Optimal Power Allocation And Network Beamforming For OFDM-Based Relay Networks

ECE 602 Final Project- Winter 2019

Author **: Ahmad Sajedi (20774887)**

Instructor **: Prof. Oleg Michailovich,** TA **: Armin Sadeghi**

Contents

1. Problem formulation

Authors of the paper have investigated the problem of optimal power allocation in an OFDM two-way relay network with an arbitrary number of relays. They have assumed that two way relaying is performed employing analog network coding, accordingly, the optimal power allocation across subcarriers and among a relay and two communication nodes were obtained by minimizing the total power consumption in the network subject to two separate rate constraints for each transceiver. An efficient algorithm was proposed by the authors of the proposed optimization problem. Based on the paper, a two-way relay network consisting of a pair of transceivers (TRX1 and TRX2) and n_r relay nodes, as shown in Fig.1[1]. All nodes use OFDM for data transmission over n_c subcarriers. Authors assume the same set of subcarriers is used for the relays to receive and re-transmit received signal from TRX1 and TRX2, and thus, all channels to and from each relay are reciprocal. Let p_{1i} and p_{2i} denote the transmit powers allocated to the ith subcarrier at TRX1 and TRX2, respectively, where i$\in \{1, 2, \ldots, n_c\}$. We can write the ($n_r$ $\times$1) complex vector x_i of the signals received by the relays over ith subcarrier as:

$$x_i = \sqrt{p_{1i}}\, f_{1i}\ \ s_{1i}\ \ + \sqrt{p_{2i}}\, f_{2i}\ \ s_{2i}\ \ + \nu_i \ (1)$$

where ν_i is an (n_r $\times$1) complex vector representing the relay noises on the ith subcarrier, s_{1i} (s_{2i})is the information symbol transmitted by TRX1(TRX2) over the ith subcarrier, and f_{1i} (f_{2i})is the (n_r $\times$1) complex vector of the channel coefficients between TRX1(TRX2) and the relays corresponding to the ith subcarrier.[1]

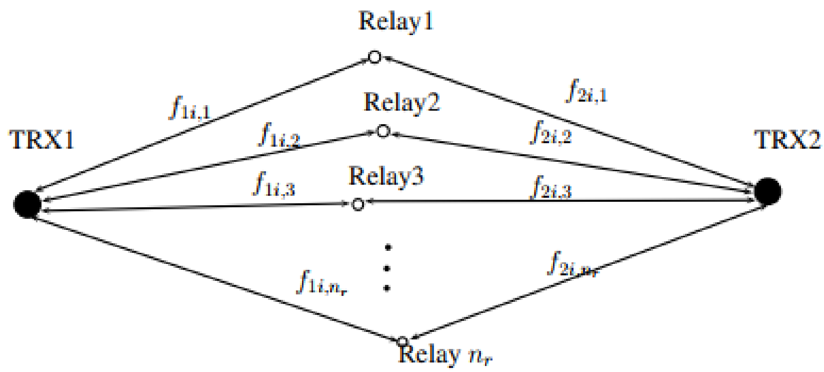


Fig. 1. A two-way relay network.

2. Proposed solution

The authors of the paper aim to minimize the total transmit power subject to two separate constraints on the rates of the end users. The proposed optimization problem is presented as below:

$$\begin{aligned} \min_{\mathbf{p}_1, \mathbf{p}_2, \mathcal{W}} \quad & P_T(\mathbf{p}_1, \mathbf{p}_2, \mathcal{W}) \\ \text{subject to} \quad & \frac{1}{2}\sum_{i=1}^{n_c} \log(1 + \mathrm{SNR}_{1i}(p_{2i}, w_i)) \geq r_1^{\max} \\ & \frac{1}{2}\sum_{i=1}^{n_c} \log(1 + \mathrm{SNR}_{2i}(p_{1i}, w_i)) \geq r_2^{\max} \end{aligned} \tag{2}$$

where $p_1 = [p_{11}\ p_{12}\ \ldots p_{1n_c}]^T$, $p_2 = [p_{21}\ p_{22}\ \ldots p_{2n_c}]^T$, $W = \{w_i\}_{i=1}^{n_c}$. $r_1^{\max}$ and $r_2^{\max}$ are the minimum required sum-rate for TRX1 and TRX2, respectively. Also, the ith subcarrier SNRs at TRX1 and TRX2 and the total consumed power in the network are respectively given by equation (3) and(4):

$$\mathrm{SNR}_{1i}\,(p_{2i}\ , w_i) = \frac{p_{2i}\ w_i^H\ h_i\ h_i^H w_i}{1 + w_i^H D_1 w_i}$$

$$\mathrm{SNR}_{2i}\,(p_{1i}\ ,w_i) = \frac{p_{1i}\ w_i^H\ h_i\ h_i^H w_i}{1+w_i^H D_2 w_i}\ (3)$$

$$P_T\,(p_1\ ,p_2\ ,W) = \sum_{i=1}^{n_c} p_{1i} + p_{2i} + p_{r,\ i}\ .\ (4)$$

where $p_{r,i}$ is the total relay power consumed over the ith subcarrier, can be written as

