

ODFM 频域模型验证

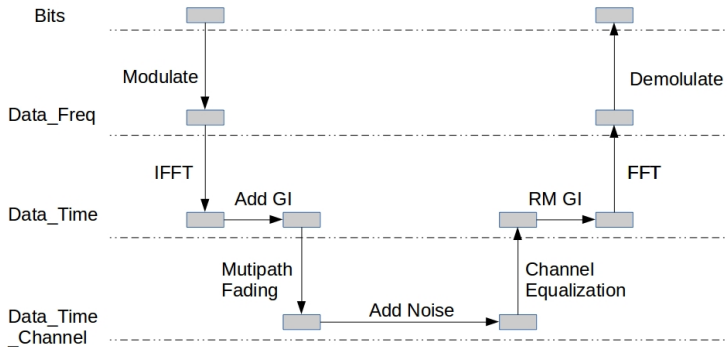
李昊

2018 年 6 月 20 日

通信系统仿真大作业

整体框架

整体框架-流程描述



既然是频域 -> 时域 -> 频域模型，
那么能否将时域部分封装为一个频域的矩阵，
而不必从时域来考虑问题？

$$\hat{Y} = FHF^{-1}X + FW$$

$$Y = \hat{Y} * \frac{H_0^*}{||H_0||^2}$$

$$H_0 = \frac{P_{Re}}{P_{Tr}}$$

数字实现

方法一：加循环前缀和循环延时

$HX =$

$$\begin{pmatrix} h(Q-1) & \dots & h(0) & \dots & 0 & 0 & 0 & \dots & 0 \\ 0 & h(Q-1) & \dots & h(0) & \dots & 0 & 0 & 0 & \dots \\ \dots & 0 & h(Q-1) & \dots & h(0) & \dots & 0 & 0 & 0 \\ 0 & \dots & 0 & h(Q-1) & \dots & h(0) & \dots & 0 & 0 \\ 0 & 0 & \dots & 0 & h(Q-1) & \dots & h(0) & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 & h(Q-1) & \dots & h(0) & \dots \\ \dots & 0 & 0 & 0 & \dots & 0 & h(Q-1) & \dots & h(0) \end{pmatrix} \times \begin{pmatrix} X(-N_{CP} - Q + 1) \\ \dots \\ X(-N_{CP}) \\ \dots \\ -1 \\ 0 \\ 1 \\ \dots \\ N \end{pmatrix}$$

if $n < 0$, then $X(n) = X(n + N)$

方法二：由于循环矩阵的特性，在矩阵运算中可以将其省略

$HX =$

$$\begin{pmatrix} h(0) & 0 & \dots & h(Q-1) & \dots & h(1) \\ h(1) & h(0) & 0 & \dots & h(Q-1) & \dots \\ \dots & h(1) & h(0) & 0 & \dots & h(Q-1) \\ h(Q-1) & \dots & h(1) & h(0) & 0 & \dots \\ \dots & h(Q-1) & \dots & h(1) & h(0) & 0 \\ 0 & \dots & h(Q-1) & \dots & h(1) & h(0) \end{pmatrix} \times \begin{pmatrix} X(0) \\ X(1) \\ \dots \\ \dots \\ \dots \\ X(N) \end{pmatrix}$$

$$F = e^{(-j2\pi(k-1)(n-1)/N)}$$

$$F^{-1} = e^{(j2\pi(k-1)(n-1)/N)}$$

$$W = randn() * NoisePower$$

数字实现——主函数

```
1  %%配置参数
2  rng(100)
3  Delays = [0 1 2];MaxDelay = max(Delays);
4  Weights = [3 4 5];
5  NumCarry = 10;
6  NumCP = 2;
7  NumSymble = 4;
8  Mod_level = 4;
9  SNR_db = 10;
10 %%开始运行
11 X_bits = gen_X(NumCarry, NumSymble*2);
12 X_data = ofdm.modulation(X_bits, Mod_level);
13 W = gen_W(NumCarry, NumSymble, SNR_db);
14 H = gen_H(NumCarry, Delays, Weights);
15 H_old = gen_H_old(NumCarry, Delays, Weights, NumCP);
16 Y_data = way_matrix(X_data, H , W);
17 Y_data_old = way_matrix_old(X_data, H_old , W, NumCP, MaxDelay);
18 Y_bits = ofdm.demodulation(Y_data, Mod_level);
19 Y_bits_old = ofdm.demodulation(Y_data_old, Mod_level);
20 diff_Y_Y_old=sum(sum(Y_bits~=Y_bits_old));
21 dif_Y_X=sum(sum(Y_bits~=X_bits));
22 sprintf('两种方法得到的Y_bits差异值为%d', diff_Y_Y_old)
23 sprintf('在SNR_DB=%d的情况下,误比特个数为%d',SNR_db, dif_Y_X)
```


数字实现— 方法二

```
1
2 %% 通过矩阵的方式完成一个frame的ofdm, 输入和输出都是调制信号
3 % input:
4 %   X: 调制后的发送信号, 格式为[preamble, x1, x2...], preamble, xn 都是列向量
5 %   H: 信道矩阵
6 %   W: 噪声矩阵
7 % output:
8 %   Y: 输出序列
9 function Y = way_matrix(X, H, W)
10     [NumCarry, NumSymbble] = size(X);
11     [F, Fi] = gen_F(NumCarry);
12     Y = F*(H*(Fi*(X)))+F*W;
13     Hi = Y(:,1)./X(:,1);
14     Hi = repmat(Hi,[1,NumSymbble]);
15     Y = Y./Hi;
16 end
```

数字实现— 方法一

```
1
2 %% 通过矩阵的方式完成一个frame的ofdm, 输入和输出都是调制信号
3 % input:
4 %   X: 调制后的发送信号, 格式为[preamble, x1, x2...], preamble, xn 都是列向量
5 %   H: 信道矩阵
6 %   W: 噪声矩阵
7 % output:
8 %   Y: 输出序列
9 function Y = way_matrix_old(X, H, W, NumCP, MaxDelay)
10     [F, Fi] = gen_F(size(W,1));
11     X_time = (Fi*(X));
12     X_time_CP_Delay = [X_time(end-(NumCP+MaxDelay-1):end,:);X_time];
13     X_time_CP_H = H*X_time_CP_Delay ;
14     X_time_H_W = X_time_CP_H(1+(NumCP):end,:)+W;
15     Y = F*X_time_H_W;
16     Hi = Y(:,1)./X(:,1);
17     Hi = repmat(Hi,[1,size(X,2)]);
18     Y = Y./Hi;
19 end
```

结果分析

结果分析

ans =

两种方法得到的Y_bits差异值为0

ans =

在SNR_DB=10的情况下，误比特个数为2

1. 两种方法得到的结果一致，符合预期
2. 效果相同，但是方法二明显更加节省资源

最后要说的话—广告时间

感谢老师同学们的耐心帮助

这次的报告托管在<https://github.com/lihao2333/mtheme>

这次的代码托管

在https://github.com/lihao2333/TP_TelSimulation

期待你的 follow 和 star