1. Network under study.

The PPI PSP network. The Largest Connected Component (LCC): N = 4495, E = 20444.

2. The Clustering Algorithms

Clustering the pre- and post-synaptic PPI networks was performed using a non exhaustive set of commonly used clustering algorithms. Modularity-Maximisation based algorithms tested include the popular agglomerative 'Fast-Greedy Community' algorithm (fc) (Clauset et al., 2004)[1], process driven agglomerative random walk algorithm 'Walktrap' (wt) (Pons & Latapy, 2006)[2], and coupled Potts/Simulated Annealing algorithm 'SpinGlass' (sg) (Reichardt & Bornholdt, 2006; Traag & Bruggeman, 2008)[3,4], the divisive spectral based 'Leading-Eigenvector' (lec) (Newman, 2006)[5] and fine-tuning (Spectral) (Mclean et al., 2016)[6] algorithms, and the hierarchical agglomerative 'Louvain' algorithm (louvain) (Blondel et al., 2008)[7]. Non-Modularity based algorithms used include the information -theoretic based 'InfoMAP' algorithm (infomap) (Rosvall et al., 2007; Rosvall et al., 2010)[8,9], and the Bayesian statistical inference Mixed-Membership Stochastic Blockmodel (MMSBM), to find overlapping community structure, (SVI) (Gopalan & Blei, 2013) [11].

We employed a reclustering approach to these algorithms, to split their super-sized complexes into sub-complexes. Here we define super-sized complexes as those >= 10% of the number of vertices in the network. For example, for the PSP PPI network with 4495 vertices this would imply communities of size >= 449. For each super-sized complex originally uncovered by the clustering algorithm, we reapply that algorithm to those edges and vertices found internal to that complex. Using this approach we reclustered the following algorithms: lec (lec2), fc (fc2), wt (wt2), sgG1 (sgG1_2) and louvain (louvain2).

3. Annotation set for study

GO Biological Process (BP) terms: 115 GO terms related to synaptic function.

4. Enrichment tests.

The hypergeometric distribution (1) was used to calculate the significance of enrichment of each cluster for each annotation:

$$P(X=\mu_{fc};\mu_{fc},Cn,F,N)=\frac{\binom{F}{\mu_{fc}}\binom{N-F}{Cn-\mu_{fc}}}{\binom{N}{Cn}} \quad \text{-(1)}$$

$$P-value(\mu_{fc})=\sum_{i=0}^{\mu_{fc}}\begin{cases}P(X=i) & : \quad P(X=i)\leq P(X=\mu_{fc})\\ 0 & : \quad P(X=i)>P(X=\mu_{fc}).\end{cases} \quad \text{-(2)}$$

$$P-value(\mu_{fc})=\sum_{i=0}^{\mu_{fc}}\begin{cases}P(X=i) & : \quad P(X=i)\geq P(X=\mu_{fc})\\ 0 & : \quad P(X=i)<P(X=\mu_{fc}).\end{cases} \quad \text{-(3)}$$

Where in (1) N is the total number of genes in the network; Cn the number of genes in the community; F the total number of functional annotated genes in the network, and μ_{fc} the number of functional annotated genes per community. Enrichment was calculated using eqn (2), which gives us the one-sided exact Fisher test, while depletion was calculated using alternative test given in eqn (3). P-values calculated where corrected using the Benjamini and Yekutieli (B-Y) [59] procedure, and tested against the more stringent Bonferroni correction at the 0.05 (*), 0.01 (**) and 0.001 (***) significance levels.

NOTE: we can repeat this for the two-sided fisher test, and adjusted p.values.

