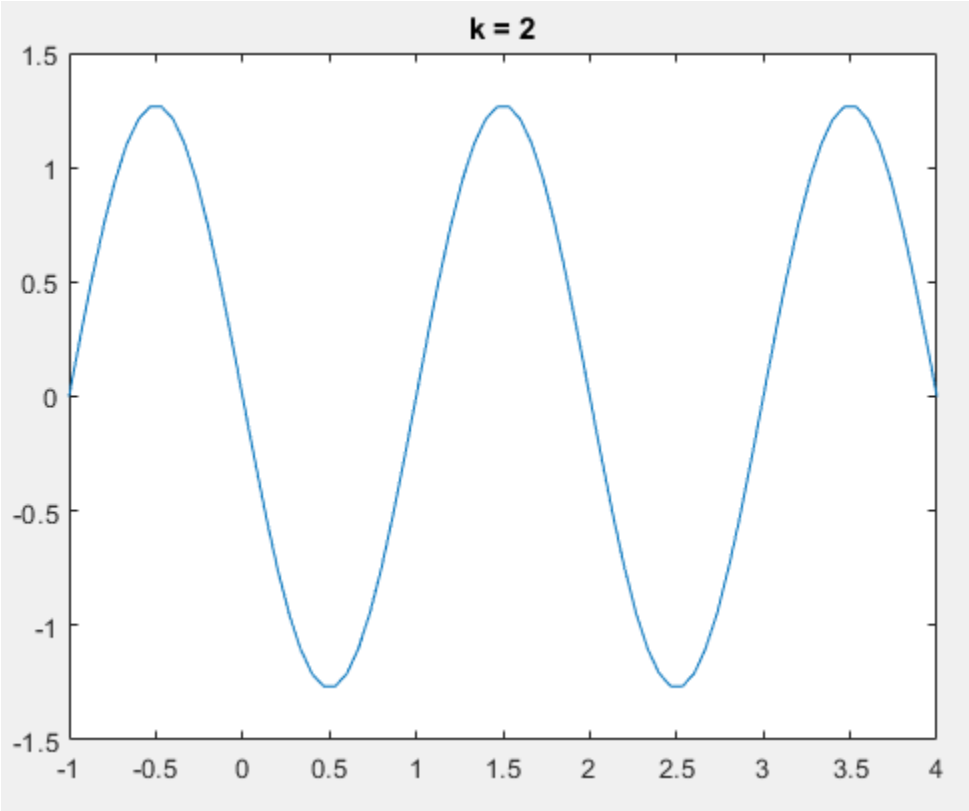
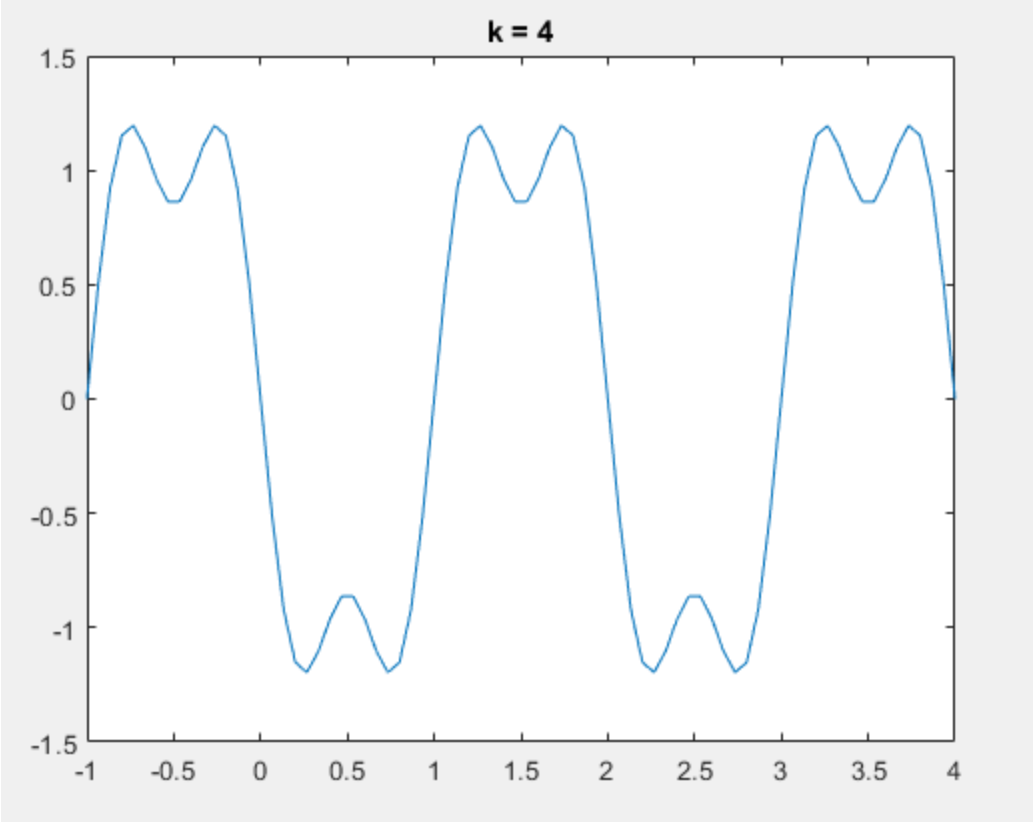
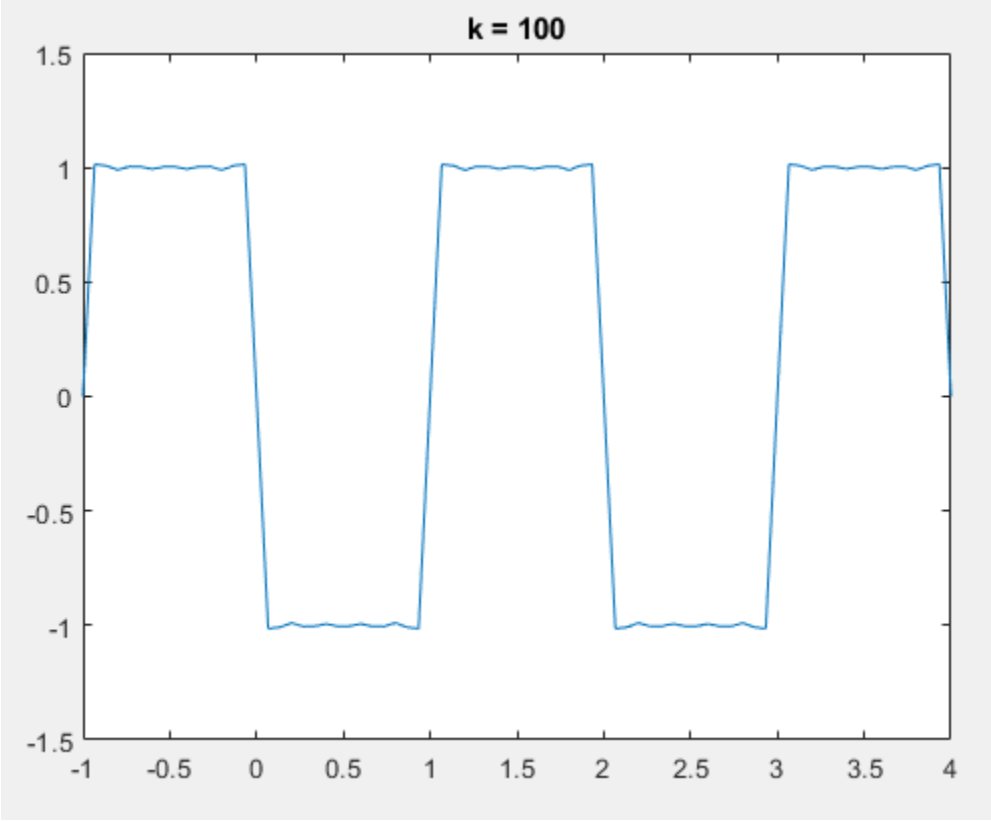
Part 1 a

