Exercise 7.1

a) Impulse response for IRR filter

function ran = random0(n, m)(1,3)

```
>> random0(4)
positive integer for seed: 1024

ans =

    0.0598    1.0000    0.0905    0.2060
    0.2620    0.6432    0.6325    0.8392
    0.6278    0.5463    0.7551    0.4554
    0.3177    0.9105    0.1289    0.6230

>> random0(4)
Enter a positive integer for seed: 32

ans =

    0.1441    0.1992    0.8157    0.8275
    0.6807    0.8620    0.3278    0.9322
    0.8562    0.0354    0.5895    0.2502
    0.7901    0.4512    0.3735    0.0233

>> random0(4)
Enter a positive integer for seed: 1024

ans =

    0.0598    1.0000    0.0905    0.2060
    0.2620    0.6432    0.6325    0.8392
    0.6278    0.5463    0.7551    0.4554
    0.3177    0.9105    0.1289    0.6230

>> random0(2, 3)
Enter a positive integer for seed: 32

ans =

    0.1441    0.1992    0.8157
    0.8275    0.6807    0.8620
```

b) IRR filters

The given formula generates random numbers that are almost
% equally distributed between 0 and 1.

%
>> a1 = random0(1, 20000);
Enter a positive integer for seed: 1024
>> mean(a1)

ans =

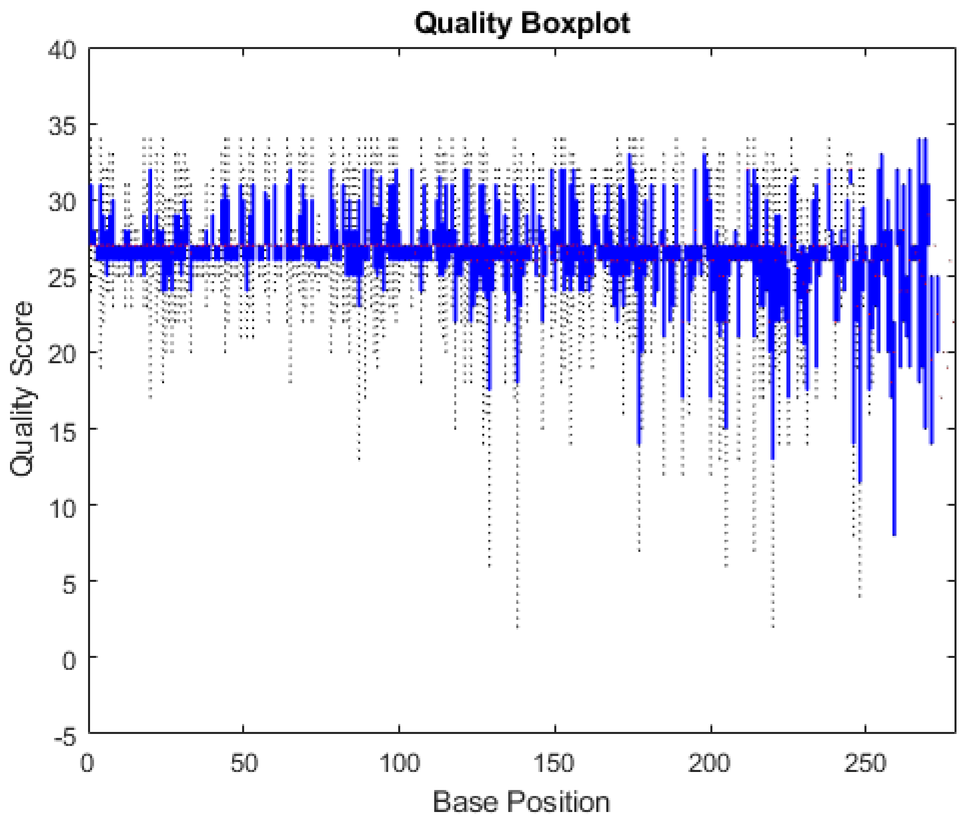
0.5020

>>std(a1)

ans =

0.2881

>>hist(a1, 10);



Exercise 7.2

a) **Nuling FIR**
B. xm = 0:50:50*50; % 50 points from 0 to 2450, in steps of 50

[X,Y] = meshgrid(xm,xm);; % use xm twice for symmetric grid.