5. Odds Ratio

We calculate the log of the Odds Ratio (OR) as:

$$\ln(\mathrm{OR})=\ln\left(\frac{(a*d)}{(b*c)}\right)=\ln\left(\frac{\mu_{fc}*(N-F+\mu_{fc}-Cn)}{(Cn-\mu_{fc})*(F-\mu_{fc})}\right) \quad \text{-(4)}$$

And it's upper and lower 95% Confidence Interval:

$$\mathrm{CI(ln(OR))}=\ln(\mathrm{OR})\pm 1.96*(1/a+1/b+1/c+1/d)^{1/2} \quad \text{-(5)}$$

A description of the Odds Ratio and 95% Confidence Interval can be found here:

[1] Bland, J. B. and Altman, D. G. The odds ratio. BMJ, 2000;320:1468, (2000).

[2] Szumilas, M. Explaining Odds Ratios. J. Can. Acad. Child. Adolesc. Psychiatry, 19(3): 227-229, (2010).

Alg	**FN**	**CN**	**FN*CN**	**P <= 0.05**	**Palt <= 0.05**	**ln(OR) > 0**	**ln(OR) > 0 at lower 95% CI**	**P sig.**	**Palt sig.**	**% of Enriched Coms**
lec	115	7	805	207	124	346	107	107	0	13.3
Louvain	115	17	1955	253	137	490	156	151	0	7.72
Louvain2	115	73	8395	516	76	2091	564	454	0	5.40
SVI	115	63	7360	407	76	2269	441	380	0	5.16
sgG1	115	35	4025	314	32	585	219	185	0	4.60
lec2	115	59	6785	362	148	1429	375	291	0	4.29
Fc	115	35	4025	247	67	570	208	164	0	4.07
sgG1_2	115	85	9775	453	110	1501	457	364	0	3.72
fc2	115	94	10810	438	94	1776	497	368	0	3.40
Spectral	115	130	14950	497	66	1713	480	377	0	2.52
infomap	115	319	36685	949	22	3957	1555	920	0	2.50
wt2	115	1356	155940	1794	32	3101	1727	1009	0	0.65
wt	115	1215	139725	1494	33	2253	1272	761	0	0.54

Table 1: Ratio of functionally enriched communities for each algorithm using the GO BP terms.

Where in Table 1 'FN' is the number of GO BP terms. 'CN' is the number of Communities. 'FN*CN' is the number of functional communities, i.e. the number of GO BP terms * the number of Communities. 'P <= 0.05', the number of functional communities with p.value (see eqn (2)) <= 0.05. 'Palt <= 0.05' the number of functional communities with p.value (see eqn (3)) <= 0.05. 'ln(OR) >0', the number of functional communities where log of the Odd Ratio (see eqn (4)) is > 0. 'ln(OR) >0 at lower 95% CI', is the number of functional communities where log of the Odds Ratio and it's lower 95% CI (see eqn (5)) are both > 0. 'Psig', is the number of functional communities with P <= 0.05 AND ln(OR) >0 at the 95% CI. 'Palt.sig', is the number of functional communities with Palt <= 0.05 AND ln(OR) >0 at the 95% CI. '% Enriched Coms', is the ratio of 'P.sig' to 'FN*CN'.

The reason for us wanting to use the Odds Ratio, see Table 1, is to distinguish functionally enriched communities relative to functionally depleted communities. It can be seen from Table 1 that ranking algorithms according to the percentage of functionally enriched communities does not tell us the size of community the enrichment originates from, i.e. if enrichment occurs within a ‘super-sized’ community or small (< 5 nodes) community.

6. Fold-enrichment (Fe)

For each algorithm, we plotted the fraction of functionally enriched communities (see P.sig in Table 1) greater or equal to the log of its Fold-enrichment (Fe) value, measured at 100 intervals taken from 0 to the maximum Fe value (the maximum found from all algorithms studied). The distribution for each algorithm is shown in Figure 1.

$$\log 2(\mathrm{Fe}) = \log 2\left(\frac{\left(\frac{\mu_{fc}}{F}\right)}{\left(\frac{Cn}{N}\right)}\right) \qquad \textbf{- (6)}$$

Eqn (6) gives the Fold-enrichment value. For each algorithm we measure the difference in fraction of enriched communities between log2(Fe) ‘cut off’ values of 0.5 (this translates to interval point 7 out of 100) and 4.8 (which is the interval point 54 out of 100). This difference is recorded in Table 2. Figure 3 illustrates why we might choose the the cut-off values, but it might make more sense to compare the distribution to how closely they fit to a sigmoid function?

