Computer Code Documentation

To facilitate solution of the examples, some computer source code, dynamic link libraries and executables are provided. These codes have been run under Windows with Lahey/Fujitsu Fortran 95 5.60, Excel 2003, and Visual C++ 6.0. Executables and the dynamic link libraries are created from the Fortran code using the make file EXmake.mak. The C++ executables are created using the various Visual C++ project files. If you would like to contribute code to this project, e.g. for other operating systems or Matlab, I would be happy to consider it.

The Fortran code uses many features that are specific to Fortran 90/95 such as modules, data protection, optional arguments, array syntax, and array valued functions. The primary code for creating the coefficients is in OCC.f90. It is implemented as a Fortran 90 module with private data and array valued functions which return the requested coefficients. LUSolve.f90 contains a number of routines for solving linear algebraic equations. These are implemented as a module with private data and only selected public interfaces. Internally some of the routines come from old FORTRAN 77 Linpack. There is a lot of useful code like this on the internet. Why reinvent the wheel? I just put a wrapper around it to create a modern interface. I have written some of the other code with extensive use of array syntax. I believe most of this code is a good example of how Fortran 90 should be written.

The modern Fortran 90 interface creates a problem for mixed language programming. OCCdll.f90 interfaces to the modules with a plain interface which is easily called from Excel, C++ or other languages. Excel spreadsheets also require some visual basic code and must be able to find OCCdll.dll in order to execute properly. The C++ code requires OCCdllvc.lib to link and OCCdllvc.dll to execute.

OCC.f90 - Orthogonal Collocation Coefficient Module

CollocSet - set symmetry, type & geometry
CollocCoef - call following 5 functions to get all the arrays
Xpoints - returns quadrature points
Weights - returns quadrature weights
Acoeff - returns first derivative matrix
Bcoeff - returns second derivative matrix
Ccoeff - returns At*W*A for Lobatto -wB for Gauss
MassMat - Galerkin or Moments mass matrix
AnonSym - matrix for first derivative of odd function
Interp – returns interpolates values
InterpCoef - returns interpolating polynomials
CollocSym - returns the current setting for the symmetry

LUSolve.f90 - Linear Algebra Module

LUSolve - solves Ax = b after factoring A (Linpack)
LUFactr - factors A (Linpack)
LUSubst - solves Ax = b given factored A (Linpack)
LUInvert - calculates inverse of A (Linpack)

LUSolveSym - solves Ax = b when A is symmetric, positive definite
LUFactrS - symmetric factorization
LUSubstS - symmetric solve given factored A

Dynamic Link Library Routines (OCCdll.f90 interface routines to create OCCdll.dll for Excel and OCCdllvc.dll and OCCdllvc.lib for Visual C++)

OCC_Set – calls CollocSet to set parameters
OCC_X – calls Xpoints to get points
OCC_W – calls Weights to get quadrature weights
OCC_A – calls Acoeff
OCC_B – calls Bcoeff
OCC_C – calls Ccoeff
OCC_D – calls MassMat
OCC_L – calls InterpCoef
OCC_Anon – calls AnonSym
LUSolve – calls LUSolve (above) to solve a general matrix problem
LUSolveS – calls LUSolveSym to solve a symmetric matrix problem

OCCdll.bas, LUFactor.bas – visual basic code to interface with OCCdll.dll

This Visual basic code is needed to interface with the dynamic link library. This code is embedded in the Excel spreadsheets.

Test.f90, Test.exe – coefficients in text file

This is a simple console application to illustrate use of the coefficient module. It is created using the make file EXmake.mak or by compiling and linking it with OCC.f90. A copy of the executable, Test.exe, is provided. The coefficients are output as tab delimited text so they are easily imported by Excel.

Test.xls – create coefficients using OCCdll.dll

This is an Excel spreadsheet which shows how to create the coefficients from within a spreadsheet by using the supplied dynamic link library. Just replicate the steps outlined in the spreadsheet. To create the dll use the make file EXmake.mak.

EX01.f90 – asymmetric reaction-diffusion

Fortran 90 code which solves the asymmetric reaction-diffusion problem.

EX01.xls – asymmetric reaction-diffusion

Excel spreadsheet which solves the asymmetric reaction-diffusion problem.

EX01.cpp – asymmetric reaction-diffusion

The TestCpp directory contains a Visual C++ console application to solve the asymmetric reaction-diffusion problem. The directory includes the project, workspace and other ancillary files for use with the Visual C++ system.