دانشگاه صنعتی امیرکبیر

(پلی تکنیک تهران)

دانشکده مهندسی کامپیوتر و فناوری اطلاعات

گزارش تمرین سوم (DTPC)

نگارش

جابر بابکی

استاد

دکتر مهدی راستی

فروردین ۱۳۹۸

In this homework, apply TPC, OPC and DTPC algorithms to a simple wireless network with 6 users. For the TPC and DTPC, each user has a minimum target-SINR denoted by GAMA(user), which is assumed to be the same for all users for sake of simplicity. The users should be uniformly distributed in the cell. The following parameters are fixed for both constrained and unconstrained transmit power cases:

- Background noise power 10^-10
- OPC constant 0.05
- Path gain 0.1d^-3

1- Unconstrained Transmit Power Case

Simulate the system under a condition that the minimum target-SINRs for all users are feasible assuming no constraint on maximum power.

```
clc;
close all;
clear all;

NOISE=1e-10;
OPCConstant=10e-4;
powerInitiate=rand(1,6)*1e-10;
GAMA=0.01;

gHatEU=ones(1,6)*GAMA;
distnceVector=FuncPosition();%create system model and calculate
pathGainVector=FuncPathGain(distnceVector);%calculate pathgaine
sinr=FuncSINR(pathGainVector,powerInitiate,NOISE);

DTPC=FuncDTPC(pathGainVector,gHatEU,OPCConstant,NOISE );
FuncFigure(DTPC);
```

سیستم را با شرایط خواسته شده ایجاد کردیم، در مورد توابع استفاده شده در زیر توضیح می دهم:

تابع تعیین موقعیت FuncPosition:

```
function [ distnceVector ] = FuncPosition()
    clear all;
    BsCOVRAGE=500;%base station coverage area
    Bs1POSITION=[250,250];
    userJoinBs1=ones(6,1);
    for n=1:6
        x=rand(1)*BsCOVRAGE;
        y=rand(1)*BsCOVRAGE;
        if(x==Bs1POSITION(1)&&y==Bs1POSITION(2))
            x=x+1;
            y=y+1;
        end
        userJoinBs1(n,1)=complex(y,x);
    end
    distnceVector=FuncDistance(userJoinBs1,Bs1POSITION);
end
```

در تابع تعیین موقعیت، موقعیت هر کاربر به صورت یکنواخت در محدوده پوشش سلول که شعاع ۲۵۰ هست ایجاد شده است و سپس فاصله هر کاربر با BS توسط تابع FuncDistance مشخص شده است.

تابع محاسبه فاصله FuncDistance:

```
function [ dis ] = FuncDistance( userJoinBs1,Bs1POSITION)
    A=complex(Bs1POSITION(1),Bs1POSITION(2));
    for s=1:6
            temp=userJoinBs1(s,1)-A;
            dis(s)=abs(temp);
    end
end
```

تابع محاسبه گین FuncPathGain:

```
function [ pGain ] = FuncPathGain( dis )
    pGain=0.09*(dis.^-3);
end
```

در این تابع براساس رابطه داده شده گین هر کاربر محاسبه می شود.

تابع FuncDTPC:

```
function [ DTPC ] = FuncDTPC(pathGainVector,GAMA,eta,NOISE)
    iteratin=1000;
    powerInitiate=ones(1,6)*1e-10;
    sinr(1,:)=FuncSINR(pathGainVector,powerInitiate,NOISE);
    for j=2:iteratin
        for i=1:6
            powerInitiate(j,i)=max(eta*(sinr((j-1),i)/powerInitiate((j-
1),i)),powerInitiate((j-1),i)*(GAMA(i)/sinr((j-1),i)));
        end
        sinr(j,:)=FuncSINR(pathGainVector,powerInitiate(j,:),NOISE );
        if(abs(powerInitiate(j,:) - powerInitiate(j-1,:))<=0.00001)
            break
        end
    end
    DTPC{1}=powerInitiate;
    DTPC{2}=sinr;
    DTPC{3}=j;
end
```

در این تابع الگوریتم DTPC اجرا می شود، برای اجرای این الگوریتم نیاز به این داریم که در بازه تکراری الگوریتم اجرا و توان اپدیت شود به همین جهت در حلقه while با تکرار ۱۰۰۰ قرار داده شده البته، البته مقدار خطایی هم جهت پایان برنامه در نظر گرفته شده است. در داخل حلقه while به ازای هر کاربر محاسبه زیر انجام می شود:

$$p_i(t+1) = \max\{\hat{\gamma}_i \frac{p_i(t)}{\gamma_i(t)}, \eta_i \frac{\gamma_i(t)}{p_i(t)}\}$$

در هر با اجرا ما باید توان کاربران را تعیین کنیم به همین جهت نیاز به محاسبه تداخل و SNRکاربران داریم که در داخل حلقه For محاسبه می شود و در نهایت power وSNR را به عنوان خروجی برمی گرداند.

تابع رسم نمودار FuncFigure:

```
function FuncFigure( DTPC )
    powerInitiate=DTPC{1};
    gamma=DTPC{2};
    j=DTPC{3};
    x=1:(j);
    figure(1)
    plot(x,powerInitiate),grid on;
    xlabel('iteration')
    ylabel('power')
    title('Plot of DTPC Power')
    figure(2)
```

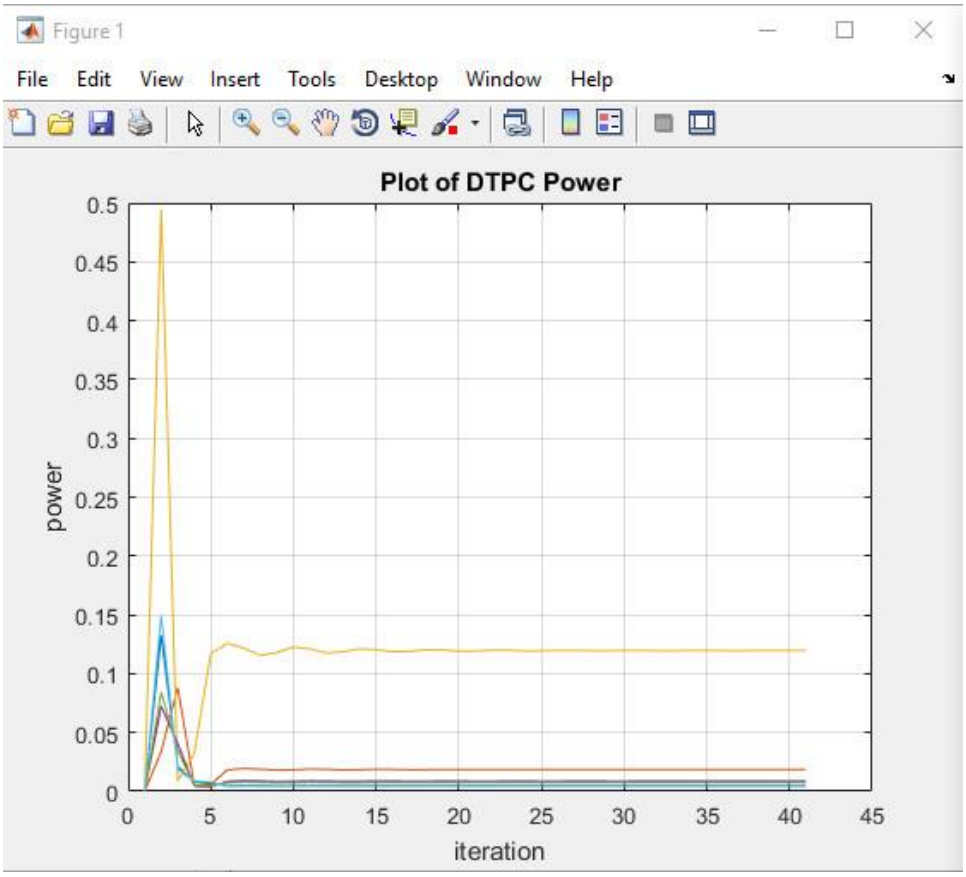
```
    plot(x,gamma),grid on;
    xlabel('iteration')
    ylabel('SINR')
    title('Plot of DTPC SINR')
end
```

