Sheet 3

Create a directory named sheet03 and make it your working directory.

Example 0. To get warmed up, find the matrices A and Y for the data block

$$1, \quad 3, \quad 2.75, \quad 0.75, \quad 3.1, \quad 2.6, \quad 0.8, \quad 2.9, \quad 2.8$$

with $p = 3$. Then use Matlab to find $V = (a_1, a_2, a_3)'$.
Next, using the numbers $(x_1, x_2, x_3, a_1, a_2, a_3)$, perform (by hand) the reconstruction the receiver will do:

$$\begin{aligned}
x_{p+1} \approx \tilde{x}_{p+1} &= a_1 x_p + a_2 x_{p-1} + a_3 x_{p-2} + \cdots + a_p x_1 \\
x_{p+2} \approx \tilde{x}_{p+2} &= a_1 \tilde{x}_{p+1} + a_2 x_p + a_3 x_{p-1} + \cdots + a_p x_2 \\
x_{p+3} \approx \tilde{x}_{p+3} &= a_1 \tilde{x}_{p+2} + a_2 \tilde{x}_{p+1} + a_3 x_p + \cdots + a_p x_3 \\
x_{p+4} \approx \tilde{x}_{p+4} &= a_1 \tilde{x}_{p+3} + a_2 \tilde{x}_{p+2} + a_3 \tilde{x}_{p+1} + \cdots + a_p x_3 \\
&\vdots
\end{aligned}$$

(for any index m bigger than p, you must use the approximation $\tilde{x}_m$—after all, the receiver only knows the data up to the index p). Thus the data the receiver will end up with is

$$(x_1, x_2, \ldots x_p, \tilde{x}_{p+1}, \tilde{x}_{p+2}, \ldots, \tilde{x}_N).$$

The answer I calculated is $(1, 3, 2.75, .9131, 2.9493, 2.7052, .8280, 2.8981, 2.7532)$.

Example 1. a) Go to my homepage, click on Pre-REU and save the file data5.csv to your current directory. Then in Matlab load the data file as `data = load(···)`—you check from Sheet 2 what goes inside the `load` command. The file data5.csv is a large matrix whose first row is the set of indexes $1, \ldots, N$ and whose second row is a single data block $x_1, \ldots, x_N$. Extract the rows:

```
x = data(1,:);
y = data(2,:);
```

Recall that the `data(m,:)` command will extract the $m-$th row of the matrix.

Your first task is to try to guess the appropriate value of p. For this, plot the data block `y` as a function of the index and connect the dots:

```
plot(x,y,x,y,'o')
```

The first `x,y` connects the data points and the second one plots the data points with a $\circ$. You will see a semi-repetitive pattern of spikes. What is the natural choice for p? (Recall p is the number of data points in the semi-repetitive block).

b) On my homepage under the link Pre-REU save the m-files lpcA.m and lpcY.m to your current directory. These files will take the data block `y` and automatically form the matrices A and Y.

```
p = whatever you picked;
A = lpcA(p,y);
Y = lpcY(p,y);
```

Now you are ready to find the matrix V—do it.

c) I have written an algorithm to generate the reconstuction of the data from $(x_1, \ldots, x_p, a_1, \ldots, a_p)$ that the receiver must perform. It is called lpcreconstruct.m. To use it you need to input $(x_1, \ldots, x_p, a_1, \ldots, a_p)$ and the length N of the whole data block:

```
p = whatever you chose;
N = length(y);
sent = y(1:p);
f = lpcreconstruct(sent,V,N);
```

Plot the data the receiver reconstructs along with the original data:

```
plot(x,f,'r',x,y,'k','linewidth',2)
legend('Compressed','Signal')
```

Here the `'linewidth',2` statement makes the lines wider.

d) Compute the relative error `norm((f'-Y)')/norm(Y')`.

e) Repeat this for p one unit bigger and one unit smaller than the choice of p you made in part a). Which gives a better relative error? Is that a surprise?

e) Here is another way to help you choose p. Plot the relative error as a function of p. For this, go to my homepage and save the file rerrorp.m in your current directory. This will compute the relative error for you. Here is how to use it:

```
q = 1:floor(length(x)/2);
f = rerrorp(q,y);
plot(q,f)
```

The value for `q` will tell Matlab to restrict attention to those blocksizes p that are less than or equal to half of the total blocksize N. After all, you would be sending the information $(x_1, \ldots, x_p, a_1, \ldots, a_p)$ and if $p > N/2$ then you would be sending more than N data points! You'd might as well have sent the original block $(x_1 \ldots, x_N)$!

Looking at the plot should give you an idea of what to choose for p.

Also, it is useful to plot vertical lines to help you accurately pick the proper value of p. Look at the plot to obtain the range of the $y-$values; let's say they ranged from 0 to 8. After the `plot` statement, enter

```
line(6,linspace(0,8));
```

This will put a vertical line at $x = 6$. You can include more than one `line` statement.

Example 2. Repeat this for the data set data6.csv.

Example 3. There is a way to improve the quality of the compressed data to reduce the error in the compression. Rather than send $(x_1, \ldots, x_p, a_1, \ldots, a_p)$, if the error between x_{p+k} and its estimate is bigger than some tolerance, also send x_{p+k}.

For example, say the tolerance is .3, $p = 2$, the data block is 7 units long, with

$$AV = \begin{pmatrix} 1.2 \\ 1.5 \\ 4 \\ 5 \\ 7.5 \end{pmatrix} \quad \text{and} \quad Y = \begin{pmatrix} x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 4.1 \\ 4.6 \\ 7.6 \end{pmatrix}.$$

Recall we approximate Y by AV, so the error in approximating x_4 and x_6 is more than the tolerance and the errors in the remaining entries are smaller than the tolerance. Thus we would send $(x_1, x_2, a_1, a_2, x_4, x_6)$ rather than (x_1, x_2, a_1, a_2).

Let's do this for the data set data5.csv we used in Example 1. Download the files lpcpercent.m, lpccompress.m, lpcreconstruct.m and implpc.m to your current directory (it should be sheet03) from my homepage. They are programs to implement the procedure outlined above, to plot the compressed signal and the original signal on the same graph and to compute the approximate compression. Here is how to use it:

```
data = load(you figure out what goes here);
p = 6; tol = You choose this;
implpc(data,p,tol);
```

Play around with the tolerance `tol` and see what happens.

Example 4. Download the data set data7.csv from my homepage. This time the data set has more than one block, so you need to figure out how many blocks are present. Then handle each block separately and use the `rerrorp` procedure to find the correct value of p for each block. Plot the compressed signal and original signal on the same coordinate axes.