A general pipeline for quality and statistical assessment of protein interaction data using R and Bioconductor

Tony Chiang[1,2] & Denise Scholtens[3]

[1]EMBL-EBI, Wellcome Trust Genome Campus, Hinxton, Cambridge, UK. [2]Department of Computational Biology, Fred Hutchinson Cancer Research Center, Seattle, Washington, USA. [3]Department of Preventive Medicine, Northwestern University, Chicago, Illinois, USA. Correspondence should be addressed to D.S. (dscholtens@northwestern.edu).

Published online 26 March 2009; doi:10.1038/nprot.2009.26

© 2009 Nature Publishing Group http://www.nature.com/natureprotocols

The systematic mapping of protein interactions by bait–prey techniques, including affinity purification-mass spectrometry or the yeast two-hybrid system, contributes a unique and relevant perspective on the comprehensive picture of cellular machines. We describe here a protocol for statistical analysis of node-and-edge graph representations of these data using R and Bioconductor, recognizing that steps may be added or omitted depending on the data set at hand. The fundamental purpose of such analyses is feature estimation, defined here as the estimation of data-type-specific biological features, such as protein complex composition and the physical interaction integrity of known or estimated complexes. In preparation for feature estimation tasks, we outline a progression through three analytic components common to all bait–prey data types: preliminary setup, exploratory analysis and quality assessment. The end result is a collection of descriptive and inferred characteristics of the data, ready for biological interpretation in a computationally tractable form.

INTRODUCTION

Determining protein interaction partners is essential to deciphering cellular machinery, as proteins often work together to form modular functional units. Data describing these interactions provide one point of view in the study and modeling of complex cellular systems. Different types of protein interactions are required to accomplish biological activities and can be detected by various ‘bait–prey’ assay systems. For example, yeast 2-hybrid (Y2H) technology aims to detect ‘direct’ physical interactions between protein pairs[1–7]. Affinity purification or coimmunoprecipitation (Co-IP) followed by mass spectrometry (AP-MS and CoIP, respectively) assay ‘indirect’ protein complex comembership interactions[8–15]. Data from these technologies can be recorded in node-and-edge graphs, with nodes representing genes/proteins and edges denoting observed interactions.

In response to the recent large-scale generation of high-throughput protein interaction data sets, statistical models have been designed to estimate relevant features given the observed data. Several models can be used to quantify classic features of experimental data, including measurement error and bias, as specifically noted for both AP-MS and Y2H bait–prey graphs[16–18]. The results can then be used to estimate biological features uniquely relevant to different bait–prey data types, e.g., protein complexes from AP-MS data[19]. Rigorous statistical analyses of protein interaction data, along with similarly careful treatments of other genomics and proteomics data types, are crucial for understanding comprehensive cellular processes.

This protocol provides computational guidelines for analyses of bait–prey protein interaction data using the statistical environment R (see ref. 20) and the relevant software packages available through the Bioconductor project[21]. We develop one specific pipeline in support of our proposed protein complex feature estimation methodologies[16,17,19]; a wide variety of other software and analysis techniques with different outcomes exist for protein interaction data and merit exploration by the user[22–25]. We assume availability of R on the user’s computer, basic familiarity with R programming as well as an experimental design that yields bait–prey data recordable on a graph. After providing preliminary setup instructions for data compilation into a uniform data structure, we guide the user through exploratory analyses and quality assessment techniques that are generally applicable to all bait–prey data. Using the results from the exploratory analyses and quality assessment, we perform feature estimation depending on the type of protein interactions included in the data set of interest. Here, we address specific options for AP-MS data and Y2H data. Other bait–prey systems, e.g., synthetic genetic technologies, have their own uniquely estimable features but are not addressed here.

One of the fundamental limitations of data analyses is the frequency with which the results are irreproducible. To ameliorate this artifact, we have written this document as a part of a compendium that weaves the written text with the data and code used to generate the results (**Supplementary Method 1** online: *ppiProtocol.Rnw*). By obtaining the compendium used to create this document, the reader may not only reproduce all of the analyses conducted in this protocol, but also modify or supplement the original methodologies. The code without the text used to generate this document is also provided (**Supplementary Method 2** online: *ppiProtocol.R*). Additionally, a file containing the references (**Supplementary Method 3** online: *ppiProtocol.bib*) and three figure files (**Supplementary Figs. 1–3** online: *directedGraph.pdf, flowchart.pdf* and *Gavin2006Screenshot.png*) are required to run *ppiProtocol.Rnw*. The data package *ppiProtocolData_1.1.0.tar.gz* (**Supplementary Data 1** online) contains all data sets used in these analyses.