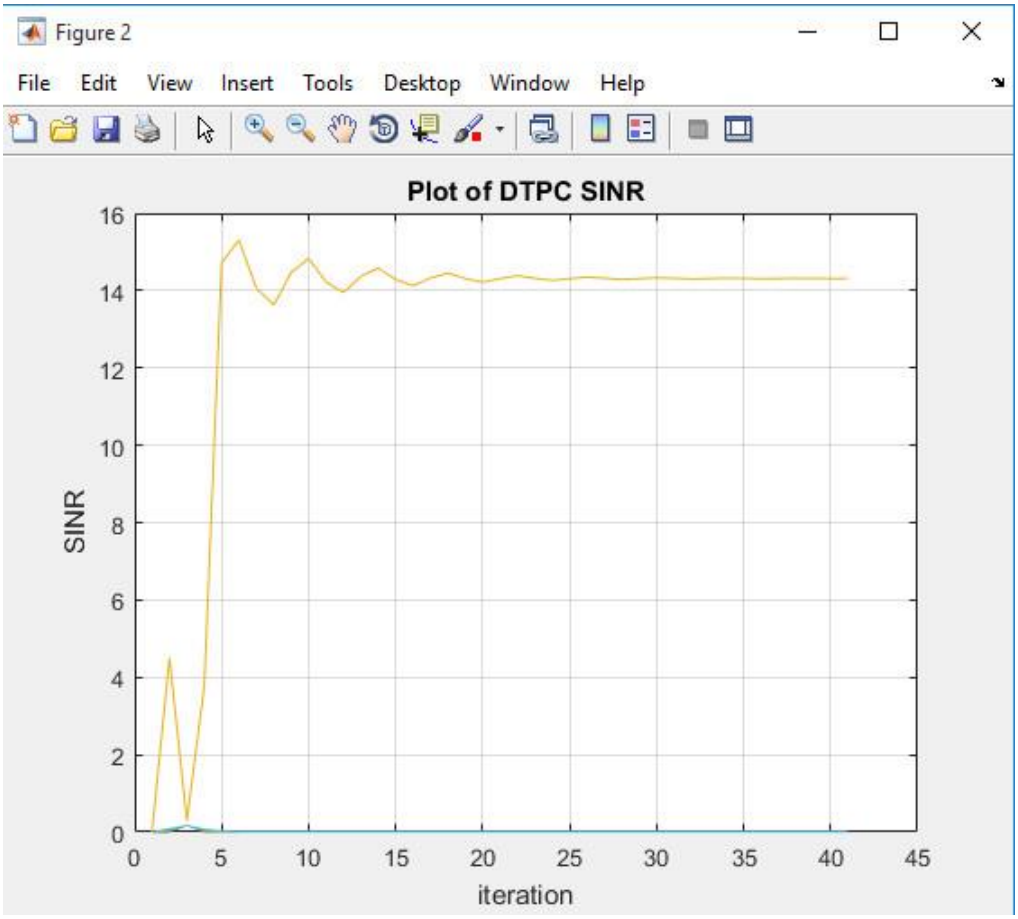
در نهایت power و SNR را در نمودار نمایش می دهد.

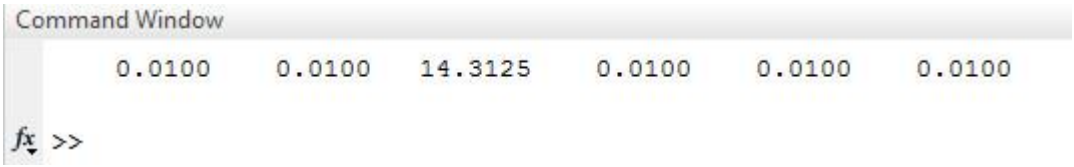
1-1 Plot the SINR and power of each user versus the number of iterations (a measure of time), for several values of minimum target-SINR.

در این بخش ما target SNR را برای ۴ مقدار مختلف که همگی این مقادیر در سیستم feasible انجام شده را تست می کنیم

ابتدا مقدار target snr را برابر 0.01 قرار می دهیم و نمودار های زیر بدست می آید.







همانطور که در شکل های بالا نشان داده شده است، تمام کاربران به SNR هدفشان رسیدند و سیستم همگرا شده است و کاربر سوم که رنگ قهوه ای در نمودار نشان داده شده است از SNR هدفش مقدار بیشتری هم گرفته است و توانش هم نسبت به بقیه کاربران بیشتر است.

مقدار بعدی target SNR را برابر ۰,۰۹ قرار می دهیم و نمودار های زیر بدست می آید.

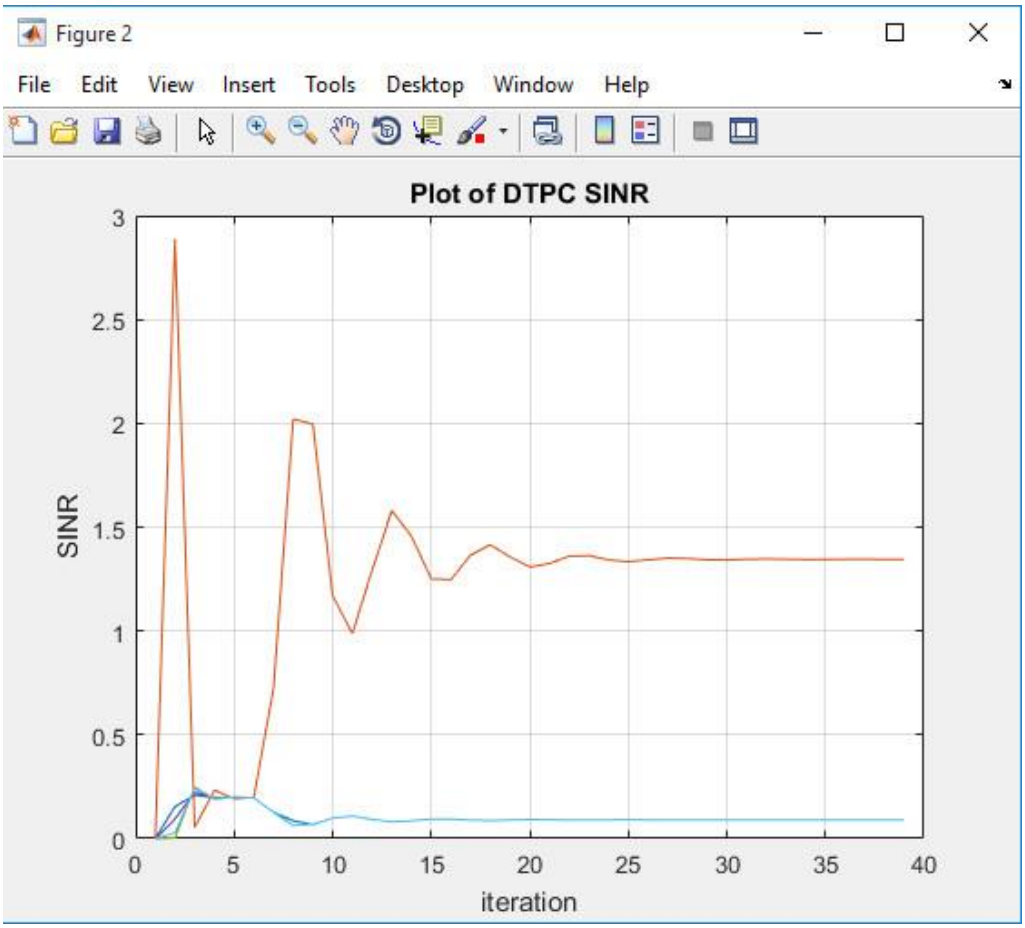


Figure 2
File Edit View Insert Tools Desktop Window Help
Plot of DTPC SINR
SINR
iteration
0 0.5 1 1.5 2 2.5 3
0 5 10 15 20 25 30 35 40

