MPSpack tutorial

Alex Barnett[*] and Timo Betcke[†]

July 15, 2009

Abstract

This is a short tutorial showing how boundary-value problems may be numerically solved simply and accurately with the MPSpack toolbox for MATLAB. We assume basic familiarity with MATLAB and with partial differential equations.

1 About this tutorial

This tutorial is designed for 'bottom-up' learning of the features of MPSpack, i.e. by progressing through simple examples. In that sense it complements the user manual which describes the theoretical framework in broad strokes and therefore could be considered 'top-down'. We will skip the mathematics behind the solution techniques, focusing on computing and plotting useful PDE solutions.

Throughout we will identify the plane $\mathbb{R}^2$ with the complex plane $\mathbb{C}$, by the usual map $z = x + iy$. In other words $(2, 3)$ and $2+3i$ represent the same point. We use `teletype` font to designate commands that may be typed at the MATLAB prompt. All the code examples in this document, and code to generate the figures, is found in `tutorial.m` in the `examples/` directory.

2 Solving Laplace's equation in a disc

We start by setting up a domain in $\mathbb{R}^2$. Domains are built from segments which define their boundary. To make the unit disc domain, we first need a circle segment with center 0, radius 1, and angle range $[0, 2\pi)$, as follows,

```
s = segment([], [0 1 0 2*pi])
```

[*]Department of Mathematics, Dartmouth College, Hanover, NH, 03755, USA
[†]Department of Mathematics, University of Reading, Berkshire, RG6 6AX, UK

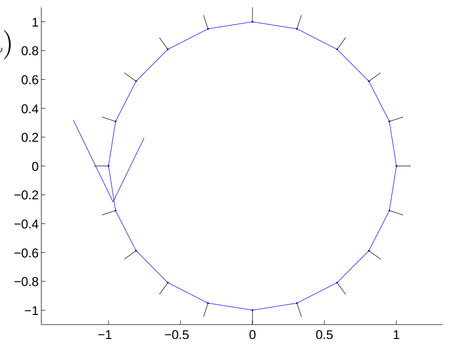
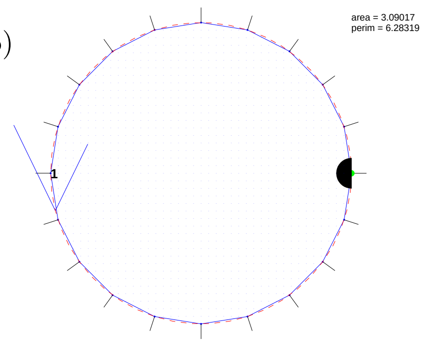
a)  b) 

Figure 1: a) circular closed segment, b) unit disc domain. Both have a periodic trapezoidal quadrature rule with $M = 20$ quadrature points

The object `s` is indeed a circular segment, as we may check by typing `s.plot`, producing Fig. 1a. All segments have a *sense*, i.e. direction of travel: for this segment it is counter-clockwise, as shown by the downwards-pointing arrow symbol overlayed onto the segment at about 9 o'clock.[1] Notice also normal vectors (short 'hairs') pointing outwards at each boundary point; our definition is that normals on a segment always point to the *right* when traversing the sense of the segment.

We create the domain interior to this segment with

```
d = domain(s, +1)
```

where the second argument (here $+1$, the only other option being -1) specifies that the domain is to the 'standard' side of the segment, which we take to be such that the normals point *away from* the domain. That is, with $+1$ the domain lies to the *left* of the segment when traversed in its correct sense (with -1 the domain would lie to the right of the segment.) Typing `d.plot` produces[2] Fig. 1b. Note that perimeter and area are automatically labelled (these are only rough approximations intended for sanity checks).

Laplace's equation $\Delta u = 0$ is Helmholtz's equation with wavenumber zero, which we set for this domain with,

```
d.k = 0;
```

Our philosophy is to approximate the solution in the domain by a linear combination of *basis functions*, each defined over the whole domain. We choose

[1] In fact, segments are parametrized internally as function $z(t)$ of a real variable $t \in [0, 1]$, and the sense is the direction of increasing t. Segment `s` stores this function as `s.Z`.