A second limitation of the analyses presented here is the rapidity with which the software evolves, particularly as it is open source. At the end of this protocol, we document the version of the software packages used in the analysis lest the end user is unable to

© 2009 Nature Publishing Group http://www.nature.com/natureprotocols

reproduce the results. In R, once all the software packages have been loaded into the environment, a call of the function *sessionInfo()* shows the version of R and all the add-on packages.

Graphical representation of bait–prey data

A graph can be represented as G = (V,E), where V is the set of nodes and E is the set of edges between the nodes. For our purposes, the nodes in V represent genes/proteins and the edges in E represent detected interactions. (For a technical and detailed treatment on graphs, see refs. 26 and 27. For specific applications in molecular biology, see refs. 28–30.) In the analyses presented here, the true protein interaction graph is assumed to be an undirected graph with symmetric edges connecting pairs of interacting proteins. In contrast, the observed protein interaction graph is a directed graph with edges extending from baits to prey in accordance with interactions detected by the affiliated technologies.

Bait–prey technologies test for all relationships involving a set of prespecified baits and (i) prespecified prey (such as in the Y2H technologies) or (ii) all other prey proteins expressed under the cellular conditions of the experiment (such as Co-IP or AP-MS). Experimental results are typically reported only for proteins that are observed in at least one interaction, leaving uncertainty as to which proteins were available at the time of interaction testing. To ameliorate this problem, Chiang *et al.*[16] introduced the concepts of viable baits (VBs) and viable prey (VP), the former being the set of bait proteins that successfully detect at least one interaction partner and the latter being the set of all proteins detected by at least one bait. It is possible for a protein to be both a viable bait and a viable prey (VBP), a viable bait-only (VBO) protein or a viable prey-only (VPO) protein. If a protein is neither a VB nor a VP, it is generally not reported with the data set.

Figure 1 shows the importance of distinguishing between viable baits and prey in an experiment using a simple data set containing seven interactions between bait–prey pairs. Such a data set would commonly be reported with no distinction between bait and prey. This practice leads to ambiguity when an edge is missing between two proteins, as it is unclear if the absence of an edge represents a tested but unobserved interaction or a relationship that has yet to be queried. In a directed graph representation that distinguishes between baits and prey, it is evident that edges between pairs of VBP proteins are tested bidirectionally, edges connecting VBs to VPOs tested unidirectionally and edges within pairs of VBOs or VPOs not tested at all. Such a distinction is crucial for basing statistical inference only on observed data. All analyses in this protocol require such a distinction and a directed graph representation of the data.

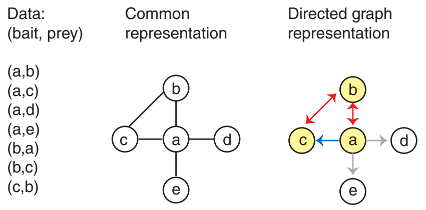


Figure 1 | Seven ordered pairs representing protein interaction (bait, prey) pairs. The graph on the left is a common graphical representation of the data in which there is no distinction between baits and prey and thus ambiguity as to the number of assays performed on each edge. The directed graph representation on the right brings clarity to the reported data, differentiating between bidirectionally tested edges connecting pairs of VBP nodes, unidirectionally tested edges between VBs and VPOs and untested edges between pairs of VPOs. The yellow nodes represent VBP nodes, whereas white nodes represent VPO nodes. Red arrows indicate bidirectionally tested and observed edges; blue arrows indicate bidirectionally tested but unidirectionally observed edges; gray arrows indicate unidirectionally tested and observed edges.

Protocol design

The analysis of protein interaction action data as outlined in this protocol can be partitioned into four distinct categories:

- Preliminary setup
- Exploratory analysis
- Quality assessment
- Feature estimation

In our approach, preliminary setup, exploratory analysis and quality assessment are uniformly applicable to all bait–prey data sets, whereas feature estimation depends on the nature of the data being analyzed. The flow chart in **Figure 2** lays out the order of analysis followed in this protocol, differentiating between generally applicable and data-type-specific analysis steps. From Steps 1–7 of this protocol, we describe the methodology, as it pertains to the analysis of a collection of protein interaction data sets. These steps allow the user to conduct analyses on a number of data sets. From Step 8 onward, we focus our attention on a single data set, as the procedures and methodology become more technical. In principle, all the steps taken in this protocol may be conducted either on a

