

----- **HOJA 3: ENTREGA FINAL** -----

Los ejercicios de esta hoja sirven como guión para la entrega final del tema de posicionamiento GPS, que será un fichero comprimido tipo .7z o similar con:

- Un directorio con todos los programas y datos necesarios para hacer los ejercicios de esta hoja. **Aseguraros de que no falta nada necesario para ejecutar vuestros ejemplos (funciones auxiliares, ficheros SP3, ...).**
- Fichero pdf con las **respuestas, gráficas, código**, etc. que se piden en los ejercicios que hagáis de esta hoja. Al final de cada ejercicio **indicad como reproducir los resultados** mostrados. Algo así como: "para obtener estas resultados y/o gráficas hay que ejecutar los scripts ejer1a y ejer1b en mi directorio..."

La entrega mínima son los ejercicios 1 y 2: el ejercicio 1 comprueba que las rutina básica de posicionamiento (get_pos + resto funciones auxiliares) os funciona y presenta de forma gráfica vuestros resultados, a la vez que pide un mínimo de información estadística como errores medios, máximos, etc. En el ejercicio 2 se cuantifica el efecto de una mala geometría (distribución) de los satélites en el cálculo de la posición, introduciendo el concepto de PDoP.

El resto de los ejercicios son independientes y se refieren a la implementación de las mejoras de las que hemos hablado en clase, estudio de situaciones particulares, etc.

Para cargar los datos con los que trabajar, haced `>>load obs`.

Veréis varias estructuras (yebe, vill, alac, alac2, ...) correspondientes a datos de diferentes estaciones. Cada una de estas estructuras corresponde a las observaciones tomadas durante varias horas desde una estación. Los campos más importantes son:

- **.tow:** tabla de tamaño 1 x NT con los NT tiempos (time of week) en segundos en los que la estación ha tomado datos.
- **.prn:** vector de tamaño Nsat x 1, con la lista de los PRNs de todos los satélites vistos por la estación en algún momento durante el día.
- **.obs:** tabla de Nsat x NT con los pseudo-rangos medidos a los Nsat satélites en los NT instantes de tiempo. Para los satélites no visibles en ese momento las casillas tendrán un valor -1.

Junto con las observaciones de la estación necesitaremos los correspondientes datos orbitales del día para saber donde están los satélites. En cada ejercicio se especifica el ficheros SP3 que hay que usar para procesar los datos.

Ejercicio 1: En este ejercicio procesaremos los datos de la estación **yebe**, con NT=900 observaciones tomadas cada minuto durante 15 horas (2am -> 5pm). El fichero SP3 con los datos orbitales a usar para ese día es **IGS13230.sp3**.

Reservar una matriz S de tamaño 4xNT para guardar los resultados y hacer un bucle barriendo los NT instantes de tiempo, calculando la posición con vuestra rutina get_pos. En cada paso del bucle debemos:

- 1) Extraer la k-esima columna del campo .obs con todos los pseudo-rangos de ese momento. Extraer de entre ellos los que sean válidos (que sean positivos). Extraer también los PRNs correspondientes a esas medidas. Al final terminamos con vector con n medidas y otro con n PRNs, todo ello para el tiempo de recepción Tr dado por el campo .tow.
- 2) Usando la mejor de vuestras rutinas de posicionamiento resolver el problema de posicionamiento con los datos anteriores.
- 3) Guardar el vector solución (4x1) en la k-ésima columna de la matriz de resultados S. Al terminar, en las filas de la matriz S (4xNT), tendréis las 4 componentes (x,y,z, cdt) de las soluciones para los NT tiempos.

