MPI and PETSc

A brief tutorial

Praveen Chandrashekar
https://cpraveen.github.io

MPI: Hello, World!

- https://github.com/cpraveen/parallel/tree/master/mpi
- Compile the code

```
mpif77 -o hello hello.f
mpif90 -o hello hello.f90
mpicc  -o hello hello.c
```

- If you have a 4-core cpu, run like this

```
mpirun -np 4 hello
```

- `mpicc`, etc. are just wrappers; see the actual command (this works with openmpi only)

```
mpicc --showme
```

integrate1: blocking send/recv

- Compute using R ranks; partition the domain

  $$[a, b] = \bigcup_{n=1}^{R} [a_n, b_n]$$

  and compute as

  $$I = \int_a^b \cos(x)dx = \sum_{n=1}^{R} \int_{a_n}^{b_n} \cos(x)dx = \sum_{n=1}^{R} [\sin(b_n) - \sin(a_n)]$$

- Compile

  ```
  mpif90 -o integrate1 integrate1.f90
  mpicc  -o integrate1 integrate1.c
  ```

- Run

  ```
  mpirun -np 4 integrate1
  ```

PETSc resources

- http://www.petsc.org
- Manual pages
- Notes by Matt Knepley
- Book by Ed Bueler: PETSc for Partial Differential Equations: Numerical Solutions in C and Python
  Codes from the book: https://github.com/bueler/p4pdes
- PETSc has many examples: see
  http://cpraveen.github.io/teaching/petsc.html
  and
  https://petsc.org/release/tutorials

parallel/petsc/hello.c

```
make hello
./hello -help intro
./hello -help
./hello
mpirun -n 4 ./hello
mpiexec -n 4 ./hello
```

Variable types

- PetscInt: usually 32-bit integer, can be configured for 64-bit integer while compiling Petsc, which is needed for large meshes
- PetscErrorCode: integer return type from Petsc functions
- PetscMPIInt: use this to pass to MPI functions, e.g., to get rank, size, etc.
- PetscReal: usually real double
- PetscScalar: usually real double, can be complex double also. It is used for the scalar field in a vector space and for entries of vectors and matrices.

Printing

- PetscPrintf: similar to printf in C, not collective
- `PetscPrintf(PETSC_COMM_WORLD,format,variables)`
  prints only on rank=0 process
- `PetscPrintf(PETSC_COMM_SELF,format,variables)`
  prints on every rank
- By default
  ```
  PETSC_COMM_WORLD = MPI_COMM_WORLD
  PETSC_COMM_SELF = MPI_COMM_SELF (always)
  ```
- PetscFPrintf: Prints to a file

p4pdes/c/ch1/e.c

- Compute the value of e in parallel

  $$e = \sum_{n=0}^{\infty} \frac{1}{n!}$$

- Each rank computes one term in the sum
- If total ranks = r, we compute

  $$e \approx \sum_{n=0}^{r-1} \frac{1}{n!} = 1 + 1 + \frac{1}{2!} + \ldots + \frac{1}{(r-1)!}$$

- rank = n computes $\dfrac{1}{n!}$
- Accuracy increases as number of ranks increase

Creating DMDA

- DMDACreate1D
- DMDACreate2D
- DMDACreate3D
- DMBoundaryType
- DMDAStencilType

DMDA info

- DMDAGetInfo: get full mesh sizes
- DMDAGetCorners: get range of locally owned grid
- DMDAGetGhostCorners: get range of local grid including ghost points
- All of above info can be obtained together using: DMDALocalInfo and DMDAGetLocalInfo

Vectors

- Global vector: distributed vector without ghost values
- Local vector: has ghost values
- DMCreateGlobalVector and DMGetGlobalVector
- DMCreateLocalVector and DMGetLocalVector
- VecSetValues
- VecAssemblyBegin and VecAssemblyEnd

Arrays from Vectors

- Useful for working with Cartesian/structured grids
- VecGetArray and VecRestoreArray
- VecGetArrayRead and VecRestoreArrayRead
- VecPlaceArray
- DMDAVecGetArrayDOF and DMDAVecRestoreArrayDOF
- DMDAVecGetArrayDOFRead and DMDAVecRestoreArrayDOFRead

cfdlab/petsc

- convect1d
- burger1d
- convect2d
- euler2d/ssprk.c
- euler2d/ts.c
- euler2d/fdweno.c

BVP: Tutorial examples

- ex2: 2-D Poisson equation
  https://petsc.org/release/src/ksp/ksp/tutorials/ex2.c.html
- ex46: 2-D Poisson equation, using DM
  https://petsc.org/release/src/ksp/ksp/tutorials/ex46.c.html