Detector Modeling Project

IMGS-442 Imaging System Analysis Modeling, Dr. Hailstone

Jared Gregor

In this project, computer simulations are used to help better understand the performance limitations of electronic imaging devices. To do this, the theory surrounding how photons interact with detector arrays must be well understood. When photons are treated as individual particles, the photon/ detector array problem simply becomes a question of probability. Using poisson statistics, photon detection can be quantified and modeled. Noise can also be well taken into account using the set of provided equations.

Using MATLAB, equations can be used to model detector quantum efficiency, DQE. The DQE calculation takes into account different types of noise including, A-D conversion noise and read noise. DQE can be plotted against the number of incident photons to understand the effects of exposure on the detector array. This can be repeated varying the different types of noise to understand the individual effects of each.

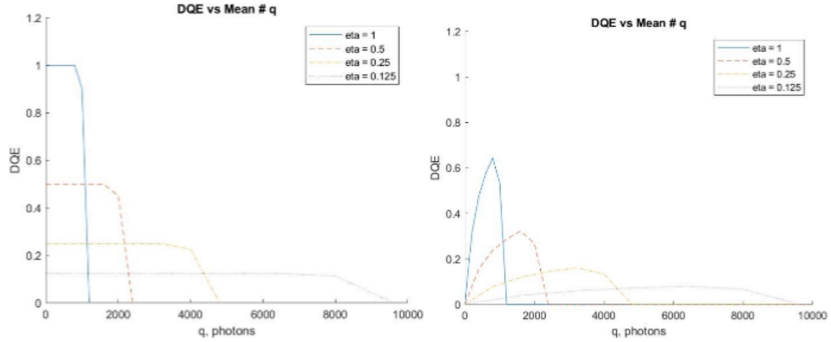

Fig 1. DQE vs Mean # q without noise (left) with read noise = 10 e and 4-bit A-D (right).

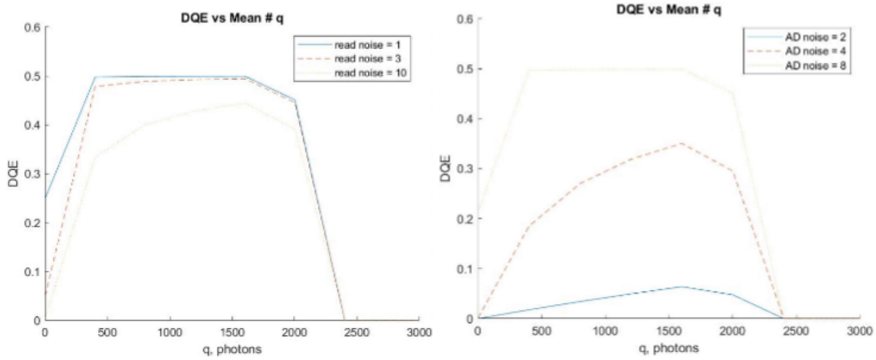

Fig 2. DQE vs Mean # q varying read noise (left) and varying A-D noise (right).