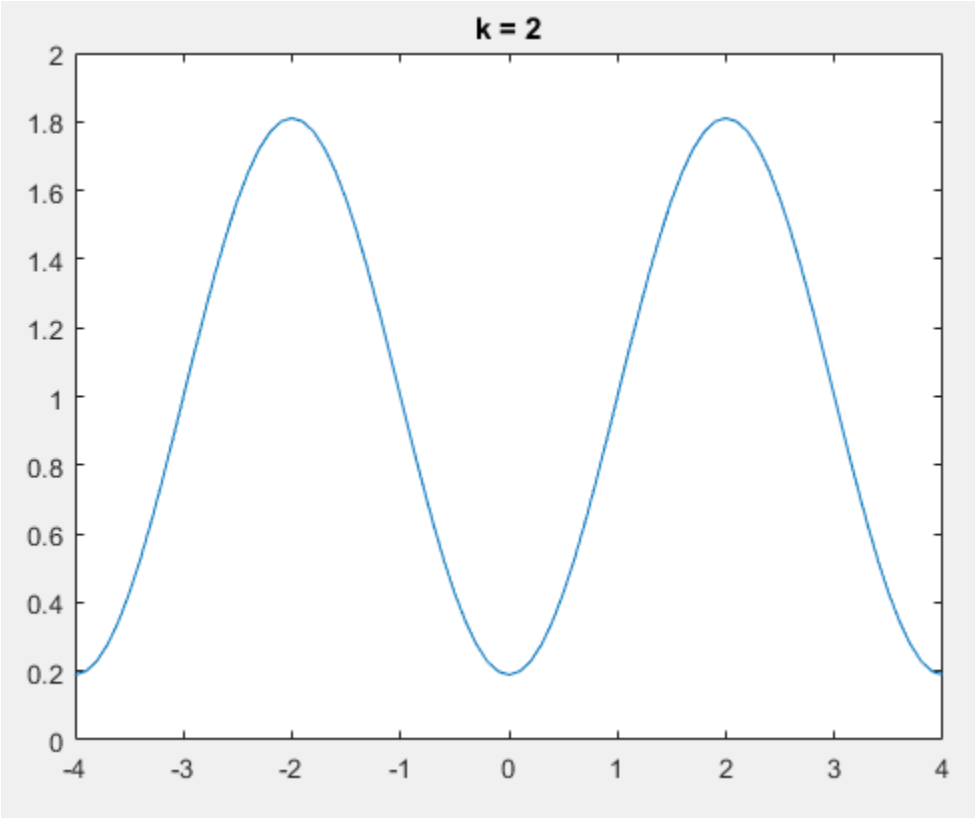
As k increases it starts to look more like the square wave function.

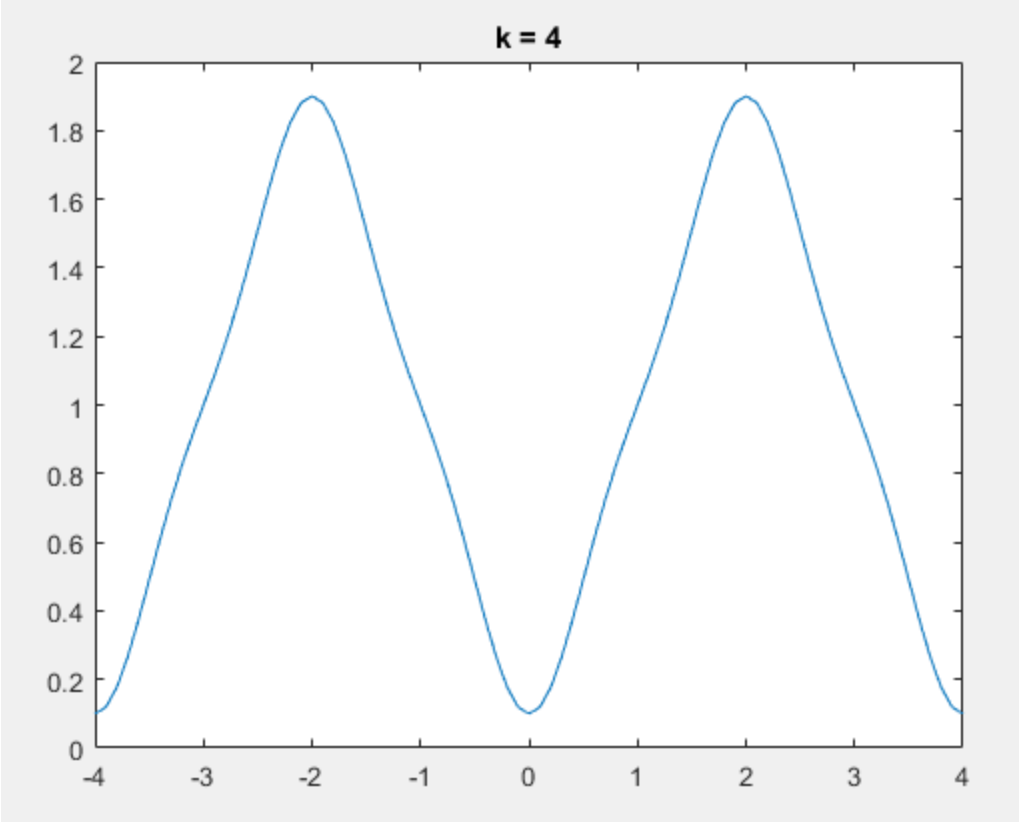