Z = zeros(50); % blank elevation data

for m = 1:numel(X); % for each index in X (this loops over the whole matrix)

xrng = XDATA < X(m)+25 & XDATA > X(m) - 25; % points where x data withing 25 meters

yrng = YDATA < Y(m)+25 & YDATA > Y(m) - 25; % y data withing 25 meters

b) **Normalized radian frequency**
Z(m) = mean(ZDATA(xrng&yrng)); % take the mean of elevation points within grid range

```
2.a) x = randn(6,1);
y = randn(6,1);
A = [x y 2*y+3];
R = corrcoef(A)
```

R = *3×3*

```
    1.0000   -0.6237   -0.6237
   -0.6237    1.0000    1.0000
   -0.6237    1.0000    1.0000
```

```
A = randn(10,1);
B = randn(10,1);
R = corrcoef(A,B)
```

R = *2×2*

```
    1.0000    0.4518
    0.4518    1.0000
```

```
A = randn(50,3);
A(:,4) = sum(A,2);
[R,P] = corrcoef(A)
```

R = *4×4*

```
    1.0000    0.1135    0.0879    0.7314
```

```
    0.1135    1.0000   -0.1451    0.5082
    0.0879   -0.1451    1.0000    0.5199
    0.7314    0.5082    0.5199    1.0000
```

P = *4×4*

```
    1.0000    0.4325    0.5438    0.0000
    0.4325    1.0000    0.3146    0.0002
    0.5438    0.3146    1.0000    0.0001
    0.0000    0.0002    0.0001    1.0000
```

```
A = randn(50,3);
A(:,4) = sum(A,2);
[R,P,RL,RU] = corrcoef(A)
```

R = *4×4*

```
    1.0000    0.1135    0.0879    0.7314
    0.1135    1.0000   -0.1451    0.5082
    0.0879   -0.1451    1.0000    0.5199
    0.7314    0.5082    0.5199    1.0000
```

P = *4×4*

```
    1.0000    0.4325    0.5438    0.0000
    0.4325    1.0000    0.3146    0.0002
    0.5438    0.3146    1.0000    0.0001
    0.0000    0.0002    0.0001    1.0000
```

RL = *4×4*

```
    1.0000   -0.1702   -0.1952    0.5688
   -0.1702    1.0000   -0.4070    0.2677
   -0.1952   -0.4070    1.0000    0.2825
    0.5688    0.2677    0.2825    1.0000
```

RU = *4×4*

```
    1.0000    0.3799    0.3575    0.8389
    0.3799    1.0000    0.1388    0.6890
    0.3575    0.1388    1.0000    0.6974
```

```
    0.8389    0.6890    0.6974    1.0000
```

The matrices RL and RU give lower and upper bounds, respectively, on each correlation coefficient according to a 95% confidence interval by default.

```
2b)[R,P,RL,RU] = corrcoef(A,'Alpha',0.01)
```

R = *4×4*

```
    1.0000    0.1135    0.0879    0.7314
    0.1135    1.0000   -0.1451    0.5082
    0.0879   -0.1451    1.0000    0.5199
    0.7314    0.5082    0.5199    1.0000
```

P = *4×4*

```
    1.0000    0.4325    0.5438    0.0000
    0.4325    1.0000    0.3146    0.0002
    0.5438    0.3146    1.0000    0.0001
    0.0000    0.0002    0.0001    1.0000
```

RL = *4×4*

```
    1.0000   -0.2559   -0.2799    0.5049
   -0.2559    1.0000   -0.4792    0.1825
   -0.2799   -0.4792    1.0000    0.1979
    0.5049    0.1825    0.1979    1.0000
```

RU = *4×4*

```
    1.0000    0.4540    0.4332    0.8636
    0.4540    1.0000    0.2256    0.7334
    0.4332    0.2256    1.0000    0.7407
    0.8636    0.7334    0.7407    1.0000
```

NaN.

```
A = randn(5,3);
A(1,3) = NaN;
A(3,2) = NaN;
A
```

```
A = 5×3

    0.5377   -1.3077       NaN
    1.8339   -0.4336    3.0349
   -2.2588       NaN    0.7254
    0.8622    3.5784   -0.0631
    0.3188    2.7694    0.7147
```

```
R = corrcoef(A,'Rows','complete')
```

```
R = 3×3

    1.0000   -0.8506    0.8222
   -0.8506    1.0000   -0.9987
    0.8222   -0.9987    1.0000
```

```
R = corrcoef(A,'Rows','all')
```

```
R = 3×3
```

```
R = corrcoef(A,'Rows','pairwise')
```

```
R = 3×3

    1.0000   -0.3388    0.4649
   -0.3388    1.0000   -0.9987
    0.4649   -0.9987    1.0000
```

