`MPSpack` user manual

Alex Barnett[*]and Timo Betcke[†]

code revision 36: January 21, 2009

Abstract

`MPSpack` is a fully object-oriented MATLAB toolbox that implements the Method of Particular Solutions and related methods (*e.g.* fundamental solutions) for efficient solution of Laplace eigenvalue, interior/exterior Helmholtz problems (*e.g.* wave scattering), and related problems, on piecewise-constant 2D domains. It is designed to be simple to use, and to enable highly-accurate solutions. This is the user manual.

1 Overview

`MPSpack` is a MATLAB toolbox to solve Helmholtz boundary-value problems (BVP) and eigenvalue problems (EVP) with particular and fundamental solution methods.

Give example commands, eg eigenmodes of disc.

Define BVP, EVP

Fig. 1 shows how a geometry is built up from objects, and the relationships between them.

2 Installation

Requirements:

[*]Department of Mathematics, Dartmouth College, Hanover, NH, 03755, USA
[†]School of Mathematics, The University of Manchester, Manchester, M13 9PL, UK

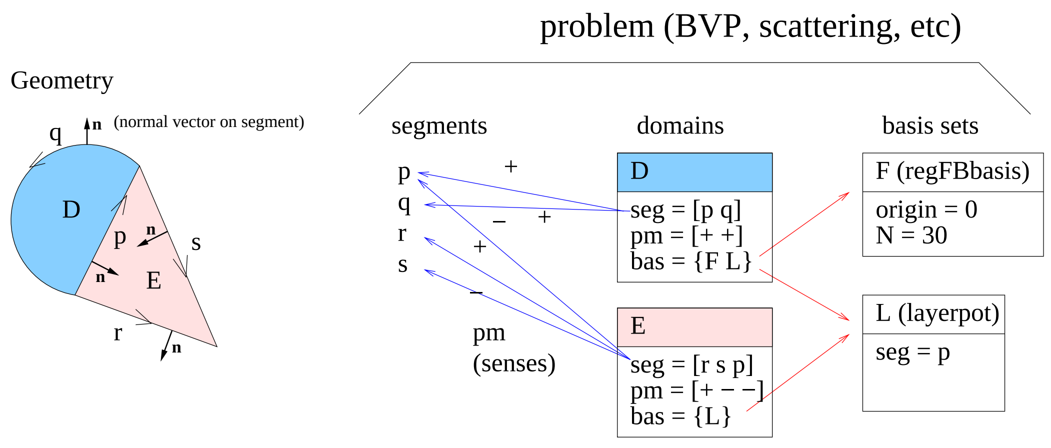

Figure 1: Relationship of `MPSpack` objects used to solve a typical BVP or EVP. The physical geometry is shown on the left, and the code structures on the right. Each segment has a normal direction. Each domain is built from segments with a list of signs (+ or −) which define the sense in which the segments are used to form the boundary. Basis sets affect the function values inside each domain in which they are referenced. A basis set may influence more than one domain.

- MATLAB version 7.6 (2008a) or newer.

We make use of the object-oriented programming ability of MATLAB which unfortunately means this toolbox cannot work with versions earlier than 7.6.

The project is hosted at `http://code.google.com/p/mpspack`

There are two alternative methods to download and unpack:

1. Get a gzip-compressed tar archive from

   `http://code.google.com/p/mpspack/downloads/list`.

   In a UNIX environment you may now unpack this with

   `tar zxvf mpspack-r*.tar.gz`

   This creates the directory tree `mpspack-r*`, where `*` represents the revision (version) number.

2. First install the `subversion` (SVN) version control software from

   `http://subversion.tigris.org`

Anonymous check out of MPSpack is then via the subversion command:

```
svn checkout http://mpspack.googlecode.com/svn/trunk/ mpspack-read-only
```

This creates a directory mpspack-read-only in which you will find the mpspack directory tree.

You might prefer a more user-friendly graphical subversion client such as those listed at

http://subversion.tigris.org/links.html#clients

To compile the faster basis evaluation routines (Fortran and MEX files) in a UNIX environment, from the mpspack subdirectory, type make

You should now add the mpspack subdirectory to your MATLAB path, for instance by adding the line

```
addpath 'path/to/mpspack-read-only/mpspack';
```

to your MATLAB startup.m file

You are now ready to use MPSpack !

2.1 Boundary-value problems

Let $\Omega \subset \mathbb{R}^2$

$$\begin{aligned} (\Delta + k^2)u &= 0 \qquad \text{in } \Omega & (1)\\ au + bu_n &= f \qquad \text{on } \partial\Omega & (2) \end{aligned}$$

In our implementation we restrict to functions a, b that are constant on segments used to define the boundary $\partial\Omega$.

