(a) Create pagerank1(G) by modifying pagerank so that it just computes the PageRanks, but does not do any plotting or printing.

Ans:

```
Unset
```

(b) Create pagerank2(G) by modifying pagerank1 to use inverse iteration instead of solving the sparse linear system. The key statements are

```
x = (I - A) e

x = x/sum(x)
```

What should be done in the unlikely event that the backslash operation involves a division by zero?

Ans:

```
Unset
```

```
x = (I - A + reg_param * I) \ e;
x = x / sum(x); % Normalize x to make it a probability
distribution
end
```

To avoid zeros in division, we can add a small regularization term to the numerator I-A, which in this case I have taken as an Identity matrix time small constant(1e-9).

(c) Create pagerank3(G) by modifying pagerank1 to use the power method instead of solving the sparse linear system. The key statements are

```
G = p*G*D
z = ((1-p)*(c~=0) + (c==0))/n;
while termination_test
    x = G*x + e*(z*x)
end
```

What is an appropriate test for terminating the power iteration?

(d) Use your functions to compute the PageRanks of the six-node example discussed in the text. Make sure you get the correct result from each of your three functions.

Ans:

Power Iteration can be stopped when the difference of two matrices generated in consecutive iterations is less than the tolerance level considered.

In this code sample, I have taken the 1 Norm (maximum column sum) of the difference matrix, checking if it's greater than or equal to 1e-6 for convergence.

(d) Use your functions to compute the PageRanks of the six-node example discussed in the text. Make sure you get the correct result from each of your three functions.

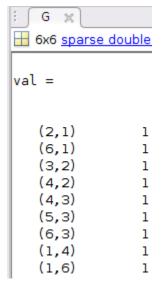
Ans:

Variables

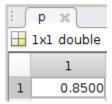
U:



G:



P:



Outputs:

pagerank1(U,G,p):

```
ans =

0.3210
0.1705
0.1066
0.1368
0.0643
0.2007
```

pagerank2(U,G,p):

```
ans =
     0.3210
     0.1705
     0.1066
     0.1368
     0.0643
     0.2007
pagerank3(U,G,p):
ans =
     0.3210
     0.1705
     0.1066
     0.1368
     0.0643
     0.2007
     (b) Use this function as a template to write a function that computes PageR-
 ank in some other programming language.
 [n,n] = size(G);
 for j = 1:n
    L{j} = find(G(:,j));
    c(j) = length(L{j});
 end
% Power method
p = .85;
delta = (1-p)/n;
x = ones(n,1)/n;
z = zeros(n,1);
cnt = 0;
while max(abs(x-z)) > .0001
    z = x;
    x = zeros(n,1);
    for j = 1:n
       if c(j) == 0
          x = x + z(j)/n;
          x(L{j}) = x(L{j}) + z(j)/c(j);
       end
    end
    x = p*x + delta;
    cnt = cnt+1;
Ans: PageRank in Python:-
```

```
from copy import deepcopy
def pagerankpow(G):
```

```
return x, cnt

G = [[0., 0., 0., 1., 0., 1.],
        [1., 0., 0., 0., 0.],
        [0., 1., 0., 0., 0.],
        [0., 1., 1., 0., 0., 0.],
        [0., 0., 1., 0., 0., 0.],
        [1., 0., 1., 0., 0., 0.]]

x, cnt = pagerankpow(G)

print("PageRank vector: ", x)
print("Number of iterations: ", cnt)
```