PISCS Documentation

Revised December 16, 2020

Contents

List of Figures

List of Data Files

Chapter 1

Getting Started

1.1 Introduction

The Poisson Integral Solver with Curved Surfaces (PISCS) is a package written in COSYScript for MSU COSY Infinity v9.2 and higher. PISCS is a 3-D Poisson boundary value problem solver accelerated by the fast multipole method (FMM). In this case, the Poisson BVP represents the electrostatic interactions within a charged particle distribution as a supplement to beam physics computations.

1.2 Setup

You must install MSU COSY Infinity to use PISCS. Contact the beam physics group at the COSY website (http://bt.pa.msu.edu/index_cosy.htm) for permission and instructions. PISCS does not use the COSY beam physics package (cosy.fox) to run.

After COSY is installed, PISCS will need some things to run.

1. piscs.fox
2. fmmcpp.zip
3. A parameter file
4. A particle distribution file
5. A structure file
6. (Optional) A boundary condition file

It is recommended to store these files in a single folder. To set up PISCS:

1. Extract fmmcpp.zip.
2. Compile the fmmcpp executable using at least a C++11 compiler. A Makefile is provided for the GNU or Intel compiler on Linux.
3. (Recommended) Create a folder for temporary files.

This concludes the main setup.

1.3 Running PISCS

COSY 9.2 and higher allows running PISCS using command-line arguments. Assuming the COSY executable is 'cosy', the basic command is

```
cosy piscs.fox <parameter_file (optional)> <diagnostic flag (optional)>
        <Rotated M2L flag (optional)> <evaluation flag (optional)>
        <evaluation file (optional)>
```

The parameter file contains all the filenames and input parameters required by PISCS. If no file is specified, then the default is 'parameters.dat' (1.3.1) in the same directory as PISCS. The diagnostic flag is optional. The default value is '0', which does nothing. Any nonzero value will print timing information for the major PISCS procedures (if only these times are desired, then set value to '-1'). A value of '1' will print timing information for the major FMM procedures. A value of '2' or '3' will print GMRES diagnostic information. The rotated M2L operator has been shown to be more efficient for larger problems (FMM Publication) and thus may reduce the computational time. The evaluation flag will evaluate the electric potentials and fields at locations specified in the evaluation file if the flag is set to '1'. This useful when the user wants to sample the potentials due to a particle distribution at locations different from the particle locations.

Note, PISCS can optionally be run as an executable by the program Run Cosy (downloadable package from MSU). All filenames must include relative paths. The input lines are (sample taken form the parameters.dat file included in the piscs_example folder) shown in 1.3.1.

The temporary folder must be created beforehand. The particle file contains the space or comma separated particle coordinates X,Y,Z, with one particle per line. The structure file is explained in the next sections (1.4, 1.5). The element order describes the order of the polynomial used to interpolate a curved surface element. 1$^{\text{st}}$ order corresponds flat elements, 2$^{\text{nd}}$ order elements are interpolated using a quadratic polynomial, etc. The boundary conditions can be described in two ways. The first (as shown in 1.3.1) is to include a single value that specifies the potential over the entire surface. If a varying boundary potential

tmp/	Temporary folder for FMM data files (*OutputPath*)
N=1000.dat	Particle file (*SourceFile*)
protons	Particle species (*ChargeFile*)
sphere_order=1.struct	Structure file (*TargetFile*)
0	Minimum clustering parameter (*MinQ*)
./fmmcpp.intel	Relative path to FMMCPP executable (*fmmcpp_program*)
6	FMM multipole order (*P_Order*)
1	Element order (*El_Order*)
Dirichlet	Boundary condition type (*BCFlag*)
0	QBX flag (*QBXflag*)
10	Boundary conditions (*BoundF*)
0	Preconditioning flag (*PFlag*)
0	Select Unit scaling (*FType*)
0	Memory Parameter (*Mempar*)

Data File 1.3.1: parameters.dat

is desired, then this inputs links to a file with the boundary condition (numerical value) is listed for each point in the structure file (the order of boundary values should correspond to the point order in the structure file). The QBX flag ('0' or '1') determines whether the QBX near-boundary refinement is applied (2.2). The preconditioning is generally unnecessary for PISCS and can be left inactive. The unit scaling is included to improve numerical stability of the calculations. Use '0' for SI units, or '1' for natural units. Specifying a memory parameter specifically for FMM evaluations can improve the computational efficiency for certain applications. The default (if '0' is specified) is 50000, where 1 signifies the memory required to store a single double precision number.

1.4 Structure File

The structure file uses a particular format to describe the bounding surface. It is assumed the origin is always inside the surface.