Alg	**Frac. Enriched Coms log2(Fe) >0.5 && log2(Fe) < 4.8**	**Frac. Enriched Coms log2(Fe) >4.8 && log2(Fe) < 8.0**
SVI	0.89	0
Louvain2	0.82	0.12
Spectral	0.49	0.08
lec2	0.48	0.09
sg1_2	0.48	0.18
fc2	0.45	0.14
infomap	0.22	0.67
fc	0.19	0.16
Wt2	0.17	0.3
wt	0.13	0.25
louvain	0.12	0.06
sgG1	0.04	0.13
lec	0.01	0

Table 2: Ranking of the algorithms according to the highest difference in fraction of enriched communities, as measured between log2(Fe) >= 0.5 and log2(Fe) <= 4.8.

7. Sigmoid Fit

We tested how well each distrubution fitted to a generalised sigmoid function:

$$y=a+\frac{b-a}{1+\exp^{-c(x-d)}} \quad \text{-(7)}$$

Where ‘a’ gives the sigmoid’s lower asmotote, ‘b-a’ the sigmoid’s maximum value, ‘c’ the rate of change of the sigmoid and ‘d’ the x value of the sigmoids midpoint. An ideal sigmoid for our case would occur when: a = 0, b = 1, c = -2, and d = 3. Of the four parameters, it is the rate of change of the sigmoids curve which is of interest to us. We tested how well each distribution (in Figure 1) fitted to a set of five ‘idealised’ sigmoid function with the other parameters fixed, and ‘c’ set too [-10, -5, -2, -1, -0.5]. We used the two-sample Kolmogorov-Smirnov (KS) test to the goodness of fit of each distribution to our set of five idealised sigmoid curves.

We used R’s “minpack.lm” non-linaear fitting package to fit the generalised sigmoid function to each distribution given in Figure 1. In addition to fitting each distribution, we also tested how well the distributions fitted the sigmoid function when noise was added to each data point. We added noise to each data point by randomly sampling from a Gaussian distribution with mean zero, and standard deviation of [0.01, 0.05, 0.1, 0.5].

From Tables 3 and 4, and Figures 4 and 5 that we cannot rule out the louvain2 and SVI algothims following a sigmoid distribution with rate of change c=-2. We see the sgG1_2, Spectral and fc2 algorithms are more likely to follow sigmoid distribution with shallower slopes, i.e. c=-0.5, which we would expect from algorithms generating more smaller (< 5 nodes) communities.

Alg	**Converge**	**'c' = -10**	**'c' = -5**	**'c' = -2**	**'c' = -1.0**	**'c' = -0.5**
lec	YES	0	0	0	0	0
Louvain	YES	E-12	E-12	E-13	0	0
Louvain2	YES	E-7	E-5	0.2	0.008	E-7
SVI	YES	E-5	0.003	0.02	0.003	E-8
SgG1	YES	E-11	E-11	E-9	E-7	E-6
lec2	NO	E-10	E-9	E-7	0.0006	0.0007
Fc	NO	E-11	E-11	E-11	E-11	E-12
sgG1_2	YES	E-10	E-9	E-7	0.0003	0.0002
fc2	NO	E-11	E-10	E-9	E-6	0.005
Spectral	YES	E-11	E-10	E-8	E-5	E-5
infomap	NO	E-12	E-12	E-13	E-15	0
wt2	NO	E-12	E-12	E-14	E-16	0
wt	YES	E-12	E-12	E-14	E-16	0

Table 3: KS-test goodness of fit p.values, fitting each distribution (no noise) to generalised sigmoid function (eqn 7), compared to our set of five 'idealised' sigmoid curves with c = [-10, -5, -2, -1, -0.5].