Al resolver la primera vez (k=1) usad $X=[4855000; -325000; 4115000; 0.00]$ como hipótesis inicial (posición aproximada de Madrid y error de reloj nulo). Para el resto de tiempos (k>1) usad como punto de partida la solución del paso anterior. Adjuntar vuestro código

Extraer la 4ª fila (error de reloj del receptor cdt). **Cambiar las unidades de metros a msec (1 msec = 1/1000 segundos) y hacer un gráfico de dicho error.**

Extraer la submatriz xyz (3xNT) con solo las posiciones obtenidas. Aplicadle la rutina xyz2llh() para obtener una matriz 3 x NT con las nuevas coordenadas de longitud, latitud y altura. Calcular la media de cada coordenada (aplicando mean a las diferentes filas) y rellenar la tabla siguiente (dar las coordenadas X,Y,Z, altura con una precisión de 1 metro, y longitud/latitud con 5 decimales)

Media	X	Y	Z
Media	longitud	Latitud	altura

Introducid las coordenadas de latitud/longitud (5 decimales) en Google Maps. **Adjuntar captura de la foto aérea de la zona con el punto en cuestión.**

Extraer la fila con las alturas y guardarlas en un vector H. Luego, extraer las primeras 2 filas con las coordenadas de longitud/latitud y aplicarles la rutina ll2utm para pasarlas a coordenadas UTM planas. Extraer las coordenadas Este y Norte en sendos vectores (E,N).

Los datos usados son de una estación de referencia cuya posición verdadera (con una precisión ~ cm) es conocida y se da en el campo **XYZ**. Aplicad a dicha posición las mismas funciones (xyz2llh + ll2utm) para obtener las coordenadas exactas (altura H0, este E0, norte N0) de la estación.

Restando las coordenadas exactas a los resultados (H, E, N) obtendremos los **verdaderos errores** en altura (dH) y en la horizontal (dE y dN). Con estos errores podemos ilustrar gráficamente la dispersión de medidas tanto en el plano horizontal (E,N) como en altura (H). Hacer plots de las **errores** en altura (dH en 1D) y en el plano (dE/dN en 2D). Para el gráfico 2D poned los errores en la coordenada E (dE) en el eje de X's y los errores en la coordenada N (dN) en el eje de las Y's. Incluso con la versión más básica de get_pos() estos errores no deben superar los 100 m. [Adjuntar ambas gráficas de errores.](#)

Calculad la media de dichos errores en valor absoluto usando mean(abs(.)). Calcular también el máximo error en valor absoluto con max(abs(.)).

	dE	dN	dH
media (error)			
máximo (error)			

Por último crear un vector r (1xNT) con la distancia (en horizontal) entre cada uno de nuestros resultados y la posición real de la estación. Como conocemos las distancias en Este (dE) y Norte (dN) basta hacer:

$$r = \sqrt{dE^2 + dN^2}$$

para todos los elementos de los vectores dE y dN.


[Sabiedo que el CEP50 es la distancia D tal que el 50% de los valores de r es menor o igual que D, determinad \(tanteando\) su valor. Repetir para obtener el valor de CEP95.](#)

El archivo 'yebes.jpg' corresponde a un mapa de la zona que podéis cargar y ver con los comandos: `im=imread('yebes.jpg');` `image(im)`

Este mapa está calibrado en el mismo datum (WGS84) en que hemos hecho obtenido nuestros resultados. Su esquina superior izquierda corresponde a las coordenadas **E0=492000; N0=4487000**; Su resolución es de 1 metro: cada píxel a la derecha supone sumar 1m a la coordenada Este, mientras que cada píxel hacia abajo supone restar 1m a la coordenada Norte. [Indicad la fórmula para pasar de coordenadas \(E,N\) a píxeles \(px,py\) sobre la imagen.](#)

Convertir todas las coordenadas E,N a posiciones en píxeles y pintarlas sobre el mapa como puntos azules (usar el modificador 'b.') para ver la dispersión de los resultados. (usar hold on para pintar sobre el mapa sin machacarlo).

