C++ code 11.5.13: Sub-problem (11-5.i): function `dgcl()` ↪ GitLab

```
template <typename FUNCTOR, typename NUMFLUX>
Eigen::VectorXd dgcl(Eigen::VectorXd mu0, FUNCTOR &&f, NUMFLUX &&F, double T,
                     int Ml, int Mr, double h, unsigned int m) {
  Eigen::SparseMatrix<double> B = compBmat(Ml, Mr, h);
  Eigen::SparseMatrix<double> Binv = B.cwiseInverse();

  auto G_bound = [&f, &F, T, Ml, Mr, h](const Eigen::VectorXd &mu) {
    return G(mu, std::forward<FUNCTOR>(f), std::forward<NUMFLUX>(F), Ml, Mr, h);
  };
  // Timestepping based on explicit midpoint rule, a 2-stage explicit
     Rune-Kutta
  // single-step method
  double tau = T / m;
  for (int i = 0; i < m; ++i) {
    // First compute the increment and then update the state vector, see
    // [Lecture → Eq. (11.4.1.12)]
    Eigen::VectorXd k = -Binv * G_bound(mu0);
    mu0 = mu0 - tau * Binv * G_bound(mu0 + 0.5 * tau * k);
  }
  return mu0;
}
```