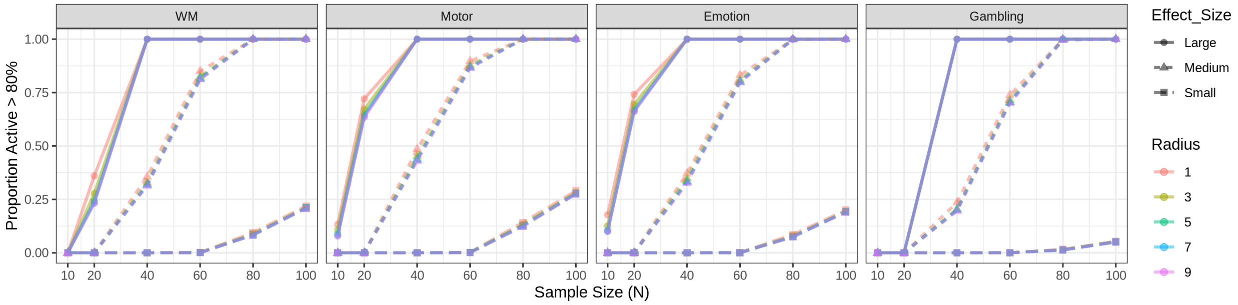

Figure S1: Proportion of active voxels estimated using an uncorrected threshold (0.001) for the locally moderated t-statistic with different neighborhoods. Results are organized by task and effect size group. Smaller neighborhoods provide improved results. Note if two lines overlap their colors are mixed.

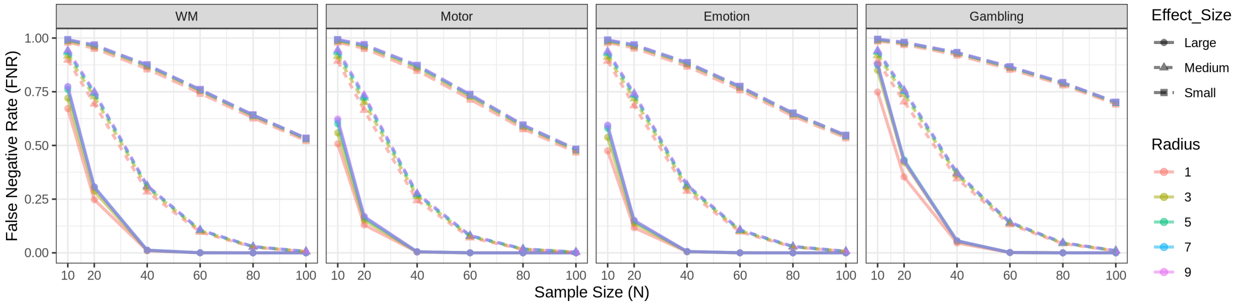

Figure S2: False negative rates estimated using an uncorrected threshold (0.001) for the locally moderated t-statistic with different neighborhoods. Results are organized by task and effect size group. Smaller neighborhoods provide improved results. Note if two lines overlap their colors are mixed.

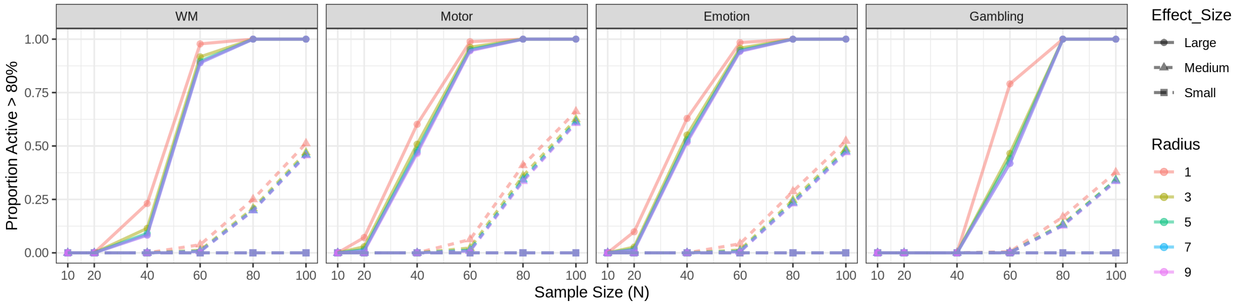

Figure S3: Proportion of active voxels estimated using an FWER-corrected threshold (0.05) for the locally moderated t-statistic with different neighborhoods. Results are organized by task and effect size group. Smaller neighborhoods provide improved results. Note if two lines overlap their colors are mixed.

