C++ code 7.8.5: Sub-problem (7-8.e): Auxiliary function `solveGenStageEquation()`
→ GITHUB

```cpp
// Use Newton method to approximate a stage.
template <typename Functor, typename Jacobian>
Eigen::VectorXd SolveGenStageEquation(Functor &&f, Jacobian &&df,
                                      const Eigen::VectorXd &y,
                                      const Eigen::VectorXd &b, double h,
                                      double rtol = 1E-6, double atol = 1E-8) {
  // Need to solve the equation lhs(g) = g - h*f(y+g)/4 - b = 0.
  // lhs and its Jacobian Jlhs
  auto lhs = [f, y, b, h](const Eigen::VectorXd &g) {
    Eigen::VectorXd val = g - 0.25 * h * f(y + g) - b;
    return val;
  };
  auto Jlhs = [df, y, h](const Eigen::VectorXd &g) {
    // Jlhs(g) = Id - h*df(y+g)/4
    int dim = y.size();
    Eigen::MatrixXd Jval =
        Eigen::MatrixXd::Identity(dim, dim) - 0.25 * h * df(y + g);
    return Jval;
  };

  // Perform Newton iterations:
  Eigen::VectorXd g = Eigen::VectorXd::Zero(y.size());  // initial guess g=0.
  Eigen::VectorXd delta =
      -Jlhs(g).lu().solve(lhs(g));  // Newton correction term.
  int iter = 0;
  int maxiter = 100;  // If correction based termination does not work.
  while (delta.norm() > atol && delta.norm() > rtol * g.norm() &&
         iter < maxiter) {
    g = g + delta;
    delta = -Jlhs(g).lu().solve(lhs(g));
    iter++;
  }
  return g + delta;  // Perform the final step
}
```