1.1. Problema. Si implementi una function per il metodo dell'eliminazione gaussiana (cf. [9]) con pivoting parziale (per righe) per il calcolo del determinante di una matrice. La function deve restituire in output il determinante e la matrice triangolare superiore che si ottiene come risultato del metodo; deve saper gestire anche il caso del pivoting nullo (matrice singolare).

1.1.1. Risoluzione. Per generare una matrice di numeri casuali di ordine n si usa il comando

`A=rand(n);`

Salviamo la function richiesta in un file `meg.m`

```
function [U, deter]=meg(A)
n=size(A,1);
U=A;       % INIZIALIZZAZIONI.
deter=1;
for k=1:1:n-1
    [piv, j]=max(abs(A(k:n,k))); % PIVOTING.

    if (piv == 0)    % CASO DETERMINANTE 0.
       deter=0;
       return
    end

    if (j ~=1)       % SCAMBIO RIGHE.
       temp=A(j+k-1,:);
       A(j+k-1,:)=A(k,:);
       A(k,:)=temp;
       deter=-deter;
    end

    % deter=deter*A(k,k);

    % ELIMINAZIONE GAUSSIANA.

    for index=k+1:1:n
        m(index,k)=A(index,k)/A(k,k);
        A(index,k)=0;
        for j=k+1:1:n
            A(index,j)=A(index,j)-m(index,k)*A(k,j);
        end
    end

end

U=A;
det_U=prod(diag(U)); % CALCOLO DETERMINANTE
                           % MATRICE FINALE "U".
deter=deter*det_U;
```

Qualche commento sul codice:

1. Nel ciclo for che comincia con l'istruzione `for k=1:1:n-1`, il codice calcola l'elemento $a_{j,k}$ con $j \geq k$ che sia più grande in modulo. Facciamo un esempio sull'uso della funzione `max` in Matlab:

```
>> u=[100; 20; 100; 30]
u =
   100
    20
   100
    30
>> [s,t]=max(u)
s =
   100
t =
     1
>>
```

La variabile s descrive il massimo valore tra le componenti del vettore u, mentre t dice in quale indice del vettore viene assunto. Osserviamo che u ha il suo massimo nella prima e nella terza componente di u, ma che di default, in Matlab/Octave viene scelto quale indice t il più piccolo intero positivo per cui tale massimo viene assunto (nell'esempio $t = \min(1,3) = 1$).
Questa considerazione sulla funzione `max` di Matlab/Octave ha dei riflessi sull'algoritmo `meg.m`. Qualora il massimo di $|a_{j,k}|$ con $j \geq k$ sia assunto in più indici, tra questi viene scelto il minore.

2. Nella porzione di codice

```
if (piv == 0)
    deter=0;
    return
end
```

si stabilisce che se il massimo $|a_{j,k}|$ con $j \geq k$ è uguale a 0, allora il determinante di A è 0. Il comando `return` blocca immediatamente la funzione e assegna ad `U` e `deter` i valori fino allora assegnati. Per avere un'idea perchè ciò succeda facciamo un esempio:

```
>> A=[3 4 5 6 7; 0 8 9 1 2; 0 0 0 1 6; 0 0 0 4 9; 0 0 0 5 2]
A =
     3     4     5     6     7
     0     8     9     1     2
     0     0     0     1     6
     0     0     0     4     9
     0     0     0     5     2
>> det(A)
ans =
     0
>>
```

Le prime 3 colonne generano un sottospazio di $\mathbb{R}^5$ di dimensione 2. Quindi i vettori $(3,0,0,0,0)$, $(4,8,0,0,0)$, $(5,9,0,0,0)$ sono linearmente *dipendenti* e conseguentemente il determinante della matrice A è nullo. Questa idea si estende al caso generale. Se tutte le componenti $a_{j,k}$ con $j \geq k$ sono nulle, le prime k colonne generano un sottospazio di dimensione $k-1$ e quindi sono linearmente dipendenti. Di conseguenza il determinante di a è 0.

3. Nel blocco

```
if (j ~=1)
    temp=A(j+k-1,:);
    A(j+k-1,:)=A(k,:);
    A(k,:)=temp;
    deter=-deter;
end
```

si nota che $a_{j+k-1,k} \geq a_{s,k}$ per $s = k, \ldots, n$ e quindi si scambiano la riga $j+k-1$-sima con la k-sima, tenendo in mente che lo scambio di righe produce una matrice A' il cui determinante ha valore assoluto uguale a quello di A ma segno opposto. Vediamone un esempio:

```
>> A=[1 2; 3 4]
A =
     1     2
     3     4
>> B=A([2 1],[1 2])
B =
     3     4
     1     2
>> det(A)
ans =
    -2
>> det(B)
ans =
     2
>>
```

4. La porzione di codice

```
U=A;
det_U=prod(diag(U)); % CALCOLO DETERMINANTE
                           % MATRICE FINALE "U".
deter=deter*det_U;
```