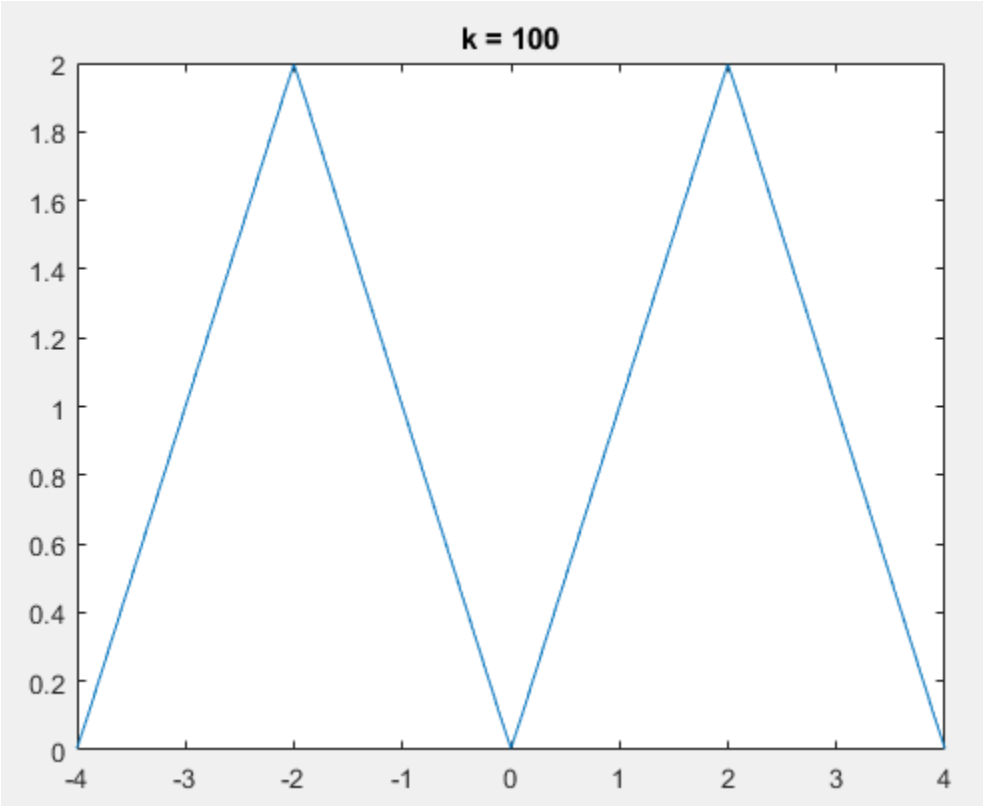

k = 4
1.5
1
0.5
0
-0.5
-1
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
3
3.5
4


k = 100
1.5
1
0.5
0
-0.5
-1
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
3
3.5
4

Part 1 b)
This function seems to converge faster.







