```
function [x,X,k] = JoshJacobi(A,b,x0,TOL,maxI)
% take A matrix
% take b vector
% take x0
arguments
    A{mustBeNumeric,mustBeReal}
    b{mustBeNumeric,mustBeReal,mustBeVector}
    x0{mustBeNumeric,mustBeReal,mustBeVector}
    TOL(1,1) {mustBeNumeric,mustBeReal,mustBePositive} = .01
    maxI (1,1) {mustBeNumeric,mustBeReal,mustBePositive,mustBeInteger} = 100
end
% create defualt maxI
% measure n as dimension
[n,m] = size(A);
% verify they are all same dim and A is square
if n ~= m
    error("A must be square matrix.")
elseif length(b) ~= n
    error("b must be the same size as A.")
elseif length(x0) ~= n
    error("x0 must be the same size as A.")
end

clear m % m and n have been verified to contain the same value
% test if it is SDRD as a test for convergece
myWarnings = "";
for i = 1:n % step through each row
    r = A(i,:); % get row in question
    d = abs(r(i)); % get the magnitude of the diagonal (row i, col i)
    s = sum(abs(r)) - d; % get the sum of the rest of the row
    if d <= s
        myWarnings(1) = "A is not strictly diagonally row dominant, it may not
 converge.";
    end
    if d == 0
        myWarnings(2) = "A has zero(s) in its diagonal,  it may not
 converge.";
    end
end
% send warning if it wont neccisarily converge
for warn = myWarnings
    if warn ~= ""
        warning(warn)
    end
end
clear d s r myWarning warn i % clear vars from SDRD check

% do jacobi iteration, use k as iterator and i , j as indicies
% X will be 2 dimensional with indexs (i,k)
% A will be 2 dimensional with indexs (i,j)
% b will be 1 dimensional with index (i)
% the value of X(i,k+1) is given by
% X(i,k+1) =( 1/A(i,i) ) * ( b(i) - ( (sumOverj ( A(i,j) * X(j,k) )- A(i,i) *
 X(i,k))
k = 1;
X(:,k) = x0;
err = inf;
while (err>TOL) & (k < maxI) % run until TOL or maxI met
    for i = 1:n % for each row
        s = 0;
        for j = 1:n % for each j
             s = s + A(i,j) * X(j,k); % sumOverj ( A(i,j) * X(j,k)
        end
        X(i,k+1) =( 1/A(i,i) ) * ( b(i) - (s - A(i,i) * X(i,k) ) );
    end
    k = k + 1;
    err = norm( (X(:,k)-X(:,k-1)) ,inf);
end
x = X(:,k);

if k >= maxI
    x = [];
    warning("Convergence failed")
end
```

Published with MATLAB® R2022a