Resolución de Sistemas de Ecuaciones por el Metodo de LU y PLU

UNIVERSIDAD NACIONAL DE SAN AGUSTÍN
Escuela Profesional de Ciencia de la Computación

Lehi Quincho Mamani
Mayo 2018

Crear una función "DescompLU(A)" que halle la descomposición LU de una matriz A, y escribe "No admite descomposicón LU" en caso de que no tenga descomposición en LU.

Función que halla la descomposicón en LU

```
    //HALLANDO LA DESPOMPOSICÓN EN LU
    //Creamos las matrices L y U
    double ** L=crear(n,n);
    double ** U=crear(n,m);
    copiar(A,U,n,m);
    escalona(U,L,n,m);
    //Sumar la matriz L con la identidad
    double **I=crear(n,n);
    identidad(I,n,n);
    L=sumaMat(L,I,n,n);
```

Función Escalona

```
void escalona(double **&U,double **&L,int n,int m)
{
    for(int i=0;i<m;i++)
    {
        for(int j=i+1;j<n;j++)
        {
            double num=U[j][i];
            double denom=U[i][i];

            L[j][i]=num/denom;
            if(L[j][i] == 0)
                cout<<"[ERROR] No Admite descomposicón en LU. "<<endl;
            for(int k=i;k<m;k++)
                U[j][k]=U[j][k]-((U[i][k]/denom)*num );
        }
    }
}
```

Crear una función "ResolverSistConLU(A,b)" que resuelve el sistema Ax=b usando descomposición en LU

```
    //Sustitución Progresiva
    // Ly=Pb
    double **Ly=contact(L,n,n,b);
    double *y=crear(n);
    sustProg(Ly,n,m+1,y);

    //Sustitución Regresiva
    // Ux =y
    double **Ux=contact(U,n,m,y);
    double *x=crear(n);

    sustRegre(Ux,n,m+1,x);

    return x;
}
```

```
void sustRegre(double **&A, int n, int m, double *&res)
{
    double suma=0;
    res[n-1]=A[n-1][m-1]/A[n-1][n-1];
    for(int i=n-2;i>=0;i--)
    {
        suma=0;
        for(int j=i+1;j<n;j++)
            suma=suma+A[i][j]*res[j];
        res[i]=(A[i][m-1]-suma)/A[i][i];
    }
}

void sustProg(double **&A, int n, int m, double *&res)
{
    double suma=0;
    res[0]=A[0][m-1]/A[0][0];

    for(int i=1;i<n;i++)
    {
        suma=0;
        for(int j=0;j<i;j++)
            suma=suma+A[i][j]*res[j];
        res[i]=(A[i][m-1]-suma)/A[i][i];
    }
}
```

Crear una función “DescompPLU(A)” que halla la descomposicón PLU de una matriz A, aplicando la técnica del pivoteo Parcial.

Despomposición PLU(A)

```
{
    double ** P=crear(n,n);
    double ** L=crear(n,n);
    double ** U=crear(n,m);

    copiar(A,U,n,m);
    identidad(P,n,m);

    //Escalonamiento con Pivoteo
    escalonaPiv(U,P,L,n,m);

    //Sumar la matriz L con una identidad
    double **I=crear(n,n);
    identidad(I,n,n);
    L=sumaMat(L,I,n,n);

    //Verificación
        //double **PA=productoMat(P,n,n,A,n,m);
        //double **LU=productoMat(L,n,n,U,n,m);
        //imprimir(PA,n,m);
        //imprimir(LU,n,m);
```

Escalonado con Pivoteo parcial .

```
void escalonaPiv(double **&U,double **&P, double **&L,int n,int m)
{
    for(int i=0;i<m;i++)
    {
        int indexMax;
        maxIndex(U,n,m,i,i,indexMax);

        pivoteo(U,n,m,indexMax,0,i);
        pivoteo(P,n,n,indexMax,0,i);
        pivoteo(L,n,n,indexMax,0,i);

        for(int j=i+1;j<n;j++)
        {
            double num=U[j][i];
            double denom=U[i][i];

            L[j][i]=num/denom;

            for(int k=i;k<m;k++)
                U[j][k]=U[j][k]-((U[i][k]/denom)*num );
        }

    }
}
```

Crear una función “ResuelveSistConPLU(A,b)” que resuelve el sistema Ax=b usando despomposición en PLU con pivoteo parcial

Resolviendo el Sistema Ax=b usando PLU con pivoteo (Que se encuentra como respuesta del enunciado anterior).

```
    //Sustitución Progresiva
    // Ly=Pb
    double *Pb=productoMatVec(P,n,n,b,n);
    double **Ly=contact(L,n,n,Pb);
    double *y=crear(n);
    sustProg(Ly,n,m+1,y);

    //Sustitución Regresiva
    //  Ux=y
    double **Ux=contact(U,n,m,y);
    double *x=crear(n);
    sustRegre(Ux,n,n+1,x);
    return x;
}
```

Sustitución Progresiva y Regresiva

```
void sustRegre(double **&A, int n, int m, double *&res)
{
    double suma=0;
    res[n-1]=A[n-1][m-1]/A[n-1][n-1];
    for(int i=n-2;i>=0;i--)
    {
        suma=0;
        for(int j=i+1;j<n;j++)
            suma=suma+A[i][j]*res[j];
        res[i]=(A[i][m-1]-suma)/A[i][i];
    }
}

void sustProg(double **&A, int n, int m, double *&res)
{
    double suma=0;
    res[0]=A[0][m-1]/A[0][0];

    for(int i=1;i<n;i++)
    {
        suma=0;
        for(int j=0;j<i;j++)
            suma=suma+A[i][j]*res[j];
        res[i]=(A[i][m-1]-suma)/A[i][i];
    }

}
```