donde x e y son vectores, el tercer argumento indica características especiales (**opciones**) que tendrá el dibujo. En este caso particular, los pares (x_i, y_i) son puntos de una recta y el gráfico se hará con color rojo.

Observación 1 *Octave no grafica funciones, grafica pares de puntos y los une con segmentos de recta.*

Al ejecutar *lineal* se observa una pequeña pausa entre el primer gráfico y los siguientes. Esto se obtene usando la instrucción:

```
sleep(3);
```

El comando:

```
for a=a0+deltaa:deltaa:a1;
```

se utiliza para realizar algo un conjunto de acciones varias veces, en cada una de ellas *a* tomará los valores del rango **a0+deltaa:deltaa:a1**.

La sintaxis del comando **for** es:

```
for variable=rango
   acciones
end
```

Observando detenidamente los gráficos generados por la función *lineal* se ve que la escala en el eje de las ordenadas cambia a medida que se dibujan nuevas rectas. Esto se debe a que *Octave* elige los intervalos de graficación en forma automática. Para fijar el área del gráfico se debe usar la función **axis**.

En *lineal.m* se tiene:

```
% axis([-3,3,-3,3]);
```

el % le dice a *Octave* que la línea no debe ser considerada. Las líneas que comienzan con % son llamadas comentarios. La sintaxis del comando axis es:

```
axis([xmin,xmax,ymin,ymax,zmin,zmax]);
```

Cada vez que se ejecuta *lineal*, se obtiene el mismo resultado. Para cambiar el resultado hay que modificar el archivo antes de ejecutarlo nuevamente. Para facilitar la interacción entre el usuario y el **script** *Octave* permite modificar algún parámetro mientras el script se está ejecutando. La línea:

```
% a0=input("a inicial ==> ");
```

muestra el mensaje *a inicial* ==> en la pantalla y queda a la espera de una respuesta del usuario, el valor ingresado es almacenado en la variable $a0$ para ser utilizada cuando se necesite.

Ejercicio 1 *Usando como ejemplo **lineal.m** explicar el efecto que produce variar b en la ecuación de la recta.*

Ejercicio 2 *Proponer otra forma de abordar el problema planteado.*

1.5. Segunda situación problemática.

Con lo aprendido hasta el momento, **¿Como se puede marcar una región encerrada entre dos curvas?**

En el método traicional, para graficar la región entre dos curvas, se dibujan las dos curvas y a continuación se trazan segmentos de recta que unen puntos de las dos curvas.

Como ejemplo se usarán las curvas definidas por el gráfico de $f(x) = sin(x)$ y el de $g(x) = 0$. Las funciones $f(x)$ y $g(x)$ están definidas en todo R, pero para hacer el gráfico se tomará $x \in [-6,6]$.

En *region.m* se muestra el resultado de llevar a la computadora el método tradicional.

region

```
xmax=6;
xmin=-6;
nx=20;
x=[xmin:(xmax-xmin)/nx:xmax];
y1=sin(x);
y2=0*x;
figure(1)
hold on;
plot(x,y1,'r')
plot(x,y2,'k')
axis([min(x),max(x),min([y1,y2]),max([y1,y2])]);
for i=1:columns(x)
        plot([x(i),x(i)],[y1(i),y2(i)],'b');
        sleep(1);
end
hold off;
```

Para destacar, se pueden hacer operaciones al definir un rango.

```
x=[xmin:(xmax-xmin)/nx:xmax];
```

Los argumentos de las funciones de *Octave* pueden ser escalares o vectores.

```
y1=sin(x);
```

No es necesario calcular los vectores para después usarlos en una función.

```
axis([min(x),max(x),min([y1,y2]),max([y1,y2])]);
```

Ejercicio 3 *Modificar region.m para que las líneas estén más juntas.*

Ejercicio 4 *Modificar region.m para pintar solo cuando $f(x)$ es mayor que $g(x)$.*

La solución al problema consiste en dibujar solo los segmentos que cumplan la condición $f(x) > g(x)$.

region1

```
xmax=6;
xmin=-6;
nx=100;
x=[xmin:(xmax-xmin)/nx:xmax];
y1=sin(x);
y2=0*x;
figure(1)
hold on;
plot(x,y1,'r')
plot(x,y2,'k')
axis([min(x),max(x),min([y1,y2]),max([y1,y2])]);
for i=1:columns(x)
        if (y1(i)>y2(i))
          plot([x(i),x(i)],[y1(i),y2(i)],'b');
        endif
end
hold off;
```

La sintaxis de **if** es:

```
if (condicion)
  acciones
elseif (condicion)
  acciones
else
  acciones
endif
```

Otra opción para resolver la segunda situación problemática, obliga a profundizar el estudio de los comandos de *Octave*.

region2

```
xmax=6;
xmin=-6;
nx=60;
x=[xmin:(xmax-xmin)/nx:xmax];
x2=[xmax:-(xmax-xmin)/nx:xmin];
y1=sin(x);
y2=cos(x2);
figure(1)
fill([x,x2],[y1,y2],'y')
```

Las opciones de **fill** son las mismas que las de **plot**.

Ejercicio 5 *Graficar el complento de la región sombreada de la figura 2*

Ejercicio 6 *Idem al ejercicio 4 pero usando el comando fill. ¿Que dificultades aparecen en este problema?*

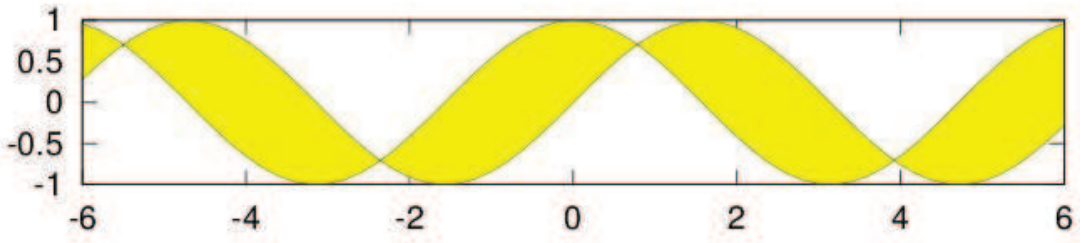

Figura 2: Grafico generado por *region2*