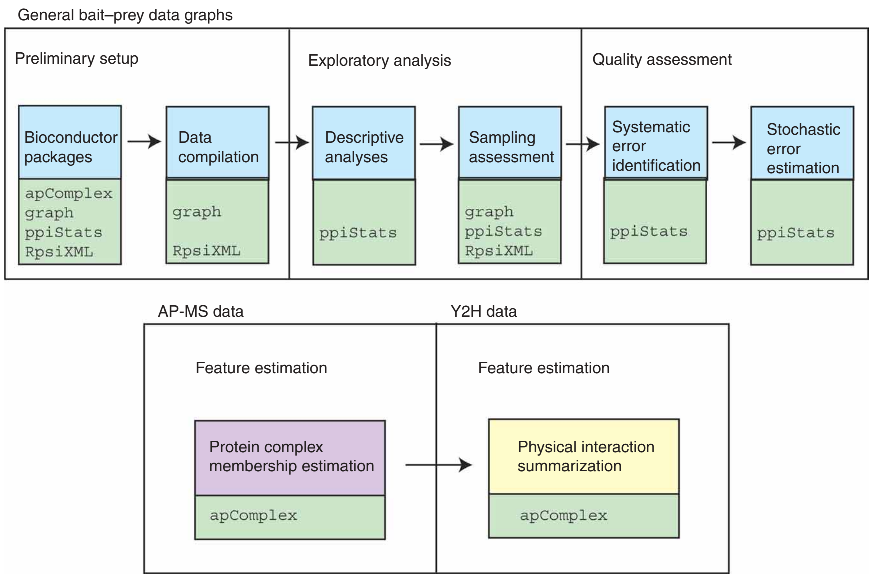


Figure 2 | Flowchart describing the outline of analyses presented in this protocol. The top panel is relevant to analysis of general bait–prey graphs and each blue box contains the type of suggested analyses. The bottom panel outlines data-type–specific steps with the suggested analyses highlighted in different colors according to data type. The green boxes contain the names of the Bioconductor packages required to execute the analyses for each step as outlined in this protocol.

© 2009 Nature Publishing Group http://www.nature.com/natureprotocols

single data set or multiple data sets in parallel. Before enumerating a step-by-step procedure for analysis, we first discuss the various components of our full bait–prey graph analysis.

Preliminary setup

Bioconductor packages. Four Bioconductor packages and their dependencies are used in the analyses presented in this protocol: *apComplex*, *graph*, *ppiStats* and *RpsiXML*. See http://cran.r-project.org and http://www.bioconductor.org for installation instructions for R and the Bioconductor packages, respectively. In addition, the data sets used in this specific protocol are available in a package called *ppiProtocolData* (**Supplementary Data 1**).

Data compilation. Data must be obtained either from primary sources or from repositories in which they are stored and then transformed into a uniform data structure, e.g., an object of class *psimi25Graph* or *graphNEL* as described in PROCEDURE. We assume that a protein interaction data set consists of a uniform naming scheme for the interactors, clearly defines the type of interaction assayed (i.e., Y2H or AP-MS) and identifies the bait and prey for each interacting pair.

Exploratory analysis

Descriptive analyses. To characterize the sampled bait population and the covered prey population, we ascertain the following sets of proteins involved in an interaction in a data set: VB, VP and VBP. These sets serve as reasonable proxies for the bait and prey population when this information is not otherwise available.

Additionally, we ascertain the total number of directed interactions (TI) reported in the data set. Only edges extending from VB nodes to VP nodes are deemed tested. The TI statistic, therefore, must be interpreted against the number of edges queried in the graph, not the total number of possible edges.

Sampling assessment. The selected bait proteins (and prey proteins depending on the technology) can give insight into the sampling scheme used by each investigator and therefore clarify potential feature estimation results. Observed data and therefore analysis results will only be available for proteins that are amenable to the technology and/or expressed under the conditions of the experiment, here called 'viable proteins'. Different data sets covering distinct sections of the interactome may result from intentional bait selection, or from differential response of baits and prey to ambient conditions if the sampled bait population is relatively invariant for the data sets.

The study of protein viability also allows detection of enrichment of genes or proteins with respect to certain functional annotations in the data set. Assuming a hypergeometric distribution, we may discover over- and underrepresentation within certain gene sets, e.g., those defined by Gene Ontology[31] (GO) or Kyoto Encyclopedia of Genes and Genomes[32] pathways. Such analyses may help to elucidate bias for or against certain types of proteins for each experiment and technology.

