

Satélites Artificiales y Geomática

PRÁCTICA 1

Movimiento del Polo

ADRIÁN FERNÁNDEZ TEJADA

ADRIÁN SÁNCHEZ LOUREIRO

2023

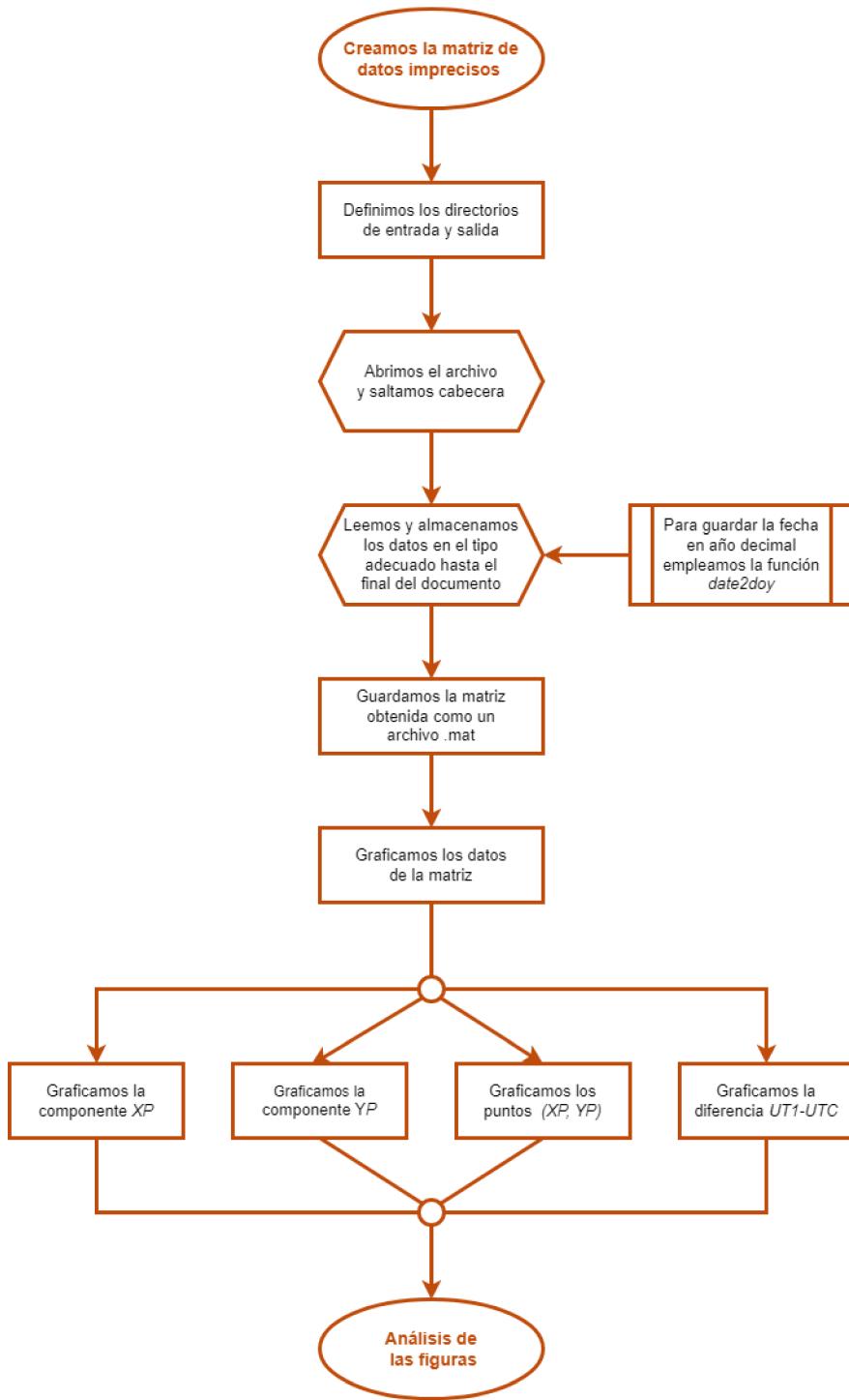
Tratamiento de Datos Imprecisos

Conceptos

- Movimiento del Polo: A fines del siglo pasado se detectaron variaciones en la latitud del Observatorio de Berlín. Esto condujo a la realización de una campaña de observación de la que participaron varios observatorios de la época, que rápidamente pusieron de manifiesto que las variaciones de latitud y longitud observadas se debían al cambio de posición del eje de rotación dentro de la Tierra misma ("movimiento del polo"). Este movimiento del origen del sistema de referencia terrestre (polo de rotación) cambia de posición los paralelos y meridianos respecto de la superficie terrestre, en consecuencia cambian las coordenadas "latitud y longitud". Este problema del movimiento del polo presenta un interés doble, encontrar los mecanismos geofísicos y astronómicos que lo generan, y encontrar la manera de corregir las determinaciones de latitud y longitud para contar con un sistema de referencia terrestre que está fijo a la Tierra.
- Escala de tiempo UT1: El Tiempo Universal UT1, es el tiempo del reloj que define la Tierra, que realiza un giro cada 24 horas. Presenta inestabilidades de corto periodo al nivel de 10^{-8} , disminuyendo lentamente la duración del día (casi 0.002 s/siglo). UT1 es uno de los productos del IERS (Servicio Internacional de Rotación de la Tierra y Sistemas de Referencia), basado en observaciones VLBI (interferometría de muy larga línea de base), que consiste en la observación de uno o varios objetos celestes con la ayuda de un gran número de radiotelescopios ubicados en distintas partes de la Tierra.
- Escala de tiempo UTC: UTC es una escala de tiempo de gran exactitud y estabilidad, basada en unos 450 relojes atómicos distribuidos por unos setenta y cinco laboratorios en todo el mundo. UTC utiliza como unidad de tiempo el segundo del Sistema Internacional (SI) de unidades.

```
pathe_diag = '/home/desk-chaguarma/Documents/UNI/SAG/PRACTICAS/PR1/
DIAGRAMAS/';

[im, map, alpha] = imread([pathe_diag 'PR1-B1.png']);
figure;
f = imshow(im);
set(f, 'AlphaData', alpha);
```



Tablas de datos

En este proyecto, emplearemos un tipo de variable que almacena los datos como una tabla.

Definamos la tabla `tablaI` que almacenará los datos:

fecha, X-polo, Y-polo, UT1-UTC, RMS-xp, RMS-yp, RMS-dt.

Tomando los datos del fichero BULLET.ERP, que almacena datos desde 2000-01-02 a 2018-09-22.

Primero, definimos el tipo de dato que almacenará cada columna y su nombre.

```
varTipos = [ "double", "double", "double", "double", "double", "double", "double" ];
varNames = [ "Fecha", "X", "Y", "UT", "RMS-X", "RMS-Y", "RMS-T" ];
```

Segundo, una variable auxiliar en la que almacenaremos el número de columnas.

```
sz = [1 length(varNames)];
```

Finalmente, definimos la tabla con lo citado anteriormente.

```
tablaI = table('Size',sz,'VariableTypes',varTipos,'VariableNames',varNames);
```

A continuación, usamos la variable auxiliar *fila* para almacenar el nº de filas que tiene la tabla.

```
fila = 0;
```

Código Realizado

Definimos en primer lugar la ruta donde se encuentra nuestro archivo de datos en la variable *directorio_raiz*. También una variable, *directorio_datos*. Finalmente, la ruta completa se almacena en *pathe_entrada*.

```
directorio_raizs = '/home/desk-chaguarma/Documents/UNI/SAG/PRACTICAS/PRI/';
directorio_dats = 'IMPRECISOS/';
directorio_sal= 'SALIDA/';
directorio_diag= 'DIAGRAMAS/';

pathe_entrada=[directorio_raizs directorio_dats];
pathe_diag=[directorio_raizs directorio_diag];
```

El nombre del documento a trabajar lo almacenamos en *doc*.

```
doc = "BULLET_A.ERP";
```

Abrimos el documento deseado con la función *fopen* habiendo concatenado la ruta y el nombre del archivo.

```
fidin = fopen(strcat(pathe_entrada,doc), 'r')
```

```
fidin = 5
```

Definimos una variable para quitar las primeras 6 líneas de documentación del archivo. Saltamos el nº de líneas deseado con un bucle.

```
elim = 6;
```

```

for i = 1:elim
    fgetl(fidin);
end

```

A partir del patrón del archivo, obtenemos datos de cada fila, con un bucle `while` con la condición que lea el archivo hasta que llegue al final de él.

```

while ~feof(fidin)

```

Leemos la línea con la función `fgetl` y tomamos los tres primeros caracteres para controlar si hemos llegado al final del archivo. A partir de una función `if` controlamos si ha llegado al final.

```

registro = fgetl(fidin);
control_fin=registro(1:3);
if control_fin=='EOF' %Puedes introducir este texto al final del doc como
parada
    break
else

```

Una vez comprobado que no es el final del documento, con la función `split`, dividimos la fila de datos que están separados por espacios y los almacenamos como un vector de caracteres en `registro`. Además, al añadir una nueva fila, aumentamos el contador `fila`,

```

fila = fila + 1;
registro = split(registro);
registro = registro (1:13);

```

Para almacenar la fecha, primero obtenemos la cadena de la fecha (yyyy-MM-dd) siguiendo la disposición de los datos en el archivo. Tras ello, a partir de la función `date2doy` obtenida de [aquí](#), transformamos la fecha a año con decimal.

```

fecha = strcat(registro(1), "-", registro(2), "-", registro(3));

[doy,frac] = date2doy(datenum(fecha));

frac = extractAfter(string(frac),1);
fecha = strcat(registro(1),frac);

