spmR: an R package for fMRI data analysis

Yves Rosseel
Department of Data Analysis
Ghent University

Workshop on Statistics and Neuroimaging 2011
WIAS, Berlin, November 25, 2011

Overview

1. Software for fMRI data analysis (in R)
2. What is spmR?
3. What has (NOT) been ported?
4. How did we proceed?
5. Example
6. The future of spmR

Software for fMRI data analysis

- SPM (Matlab)
- FSL (binary, written in C and C++)
- AFNI (binary, written in C)
- BrainVoyager (closed-source)
- …
- Neuroimaging in Python (`http://nipy.sourceforge.net/`)

Software for fMRI data analysis in R

- CRAN Task View: Medical Image Analysis
  - **AnalyzeFMRI** (GLM + ICA, includes tk/tcl based GUI)
  - **arf** (Activated Region Fitting; uses Gaussian shape spatial models to parameterize active brain regions)
  - **fmri** (structural adaptive smoothing methods)
  - **neuroim** (R-Forge: S4 classes for handling brain imaging data)
  - **cudaBayesreg** (provides a CUDA implementation of a Bayesian multilevel model for the analysis of brain fMRI data)
- JSS, Vol. 44 (Oct 2011): Special Volume on Magnetic Resonance Imaging in R
  - **arf3DS4** (arf with S4 classes)
  - **neuRosim** (simulating fMRI data)

What is spmR?

- **spmR** is an R package for fMRI data analysis
- **spmR** is nothing more than an R port of (parts of) the widely used SPM package (`http://http://www.fil.ion.ucl.ac.uk/spm`)
- for standard fMRI analyses, the **spmR** package can be used as a plugin replacement for SPM, yielding exactly the same results
- **spmR** can be used instead of SPM:
  - if the Matlab environment is not available (for example in high-performance computing environments)
  - if the fMRI analysis is just a part of a larger pipeline which is entirely written in R
  - if you need to understand what SPM is doing and you are more comfortable reading R code
- only fMRI (no EEG, PET, . . .)

How to get it?

- **spmR** is not on CRAN
- you can install it from our local R archive:

```
install.packages("spmR",
                 repos="http://www.da.ugent.be")
```

- current version: 0.8-1
- no documentation, only source code, a few functions, and 1 example script

Which parts of SPM have been ported?

- using **spmR** you can:
  - read in a 4D nifti file (**spmR** uses Rniftilib)
  - specify your design
  - fit the GLM for each voxel (using the SPM algorithms)
  - optionally write out images for the parameters/residuals
  - specify and estimate one or several T/F contrasts
  - optionally write out contrast images and an SPM{T} or SPM{F} map
  - corrections for multiple testing (Random Field, FWE only)
  - print the results table

Which parts of SPM have NOT been ported (yet)?

- no GUI
- no preprocessing
- no second-level analysis (multiple subjects)
- multiple sessions: not fully tested
- conjunctions (combining several contrasts): not fully implemented
- FDR: not finished yet
- …

How did we proceed? (1)

- studying the SPM5 Matlab code + literature: trying to figure out what is happening (somewhere in 2008)
- version 0.1 – 0.6: translating the 'logic' in R
  - using `lm()` to fit the regression model
  - writing all c-code in R
  - representation of voxel data: just a 4D array (no `spm_vol`, `spm_get_dat`
- big update: from SPM5 to SPM8: many changes
- more updates, more changes, giving up
- version 0.7: starting over again, but staying much closer to the original Matlab code
- version 0.8: adding some c-code again, adding multiple testing code

How did we proceed? (2)

- the source directory in spmR contains two types of files:
  - `spm_*` files correspond very closely to the original SPM matlab files:
    - `spm_add.R`
    - `spm_bwlabel.R`
    - `spm_Ce.R`
    - `spm_clusters.R`
    - `spm_conman.R`
    - `spm_contrasts.R`
    - `spm_dctmtx.R`
    - `spm_defaults.R`
    - `spm_dx.R`
    - `spm_est_smoothness.R`
    - `spm_FcUtil.R`
    - `spm_filter.R`
    - `spm_fMRI_design.R`
    - `spm_get_bf.R`
    - `spm_get_lm.R`
    - `spm_get_ons.R`
    - `spm_getSPM.R`
    - `spm_global.R`
    - `spm_hrf.R`
    - `spm_list.R`
    - `spm_max.R`
    - `spm_orth.R`
    - `spm_P_Bonf.R`
    - `spm_P.R`
    - `spm_P_RF.R`
    - `spm_Q.R`
    - `spm_reml.R`
    - `spm_resels_vol`
    - `spm_resss.R`
    - `spmr.R`
    - `spm_sample_vol`
    - `spm_spm.R`
    - `spm_sp.R`
    - `spm_SpUtil.R`
    - `spm_uc.R`
    - `spm_Volterra.R`
  - `spmr_*` files for user-visible functions
- the SPM object in spmR contains the same fields as the SPM.mat object in Matlab/SPM