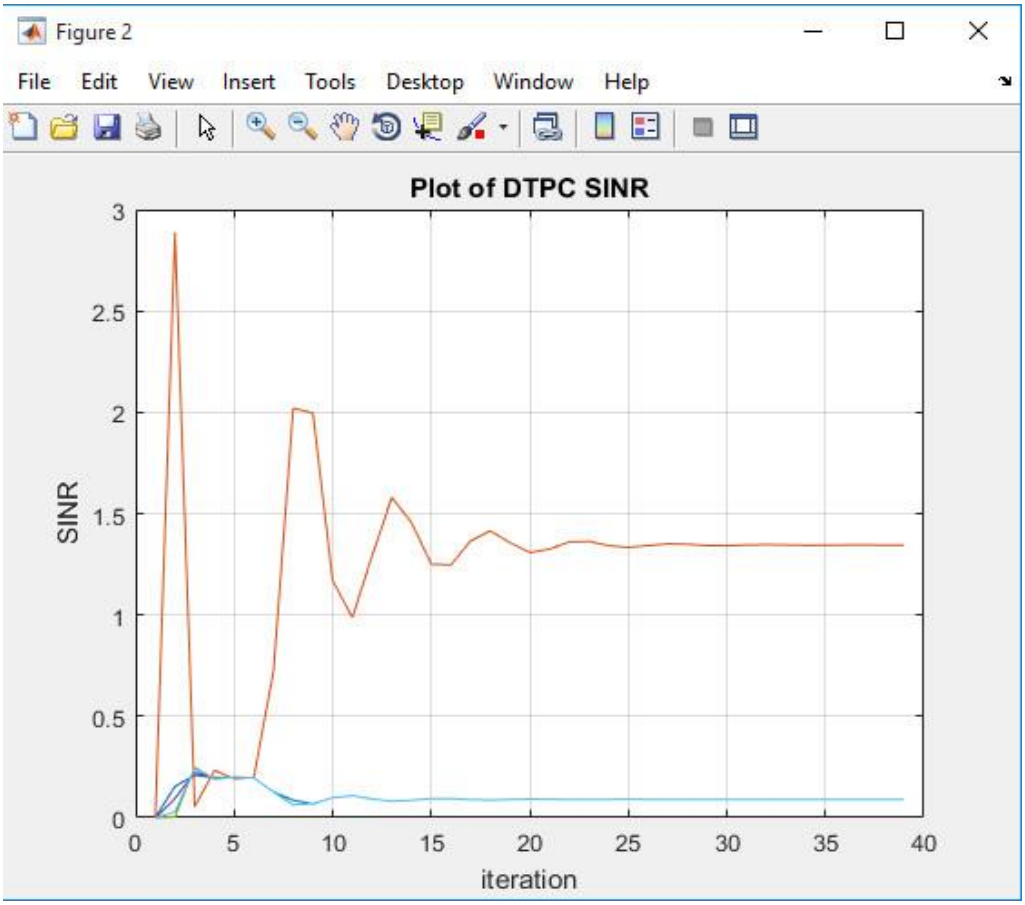
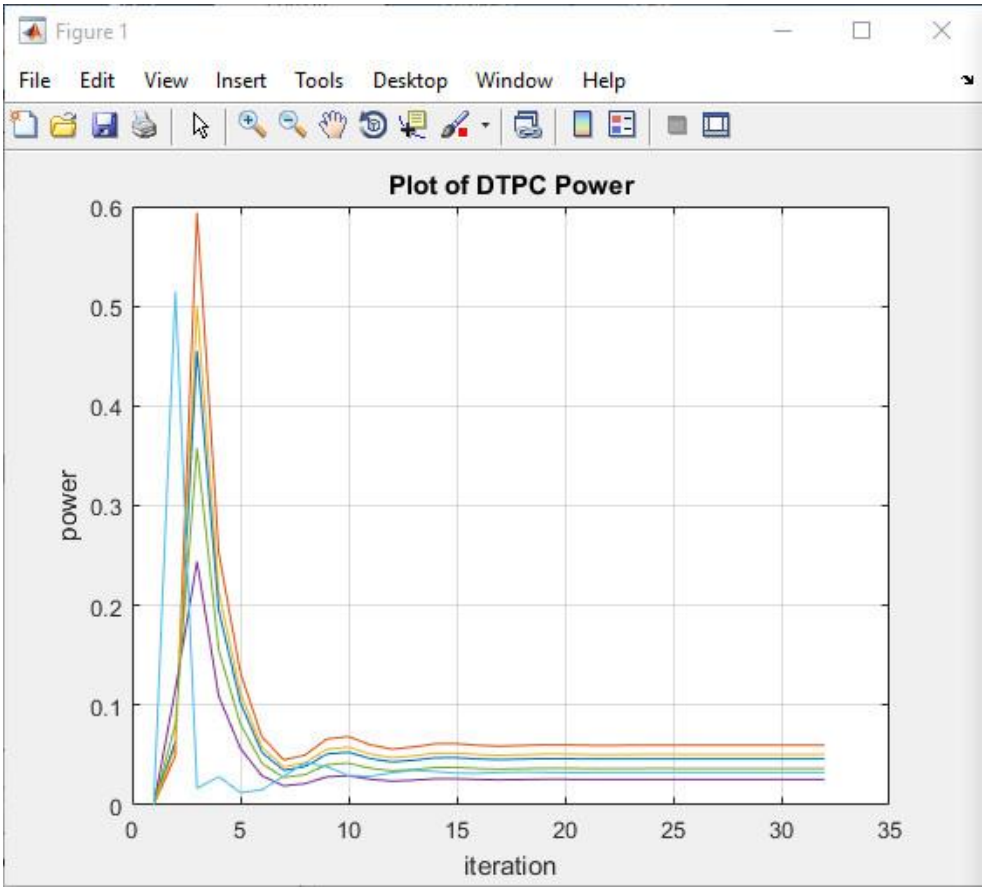