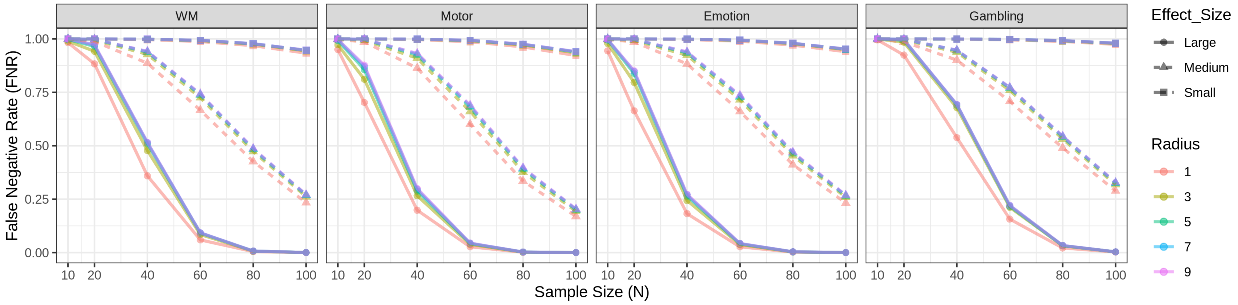

Figure S4: False negative rates estimated using an FWER-corrected threshold (0.05) for the locally moderated t-statistics approach with different neighborhoods. Results are organized by task and effect size group. Smaller neighborhoods provide improved results. Note if two lines overlap their colors are mixed.

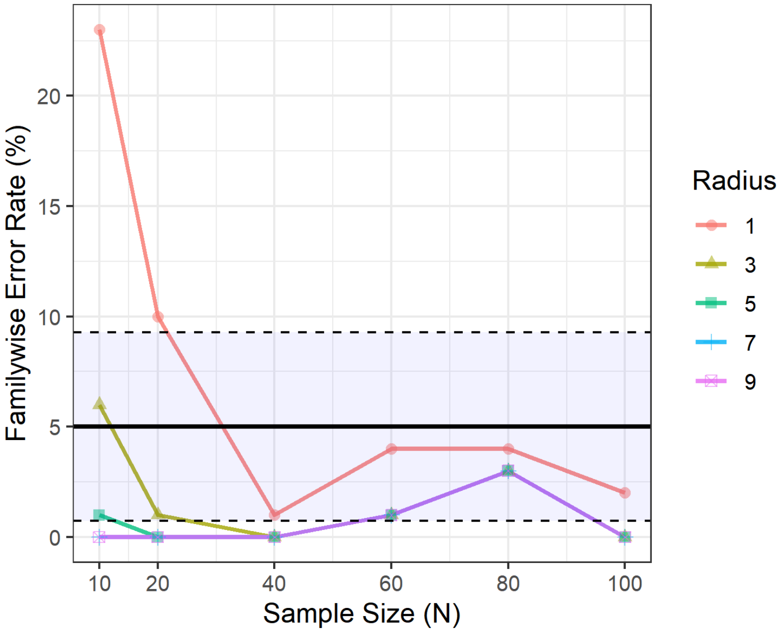

Figure S5: The empirical error control for the locally moderated t test, estimated using the fake 'working memory' data. The results show the FWER corresponding to different neighborhood sizes and sample sizes. In general, the FWER is controlled at the 5% level, indicated by the solid horizonal black line. However, neighborhoods with smaller radius have excessive FWER at small sample sizes. Note if two lines overlap their colors are mixed.