Quality assessment

Systematic error identification. The bait–prey distinction can further help us to dissociate systematic bias from stochastic variation. As two proteins either truly interact or do not in a given condition, with perfectly sensitive and specific technology, we should find reciprocated positive or negative observations if both proteins are VBPs. Actual technologies frequently yield unreciprocated observations signaling some type of measurement error. If unreciprocated interactions were due only to random measurement error, then in-degree and out-degree for any VBP protein should be roughly equal. Using a binomial test to diagnose large disparities of in- and out-degree[16], we can segregate proteins into groups that might be biased in the assay system and groups that do not show this type of bias. Proteins affected by systematic errors should be removed before downstream analysis.

Stochastic error estimation After diagnosing proteins prone to systematic error and removing them from further analysis, we can estimate the stochastic false-negative probability (P_{FN}) and false-positive probability (P_{FP}) of the technology yielding a particular data set using the remaining set of VBP proteins. Chiang *et al.*[16] describe a method of moments approach to estimate a family of solutions for P_{FN} and P_{FP} pairs given an observed VBP graph. Using gold standard (GS) sets of protein interactions, we can identify a single point estimate for P_{FN} from the family of solutions, and then identify the corresponding P_{FP} paired estimate[17]. Estimates of P_{FN} and P_{FP} provide useful comparative measures across experiments and are also necessary parameters for feature estimation.

Feature estimation

Protein complex membership estimation. Having identified the viable bait and prey populations, diagnosed bait proteins subject to systematic error and estimated stochastic error probabilities for a data set, estimation of biological features can proceed. Given a set of protein complex comembership data from an AP-MS or Co-IP experiment, one highly relevant estimation task is that of cataloging protein complex membership. We use a penalized likelihood approach for this estimation task[19]. Complex estimates can be sorted into three different types: multibait-multiedge estimates (MBME), single-bait-multihit (prey) and unreciprocated bait–bait estimates.

Physical interaction summarization. Given a catalog of protein complexes and their members (either known or estimated from AP-MS data), it is natural to overlay the set of physical interactions reported by Y2H experiments. These analyses yield candidate physical structures for individual protein complexes.

MATERIALS

EQUIPMENT

R and Bioconductor We have chosen to work within the R statistical environment[20] and use the freely available and open source catalog of computational libraries that comprise the Bioconductor project[21]. Here, we reference the primary protein interaction data-related software packages that we use for analysis and list the relevant functions for each package in **Table 1**.

R/Bioconductor packages and functions

- *apComplex*: estimates protein complex membership from AP-MS/CoIP data; estimates the physical structures using Y2H data.

TABLE 1 | The relevant classes and functions for bait–prey interaction analysis contained in the packages (given as column headings).

ppiStats	Graph	RpsiXML	apComplex
createSummaryTables	**graphNEL**	**psimi25Graph**	findComplexes
estErrProbMethodOfMoments	ftM2graphNEL	getAbstractByPMID	plotComplex
estimateCCMErrorRates	removeSelfLoops	separateXMLDataByExpt	sortComplexes
findAdjacent	subGraph	translateID	
idSystematic			
idViableProteins			
inOutScatterCharts			
ppiBuildParams4GO			
ppiHGTest4GO			
viabilityCharts			
separateExptBySize			

A complete description of each class object (bold entries) or function (entries in normal font) can be found within its corresponding help page. The uses of all functions are shown within the scope of this protocol.

© 2009 Nature Publishing Group http://www.nature.com/natureprotocols

- *graph*: instantiates the graphical data structures within R.
- *ppiStats*: contains a variety of statistical methods and tests for protein interaction analysis.
- *RpsiXML*: parses molecular interaction databases that implement a compliant PSI-MI XML2.5 schema[33].

Data sets In addition, all data sets used in this document are available in a package called *ppiProtocolData* (**Supplementary Data 1**). Although this package is not required to adapt the analyses presented here to other data sets, it is provided so that users can reproduce the results reported here.

PROCEDURE

Preliminary setup: Bioconductor packages

1| Load Bioconductor packages required for analyses using the library command:

```
> library("apComplex")
> library("graph")
> library("ppiStats")
> library("RpsiXML")
> library("ppiProtocolData")
```

▲ **CRITICAL STEP** Loading of these libraries first requires installation of the packages; see http://www.bioconductor.org/docs/install/for installation instructions.

▲ **CRITICAL STEP** R and Bioconductor are open source and freely available software tools maintained by a number of different contributors. When possible, the present stable released version of each software package is recommended for use.

▲ **CRITICAL STEP** Installation of these packages depends on the installation of the package *Rgraphviz*, which can be difficult on Linux systems. The Bioconductor Mailing List (http://www.bioconductor.org/docs/mailList.html) is a good resource for receiving help in installation.

