```
Input: matrix l, matrix d, matrix u, array b,
        array x0, float tol, int n_max
Output: int iter, array x, float E

begin jacobiMethod
        matrix T = dot_product(inverse_matrix(d),(l + u))
        matrix C = refact_matrix(inverse_matrix(
                time_matrix(inverse_matrix(d),b)),(b.size,1))
        float E <- infinity_value()
        array xant <- xo.transpose()
        int cont <- 0

        array values, array normalized_eigenvectors <- eigen_values(T)
        float spectral_radius <- maximum_value(absolute_value(values))

        if (spectral_radius > 1)
                return ERROR
        end if

        while ((E > tol) and (cont < nmax))
                matrix xact <- dot_product(T, xant) + C
                E <- norm2(xant - xact)
                xant <- xact
                cont <- cont + 1
        end while
        cont, E, xant.transpose()[0]
end
```