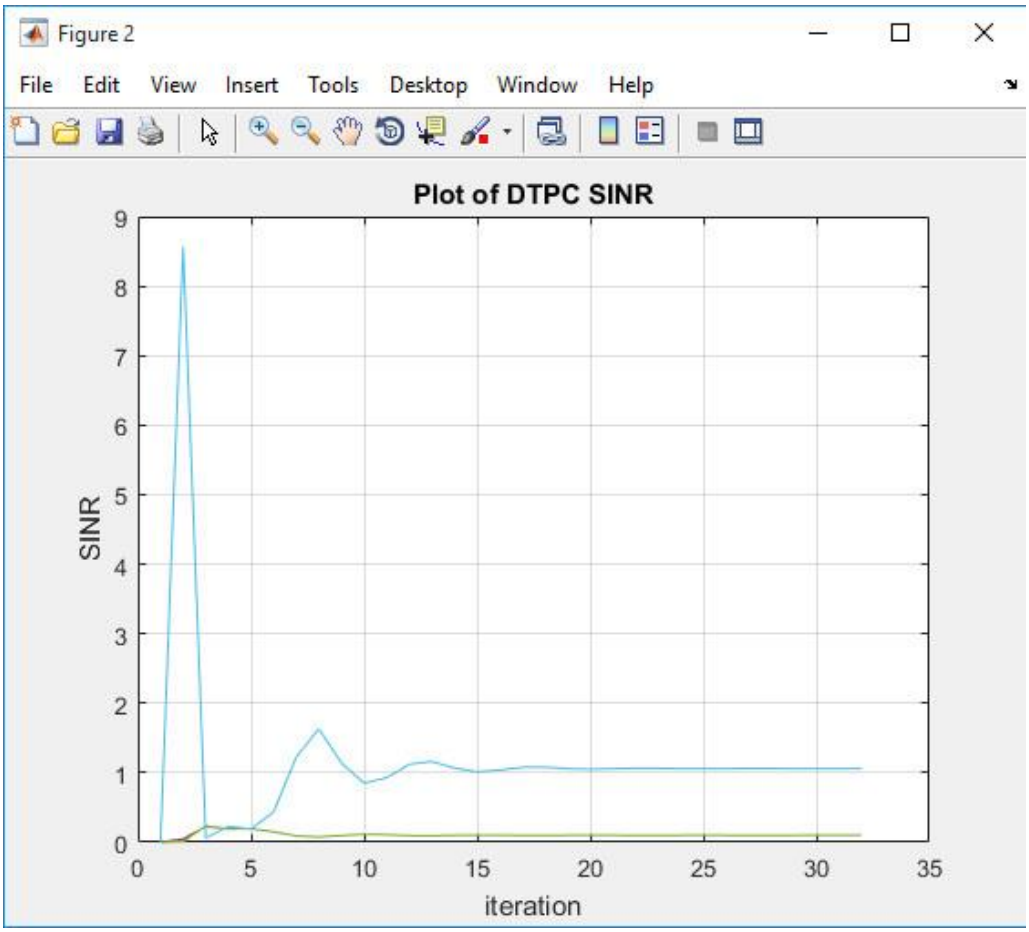
Figure 2
File Edit View Insert Tools Desktop Window Help
Plot of DTPC SINR
SINR
iteration
0 0.5 1 1.5 2 2.5 3
0 5 10 15 20 25 30 35 40

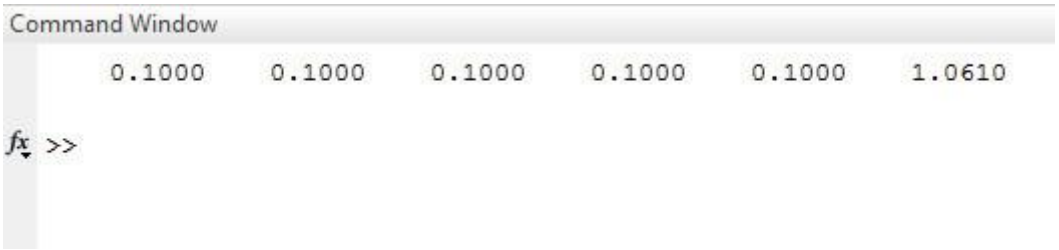
```
Command Window
    0.0900    1.3447    0.0900    0.0900    0.0900    0.0900

fx >>
```

مشاهده می شود که تمام کاربران به SNR هدفشان رسیدند و کاربر شماره ۲ که با رنگ قهوه ای نشان داده شده است مقدار SNR بیشتری نسبت به SNR هدفش دریافت کرده و توان بیشتری هم گرفته است.

مقدار بعدی target SNR را برابر ۰,۱ قرار می دهیم که بالاترین مقدار نسبت بـه مقـدار هـای قبلـی است و نمودار های زیر بدست می آید.







همانطور که در شکل های بالا دیده می شود تمام کاربران به SNR هدفشان رسیدند و سیستم feasible است و کاربر ۶ که با رتگ ابی نشان داده شده است بیشتر از مقدار SNR هدفش دریافت کرده است. که در واقع این همان کاربری است که مطابق الگوریتم OPC عمل می کند و توانسته به SINR ای بالاتر از مقدار حداقلی Target SINR دست یابد.

با توجه به تغییر سه مقدار قبلی می توان دید که هر چقدر مقدار SNR هدف کاربران افزایش می یابد، مقدار افزایش یافته به کاربری بیش از حد SNR هدفش گرفته، کمتر از مقدار های قبلی است.

1-2 Compare and discuss the results in terms of outage ratio and system throughput for the TPC, OPC and DTPC algorithms with the change in minimum target-SINR.

در این قسمت سوال خواسته شده است که الگوریتم‌های TPC, OPC, DTPC را از نظر نرخ outage و throughput مقایسه شود.

کد های این قسمت به صورت زیر نوشته شده است:

```
clc;
close all;
clear all;

NOISE=1e-10;
OPCConstant=10e-4;
powerInitiate=rand(1,6)*1e-5;

output=FuncCompaireAlg(powerInitiate,NOISE,OPCConstant);
FuncFigureCompair(output);


function [ output ] = FuncCompaireAlg(powerInitiate,NOISE,OPCConstant)
    umber=100;
    c=0;
    for gamaHat=0.01:0.01:0.3
        c=c+1;
        for i=1:umber
            gHatEU=ones(1,6)*gamaHat;
            distnceVector=FuncPosition();%create system model and
            calculate distance;
            pathGainVector=FuncPathGain(distnceVector);%calculate
            pathgaine
            TPC(c,i) = FuncTPC(pathGainVector,NOISE,gHatEU);
            OPC(c,i) =
            FuncOPC(pathGainVector,powerInitiate,gHatEU,NOISE,OPCConstan
            t );
            DTPC(c,i) = FuncDTPCO(
            pathGainVector,gHatEU,OPCConstant,NOISE );
        end
        disp(gamaHat);
        TPCR(c)=sum(TPC(c,:))/umber ;
        OPCR(c) = sum(OPC(c,:))/umber ;
        DTPCR(c) = sum(DTPC(c,:))/umber;
    end
    output{1}=TPCR;
    output{2}=OPCR;
    output{3}=DTPCR;
end
```

کد مربوط به این قسمت در واقع به این صورت عمل می کند که حلقه for از از snr هدف برابر با 0.01 شروع می شود و هر بار مقدار 0.01 به آن اضافه می شود تا به SNR هدف 0.3 برسد در داخل حلقه

هر بار الگوریتم های TPC و DTPC و OPC با پوزیشن های مختلف کاربران اجرا می شوند سپس میانگین محاسبه و میزان outage بدست می اید، در هر یک از الگوریتم ها کاربری را outage در نظر گرفتم که به SNR هدف خود نرسیده باشد. الگوریتم های TPC و OPC وDTPC در زیر نشان داده شده است.

