Introduction to PETSc

Data management

Loïc Gouarin

Laboratoire de Mathématiques d'Orsay

May 13-15, 2013

Data management

PETSc offers two types of data management

- DMDA: data management for structured mesh
- DMPlex (*or DMMesh*): data management for unstructured mesh

These structures define for each process

- local portion of the mesh,
- ghost points,
- communications with the neighbourhood to update ghost points,
- global and local mapping,
- ...

DMDA creation

```
int DMDACreate2d(MPI_Comm comm,
                 DMDABoundaryType xperiod,
                 DMDABoundaryType yperiod,
                 DMDAStencilType st, int M,
                 int N,int m,int n,int dof,int s,
                 int *lx,int *ly,DM *da)
```

- `xperiod` and `yperiod`: type of ghost nodes.

  DMDA_BOUNDARY_NONE, DMDA_BOUNDARY_GHOSTED, DMDA_BOUNDARY_PERIODIC
- `st`: stencil type.

  DMDA_STENCIL_BOX or DMDA_STENCIL_STAR
- `M` and `N`: global dimension in each direction.
- `m` and `n`: number of processors in each direction.
- `dof`: number of degrees of freedom per node.
- `s`: stencil width.

Local and global vectors

Creation

```
int DMCreateGlobalVector(DM da,Vec *g)
int DMCreateLocalVector(DM da,Vec *l)
```

Scatter a global vector into its local parts including the ghost points

```
DMGlobalToLocalBegin(DM da,Vec g,
                     InsertMode iora,Vec l);
DMGlobalToLocalEnd(DM da,Vec g,
                   InsertMode iora,Vec l);
```

Scatter a local vector into the global vector

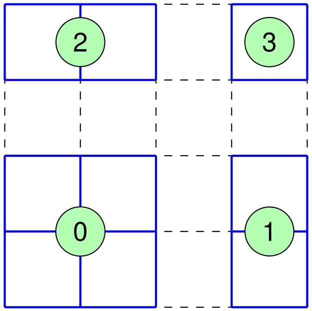
```
DMDALocalToGlobalBegin(DM da,Vec l,
                       InsertMode iora,Vec g);
DMDALocalToGlobalEnd(DM da,Vec l,
                     InsertMode iora,Vec g);
```

`InsertMode` can be either `INSERT_VALUES` or `ADD_VALUES`.

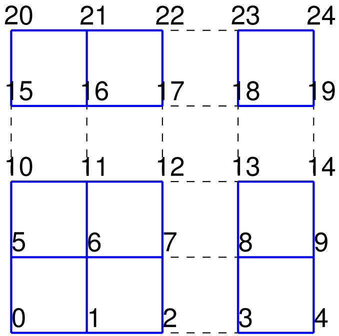
First example

```c
#include "petsc.h"

int main(int argc, char **argv){
  int nx=5, ny=5;
  DM dm;
  Vec g;

  PetscInitialize(&argc, &argv, NULL, NULL);

  DMDACreate2d(PETSC_COMM_WORLD,
               DMDA_BOUNDARY_NONE, DMDA_BOUNDARY_NONE,
               DMDA_STENCIL_STAR,
               nx, ny, PETSC_DECIDE, PETSC_DECIDE,
               1, 1, PETSC_NULL, PETSC_NULL, &dm);
  DMCreateGlobalVector(dm, &g);
  ...
  VecDestroy(&g);
  PetscFinalize();
  return 0;
}
```

