Task 3d

```
clear, clf
N = 2e6;
a = 1.4;
b = 0.3;
transient_length = 1e3;
epsilon_min = 1e-3;
epsilon_max = 2*1e-2;
epsilon_range = logspace(log10(epsilon_min), log10(epsilon_max),
 1000);
q_range = 0:0.15:4;
D_q = zeros(length(q_range));
q_data = zeros(length(q_range), length(epsilon_range));

x0 = 0; y0 = 0;
[x_data,y_data] = get_henon_data(x0, y0, transient_length, a, b, N);
x_min = min(x_data);x_max = max(x_data);
y_min = min(y_data);y_max = max(y_data);

plot0 = zeros(1,length(epsilon_range));
plot1 = zeros(1,length(epsilon_range));
plot2 = zeros(1,length(epsilon_range));

for i_epsilon = 1:length(epsilon_range)
    epsilon = epsilon_range(i_epsilon);
    P = zeros(ceil((x_max - x_min)/epsilon), ceil((y_max-y_min)/
epsilon));

    for i_point = 1:length(x_data)
        x = x_data(i_point);
        y = y_data(i_point);
        P_x_index = floor((x - x_min)/epsilon)+1;
        P_y_index = floor((y - y_min)/epsilon)+1;
        P(P_x_index, P_y_index) = P(P_x_index, P_y_index) + 1;
    end

    P = P/sum(sum(P));
    P_nonzero = P(P~=0);

    for i_q = 1:length(q_range)
        q = q_range(i_q);
        q_data(i_q, i_epsilon) = 1/(1-q)*log(get_Iq(P_nonzero,q));
    end
end

for i_q = 1:length(q_range)
    q = q_range(i_q);
    p = polyfit(log(1./epsilon_range), q_data(i_q,:), 1);
    D_q(i_q) = p(1);
end
```

```
clf
plot(q_range, D_q, '.', 'MarkerSize', 25)
set(gca, 'FontSize', 20)
xlabel('$q$', 'interpreter', 'latex')
ylabel('$D_q$', 'interpreter', 'latex')
title('$D_q$ against $q$. For each calculation, 1000 logarithimically
 spaced $\epsilon \in [10^{-3}, 2\cdot 10^{-2}]$ were
 used.', 'interpreter', 'latex')
ylim([1, 1.25])


function Iq = get_Iq(P_nonzero, q)

    Iq = sum(sum(P_nonzero.^q));
end

function [x,y] = get_henon_data(x0, y0, transient_length, a, b, N)
    [x0, y0] = discard_initial_transient(x0, y0, transient_length, a,
 b);

    x = zeros(1, N);
    y = zeros(1, N);

    x(1) = x0;
    y(1) = y0;

    for n = 1:N-1
        x(n+1) = y(n) + 1 - a*x(n)^2;
        y(n+1) = b*x(n);
    end
end

function [x0, y0] = discard_initial_transient(x0, y0,
 transient_length, a, b)
    x = zeros(1, transient_length);
    y = zeros(1, transient_length);

    x(1) = x0;
    y(1) = y0;

    for n = 1:transient_length-1
        x(n+1) = y(n) + 1 - a*x(n)^2;
        y(n+1) = b*x(n);
    end

    x0 = x(transient_length);
    y0 = y(transient_length);
end
```

Published with MATLAB® R2020a