

C++ code 2.5.4: Constructor of **NodalPotentials**

```
2 NodalPotentials::NodalPotentials(double R, double Rx)
3     // TODO: (2-5.d) initializer list.
4     // START
5     : R_(R),
6       Rx_(Rx)
7 // END
8 {
9     // TODO: (2-5.d) Build the matrix and the r.h.s. vector by either
10    using
11    // the topology struct or by hardcoding the matrix and vector.
12    // START
13    assert(Rx_ != 0. && "Rx should not be equal to zero!");
14    xi_ = R_ / Rx_;
15
16    // Build the topology for a source voltage of 1.
17    Topology top(1.);
18
19    // Automatically build A_Rx filling from topology
20    // A_Rx is the matrix  $A_R$  (i.e. with  $R_x = R$ )
21    Eigen::MatrixXd A_Rx = Eigen::MatrixXd::Zero(15, 15);
22    for (resistor& res : top.T) {
23        // Shift indices down by 1 (indices in Figure start from 1)
24        std::size_t i = res.first - 1;
25        std::size_t j = res.second - 1;
26        // Fill diagonal: each resistance contributes += R
27        // into the diagonal of both nodes
28        A_Rx(i, i) += 1;
29        A_Rx(j, j) += 1;
30        // Fill the off diagonal (negative part in  $\Delta V_{ij}$ ,
31        // the matrix is kept symmetric
32        A_Rx(i, j) -= 1;
33        A_Rx(j, i) -= 1;
34    }
35
36    // Fill in the rest (source and ground), i.e.
37    // components with  $\Delta V_{ij}$  with  $j > 15$ .
38    // Each node  $i$  connected to ground or source contributes
39    // to its own diagonal with R
40    for (voltage& volt : top.S) {
41        // Shift index down by 1 and get voltage in  $V_2$ 
42        int i = std::get<0>(volt) - 1;
43        // Add resistance to matrix diagonal: contribution of source
44        // current
45        // scaled by R.
46        A_Rx(i, i) += 1;
47    }
48
49    // Add resistance between node 14 and 15
50    A_Rx(13, 13) += xi_;
51    A_Rx(13, 14) -= xi_;
52    A_Rx(14, 13) -= xi_;
53    A_Rx(14, 14) += xi_;
54
55    // Set up r.h.s.
56    Eigen::VectorXd b = Eigen::VectorXd::Zero(15);
57    b(5) = 1.;
```