```matlab
clear all;
close all;

N = 128;
global forest;

%low p 0.1 and low f 0.5 high p 0.2 high f 0.9
%%Exercise 1
p = 0.2;
f = 0.5;
forest = zeros(128,128);
fire = 0;
step = 0;
while (step<40)
    step = step + 1;
    for i = 1:N
        for j = 1:N
            if (rand(1)<=p & forest(i,j) == 0)
                forest(i,j) = 1;
            end
        end
    end

    if (rand(1)<=f)
        i=randi(N);
        j=randi(N);
        if forest(i,j)==1
            fire=1;
        end
        spread(i,j,N);
    end
    burnt = zeros(16384,2);
    tree = zeros(16384,2);
    c1 = 1;
    c2 = 1;

    for i=1:N
        for j=1:N
            if forest(i,j) == 1
                tree(c1,1) = i;
                tree(c1,2) = j;
                c1 = c1+1;
            elseif forest(i,j) == 2
                forest(i,j) = 0;
                burnt(c2,1) = i;
                burnt(c2,2) = j;
                c2 = c2+1;
            end
        end
    end
    tree = tree(1:c1-1,:);
    burnt = burnt(1:c2-1,:);

    figure(step);
    plot(tree(:,1),tree(:,2),'g.');
    hold on;
    plot(burnt(:,1),burnt(:,2),'r.');
    t=sprintf('p=%f and f=%f - step %d',p,f,step);
    title(t);
    xlabel('x');
    ylabel('y');
    xlim([0 129]);
    ylim([0 129]);
    name = sprintf('%d.png',step);
    exportgraphics(gcf,name);
end


%Exercise 2
p = 0.05;
f = 0.7;
ratio = zeros(1000,1);
ratio_sim = zeros(1000,1);
aux=zeros(128,128);
for a=1:1000
    forest = zeros(N,N);
    forest = aux;
    fire = 0;
    step = 0;
    while (fire==0)
        step = step + 1;
        for i = 1:N
            for j = 1:N
                if (rand(1)<=p & forest(i,j) == 0)
                    forest(i,j) = 1;
                end
            end
        end
        if (rand(1)<=f)
            i=randi(N);
            j=randi(N);
            if forest(i,j)==1
                fire=1;
            end
            spread(i,j,N);
        end
    end

    burnt = zeros(16384,2);
    tree = zeros(16384,2);
    c1 = 1;
    c2 = 1;

    for i=1:N
        for j=1:N
            if forest(i,j) == 1
                tree(c1,1) = i;
                tree(c1,2) = j;
                c1 = c1+1;
            elseif forest(i,j) == 2
                forest(i,j) = 0;
                burnt(c2,1) = i;
                burnt(c2,2) = j;
                c2 = c2+1;
            end
        end
    end
    aux = forest;
    tree = tree(1:c1-1,:);
    burnt = burnt(1:c2-1,:);
    ratio(a) = (c2-1)/(128*128);

    size_forest = c2+c1-2;
    forest = zeros(N,N);
    random_forest(size_forest,N);   %Create new random forest of that
size
    loc = pick_fire(N);
    spread(loc(1),loc(2),N);
     burnt = zeros(16384,2);
    tree = zeros(16384,2);
    c1 = 1;
    c2 = 1;

    for i=1:N
        for j=1:N
            if forest(i,j) == 1
                tree(c1,1) = i;
                tree(c1,2) = j;
                c1 = c1+1;
            elseif forest(i,j) == 2
                burnt(c2,1) = i;
                burnt(c2,2) = j;
                c2 = c2+1;
            end
        end
    end
    tree = tree(1:c1-1,:);