```

Finalmente, almacenamos los datos deseados en la matriz definida anteriormente. Almacenamos los datos en formato `double`.

```

tablaI(fila,:) = {double(string(fecha)), double(string(registro(6))),
double(string(registro(7))), double(string(registro(8))),
double(string(registro(11))), double(string(registro(12))),
double(string(registro(13)))};
end
end

fclose(fidin);

```

Aquí tenemos el numero de datos almacenados y la tabla obtenida.

```
fila
```

```
fila = 6839
```

```
tablaI(1:8,:)
```

```
ans = 8×7 table
```

	Fecha	X	Y	UT	RMS-X	RMS-Y	RMS-T
1	2.0000e+03	0.0435	0.3776	0.3546	6.0000e-05	7.0000e-05	1.2000e-05
2	2.0000e+03	0.0436	0.3774	0.3539	7.0000e-05	8.0000e-05	1.1000e-05
3	2.0000e+03	0.0435	0.3771	0.3532	7.0000e-05	9.0000e-05	1.2000e-05
4	2.0000e+03	0.0432	0.3768	0.3528	8.0000e-05	9.0000e-05	1.8000e-05
5	2.0000e+03	0.0432	0.3765	0.3523	9.0000e-05	9.0000e-05	1.8000e-05
6	2.0000e+03	0.0435	0.3763	0.3520	8.0000e-05	9.0000e-05	1.8000e-05
7	2.0000e+03	0.0437	0.3761	0.3515	1.1000e-04	9.0000e-05	1.6000e-05
8	2.0000e+03	0.0438	0.3759	0.3511	1.2000e-04	1.0000e-04	1.8000e-05

Guardamos la tabla tablaI de los datos imprecisos para compararla en otro archivo con los datos precisos.

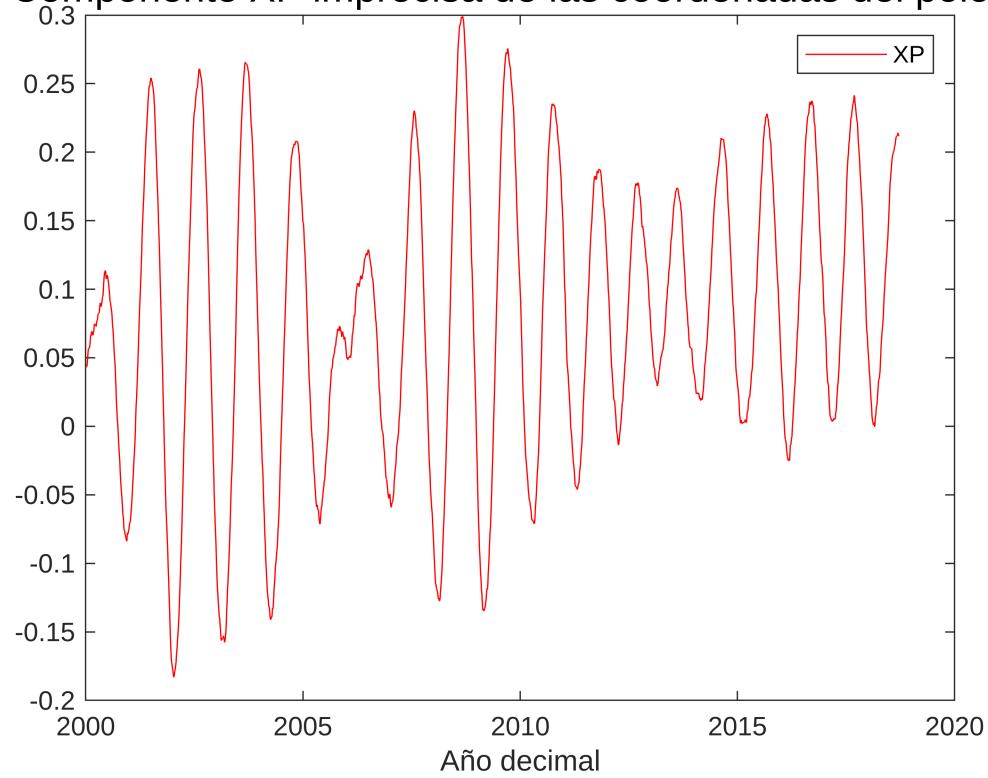
```
save([directorio_raizs directorio_sal, 'tablaImprecisos.mat'], "tablaI")
```

Graficando Datos

En esta gráfica representamos la variación de posición en X del polo durante el tiempo.

```
figure;
plot(tablaI.Fecha,tablaI.X,"red")
legend("XP")
xlabel('Año decimal')
sgtitle('Componente XP imprecisa de las coordenadas del polo')
```

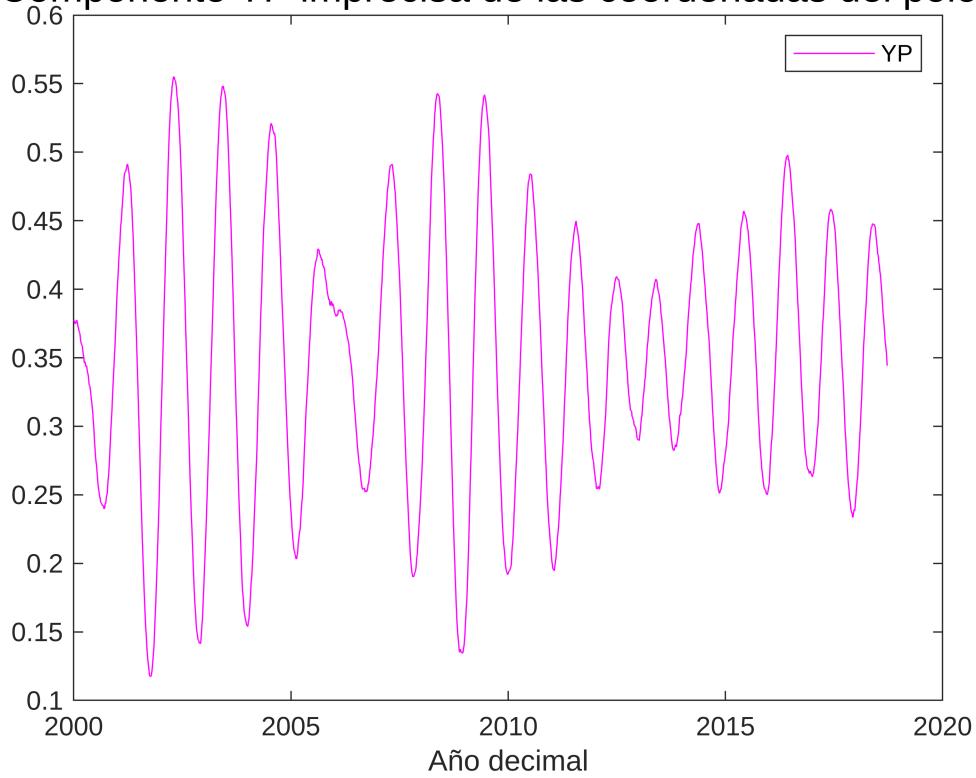
Componente XP imprecisa de las coordenadas del polo



En esta gráfica representamos la variación de posición en Y del polo durante el tiempo.

```
figure;
plot(tablaI.Fecha,tablaI.Y, "magenta")
legend( "YP" )
xlabel( 'Año decimal' )
sgtitle( 'Componente YP imprecisa de las coordenadas del polo' )
```

Componente YP imprecisa de las coordenadas del polo

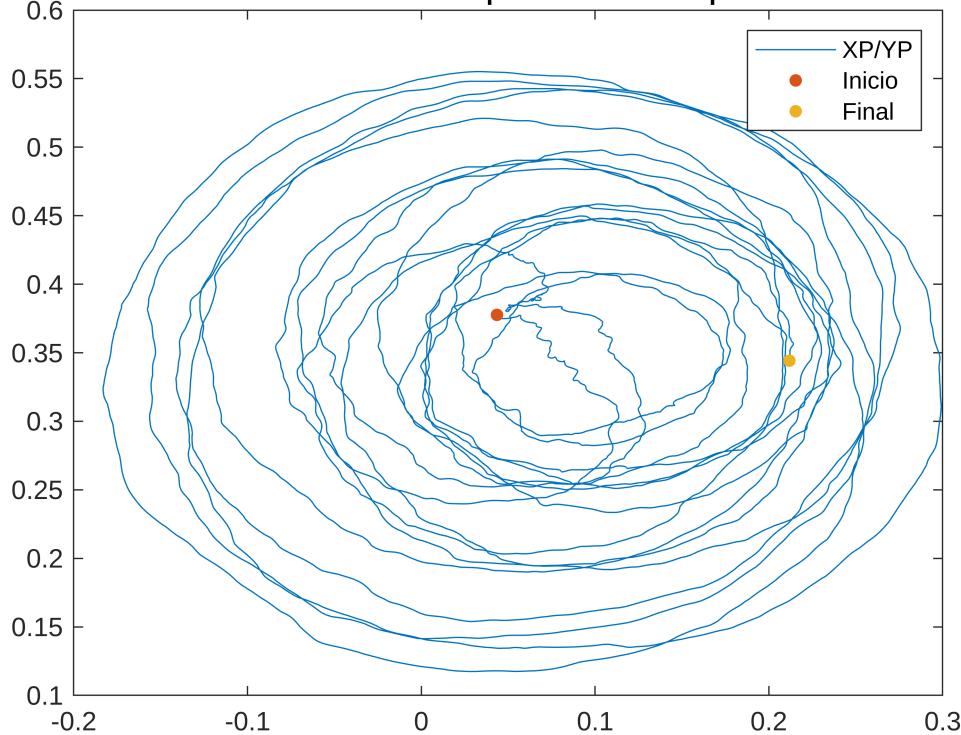


Se puede ver que ambas componentes varían de forma parecida, pero la componente Y toma valores mayores. Siendo el máximo para X de 0.3 y para Y 0.55. Parece que describe un periodo cada ciertos años, pero no tenemos suficientes datos para saberlo.