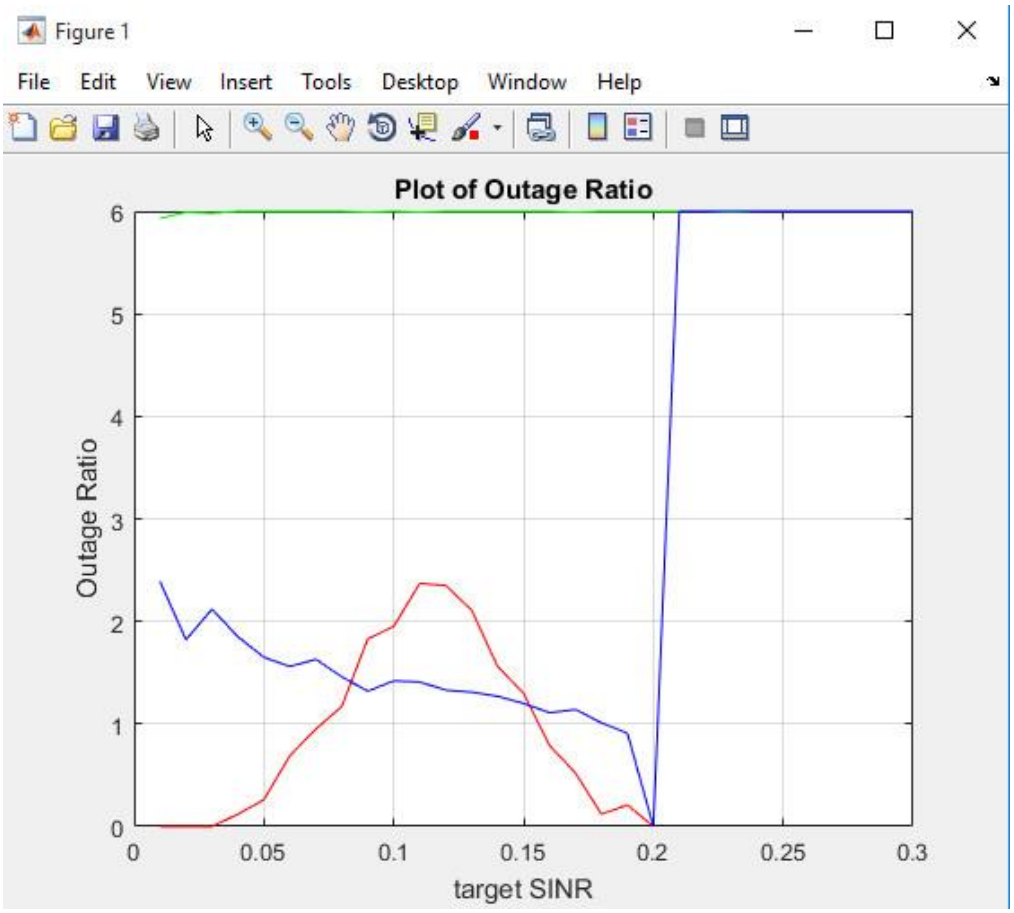
```
function [outage] = FuncDTPCO(pathGainVector,GAMA,eta,NOISE)
    outage=0;
    iteratin=1000;
    powerInitiate=ones(1,6)*1e-5;
    sinr(1,:)=FuncSINR(pathGainVector,powerInitiate,NOISE);
    for j=2:iteratin
        for i=1:6
            powerInitiate(j,i)=max(eta*(sinr((j-1),i)/powerInitiate((j-
1),i)),powerInitiate((j-1),i)*(GAMA(i)/sinr((j-1),i)));
        end
        sinr(j,:)=FuncSINR(pathGainVector,powerInitiate(j,:),NOISE );
        if(abs(powerInitiate(j,:) - powerInitiate(j-1,:))<=0.00001)
            break
        end
    end
    for i=1:6
        if (abs(sinr(j,i)-GAMA(i))>0.001)
            outage=outage+1;
        end
    end
end
```

```
function [ outage ] = FuncOPC(
pathGainVector,powerInitiate,gHatEU,NOISE,OPCConstant )
   G(1,:)=FuncSINR(pathGainVector,powerInitiate(1,:),NOISE);
   IterNum=500;
   outage=0;
   for T = 2:IterNum
        for i = 1 : 6
            powerInitiate(T,i) = (OPCConstant*G(T-1,i))/powerInitiate(T-
1,i);
        end
        G(T,:)=FuncSINR(pathGainVector,powerInitiate(T,:),NOISE);
        if(abs(powerInitiate(T,:) - powerInitiate(T-1,:))<=0.0001)
            break
        end
   end
   for i=1:6
       if (abs(G(T,i)-gHatEU(i))>0.001)
           outage=outage+1;
       end
    end
end
```

```
function [ outage ] = FuncTPC(pathGainVector,NOISE,gHatEU )
```

```
    ITERATION=1000;
    randomPower = 1e-3*rand(1,6);
    gamma(1,:)=FuncSINR(pathGainVector,randomPower,NOISE);
    outage=0;
    for j=2:ITERATION
        for i=1:6
            randomPower(j,i)=randomPower((j-1),i)*(gHatEU(i)/gamma((j-
            1),i));
        end
        gamma(j,:)=FuncSINR(pathGainVector,randomPower(j,:),NOISE);
        if (abs(randomPower(j,:)-randomPower(j-1,:))<=0.0001)
            break
        end
    end
    for i=1:6
        if (abs(gamma(j,i)-gHatEU(i))>0.001)
            outage=outage+1;
        end
    end
end
```

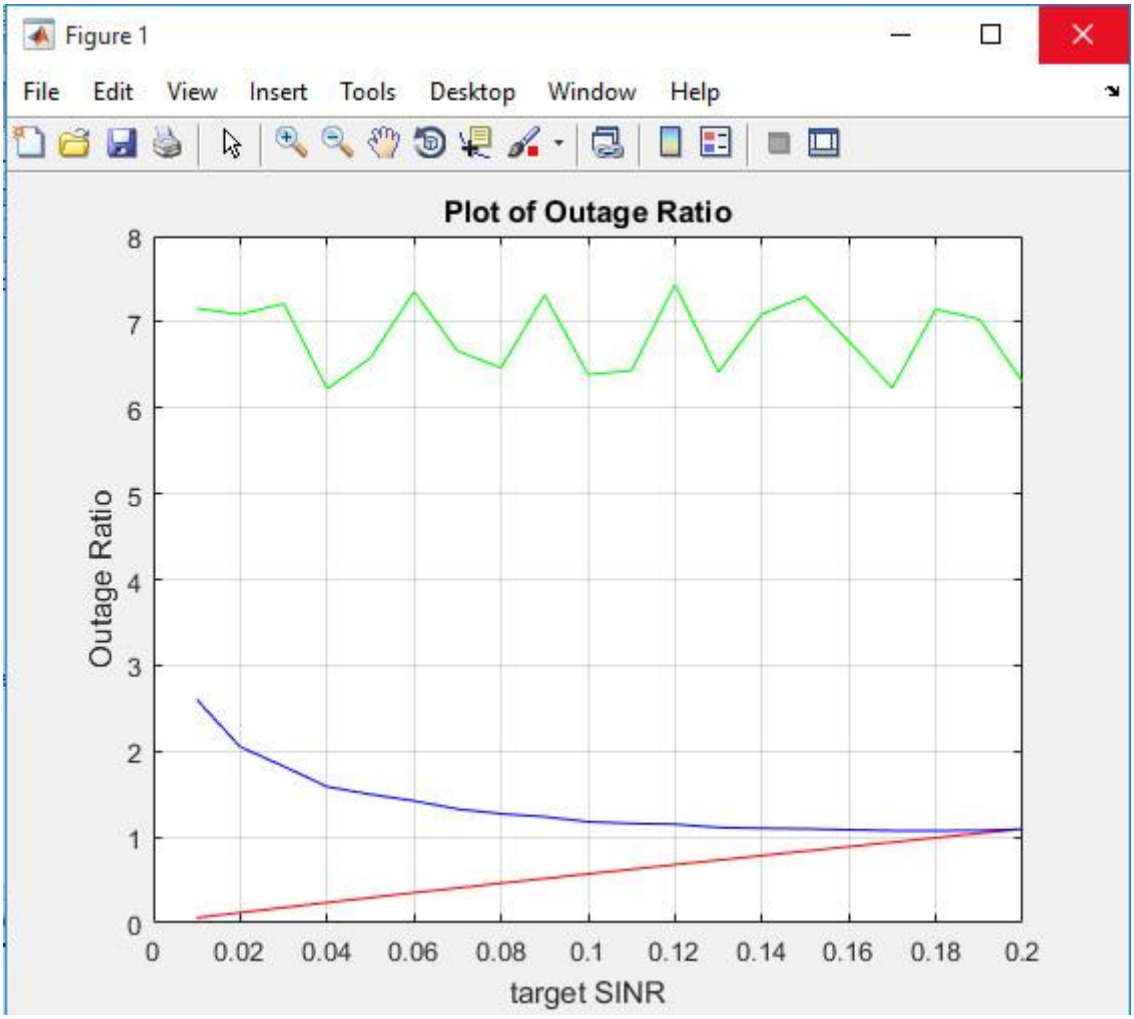
بعد از اجرای نتایج به صورت زیر بدست آمده است:



در نمودار رنگ قرمز الگوریتم TPC است و رنگ سبز الگوریتم OPC و رنگ آبی الگوریتم DTPC است.

با توجه به شکل نشان داده می شود که هر چه snr هدف افزایش می یابد سیستم به سمت infesiable پیش می رود. همانطور که مشاهده می شود زمانی که سیستم feasible است میزان outage در الگوریتم DTPC و TPC برابر صفر است و میزان گذردهی DTPC بیشتر است.

نکته ای دیگر که در شکل دیده می شود این است که میزان outage الگوریتم DTPC از یه SNR هدفی به بعد برابر می شود بار TPC و OPC و هر چقدر به سیستم infeasible می رسیم اختلاف این دو الگوریتم کم می شود.



OPCسبز رنگ و TPC قرمز و DTPC آبی است.

مقدار Throughput در OPC بالاتر(شکل بالا نمودار گذردهی است) از دو الگوریتم دیگر است زیرا OPCهیچ محدودیتی ندارد که همه کاربران به یک مقدار حداقلی SINR هدفشان برسند، و فقط مقدار SINRیک کاریر که گین خوبی دارد را افزایش می دهد که همین عامل باعث افزایش throughput در OPC می گردد .اما میزان outage زیاد می شود. در الگوریتم DTPC چون می خواهیم یک مقدار حداقلی SINR را به همه کاربران بدهیم، گذردهی مقداری کاهش می یابد. اما مقایسه میزان گذردهی دو الگوریتم DTPC و TPC نشان می دهد که DTPC بهتر است زیرا در الگوریتم TPCهمه کاربران به یک مقدار ثابت از SNR دست می یابند، درصورتیکه در DTPC همه کاربران به SNR می رسند و حتی تعداد محدودی از کاربران می توانند به مقدار بالاتری از SINR برسند، که همین عاملی است برای افزایش مقدار گذردهی در DTPC. در موقعه ای که سیستم به سمت infeseable می رود گذردهی TPC و DTPC به هم نزدیک می شود.

1-3 Do all users obtain a SINR greater than their minimum target-SINR in DTPC? if not, which user(s) obtain a SINR greater than the minimum target-SINR in DTPC?

در DTPC تمام کاربران به SINR ای بالاتر از Target SINR نمی توانند دست پیدا کنند، بلکه تنها کاربرانی که وضعیت کانالی خوبی دارند و در حقیقت می توان اینطور بیان کرد که تنها کاربرانی که مشابه الگوریتم OPC عمل می کنند که همان کاربران با وضعیت کانالی خوب هستند می توانند به SINRای بالاتر دست پیدا کنند و سایر کاربران در همان حد Target SINR باقی می مانند که این کاربران همان کاربران با وضعیت کانالی بعد هستند که در واقع مشابه الگوریتم TPC عمل کرده اند.

1-4 Could you modify the unconstrained DTPC so that all users benefit from available resources (i.e., all users obtain a SINR greater than the minimum target-SINR)

در این قسمت خواسته شده که آیا میتوان کاری کرد که کاربران از منابع در دسترس به گونه ای کنند. مثلا اینکه SINR همه کاربران بالاتر از target-SINR شود؟

خب در الگوریتم DTPC تعداد محدودی کاربر می توانند به SINR ای بالاتر از Target SINR دست پیدا کنند و اگر بخواهیم در حالتی که سیستم Feasible است، همه کاربران به صورت منصفانه مقدار SINRخود را افزایش دهند. یک راه حل می تواند این باشد مثلا بیایم SNR همه کاربران در یک عدد ثابت بزرگتر از ۱ ضرب کرد به گونه که بردار SNR بدست آمده هنوز f easi ble باشد. اما در این حالت هم چون به همه کاربران مقدار ثابتی اضافه می شود ممکن است منصفانه نباشد. روش دیگر به این صورت است که یک ضریبی را برای SNR هر کاربر در نظر بگیریم و این ضریب را ماکسیمایز کنیم.

2- Constrained Transmit Power Case

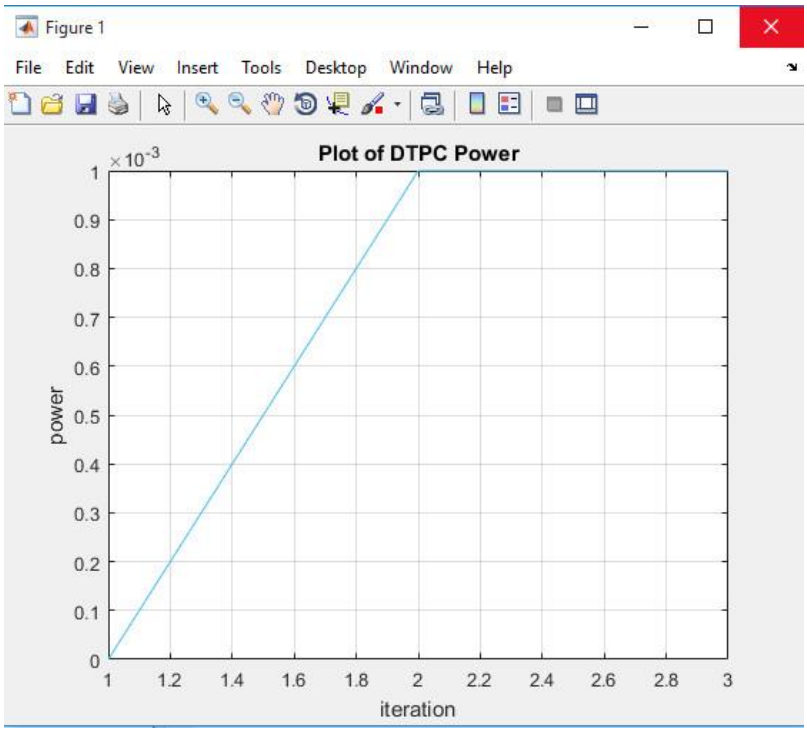
Now consider a constrained DTPC in which pi = 1 mW for all i.