    burnt = burnt(1:c2-1,:);
    ratio_sim(a) = (c2-1)/(128*128);
end

figure(5);
a = 1:1000;
ratio = sort(ratio);
ratio = flip(ratio);
ratio_sim = sort(ratio_sim);
ratio_sim = flip(ratio_sim);
rankk = a./1000;
loglog(ratio,rankk,'b.-');
hold on;
loglog(ratio_sim,rankk,'r.-');
legend({'Original forest','Simulated forest'});
xmin=min(ratio(1000),ratio_sim(1000));
xmax=max(ratio(1),ratio_sim(1));
xlim([xmin*0.99 xmax*1.2]);
xlabel('Relative fire size');
ylabel('cCDF');
%exportgraphics(gcf,'ex2.png');


%Exercise 3: log(y) = mlog(x)+b
```

```
ratio_log = flip(ratio);
rankk_log = flip(rankk);
ratio_log = log(ratio_log);
rankk_log = log(rankk_log);
x = ratio_log(1:200);
y = rankk_log(1:200);
c = polyfit(x,y,1);
y_est = polyval(c,x);
%m=zeros(10,1);
figure(6);
plot(ratio_log,rankk_log,'b.');
hold on;
plot(x,y_est,'r--','LineWidth',2);
lab = sprintf('slope = %.2f',c(1));
legend({'Fire Data',lab});
xlabel('log(Relative fire size)');
ylabel('log(cCDF)');
%exportgraphics(gcf,'1.png');

flag=0;
for j=3:10
    x = ratio_log(j*100-99:j*100);
    y = rankk_log(j*100-99:j*100);
    c = polyfit(x,y,1);
    y_est = polyval(c,x);
    figure(10+j);
    plot(ratio_log,rankk_log,'b.');
    hold on;
    plot(x,y_est,'r--','LineWidth',2);
    lab = sprintf('slope = %.2f',c(1));
    legend({'Fire Data',lab});
    name = sprintf('%d.png',j);
    xlabel('log(Relative fire size)');
    ylabel('log(cCDF)');
    %exportgraphics(gcf,name);
    if (c(1)<-0.9 & flag == 0)
        flag = j-1;
    end
end
x = ratio_log(1:100*flag);
y = rankk_log(1:100*flag);
c = polyfit(x,y,1);
y_est = polyval(c,x);
%rank=ratio^m * 10^b
rankk_app =(ratio.^(c(1)));
rankk_app = rankk_app/max(rankk_app);
figure(7);
plot(ratio_log,rankk_log,'b.');
hold on;
plot(x,y_est,'r--','LineWidth',2);
lab = sprintf('final approximation: %.2f',c(1));
legend({'Fire Data',lab});
xlim([ratio_log(1)*0.99 ratio_log(1000)*1.2]);
xlabel('log(Relative fire size)');
ylabel('log(cCDF)');
%exportgraphics(gcf,'ex3_a.png');
```

```
%{
loglog(ratio,rankk,'b.-');
hold on;
loglog(ratio,rankk_app,'r.-');
xmin=ratio(1000);
xmax=ratio(1);
xlim([xmin*0.99 xmax*1.2]);
%}

tau=1-c(1);
%r_i=zeros(200,1);
%for i=1:200
   %r_i(i)=rand(1);
%end
%r_i=sort(r_i);
%r_i=flip(r_i);
xmin=ratio(1000);
Xi=xmin.*((rankk).^(-1/(tau-1)));
%Xi=Xi./max(Xi);
figure(8);
loglog(ratio,rankk,'b.-');
hold on;
loglog(Xi,rankk,'r.-');
lab = sprintf('power law with \\tau = %.3f',tau);
legend({'Fire Data',lab});
xlim([ratio_log(1)*0.99 ratio_log(1000)*1.2]);
xlim([ratio(1000)*0.99 ratio(1)*1.2]);
xlabel('Relative fire size');
ylabel('cCDF');
%exportgraphics(gcf,'ex3_b.png');


%Excersise 4
N=[8 16 32 64 128 256 320];
p = 0.1;
f = 0.7;
tau = zeros(7,1);

for nn=1:7
    disp(nn);
    aux=zeros(N(nn),N(nn));
    ratio = zeros(1000,1);
    for a=1:1000
        forest = zeros(N(nn),N(nn));
        forest = aux;
        fire = 0;
        step = 0;
        while (fire==0)
            step = step + 1;
            for i = 1:N(nn)
                for j = 1:N(nn)
                    if (rand(1)<=p & forest(i,j) == 0)
                        forest(i,j) = 1;