Ahora vemos la relación de posición de X e Y, y vemos que la posición del polo va variando describiendo circunferencias.

```
figure;
plot(tablaI.X,tablaI.Y)
hold on
plot(tablaI.X(1),tablaI.Y(1),".",'MarkerSize',16)
plot(tablaI.X(fila),tablaI.Y(fila),".",'MarkerSize',16)
legend("XP/YP","Inicio","Final")
sgtitle('Coordenadas imprecisas del polo')
hold off
```

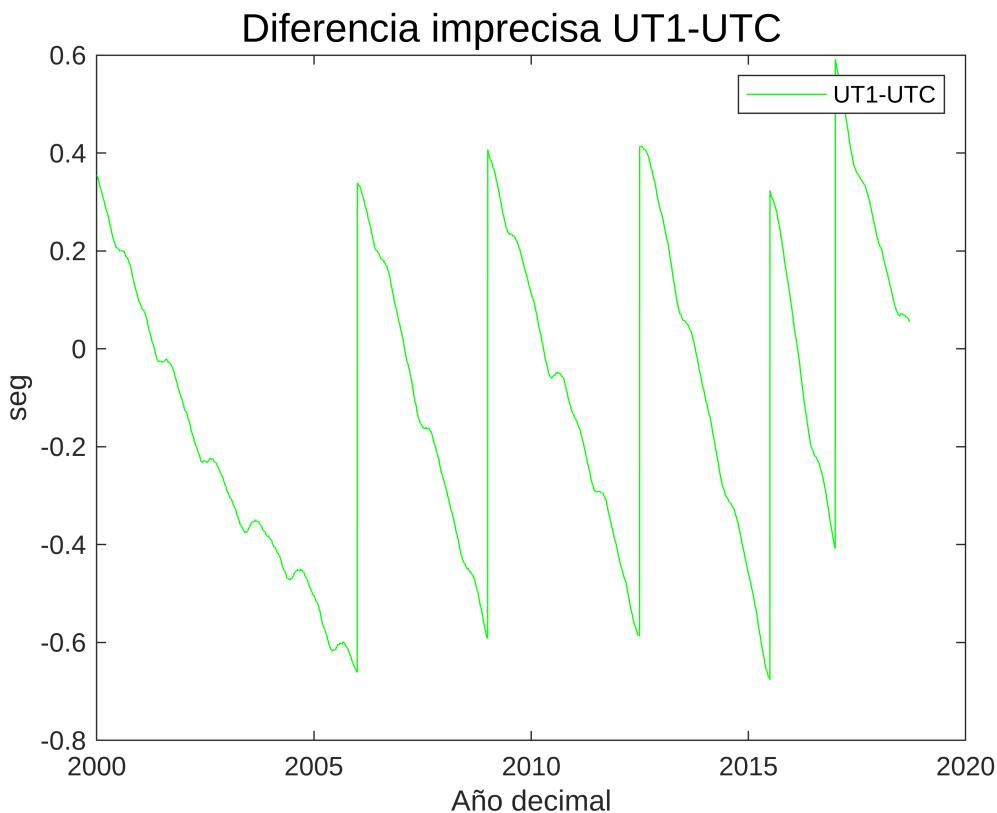
Coordenadas imprecisas del polo



Vemos que el polo describe elipses irregulares, superponiéndose.

Por último, graficamos las diferencias de UT1, que es la corrección de UT0 por el efecto del movimiento polar en la longitud del sitio de observación, y UTC, que es el tiempo universal coordinado.

```
figure;
plot(tablaI.Fecha,tablaI.UT, "green")
legend( "UT1-UTC" )
xlabel( 'Año decimal' )
ylabel( 'seg' )
sgtitle( 'Diferencia imprecisa UT1-UTC' )
```



Finalmente, para el tiempo observamos que su corrección hace una gráfica escalonada.

Adrián Fernández Tejada - Adrián Sánchez Loureiro

Funciones definidas

Aquí tenemos la función citada anteriormente, que nos transforma la fecha a un número decimal.

```
function [doy,fraction] = date2doy(inputDate)
%DATE2DOY Converts the date to decimal day of the year.
% [doy,fraction] = date2doy(inputDate)
%
% Descriptions of Input Variables:
% inputDate: Input date as a MATLAB serial datenumber
%
% Descriptions of Output Variables:
% doy: Decimal day of the year. For instance, an output of 1.5 would
%       indicate that the time is noon on January 1.
% fraction: Outputs the fraction of a year that has elapsed by the input
%           date.
%
% Example(s):
% >> [doy,frac] = date2doy(datenum('1/25/2004'));
%
```

```

% See also:

% Author: Anthony Kendall
% Contact: anthony [dot] kendall [at] gmail [dot] com
% Created: 2008-03-11
% Copyright 2008 Michigan State University.

%Want inputs in rowwise format
[doy,fraction] = deal(zeros(size(inputDate)));
inputDate = inputDate(:);

%Parse the inputDate
[dateVector] = datevec(inputDate);

%Set everything in the date vector to 0 except for the year
dateVector(:,2:end) = 0;
dateYearBegin = datenum(dateVector);

%Calculate the day of the year
doyRow = inputDate - dateYearBegin;

%Optionally, calculate the fraction of the year that has elapsed
flagFrac = (nargout==2);
if flagFrac
    %Set the date as the end of the year
    dateVector(:,1) = dateVector(:,1) + 1;
    dateYearEnd = datenum(dateVector);
    fracRow = (doyRow - 1) ./ (dateYearEnd - dateYearBegin);
end

%Fill appropriately-sized output array
doy(:) = doyRow;
if flagFrac
    fraction(:) = fracRow;
end
end

```

Tratamiento de Datos Precisos

Conceptos

- Semana GPS: El GPS proporciona información horaria precisa contando el número de segundos en una semana y este número semanal se almacena como un número binario de diez bits. Esto significa que el rango total de semanas se limita a 1024 (es decir, 19,7 años) antes de pasar de nuevo a 0.
- Día Juliano Modificado: El calendario juliano es un método para identificar el día actual a través de la cuenta del número de días que han pasado desde una fecha pasada y arbitraria. El número de días se llama día juliano, abreviado como DJ. El origen, DJ=0, es el 1 de enero de 4713 A.C. (o 1 de enero de -4712, ya que no hubo año 0). Los días julianos son muy útiles porque hacen que sea muy sencillo determinar el número de días entre dos eventos, solo con restar los números de sus días julianos. Hacer ese cálculo con el calendario normal (gregoriano) es muy difícil, ya que los días se agrupan en meses, que contienen un número variable de días, complicado además por la presencia de los años bisiestos.

```
pathe_diag = '/home/desk-chaguarma/Documents/UNI/SAG/PRACTICAS/PR1/
DIAGRAMAS/';
figure;
[im, map, alpha] = imread([pathe_diag 'PR1-B2.png']);
f = imshow(im);
set(f, 'AlphaData', alpha);
```