Code, including the code for part 2(sound_fcn):

```
%a0 = 0
%ak = (e^j*pi*k+ e^-j*pi*k - 2)/(2*j*pi*k)
%   if k odd -> -2/(j*pi*k)
%   if k even ->0
function [] = graph_func_test()
   graph_square_wave(2);
   graph_square_wave(4);
   graph_square_wave(100);
   graph_triangle_wave(2);
   graph_triangle_wave(4);
   graph_triangle_wave(100);
   sound_fcn();
end

function [] = sound_fcn()
   signal1 = @(t) cos(2*pi*220*2^(10/12)*t*8);
   signal2 = @(t) cos(2*pi*220*2^(6/12)*t*2);
   signal3 = @(t) cos(2*pi*220*2^(8/12)*t*8);
   signal4 = @(t) cos(2*pi*220*2^(5/12)*t*2);

   fs = 8000;
   n1 = 2;
   t8 = 1/fs: 1/fs:n1/8;
   disp(t8(1));
   t2 = 1/fs: 1/fs:n1/2;
   sd = zeros(1,round(length(t8)/10));
   rest = zeros(1,length(t8));

   t1 = 1/fs: 1/fs:n1;
   signal1_arr = arrayfun(signal1, t1);
   signal2_arr = arrayfun(signal2, t1);
   signal3_arr = arrayfun(signal3, t1);
   signal4_arr = arrayfun(signal4, t1);

   final_signal = [signal1_arr, sd, signal1_arr, sd,signal1_arr, sd, signal2_arr, sd, rest, sd,
signal3_arr, sd,signal3_arr, sd,signal3_arr, sd, signal4_arr];
   sound(final_signal);

end
function [] = graph_triangle_wave(k_range)
   T0 = 4;
   function [ak] = ak_fcn(k)
      if k == 0
         ak = 1;
```

```
        else
            ak = (1-(-1)^k)/(-2*pi^2*k^2);
        end
    end

    aks = zeros(1, 2*k_range + 1);

    %start
    k = -(length(aks) - 1) / 2;
    for n = 1 : 2*k_range +1
        if n ~= k_range + 1
            aks(n) = ak_fcn(k);
        end
        k = k + 1;
    end

    function [x] = compute_x(aks, t, T0)
        x = 1;
        k = -(length(aks) - 1) / 2;
        for n = 1 : length(aks)
            x = x + aks(n) * exp(j*k*2*pi*t/T0);
            k = k + 1;
        end
    end
    aks = 4*aks;

    t_start = -4;
    t_end = 4;

    T_sampling = T0/40;
    t = t_start:T_sampling:t_end;

    plot(t, arrayfun(@(x)compute_x(aks, x, T0), t));
    title("k = " + k_range);

end
function [] = graph_square_wave_2(k_range)
    T0 = 2;
    function [ak] = ak_fcn(k)
        if k == 0
            ak = 1/2;
        elseif mod(k, 2) == 0
            ak = 0;
        else
```

```
        ak = 1/(j*pi*k);
    end
end

aks = zeros(1, 2*k_range + 1);

%start
k = -(length(aks) - 1) / 2;
for n = 1 : 2*k_range +1
    if n ~= k_range + 1
        aks(n) = ak_fcn(k);
    end
    k = k + 1;
end

function [x] = compute_x(aks, t, T0)
    x = 0;
    k = -(length(aks) - 1) / 2;
    for n = 1 : length(aks)
        disp(k);

        x = x + aks(n) * exp(j*k*2*pi*t/T0);
        k = k + 1;
    end
end

t_start = -1;
t_end = 4;

T_sampling = T0/30;
t = t_start:T_sampling:t_end;

plot(t, arrayfun(@(x)compute_x(aks, x, T0), t));
title("k = " + k_range);

end

%    function [ak] = ak_fcn(k)
%        if mod(k, 2) == 0
%            ak = -2/(j*pi*k);
%        else
%            ak = 0
%        end
%        %ak = (exp(j*pi*k) + exp(-j*pi*k) -2) / (2*j*pi*k);
```

```
%     end

function [] = graph_square_wave(k_range)
    T0 = 2;
    function [ak] = ak_fcn(k)
        ak = (exp(j*pi*k) + exp(-j*pi*k) -2) / (2*j*pi*k);
    end

    aks = zeros(1, 2*k_range + 1);

    %start
    k = -(length(aks) - 1) / 2;
    for n = 1 : 2*k_range +1
        if n ~= k_range + 1
            aks(n) = ak_fcn(k);
        end
        k = k + 1;
    end

    function [x] = compute_x(aks, t, T0)
        x = 0;
        k = -(length(aks) - 1) / 2;
        for n = 1 : length(aks)
            disp(k);

            x = x + aks(n) * exp(j*k*2*pi*t/T0);
            k = k + 1;
        end
    end

    t_start = -1;
    t_end = 4;

    T_sampling = T0/30;
    t = t_start:T_sampling:t_end;

    plot(t, arrayfun(@(x)compute_x(aks, x, T0), t));
    title("k = " + k_range);

end
```