Alg	**Converge**	**‘c’ = -10**	**‘c’ = -5**	**‘c’ = -2**	**‘c’ = -1.0**	**‘c’ = -0.5**
lec	YES	0	0	0	0	0
Louvain	YES	E-12	E-12	E-14	0	0
Louvain2	YES	E-8	E-6	0.06	0.005	E-6
SVI	YES	E-8	E-6	0.04	0.0001	E-8
SgG1	YES	E-11	E-11	E-10	E-8	E-5
lec2	NO	E-10	E-9	E-7	0.001	0.008
Fc	NO	E-11	E-11	E-11	E-10	E-11
sgG1_2	YES	E-11	E-10	E-8	E-5	0.0002
fc2	YES	E-11	E-10	E-8	E-6	0.002
Spectral	YES	E-10	E-9	E-7	0.0003	0.0001
infomap	NO	E-12	E-12	E-13	E-16	0
wt2	YES	E-12	E-12	E-13	E-15	0
wt	YES	E-12	E-12	E-14	0	0

Table 4: KS-test goodness of fit p.values, fitting each distribution with noise (randomly sampling from Gaussian distribution with mean=0, sd=0.1) to generalised sigmoid function (eqn 7), compared to our set of five ‘idealised’ sigmoid curves with c = [-10, -5, -2, -1, -0.5].

8. Discussion/Points

A large drop off at the beginning of these plots is indicative of enrichment occurring in either large communities, or in communities with small mu_fc relative to F. We don’t want the first case, and probably less interested in the second case? High values of log2(Fe) implies small communities sizes or large mu_fc relative to F. The first case we don’t want, the second case we are interested in. Algorithms which still have a high faction of enriched communities at the second cut-off value (i.e. log2(Fe) = 4.8) are infomap, wt, wt2, which have a vast number of small, i.e. < 5 nodes, in size. In general, the reclustered algorithms outperform (looking at the rankings in Table 2), just applying the algorithm as it is.

