```
function [x,nit]=SOR_method(x0,A,b,tol,omega)

% Check the inputs
narginchk(3,5);
if nargin<4
    tol=1e-6;
end %if
if nargin<5
    omega=1.1;
end %if


% Setup iterations
maxit=1000;    %max number of iterations
n=size(A,1);  %system size
residual=10*ones(n,1);
difftot=1e3+tol;   %max sure we enter iterations
x=x0;


% Perform iterations
it=1;
while(difftot>tol && it<=maxit)
    difftotprev=difftot;
    resprev=residual;
    xprev=x;
    for i=1:n
        residual(i)=b(i);
        for j=i:n
            residual(i)=residual(i)-A(i,j)*x(j);

        end %for
        for k=1:i-1
            residual(i)=residual(i)-A(i,k)*xprev(k);
        end
    x(i)=xprev(i)+omega*residual(i)/A(i,i);
    end
    difftot=sum(abs(residual-resprev));


    if (difftot>difftotprev & it>2)
        error('Solution appears to be diverging, check diagonal
 dominance...')
    end %if
    it=it+1;
end %while

nit=it-1;
if (nit==maxit)
    warning('Solution may not have converged fully...')
end %if
end %function
```

Published with MATLAB® R2020a