                    end
                end
            end
            if (rand(1)<=f)
                i=randi(N(nn));
                j=randi(N(nn));
                if forest(i,j)==1
                    fire=1;
                end
                spread(i,j,N(nn));
            end
        end

        burnt = zeros(N(nn)*N(nn),2);
        tree = zeros(N(nn)*N(nn),2);
        c1 = 1;
        c2 = 1;

        for i=1:N(nn)
            for j=1:N(nn)
                if forest(i,j) == 1
                    tree(c1,1) = i;
                    tree(c1,2) = j;
                    c1 = c1+1;
                elseif forest(i,j) == 2
                    forest(i,j) = 0;
                    burnt(c2,1) = i;
                    burnt(c2,2) = j;
                    c2 = c2+1;
                end
            end
        end
        aux = forest;
        tree = tree(1:c1-1,:);
        burnt = burnt(1:c2-1,:);
        ratio(a) = (c2-1)/(N(nn)*N(nn));
    end
    a = 1:1000;
    ratio = sort(ratio);
    ratio = flip(ratio);
    rankk = a./1000;
    ratio_log = flip(ratio);
    rankk_log = flip(rankk);
    ratio_log = log(ratio_log);
    rankk_log = log(rankk_log);
    x = ratio_log(1:500);
    y = rankk_log(1:500);
    c = polyfit(x,y,1);
    y_est = polyval(c,x);

    flag=0;
    for j=6:10
        x = ratio_log(j*100-99:j*100);
        y = rankk_log(j*100-99:j*100);
        c = polyfit(x,y,1);
        if (c(1)<-1.5 & flag == 0)
            flag = j-1;
        end
    end
    x = ratio_log(1:100*flag);
```

```
    y = rankk_log(1:100*flag);
    c = polyfit(x,y,1);
    tau(nn)= 1 - c(1);
end

x=[0 1/1024 1/700 1/900 1/512 1/400 1/256 1/190 1/128 1/64 1/32 1/20 1/16
1/8];
c = polyfit(1./N,tau,5);
y_est = polyval(c,x);
y=interp1(1./N,tau,x,'makima','extrap');
z=sqrt([1/8 1/16 1/32 1/64 1/128 1/256 1/320]);
c = polyfit(z,tau,1);
figure(9);
plot(1./N,tau,'c.-','LineWidth',2);
hold on;
plot(x,y_est,'r--','LineWidth',1.5);
legend({'Original data','Extrapolated data'});
xlabel('1/N');
ylabel('\tau');
x=0:0.0005:0.125;
y=c(1).*sqrt(x) + c(2);
figure(10);
plot(1./N,tau,'c.-','LineWidth',2);
hold on;
plot(x,y,'r--','LineWidth',1.5);
legend({'Original data','Extrapolated data'});
xlabel('1/N');
ylabel('\tau');



%%Functions
function loc = pick_fire(N)
    global forest;
    loc=zeros(2,1);
    ok=0;
    while (ok==0)
        i=randi(N);
        j=randi(N);
        if forest(i,j)==1
           loc(1)=i;
           loc(2)=j;
           ok=1;
        end
    end
end

function random_forest(size,N)
    global forest;
    trees = 0;
    while(trees<size)
        i = randi(N);
        j = randi(N);
        if forest(i,j)==0
            trees = trees + 1;
            forest(i,j) = 1;
        end
    end
end


function spread(i,j,N)
global forest;
    if forest(i,j) == 1
        forest(i,j) = 2;
        if (i-1 > 0)
            spread(i-1,j,N);
        end
        if (i+1<=N)
            spread(i+1,j,N);
        end
        if (j-1 > 0)
            spread(i,j-1,N);
        end
        if (j+1<=N)
            spread(i,j+1,N);
        end
    end
end
```

