C++ code 5.2.53: Implementation of `brachistochrone()` ↬ GitLab

```cpp
template <typename RECORDER = std::function<
              void(const Eigen::Matrix<double, 2, Eigen::Dynamic> &)>>
Eigen::Matrix<double, 2, Eigen::Dynamic> brachistochrone(
    unsigned int M, Eigen::Vector2d a, Eigen::Vector2d b, double atol,
    double rtol, unsigned int itmax,
    RECORDER &&rec = [](const Eigen::Matrix<double, 2, Eigen::Dynamic> &)
        -> void { return; }) {
  // Initialize knots
  Eigen::Matrix<double, 2, Eigen::Dynamic> knots(2, M + 1);
  // Linear interpolant as intial guess
  for (int i = 0; i <= M; ++i) {
    double s = static_cast<double>(i) / M;
    knots.col(i) = (1.0 - s) * a + s * b;
  }
  Eigen::Matrix<double, 2, Eigen::Dynamic> knots_prev = knots;
  // Record the initial conditions of the knots
  rec(knots);
  // Perform the fixed point-iteration
  unsigned int q = 0;
  do {
    // Store the previous knots data for comparison later
    knots_prev = knots;

    // Compute R-matrix and RHS
    Eigen::SparseMatrix<double> R = matR(knots);
    Eigen::VectorXd rhs = compute_rhs(knots, a, b);
    // Initialize the LU solver
    Eigen::SparseLU<Eigen::SparseMatrix<double>> solver;
    // Do LU decomposition once
    solver.compute(R);
    if (solver.info() != Eigen::Success) {
      std::cerr << "Cannot factorize R!" << std::endl;
      return knots;
    }
    // Solve the first system
    Eigen::VectorXd uh = solver.solve(rhs.segment(0, M - 1));
    for (int i = 0; i < M - 1; ++i) knots(0, i + 1) = uh(i);
    // Solve the second system
    Eigen::VectorXd uh2 = solver.solve(rhs.segment(M - 1, M - 1));
    for (int i = 0; i < M - 1; ++i) knots(1, i + 1) = uh2(i);
    // Update and record
    rec(knots);
    // Abort when max iterations is reached or error is small enough.
  } while ((++q < itmax) &&
           (Brachistochrone::L2norm(knots - knots_prev) >
            std::min<double>(atol, rtol * Brachistochrone::L2norm(knots))));

  // Output a warning if fixed-point iteration was aborted because max
  // iterations was reached.
  if (q == itmax)
    std::cout << "M = " << M
              << " : Max iterations reached. Truncated with L2 error: "
              << L2norm(knots - knots_prev) << "\n";
  return knots;
};
```