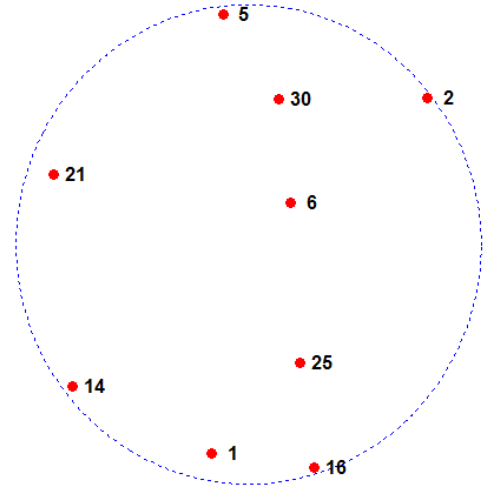
[Adjuntad la imagen resultado y un zoom sobre la zona en cuestión para apreciar los resultados.](#)

Los puntos deben caer en una zona del mapa etiquetada como Observatorio de Yebes, idealmente sobre el símbolo de un vértice topográfico 

Ejercicio 2 (efecto de la geometría de los satélite)

El gráfico adjunto ("skyplot") muestra los 9 satélites que eran visibles desde la posición de un observador cerca de Madrid ([4855000; -325000; 4115000]), a las 2 de la mañana del 15/05/2005.

Este tipo de gráfico muestra la distribución en el cielo de los satélites desde el punto de vista del observador. Por ejemplo, el satélite 5 (en la zona de arriba) estaría al Norte, muy cerca del horizonte (al estar muy cerca del borde del círculo). Por el contrario, el satélite 21 estaría en el Oeste (ligeramente al Norte) y un poco más alto sobre el horizonte. El satélite 6 estaría casi sobre nuestras cabezas (centro del círculo). La disposición geométrica de los satélites al tomar las medidas influye en los posibles errores de la posición obtenida.



Este efecto se mide con el concepto del PDoP (Position Dilution of Precision). Valores altos del PDoP hacen que los inevitables errores en las medidas afecten de forma exagerada al cálculo de la posición. El PDoP se calcula a partir de la matriz H. Primero se calcula la matriz Q como:

$$Q = (H^T H)^{-1}$$

(en MATLAB H traspuesta = H'). Observad que aunque la matriz H es de tamaño Nsat x 4 (Nsat = nº de satélites visibles), la matriz Q es siempre 4x4.

El PDoP se calcula usando los tres primeros elementos de la diagonal de Q:

$$\sigma_x = \sqrt{q_{11}} \quad \sigma_y = \sqrt{q_{22}} \quad \sigma_z = \sqrt{q_{33}} \quad \rightarrow \quad PDoP = \sigma_{XYZ} = \sqrt{q_{11} + q_{22} + q_{33}}$$

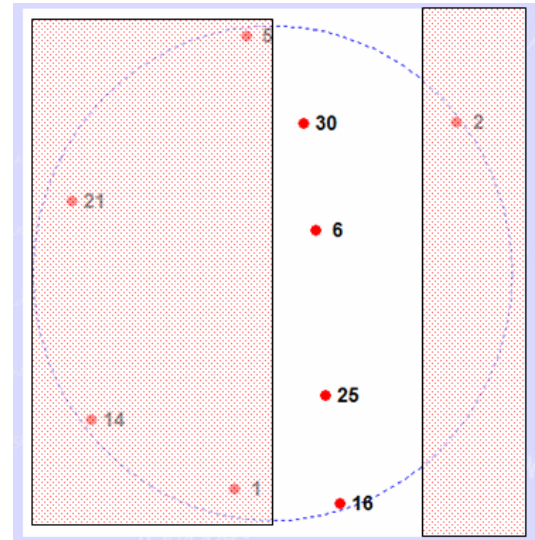
Escribir un script que calcule el PDoP para la situación anterior:

1. Con `get_data_sats()` obtener la matriz XYZ (3x9) con las posiciones de los satélites involucrados (1, 2, 5, 6, 14, 16, 21, 25 y 30) para $t=7200$ (usando 8 nodos). Usad datos del fichero **IGS13230.SP3**
2. Suponed que la posición del GPS era **pos=~[4855000; -325000; 4115000]** y usad la función `get_HR()` para obtener la matriz H.
3. A partir de H calcular la matriz Q y obtener el PDoP correspondiente.

