Eliminazione di Gauss

Si consideri un isomorfismo tra spazi vettoriali di dimensione n

$$A : \mathcal{U} \to \mathcal{V}$$

Si vuole costruire un algoritmo che realizzi l'isomorfismo inverso, cioè un algoritmo che trasformi un vettore $b \in \mathcal{V}$ nel vettore $x := A^{-1}b$.

Siano

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix} \qquad \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} \qquad \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

rispettivamente la matrice di A e le $n-ple$ delle componenti di b e x corrispondenti ad una coppia di basi di $\mathcal{U}$ e $\mathcal{V}$.

Si può definire l'algoritmo suddetto nel seguente modo.

Si sostituiscano le righe di $\mathbf{A}$ dalla seconda in poi, e i corrispondenti elementi di $\mathbf{b}$, con le combinazioni lineari descritte dalle seguenti espressioni (omettendo quelle per i termini della prima colonna al disotto di a_{11})

$$c := a_{21}/a_{11}$$
$$a^{(1)}_{22} := a_{22} - ca_{12} \qquad a^{(1)}_{23} := a_{23} - ca_{13} \quad \dots \quad a^{(1)}_{2n} := a_{2n} - ca_{1n} \qquad b^{(1)}_{2} := b_2 - cb_1$$

$$c := a_{31}/a_{11}$$
$$a^{(1)}_{32} := a_{32} - ca_{12} \qquad a^{(1)}_{33} := a_{33} - ca_{13} \quad \dots \quad a^{(1)}_{3n} := a_{3n} - ca_{1n} \qquad b^{(1)}_{3} := b_3 - cb_1$$

$$\dots$$

$$c := a_{n1}/a_{11}$$
$$a^{(1)}_{n2} := a_{n2} - ca_{12} \qquad a^{(1)}_{n3} := a_{n3} - ca_{13} \quad \dots \quad a^{(1)}_{nn} := a_{nn} - ca_{1n} \qquad b^{(1)}_{n} := b_n - cb_1$$

Una descrizione formale di tali operazioni è data dalla seguente frase *Pascal*

```
for i:=2 to n do
begin
  c := a[i,1]/a[1,1];
  for k:=2 to n do  a[i,k] := a[i,k] - c * a[1,k];
  b[i] := b[i] - c * b[1];
end;
```

Si applichi poi tale algoritmo alla parte di $\mathbf{A}$ ottenuta omettendo la prima riga e la prima colonna, e così via. La descrizione precedente si trasforma nella seguente

```
for j:=1 to n-1 do
begin
   for i:=j+1 to n do
   begin
     c := a[i,j]/a[j,j];
     for k:=j+1 to n do  a[i,k] := a[i,k] - c * a[j,k];
     b[i] := b[i] - c * b[j];
   end;
end;
```

Se risulta sempre definita la espressione di c (ovvero se è sempre $a_{jj} \neq 0$), questo algoritmo produce una nuova matrice $\mathbf{U}$, *triangolare superiore*, i cui elementi sono nell'array `a`, e un nuovo vettore $\mathbf{y}$, i cui elementi sono nell'array `b`, tali che

$$\mathbf{U}\mathbf{x} = \mathbf{y}$$

La valutazione del vettore $\mathbf{x}$ può infine essere fatta per *sostituzione all'indietro* secondo la seguente descrizione

```
for i:=n downto 1 do
begin
     s:=0; for k:=i+1 to n do  s := s + a[i,k] * b[k];
     b[i] := (b[i] - s)/a[i,i];
end;
```

L'algoritmo così costruito può essere modificato in modo da essere applicabile anche al caso in cui accada che $a_{jj} = 0$. Se esiste un elemento $a_{ij} \neq 0$, $i > j$, allora è sufficiente scambiare la riga j con una delle successive, altrimenti la matrice $\mathbf{A}$ risulta singolare. In generale è però opportuno[1]prevedere di scambiare delle righe in modo da ottenere di volta in volta un coefficiente c il cui valore assoluto sia il più piccolo possibile. Questo si può realizzare in generale scambiando la riga j con la riga m tale che

$$|a_{mj}| > |a_{jj}| \quad j < m \leq n$$

risultando in tal modo sempre $|c| \leq 1$. L'algoritmo risulta così modificato

```
for j:=1 to n-1 do
begin
   m:=j; for i:=j+1 to n do  if (abs(a[i,j]) > abs(a[m,j])) then m:=i;
   if m > j then for k:=j to n do
   begin
      s:=a[j,k]; a[j,k]:=a[m,k]; a[m,k]:=s;
   end;
```

[1] Si vedrà più avanti il perché.

```
    s:=b[j]; b[j]:=b[m]; b[m]:=s;
    for i:=j+1 to n do
    begin
      c := a[i,j]/a[j,j];
      for k:=j+1 to n do  a[i,k] := a[i,k] - c * a[j,k];
      b[i] := b[i] - c * b[j];
    end;
end;
```

Una diversa strategia[2]consiste nel *pesare* i termini a_{ij}, che si confrontano per decidere quali righe scambiare, con il più grande valore assoluto dei temini della riga a cui appartiene. Questo criterio di confronto (detto *equilibratura*) è realizzato nel seguente algoritmo

```
for i:=1 to n do
begin
  s:=abs(a[i,1]);
  for k:=2 to n do if (abs(a[i,k]) > s) then s:=abs(a[i,k]);
  w[i]:=s;
end;
for j:=1 to n-1 do
begin
   m:=j;
   for i:=j+1 to n do if(abs(a[i,j])/w[i] > abs(a[m,j])/w[m]) then m:=i;
   if m > j then for k:=j to n do
   begin
      s:=a[j,k]; a[j,k]:=a[m,k]; a[m,k]:=s;
   end;
   s:=b[j]; b[j]:=b[m]; b[m]:=s;
   for i:=j+1 to n do
   begin
     c := a[i,j]/a[j,j];
     for k:=j+1 to n do  a[i,k] := a[i,k] - c * a[j,k];
     b[i] := b[i] - c * b[j];
   end;
end;
```

[2] Se ne darà più avanti una motivazione.