C++ code 3.8.5: Function `solveImpedanceBVP`: Computation of volume contribution to Galerkin matrix ↪ GITHUB

```
Eigen::VectorXd solveImpedanceBVP(
    const std::shared_ptr<lf::uscalfe::FeSpaceLagrangeO1<double>> &fe_space_p,
    Eigen::Vector2d g) {
  // Related implementations:
  // Homework problem ErrorEstimatesForTraces:
  //
     https://gitlab.math.ethz.ch/ralfh/npdecodes/tree/master/homeworks/ErrorEstimatesFor

  // Pointer to current mesh
  std::shared_ptr<const lf::mesh::Mesh> mesh_p = fe_space_p->Mesh();
  // Obtain local->global index mapping for current finite element space
  const lf::assemble::DofHandler &dofh{fe_space_p->LocGlobMap()};
  // Dimension of finite element space
  const lf::uscalfe::size_type N_dofs(dofh.NumDofs());

  Eigen::VectorXd discrete_solution(N_dofs);

  // I : ASSEMBLY
  // Matrix in triplet format holding Galerkin matrix, zero initially.
  lf::assemble::COOMatrix<double> A(N_dofs, N_dofs);
  // Right hand side vector, must be initialized with 0!
  Eigen::Matrix<double, Eigen::Dynamic, 1> phi(N_dofs);
  phi.setZero();

  // I.i : Computing volume matrix for negative Laplace operator
  // Initialize object taking care of local mass (volume) computations.
  lf::uscalfe::LinearFELaplaceElementMatrix elmat_builder{};
  // Invoke assembly on cells (co-dimension = 0 as first argument)
  // Information about the mesh and the local-to-global map is passed through
  // a Dofhandler object, argument 'dofh'. This function call adds triplets to
  // the internal COO-format representation of the sparse matrix A.
  lf::assemble::AssembleMatrixLocally(0, dofh, dofh, elmat_builder, A);
```