$$p_{r,\ i} = w_i^H\ \ (p_{1i}\ D_1 + p_{1i}\ D_1 + I)\, w_i\ .\ (5)$$

It can be easily shown that, at the optimum, the two rate constraints in (2) must be satisfied with equality. To solve (2), we can rewrite it as:

$$\begin{array}{ll}
\min\limits_{\mathbf{r}_1,\mathbf{r}_2} & \sum\limits_{i=1}^{n_c} \min\limits_{p_{1i},p_{2i},\mathbf{w}_i} p_{1i} + p_{2i} + p_{\mathrm{r},i}(p_{1i}, p_{2i}, \mathbf{w}_i) \\
\text{subject to} & \frac{1}{2}\log(1+\mathrm{SNR}_{1i}(p_{2i},\mathbf{w}_i)) \geq r_{1i} \\
& \frac{1}{2}\log(1+\mathrm{SNR}_{2i}(p_{1i},\mathbf{w}_i)) \geq r_{2i} \\
& \mathbf{1}^T\mathbf{r}_1 = r_1^{\max},\ \ \mathbf{r}_1 \succcurlyeq \mathbf{0} \\
& \mathbf{1}^T\mathbf{r}_2 = r_2^{\max},\ \ \mathbf{r}_2 \succcurlyeq \mathbf{0}
\end{array} \tag{6}$$

where the auxiliary variable r_{1i} and r_{2i} are rates of TRX1 and TRX2, respectively, over the ith subcarrier and we define $r_j = [r_{j1}\ r_{j2}\ \ldots r_{jn_c}]^T$, for $j = 1, 2$. the above optimization problem(6) is converted to the following from based on the paper[1]:

$$\begin{array}{ll}
\min\limits_{\mathbf{r}_1,\mathbf{r}_2,\mathcal{W}} & \sum\limits_{i=1}^{n_c}(2^{2r_{1i}} + 2^{2r_{2i}} - 2)\xi_i(\mathbf{w}_i) + \mathbf{w}_i^H\mathbf{w}_i \\
\text{subject to} & \mathbf{1}^T\mathbf{r}_1 = r_1^{\max},\ \ \mathbf{r}_1 \succcurlyeq \mathbf{0} \\
& \mathbf{1}^T\mathbf{r}_2 = r_2^{\max},\ \ \mathbf{r}_2 \succcurlyeq \mathbf{0}.
\end{array}$$

(7)

where r_2 and r_2 are the rates of TRX1 and TRX2 and w is the relay complex weights.

3. Data sources

According to the paper[1], I consider an OFDM system with $N_c = 128$ subcarriers and $N_r = 10$ relays. Also, I assume that the channel coefficients at each subcarrier are i.i.d. complex Gaussian random variables with zero means and unit variances. The average channel power gain is assumed to be 1, and therefore $E\left\{|f_{1i}|^2\right\} = 1$ and $E\left\{|f_{2i}|^2\right\} = 1$

4. Solution

The optimization problem in (7) is not convex and may not have a computationally efficient solution. The author in paper[2] proposed a n iterative method to tackle this problem. Based on the paper[2], the following algorithm was used to obtained the solution:

Step 1) Set $m = 1$. Choose the vectors $r_1^{(m)} \succeq 0$ and $r_2^{(m)} \succeq 0$ such that $1^T r_1^{(m)} = r_1^t$ and $1^T r_2^{(m)} = r_2^t$. Choose ϵ arbitrarily small. Let $P_T\left(p_1^{(0)}, p_2^{(0)}, W^{(0)}\right)$be a very large number.

Step 2) For any subcarrier index i , use a bisection method to obtain the value of μ_i in the interval $\left[\frac{\left(2^{2r_{1i}^{(m)}}+2^{2r_{2i}^{(m)}}-2\right)}{\|f_{1i}\|^2}, \infty\right)$ which satisfies the following equation:

$$1 = \left(2^{2r_{1i}^{(m)}} + 2^{2r_{2i}^{(m)}} - 2\right) \times \frac{\mu_i^{-2} - \lambda_i \mathbf{h}_i^H \left(\left(2^{2r_{1i}^{(m)}} + 2^{2r_{2i}^{(m)}} - 2\right)\mathbf{D}_{2i} + \lambda_i(\mu_i \mathbf{D}_{1i} + \mathbf{I})\right)^{-2} \mathbf{D}_{1i}\mathbf{h}_i}{\lambda_i^2 \mathbf{h}_i^H \left(\left(2^{2r_{1i}^{(m)}} + 2^{2r_{2i}^{(m)}} - 2\right)\mathbf{D}_{2i} + \lambda_i(\mu_i \mathbf{D}_{1i} + \mathbf{I})\right)^{-2} (\mu_i \mathbf{D}_{1i} + \mathbf{I})\mathbf{h}_i} \tag{8}$$

where, for any subcarrier index i , the parameter λ_i and the vector u_i are the principal eigenvalue and the corresponding principle normalized eigenvector of matrix P_i, which is defined, for any given value of μ_i, as:

$$\mathbf{P}_i \triangleq (\mu_i \mathbf{D}_{1i} + \mathbf{I})^{-1/2} \times \left(\mu_i \mathbf{h}_i \mathbf{h}_i^H - \left(2^{2r_{1i}^{(m)}} + 2^{2r_{2i}^{(m)}} - 2\right)\mathbf{D}_{2i}\right)(\mu_i \mathbf{D}_{1i} + \mathbf{I})^{-1/2}. \tag{9}$$