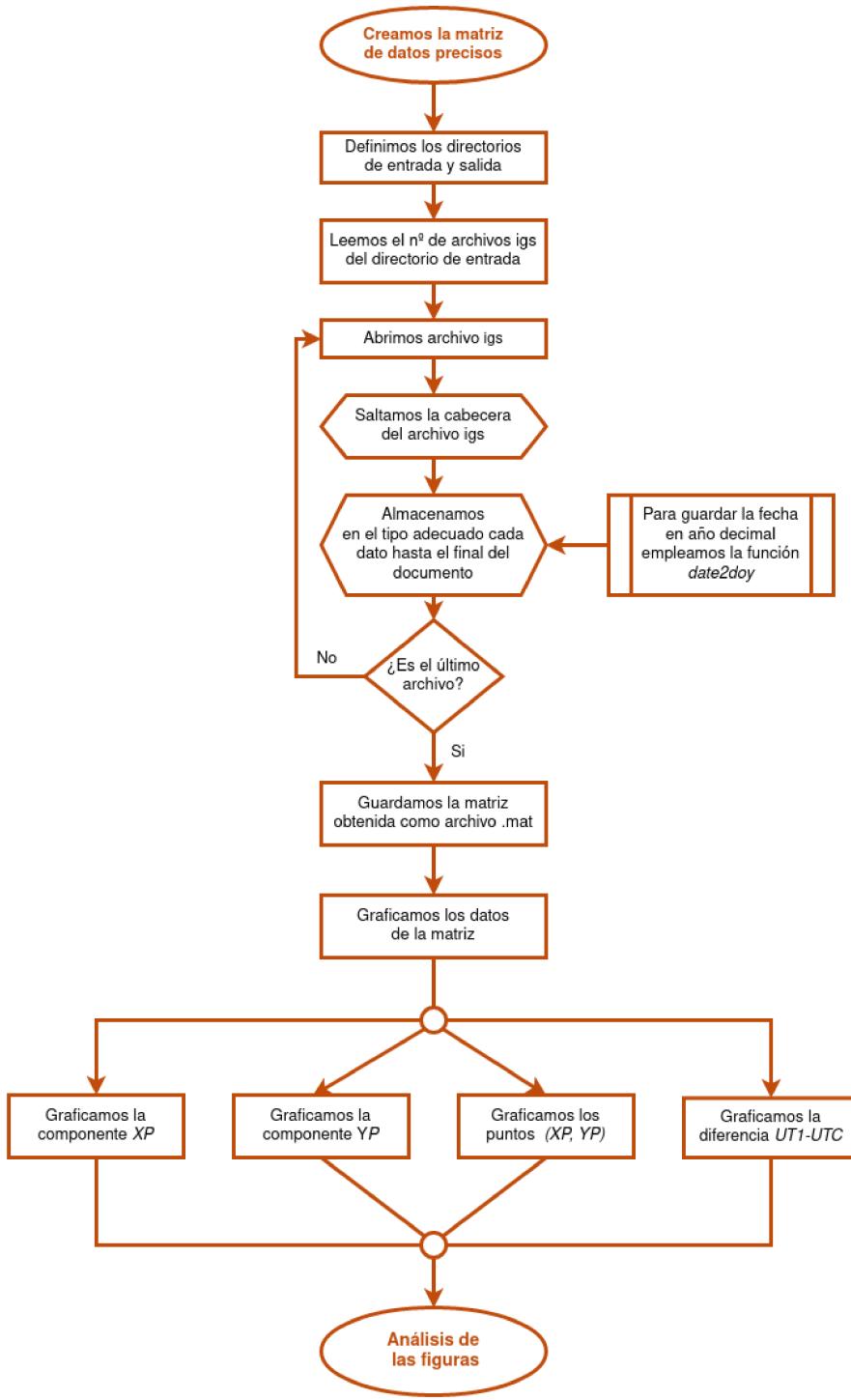


Tabla de datos

Definamos la tabla `tablaP` que almacenará los siguientes datos que extraeremos de los archivos en la carpeta `/PRECISOS`:

`fecha, X-pol, Y-pol, UT1-UTC, Xsig, Ysig, UTsig.`

Tomaremos datos desde 51545.50 a 58383.50 en fecha juliana modificada.

Primero, definimos el tipo de dato que almacenará cada columna y su nombre.

```
varTipos = [ "double", "double", "double", "double", "double", "double", "double" ];
varNames = [ "Fecha", "X", "Y", "UT", "X-SIG", "Y-SIG", "UT-SIG" ];
```

Segundo, una variable auxiliar en el que almacenaremos el número de columnas.

```
sz = [1 length(varNames)];
```

Finalmente, definimos la tabla con lo citado anteriormente.

```
tablaP = table('Size',sz,'VariableTypes',varTipos,'VariableNames',varNames);
```

A continuación, la variable auxiliar *fila* para almacenar el nº de filas que tiene la tabla.

```
fila = 0;
```

Código Realizado

Definimos en primer lugar la ruta donde se encuentra nuestro archivo de datos en la variable *directorio_raiz*. También una variable, *directorio_datos*, con el nombre de la carpeta donde están los datos. Finalmente, la ruta completa se almacena en *pathe_entrada*.

```
directorio_raiz = '/home/desk-chaguarma/Documents/UNI/SAG/PRACTICAS/PR1/';
directorio_datos = 'PRECISOS/';
directorio_sal= 'SALIDA/';
directorio_diag= 'DIAGRAMAS/';

pathe_entrada=[directorio_raiz directorio_datos];
pathe_diag=[directorio_raiz directorio_diag];
```

Primero, la función *fullfile* crea una especificación de archivo completa a partir de la carpeta y nombres de archivo especificados. Finalmente, almacenamos en *archivos* la enumeración dada por la función *dir*. Esta enumera los archivos y carpetas que coinciden con el nombre dado.

```
archivos = dir(fullfile(pathe_entrada, "igs*"));
```

Tras lo anterior, almacenamos el número de archivos reconocidos en *numero*.

```
numero=size(archivos)
```

```
numero = 1x2
      977     1
```

Para obtener los datos de los archivos .erp, leeremos línea por línea estos documentos. Para llegar a las líneas de datos, saltaremos un número determinado de líneas de cabecera.

Hay dos formatos de cabecera en los archivos .erp, con su número específico de líneas a omitir. Hemos almacenado estos números en *elim* y *elimEOP*.

```
elim = 7;
elimEOP = 2;
```

Para leer cada archivo, realizaremos un bucle *for* en el que leeremos del primer al último archivo reconocido.

```
for i=1:numero(1)
    % Tomamos el nombre del archivo i
    fichero_entrada=[archivos(i).name];
    % Abrimos el archivo i empleando fopen.
    fidin = fopen([path_entrada fichero_entrada], 'r');
```

Distinguiremos el formato del documento abierto en *fidin*, a partir de los tres primeros caracteres de la segunda fila del documento. Tras ello, con un bucle *for*, saltamos el número de líneas necesarias. En caso de no seguir un formato, se devuelve el nombre del archivo y paramos el bucle.

```
fgetl(fidin);
registro = fgetl(fidin);

if registro(1:3) == 'EOP'
    % Nos saltamos elimEOP filas
    for h = 1:elimEOP
        fgetl(fidin);
    end
elseif registro(1:3) == 'Sou'
    % Nos saltamos elim filas
    for h = 1:elim
        fgetl(fidin);
    end
else
    fichero_entrada
    break
end
```

Una vez omitidas las líneas de cabecera, identificado el patrón del archivo, obtenemos los datos de cada fila.

Para ello, emplearemos un bucle *while*, que lea el archivo hasta su final.

```
while ~feof(fidin)
```

Leemos la línea con la función *fgetl* y tomamos los tres primeros caracteres para controlar si hemos llegado al final del archivo. A partir de una función *if* controlamos si ha llegado al final.

```
registro = fgetl(fidin);
control_fin = registro(1:3);
```

```

if control_fin=='EOF'
    break

```

Una vez comprobado que no es el final del documento, con la función *split*, dividimos la fila de datos que estan separados por espacios y los almacenamos como un vector de caracteres en *registro*. Además, al añadir una nueva fila, aumentamos el contador *fila*,

```

else
    fila = fila + 1;
    registro = split(registro);
    registro = registro(1:8);

```

Para almacenar la fecha, primero transformamos el dato de fecha juliana modificada a formato año-mes-día (yyyy-MM-dd). Tras ello, a partir de la función *date2doy* obtenida [aquí](#), transformamos la fecha a año con decimal.

```

fecha = datetime(str2num(cell2mat(registro(1))),
'ConvertFrom', 'mjd', 'Format', 'yyyy-MM-dd');

[doy,frac] = date2doy(datenum(fecha));

frac = extractAfter(string(frac),1);
fecha = strcat(string(year(fecha)),frac);

```

Finalmente, almacenamos los datos deseados en la tabla definida anteriormente. Almacenamos los datos en formato double.

```

tablaP(fila,:) = {double(string(fecha)),
double(string(registro(2)))*10^(-6), double(string(registro(3)))*10^(-6),
double(string(registro(4)))*10^(-7), double(string(registro(6))),
double(string(registro(7))), double(string(registro(8)))};
end
end
fclose(fidin);
end
%fclose(fidsal);

```

Tabla Obtenida

Aquí tenemos la tabla obtenida y el número de datos almacenados.

```
fila
```

```
fila = 6839
```

```
tablaP(1:8,:)
```

```
ans = 8x7 table
```

	Fecha	X	Y	UT	X-SIG	Y-SIG	UT-SIG
1	2.0000e+03	0.0436	0.3777	0.3542	71	33	208
2	2.0000e+03	0.0436	0.3773	0.3535	65	32	190
3	2.0000e+03	0.0433	0.3770	0.3530	55	37	247
4	2.0000e+03	0.0431	0.3766	0.3525	54	37	316
5	2.0000e+03	0.0433	0.3763	0.3521	61	38	290
6	2.0000e+03	0.0437	0.3763	0.3517	71	30	293
7	2.0000e+03	0.0438	0.3762	0.3513	77	44	281
8	2.0000e+03	0.0437	0.3757	0.3508	62	26	255

Guardamos la tabla tablaP de los datos precisos para compararla en otro archivo con los datos imprecisos.

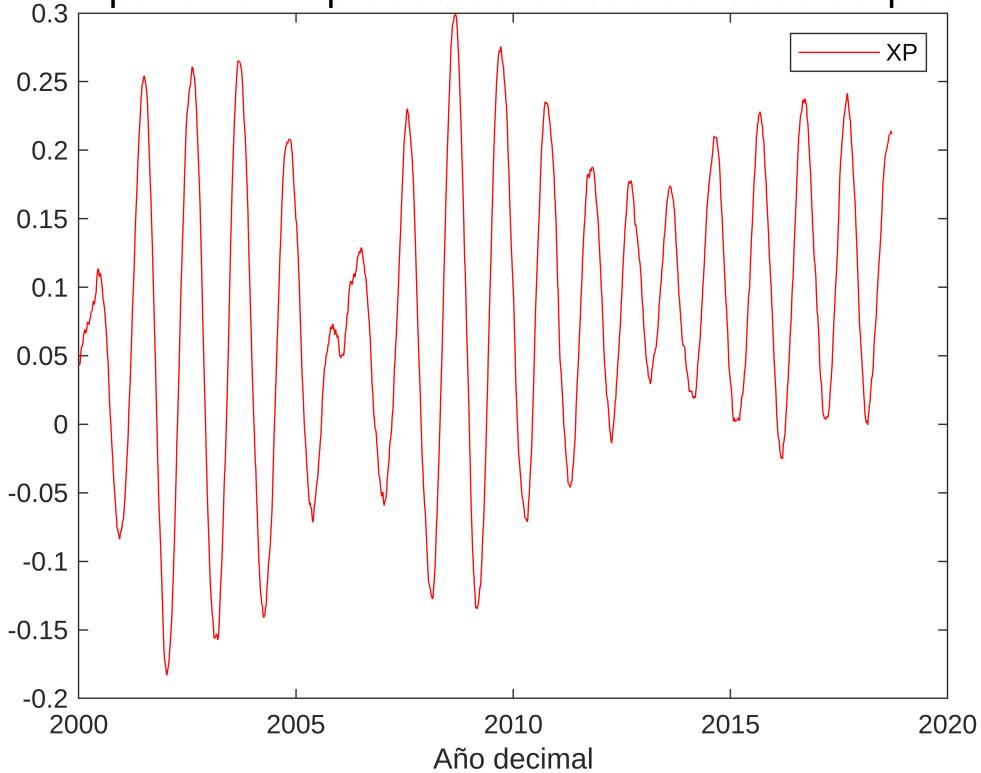
```
save([directorio_raiz directorio_sal, 'tablaPrecisos.mat'], "tablaP")
```

