Statistical Topology and the Random Interstellar Medium

Code and Data

Contents

1 Introduction

To reproduce the results in the paper you will need the GASS data as a text file `gass.txt` and a suite of R routines `stattop.r`, both of which are provided.

If you prefer to download the original data yourself, Section 2 provides instructions for downloading the data as seven separate data cubes. We have provided Matlab code `combineFits.m` to create `gass.txt` from these.

The remaining sections describe how to reproduce the analysis and results in the paper. For all of these, the first step is to source the whole of `stattop.r` into R.

2 GASS data

A. How to download the observational data from the GASS data server

Open the link `http://www.astro.uni-bonn.de/hisurvey/gass/index.php`

Fill in your name and working e-mail in the **User information** box in the table and choose the following parameters in the menu.

Data type:	Clean data GASS II	
Coordinate system:	Galactic coordinates (l, b)	
Center:	RA [hms]/l[°]	178
	Dec [±° ′ ″]/b[°]	-50
Width:	RA/l[°]	25
	Dec/b[°]	25
Degree per Pixel:	[°]	0.08
Smoothing Kernel Radius FWHM:	[°]	0.125
Velocity Range (VLSR):	vmin [km/s]	20
	vmax [km/s]	42
	Δv	0.8

Check your input one more time and press **Create job**.

A letter with the link to the chosen data region will be sent to your e-mail address. Download the `.gz` file following the link in the letter.

As the server does not allow to download a data cube larger than 25 degrees in longitude width, you must repeat the procedure described above with different values of **Center**: RA [hms]/l[°], namely change 178 to 216, 254, 292, 330, 8, 46 in turn. Keep all other parameters in the table the same. Be sure that every time you choose GASS II (not GASS III) as **Data type**.

B. How to combine the data into a single file for analysis in R

1. Put all 7 downloaded .gz files in one folder. Download and put the matlab script `combineFits.m` in the same folder.
2. Open Matlab. The folder with your data must be open as a Current folder in the Matlab window.
3. To concatenate all data cubes, integrate along the line-of sight velocity and get Figure 1 from the article you must run our matlab script. At the matlab command line, indicated by the prompt (>>), enter

```
>> combineFits
```

The field will be visualised and there will be three variables in the Matlab workspace:

- **data** is the observational field of $T(l, b)$ shown in Figure 1 of the paper;
- **L** are longitude values for the field;
- **B** are latitude values for the field.

To save the variables to `.txt` files:

```
>> save('gass.txt','data','-ascii')
>> save('L.txt','L','-ascii')
>> save('B.txt','B','-ascii')
```

3 Getting started in R

As stated, the whole of `stattop.r` should be sourced into R. At the time of writing we are using R version 3.5.3 and we require the following packages:

- TDA
- RandomFields
- mvtnorm
- mgcv
- plot3D

To read the GASS data and prepare for analysis, run

```
> setrealdat()
```

This produces three objects (`obs1, obs2, obs2`), one for each region. The objects contain the data, persistence diagrams and various summaries. There is further information as comments in the code.

The routine includes an option to choose one of five different ways of detrending (as in the supplementary material):

- `setrealdat(trend=1)` Low order polynomial
- `setrealdat(trend=2)` High order polynomial
- `setrealdat(trend=3)` Default thin plate smoothing spline
- `setrealdat(trend=4)` Thin plate smoothing spline, higher penalties
- `setrealdat(trend=5)` No trend removal

The default is `trend=3`.

4 Introductory example

Figure 2 of the main paper shows four level sets for a simple 10×10 simulated field. This plot can be obtained via

```
> getbetplot()
```

The field itself is shown in Figures 1a and 1b of the supplementary material, as a 3D barplot and a greyscale image respectively. These can be obtained by:

```
> getexampleplot(use3d=TRUE)
```

```
> getexampleplot(use3d=FALSE)
```

5 Simulations and analytic results

Figure 4 of the supplementary material shows five simulated 256×256 fields, one for each of the five distributions described in Section 4 of the main paper. These fields can be obtained via

```
> plotsims()
```

If you want further illustrations, use

```
> plotsims(useseed=FALSE)
```

Table 1 of the paper gives the expected number of components or holes for Gaussian random fields. Table 2 gives the approximate standard deviations together with simulation equivalents as a check. These two tables can be obtained by

```
> gettab1()
```

and

```
> tbs12=gettab2(useseed=TRUE,seed=10)
```

Getting Table 1 takes about one minute to run. (All times in this document were obtained on a Dell OptiPlex7050 desktop computer running Ubuntu). Table 2 is slower and so a counter is printed to help monitor progress. The routine takes about one hour to complete.