Example Matlab file

```
function P = spm_P_Bonf(Z,df,STAT,S,n)
% Returns the corrected P value using Bonferroni


if      STAT == 'Z'
  P = 1 - spm_Ncdf(Z);
elseif  STAT == 'T'
  P = 1 - spm_Tcdf(Z,df(2));
elseif  STAT == 'X'
  P = 1 - spm_Xcdf(Z,df(2));
elseif  STAT == 'F'
  P = 1 - spm_Fcdf(Z,df);
end

P = S*P.^n;
P = min(P,1);
```

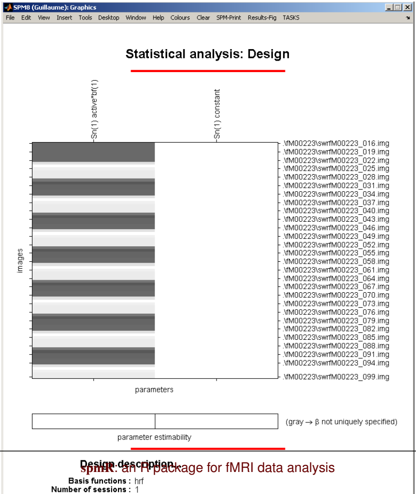
Example R file

```
spm_P_Bonf <- function(Z,df,STAT,S,n) {
   if(STAT == "Z") {
        P <- 1 - pnorm(Z)
    } else if(STAT == "T") {
        P <- 1 - pt(Z, df[2])
    } else if(STAT == "X") {
        P <- 1 - pchisq(Z, df[2])
    } else if(STAT == "F") {
        P <- 1- pf(Z, df[1], df[2])
    } else {
        stop("wrong value for STAT argument", STAT)
    }

    P <- S*P^n
    P <- min(P, 1.0)
    P
}
```

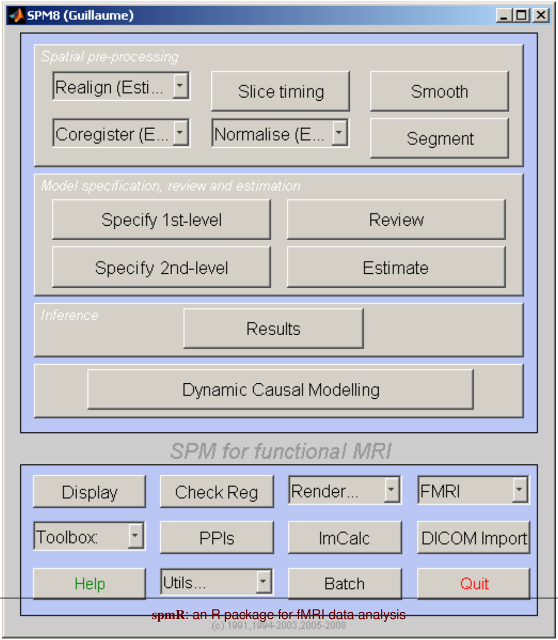
Example: Auditory dataset from the SPM manual

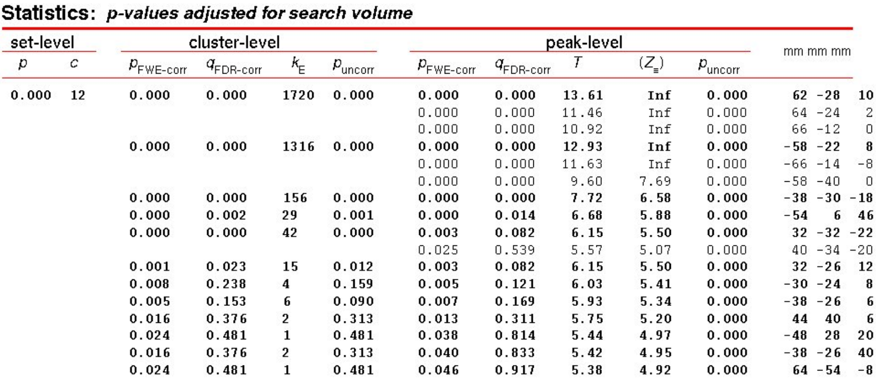
- the ‘auditory fMRI data’ from the SPM manual
- single subject/session, 84 scans, block design (30s On, 30s Off)



Design description...

