```
function [dout] = MSKdemod(y, varargin)
%
%args: A, FS, SYMBOL_FREQUENCIES, DATA_RATE MODULATION_T
%

numvarargs = length(varargin);
if numvarargs > 5
    error('too many arguments');
end
optargs = {1, 44100, 882, 800, 882 , 1/882*40 };
optargs(1:numvarargs) = varargin; %set unset args to defaults

[A,FS,DATA_RATE,CENTER_FREQ] = optargs{:};
SAMPLING_T = 1/FS;
MODULATION_T = 1/DATA_RATE;
fc = CENTER_FREQ;
validateattributes(y,{'numeric'},{'row'})
fdev = 1/(4*MODULATION_T)
SYMBOL_FREQUENCIES = [fc-fdev fc+fdev]
startTime = 0; %start=0 seconds
endTime = SAMPLING_T*length(y); %end=1 seconds
Trange = startTime:SAMPLING_T:endTime-SAMPLING_T;
idx = 1;
dout = [];
decision_list = [];

% http://www.electronics.dit.ie/staff/amoloney/lecture-26.pdf

r0_Isum = 0;
r0_Qsum = 0;

r1_Isum = 0;
r1_Qsum = 0;


for curTime=Trange
  % NCO implementation, take powers in sample and increment
  phi_d0_inphase = sqrt(2/
SAMPLING_T)*cos(2*pi*SYMBOL_FREQUENCIES(1)*Trange(idx));
  phi_d0_quad    = sqrt(2/
SAMPLING_T)*sin(2*pi*SYMBOL_FREQUENCIES(1)*Trange(idx));
  phi_d1_inphase = sqrt(2/
SAMPLING_T)*cos(2*pi*SYMBOL_FREQUENCIES(2)*Trange(idx));
  phi_d1_quad    = sqrt(2/
SAMPLING_T)*sin(2*pi*SYMBOL_FREQUENCIES(2)*Trange(idx));

  r0_Isum = r0_Isum + phi_d0_inphase*y(idx);
  r0_Qsum = r0_Qsum + phi_d0_quad*y(idx);

  r1_Isum = r1_Isum + phi_d1_inphase*y(idx);
  r1_Qsum = r1_Qsum + phi_d1_quad*y(idx);
```

```
  %TABULATE POWERS IN FREQS, CHOOSE BIT
  if mod(idx,FS/DATA_RATE) == 0
     r0 = r0_Isum^2 + r0_Qsum^2;
     r1 = r1_Isum^2 + r1_Qsum^2;
     dout = [dout r1>r0];

     r0_Isum = 0;
     r0_Qsum = 0;
     r1_Isum = 0;
     r1_Qsum = 0;
  end
  idx = idx + 1;
end

end
```

Published with MATLAB® R2015b