Graficando Datos

En esta gráfica representamos la variación de posición en X del polo durante el tiempo.

```
figure;
plot(tablaP.Fecha,tablaP.X, "red")
legend("XP")
xlabel('Año decimal')
sgtitle('Componente XP precisa de las coordenadas del polo')
```

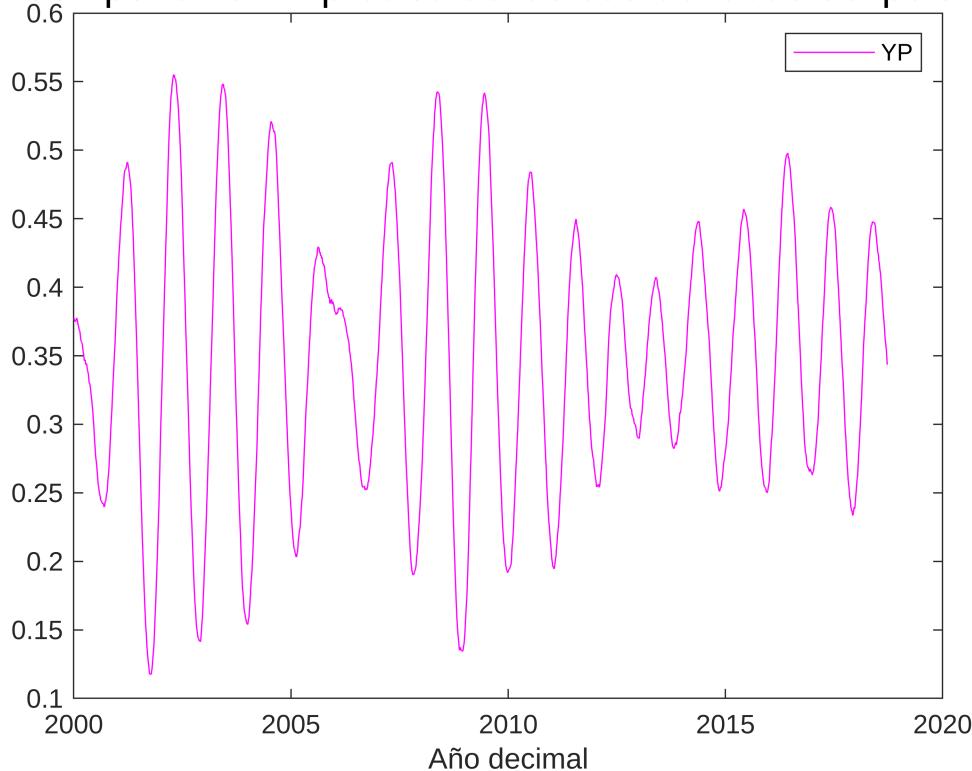
Componente XP precisa de las coordenadas del polo



En esta gráfica representamos la variación de posición en Y del polo durante el tiempo.

```
figure;
plot(tablaP.Fecha,tablaP.Y, "magenta")
legend( "YP" )
xlabel( 'Año decimal' )
sgtitle( 'Componente YP precisa de las coordenadas del polo' )
```

Componente YP precisa de las coordenadas del polo

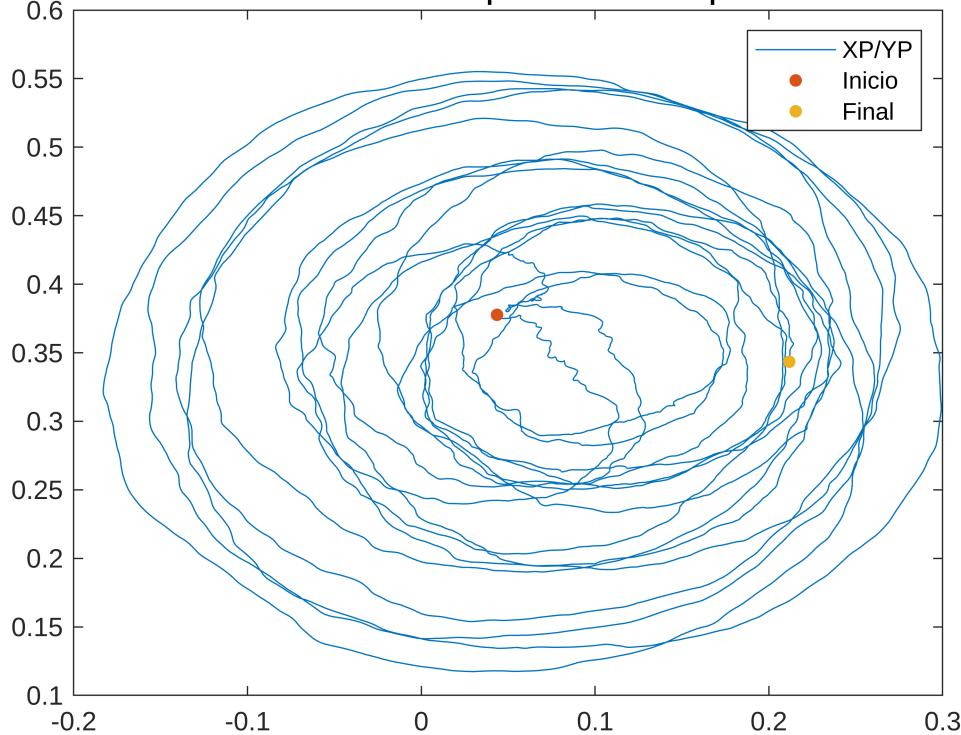


Se puede ver que ambas componentes varían de forma parecida, pero la componente Y toma valores mayores. Siendo el máximo para X de 0.3 y para Y 0.55. Parece que describe un periodo cada ciertos años, pero no tenemos suficientes datos para saberlo.

Ahora vemos la relación de posición de X e Y, y vemos que la posición del polo va variando describiendo circunferencias.

```
figure;
plot(tablaP.X,tablaP.Y)
hold on
plot(tablaP.X(1),tablaP.Y(1),".",'MarkerSize',16)
plot(tablaP.X(fila),tablaP.Y(fila),".",'MarkerSize',16)
legend("XP/YP","Inicio","Final")
sgtitle('Coordenadas imprecisas del polo')
hold off
```

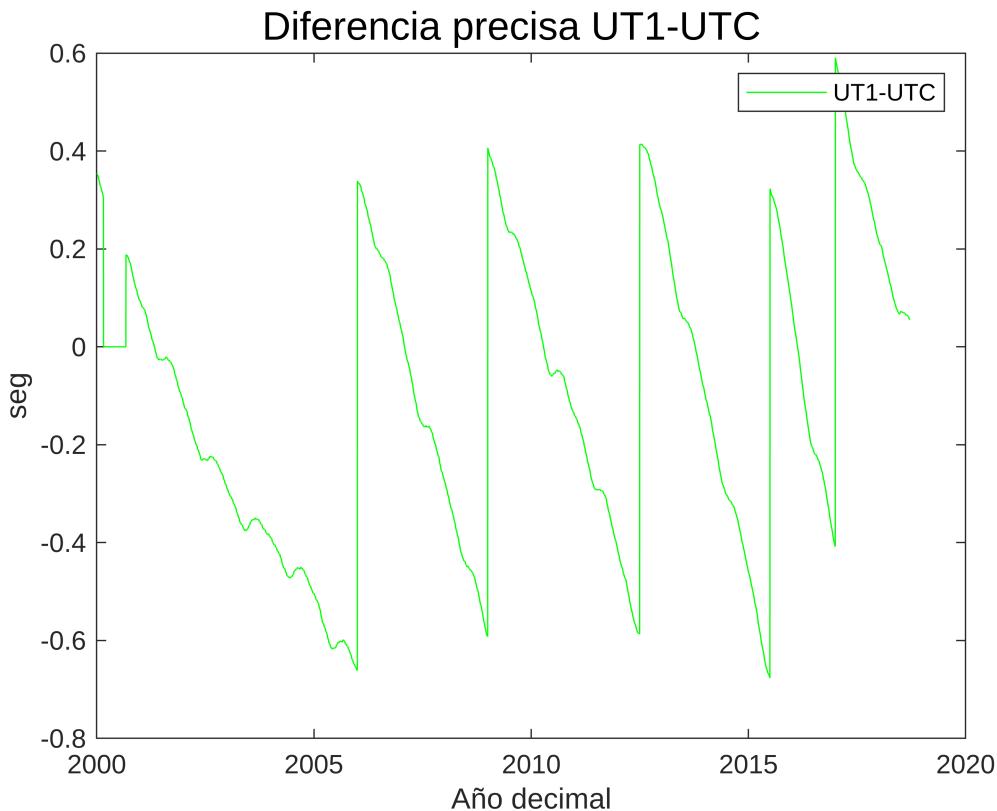
Coordenadas imprecisas del polo



Vemos que el polo describe elipses irregulares, superponiéndose.