[2] There are extra plotting options and features that are described in documentation such as `help domain.plot`. E.g. in this figure a grid of points interior to the domain has been included, achieved with `opts.gridinside=0.05; d.plot(opts);`

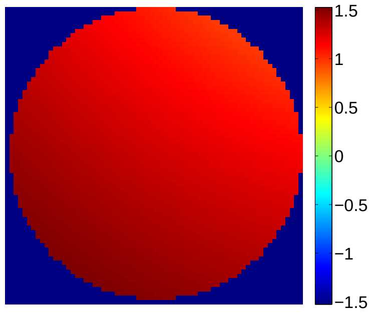
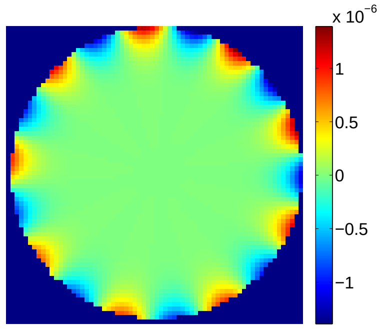
a)  b) 

Figure 2: a) Numerical solution field u, b) pointwise error $u - f$, for Laplace's equation in the unit disc with $M = 20$ quadrature points and 8th-order harmonic polynomials.

8th-order harmonic polynomials $u(z) = \sum_{n=0}^{8} c_n \operatorname{Re} z^n + \sum_{n=1}^{8} c_{-n} \operatorname{Im} z^n$, where $\boldsymbol{c} := \{c_n\}_{n=-8}^{8} \in \mathbb{R}^{17}$ is a coefficient vector, based at the origin 0, using the command

```
d.addregfbbasis(0, 8);
```

Let's specify Dirichlet boundary data $f(z) = \ln |z - 2 - 3i|$ for z on the segment[3] by representing this as an anonymous function `f` and associating it with one side of the segment,

```
f = @(z) log(abs(z-2-3i));
s.setbc(-1, 'd', [], @(t) f(s.Z(t)));
```

Note that we needed to pass in a function not of location z, but of the segment parameter t; this was achieved by wrapping `f` around the parametrization function `s.Z`. The first argument -1 expresses that the boundary condition is to be understood in the limit approaching from the side *opposite* the segment's normal direction, which is where the domain is located. The second argument `'d'` specifies that the data is Dirichlet.

Finally we use the domain to make a boundary-value problem object `p`,

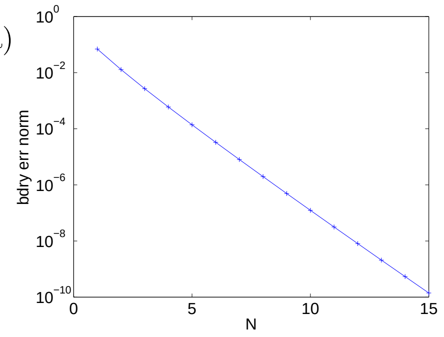
```
p = bvp(d);
```

and may then solve (in the least-squares sense) a linear system for the coefficients

```
p.solvecoeffs;
```

If it is needed, `p.co` now contains the coefficients vector $\boldsymbol{c}$. To evaluate and plot the solution we simply use,

[3]In other words, $f(x, y) = \ln \sqrt{(x-2)^2 + (y-3)^2}$ for points (x, y) on the boundary.

a)


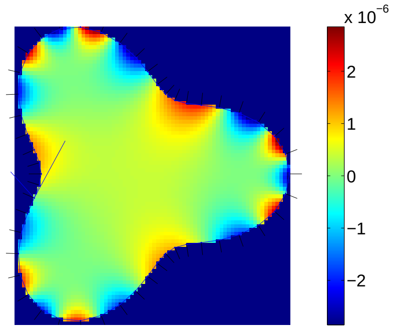
b)


Figure 3: a) Convergence of boundary error L^2 norm for harmonic polynomials for Laplace equation in the unit disc, b) solution error for same boundary data f in a smooth star-shaped domain (normals also shown).

```
    p.showsolution;
```

The software chose an appropriate grid covering the domain (points outside the domain are made transparent), giving Fig. 2a.

3 Accuracy, convergence, and smooth domains

How accurate was our numerical solution u? One measure is the L^2 error on the boundary, and is estimated by

```
    p.bcresidualnorm
```

which returns 2.09×10^{-6}. However, since the function $f(z)$ is already harmonic in the domain, it is in fact the unique solution, and we may plot the pointwise error in u by passing in the analytic solution as an option,

```
    opts.comparefunc = f; p.showsolution(opts);
```

giving Fig. 2b. Note that the color scale is 10^{-8}.