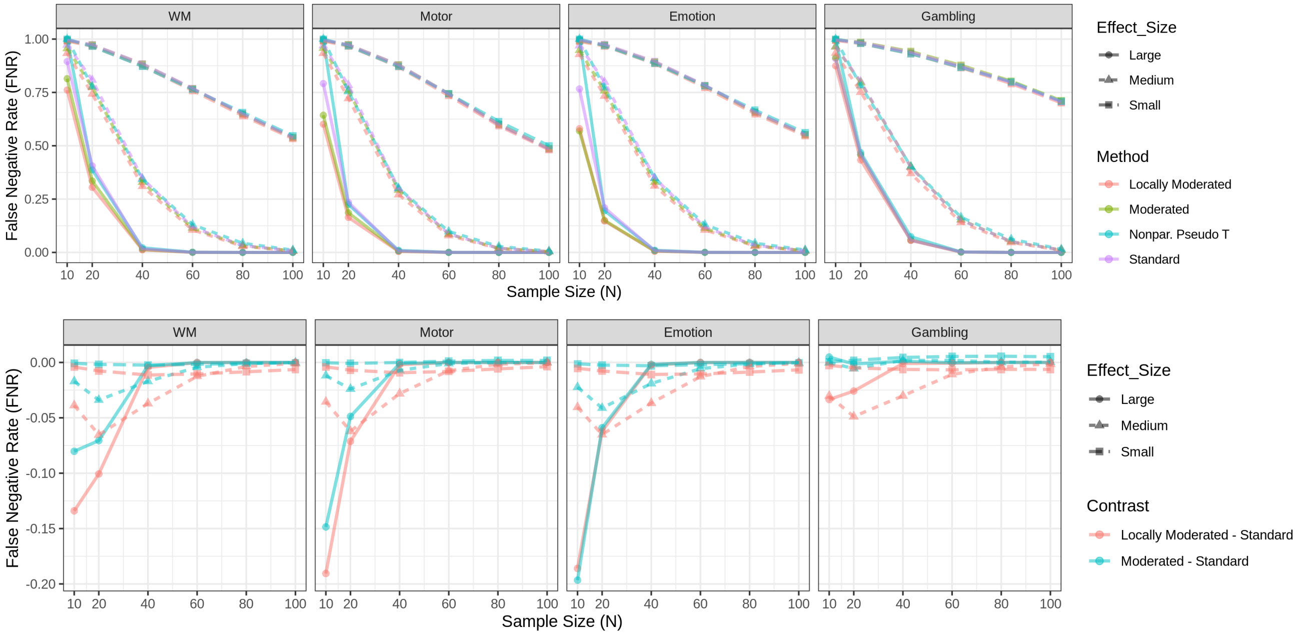

Figure S6: False negative rates for each task using an uncorrected p-value threshold at 0.001. (Top) Rates are shown for four different methods: standard t-statistics, moderated t-statistics, locally moderated t-statistics with $r = 5$, and non-parametric pseudo t-statistics. Rates are stratified according to effect size group. (Bottom) The difference in false negative rate between moderated/locally moderated t-statistics and the standard t-statistic is shown for each task and effect size group. The results highlight the differences in results for small sample sizes. The results illustrate that locally moderated t-statistics performs significantly better than the other methods for small sample sizes. Note if two lines overlap their colors are mixed.