If you just want a quick look at some special cases you might try, for example

```
> ensims(d=64,nsim=50,usesquare=TRUE)
```

or perhaps

```
> ensims(d=64,nsim=50)
```

The output is the mean number of features, the standard deviation and the standard error of the mean.

We now turn to Table 3 of the main paper, which compares the correlations for the five distributions at various distances. This can be obtained using:

```
> getcordat()
```

Again a counter helps monitor progress. This routine takes about 12 minutes to run.

Figure 4 of the paper plots the geometric summary statistics for the same simulated fields as for Table 3. To get the plot use:

```
> geomplot()
```

This routine also takes about 12 minutes.

The sizes and power for the nonparametric tests for differences between single realisations of fields are given in Table 4. The results are based on 1000 simulations from each distribution, with topology of nine subregions being calculated for each simulated field. Our routine to generate multiple simulations and calculate the geometric summaries is:

```
> simsums=splitsims(simstart=1,simend=1000)
```

This routine takes about 2.5 hours to run. Because it is so slow, the results have been saved and provided to the journal:

```
> write.table(simsums,"simsums.dat",row.names=FALSE)
```

Finally for the simulation section, to complete Table 4:

```
> pow=splittests(simsums=read.table("simsums.dat",header=TRUE))
```

There will be some warning messages related to ties in the Wilcoxon tests. These can safely be ignored.

The fifth column of the results matrix `pow` has the sizes and powers for the combined tests. The third and fourth columns use tests based on counts and filamentarity separately. The first two columns index the distributions being compared.

6 Analysis of GASS data

In Section 5 of the main paper we consider the topological properties of the three GASS regions. We obtain persistence diagrams and convex peels for Region 1 in Figure 3 of the main paper:

```
> getpersfig()
```

And for all three regions in supplementary material Figures 2 and 3:

```
> getallpers()
```

```
> getallconvpeels()
```

In Figure 5 of the main paper we compare the three convex peels for each of components and holes and we show how the number of features accumulate as we filter through the level sets:

```
> getappfig()
```

Table 5 summarises the counts, and Table 6 tests for differences between regions using the sub-region approach with Wilcoxon tests on the filamentarity and number statistics:

```
> getcountsum()
```

```
> compregions()
```

The final table to be reproduced is Table 7, which explores the use of bottleneck and/or Wasserstein distances in comparing regions. This is obtained from

```
> getdistsum()
```

It is quite slow, taking about an hour to complete. A counter runs to 26 27 three times.

7 Miscellaneous

The instructions above detail how to replicate the results in the paper. Input parameters can be changed for further exploration of properties if desired: see the comments in the routines themselves.

In addition, if you want to explore further yourself, the following may be useful:

```
> dat=simdat(d=256,eta1=20,nu1=0.5,dist="Gauss",
+ usedefault=FALSE, usesquare=FALSE)
```

This generates a 256×256 Gaussian random field, with the cross neighbourhood for connections between pixels. The field is generated with a Matern correlation function with shape and scale parameters 0.5 and 20 respectively. The other distributions used in the paper can be obtained by setting `dist` to `chisq`, `chisq3`, `T`, or `F`. In those cases the root correlation in generating the underlying Gaussian random fields has the Matern form. Setting `usedfault=TRUE` generates the fields used in the paper.

The simulated field is in a matrix `dat$fmat`. It can be visualised using any standard tool, such as

```
> image(dat$fmat)
```

Points in the persistence diagram are in `dat$pers`. To plot for components and holes respectively use

```
> persplot1(dat$pers,usetype=1)

> persplot1(dat$pers,usetype=2)
```

To find the convex peels you first need to extract the x and y coordinates, which are in the second and third columns of the persistence diagram object. The first column indicates components (0) or holes (1). For example, for components:

```
> x=dat$pers[dat$pers[,1]==0,2]

> y=dat$pers[dat$pers[,1]==0,3]
```

To plot the convex peels down to 95% of points and save the final convex hull:

```
> ch=convpeelplot(x,y,prop=0.95)
```

You can then get the geometric summaries of the final convex hull (in order centroids, area, perimeter, filamentarity) using

```
> chsum(ch)
```