# Demo: Simulating 802.11-Like MCS Selection

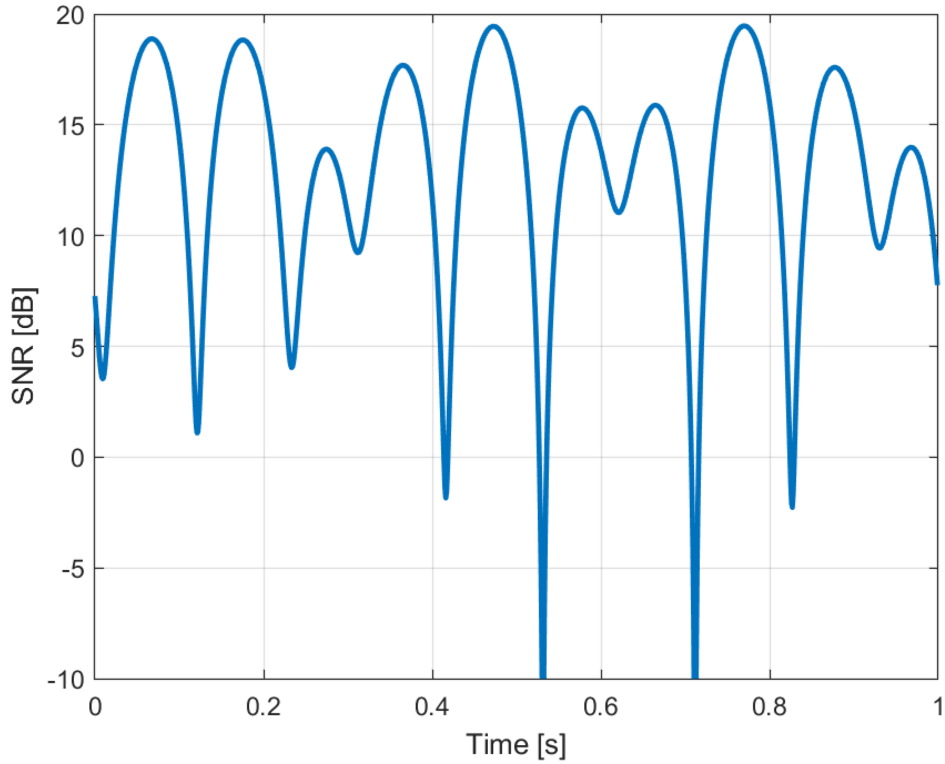
We first look at MCS selection. Suppose that a channel has fading with three paths with the following parameters. For now, we ignore the delays of the paths since we will assume it is not frequency selective.

```
snrAvg = 15;           % Average SNR in dB
gain = [0, -3, -5]';   % Relative path gains in dB
fd = [0, 10, 7]';      % Doppler shift in Hz

% Random initial phases
npaths = length(gain);
phaseInit = 2*pi*rand(npaths,1);
```

Suppose we transmit a packet every period of 1ms (`tstep=1e-3) for nt=1000` steps. We can plot the time evolution of a random realization of the channel as follows.

We can then generate a typical trajectory:

```
% Convert gains to magnitude and normalize to unit average total power
gainLin = db2mag(gain);
gainLin = gainLin / norm(gainLin);

% Compute times
tstep = 1e-3;
nt = 1e3;
t = (0:nt-1)'*tstep;

% Compute the narrowband response
h = gainLin'.*exp(1i*2*pi*t*fd' + 1i*phaseInit');

% Compute the fading gain in dB and add it to the average SNR
fade = 10*log10( abs(sum(h,2)).^2 );
snr = snrAvg + fade;

% Plot the SNR over time
clf;
plot(t,snr, 'LineWidth',2);
xlabel('Time [s]');
ylabel('SNR [dB]');
ylim([-10,20]);
grid on;
```




