King Fahd University of Petroleum & Minerals
Computer Engineering Dept

Water Filling Algorithm for OFDMA Systems

Ashraf S. Hasan Mahmoud
Email: ashraf@kfupm.edu.sa

Optimal Power Allocation – Water Filling

- Water filling procedure provides the optimal power allocation for a given subchannel assignment.
- The cost function specified on slide 9 along with the constraints is given by

$$L = \sum_{k=1}^{K} \sum_{n \in \Omega_k} \frac{1}{N} \log_2\left(1 + p_{k,n} H_{k,n}\right) + \lambda_1 \left(\sum_{k=1}^{K} \sum_{n \in \Omega_k} p_{k,n} - P_{\text{total}} \right)$$

$$+ \sum_{k=2}^{K} \lambda_k \left(\sum_{n \in \Omega_1} \frac{1}{N} \log_2\left(1 + p_{1,n} H_{1,n}\right) - \frac{\gamma_1}{\gamma_k} \sum_{n \in \Omega_k} \frac{1}{N} \log_2\left(1 + p_{k,n} H_{k,n}\right) \right)$$

- Differentiating L with respect to the unknown $p_{k,n}$ results in the following equality

$$\frac{H_{k,m}}{1 + H_{k,m} p_{k,m}} = \frac{H_{k,n}}{1 + H_{k,n} p_{k,n}} \qquad \forall m, n \in \Omega_k, k = 1, 2, \cdots, K$$

or

$$\frac{1}{H_{k,m}} + p_{k,m} = \frac{1}{H_{k,n}} + p_{k,n} = \text{constant} \qquad \forall m, n \in \Omega_k, k = 1, 2, \cdots, K$$

- This constant is referred to as the "water level"

Water Filling Algorithm

- Given
  - A set of channel gains: $H_1, H_2, \ldots, H_n$
  - A total power P_{tot}
- Required – user water filling to find optimal allocations $p_1, p_2, \ldots, p_n$.

- Solution:
- we must find $p_1, p_2, \ldots, p_n$ such that

$$1/H_1 + p_1 = 1/H_2 + p_2 = \ldots = 1/H_n + p_n = C$$

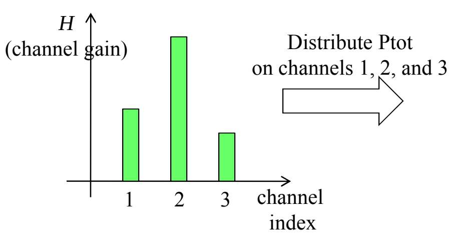
  - where $p_1 + p_2 + \ldots + p_n = P_{tot}$, and
  - C is the water level

- Note the $p_i = \max(C - 1/H_i, 0) = (C-1/H_i)^+$ for $i = 1, 2, \ldots, n$.

This is what Mohanram is using in step 4(e) of his algorithm!!

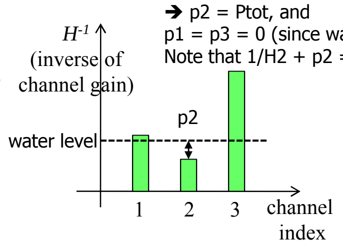
Water Filling Algorithm (2)

- Consider the following examples:



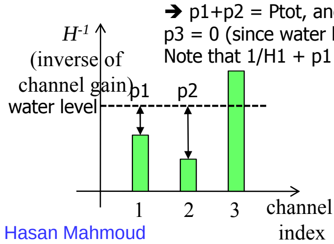
Case 1: Ptot small

→ $p_2 = P_{tot}$, and
$p_1 = p_3 = 0$ (since water level is low)!!
Note that $1/H_2 + p_2$ = water level



Case 2: Ptot medium

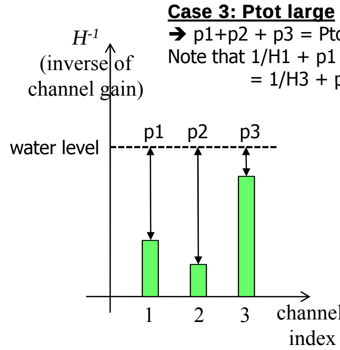
→ $p_1+p_2 = P_{tot}$, and
$p_3 = 0$ (since water level is not enough)
Note that $1/H_1 + p_1 = 1/H_2 + p_2$ = wat



Water Filling Algorithm (3)

- Observation on water filling
  - The algorithm starts by allocating power to the strongest channels
  - The stronger channel is always allocated power greater than that allocated for relatively weaker channel
  - If Ptot is not sufficient (i.e. water level is not high), for weak channels $(C - 1/Hi)^+$ will be equal to zero – i.e. the power allocation will be zero.