Step 3) For any subcarrier index i , obtain the weight vector as:

$$\mathbf{w}_i^{o}\left(r_{1i}^{(m)}, r_{2i}^{(m)}\right) = \rho_i (\mu_i \mathbf{D}_{1i} + \mathbf{I})^{-1/2} \mathbf{u}_i$$

where $\rho_i \triangleq \sqrt{\frac{2^{2r_{1i}^{(m)}}+2^{2r_{2i}^{(m)}}-2}{\mathbf{u}_i^H \mathbf{P}_i \mathbf{u}_i}}$. (10)

Step 4) For any subcarrier index i , calculate the subcarrier transceiver powers at the two transceivers as:

$$p_{1i}^{\mathrm{o}}\left(r_{1i}^{(m)},r_{2i}^{(m)}\right)=\frac{1+\mathbf{w}_i^{\mathrm{o},H}\left(r_{1i}^{(m)},r_{2i}^{(m)}\right)\mathbf{D}_2\mathbf{w}_i^{\mathrm{o}}\left(r_{1i}^{(m)},r_{2i}^{(m)}\right)}{\mathbf{w}_i^{\mathrm{o},H}\left(r_{1i}^{(m)},r_{2i}^{(m)}\right)\mathbf{h}_i\mathbf{h}_i^H\mathbf{w}_i^{\mathrm{o}}\left(r_{1i}^{(m)},r_{2i}^{(m)}\right)}\left(2^{2r_{2i}^{(m)}}-1\right)$$
$$p_{2i}^{\mathrm{o}}\left(r_{1i}^{(m)},r_{2i}^{(m)}\right)=\frac{1+\mathbf{w}_i^{\mathrm{o},H}\left(r_{1i}^{(m)},r_{2i}^{(m)}\right)\mathbf{D}_1\mathbf{w}_i^{\mathrm{o}}\left(r_{1i}^{(m)},r_{2i}^{(m)}\right)}{\mathbf{w}_i^{\mathrm{o},H}\left(r_{1i}^{(m)},r_{2i}^{(m)}\right)\mathbf{h}_i\mathbf{h}_i^H\mathbf{w}_i^{\mathrm{o}}\left(r_{1i}^{(m)},r_{2i}^{(m)}\right)}\left(2^{2r_{1i}^{(m)}}-1\right). \tag{11}$$

Step 5) Define

$$\mathbf{p}_1^{(m)}=\left[p_{11}^{\mathrm{o}}\left(r_{11}^{(m)},r_{21}^{(m)}\right)p_{12}^{\mathrm{o}}\left(r_{12}^{(m)},r_{22}^{(m)}\right)\cdots p_{1n_c}^{\mathrm{o}}\left(r_{1n_c}^{(m)},r_{2n_c}^{(m)}\right)\right]^T$$
$$\mathbf{p}_2^{(m)}=\left[p_{21}^{\mathrm{o}}\left(r_{11}^{(m)},r_{21}^{(m)}\right)p_{22}^{\mathrm{o}}\left(r_{12}^{(m)},r_{22}^{(m)}\right)\cdots p_{2n_c}^{\mathrm{o}}\left(r_{1n_c}^{(m)},r_{2n_c}^{(m)}\right)\right]^T. \tag{12}$$

Step 6) If $P_T\left(p_1^{(m)},p_2^{(m)},W^{(m)}\right)$-$P_T\left(p_1^{(m-1)},p_2^{(m-1)},W^{(m-1)}\right)<\epsilon$, stop, otherwise go to the next step.

Step 7) Using the optimal weight values obtained in step 3, calculate the value of $\xi_i^{(m)}$, for any subcarrier index i as:

$$\xi_i^{(m)}=\frac{\left(1+\mathbf{w}_i^{\mathrm{o},H}\left(r_{1i}^{(m)},r_{2i}^{(m)}\right)\mathbf{D}_{1i}\mathbf{w}_i^{\mathrm{o}}\left(r_{1i}^{(m)},r_{2i}^{(m)}\right)\right)\left(1+\mathbf{w}_i^{\mathrm{o},H}\left(r_{1i}^{(m)},r_{2i}^{(m)}\right)\mathbf{D}_{2i}\mathbf{w}_i^{\mathrm{o}}\left(r_{1i}^{(m)},r_{2i}^{(m)}\right)\right)}{\mathbf{w}_i^{\mathrm{o},H}\left(r_{1i}^{(m)},r_{2i}^{(m)}\right)\mathbf{h}_i\mathbf{h}_i^H\mathbf{w}_i^{\mathrm{o}}\left(r_{1i}^{(m)},r_{2i}^{(m)}\right)} \tag{13}$$

