Examen de Prácticas de Laboratorio de Cálculo

Grupo B: PL-07

Resolver los siguientes ejercicios utilizando `Matlab` *indicando* ***los comandos utilizados para resolverlos***, ***las respuestas obtenidas*** *y* ***las representaciones gráficas que se piden***.
Adjuntar un archivo PDF en la tarea del Campus Virtual creada para esta prueba, identificándolo con vuestro nombre y apellidos.

Corrección

1. **Dada la función:**

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{\frac{-x^2}{2}}$$

a) **(1 Punto) Calcular los límites:** $\lim_{x\to+\infty} f(x)$ **y** $\lim_{x\to-\infty} f(x)$**.**

Se introduce la función en `Matlab`, se muestra por pantalla utilizando el comando `pretty` (para comprobar que se ha informado a `Matlab` de la función correcta, ¡no faltan parténtesis!) y se ejecutan los comandos para calcular los límites pedidos:

```
syms x
f(x)=1/sqrt(2*pi)*exp(-x^2/2);
pretty(f)
limit(f(x),x,+Inf)
limit(f(x),x,-Inf)
```

Se obtiene:

```
   /    2 \
   |   x  |
exp| - -- | 7186705221432913
   \    2 /
----------------------------
      18014398509481984


ans =

0


ans =

0
```

Esto es, el valor de cada uno de los límites es 0.

b) **(1 Punto) Obtener los intervalos de monotonía (en dónde es creciente y decreciente), proporcionando los máximos y mínimos que tuviere (sean relativos o absolutos).**

Se calcula la derivada (primera) de la función y los números reales en los que se anula:

```
fprima(x)=diff(f,x);
double(solve(fprima))
```

```
ans =

     0
```

El dominio de la función es $\mathbb{R}$, puesto que el denominador que aparece en la definición de la función es una constante y no se anula en ningún número real; por tanto, no se debe añadir ningún otro *valor conflictivo* para estudiar la monotonía de la función. Evaluamos la función en cada una de las regiones de los números reales que divide 0; se toman (por ejemplo) los valores -1 y 1. Como las expresiones simbólicas de las imágenes son tediosas por los números que aparecen, convertimos dichas expresiones en formato numérico.

```
double(fprima([-1,1]))
```

```
ans =

    0.2420   -0.2420
```

Por todo lo obtenido, se deduce que la función es creciente ($f'(x) \geq 0$) sobre el intervalo $(-\infty, 0)$ y decreciente ($f'(x) \leq 0$) sobre el intervalo $(0, +\infty)$. Así, la función posee un máximo en $x = 0$.

c) **(1 Punto) Obtener los intervalos de curvatura (en dónde es cóncava y convexa), proporcionando los puntos de inflexión que tuviere.**

Se calcula la derivada segunda de la función y los números reales en los que se anula:

```
fsegunda(x)=diff(f,x,2);
double(solve(fsegunda))
```

```
ans =

-1
 1
```

Se calculan las imágenes de la función en cada una de las tres regiones delimitadas por los valores obtenidos (por ejemplo, en -2, 0 y 2).

```
fsegunda([-2,0,2])
```

```
ans =

    0.1620   -0.3989    0.1620
```

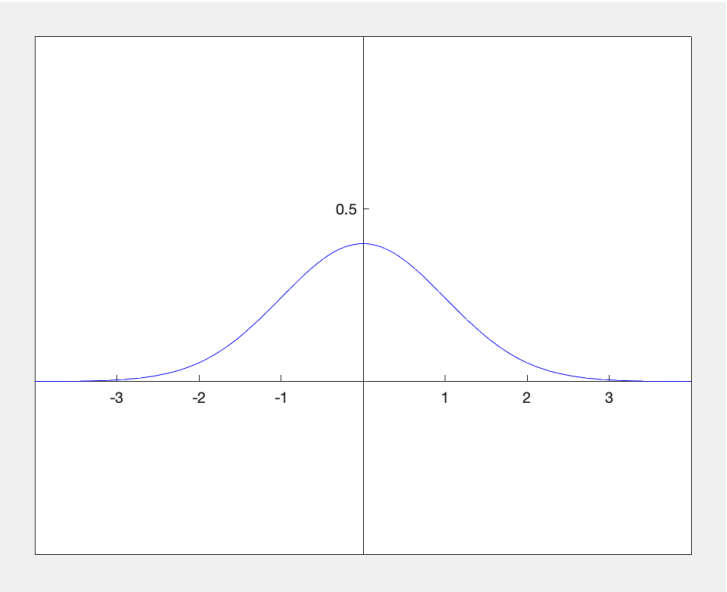
La función es cóncava (convexa para Set) en los intervalos $(-\infty, -1) \cup (1, +\infty)$ y convexa (cóncava para Set) en el intervalo $(-1, 1)$. Por tanto, la función posee dos puntos de inflexión situados en $x = -1$ y $x = 1$.