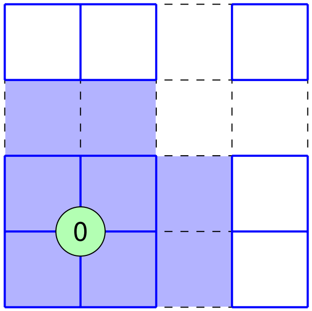
Local portion on each process



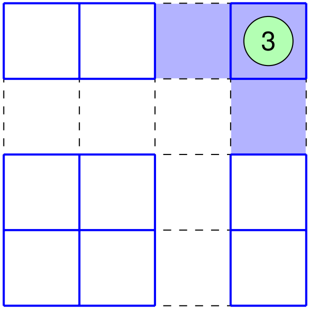
Local portion on each process



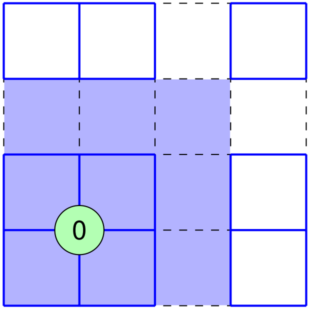
Ghost points: DMDA_STENCIL_STAR



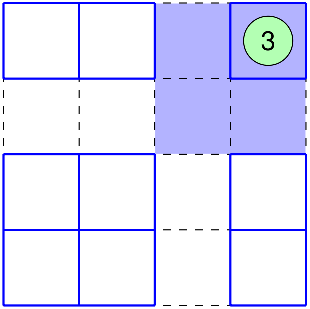
Ghost points: DMDA_STENCIL_STAR



Ghost points: DMDA_STENCIL_BOX

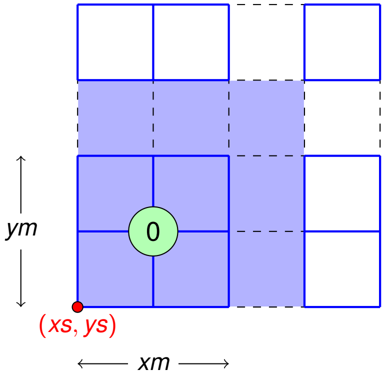


Ghost points: DMDA_STENCIL_BOX



How to get grid information?

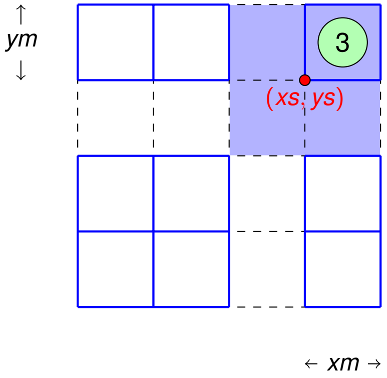
```
DMDAGetCorners(DM da,
               int *xs,int *ys,int *zs,
               int *xm,int *ym,int *zm);
```



Use PETSC_NULL if you want to omit a parameter.

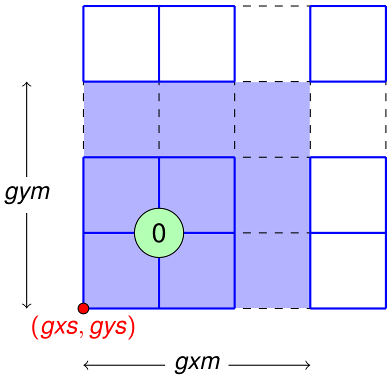
How to get grid information?

```
DMDAGetCorners(DM da,
               int *xs,int *ys,int *zs,
               int *xm,int *ym,int *zm);
```



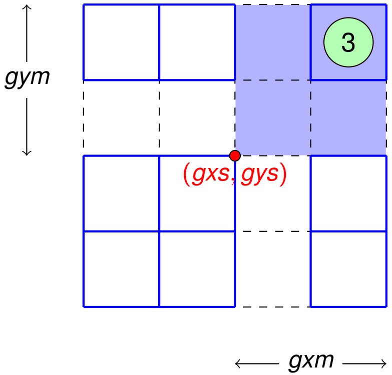
How to get grid information?

```
DMDAGetGhostCorners(DM da,
                    int *gxs,int *gys,int *gzs,
                    int *gxm,int *gym,int *gzm);
```



How to get grid information?

```
DMDAGetGhostCorners(DM da,
                    int *gxs,int *gys,int *gzs,
                    int *gxm,int *gym,int *gzm);
```



How to get grid information?

```
DMDAGetLocalInfo(DM da,DMDALocalInfo *info)
```

```
typedef struct {
  PetscInt dim,dof,sw;
/* global number of grid points in each direction */
  PetscInt mx,my,mz;
/* starting point of this processor, excluding ghosts */
  PetscInt xs,ys,zs;
/* number of grid points on this processor, excluding ghosts *
  PetscInt xm,ym,zm;
/* starting point of this processor including ghosts */
  PetscInt gxs,gys,gzs;
/* number of grid points on this processor including ghosts *
  PetscInt gxm,gym,gzm;
/* type of ghost nodes at boundary */
  DMDABoundaryType bx,by,bz;
  DMDAStencilType  st;
  DM               da;
} DMDALocalInfo;
```

DMDA offers functions for vector manipulation

Local (ghosted) work vectors

```
DMGetLocalVector(DM da,Vec *l);
.... use the local vector l
DMRestoreLocalVector(DM da,Vec *l);
```

Accessing the vector entries for DMDA vectors

```
PetscScalar **f,**u;
...
DMDAVecGetArray(DM da,Vec local,&u);
DMDAVecGetArray(DM da,Vec global,&f);
...
f[i][j] = u[i][j] - ...
...
DMDAVecRestoreArray(DM da,Vec local,&u);
DMDAVecRestoreArray(DM da,Vec global,&f);
```

References

1. PETSc documentation
   http://www.mcs.anl.gov/petsc/documentation/index.html
2. PETSc tutorial
   http://www.mcs.anl.gov/petsc/documentation/tutorials/index.html