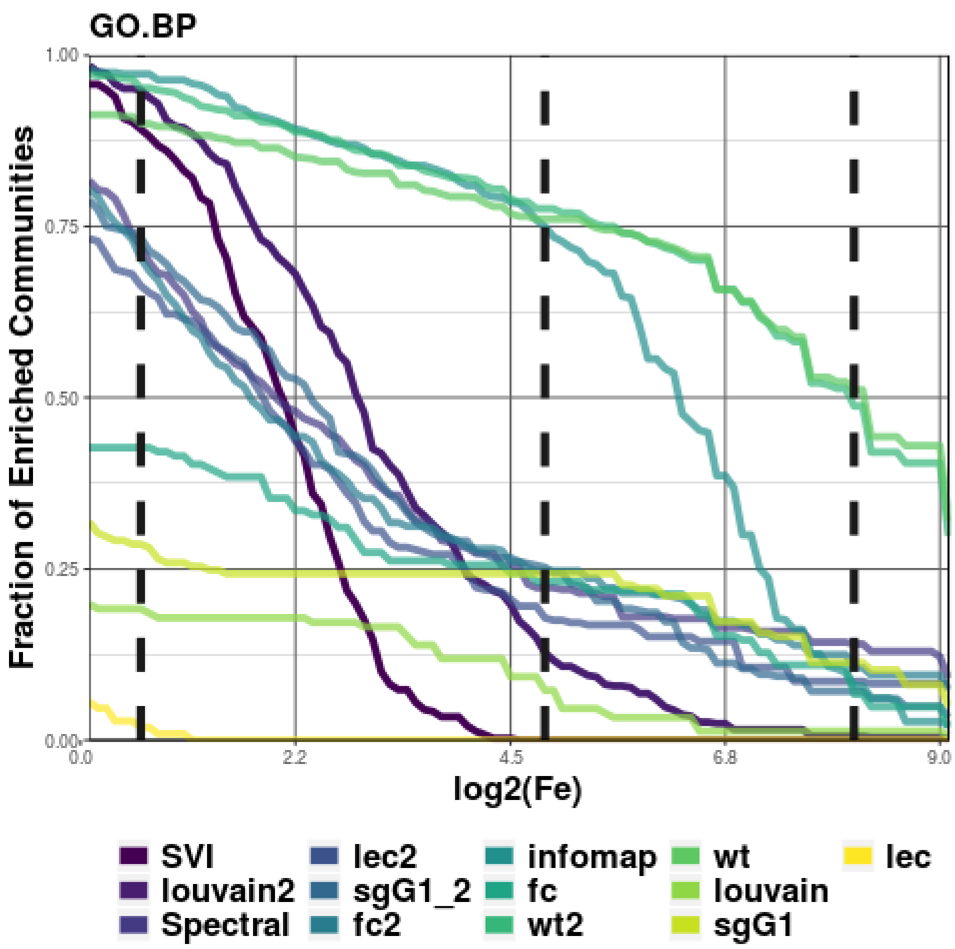

Figure 1: Distribution of the Fraction of enriched communities for each algorithm, >= the log of it's Fold-enrichment (Fe) value, measured over an interval of 100 points from 0 to log(Fe) max.

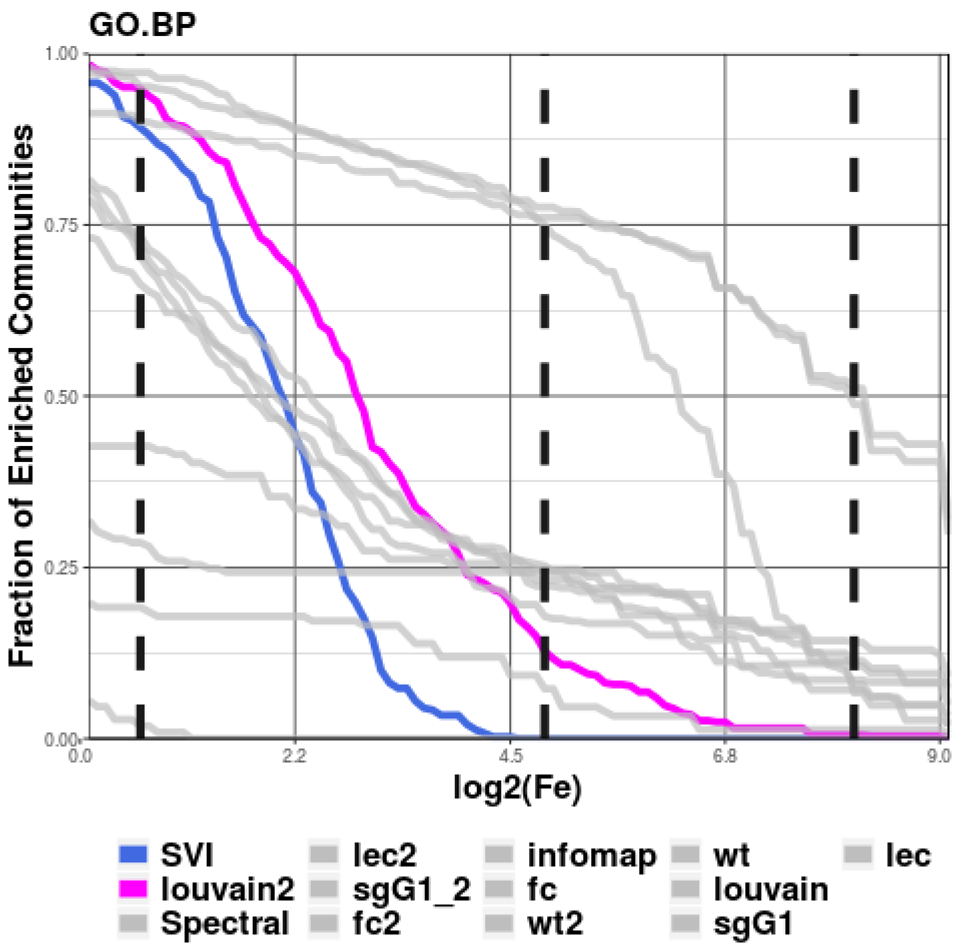

Figure 2: Highlighting the top two ranked algorithms from Table 2.