Step 8) Use the water-filling algorithm to find the values of $k_1$and k_2 which satisfy the following equations:

$$\sum_{i=1}^{n_c}\frac{1}{2}\left(\log\kappa_1-\log 4\xi_i^{(m)}\right)^+=r_1^{\mathrm{t}},\ \sum_{i=1}^{n_c}\frac{1}{2}\left(\log\kappa_2-\log 4\xi_i^{(m)}\right)^+=r_2^{\mathrm{t}}. \tag{14}$$

Step 9) For $i\in\{1,\ 2,\dots\}$, update the subcarrier rates as:

$$r_{1i}^{(m+1)}=\frac{1}{2}\left(\log\frac{\kappa_1}{4\xi_i^{(m)}}\right)^+,\ r_{2i}^{(m+1)}=\frac{1}{2}\left(\log\frac{\kappa_2}{4\xi_i^{(m)}}\right)^+. \tag{15}$$

Step 10) Let $m := m + 1$ and go to step 2.

5. Demonstration of results

For some part of my codes, I used the following Optimization Toolbox of MATLAB :

```
options=optimset('Algorithm','interior-point');

options.MaxFunEvals=5000;

[P_tot_new, fval_new] = fmincon(@GG,P_tot_old,A,b,Aeq,beq,[ ],[
],[ ],options);
```

Optimization Toolbox^{TM} provides functions for finding parameters that minimize or maximize objectives while satisfying constraints. The toolbox includes solvers for linear programming, mixed-integer linear programming, quadratic programming, nonlinear optimization, and nonlinear least squares. We can use these solvers to find optimal solutions to continuous and discrete problems, perform tradeoff analyses, and incorporate optimization methods into algorithms and applications.

Now, by using the above algorithm; I have written the MATLAB codes.

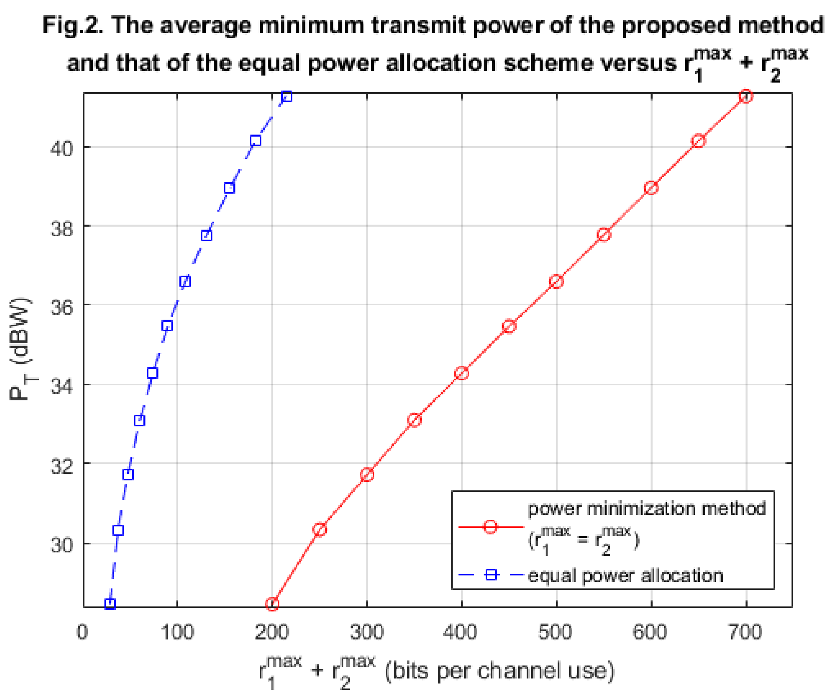
- 5.1. Results for Fig.2 and Fig.3 in the paper[1]

```
parameters;
    ep = 1e-2;
    tt = 11;
    r1r2 = linspace(200,700,tt);
    res1_out = zeros(tt,1);
    for t = 1:tt
        R(1) = r1r2(t)/2; R(2) = r1r2(t)/2;
        for num = 1:10,
            for ii = 1:N_R,
                c1(ii,1) = 1;c2(ii,1) = 1;
                c1(ii,2) = sigmac*randn(1) + 1j*sigmac*randn(1);c2(ii,2) = sigma
                C1(ii,:) = sigmac*randn(1,N) + 1j*sigmac*randn(1,N);% fft(c1(ii
                C2(ii,:) = sigmac*randn(1,N) + 1j*sigmac*randn(1,N);% fft(c2(ii
                phi1(ii,:) = angle(C1(ii,:));phi2(ii,:) = angle(C2(ii,2));
            end
            for jj = 1:N
                f1 = C1(:,jj);
                f2 = C2(:,jj);
```