If Ω is an exterior domain, we may wish to impose additional boundary conditions at infinity, such as Sommerfeld's radiation condition. This may be achieved by choosing basis sets satisfying this condition.

See [1] for an example of using the Method of Fundamental Solutions to solve an interior Helmholtz problem.

To solve a BVP the usual flow is as follows:

1. define piecewise-analytic segments forming all boundaries
2. define domains using various of these segments as their boundaries
3. choose MPS basis set(s) within each domain

4. set up inhomogeneous boundary or matching conditions on each segment
5. solve a linear system to get the basis coefficient vector
6. evaluate solution on desired points
7. plot solution or compute error estimates

...

2.2 Eigenvalue problems

2.3 Geometry

...

When domains are built from segments, each segment stores information about which domain, if any, it is connected to on each side. If segments are to be reused to construct new domains, while erasing any old domains, the current way to do this is via the `s.disconnect` command, where `s` is a segment handle or array of such. This clears these segments of any connections with domains.

The segment object field responsible for this is `s.dom`, which is a 1-by-2 cell array of handles to the domain on the segment's natural normal side and back side respectively. If there is no domain on a particular side, the cell element is empty.

2.4 Boundary conditions

A segment Γ may contain a boundary condition on only one of its sides, of the form

$$au(s) + bu_n(s) = f(s) \qquad s \in \Gamma, \quad a, b \in \mathbb{C}$$

or, alternatively matching conditions of the form

$$\begin{aligned} a^+u^+(s) + a^-u^-(s) &= f(s) \qquad (3) \\ b^+u_n^+(s) + b^-u_n^-(s) &= g(s) \qquad (4) \end{aligned}$$

where $a^+, a^-, b^+, b^- \in \mathbb{C}$, and where u^+ is the limiting value approaching the segment from its positive normal side. In the above f and g are functions on the boundary. In our implementation the user may supply these as handles to functions of segment parameter $t \in [0, 1]$, or as a data column vector on the segment quadrature points.

The segment fields `a` and `b` store a and b for a boundary condition, or $[a^+, a^-]$ and $[b^+, b^-]$ respectively for a matching condition.

2.5 Basis sets

Basis sets exist independently of domains or segments [1]. However, for a basis set to influence a domain `d`, its handle must appear in the cell array field `d.bas` This is set up automatically when you perform `d.addregfbbasis()` or any of the other

There will be more elaborate basis set types which naturally affect two domains, such as dielectric corner and transmission layer potential objects; their handles will appear in the `bas` cell array of more than one domain.

3 In-depth examples

4 Test routines

The code `testbvp.m` shows the main steps for solution of a BVP on a variety of domains.

5 Limitations

- Checking of whether inside a domain is approximate, based on approximating polygons.

6 List of improvements to make

List to be prioritized:

[1]apart from layer potentials which require a segment on which their source density is placed

- make `segment.bdryfunc` which creates u, u_n data given a field function and its gradient field function, useful for choosing BC data corresponding to exact fields
- make `domain.setbc` which uses one BC data function on all segments
- eigenvalue problems: MPS, scaling method
- segment methods to create analytic interpolant function from boundary point data, enabling user to specify a segment using points on a curve.
- add better automated ways to choose MFS charge points, based on JCP paper
- add layer-potential bases, and associated periodic quadrature methods
- write `segment.bdrysolution` which evaluates u, u_n on one or other side of a boundary. For layer potentials this would take into account jump relations. This should then be used in `problem.fillbcmatrix`
- keep discrepancy and evaluation matrices, for efficiency in multiple right-hand sides.
- classes which symmetrize basis sets for single reflection, C_4, etc symmetry, by wrapping the calls to basis evaluations using reflection points
- dielectric constants, overall frequency, automatically changing k in domains
- periodic boundary conditions connecting values and derivatives on given sides of two different segments, needed for photonic crystals
- allow nufbbasis to accept empty k input, like other basis types.
- make better addnufbbasis which attaches to a requested corner.
- basisgeom for nufbbasis, split off utils.plotfan from domain.plot and use.

7 Known bugs

Please alert the authors to any bugs you discover, including a description of how to reproduce the behavior, using the interface at
`http://code.google.com/p/mpspack/issues/list`

References

[1] A. H. Barnett and T. Betcke, *Stability and convergence of the Method of Fundamental Solutions for Helmholtz problems on analytic domains*, J. Comput. Phys., *in press* (2008). `arXiv:0708.3533 [math.NA]`.