C++ code 11.8.26: Sub-problem (11-8.n): function `simulateAdvection()` ⇝ GitLab

```cpp
template <typename FUNCTOR, typename VECTORFIELD>
Eigen::VectorXd simulateAdvection(
    const lf::assemble::DofHandler &dofh, VECTORFIELD &&beta, FUNCTOR &&u0,
    std::shared_ptr<lf::mesh::utils::CodimMeshDataSet<
        std::array<const lf::mesh::Entity *, 4>>>
        adjacentCells,
    std::shared_ptr<lf::mesh::utils::CodimMeshDataSet<
        Eigen::Matrix<double, 2, Eigen::Dynamic>>>
        normal_vectors,
    double T) {
  // Get number of dofs
  int num_dof = dofh.NumDofs();

  // Inititialize a vector for the initial condition
  Eigen::VectorXd u0_h(num_dof);

  // Iterate over all cells
  for (const lf::mesh::Entity *cell : dofh.Mesh()->Entities(0)) {
    const lf::geometry::Geometry *geo_p = cell->Geometry();
    const Eigen::MatrixXd corners = lf::geometry::Corners(*geo_p);

    // Compute barycenter of the cell
    Eigen::Vector2d x = barycenter(corners);

    // Find the index of the DOF for the cell
    int idx = dofh.GlobalDofIndices(*cell)[0];

    u0_h[idx] = u0(x);
  }

  // Set up the number of steps according to the CFL condition
  int M = int((T / computeHmin(dofh.Mesh())) + 2);

  return solveAdvection2D(dofh, beta, u0_h, adjacentCells, normal_vectors, T,
                          M);
}
```