Por último, graficamos las diferencias de UT1, que es la corrección de UT0 por el efecto del movimiento polar en la longitud del sitio de observación, y UTC, que es el tiempo universal coordinado.

```
figure;
plot(tablaP.Fecha,tablaP.UT, "green")
legend( "UT1-UTC" )
xlabel( 'Año decimal' )
ylabel( 'seg' )
sgtitle( 'Diferencia precisa UT1-UTC' )
```



Finalmente, para el tiempo observamos que su corrección hace una gráfica escalonada.

Adrián Fernández Tejada - Adrián Sánchez Loureiro

Funciones definidas

Aquí tenemos la función citada anteriormente, que nos transforma la fecha a un número decimal.

```

function [doy,fraction] = date2doy(inputDate)
%DATE2DOY Converts the date to decimal day of the year.
% [doy,fraction] = date2doy(inputDate)
%
% Descriptions of Input Variables:
% inputDate: Input date as a MATLAB serial datenumber
%
% Descriptions of Output Variables:
% doy: Decimal day of the year. For instance, an output of 1.5 would
%       indicate that the time is noon on January 1.
% fraction: Outputs tratas_MAM_2_3he fraction of a year that has elapsed
by the input
%       date.
%
% Example(s):
% >> [doy,frac] = date2doy(datenum('1/25/2004'));
%
```

```

% See also:

% Author: Anthony Kendall
% Contact: anthony [dot] kendall [at] gmail [dot] com
% Created: 2008-03-11
% Copyright 2008 Michigan State University.

%Want inputs in rowwise format
[doy,fraction] = deal(zeros(size(inputDate)));
inputDate = inputDate(:);

%Parse the inputDate
[dateVector] = datevec(inputDate);

%Set everything in the date vector to 0 except for the year
dateVector(:,2:end) = 0;
dateYearBegin = datenum(dateVector);

%Calculate the day of the year
doyRow = inputDate - dateYearBegin;

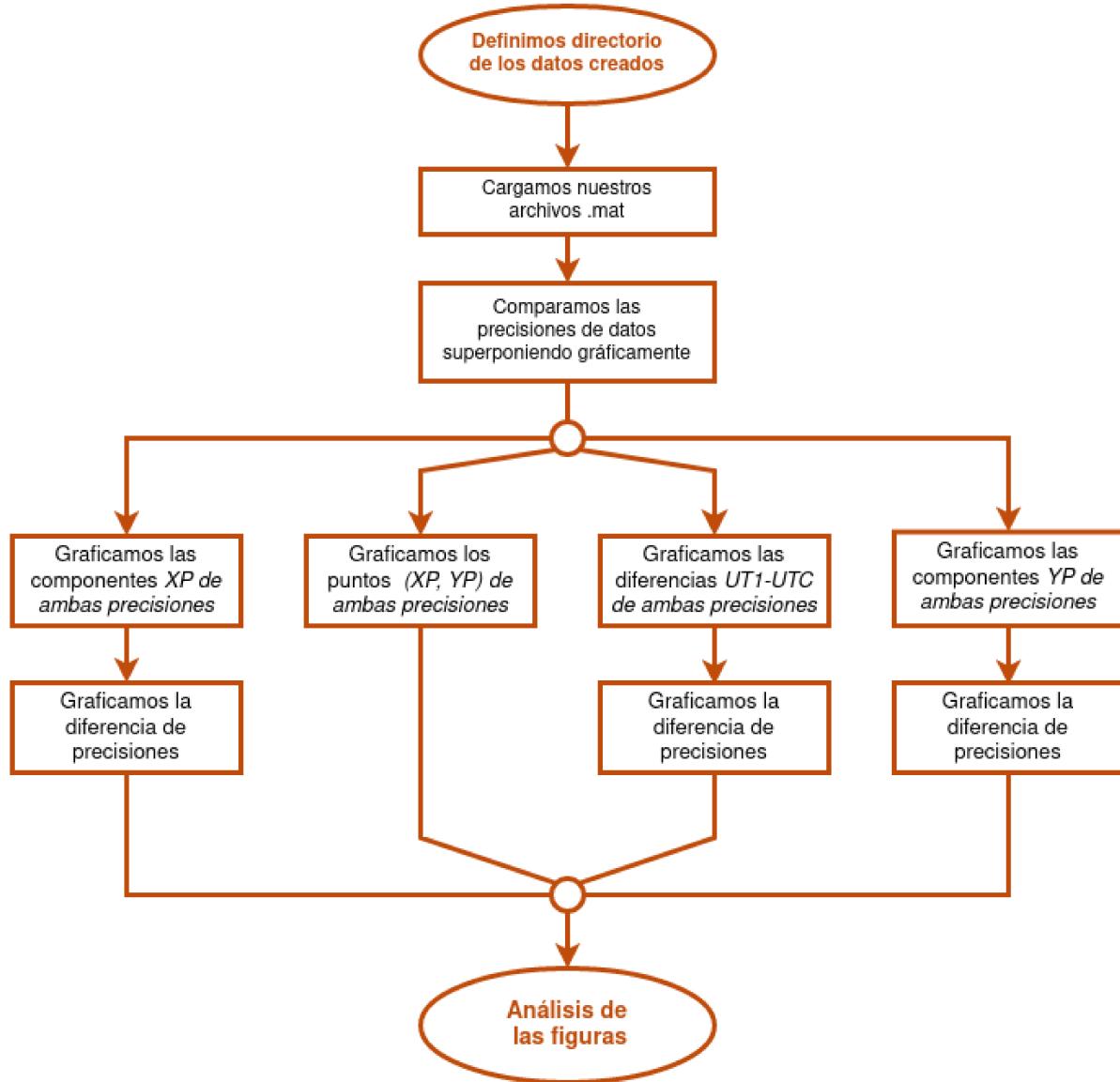
%Optionally, calculate the fraction of the year that has elapsed
flagFrac = (nargout==2);
if flagFrac
    %Set the date as the end of the year
    dateVector(:,1) = dateVector(:,1) + 1;
    dateYearEnd = datenum(dateVector);
    fracRow = (doyRow - 1) ./ (dateYearEnd - dateYearBegin);
end

%Fill appropriately-sized output array
doy(:) = doyRow;
if flagFrac
    fraction(:) = fracRow;
end
end

```

Comparando los Datos

```
pathe_diag = '/home/desk-chaguarma/Documents/UNI/SAG/PRACTICAS/PR1/  
DIAGRAMAS/';  
  
[im, map, alpha] = imread([pathe_diag 'PR1-B3.png']);  
figure; f = imshow(im);  
set(f, 'AlphaData', alpha);
```



Cargando Datos

En primer lugar cargamos los dos archivos donde guardamos las tablas de datos anteriormente creados.

```
directorio_raiz = '/home/desk-chaguarma/Documents/UNI/SAG/PRACTICAS/PR1/';  
directorio_datos = 'SALIDA/';
```

```
directorio_diag= 'DIAGRAMAS/' ;  
  
pathe_diag=[directorio_raiz directorio_diag]  
  
pathe_diag =  
'/home/desk-chaguarma/Documents/UNI/SAG/PRACTICAS/PR1/DIAGRAMAS/'
```

```
load([directorio_raiz directorio_datos 'tablaImprecisos.mat'])  
load([directorio_raiz directorio_datos 'tablaPrecisos.mat'])  
  
size(tablaI)
```

```
ans = 1x2  
6839 7
```

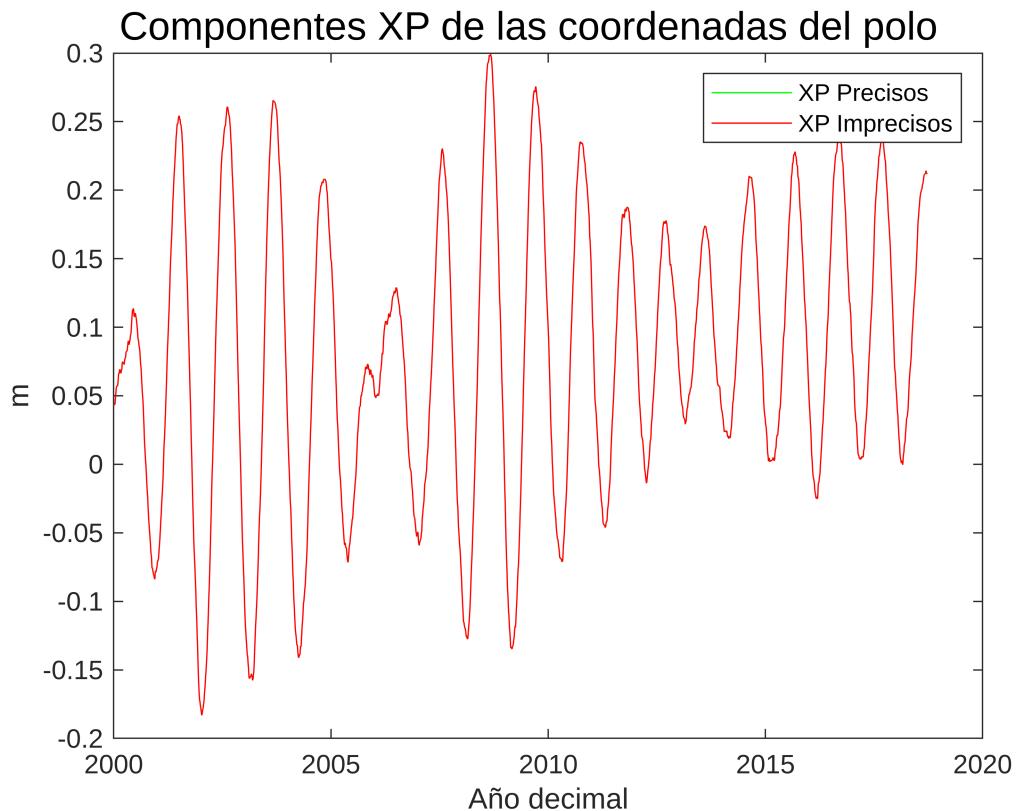
```
size(tablaP)
```

```
ans = 1x2  
6839 7
```

Polo X respecto al tiempo

En primer lugar comparamos las gráficas de la posición X del polo, y no apreciamos ninguna variación.

```
figure;  
plot(tablaP.Fecha,tablaP.X, "green")  
hold on  
plot(tablaI.Fecha,tablaI.X, "red")  
legend("XP Precisos", "XP Imprecisos")  
xlabel('Año decimal')  
ylabel('m')  
sgtitle('Componentes XP de las coordenadas del polo')  
hold off
```



Sin embargo, si en la gráfica anterior nos acercamos lo suficiente, vemos pequeñas diferencias entre los datos.