- We need a Matlab code that takes a vector of Hs and Ptot as input and returns the corresponding vector of power allocations, Ps computed as per the water filling algorithm.



Case 3: Ptot large

➔ p1+p2 + p3 = Ptot, and
Note that 1/H1 + p1 = 1/H2 + p2
= 1/H3 + p3 = water level

Note: this code implements waterfilling – it does not implement the joint power and subchannel algorithm proposed by Mohanram and Bhashyam.

Water Filling Algorithm (4) – Matlab Code for Calculation of Pi

```
function [Ps, C] = MyWaterFilling(Hs, Ptot);
% return Ps - the power allocations corresponding to the Hs
%        C  - the water level
n  = length(Hs);
Ps = zeros(size(Hs));

if (Ptot == 0)
    return; % will be zero for all channels'
end
% Ptot is NOT equal to zero –
%the at least the strongest channel will have power!!
[SortedHs, Indices]  = sort(Hs); % store the indices
C = (Ptot + sum(1./SortedHs))/n;
P = C - 1./SortedHs; % temporary power calculation
Sign = (P > 0); k = 0; % test for elimination of weak channels
while (sum(Sign) ~= (n-k)) && (k <= n)
    % eliminate the weakest channel
    k = k + 1; SHT = SortedHs(k+1:n);
    C = (Ptot + sum(1./SHT))/(n-k);
    P = C - 1./SHT;
    Sign = (P > 0);
end
Ps(Indices(k+1:n)) = P;
%fprintf('strongest k           = %3d users were allocated
     power\n', n-k);
%fprintf('Water level           = %7.3f\n', C);
%fprintf('Total allocated power = %7.3f\n', sum(P));
%fprintf('Allocations: ');
%for i=k+1:n, fprintf('P[%2d] = %7.3f, ', i-k, P(i-k)); end
%fprintf('\n');
```

The idea of the code is as follows:

- Sort the channel gains, Hs
- Since $pi = (C - 1/Hi)^+$ for I =1, 2, …, n

and $\Sigma pi = Ptot$, then

$$\Sigma pi = n C - \Sigma (1/Hi) = Ptot,$$

therefore,

$$C = (Ptot + \Sigma (1/Hi))/n$$

This is provided that C is the true water level and all pi's are positive.

Therefore, we iteratively compute C and pi's until all pi's are positive. For every failed, iteration we eliminate the weakest channel out of the remaining channels

Note that if Ptot is NOT zero, then at least we can allocate the entire Ptot to the strongest channel!!

Water Filling Algorithm in Mohanram's Paper

- The channel gains are denoted by γk,n
- Every allocated channel comes with Ptotal/N share – these shares are accumated in Pk, which is the total power for the kth user.
- Water filling is used to distribute this Pk over the channels owned by the kth user (step 4(e).
    - γ is the water level (previously called constant C)

The joint subcarrier and power allocation strategy is as follows.

1. Initialize $A = \{1, 2, 3, ...N\}$
2. $\forall\, k = 1$ to K, $A_k = \phi, P_k = 0$
3. $\forall\, k = 1$ to K,
   (a) $\gamma_k = \max_n \gamma_{k,n}$ for $n \in A$
   (b) $A_k = A_k \cup \{n\}$, $P_k = P_k + \frac{P_{total}}{N}$
   (c) $R_k = \log_2(1 + P_k \gamma_k)$
   (d) $A = A - \{n\}$
4. While $A \neq \phi$,
   (a) find i such that $\frac{R_i}{\alpha_i} \leq \frac{R_k}{\alpha_k}\ \forall\, k, i = 1$ to K
   (b) for the above i, find n such that
   $$\gamma_{i,n} \geq \gamma_{i,m}\ \forall\, n, m \in A$$
   (c) $A_i = A_i \cup \{n\}$, $P_i = P_i + \frac{P_{total}}{N}$
   (d) $A = A - \{n\}$
   (e) $R_i = \sum_{n \in A_i} log_2(1 + P_{i,n}\gamma_{i,n})$ where $P_{i,n} = \left(\gamma - \frac{1}{\gamma_{i,n}}\right)^+$ and $\sum_{n \in A_i} P_{i,n} = P_i$

The $f(x) = (x)^+$ operator indicates that $f(x) = 0$ when $x < 0$ and $f(x) = x$ when $x \geq 0$.

References

- C. Mohanram, S. Bhashyam, "A Sub-optimal Joint Subcarrier and Power Allocation Algorithm for Multiuser OFDM," IEEE COMMUNICATIONS LETTERS, VOL. 9, NO. 8, AUGUST 2005, pp. 685-687.