1. First line is the total number of elements.
2. Subsequent lines are element blocks.
3. Each element block consists of the number of nodes corresponding to the input element order.
4. One node per line. Each node is described by (space- or comma-separated) coordinates and unit normals, x, y, z, n_x , n_y , n_z.

Example 1.

$$\left.\left.\begin{array}{cccccc} x & y & z & n_x & n_y & n_z \\ & & & & & \\ & & & & & \end{array}\right] \#\text{ points}\right] \#\text{ elements}$$

$$o = \left\lfloor \frac{1}{10}\left(-15+\sqrt{5}\sqrt{29+36p+12p^2}\right)\right\rfloor$$
$$\#\text{ points} = \frac{(o+1)(o+2)}{2}$$

1.5 Generating the structure file

To generate the variable element order structure mesh, we used gmsh (http://gmsh.info/). Gmsh has many features and can be read in detail on their website. Gmsh can generate a high order mesh based on many common CAD formats. The mesh may require some refinement in gmsh to minimize distortions, however, we have found that the Gmsh built-in refinement process leads to a degradation of final results. We provide a COSYScript program, 'gmshparser.fox', for converting from gmsh's (*.msh) 2D triangular element mesh to ours. The command is as shown:

```
cosy gmshparser.fox <*.msh filename> <input mesh order> <output order>
        <output filename (optional)> <scaling factor (optional)>
        <normal direction (optional)>
```

The default output file is 'struct.dat'. For the most accurate results, set input mesh order ¿ output order. The scaling factor is a positive number which scales the structure up (if the # > 1) or down (if the # < 1), and the normal direction can flip the orientation of the structure normals (-1). The convention is for normals to point out from the surface; normals pointed the wrong way will lead to highly inaccurate results for the fields calculate in near the surface.

Running:

```
cosy gmshparser.fox h
```

will give a description of the required parameters.

Chapter 2

The PISCS Code

2.1 Introduction

A detailed description of the theoretical frameworks that underly PISCS, along with the relevant references, can be found in this dissertation.

2.2 Near-boundary Instabilities

The foundation of PISCS involves computing integrals of the Green's function (for the Poisson equation),

$$G(\mathbf{x}, \mathbf{y}) = \frac{1}{4\pi\|\mathbf{x} - \mathbf{y}\|} \tag{1}$$

or its derivative over the surface. When the surface is discretized this integral becomes a sum, with each point on the surface weighted depending on the quadrature method chosen. When the evaluation point ($\mathbf{x}$) lies close to the surface, there are points ($\mathbf{y}_i$) from the discretization for which the Green's function (Eq. 1) is nearly singular. This is not an issue when the full integral is computed. However, the error from computing a discrete density for the quadrature is bounded by the derivative of G, thus the results will be unstable when evaluating near the boundary. This can be seen in the results for evaluating the potential and fields in a conducting sphere (Fig. 3.1). Near the boundary (radius of 10m), the instability is clear.

One approach to resolve this instability is to form an expansion centered away from the surface (in the stable region) and evaluate that expansion for points near the boundary, rather than the direct quadrature sum. This approach has been called Quadrature by Expansion (QBX), and has been shown to be quite effective for two dimensional cases. We have found that it can also be quite effective in the three dimensional case.The distance of the expansion center (and the evaluation radius) from the surface will be a function of a characteristic element size (typically an area or function of side lengths), element order, and order of the expansion polynomial. The reason the expansion center and evaluation radius are not the same is due to the competing nature of computational errors. The near-boundary instability has already been discussed, however, error will also be introduced by the truncation of the expansion polynomial, and will increase as a function of the distance from the expansion center. Particularly when the discretization uses a few/low order elements, it is necessary for the expansion center to be far from the surface, otherwise, the expansion doesn't have the desired accuracy. This means that as you move from the expansion center to the surface, the truncation error initially grows before the near-boundary instability becomes significant. Thus it is desirous to only evaluate the expansion for points close enough to the surface that the boundary instability induced error (which grows without bound) is greater than the truncation error.

Current implementation utilizes an adaptive QBX process, which calculates the expansion center and evaluation radius based on the nature of the nearest surface element (size and order) for each evaluation point ($\mathbf{x}$). Current work is in process to further reduce the near boundary instabillities.