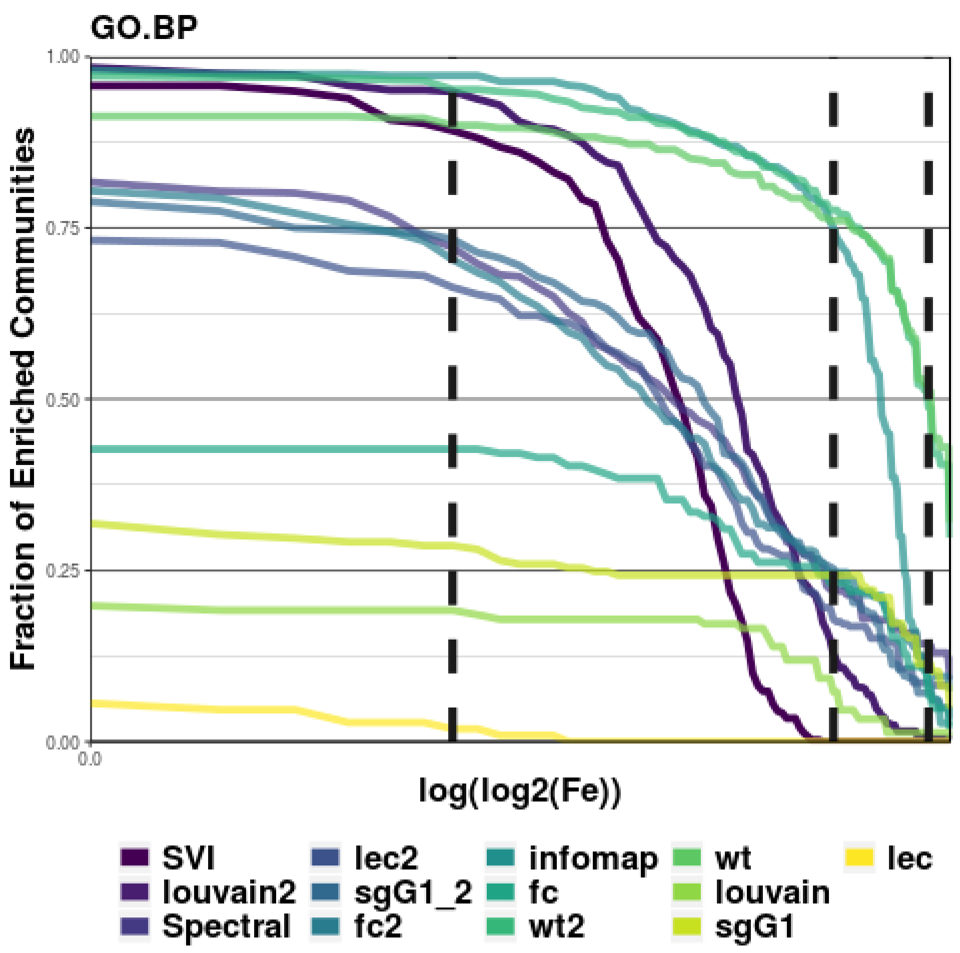

Figure 3: Dashed lines at interval points 7, 54 and 90. log2(Fe) at 7 = 0.5, log2(Fe) at 54 = 4.8, and log2(Fe) at 90 = 8.0.