è più complicata di quello che si creda. Vediamo su un esempio cosa succede di A, al variare di k.

```
>> A=[1 4 2 6; 3 2 5 7; 1 3 8 6; 1 3 5 6]
A =
     1     4     2     6
     3     2     5     7
     1     3     8     6
     1     3     5     6
>> [U, deter]=meg(A)
k =
     1
A =
    3.0000    2.0000    5.0000    7.0000
         0    3.3333    0.3333    3.6667
         0    2.3333    6.3333    3.6667
```

```
         0    2.3333    3.3333    3.6667
k =
     2
A =
    3.0000    2.0000    5.0000    7.0000
         0    3.3333    0.3333    3.6667
         0         0    6.1000    1.1000
         0         0    3.1000    1.1000
k =
     3
A =
    3.0000    2.0000    5.0000    7.0000
         0    3.3333    0.3333    3.6667
         0         0    6.1000    1.1000
         0         0         0    0.5410
U =
    3.0000    2.0000    5.0000    7.0000
         0    3.3333    0.3333    3.6667
         0         0    6.1000    1.1000
         0         0         0    0.5410
deter =
  -33.0000
>>
```

La matrice U ha un determinante uguale a quello di A a meno del segno. Visto che la differenza di segno tra A ed U è tenuta *sotto controllo* nella parte relativa al pivoting (controllare la porzione di codice in cui si scambiano le righe!), non resta che calcolare il determinante di U.
Ricordando che

- `diag` applicato a una matrice $A = (a_{i,j})$ fornisce un vettore $u = (u_k)$ tale che $u_k = a_{k,k}$;
- `prod` applicato ad un vettore u esegue $\prod_k u_k$;
- il determinante di una matrice triangolare superiore coincide per la regola di Laplace al prodotto degli elementi diagonali

si ha che in effetti il determinante di U è dato da `prod(diag(U))`.
Alternativamente si poteva togliere il blocco sopramenzionato

```
U=A;
det_U=prod(diag(U)); % CALCOLO DETERMINANTE
                           % MATRICE FINALE "U".
deter=deter*det_U;
```

e inserire

```
deter=deter*A(k,k);
```

tra lo scambio di righe e l'eliminazione gaussiana. Vediamone il motivo. Nell'esempio fatto poco fa, per $k = 1$ si esegue il pivoting per colonne, mentre negli altri casi la strategia non comporta scambi di righe (perchè ?). A partire dalla matrice iniziale A, si determinano delle matrici $A^{(k)}$, il cui determinante coincide a meno del segno con quello di A. Notiamo che fissato $\bar{k}$, la matrice $A^{(\bar{k},\bar{k})} = (A_{i,j}^{(\bar{k})})$ con $i = 1, \ldots, \bar{k}$, $j = 1, \ldots, \bar{k}$ non viene più modificata nei *passi successivi* in cui $k > \bar{k}$ (perchè ?) ed è triangolare superiore. Quindi il determinante di $A^{(\bar{k},\bar{k})}$ è uguale a quello di $A^{(\bar{k}-1,\bar{k}-1)}$ moltiplicato per $A_{\bar{k},\bar{k}}^{(\bar{k})} = A_{\bar{k},\bar{k}}^{(\bar{k},\bar{k})}$. Alla fine del processo, la matrice $A^* = A^{(n)}$ ottenuta dalle varie traformazioni, è triangolare superiore ed ha quali elementi diagonali proprio $A_{k,k}^{(k)}$ con $k = 1, \ldots, n$. Quindi il suo determinante è

$$\prod_{k=1}^{n} A_{k,k}^{(k)} = \prod_{k=1}^{n} A_{k,k}^{(k,k)}$$

che a meno del segno coincide con il determinante della matrice A.

5. si osservi che in questo codice si è preferito cambiare segno alla variabile `deter` ad ogni scambio di righe piuttosto che definire una variabile `mu` e di volta in volta porla uguale a -1 o $+1$ a seconda venga effettuato o meno un cambio di riga; se si fosse fatta questa scelta alla fine avremmo dovuto scrivere

```
function [U, deter]=meg(A)
n=size(A,1);
U=A;       % INIZIALIZZAZIONI.
for k=1:1:n-1
    mu(k)=1;  % INIZIALIZZAZIONE mu.
    [piv, j]=max(abs(A(k:n,k))); % PIVOTING.

    if (piv == 0)    % CASO DETERMINANTE 0.
       deter=0;
       return
    end

    if (j ~=1)       % SCAMBIO RIGHE.
       temp=A(j+k-1,:);
       A(j+k-1,:)=A(k,:);
       A(k,:)=temp;
       mu(k)=-1;
    end

    % deter=deter*A(k,k);

    % ELIMINAZIONE GAUSSIANA.

    for index=k+1:1:n
        m(index,k)=A(index,k)/A(k,k);
        A(index,k)=0;
        for j=k+1:1:n
            A(index,j)=A(index,j)-m(index,k)*A(k,j);
        end
    end

end

U=A;
det_U=prod(diag(U)); % CALCOLO DETERMINANTE
```