at the University of Oslo

from assignment2 import *
import numpy as np
# Initializing matrices and t values

n = 8
C = np.zeros((n, n))
S = np.zeros((n, n))
t = np.pi/16 + (np.linspace(1, 8, 8)-1)*np.pi/8

for i in range(n):
    for j in range(n):
        C[i][j] = np.cos((j)*t[i])
        S[i][j] = np.sin((j+1)*t[i])

# Verifying that the matrices C and S are semi-orthogonal:

print("Is C semi-orthogonal? " , is_semi_orthogonal(C))
print("Is S semi-orthogonal? " , is_semi_orthogonal(S))

# Finding the inverse matrices of C and S:

print("Inverse of C: ")
print_matrix(find_inverse(C))
print("Inverse of S: ")
print_matrix(find_inverse(S))

matrix_to_tex(find_inverse(C), "C_matrix")
matrix_to_tex(find_inverse(S), "S_matrix")
```

```
# This is the code used for the second mandatory assignment in
# MAT1120 - Linear Algebra
# at the University of Oslo.

import numpy as np
import matplotlib.pylab as plt
from assignment import *

# Initializing and calculating matrices C and S
n = 8
C = np.zeros((n, n))
S = np.zeros((n, n))
t = np.pi/16 + (np.linspace(1, 8, 8)-1)*np.pi/8

for i in range(n):
    for j in range(n):
        C[i][j] = np.cos((j)*t[i])
        S[i][j] = np.sin((j+1)*t[i])

def f(t):
    # This is the only line that differs between Problem 8 and 9.
    return (np.pi**2 - t**2)*np.exp(t/(2*np.pi))

def f_l(t):
    return 0.5*(f(t) + f(-t))

def f_o(t):
    return 0.5*(f(t) - f(-t))

def f_midpoint(t, yc, ys):
    c = np.cos
    s = np.sin
    return    yc[0] + yc[1]*c(t) + yc[2]*c(2*t) + yc[3]*c(3*t) + yc[4]*c(4*t)\
           + yc[5]*c(5*t) + yc[6]*c(6*t) + yc[7]*c(7*t)\
           + ys[0]*s(t) + ys[1]*s(2*t) + ys[2]*s(3*t) + ys[3]*s(4*t) + ys[4]*s(5*t)\
           + ys[5]*s(6*t) + ys[6]*s(7*t) + ys[7]*s(8*t)







time_values = np.linspace(-np.pi, np.pi, 1000)
yl = calc_vector_y(f_l, n)
yo = calc_vector_y(f_o, n)
yc = find_inverse(C).dot(yl)
ys = find_inverse(S).dot(yo)

plt.plot(time_values, f(time_values), time_values, f_midpoint(time_values, yc, ys))
plt.savefig("Problem8.pdf")
```