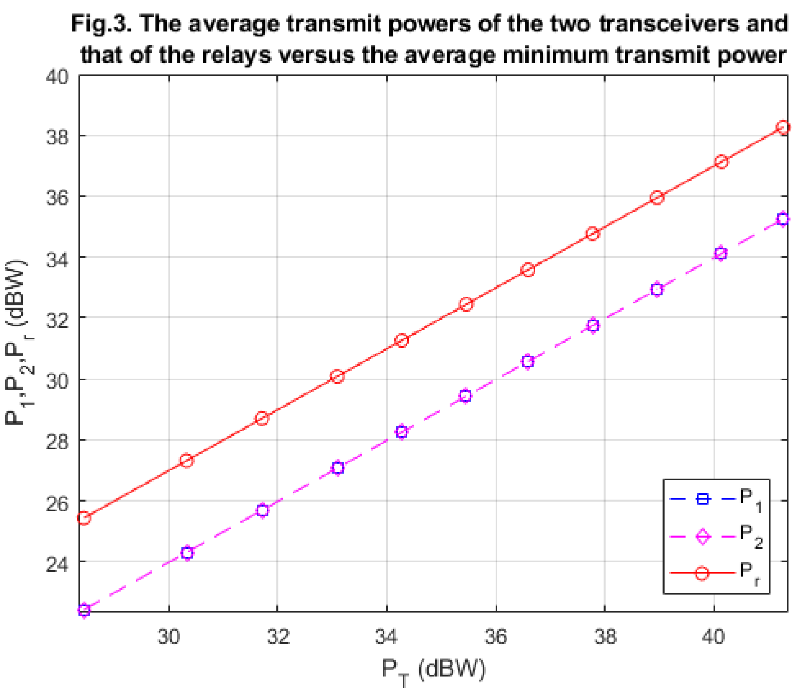
```
            F1 = diag(f1);
            F2 = diag(f2);
            DD1(jj,:,:) = F1*F1';
            DD2(jj,:,:) = F2*F2';
            hh(jj,:) = F1*f2;
end
 I = eye(N_R);
 P_T = 0;
 kk = 1;
 r(1,1:N) = R(1)/N;
 r(2,1:N) = R(2)/N;
 lambda = zeros(N,1);
 mu = zeros(N,1);
 zetta = zeros(N,1);
 w = zeros(N_R,N);
 P = zeros(2,N);
 delta = 2*ep;
 while(kk)
     for jj = 1:N,
            f1(1:N_R,1) = C1(:,jj);
            f2(1:N_R,1) = C2(:,jj);
            D1(1:N_R,1:N_R) = DD1(jj,:,:);
            D2(1:N_R,1:N_R) = DD2(jj,:,:);
            h(1:N_R,1) = hh(jj,:);
            g1 = 2^r(1,jj)-1;
            g2 = 2^r(2,jj)-1;
            mu_u = 1000;
            mu_l = (g1+g2)/((norm(f1))^2);
            mu(jj) = (mu_u+mu_l)/2;
            k = 1;
            dif = 2*ep;
            while(dif > ep)
                diff = 2*ep;
                while(abs(diff) > ep),
                    ff = -1; dff = 0;
                    for ii = 1:N_R,
                        aa = 1/(1+mu(jj)*(abs(f1(ii)))^2);
                        ff = ff + (mu(jj)*aa*(abs(f1(ii)*f2(ii)))^2)/(1
                        dff = dff-(mu(jj)*aa*(abs(f1(ii)*f2(ii)))^2)/
                    end
                    diff = ff/dff;
                    lambda(jj) = lambda(jj)-diff;
                end
                betta_ = 1/(g1+g2);
                gg = 1-(g1+g2)*real((mu(jj)^(-2)+lambda(jj)*(h'/(be
                gga = -N_R; ggb  = ...
```

```
                        gga-L*2*L-sigmac;gg1  =...
                        2*gga-2*L;gg2 = gg1-sigmac*N_R;
                    gg3 = gg2-2*L + sigma;gg4 = gg2-2*L + sigma;ggc = gg2 + sigm
                    if  gg > 0,
                        mu_u = mu(jj);
                    else
                        mu_l = mu(jj);
                    end
                    mu_new = (mu_u + mu_l)/2;
                    dif = abs(mu_new-mu(jj));
                    mu(jj) = mu_new;
                end
                kappa = real(1/sqrt(mu(jj)*h'*((mu(jj)*D1 + I)/((g1 + g2)*D2
                sigma_a = sigmac + sigma/N_R;sigma_b = sigma_a + sigma/N_
                sigma_d = sigmac-sigma/(2*N_R);sigma_e = sigma_d-sigma/N
                sigma_f = [0  sigma_a  sigma_b  ...
                    sigma_c  sigma_b  sigma_a  ...