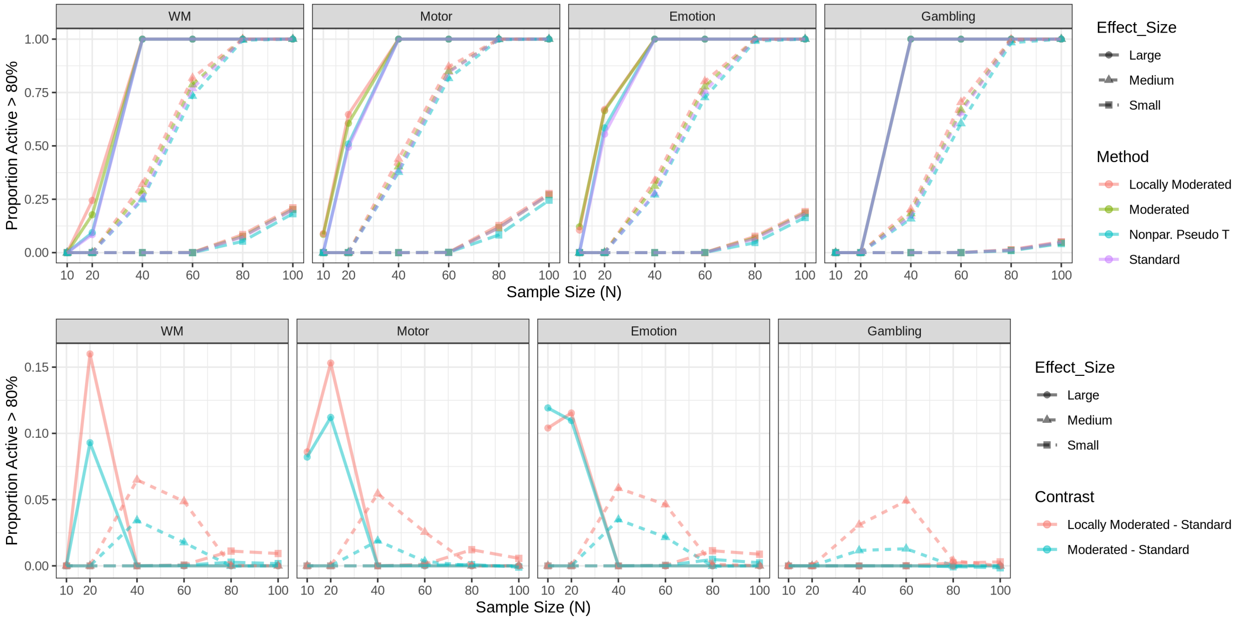

Figure S7: Proportion of active voxels in each task using an uncorrected p-value threshold at 0.001. (Top) Proportions are shown for four different methods: standard t-statistics, moderated t-statistics, locally moderated t-statistics with $r = 5$, and non-parametric pseudo t-statistics. Proportions are stratified according to effect size group. (Bottom) The difference in proportions between moderated/locally moderated t-statistics and the standard t-statistic is shown for each task and effect size group. The results shows the proportion is always larger for moderated t-statistics, particularly for small sample sizes.Note if two lines overlap their colors are mixed.

A Sample code for using the limmi R package

The example data is taken from the HCP motor task described in Section 2.2.2. The data consists of COPEs comparing finger-tapping vs baseline for $n = 15$ subjects. Before using the `limmi` package for the first time, it should be downloaded from GitHub. This can be done by typing the command:

```
install_github("muschellij2/limmi")
```

Note you will need to have the `devtools` package installed and loaded prior to running this command.

We begin the analysis by loading the necessary R-packages.

```
library(RNifti)
library(limmi)
library(neurobase)
```

Note these packages may need to be installed prior to first use.

The data we use in this example is in a folder consisting of two sub-directories. The first is named `Overlay` and contains an anatomical T1 image `single_subj_T1.nii` taken from SPM12[2]. The second is named `Motor_Data` and contains first-level COPEs from 15 subjects. These COPEs are taken from the HCP database, but have been renamed in this illustration.

We first create a variable `files` that contains paths to each of the 15 COPEs we will use in our second-level analysis.

```
files = list.files(path = "Motor_Data/", pattern=’.nii’)
files = file.path("Motor_Data", files)
files
[1] "Motor_Data/subject01.nii" "Motor_Data/subject02.nii" "Motor_Data/subject03.nii"
[4] "Motor_Data/subject04.nii" "Motor_Data/subject05.nii" "Motor_Data/subject06.nii"
[7] "Motor_Data/subject07.nii" "Motor_Data/subject08.nii" "Motor_Data/subject09.nii"
[10] "Motor_Data/subject10.nii" "Motor_Data/subject11.nii" "Motor_Data/subject12.nii"
[13] "Motor_Data/subject13.nii" "Motor_Data/subject14.nii" "Motor_Data/subject15.nii"
```

Next, we read in the anatomical T1 image that will be used later as an overlay to present results.

```
img = readNIfTI("Overlay/single_subj_T1.nii", reorient = FALSE)
```

To help facilitate the analysis, it is useful to create a mask excluding voxels whose values are zero for all subjects. As part of this process we first use the function `nifti_images_to_matrix` to place the COPEs into matrix format.

```
mat = nifti_images_to_matrix(files, verbose = FALSE)
```

[2] Statistical Parametric Mapping, version 12; http://www.fil.ion.ucl.ac.uk/spm/