Now suppose the system supports WiFi-like rates as follows:

| HT MCS | VHT MCS | Modulation | Coding | 20MHz Data Rate 800ns | 20MHz Data Rate 400ns | 20MHz Min. SNR | 20MHz RSSI | 40MHz Data Rate 800ns | 40MHz Data Rate 400ns | 40MHz Min. SNR | 40MHz RSSI |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 Spatial Stream | | | | | | | | | | | |
| 0 | 0 | BPSK | 1/2 | 6.5 | 7.2 | 2 | -82 | 13.5 | 15 | 5 | -79 |
| 1 | 1 | QPSK | 1/2 | 13 | 14.4 | 5 | -79 | 27 | 30 | 8 | -76 |
| 2 | 2 | QPSK | 3/4 | 19.5 | 21.7 | 9 | -77 | 40.5 | 45 | 12 | -74 |
| 3 | 3 | 16-QAM | 1/2 | 26 | 28.9 | 11 | -74 | 54 | 60 | 14 | -71 |
| 4 | 4 | 16-QAM | 3/4 | 39 | 43.3 | 15 | -70 | 81 | 90 | 18 | -67 |
| 5 | 5 | 64-QAM | 2/3 | 52 | 57.8 | 18 | -66 | 108 | 120 | 21 | -63 |
| 6 | 6 | 64-QAM | 3/4 | 58.5 | 65 | 20 | -65 | 121.5 | 135 | 23 | -62 |
| 7 | 7 | 64-QAM | 5/6 | 65 | 72.2 | 25 | -64 | 135 | 150 | 28 | -61 |
| | 8 | 256-QAM | 3/4 | 78 | 86.7 | 29 | -59 | 162 | 180 | 32 | -56 |
| | 9 | 256-QAM | 5/6 | | | 31 | -57 | 180 | 200 | 34 | -54 |
| 2 Spatial Streams | | | | | | | | | | | |

```
nsc = 52;           % number of data subcarriers
tsym = 4e-6;        % OFDM symbol time (using 800ns GI)

% Code rate, modulation rate, minimum SNR and data rate for each of the MCSs
codeRate = [1/2, 1/2, 3/4, 1/2, 3/4, 2/3, 3/4, 5/6]';
modRate = [1,2,2,4,4,6,6,6]';
minSnr = [2,5,9,11,15,18,20,25]';
rate = codeRate.*modRate*nsc/tsym;
```

We can find the optimal MCS for each time k by finding the highest MCS index j such that the SNR at that time is greater than the minimum SNR for that MCS. Mathematically, this is given as follows:

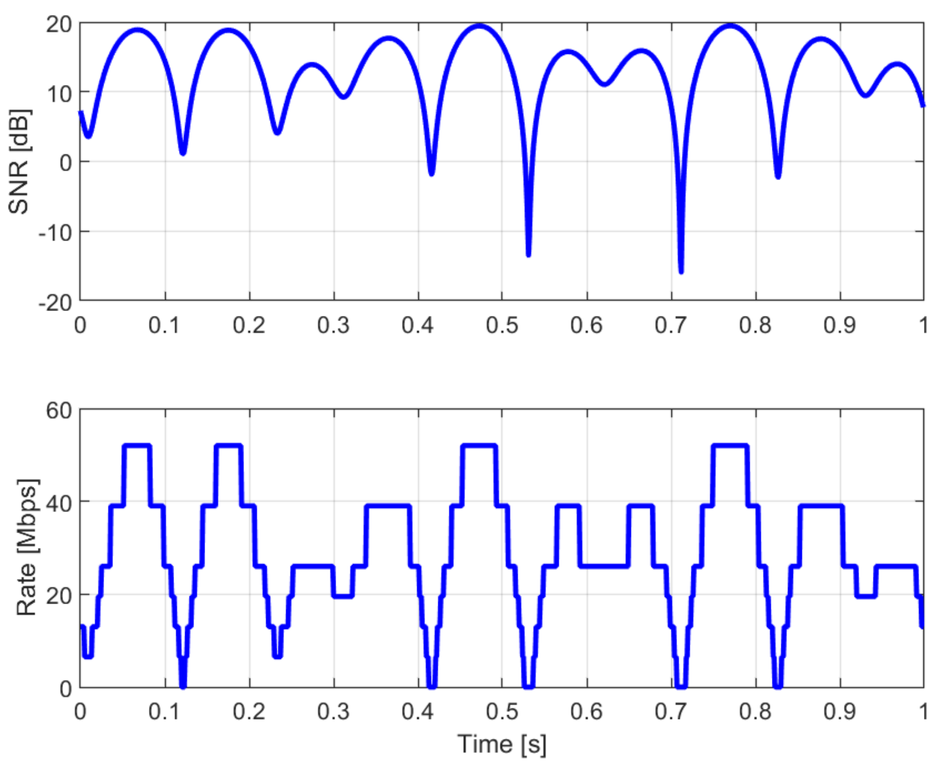
```
mcsOpt(k)  = max { j |   snr(k) >= minSnr(j) }
rateMax(k) = rate( mcsOpt(k) )
```