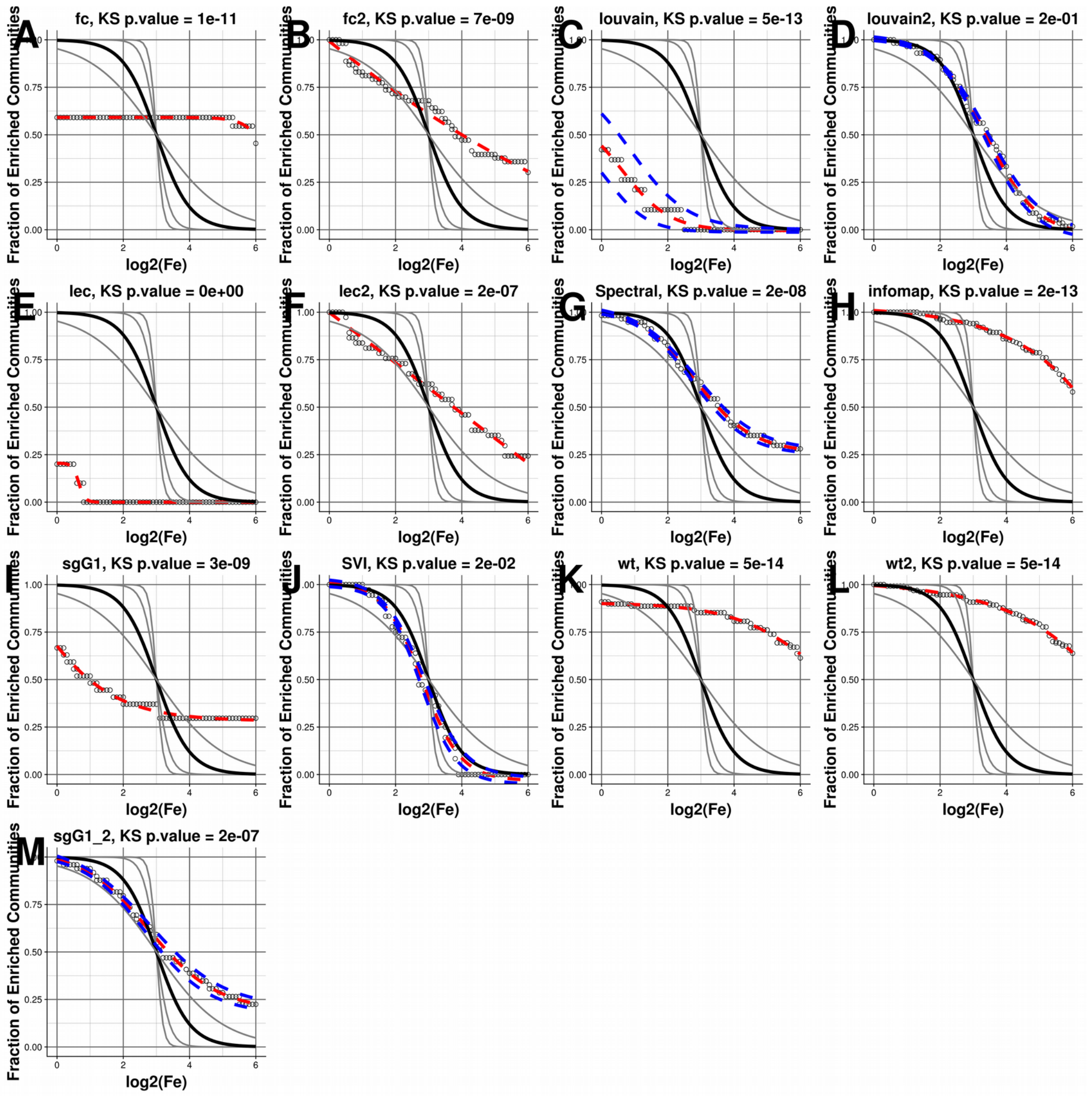

Figure 4: Sigmoid curve fit (red dashed line, with 95% CI blue dashed line) to each distribution without noise, and KS-test goodness of fit p.values relative to idealised sigmoid curve (black line) with c=-2.

