1. Construct adjoint algorithm clean up
   1. Extended algorithm to complex numbers
      1. All matrices are forced to be complex matrices
         1. Julia type issues, e.g., cannot concat a Float64 Array with a Complex{Float64} array, InexactError() when calculating rank and inverse of a Float64 or Number matrix with complex entries (no error if matrix with complex entries are exactly of type Complex{Float64})
      2. Removed tests that check for zeros of p\_k
         1. Can’t find roots of p\_k on [a,b] when p\_k involves complex numbers, e.g., 2x-3i
   2. Wrote function to construct L from symL
      1. With systematic unit tests
2. Implement transform pairs
   1. Some questions in the code
   2. Naming conventions
   3. How to test the implementation?
      1. F -> f and f -> F
   4. Complex integral
      1. (2.16a, b): Assume f(x) is a (finite sum of) Chebyshev polynomial (very good convergence, much better than Taylor), implement algorithm by hand (approxFun package, which allows integrating expansions in Chebyshev polynomial; also allows approximating a function using Chebyshev polynomial)
      2. Contour:
         1. Approximate the Gamma contours using straight lines, e.g., list of ordered pairs representing points in the complex plane. Then compute the integral using line integral packages.
         2. The Gamma contours need to avoid zeroes on the exterior. For now, construct the contours assuming user has input a list of zeroes. Later, we can either
            1. Build tools to find approximately where the zeroes are, or
            2. Build algorithm to find where exactly the zeroes are in certain cases
3. Capstone presentation
   1. Focus on big picture