d) **(1 Punto) Representar la función sobre el intervalo $[-4, 4]$ en color azul, con los valores de las imágenes comprendidas entre $[-0.5, 1]$ y verificar los resultados obtenidos analíticamente.**

Se ejecuta el comando `fplot` con los parámetros apropiados para representarla sobre el intervalo de abscisas exigido por el enunciado y en color azul. Además, se colocan los ejes de coordenadas de manera que se corten en el origen $(0, 0)$ y se define también el rango de las imágenes en el intervalo $[-0.5, 1]$:

```
fplot(f,[-4,4],'b')
ax=gca;
ax.XAxisLocation='Origin';
ax.YAxisLocation='Origin';
axis([-4,4,-0.5,1])
```

Se muestra el resultado obtenido:



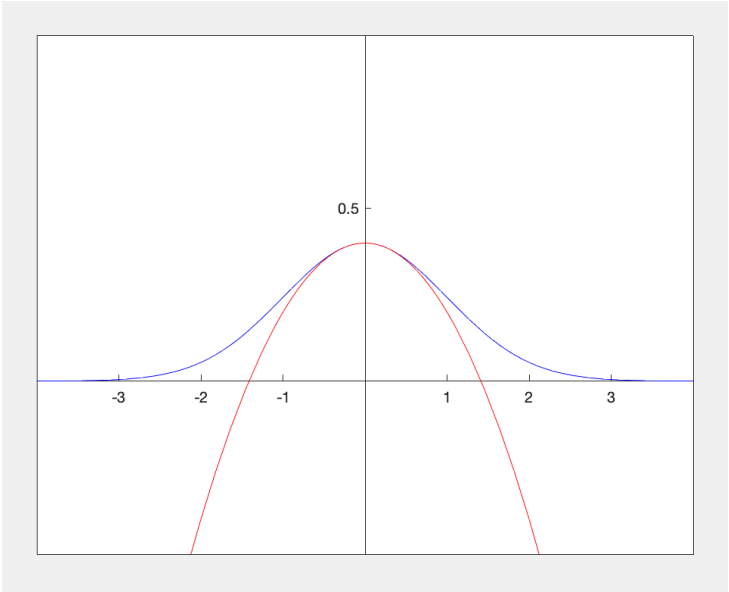
Todo cuadra con los resultados obtenidos en los apartados anteriores.

2. *a*) **(1 Punto) Calcular el *Polinomio de Taylor* de grado 2 asociado a la función alrededor del punto $x = 0$. No es necesario adjuntar la expresión del polinomio. Representar el *Polinomio de Taylor* de grado 2 en color rojo, junto con la función, sobre el intervalo $[-4, 4]$ con los valores de las imágenes comprendidas entre $[-0.5, 1]$.**

Se pide a `Matlab` el *Polinomio de Taylor* con los parámetros adecuados para hacerlo: alrededor del punto $x = 0$ y con grado $n = 2$, en general, tendrá tres sumandos (de ahí 'order', 3). Se mantiene la representación de la función (lo exige el ejercicio) y se representa el polinomio en color rojo.

```
p2(x)=taylor(f,x,0,'order',3);
hold on
fplot(p2,[-4,4],'r')
```

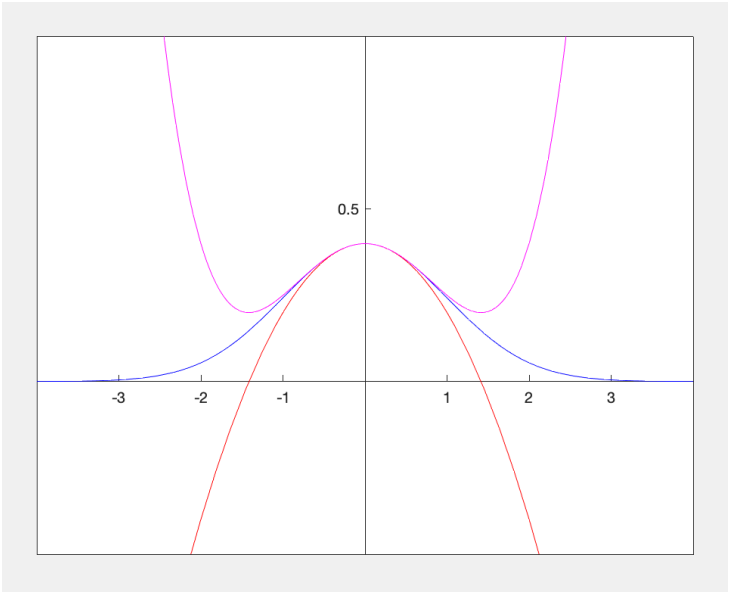
Véase la figura que se obtiene:



b) **(1 Punto) Calcular el *Polinomio de Taylor* de grado 4 asociado a la función alrededor del punto $x = 0$. No es necesario adjuntar la expresión del polinomio. Representar el *Polinomio de Taylor* de grado 4 en color magenta, junto con la función y el polinomio construido en el apartado *a*), sobre el intervalo $[-4, 4]$, con los valores de las imágenes comprendidas entre $[-0.5, 1]$.**