We can compute this in MATLAB as:

```
[rateMax, mcsOpt] = max((snr > minSnr').*rate', [], 2);
```

We can also plot the rate over time along with the SNR.

```
clf;
subplot(2,1,1);
plot(t,snr,'b-', 'LineWidth',2);
grid on;
ylabel('SNR [dB]');
subplot(2,1,2);
plot(t,rateMax/1e6,'b-','LineWidth', 2);
ylabel('Rate [Mbps]');
xlabel('Time [s]');
grid on;
```




## In-Class Problem

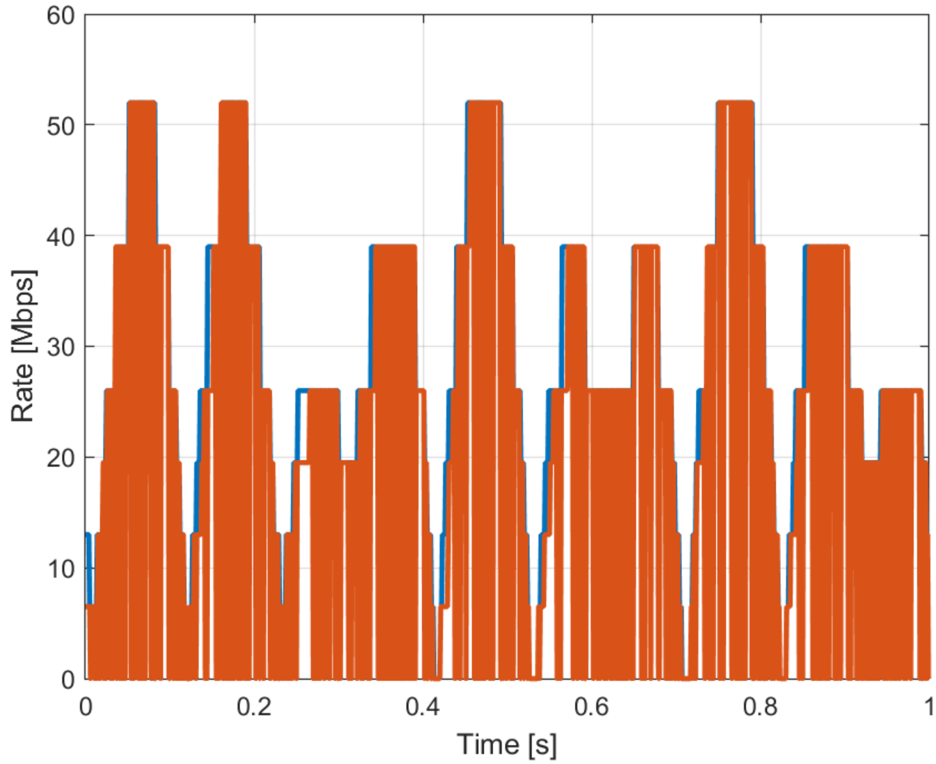
We will now try to implement a trial-and-error: Let `mcsInd(k)` be the MCS index attempted in time slot k and let `rateAdapt(k)` be the goodput in time slot k. We will adapt this MCS index as follows:

- Start with an initial MCS, `mcsInd(1)=1`.
- At each time k, check if `snr(k) >= minSnr(mcsInd(k))`, to see if the packet at time k passes.