Homework 2

SIMULATION OF COMPLEX SYSTEMS

FRANCISCO CAETANO

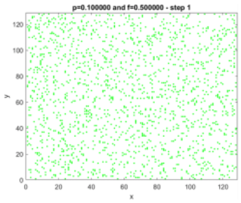
1

Exercise 1

- ❑128x128-lattice
- ❑p: probability of an empty site to turn into an occupied one
  - ❑Increasing p leads to denser forests (bigger fires)
  - ❑Decreasing p leads to low density forests (smaller fires)
- ❑f: probability of a lightning strike occurring at a random site
  - ❑Increasing f causes more fires per time step (it can delay the growth of a forest)
  - ❑Decreasing f causes less fires per time step (but it allows forests to become denser)

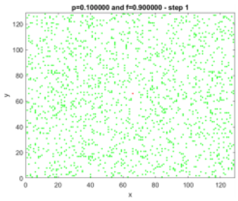
2

Low p and Low f



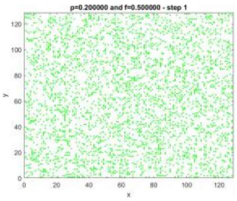
3

Low p and High f



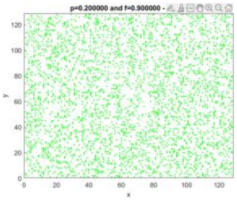
4

High p and Low f



5

High p and High f



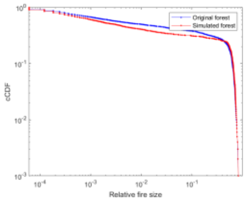
6

Exercise 2

- ❑128x128-lattice
- ❑Rank-frequency plot of forest grown with fires (clusters will start to emerge more frequently, small fires (0.001 ratio) become less probable as the forest grows)
- ❑Rank-frequency plot of forest grown without fires (random distribution of the trees, no relevant patterns)

7

Rank-frequency plots



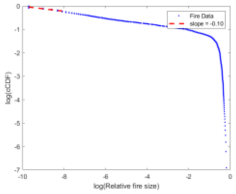
8

Exercise 3

- ❑ Rank-frequency plot with the fire data and the linear fit used to find the exponent τ
  - ❑ Find the limits for which linear approximation is valid
  - ❑ Determine τ
- ❑ Rank-frequency plot comparing the fire data and the synthetic power law data.

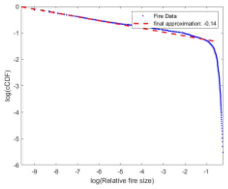
9

Limits



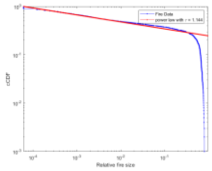
10

Determine τ



11

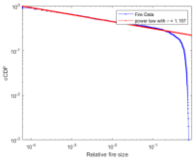
Rank-frequency plot



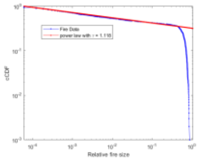
12

Is the result sensitive to the choice of parameters p and f?

INCREASE F (TO 0.9): INCREASES T

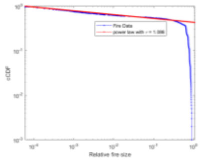


REDUCE F (TO 0.5): DECREASES T

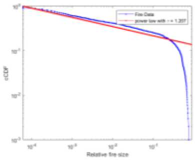


Is the result sensitive to the choice of parameters p and f?

INCREASE P (TO 0.2): DECREASES T



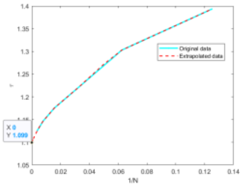
REDUCE P (TO 0.05): INCREASES T

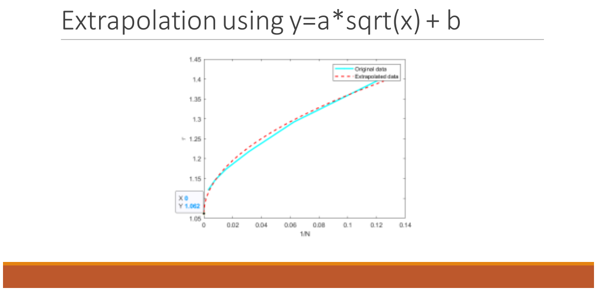


Exercise 4

- ❑Extraplation to obtain the limit of τ for N→ ∞
  - ❑Using Matlab functions
  - ❑Using y=a*sqrt(x)+b with z=sqrt(x)

Extrapolation using polynomial approximation





Extrapolation using y=a*sqrt(x) + b

17