[Adjuntad código del script, matriz Q y PDoP obtenido usando los 9 satélites.](#)

Recordad que el PDoP es un único número indicando lo fiable que es la solución obtenida con esos satélites. Al disponer de bastantes satélites y estar bien repartidos por el cielo, los valores de σ_x , σ_y , σ_z y PDoP os saldrán no mucho mayores de la unidad.

El PDoP puede empeorar notablemente si se usan menos satélites, sobre todo si están en una distribución desfavorable (en el mismo cuadrante, alineados,...). Una mala situación que es habitual es si estamos en una calle estrecha con edificios altos a ambos lados (ver gráfica adjunta). En ese caso solo vemos 4 satélites (6, 16, 25, 30) y aunque podamos resolver la posición con solo esos satélites la situación no es la ideal, lo que se reflejará en un peor PDoP (más alto).



Calcular el nuevo PDoP para los 4 satélites visibles en esta situación.

Veamos ahora el efecto práctico de tener un PDoP elevado al resolver nuestra posición. En este caso, las medidas del GPS para TODOS los satélites eran:

```
Tr=7200; prn=[1 2 5 6 14 16 21 25 30];
obs=[23096478.313; 24501265.177; 24401880.133; 20555740.449;
     24258653.281; 24581854.516; 23456029.925; 21447276.383;
     21280828.168];
```

Extraer las cuatro medidas correspondientes a los satélites 6, 16, 25 y 30. A partir de ellas vamos a repetir 1000 veces el cálculo de la posición, simulando el efecto de pequeños errores en las medidas.

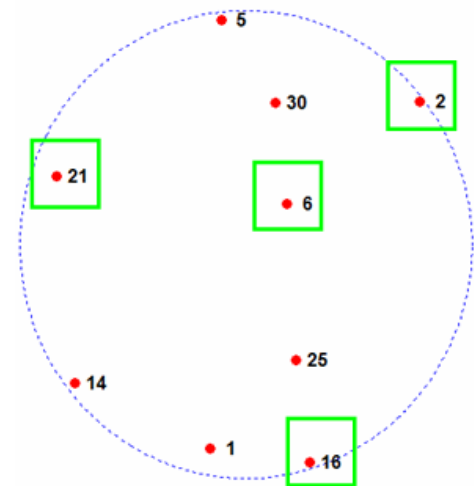
Mediante un bucle (como se hizo para la estación) resolver la posición 1000 veces. En cada paso del bucle sumaremos randn(4,1) a las 4 observaciones originales, simulando un error (diferente en cada caso) en las medidas con $\sigma=1$. Guardad los resultados en una matriz 4x1000 y calculad la desviación standard (std) de las coordenadas X, Y y Z. Como la σ del error introducido en las medidas es $\sigma=1$, las dispersión (σ) provocada en las coordenadas X, Y, Z coincidirá aproximadamente con los valores de la diagonal de la matriz Q:

$$\sigma_X = \sqrt{q_{11}} \quad \sigma_Y = \sqrt{q_{22}} \quad \sigma_Z = \sqrt{q_{33}}$$

Adjuntad los valores de las 3 desviaciones standard (std) de las coordenadas X, Y, Z obtenidas junto con las raíces cuadradas de la diagonal de la matriz Q (q_{11} , q_{22} , q_{33}).

Los receptores GPS modernos tienen múltiples "canales" por lo que pueden observar simultáneamente a todos los satélites y usarlos todos en la solución. Inicialmente los receptores GPS solo podían escuchar a un "satélite", por lo que calculaban la solución con el número mínimo de satélites posibles (4). Era por lo tanto muy importante escoger 4 buenos satélites con los que trabajar.