At this point I have fully working MATLAB code for task 1 and 2. I wrote DQE into a function with noise parameters so that it can be reused between sections. I also already started laying out my paper in LaTeX so I feel like I am at a pretty good spot.

```
%% Imaging System Analysis Modeling
% Project: Detector Modeling
% Author: Jared Gregor (jmg2586@rit.edu)
% Date: Oct 5 2020

%% Task 1
clear

% Given Variables
L = 1024; % Saturation level
eta = [1.0, 0.5, 0.25, 0.125];
q = [1:200:8001]; % Lambda/ Mean exposure
q_scaled = [q;q;q;q]; % Scale q for each eta
q_scaled = q_scaled ./ eta';


noise_read = 0; % Electrons
noise_AD = 0;

% Calculate dqe with 0 noise
dqe(1,:) = DQE(L, q, eta(1), noise_AD, noise_read);
dqe(2,:) = DQE(L, q, eta(2), noise_AD, noise_read);
dqe(3,:) = DQE(L, q, eta(3), noise_AD, noise_read);
dqe(4,:) = DQE(L, q, eta(4), noise_AD, noise_read);

% Plot DQE vs q
Plot1(q_scaled, dqe)

% Add noise
noise_read = 10; % Electrons
AD = 4; % bits
noise_AD = NoiseAD(L, AD);

% Calculate dqe with constant read and AD noise
dqe(1,:) = DQE(L, q, eta(1), noise_AD, noise_read);
dqe(2,:) = DQE(L, q, eta(2), noise_AD, noise_read);
dqe(3,:) = DQE(L, q, eta(3), noise_AD, noise_read);
dqe(4,:) = DQE(L, q, eta(4), noise_AD, noise_read);

% Plot DQE vs q with noise
Plot1(q_scaled, dqe)

%% Task 2
clear

% Given Variables
L = 1024; % Saturation level
eta = 0.5;
q = [1:200:3001]; % Lambda/ Mean exposure
q_scaled = q ./ eta;
```

```
% DQE varying read noise
noise_read = [1,3,10];
noise_AD = 0;

% Calculate DQE varying read noise and 0 AD noise
dqe(1,:) = DQE(L, q, eta, noise_AD, noise_read(1));
dqe(2,:) = DQE(L, q, eta, noise_AD, noise_read(2));
dqe(3,:) = DQE(L, q, eta, noise_AD, noise_read(3));

Plot2(q_scaled, dqe,["read noise = 1", "read noise = 3", "read noise = 10"])

% DQE varying bit level
noise_read = 0;
AD = [2,4,8];
noise_AD = NoiseAD(L, AD);

% Calculate DQE varying AD noise and 0 read noise
dqe(1,:) = DQE(L, q, eta, noise_AD(1), noise_read);
dqe(2,:) = DQE(L, q, eta, noise_AD(2), noise_read);
dqe(3,:) = DQE(L, q, eta, noise_AD(3), noise_read);

Plot2(q_scaled, dqe,["AD noise = 2", "AD noise = 4", "AD noise = 8"])

%% Task 3
noise_read = 0;
noise_AD = 0;

diameter = [5,10,20]; %Square Pixel diameter microns
multiplier = [16, 4, 2]

E = multiplier .* q;
logE = log(E);

%% Task 4
clear

%% Task 5
clear

%% Functions

% AD noise
function noise = NoiseAD(L, AD)
    noise = (L.^2)./(12.*2.^(2.*AD));
end

% DQE
function dqe = DQE(L, q, eta, noise_AD, noise_read)

    % Compute f1 (eq 6)
    f1 = 0;
```

```
    for i = 0:L-1
        f1 = f1 + (1/L) * poisscdf(i,q);
    end

    % Compute f2 (eq 9)
    f2 = (1/L) * poisscdf(L-1,q);

    % Compute f3 (eq 16)
    f3 = 0;
    for i = 0:L-1
        f3 = f3 + ((1/(L*L)) * (2*i+1) * poisscdf(i,q));
    end


    % Calculate DQE(qN) (Eq 21)
    DQE_qN = (q.*f2.*f2.*L.^2)./(noise_AD + noise_read^2 + L.^2 *((1.-f3)-(1.-f1).^2));

    % Calculate DQE(q) (Eq 24)
    dqe = DQE_qN .* eta;
end

function Plot1(q_scaled, dqe)
    figure
    hold on
    plot(q_scaled(1,:), dqe(1,:), '-')
    plot(q_scaled(2,:), dqe(2,:), '--')
    plot(q_scaled(3,:), dqe(3,:), '-.')
    plot(q_scaled(4,:), dqe(4,:), ':')
    xlim([0 10000])
    ylim([0 1.2])
    legend('eta = 1', 'eta = 0.5','eta = 0.25','eta = 0.125')
    title('DQE vs Mean # q')
    ylabel('DQE')
    xlabel('q, photons')
    hold off
end

function Plot2(q_scaled, dqe, leg)
    figure
    hold on
    plot(q_scaled, dqe(1,:), '-')
    plot(q_scaled, dqe(2,:), '--')
    plot(q_scaled, dqe(3,:), ':')
    xlim([0 3000])
    ylim([0 0.6])
    legend(leg(1), leg(2), leg(3))
    title('DQE vs Mean # q')
    ylabel('DQE')
    xlabel('q, photons')
    hold off
end
```

```
function Plot3(q_scaled, dqe, leg)
    figure
    hold on
    plot(q_scaled, dqe(1,:), '-')
    plot(q_scaled, dqe(2,:), '--')
    plot(q_scaled, dqe(3,:), ':')
    xlim([0 3000])
    ylim([0 0.6])
    legend(leg(1), leg(2), leg(3))
    title('DQE vs Mean # q')
    ylabel('DQE')
    xlabel('q, photons')
    hold off
end

function Plot4(q_scaled, variance, leg)
    figure
    hold on
    plot(q_scaled, variance(1,:), '-')
    plot(q_scaled, variance(2,:), '--')
    plot(q_scaled, variance(3,:), ':')
    xlim([0 3000])
    ylim([0 0.6])
    legend(leg(1), leg(2), leg(3))
    title('Variance vs normalized mean count level')
    ylabel('DQE')
    xlabel('q, photons')
    hold off
end

function Plot5(q_scaled, dqe, leg)
    figure
    hold on
    plot(q_scaled, dqe(1,:), '-')
    plot(q_scaled, dqe(2,:), '--')
    plot(q_scaled, dqe(3,:), ':')
    xlim([0 3000])
    ylim([0 0.6])
    legend(leg(1), leg(2), leg(3))
    title('DQE vs log(mean # photons / 400 microns^2)')
    ylabel('DQE')
    xlabel('log E, photons/400 micron^2')
    hold off
end
```