C++ code 9.2.18: Convergence test for SDIRK-2 in the scalar case ↬ GitLab

```
  double diff;  // temporary variable used to compute error at various nodes
  double max_norm_errors[nIter];  // errors vector for all approx. sols
  std::vector<double> approx_sol_vec;
  for (int k = 0; k < nIter; k++) {
    m = 10 * std::pow(2, k);

    // Creating exact solution vector. This vector is created by evaluating the
    // exact solution using the analytic formula y(t) = exp(-t) at the
    // equidistant nodes of the time steps.
    double step_size = 2.0 / m;  // "tau" in this example
    std::vector<double> exact_sol_vec;
    for (int i = 0; i < m + 1; i++) {
      exact_sol_vec.push_back(std::exp(-i * step_size));
    }
    // Computing approximate solution
    approx_sol_vec = sdirk2SteppingLinScalODE(m);
    // Computing the error in the maximum norm
    for (int i = 0; i < m + 1; i++) {
      max_norm_errors[k] = 0;
      diff = std::abs(approx_sol_vec.at(i) - exact_sol_vec.at(i));
      max_norm_errors[k] =
          (diff > max_norm_errors[k]) ? diff : max_norm_errors[k];
    }
  }
  // Computing rates of convergence
  double rates[nIter - 1];
  double avg_rate = 0.0;
  for (int k = 0; k < nIter - 1; k++) {
    rates[k] = log2(max_norm_errors[k] / max_norm_errors[k + 1]);
    avg_rate += rates[k];
  }
  avg_rate = avg_rate / (nIter - 1);
```