2.2.1 Adaptive QBX

The expansion center (distance from the surface) c is selected as a function of element order N and the characterlistic length h of the nearest element. The characteristic length is given by

$$h = \sqrt{2.3A}, \tag{2}$$

where A is the element area (calculated from the Jacobian of the interpolation matrix). Eq. 2 assumes that the triangular elements are approximately equilateral (a reasonable assumption if the mesh has been generated correctly). The expansion center radius is then given by

$$c = 2^{\frac{1}{N}} h. \tag{3}$$

Here N represents the and the norm order (number of nodes on an edge) for the NVF interpolation. The pre-factor (2 for linear surface elements) is chosen from studies with the flat parametrization triangle, which found that the integral kernel is slowly varying over the flat panel in 3D if the nearest distance to the target point is greater than the edge length of the element.

The radius of evaluation of the expansion (r) is set to $r = 0.75c$.

Chapter 3

PISCS Example

3.1 Introduction

A simple example is provided to serve as an introduction for PISCS. It sets up a perfect conducting sphere (radius = 10m) with 1000 protons randomly arrayed inside, and with a boundary potential of 10V under Dirichlet boundary conditions. The sphere is described by 80 constant (ie. flat, order = 1) elements. See Data File 1.3.1 for an example of how these values are specified. The potentials and fields at each of the protons are specified in the output files *potentials* and *fields* (located in the same directory as PISCS).

3.2 Contents

All of the files required to run this example are contained in the piscs_example.zip archive. The contents are:

1. N=100.dat
2. N=1000.dat
3. parameters.dat
4. sphere_order=1.struct
5. sphere_order=2.struct
6. sphere_order=3.struct
7. tmp/

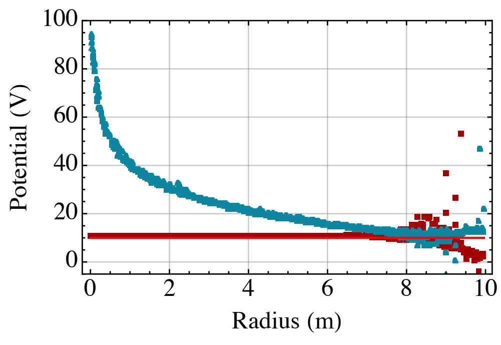
The first two files are particle files whose name specifies the number of particles described. Note, all the particles in file 1 are clustered around the center of the sphere, while those in file 2 fill the entire sphere and are sampled from a uniform distribution in radius. File 3 is an example of the parameter file (see 1.3.1). Note that the relative file paths will need to be changed if the files are moved to a different directory. Also, it may be necessary to update the FMMCPP executable depending on the compiler used. The next three files are structure files for a 10m radius sphere discretized into 80 triangular elements at the element order specified in the file name. The last file is an empty folder which will be used to store the temporary PISCS files (mostly for the FMM).

3.3 Running the Example

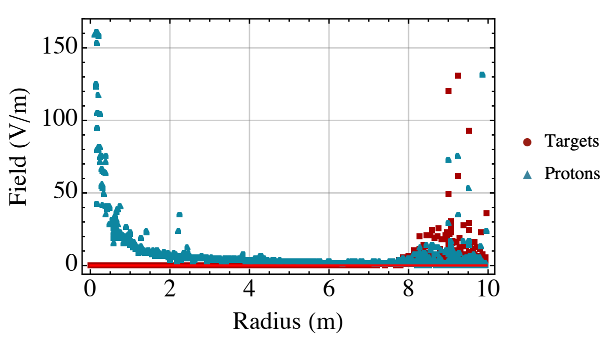
Extract the archive containing the example files. Set up PISCS and the FMMCPP executable as described in section 1.2.

3.3.1 Basic Evaluation

Initially, no boundary refinement in included. By changing the particle species to 'targets' in the parameters.dat (1.3.1) we can compute the potentials and fields due simply to the boundary, which provides a useful comparison to analytic results. By Gauss' Law, since there are no sources within the conducting sphere, the electric field should be uniformly zero. Thus the electric potential must be constant and the same as the conditions on the boundary (due to continuity).



(a)



(b)

Figure 3.1: Plots of the output potentials (a) and fields (b) when the example is run for the 1^{st} order sphere using 1000 particles as targets and as macro-protons. In both plots, a red line indicates the theoretical values when no charges are present.

Plotted in figure 3.1 are the resulting potentials (left) and electric field magnitudes (right) for both protons and targets. Since 1000 protons would not have a noticeable effect on the results in this scale, they are treated as macro-particles each with a charge of 1×10^8 e. Also plotted is a line (dark blue) specifying the theoretical values for targets (ie. 10V for the potentials and 0V/m for the fields). For the targets (red), the results are close to theoretical values away from the surface. The protons (light blue) show large values near the center of the sphere (as expected given the small distances between particles) and decreases toward the boundary values as the radius increases (fulfilling the condition of continuity of the potentials and fields for all $r \leq 10$m). The near-boundary instabilities are more noticeable when targets are evaluated, but can also be seen for protons.

