

# Solution to a Tri-diagonal Matrix Equation

In general inverting an  $N \times N$  matrix requires  $N^3$  operations, so even with a supercomputer you can't invert a very big matrix  $N < \text{few thousand}$

But consider a system of equations of the form

$$A_j u_{j+1} + B_j u_j + C_j u_{j-1} = D_j \quad (1)$$

Then we can solve this for the vector  $\vec{u}$  with  $\mathcal{O}(N)$  operations as follows:

We seek two quantities  $E_j$  and  $F_j$  such that

$$u_j = E_j u_{j+1} + F_j \quad (2)$$

We assume the boundary conditions require

$$u_0 = 0 \text{ and } u_N = 0$$

which implies that

$$E_0 = F_0 = 0$$

then re-writing equation 2 as

$$u_{j-1} = E_{j-1} u_j + F_{j-1}$$

and plugging into equation 1 we obtain

$$u_j = -\frac{A_j}{B_j + C_j E_{j-1}} u_{j+1} + \frac{D_j - C_j F_{j-1}}{B_j + C_j E_{j-1}}$$

from which we can read off

$$E_j = -\frac{A_j}{B_j + C_j E_{j-1}}$$

and

$$F_j = \frac{D_j - C_j F_{j-1}}{B_j + C_j E_{j-1}}$$

and we sweep through the grid twice, first to get the  $E$  and  $F$  starting at  $j = 1$  and then backwards to get the  $u_j$ , starting with the BC value  $u_N = 0$ .