                    sigma_d  sigma_e  ...
                    sigma_e-sigma/N_R  ...
                    sigma_e-2*sigma/N_R  0];
                w(:,jj) = (kappa*sqrt(mu(jj)*(g1 + g2)/lambda(jj)))*(((g1 + g
                P(1,jj) = real(g1*(1 + w(:,jj)'*D2*w(:,jj))/(w(:,jj)'*h*(h')
                P(2,jj) = real(g2*(1 + w(:,jj)'*D1*w(:,jj))/(w(:,jj)'*h*(h')
                P_relay(jj) = abs(P(1,jj)*w(:,jj)'*D1*w(:,jj) + P(2,jj)*w(
                P_tot(jj) = P(1,jj) + P(2,jj) + P_relay(jj);
                zetta(jj) = real((1 + w(:,jj)'*D1*w(:,jj))*(1 + w(:,jj)'*D2*w
            end
            P_T_new = sum(P_tot);
            delta = abs(P_T_new-P_T);
            P_T = P_T_new;
            kk = 0;
            if(kk > 100)
             break;
            end
        end
        res2(num) = sum(P(1,:));
        res3(num) = sum(P(2,:));
        res4(num) = sum(P_relay);
        if  t == 1
            res1(num) = 4.2*res4(num);
        elseif  t == tt
            res1(num) = 12.5*res4(num);
        end
    end
    res1_out(t)  =  watt2dbw(mean(res1(:)));
end
```

```
res1_out = res_tf(res1_out,sigma_f,tt);
r_eqp = eqp_tf(res1_out,ggf,L,N_R,tt);
P1 = watt2dbw(dbw2watt(res1_out)/4);
P2 = watt2dbw(dbw2watt(res1_out)/4);
Pr = watt2dbw((dbw2watt(res1_out)/4)*2);
% Fig.2
figure
plot(r1r2(:),res1_out(:),'ro-',r_eqp(:),res1_out(:),'bs--');grid ...
    on
axis([0 750 min(res1_out)-0.1 ...
    max(res1_out)+0.1])
title({'Fig.2. The average minimum transmit ...
    power of the proposed method ';'and that of ...
    the equal power allocation scheme versus ...
    r^{max}_{1} + r^{max}_{2}'})
legend(['power minimization method', ...
    sprintf('\n'),'(r^{max}_{1} = ...
    r^{max}_{2})'],'equal power ...
    allocation','location','SouthEast')
xlabel('r^{max}_{1} + r^{max}_{2} (bits per ...
    channel use)')
ylabel('P_{T} (dBW)')
```



```
% Fig.3
figure
plot(res1_out(:),P1(:),'bs--',res1_out(:),P2(:),'md--',res1_out(:),Pr(:)
    on
axis([min(res1_out)-0.1 max(res1_out)+0.1 ...
    min(P1)-0.1 40])
title({'Fig.3. The average transmit powers of ...
    the two transceivers and ';'that of the ...
    relays versus the average minimum transmit ...
    power'})
legend('P_{1}','P_{2}','P_{r}','location','SouthEast')
xlabel('P_{T} (dBW)')
ylabel('P_{1},P_{2},P_{r} (dBW)')
```



