Test funzione pagerank.mlx

Casi d'uso della funzione *pagerank.mlx* .

mathwork200.mat

Calcolo del pagerank con la funzione pagerank.

Caso d'uso con la matrice presente nel file *mathwork200.mat*. Utilizziamo la funzione *pagerank.mlx* che restituisce i rank delle varie pagine e verifichiamo che la somma sia pari a 1 e che tutti gli elementi siano non negativi.

```
load mathwork200.mat;
R = pagerank(G)
```

```
R = 200×1
    0.0018
    0.0055
    0.0048
    0.0048
    0.0048
    0.0048
    0.0048
    0.0048
    0.0048
    0.0048
     ⋮
```

```
somma = sum(R)
```

```
somma = 1.0000
```

```
elementi_negativi = R(R<0)
```

```
elementi_negativi =

  0×1 empty double column vector
```

Calcolo accuratezza con il confronto con *centrality*.

```
gs = digraph(G','omitselfloops');
Pk = centrality(gs,'pagerank','FollowProbability',0.85, 'Tolerance',10^-7)
```

```
Pk = 200×1
    0.0018
    0.0055
    0.0048
    0.0048
    0.0048
    0.0048
    0.0048
```

```
    0.0048
    0.0048
    0.0048
     ⋮
```

```
err_rel = norm(R - Pk)/norm(Pk)
```

```
err_rel = 2.2551e-06
```

repubblica.mat

Calcolo del pagerank con la funzione pagerank.

Caso d'uso con la matrice presente nel file *repubblica.mat*. Utilizziamo la funzione *pagerank.mlx* che restituisce i rank delle varie pagine e verifichiamo che la somma sia pari a 1 e che tutti gli elementi siano non negativi.

```
load repubblica.mat;
R = pagerank(G)
```

```
R = 100×1
    0.0037
    0.0103
    0.0112
    0.0137
    0.0113
    0.0118
    0.0226
    0.0118
    0.0118
    0.0932
     ⋮
```

```
somma = sum(R)
```

```
somma = 1.0000
```

```
elementi_negativi = R(R<0)
```

```
elementi_negativi =

  0×1 empty double column vector
```

Calcolo accuratezza con il confronto con *centrality*.

```
gs = digraph(G','omitselfloops');
Pk = centrality(gs,'pagerank','FollowProbability',0.85, 'Tolerance',10^-7)
```

```
Pk = 100×1
    0.0037
    0.0103
    0.0112
    0.0137
```

```
    0.0113
    0.0118
    0.0226
    0.0118
    0.0118
    0.0932
     ⋮
```

```
err_rel = norm(R - Pk)/norm(Pk)
```

```
err_rel = 1.3285e-07
```

ilsole24.mat - Generato attraverso surfer.m

Calcolo del pagerank con la funzione pagerank.

Caso d'uso con la matrice presente nel file *ilsole24.mat*. Utilizziamo la funzione *pagerank.mlx* che restituisce i rank delle varie pagine e verifichiamo che la somma sia pari a 1 e che tutti gli elementi siano non negativi.

```
load ilsole24.mat;
R = pagerank(G)
```

```
R = 50×1
    0.0175
    0.0550
    0.0383
    0.0216
    0.0178
    0.0178
    0.0178
    0.0178
    0.0178
    0.0178
     ⋮
```

```
somma = sum(R)
```

```
somma = 1.0000
```

```
elementi_negativi = R(R<0)
```

```
elementi_negativi =

  0×1 empty double column vector
```

Calcolo accuratezza con il confronto con *centrality*.

```
gs = digraph(G','omitselfloops');
Pk = centrality(gs,'pagerank','FollowProbability',0.85, 'Tolerance',10^-7)
```

```
Pk = 50×1
    0.0175
    0.0550
```

```
    0.0383
    0.0216
    0.0178
    0.0178
    0.0178
    0.0178
    0.0178
    0.0178
     ⋮
```

```
err_rel = norm(R - Pk)/norm(Pk)
```

```
err_rel = 3.6514e-07
```

galeazzi.mat - Generato attraverso surfer.m

Calcolo del pagerank con la funzione pagerank.

Caso d'uso con la matrice presente nel file *galeazzi.mat*. Utilizziamo la funzione *pagerank.mlx* che restituisce i rank delle varie pagine e verifichiamo che la somma sia pari a 1 e che tutti gli elementi siano non negativi.

```
load galeazzi.mat;
R = pagerank(G)
```

```
R = 100×1
    0.0058
    0.0145
    0.0231
    0.0064
    0.0064
    0.0064
    0.0064
    0.0064
    0.0064
    0.0096
     ⋮
```

```
somma = sum(R)
```

```
somma = 1.0000
```

```
elementi_negativi = R(R<0)
```

```
elementi_negativi =

  0×1 empty double column vector
```

Calcolo accuratezza con il confronto con *centrality*.

```
gs = digraph(G','omitselfloops');
Pk = centrality(gs,'pagerank','FollowProbability',0.85, 'Tolerance',10^-7)
```

```
Pk = 100×1
```

```
    0.0058
    0.0145
    0.0231
    0.0064
    0.0064
    0.0064
    0.0064
    0.0064
    0.0064
    0.0096
     ⋮
```

```
err_rel = norm(R - Pk)/norm(Pk)
```

```
err_rel = 6.2836e-07
```

Casi di errore della funzione *pagerank.mlx* .

Caso in cui la matrice in ingresso è vuota:

```
G = [];
R = pagerank(G);
```

```
Error using pagerank (line 8)
La matrice di input è vuota.
```

Caso in cui la matrice in ingresso non è sparsa:

```
G = [1 2 4; 1 5 6; 4 5 6];
R = pagerank(G);
```

```
Error using pagerank (line 12)
La matrice di input non è sparsa.
```

Caso in cui la matrice non è di tipo sparse logic:

```
G = sparse([1 0 0; 0 1 0; 0 0 1]);
R = pagerank(G);
```

```
Error using pagerank (line 16)
La matrice di input non contiene solo elementi logici.
```

Caso in cui la matrice in ingresso non è quadrata:

```
G = sparse(logical([1 0 0; 0 1 0]));
R = pagerank(G);
```

```
Error using pagerank (line 22)
```

```
La matrice di input non è quadrata.
```

Caso in cui l'ingresso è un elemento logico:

```
G = sparse(logical(1));
R = pagerank(G)
```

```
Error using pagerank (line 26)
Dimensioni della matrice di input non corrette.
```