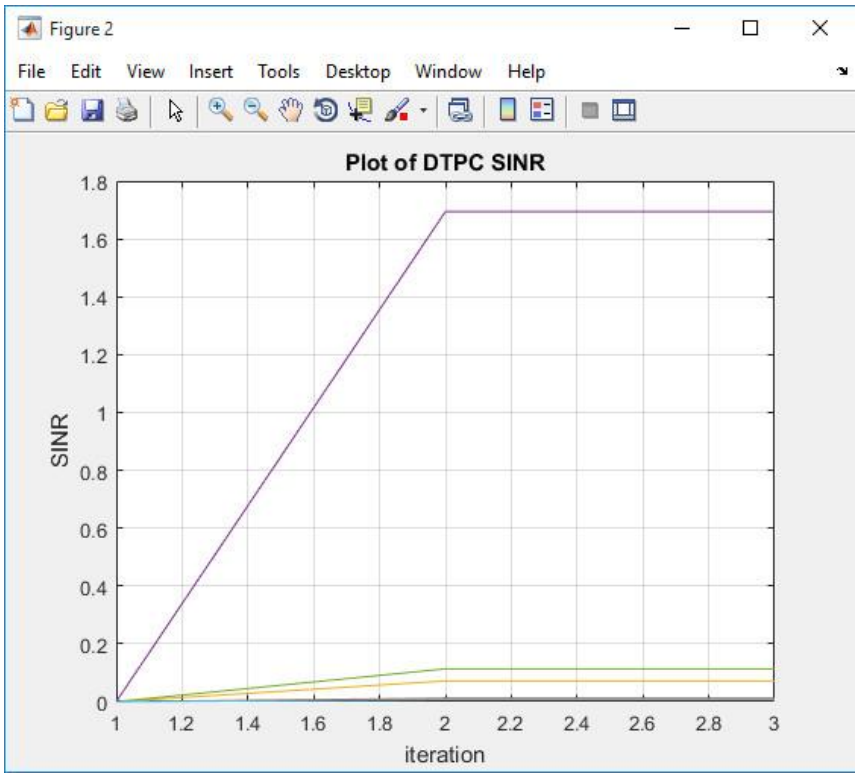
2-1 Repeat the simulation as above.

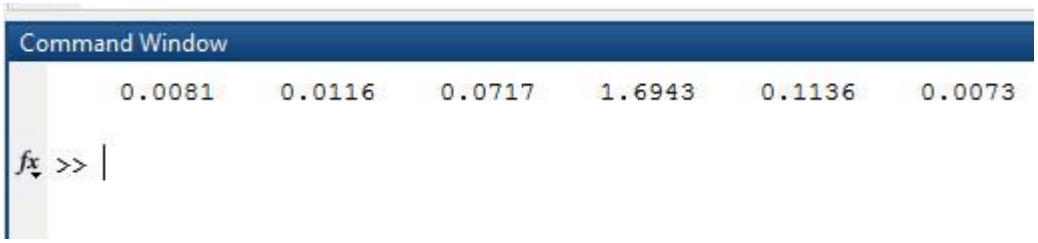
در این بخش توان کاربران محدود شده است و مقدار آن برابر است با ۱ میلی وات. در قسمت الگوریتم DTPC کد زیر اضافه شده است.

```
powerInitiate(j,i)=min(MaxP, max(eta*(sinr((j-1),i)/powerInitiate((j-
1),i)),powerInitiate((j-1),i)*(GAMA(i)/sinr((j-1),i))));
```

SNR هدف کاربران را در حالتی که 0.01 است به شکل زیر بدست امده است:







همانطور که در شکل دیده می شود دو تا از کاربران به snr هدفشان نرسیدند کاربر ۱ و کـاربر ۶. یعنـی در یک سیستم feasible هست اما دو تا کاربر به SNR هدفشان نرسیدند.

این بار سیستم را با SNR هدف 0.1 اجرا می کنیم و نمودار های زیر بدست می آید:

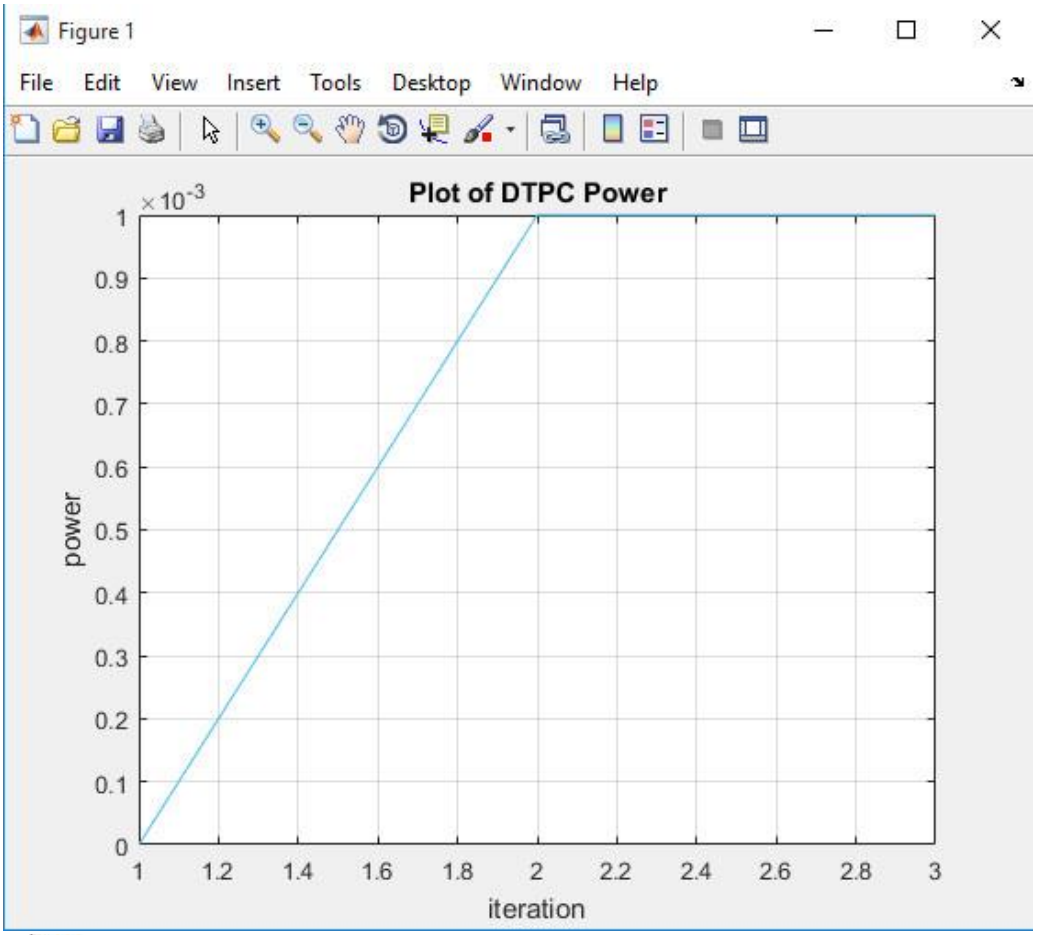


Figure 1
File Edit View Insert Tools Desktop Window Help
Plot of DTPC Power
×10^-3
power
iteration
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
1 1.2 1.4 1.6 1.8 2 2.2 2.4 2.6 2.8 3