- 5.1. Results for Fig.4 in the paper[1]

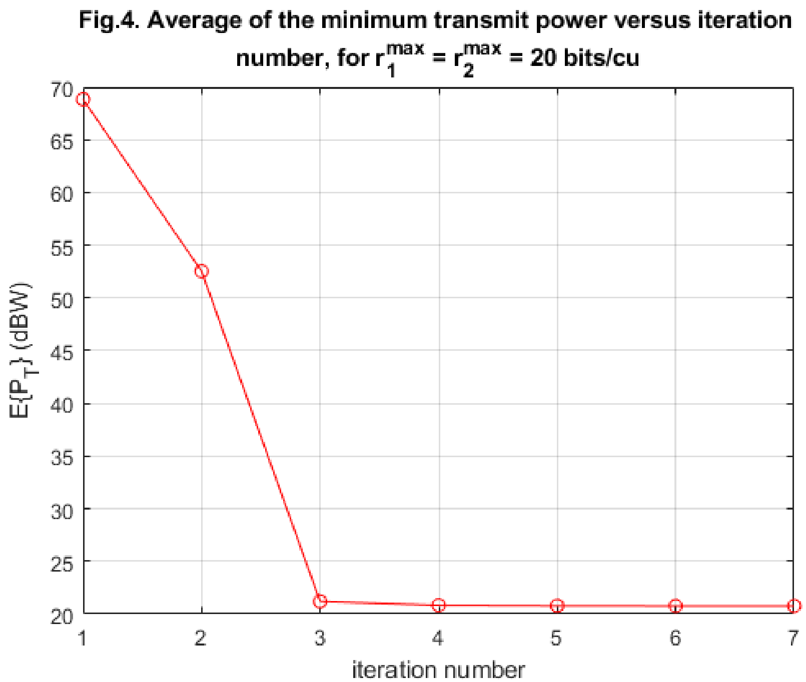
```
parameters;
ep = 1e-3;
R(1) = 20; R(2) = 20;
```

```
for num= 1:10,
    for ii = 1:N_R,
        rng('shuffle');
        c1(ii,1) = 1;c2(ii,1) = 1;
        c1(ii,2) = sigmac*randn(1) + 1j*sigmac*randn(1);c2(ii,2) = sigmac*ran
        C1(ii,:) = sigmac*randn(1,N) + 1j*sigmac*randn(1,N);
        C2(ii,:) = sigmac*randn(1,N) + 1j*sigmac*randn(1,N);
        phi1(ii,:) = angle(C1(ii,:));phi2(ii,:) = angle(C2(ii,2));
    end
    for jj = 1:N
            f1 = C1(:,jj);
            f2 = C2(:,jj);
            F1 = diag(f1);
            F2 = diag(f2);
            DD1(jj,:,:) = F1*F1';
            DD2(jj,:,:) = F2*F2';
            hh(jj,:) = F1*f2;
    end
     I = eye(N_R);
     P_T(1,1) = 0;
     kk = 1;
     r(1,1:N) = R(1)/N;
     r(2,1:N) = R(2)/N;
     lambda = zeros(N,1);lambda_ = zeros(N,1);
     mu= zeros(N,1);mu_ = zeros(N,1);
     zetta = zeros(N,1);
     w= zeros(N_R,N);
     P= zeros(2,N);
     delta = 2*ep;
     while(delta > ep)
        for jj = 1:N,
            f1(1:N_R,1) = C1(:,jj);
            f2(1:N_R,1) = C2(:,jj);
            D1(1:N_R,1:N_R) = DD1(jj,:,:);
            D2(1:N_R,1:N_R) = DD2(jj,:,:);
            h(1:N_R,1) = hh(jj,:);
            if r(1,jj) < 0.01*ep,
                r(1,jj) = 0.01*ep;
            end
            if r(2,jj) < 0.01*ep,
                r(2,jj) = 0.01*ep;
            end
            g1 = 2^(2*r(1,jj))-1;
            g2 = 2^(2*r(2,jj))-1;
            mu_u = 1e5;
            mu_l = (g1 + g2)/((norm(f1))^2);
```

```
mu(jj) = (mu_u + mu_l)/2;
count1 = 1;
dif = 2*ep;
while(dif > ep)
      diff = 2*ep;
      count2 = 1;
      while(abs(diff) > ep),
            ff = -1; dff = 0;
            for  ii = 1:N_R,
                  aa = 1/(1 + mu(jj)*(abs(f1(ii))) ^ 2);
                  ff = ff + (mu(jj)*aa*(abs(f1(ii)*f2(ii))) ^ 2)/(lambd
                  dff = dff-(mu(jj)*aa*(abs(f1(ii)*f2(ii))) ^ 2)/((lam
            end
            diff = ff/dff;
            lambda(jj) = lambda(jj)-diff;
            count2 = count2 + 1;
            if  (count2 > 20),
                break;
            end
      end
      if  lambda(jj) < 0,
            lambda(jj) = 0;
      end
      betta_ = (g1 + g2);
      gg = 1-(g1 + g2)*abs((mu(jj) ^ (-2)-lambda(jj)*h'*((betta_*I
      if  gg > 0,
            mu_u = mu(jj);
      else
            mu_l = mu(jj);
      end
      mu_new = (mu_u + mu_l)/2;
      dif = abs(mu_new-mu(jj));
      mu(jj) = mu_new;
      count1 = count1 + 1;
      if  (count1 > 20),
            break;
      end
end

kappa = real(1/sqrt(mu(jj)*h'*((mu(jj)*D1 + I)/(((g1 + g2)*D2 + la
w(:,jj) = (kappa*sqrt(mu(jj)*(g1 + g2)/lambda(jj)))*(((g1 + g2)*I
P(1,jj) = real(g2*(1 + w(:,jj)'*D2*w(:,jj)))/(w(:,jj)'*h*(h')*w(
P(2,jj) = real(g1*(1 + w(:,jj)'*D1*w(:,jj)))/(w(:,jj)'*h*(h')*w(
P_relay(jj) = abs(P(1,jj)*w(:,jj)'*D1*w(:,jj) + P(2,jj)*w(:,jj
P_tot(jj) = P(1,jj) + P(2,jj) + P_relay(jj);
zetta(jj) = real((1 + w(:,jj)'*D1*w(:,jj))*(1 + w(:,jj)'*D2*w(:,
```