Se construye el *Polinomio de Taylor* de grado $n = 4$ que, en general, tiene cinco sumandos (de ahí 'order', 5), ya están mantenidas las representaciones anteriores e indicamos a `Matlab` la representación gráfica del nuevo polinomio.

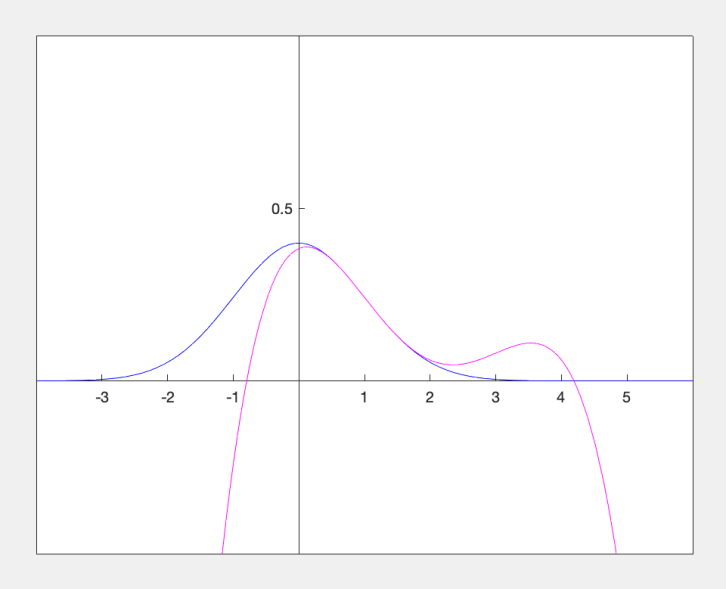
```
p4(x)=taylor(f,x,0,'order',5);
fplot(p4,[-4,4],'m')
```



c) **(1 Punto) Calcular el *Polinomio de Taylor* de grado 4 asociado a la función alrededor del punto $x = 1$. Representar, exclusivamente, este polinomio junto con la función sobre el intervalo $[-4, 6]$, con los valores de las imágenes comprendidas entre $[-0.5, 1]$.**

Dado que la representación gráfica debe contener, **exclusivamente**, lo que se pide, antes de nada se elimina todo lo dibujado previamente. Se repinta la función, se recolocan los ejes en el origen, se especifican los rangos de las abscisas y las ordenadas como se exige en el enunciado, se construye el *Polinomio de Taylor* pedido (llamado `p41(x)`) y, finalmente, se representa `p41(x)` junto con la función. Se han escogido los colores azul para la función (como siempre) y magenta para el polinomio.

```
hold off
fplot(f,[-4,6],'b')
ax=gca;
ax.XAxisLocation='Origin';
ax.YAxisLocation='Origin';
axis([-4,6,-0.5,1])
hold on
p41(x)=taylor(f,x,1,'order',5);
fplot(p41,[-4,6],'m')
```



3. **(1 Punto) Calcular el área que encierra la función sobre el eje de abscisas.**

Para calcular el área que encierra la función sobre el eje de abscisas $(-\infty, +\infty)$, sólo es necesario el siguiente comando:

```
area1=int(f,x,-Inf,+Inf)
```

Y se obtiene:

```
ans =

     1
```

4. **(2 Puntos) Calcular el área que encierra el *Polinomio de Taylor* de grado 4 calculado en el apartado *c*) del ejercicio 2) sobre el eje de abscisas.**

   Deben calcularse los puntos de corte del *Polinomio de Taylor* de grado $n = 4$ calculado en el apartado 2.*c*) (`p41(x)`) con el eje de abscisas.

```
double(solve(p41))
ans =

  -0.7955 + 0.0000i
   2.3024 - 0.6312i
   2.3024 + 0.6312i
   4.1907 + 0.0000i
```

   Teniendo en cuenta que las soluciones complejas no tienen interés para el problema en cuestión, los cortes con el eje de abscisas se producen en $x = -0.7955$ y en $x = 4.1907$ (coherente con la representación gráfica). Estos valores se convierten en los límites de integración del último comando que debe introducirse para finalizar el examen.

```
area2=double(int(p41,x,-0.7955,4.1907))

ans =

    0.8363
```