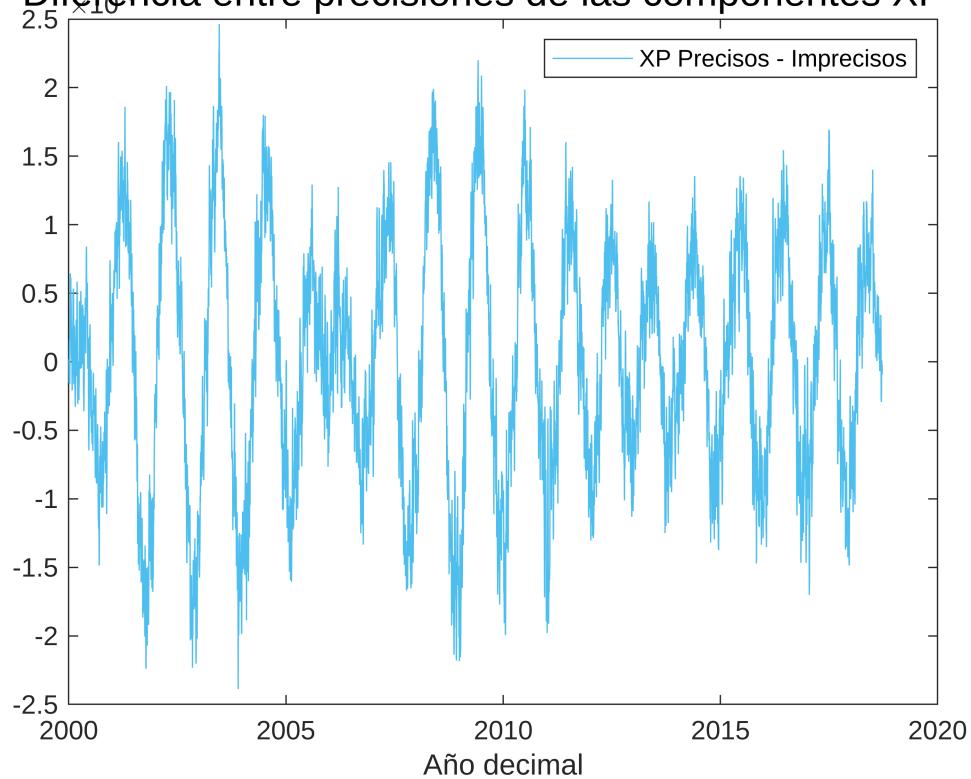
Aquí la gráfica de la diferencia de los datos de XP precisos e imprecisos.

```

figure;
plot(tablaP.Fecha,tablaP.X-tablaI.X,'Color',[0.3010 0.7450 0.9330])
legend("XP Precisos - Imprecisos")
xlabel('Año decimal')
sgtitle('Diferencia entre precisiones de las componentes XP')
hold off

```

Diferencia entre precisiones de las componentes XP

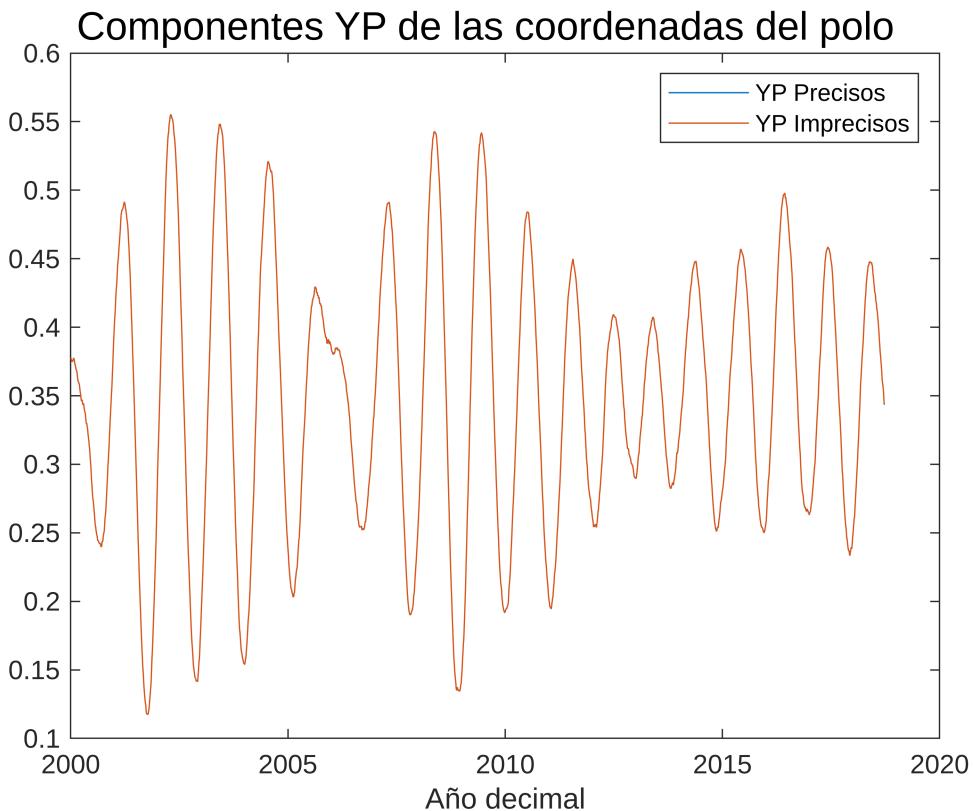


Se observa que la diferencia sigue el mismo patrón que los datos de la componente X.

Polo Y respecto al tiempo

Lo siguiente es comparar la posición Y del polo, y como antes no observamos ninguna diferencia.

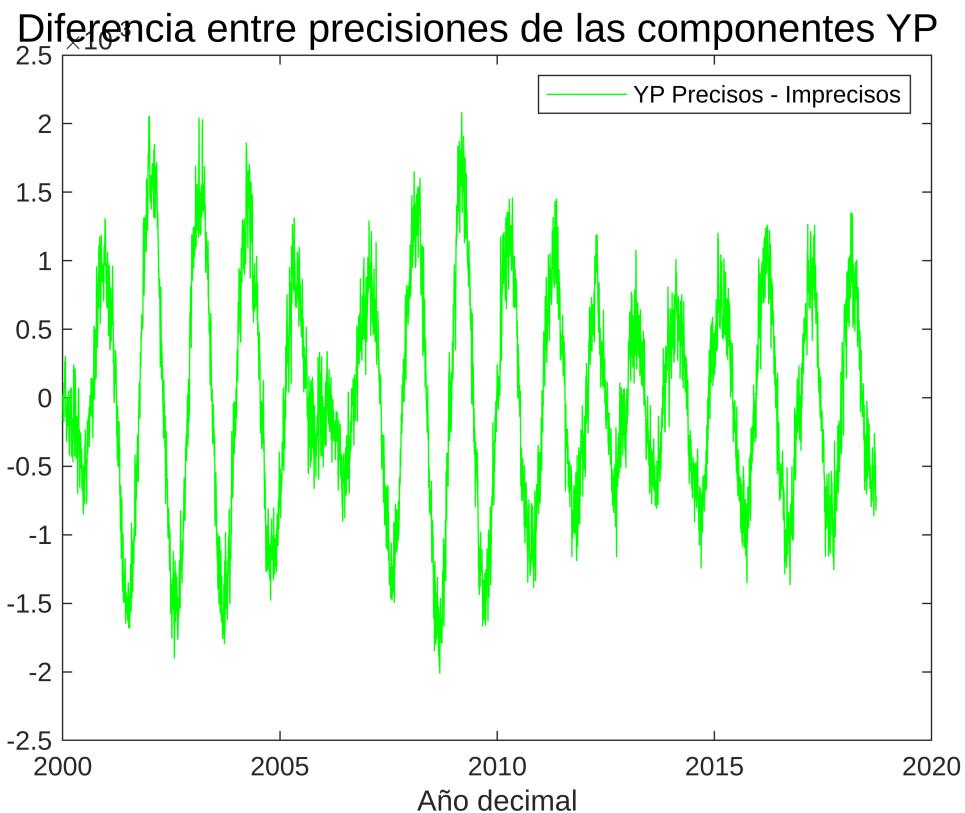
```
figure;
plot(tablaP.Fecha,tablaP.Y)
hold on
plot(tablaI.Fecha,tablaI.Y)
legend("YP Precisos","YP Imprecisos")
xlabel('Año decimal')
sgtitle('Componentes YP de las coordenadas del polo')
hold off
```



Como pasó anteriormente, si nos acercamos lo suficiente vemos pequeñas diferencias.