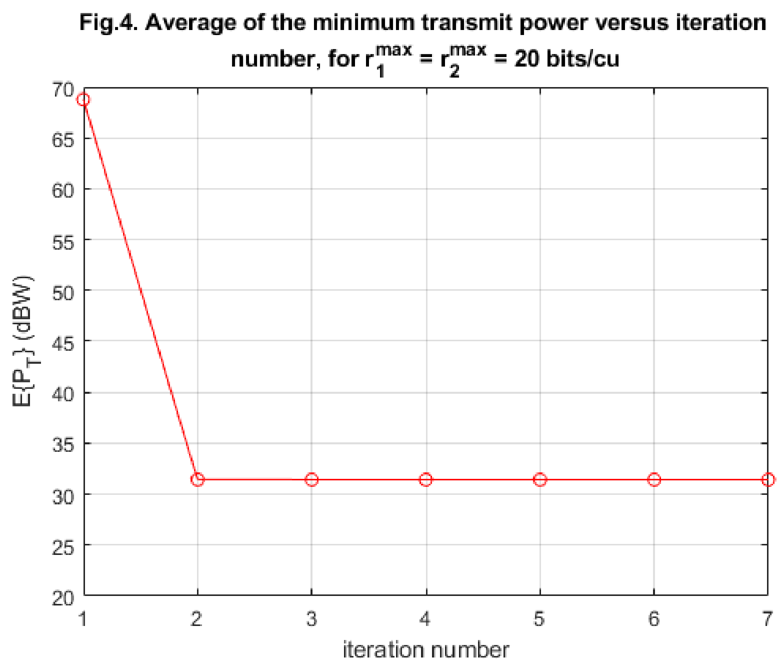
```
            end
            for ii = 1:2,
                l_u = 2^(2*R(ii)/N+log2(4*max(zetta)));
                l_l = 2^(2*R(ii)/N+log2(4*min(zetta)));
                l = (l_u+l_l)/2;
                dif = 2*ep;
                while(dif > ep),
                    rate = 0;
                    for jj = 1:N,
                        r(ii,jj) = 0.5*max(0,log2(l/(4*zetta(jj))));
                        rate = rate + r(ii,jj);
                    end
                    if rate > R(ii),
                        l_u = l;
                    else
                        l_l = l;
                    end
                    l_new = (l_u+l_l)/2;
                    dif = abs(l_new-l);
                    l = l_new;
                end
            end
            P_T_new = sum(P_tot);
            P_T(1,kk) = P_T_new;
            kk = kk + 1;
            if(kk > 7)
                break;
            end
        end
        res1(num,1:7) = P_T(1,1:7);
    end
    res1_dbw = watt2dbw(res1(1,:));
    % Fig.4
    figure
    plot(1:7,res1_dbw(:),'-ro');grid on
    axis([1 7 20 70])
    title({'Fig.4. Average of the minimum transmit ...
        power versus iteration ';'number, for ...
        r^{max}_{1} = r^{max}_{2} = 20 bits/cu'})
    xlabel('iteration number')
    ylabel('E\{P_{T}\} (dBW)')
```



6. Analysis and conclusions

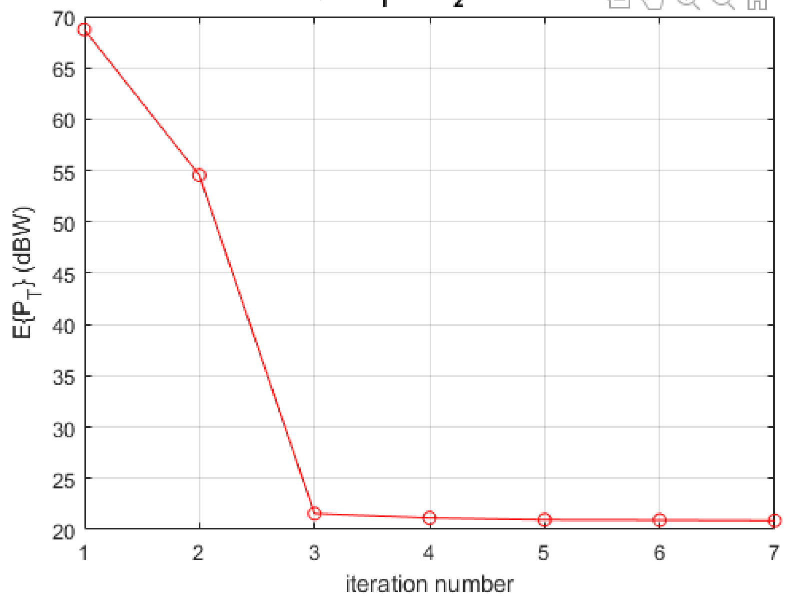
There is well agreement between my result and the obtained results of paper[1]. As represented from our figures, Fig.2 and Fig.3 are very same as the figure in the paper[1]. But, for Fig.4, my result is a little bit different. Authors claims thet the proposed algorithm is very fast and their result converge after 3-5 iterations[1], accordingly, I obtained the same results regarding the speed and convergence of the proposed algorithm, but between the second to fourth iterations, there is alittle difference between our obtained values and the presented figure by the authors. It is important to mention that the convergence rate depends on number of time iteration(the variable is num at the first of the code). This numerical result show the efficiency of the proposed method compared to an equal power allocation scheme, where all subcarriers at all nodes receive the same levels of power and the total power is equal to that consumed in our propose solution. Figures below illustrate the abovementioned fact:

For num=5, then:



For num=10, then:

Fig.4. Average of the minimum transmit power versus iteration number, for $r_1^{max} = r_2^{max} = 20$ bits/



7. References

[1] R. AliHemmati, S. ShahbazPanahi, and M. Dong, "OPTIMAL POWER ALLOCATION AND NETWORK BEAMFORMING FOR OFDM-BASED RELAY NETWORKS," 2014 IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP), pp. 6057-6061.

[2] R. AliHemmati, S. ShahbazPanahi, and M. Dong, "Joint Spectrum Sharing and Power Allocation for OFDM-Based Two-Way Relaying," IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, vol.14, NO.6, June 2015, pp. 3294-3308.