Con solo 4 satélites lo ideal es usar uno encima de nuestras cabezas y los otros tres repartidos en todas las direcciones (separados 120°) cerca del horizonte. En el caso anterior una buena elección serían los satélites 6, 2, 21 y 16 que se muestran en la figura adjunta.



Calcular los valores de σ_x , σ_y , σ_z y **PDOP** para esta nueva configuración de 4 satélites. Sin llegar a ser tan bajos como cuando usábamos los 9 satélites, los resultados deben ser mucho mejores que antes. [Adjuntad valores obtenidos.](#)

Búsqueda del mejor y peor caso posible con 4 satélites:

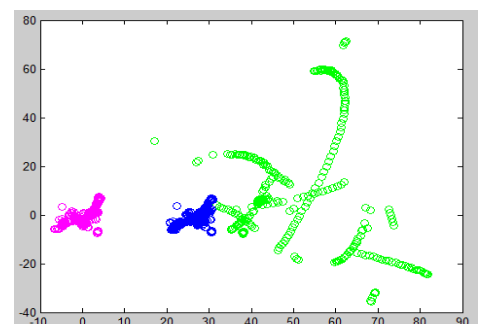
Vamos a hacer una búsqueda exhaustiva de la mejor y peor elección posible usando 4 satélites para posicionarnos. La función de MATLAB `nchoosek(lista,4)` calcula las N posibles combinaciones de una lista tomadas de 4 en 4 y las devuelve en una matriz N x 4. Usad esa función aplicada a la lista de los 9 satélites visibles para obtener todas las combinaciones de 4 satélites posibles. Calculad el PDOP de cada una de esas combinaciones, guardando los resultados en un vector 1 X N. [Adjuntad plot de ese vector usando `semilogy\(\)`.](#)

Usar la función `min` y `max` para encontrar el mejor y peor caso posible. Indicad en cada caso el valor de PDOP obtenido y los 4 satélites involucrados. En cada caso, marcar los satélites involucrados sobre el gráfico (como se ha hecho en la gráfica anterior) y adjuntad las gráficas correspondientes. ¿Coinciden con los dos casos presentados antes?

Ejercicio 3 (mostrar efecto de las diferentes correcciones al modelo):

Usando los datos de Yebes ilustraremos las mejoras al añadir al algoritmo de posicionamiento las correcciones explicadas en la hoja anterior. Basta ejecutar el programa del 1er ejercicio con las diferentes versiones de `get_pos()` para ver visualmente las mejoras de los resultados (en forma de una disminución de los errores). [En cada caso indicad la corrección añadida y volcad como antes la media y el máximo de las diferencias \(dE,dN\) obtenidas en cada caso.](#)

Mostrar gráficamente las mejoras superponiendo en una gráfica los resultados de las diferentes mejoras en el plano horizontal. Algo similar a la gráfica adjunta donde se muestran los resultados de una estación con el modelo básico (en verde), frente a ir añadiendo sucesivamente la corrección de los tiempos de vuelo (azul) y la corrección por el giro de la Tierra (magenta). [Adjuntar gráfica.](#)



Es importante que las diferentes correcciones/mejoras sean aplicadas de forma acumulativa, desde la más importante a la menos:

- 0) Versión básica (tiempo de vuelo fijo)
- 1) + corrección aplicando diferentes tiempos de vuelo a cada satélite.
- 2) + corrección por la rotación de la Tierra.
- 3) + corrección relativista.
- 4) + uso de pesos.

No tendría sentido aplicar p.e. la corrección relativista (del orden de 1-2 mt) si no se ha corregido también giro de la Tierra (con errores ~ 40-50 mt).

Si implementáis la resolución del problema usando pesos, utilizad la siguiente función de calidad basada en la elevación φ de cada satélite

$$Q = \frac{(1 - \cos(2\varphi))}{2}$$

Para depurar vuestra implementación de las correcciones, usad los ficheros de trazas (**traza1**, **traza2**, **traza3**, **traza4**) publicados en el LAB2. P.e. traza3 incluye correcciones de tiempos de vuelo τ 's + rotación Tierra + relativista.

