1 Problem 1

Derive the Richardson extrapolation theme for a series $S_N$ that converges to $L$ that uses $S_{2N}, S_{4N}$ and so forth. In other words, if you think of $S_N$ as a partial sum for an infinite series, each step uses twice as many steps as the one before.

1.1 Part a.
Derive a linear scheme for $S_N = L + \frac{A}{N} + ...$

1.2 Part b.
Derive a quadratic scheme for $S_N = L + \frac{B}{N^2} + ...$

1.3 Part c.
Derive a combined linear and quadratic scheme for $S_N = L + \frac{A}{N} + \frac{B}{N^2} + ...$

1.4 Part d.
Substitute Part a. into Part b. and show that you get the answer in Part c. In other words, performing first order (linear) scheme followed by a second order (quadratic) scheme is equivalent to doing both simultaneously.

2 Problem 2

2.1 Part a.
Program the fast series

$$\pi = \sqrt{16 \sum_{n=1}^{\infty} \frac{1}{(2n-1)^2 (2n+1)^2}} + 8$$

and use a log-log plot to determine the order of convergence of this series. You should get third order convergence. Turn in the plot and indicate by hand how you determined that the order is 3. You can use $S_{10}, S_{100}, S_{1000}, etc.$ or $S_2, S_4, S_8, etc.$ In the second case, you should probably compute more partial sums.
2.2 Part b.
Derive the third order Richardson extrapolation for your sequence of partial sums.

2.3 Part c.
Apply the scheme you derived in Part b. to the series you obtained in Part a. Demonstrate that the new series converges with order 4.

Turn in all code.

3 Problem 3
Write a semi-vectorized version of piSum:

```matlab
function s = piSumSemiVectorized(N)
    % The idea is to break up N into pieces of no more than, say, 1000000 terms.
    % Add up all the terms in a single piece with vectorized code and add up the pieces together with a loop. Make sure to add the terms and the pieces backwards.
    % Your code must work properly with N that is not a clean multiple of 1000000, for example, N = 6000123.
```

4 Problem 4
What answer does the last command produce and why? Use "help" for function you don’t know.

4.1 Part a.
```matlab
>> M = eye(500);
    >> sum(sum(M(2:2:end, 2:2:end)))
```

4.2 Part b.
```matlab
>> M = eye(500);
    >> sum(sum(M(2:2:end, 1:2:end)))
```

4.3 Part c.
```matlab
>> 10.^[ 1 2 3 4 ] - [ 10 100 1000 10000 ]
```