3.3.2 QBX Refinement

Now we consider including near-boundary refinement via QBX. In this case, the adaptive QBX parameters are $c \approx 5.4$ and $r \approx 4$. Considering the plots in figure 3.1, this results in the center falling away from the boundary instability, and the expansion evaluation radius marking the approximate start of the instability.

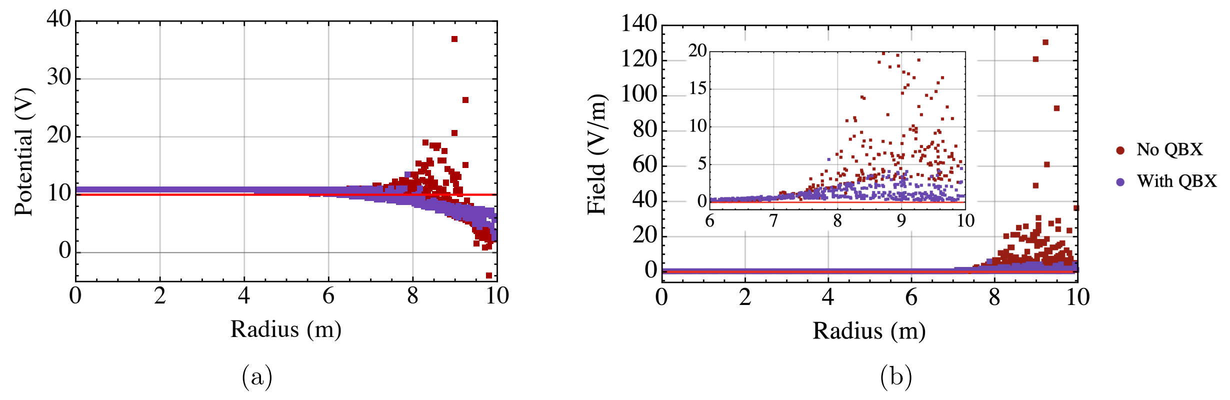
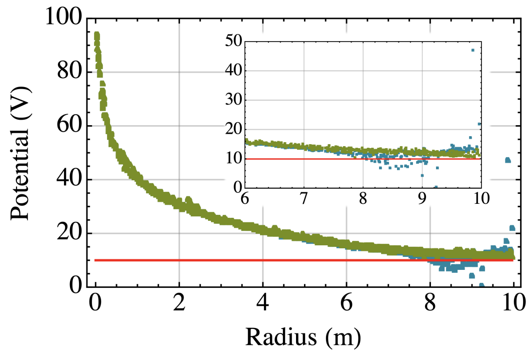


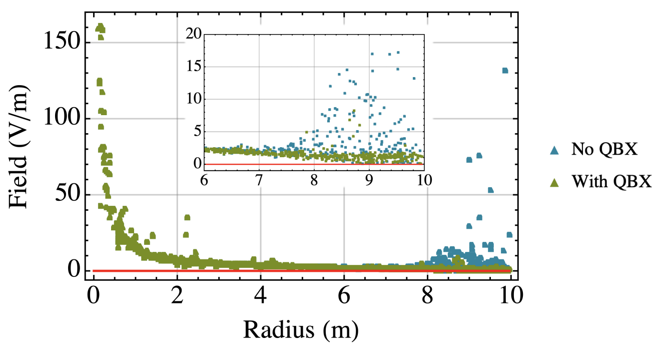
Figure 3.2: Plots of the output potentials (a) and fields (b) when the example is run for the 1st order sphere using 1000 particles as targets considering the effects of QBX. In both plots, a red line indicates the theoretical values when no charges are present.

Figure 3.2 shows the resulting potentials (left) and fields (right) for target evaluation with and without QBX refinement. Improvement (with reference to the analytic values) in the fields in particular, and also the potentials to a lesser degree is apparent. The same can also be said for the case of proton evaluation (shown in figure 3.3), both for the fields, and the protons (which can be seen to better converge on the surface potential in the insert of figure 3.3(a)).

The boundary instability is clearly not completely removed when considering the target potentials (figure 3.2(a)). This is due primarily to the simplicity of the the boundary discretization. For cases with larger numbers of elements ($\mathcal{O}(1000)$ and higher element orders



(a)



(b)

Figure 3.3: Plots of the output potentials (a) and fields (b) when the example is run for the 1st order sphere using 1000 particles as macro-protons considering the effects of QBX. In both plots, a red line indicates the theoretical values when no charges are present.

(order > 1), this QBX implementation has been shown to reduce the boundary instability errors to near the order of the numerical errors present in the bulk evaluation (ie. quadrature error).