Si implementáis el uso de pesos, unos datos donde se aprecia muy claramente la importancia de su uso son los de la estación **mer** (usad el mismo fichero **IGS13230.sp3**). En este caso hay varios satélites muy bajos sobre el horizonte cuyas medidas estropean los resultados finales si se usan sin pesos. Procesar los datos con vuestra versión sin pesos y luego con la nueva versión usando pesos. [Adjuntad gráficas de la dispersión de los resultados en superficie \(con respecto a la posición de referencia\) para ambos procesados, así como el error máximo del valor absoluto de los errores en E,N en cada caso.](#)

Ejercicio 4. Poor Man's DGPS:

Una forma alternativa de reducir los errores (en vez de usar un modelo cada vez más detallado) es hacer un procesado diferencial. Se trata de restar observaciones entre nuestro GPS y una estación de referencia (cuya posición es conocida) y plantear un modelo sobre la diferencia de observables. De esta forma los errores comunes (p.e. en la posición de los satélites o su error de reloj) se cancelarían. Si las estaciones están próximas (del orden de decenas de km) el error de nuestra posición puede bajar a ser del orden de 1 metro.

El problema es que manejar pseudorangos y calcular la posición a partir de ellos (lo que hemos aprendido en este curso) no es algo con lo que el usuario ocasional de un GPS esté familiarizado. Debido a esto mucha gente se ve tentada por una forma más sencilla de llevar a cabo este procesado diferencial.

Se trata de manejar, no los pseudorangos, sino solo las posiciones finales de los GPSs. Suponed que en un instante nuestro GPS calcula su posición (P1) y

el GPS de referencia la suya (P2). Nosotros no conocemos el error de nuestra posición, pero si el error del GPS de referencia, al conocer su verdadera posición $P_{ref} = (P2 - e)$. La idea es asumir que el error para nuestra posición será el mismo y corregir nuestra posición haciendo $(P1 - e)$.

Este enfoque es muy sencillo y no requiere usar pseudo-rangos. Se le suele llamar "Poor's man DGPS", indicando algo que no es ideal, pero que es lo único que podemos hacer si no tenemos acceso a los pseudorangos del GPS.

El problema es que este enfoque no suele funcionar muy bien. Tendríamos que estar seguros de que ambos GPS están aplicando el mismo algoritmo para que sus errores sean los mismos. Por ejemplo si uno de los GPSs está corrigiendo la rotación de la Tierra y el otro no, sus errores serían muy distintos.

Podríamos pensar que en nuestro caso si funcionaría, ya que podemos estar seguro de que usamos el mismo algoritmo el nuestro) al procesar ambos GPS.

Haremos una prueba usando los datos de otra estación (vill) como referencia. Esta estación (Villafranca del Castillo) está a unos 75 km de la de Yebes. Las observaciones de ambas estaciones han sido tomadas en (aproximadamente) los mismos instantes de tiempo.

El proceso es sencillo y muy similar al del 1er ejercicio. Se trata de hacer un bucle como entonces y en cada paso:

1. Resolver la posición P1 de Yebes usando cualquier algoritmo (no importa que sea uno de los "malos") sobre las medidas de **yebe**.
2. Resolver (con el MISMO algoritmo) la correspondiente posición P2 de la estación de referencia (Villafranca) usando las medidas tomadas en **vill**.
3. Con la posición correcta de Villafranca calcular error $e = (P2 - vill.XYZ)$.
4. Usando dicho error, corregir la posición de Yebes: $P1 = P1 - e$; y guardar la posición corregida en una matriz 3xNT de soluciones.