Basis functions : hrf
Number of sessions : 1


SPM8 (Guillaume)
Spatial pre-processing
Realign (Esti...
Slice timing
Smooth
Coregister (E...
Normalise (E...
Segment
Model specification, review and estimation
Specify 1st-level
Review
Specify 2nd-level
Estimate
Inference
Results
Dynamic Causal Modelling
SPM for functional MRI
Display
Check Reg
Render...
FMRI
Toolbox:
PPIs
ImCalc
DICOM Import
Help
Utils...
Batch
Quit
(c) 1991,1994-2003,2005-2008



Statistics: ***p-values adjusted for search volume***

set-level		cluster-level				peak-level					mm	mm	mm
p	c	$p_{\text{FWE-corr}}$	$q_{\text{FDR-corr}}$	k_E	p_{uncorr}	$p_{\text{FWE-corr}}$	$q_{\text{FDR-corr}}$	T	$(Z_\equiv)$	p_{uncorr}			
0.000	12	0.000	0.000	1720	0.000	0.000	0.000	13.61	Inf	0.000	62	-28	10
						0.000	0.000	11.46	Inf	0.000	64	-24	2
						0.000	0.000	10.92	Inf	0.000	66	-12	0
		0.000	0.000	1316	0.000	0.000	0.000	12.93	Inf	0.000	-58	-22	8
						0.000	0.000	11.63	Inf	0.000	-66	-14	-8
						0.000	0.000	9.60	7.69	0.000	-58	-40	0
		0.000	0.000	156	0.000	0.000	0.000	7.72	6.58	0.000	-38	-30	-18
		0.000	0.002	29	0.001	0.000	0.014	6.68	5.88	0.000	-54	6	46
		0.000	0.000	42	0.000	0.003	0.082	6.15	5.50	0.000	32	-32	-22
						0.025	0.539	5.57	5.07	0.000	40	-34	-20
		0.001	0.023	15	0.012	0.003	0.082	6.15	5.50	0.000	32	-26	12
		0.008	0.238	4	0.159	0.005	0.121	6.03	5.41	0.000	-30	-24	8
		0.005	0.153	6	0.090	0.007	0.169	5.93	5.34	0.000	-38	-26	6
		0.016	0.376	2	0.313	0.013	0.311	5.75	5.20	0.000	44	40	6
		0.024	0.481	1	0.481	0.038	0.814	5.44	4.97	0.000	-48	28	20
		0.016	0.376	2	0.313	0.040	0.833	5.42	4.95	0.000	-38	-26	40
		0.024	0.481	1	0.481	0.046	0.917	5.38	4.92	0.000	64	-54	-8

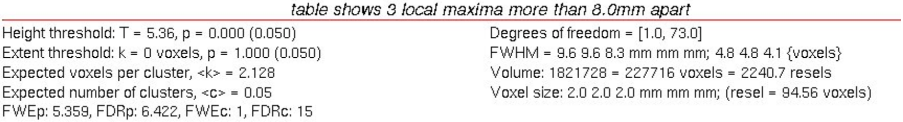


table shows 3 local maxima more than 8.0mm apart

Height threshold: T = 5.36, p = 0.000 (0.050)
Extent threshold: k = 0 voxels, p = 1.000 (0.050)
Expected voxels per cluster, <k> = 2.128
Expected number of clusters, <c> = 0.05
FWEp: 5.359, FDRp: 6.422, FWEc: 1, FDRc: 15