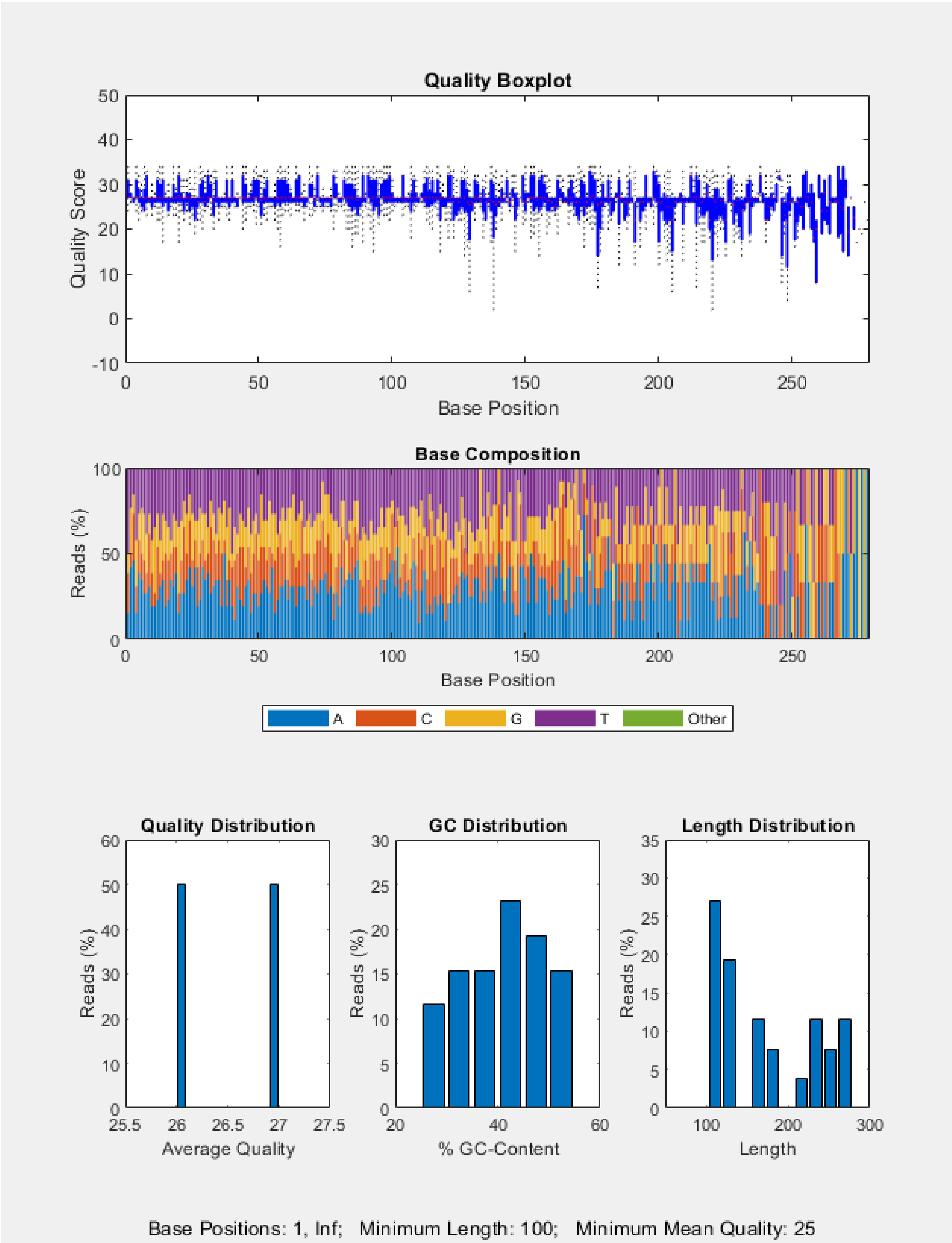
c) Analytics if IRR frequency in alpha

```
 function u=seqtan(x0)

x=x0;

y=x-tan(x);

while delta >etol

    y=x-tan(x);

    x=y;
```

d)magnitude spectrum

```
 x0=8;
```

```
etol=2e-10;
    y=x-tan(x);
    delta= abs(y-x)/abs(y)
    x=y;
    while delta >etol
        y=x-tan(x);
        delta= abs(y-x)/abs(y);
        u=y;
    end
```


Quality Boxplot
Quality Score
Base Position
Base Composition
Reads (%)
Base Position
A
C
G
T
Other
Quality Distribution
Reads (%)
Average Quality
GC Distribution
Reads (%)
% GC-Content
Length Distribution
Reads (%)
Length
Base Positions: 1, Inf; Minimum Length: 100; Minimum Mean Quality: 25

(b)

```
>> u=y;
>> u
end
    u =


        14.7997
```

B) function x= seqtan(x0,etol

```
x=1;

y=x-tan(x);

x=y;

while delta >etol

   y=x-tan(x);
   delta=abs(y-x)
   x=y;
 end
```