Aquí la gráfica de la diferencia de los datos de YP precisos e imprecisos.

```
figure;
plot(tablaP.Fecha,tablaP.Y-tablaI.Y, "green")
legend( "YP Precisos - Imprecisos" )
xlabel( 'Año decimal' )
sgtitle('Diferencia entre precisiones de las componentes YP')
hold off
```



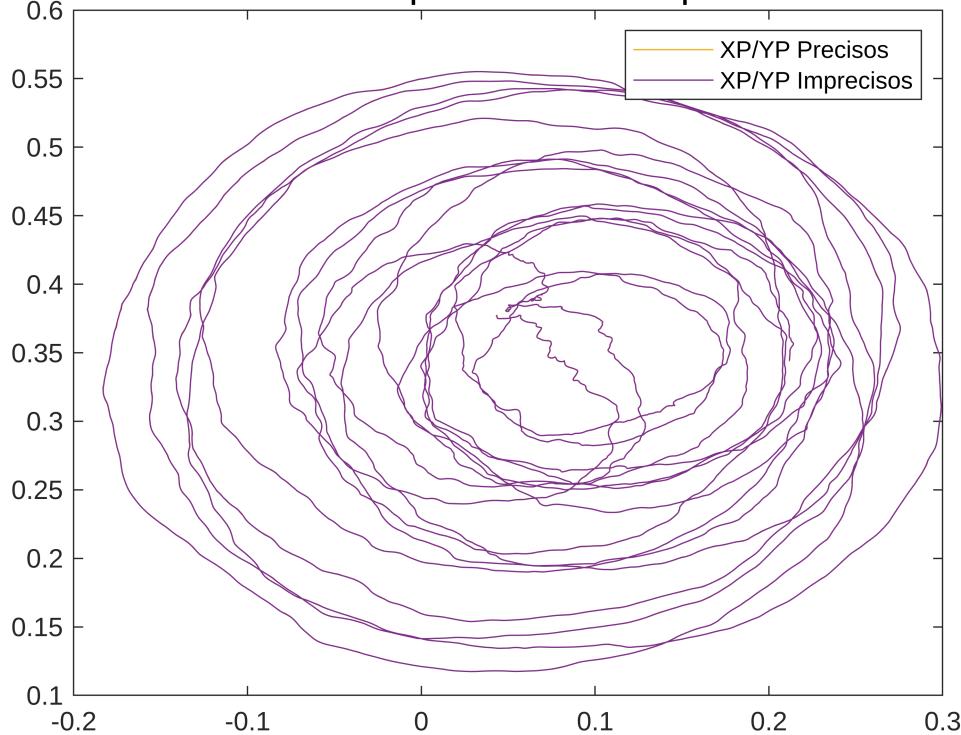
Se observa que la diferencia sigue el mismo patrón que los datos de la componente X.

Posiciones del polo X e Y

Ahora comparamos las posiciones X e Y del polo, no hay variación aparente.

```
figure;
plot(tablaP.X,tablaP.Y,'Color',[0.9290 0.6940 0.1250])
hold on
plot(tablaI.X,tablaI.Y,'Color',[0.4940 0.1840 0.5560])
legend("XP/YP Precisos","XP/YP Imprecisos")
sgtitle('Coordenadas del polo de ambas precisiones')
hold off
```

Coordenadas del polo de ambas precisiones

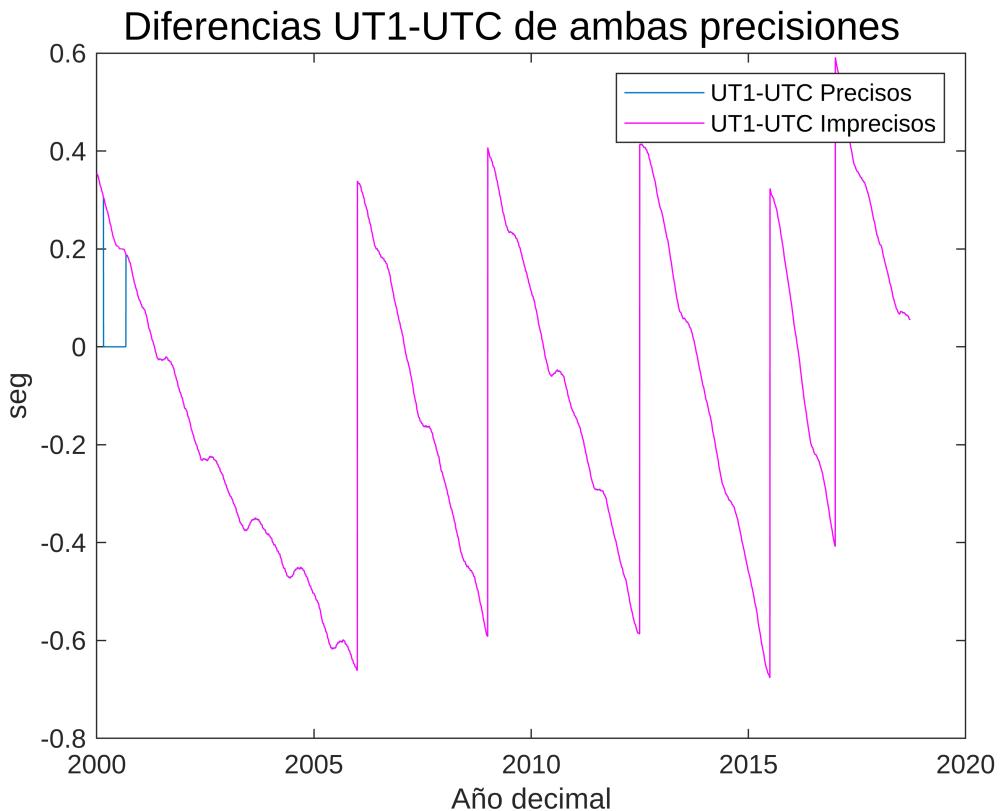


Y como era de esperar, al unir las gráficas donde estén X e Y siguen habiendo diferencias al acercarnos lo suficiente.

Variación UT1-UTC respecto del tiempo

En esta última gráfica la diferencia es más evidente, ya que en el inicio de ella hay un escalón de los datos precisos, así que podemos concluir que los otros datos no son tan acertados como los precisos.

```
figure;
plot(tablaP.Fecha,tablaP.UT)
hold on
plot(tablaI.Fecha,tablaI.UT, "magenta")
legend( "UT1-UTC Precisos", "UT1-UTC Imprecisos" )
xlabel( 'Año decimal' )
ylabel( 'seg' )
sgtitle( 'Diferencias UT1-UTC de ambas precisiones' )
hold off
```

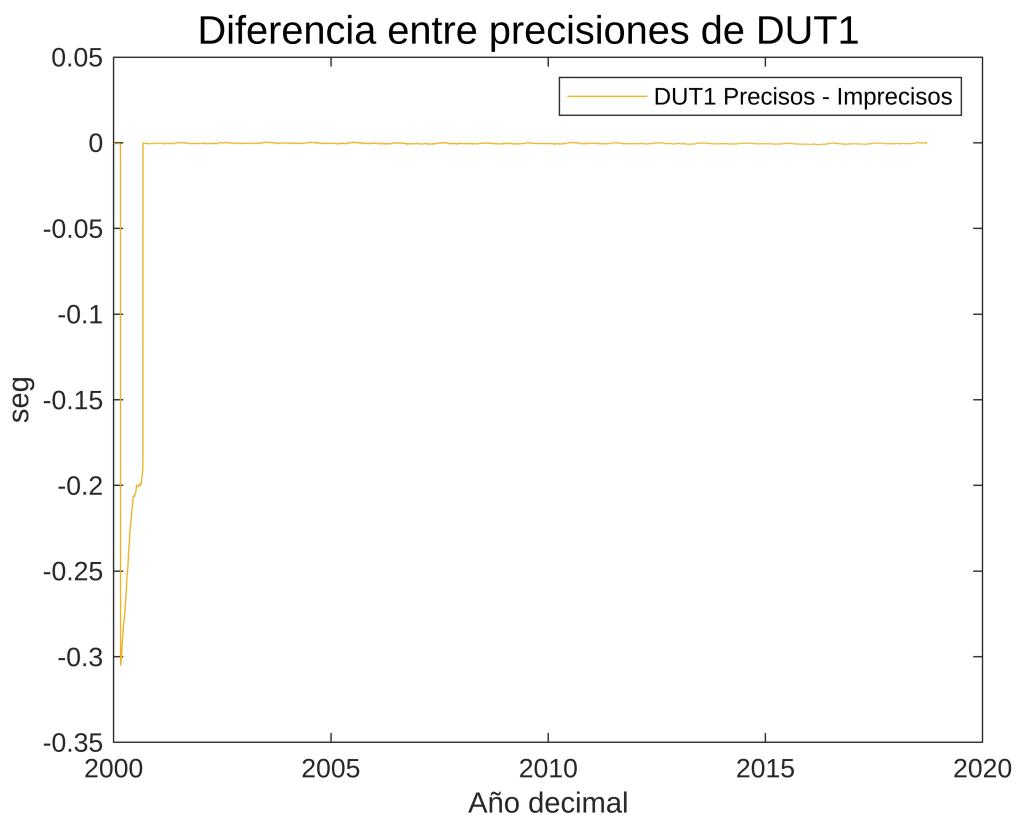


Aquí la gráfica de la diferencia de los datos de DTU1 precisos e imprecisos. Salvo el inicio, el resto están aproximados.

```

figure;
plot(tablaP.Fecha,tablaP.UT-tablaI.UT,'Color',[0.9290 0.6940 0.1250])
legend("DUT1 Precisos - Imprecisos ")
xlabel('Año decimal')
ylabel('seg')
shtitle('Diferencia entre precisiones de DUT1')

```



Adrián Fernández Tejada - Adrián Sánchez Loureiro