In the above, boundary integrals were approximated using the default of $M = 20$ quadrature points, barely adequate given the oscillatory error function in Fig. 2b. M may be easily changed either by specifying a non-empty first argument in the `segment` constructor above, or for an existing segment as follows,

```
    s.requadrature(50); p.solvecoeffs; p.bcresidualnorm
```

which now gives 1.98×10^{-6}, not much different than before. Notice that we did not have to redefine the domain `d` nor the BVP object `p`.

Exploring the convergence of the boundary error norm with the basis set order needs a simple loop and figure,

```
for N=1:15
  d.bas{1}.N = N; p.solvecoeffs; r(N) = p.bcresidualnorm;
end
figure; semilogy(r, '+-'); xlabel('N'); ylabel('bdry err norm');
```

As Fig. 3a shows, the convergence is exponential.[4]

Say we want to change the shape of segment s, to a smooth star-shaped domain expressed as by radius $R(\theta) = 1 + 0.3\cos 3\theta$ as a function of angle $0 \le \theta < 2\pi$. This is achieved by passing a 1-by-2 cell array containing the function R and its derivative $R' = dR/d\theta$ to a variant of the segment constructor,

```
s = segment.radialfunc(50, {@(q) 1 + 0.3*cos(3*q), @(q) -3*0.3*sin(3*q)});
```

We again chose $M = 50$. The analytic formula for R' is needed to compute normal derivatives to high accuracy.

One might ask: has this change to s *propagated* to the existing domain object d and BVP object p, which both refer to it? In contrast to the case of quadrature point number M above, the answer is no: s is overwritten by a newly-constructed object, while d and p still contain handles pointing to the *old* s. Furthermore, the fact that the segment had domain d attached to its 'minus' or back side has been forgotten, as have the boundary conditions. (These segment properties are described in the MPSpack user manual.) We must therefore rerun the code from Sec. 2 to construct d and p afresh, before solving.[5] The result, plotting the pointwise error as before, is shown by Fig. 3b for $N = 8$ and $M = 50$.

The radialfunc constructor above is limited to radial functions with quadrature equidistant in angle. Instead you may create a segment from arbitrary smooth parametrizations $z(t)$ for $t \in [0, 1]$, as long as $z'(t)$ is also given. For instance, a closed crescent-shaped analytic segment is produced by

```
a = 0.2; b = 0.8; w = @(t) exp(2i*pi*t);
s = segment(100, {@(t) w(t)-a./(w(t)+b), ...
                  @(t) 2i*pi*w(t).*(1 + a./(w(t) + b).^2)}, 'p');
```

[4]Asymptotically, error $\sim e^{-\alpha N}$. In fact the rate is $\alpha = \ln\sqrt{13}$, related to the conformal distance to the nearest singularity [1], which here is at $2 + 3i$.

[5]Note that in theory it would be possible to change one by one each of the segment properties, t, w, speed, etc, to define the new segment without changing its identity, but this is cumbersome. Similarly, searching and changing all references to a segment in the properties of d and p is cumbersome. Neither has been implemented since problem setup time is very rapid.

Note the nested anonymous functions for mathematical clarity. Note also the new final argument `'p'` which enforces periodic quadrature (the constructor doesn't try to guess your preferred rule). In order to get high-order (or spectral) convergence, it is recommended that you choose only smooth (or analytic) z. If periodic quadrature is used, this also applies to the 1-periodic extension of z to the real line. If $z(1) \neq z(0)$, the ends of the segment will not connect up, and the domain constructor above will report an error.

4 Helmholtz equation, exterior and non-simply connected domains

Changing from the Laplace to Helmholtz equation is as simple as setting `d.k` to a different value. We start a fresh example: a exterior Helmholtz BVP with Neumann boundary data, and the Sommerfeld radiation condition [2]. This has a unique solution.

5 Polygons and domains with corners

line segments, quadrature rules on $[0, 1]$.

corners in domains.

The small black semicircle at angle $\theta = 0$ illustrates that `MPSpack` has connected the start and end of the segment to form a 'corner'. Although we did not discuss it, this also happens when a segment is connected to itself as in Sec. 2.

6 Scattering and transmission problems

7 Corner basis sets

8 Layer potentials

References

[1] T. Betcke, *Computations of Eigenfunctions of Planar Regions*, PhD thesis, Oxford University, UK, 2005.

[2] D. Colton and R. Kress, *Inverse acoustic and electromagnetic scattering theory*, vol. 93 of Applied Mathematical Sciences, Springer-Verlag, Berlin, second ed., 1998.