Degrees of freedom = [1.0, 73.0]
FWHM = 9.6 9.6 8.3 mm mm mm; 4.8 4.8 4.1 {voxels}
Volume: 1821728 = 227716 voxels = 2240.7 resels
Voxel size: 2.0 2.0 2.0 mm mm mm; (resel = 94.56 voxels)

specify 1st level (1)

```
library(spmR)

# first session/subject
session1 <- list()
session1$scans <- c("auditory.nii.gz")
session1$nscans <- 84

# first condition
condition1 <- list(name="Condition 1",
                   onset=c(6, 18, 30, 42, 54, 66, 78),
                   duration=6
                  )
session1$cond <- list(condition1)
```

specify 1st level (2)

```
SPM <- spmr_fmri_specify_1stlevel(RT=7,
                                  units="scans",
                                  bf.name="hrf",
                                  sess=list(session1),
                                  cvi="AR(1)",
                                  output.dir=""
                                 )

# bf.name = hrf (with time and dispersion derivatives)"
```

estimate model

```
SPM <- spmr_fmri_estimate(SPM, method="classical")
```

specify and estimate t - contrasts

```
Tcontrast1 <- list(type="tcon", name="active > rest",
                   contrast=c(1,0) )

Tcontrast2 <- list(type="tcon", name="active < rest",
                   contrast=c(-1,0) )

SPM <- spmr_contrasts(SPM, consess=list(Tcontrast1,
                                         Tcontrast2))

# optionally: write out SPM t/F images
spmr_write_images(SPM, type="SPM")
```

inference: compute (corrected) p-values

```
Table <- spmr_results(SPM, type="table")

# print table
round(Table[,c(1,2,3,5,6,7,9,10,11,12,13,14)], 3)
attr(Table, "footer")
```

```
   p  c cl.p.FWE.corr cl.kE cl.p.unc p.FWE.corr      T   (Z) p.unc   X   Y   Z
1   0 12         0.000  1720    0.000      0.000 13.611   Inf     0  62 -28  10
2  NA NA            NA    NA       NA      0.000 11.458   Inf     0  32 -32 -22
3  NA NA            NA    NA       NA      0.000 10.924   Inf     0  40 -34 -20
4  NA NA         0.000  1316    0.000      0.000 12.934   Inf     0 -58 -22   8
5  NA NA            NA    NA       NA      0.000 11.629   Inf     0  32 -32 -22
6  NA NA            NA    NA       NA      0.000  9.604 7.694     0  40 -34 -20
7  NA NA         0.000   156    0.000      0.000  7.724 6.582     0 -38 -30 -18
8  NA NA         0.000    29    0.001      0.000  6.682 5.883     0 -54   6  46
9  NA NA         0.000    42    0.000      0.003  6.153 5.504     0  32 -32 -22
10 NA NA            NA    NA       NA      0.025  5.567 5.066     0 -32 -32 -20
11 NA NA         0.001    15    0.012      0.003  6.151 5.503     0  32 -26  12
12 NA NA         0.008     4    0.159      0.005  6.031 5.415     0 -30 -24   8
13 NA NA         0.005     6    0.090      0.007  5.927 5.338     0 -38 -26   6
14 NA NA         0.016     2    0.313      0.013  5.745 5.201     0  44  40   6
15 NA NA         0.024     1    0.481      0.038  5.439 4.967     0 -48  28  20
16 NA NA         0.016     2    0.313      0.040  5.422 4.954     0 -38 -26  40
17 NA NA         0.024     1    0.481      0.046  5.384 4.924     0  64 -54  -8
```

```
> attr(Table, "footer")
[1] "Height threshold: T = 5.36, p = 0.000 (0.050)"
[2] "Extent threshold: k = 0 voxels, p = 1.000 (0.050)"
[3] "Expected voxels per cluster, <k> = 2.128"
[4] "Expected number of clusters, <c> = 0.05"
[5] "FWEp: 5.359, FDRp: NA, FWEc: NA, FDRc: NA"
[6] "Degrees of freedom = [1.0, 73.0]"
[7] "FWHM = 9.6 9.6 8.3 mm mm mm; voxels = 4.8 4.8 4.1 {voxels}"
[8] "Volume: 1821728 = 227716 voxels = 2240.7 resels"
[9] "Voxel size: 2.0 2.0 2.0 mm mm mm;(resel = 94.56 voxels)"
```

The future of spmR

- once your package is on CRAN, people will start using it, and ask for support, more features, …
- I do not wish to be the maintainer of such a package (cfr. **lavaan**)
- spmR as part of a larger package for fMRI data analysis?

```
fit <- estimateModel(myModel,
                     data=myData,
                     ...,
                     mimic="SPM")
```

- open for discussion