Terminado el bucle hacer lo mismo que hicimos en el ejercicio 1 convirtiendo los resultados a coordenadas planas (E,N) + altura (H). Obtener las diferencias (dE, dN, dH) usando la verdadera posición de Yebes (yebe.XYZ). [Adjuntad código de vuestro bucle, junto con las gráficas \(dN/dE\) + dH y los valores de CEP50 y CEP95 para los errores en superficie.](#)

Veréis que los resultados no son especialmente buenos a menos que estéis usando uno de los peores algoritmos. El problema es que aunque se use el mismo algoritmo, no hay garantías de que los satélites visibles desde ambas estaciones sean los mismos. El uso de algún satélite extra en una estación introduce un error que no existe en la otra, por lo que no se cancelarán.

Vamos a repetir el proceso pero asegurándonos de que resolvemos P1 y P2 con los **mismos satélites** en ambas estaciones. No hace falta cambiar mucho el código. En el viejo código lo primero que había que hacer es determinar las observaciones válidas (aquellas que fuesen positivas) para cada estación.

Ahora usaremos solo medidas que estén disponibles (que sean positivas) en AMBAS ESTACIONES SIMÚLTANEAMENTE. Cuando tengamos las observaciones comunes (obs1/obs2) a los satélites comunes (prn) resolvemos las posiciones P1 y P2 con solo esos datos, calculamos el error de la estación de referencia y corregimos P1 como antes. [Adjuntad modificaciones del código.](#)

[Adjuntad las gráficas \(dN/dE\) + dH junto con las desviaciones standard de estas diferencias para los nuevos resultados](#) [Dad los valores de CEP50/CEP95.](#)

Incluso con el peor algoritmo los resultados deben ser bastante buenos con la mayoría de los errores ~ 1 metro. A pesar de estos buenos resultados no hay que entusiasmarse. En la vida real nuestro GPS no sabe sepa qué satélites usa la estación referencia ni podríamos forzarle a usar los mismos. Es por esto que este enfoque del "Poor's man DGPS" no es muy útil en la práctica.

5. Efecto de la Disponibilidad selectiva y cambio de "datum" en mapas

Entre los datos cargados veréis una estructura ([alac](#)) con observaciones de la estación de Alicante (2005). Procesar sus datos (fichero SP3 [IGS13230.sp3](#)) con vuestro [mejor algoritmo](#). [Calculad la posición media, obtener su latitud y longitud y adjuntar la foto de la posición usando GoogleMaps \(vista 3D\).](#) Si el algoritmo es lo suficientemente preciso podréis identificar la pequeña torre donde se encuentra posicionada la antena. Haced la típica [gráfica mostrando la dispersión de los resultados en superficie](#) (dE,dN), dando los errores medios y máximos en las coordenadas (E,N), así como el valor para CEP95.

Procesar ahora las observaciones de [alac2](#) (SP3=[IGS10426.SP3](#)). Son de la misma estación, pero están tomadas en el año 2000, cuando todavía estaba activa la disponibilidad selectiva. [Adjuntad la nueva gráfica con la dispersión de los resultados en superficie](#) (dE,dN). [Volcad errores medios/máximos en \(E,N\) y el nuevo valor de CEP95. Comparadlos con los obtenidos antes \(datos de 2005, ya con la disponibilidad selectiva desactivada\).](#)

Extraer la posición correcta de la estación (del campo .XYZ) y convertirla a coordenadas planas E0,N0. Se trata de marcar dicha posición sobre un mapa de la zona ([alac.jpg](#)). Este mapa tiene una resolución de 2 metros/pixel y su esquina superior izquierda tiene como coordenadas E0=719000,N0=4248000. En este mapa la posición del GPS está marcada como "Mareógrafo de la Bocana". [Adjuntad mapa con la posición correcta superpuesta.](#) Veréis que no coinciden. El problema es que este mapa está calibrado en el antiguo "datum" que se usaba en España (ED50) y no en el datum actual (WGS84). [¿Cuál es el error \(en metros\) debido a usar un mapa con un datum incorrecto?](#)