MPSpack tutorial

Alex Barnett[*] and Timo Betcke[†]

July 14, 2009

Abstract

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

Contents

1 About this tutorial **1**

2 Solving Laplace's equation in a smooth domain **2**

2.1 Convergence 4

3 More interesting domains, and corners **4**

4 Exterior domains **5**

5 Scattering problems **5**

6 Corner basis sets **5**

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

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

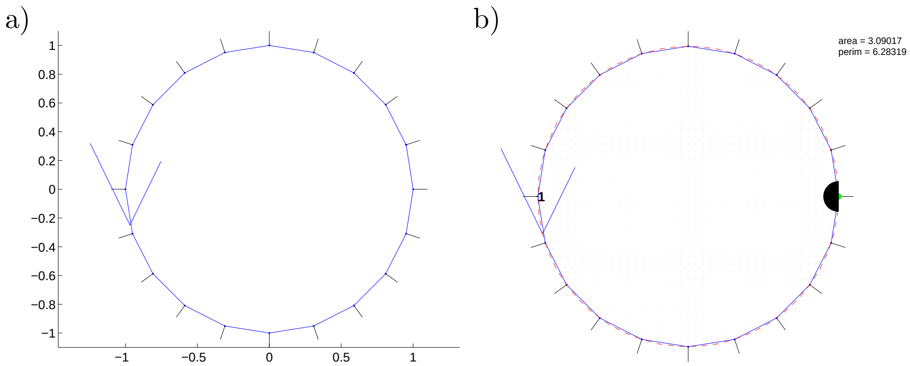

Figure 1: a) circular closed segment, b) unit disc domain. Both have a periodic trapezoidal quadrature rule with $M = 20$ quadrature points

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 smooth domain

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])
```

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)
```

[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`.

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

```
d.addregfbbasis([], 8);
```

Let's specify Dirichlet boundary data $f(x,y) = \ln \sqrt{(x-2)^2 + (y-3)^2} = \ln |z - 2 - 3i|$, for z on the segment, by representing this as an anonymous function `f` and passing it to the segment,

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

Note that in fact we needed to pass in a function of the segment parameter t, which 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. Finally we set up a BVP by passing domains to a problem object, and then may solve (in the least-squares sense) for the coefficients

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

Now `p.co` contains the coefficients vector $\boldsymbol{c}$. To evaluate and the solution we simply use,

```
p.showsolution;
```

This command evaluated selected an appropriate grid covering the domain (points outside the domain are made transparent), giving Fig. 2a.

How accurate was our numerical solution u? One measure is the L^2 error

[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)

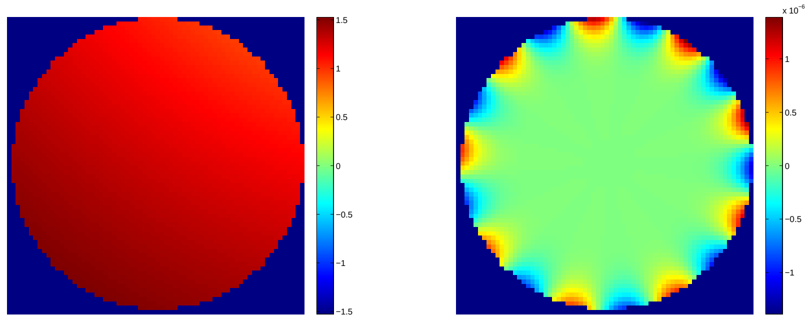

Figure 2: a) numerical solution field u, b) pointwise error $u - f$, for Laplace's equation in the unit disc with $M = 20$ quadrature points and harmonic polynomials up to 8th order

on the boundary, and is estimated by

```
p.bcresidualnorm
```

However, since the function $f(z)$ is already harmonic in $\mathbb{R}^2$, it is the unique analytic solution, and we may compare against this over the domain by passing in an option,

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

giving Fig. 2b.

2.1 Convergence

This is easy. Change M, watch

```
pr.bcresidualnorm
```

Convergence loop with N, easy.

3 More interesting domains, and corners

We create a closed segment with polar function $r(\theta) = 1 + \cos 3\theta$ for $0 \leq \theta < 2\pi$ using

```
s = segment.radialfunc([], @(th) 1 + cos(3*th), @(th)
-3*sin(3*th));
```

Notice that we pass in a cell array of two function handles: $r(\theta)$ and $r'(\theta)$. The latter is needed to compute normal directions accurately.

4 Exterior domains

5 Scattering problems

6 Corner basis sets