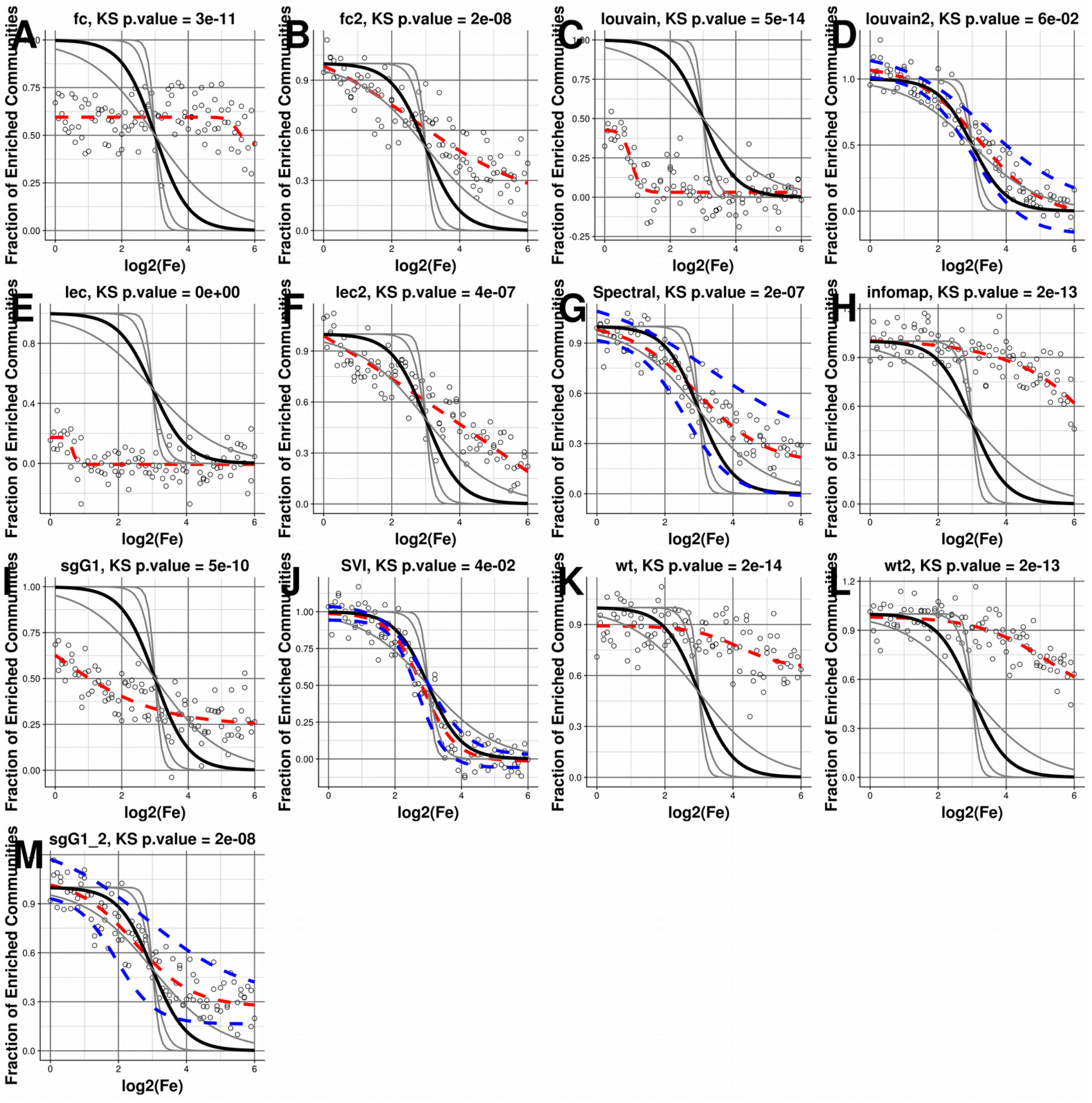

Figure 5: Sigmoid curve fit (red dashed line, with 95% CI blue dashed line) to each distribution with noise (randomly sampling from Gaussian distribution mean=0, sd=0.1), and KS-test goodness of fit p.values relative to idealised sigmoid curve (black line) with c=-2.