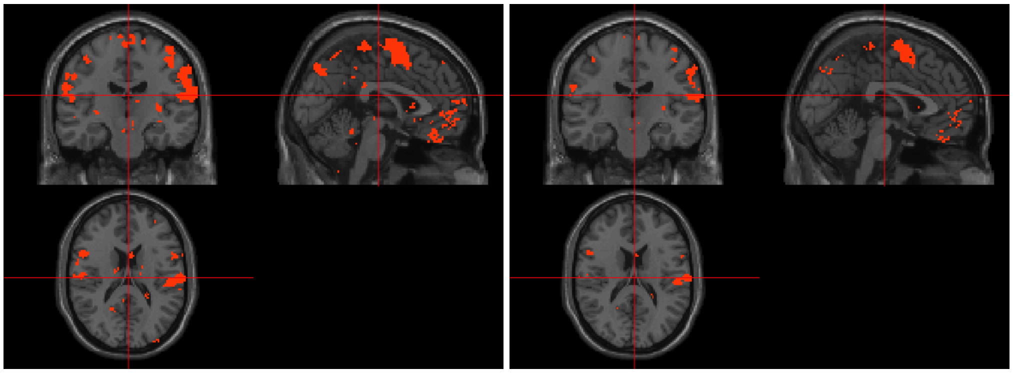

Figure S8: (Left) The results of the second-level analysis using the moderated t-statistic, with thresholding performed at p-value < 0.001 uncorrected. (Right) The results of the second-level analysis using the moderated t-statistic, with thresholding performed at p-value < 0.01 FDR-corrected using the Benjami-Hochberg approach.

```
keep = rowSums(mat > 0)
mask = array(keep, dim = dim(img))
mask = asNifti(mask, reference = img)
```

Now we are ready to use the function `nifti_eBayes` to compute the moderated t-statistic. This function takes the variables `files` and `mask` and computes the moderated t-statistic for all voxels included in the mask.

```
gb_eb = nifti_eBayes(files, mask, verbose = FALSE)
```

To visualize the results we create a map indicating voxels with p-values below 0.001 overlayed on the anatomical image; see Fig. S8 (left panel).

```
neurobase::ortho2(img, gb_eb$images$p.value <= 0.001, pdim = pdim)
```

By default, `nifti_eBayes` does a `topTable` from `limma` to adjust the p-values using the Benjamini-Hochberg approach. We can show results using these values by replacing `gb_eb$images$p.value` by `gb_eb$images$adjusted_p_value` in the command above; see Fig. S8 (right panel).

```
neurobase::ortho2(img, gb_eb$images$adjusted_p_value <= 0.01, pdim = pdim)
```

In addition, the command provides results for the standard t-statistic. We can show results using these values by replacing `gb_eb$images$p.value` by `gb_eb$images$standard_t_stat_p_val` in the command above.

Next, we run `findNearestNeighbors` to create a list of the neighbors within a specified radius for each voxel. The function `nifti_local_moderated_t_stat` can then be used to produce moderated t-statistics with respect to these pre-defined neighbors. The computation can be parallelized by specifying the parameter `mc.cores`.

```
# Specify the radius of the neighborhood
r = 2
findnn = findNearestNeighbors(maskImg = mask, radius = r, threads = 8)
```

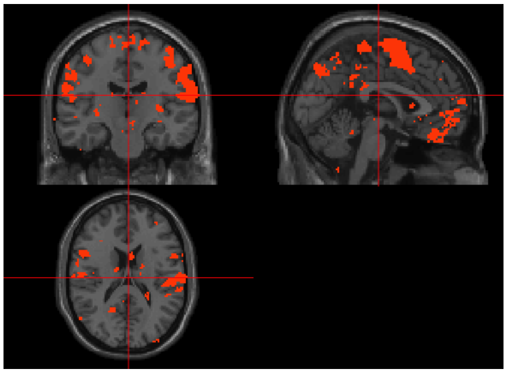

Figure S9: The results of the second-level analysis using the locally moderated t-statistic, with thresholding performed at p-value < 0.001 uncorrected.

```
localt = nifti_local_moderated_t_stat(imgs = files, mask = mask, nn = findnn,
                                      radius = r, adjust.method = "BH", mc.cores = 1)
```

To visualize the results we create a map indicating voxels with p-values below 0.001 overlayed on the anatomical image; see Fig. S9.

```
neurobase::ortho2(img,localt$pvalmap <= 0.001, pdim = pdim)
```