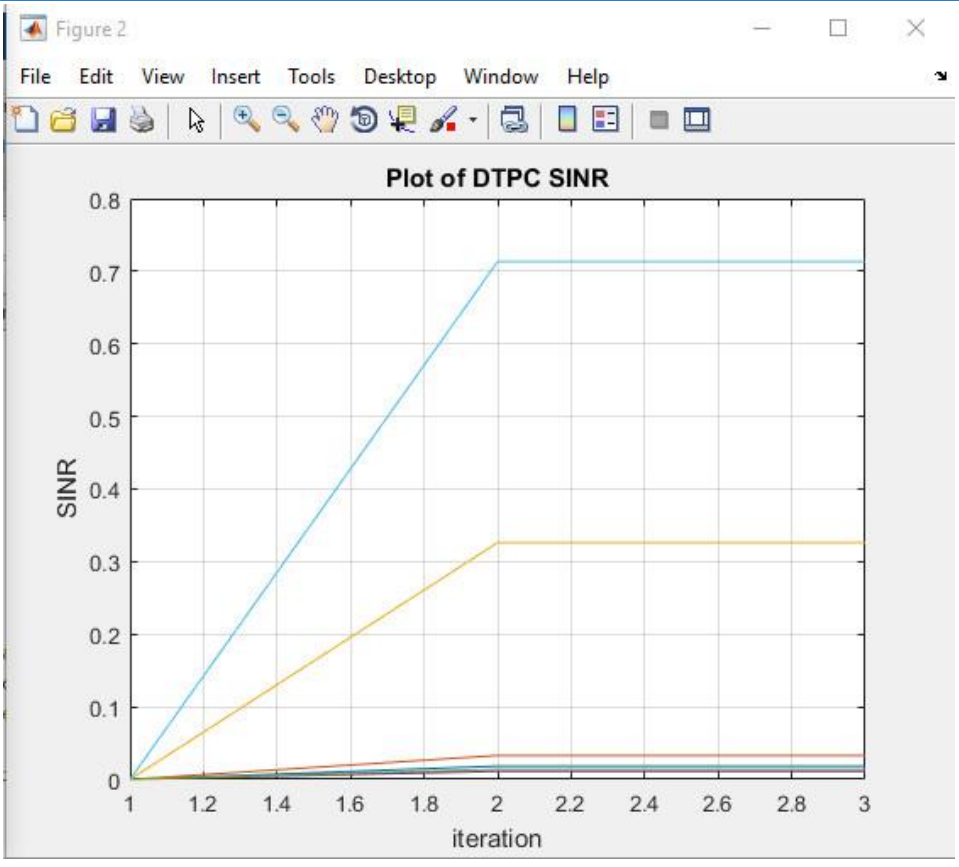
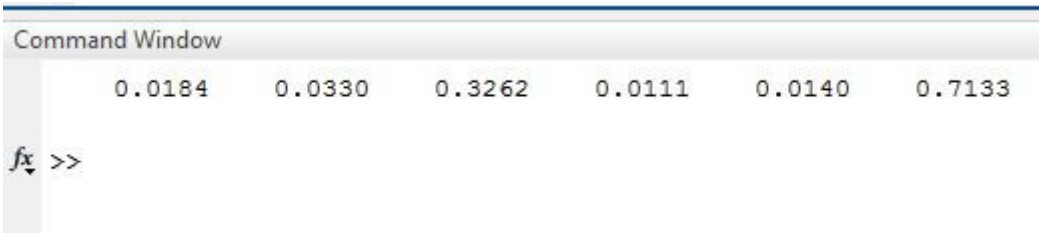


Figure 2
File Edit View Insert Tools Desktop Window Help
Plot of DTPC SINR
SINR
iteration
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8
1 1.2 1.4 1.6 1.8 2 2.2 2.4 2.6 2.8 3



مشاهده می شود کـه دو کـاربر بـه بیشـتر از SNR هدفشـان رسـیدند و ۴ کـاربر بـه SNR هدفشـان نرسیدند.

بر اساس نمودار های مختلف با SNR مختلف در حالتی که توان محدود است می توان گفت:

در الگوریتم DTPC هنگامی که توان محدود می شود امکان دارد بعضی از کاربران دیگر نتوانند به مقدار SNR هدفشان برسند . علت این امر این است که کاربر برای اینکه بتواند به SINR بالاتری دست پیدا کند، با توان بالایی در حال ارسال است که در نمودار مربوط به توان که در بالا آمده است، قابل ملاحظه است .از این رو این توان بالا باعث به وجود آمدن تداخل بر روی کاربران ضعیف می شود و در نتیجه این کاربر دیگر نتوانسته اند به میزان حداقلی SNR خودش نیز برسند.

در مورد گذردهی و outage در حالتی که سیستم توان محدود دارد هم می توان این موارد را پس از مشاهدات گفت:

وقتی سیستم feasible است، نرخ اخراج TPC همواره صفر است و Throughput سیستم نیز ثابت است و این موضوع مستقل از محدود بودن یا نبودن توان است.

در الگوریتم OPC بازهم نرخ اخراج بالاست و Throughput سیستم نسبتا زیاد است ولی به علت محدود بودن توان،Throughput کمتر از حالت توان نامحدود است.

در الگوریتم DTPC با توان محدود، چون تعدادی از کاربران با استفاده از OPC توان خود را افزایش می‌دهند تا به SINR های بالاتر دست پیدا کنند، تعدادی از کاربران که از وضعیت خوبی برخوردار نیستند، به توان ماکزیمم در TPC می‌رسند و در نتیجه نمی‌توانند به SINR حداقل خود برسند و بنابراین در DTPC با توان محدود ممکن است در سیستم feasible هم کاربرانی به SINR مطلوب نرسند و بنابراین همیشه نرخ اخراج صفر نیست.

در سیستم infeasible با توان محدود، همه الگوریتم‌ها نرخ اخراج مخالف صفر دارند .ولی الگوریتم DTPCنسبت به TPC دارای نرخ اخراج بیشتر و Throughput بیشتر است.

2-2 Does the constrained DTPC guarantee the minimum target-SINR for all users? why? Discuss and try to and a solution to resolve the problem.

خیر، زیرا در الگوریتم DTPC با توان محدود، تعدادی از کاربران با استفاده از OPC توان خود را افزایش می‌دهند تا به SINRهای بالاتر دست پیدا کنند، تعدادی از کاربران که از وضعیت خوبی برخوردار نیستند، به توان ماکزیمم درTPC می‌رسند و در نتیجه نمی‌توانند به SINR حداقل خود برسند و بنابراین در DTPCبا توان محدود ممکن است در سیستم feasible هم کاربرانی به SINRمطلوب نرسند و بنابراین همیشه نرخ اخراج صفر نیست . برای برطرف کردن این مشکل شاید یکی از راه حل ها این باشد که یک مقدار توان ماکسیمم دوم برای کاربر قوی که مشابه الگوریتم OPC عمل می کند تعریف نماییم و توان ارسالی آن را محدود کنیم تا از حدی فراتر نرود و باعث ایجاد افزایش تداخل روی سایر کاربران شود .اما پیدا کردن این حد آستانه یا مقدار دقیق توان ماکسیمم دوم، برای کاربر قوی نیاز به بررسی دقیق دارد.