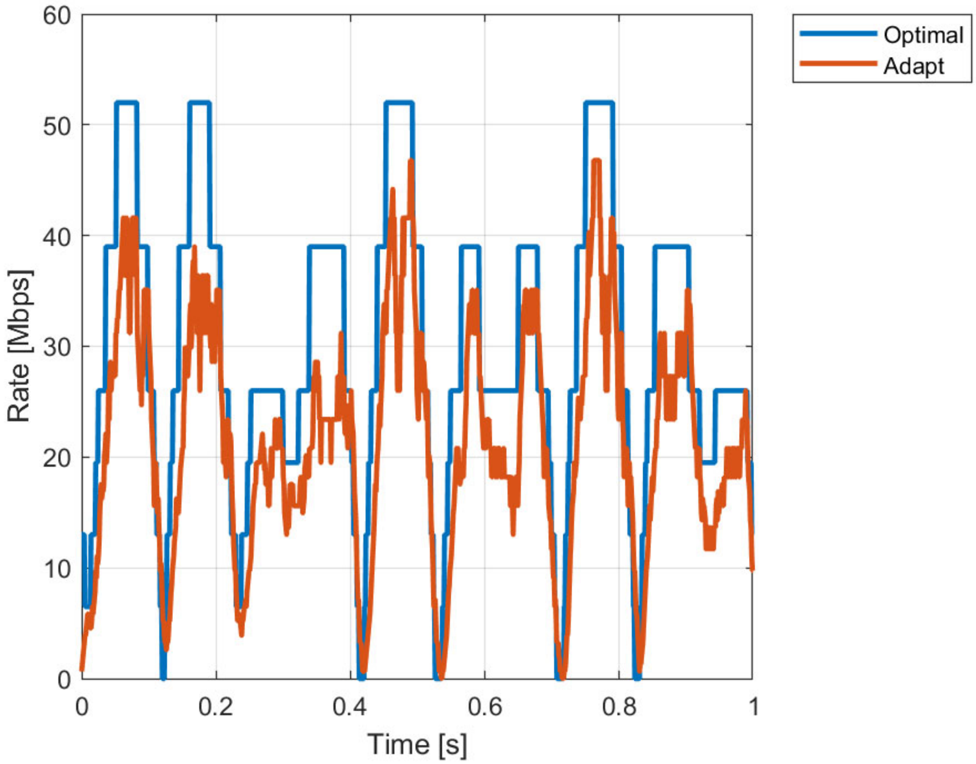
- If the packet fails, set `rateAdapt(k)=0` since no data was transmitted. Also, decrement the MCS index for the next slot.
- If the packet passes, set `rateAdapt(k)` based on the rate achieved in that time slot. Also, with probability pup try increase the MCS index for the next slot.

Plot the `rateAdapt` vs. time.

```
% TODO
mcsInd = zeros(nt,1);
nmcs = length(minSnr);
rateAdapt = zeros(nt,1);
pup = 0.3;
mcsInd(1) = 1;
for k = 1:nt-1
    if snr(k) >= minSnr(mcsInd(k))
        % Packet passes
        rateAdapt(k) = rate(mcsInd(k));
        if rand(1) < pup
            mcsInd(k+1) = min(mcsInd(k) + 1, nmcs);
        else
            mcsInd(k+1) = mcsInd(k);
        end
    else
        % Packet fails
        mcsInd(k+1) = max(mcsInd(k) - 1, 1);
    end
end
time = (0:nt-1)'*tstep;

clf;
filLen = 10;

plot(time, [rateMax rateAdapt]/1e6, 'Linewidth', 2);
ylabel('Rate [Mbps]');
xlabel('Time [s]');
grid on;
```




```
clf;
filLen = 10;
rateSmooth = filter(ones(filLen,1)/filLen,1, rateAdapt);


plot(time, [rateMax rateSmooth]/1e6, 'Linewidth', 2);
ylabel('Rate [Mbps]');
xlabel('Time [s]');
grid on;
legend('Optimal', 'Adapt', 'Location', 'northeastoutside');
```



Optimal
Adapt
Rate [Mbps]
Time [s]
0
10
20
30
40
50
60
0
0.2
0.4
0.6
0.8
1