Preliminary setup: data compilation

2| Obtain protein interaction data either from online databases (option A) or primary sources (option B).

(A) Obtain protein interaction data from online databases

(i) *RpsiXML* communicates with databases that have implemented a compliant PSI-MI XML2.5 schema. At present, these databases include *BioGRID*[34] (http://www.thebiogrid.org/downloads.php), *DIP*[35] (http://dip.doe-mbi.ucla.edu/dip/Download.cgi), *HPRD*[36] (http://www.hprd.org/download), *IntAct*[37] (ftp://ftp.ebi.ac.uk/pub/databases/intact/current/psi25) and *MINT*[38] (ftp://mint.bio.uniroma2.it/pub/release/psi/current/psi25). For this protocol, we work exclusively with the *IntAct* repository. In the example below, 20 XML files of *Saccharomyces cerevisae* protein interactions were downloaded and saved in the *extdata/*directory of the *ppiProtocolData* package. The character vector `filenames` records the relative location of these XML files to the R working directory:

```
> fold <- system.file("extdata",package="ppiProtocolData")
> filenames <- paste(fold,list.files(fold),sep="/")
```

(ii) Create experiment-specific *psimi25Graph* objects from the set of *PSI-MI XML2.5* files using the function *separateXMLDataByExpt* from the package *RpsiXML*. The output of *separateXMLDataByExpt* is a list of *psimi25Graph* objects indexed by pubmed IDs. Each *psimi25Graph* object will have vertices labeled with UniProtKB accession codes (AC) with directed edges representing detected bait–prey interactions among them:

```
> y2hGraphs <- separateXMLDataByExpt(filenames, INTACT.PSIMI25, type = "direct",
directed = TRUE)
> apmsGraphs <- separateXMLDataByExpt(filenames, INTACT.PSIMI25, type = "indirect",
directed = TRUE)
```

▲ CRITICAL STEP Databases, such as *IntAct*, *MINT* and *DIP*, that adhere to the minimal information for molecular interactions schema[39] will have bait and prey information necessary for this analysis.
▲ CRITICAL STEP It is important to keep different interaction types separate; the *type* parameter can be set to *direct* for physical binary interactions or *indirect* for protein complex comembership interactions.
▲ CRITICAL STEP The analysis steps performed in this protocol should be conducted on individual data sets of one particular interaction type without combining data sets into single graph data structures. Informative summary statistics typically vary between experiments, and the type of assay will contribute to different types of systematic errors.
▲ CRITICAL STEP Mixing edges of different types can lead to mis-estimation of biological features.

(B) Obtain protein interaction data from primary sources

(i) Data from primary sources are generally recorded as a two-column table of bait–prey interactions. We obtained the complete list of all putative protein interactions identified by Krogan *et al.*[11] and created the R dataframe `krogan2006bpTable` in the package *ppiProtocolData* to encapsulate the bait–prey interactions recorded therein. The first three rows are printed below:

```
> data(krogan2006bpTable)
> krogan2006bpTable[1:3, ]
Bait Prey
[1,] "YAL001C" "YAL001C"
[2,] "YBR103W" "YAL001C"
[3,] "YBR123C" "YAL001C"
```

(ii) Transform the data frame into a directed graph object using the R function *ftM2graphNEL* from the *graph* package:

```
> krogan2006Graph <- ftM2graphNEL(krogan2006bpTable)
```

▲ CRITICAL STEP Distinguishing between bait and prey proteins in a set of edges is necessary for correctly understanding the data recorded in a representative directed graph (see **Fig. 1**).

3| If desired, partition interaction data into small, medium and large scales by the number of bait–prey interactions using the function *separateExptBySize* from the package *ppiStats*. In the example below, small-scale experiments are arbitrarily defined as having less than 50 interactions, large-scale experiments are those with more than 100 interactions and medium-scale experiments are those in between. The output of *separateExptBySize* is a three-element list where each element contains those graph objects that are deemed small, medium or large, respectively. Splitting the data sets according to size can be helpful depending on the analyses being conducted. We focus on the large-scale data sets for this protocol as the statistical methods are most relevant to larger data sets:

```
> sortedY2H <- separateExptBySize(y2hGraphs, 50, 100)
> sortedAPMS <- separateExptBySize(apmsGraphs, 50, 100)
> names(sortedAPMS)
[1] "smallScale" "mediumScale" "largeScale"
> y2hLargeScale <- sortedY2H[["largeScale"]]
> apmsLargeScale <- sortedAPMS[["largeScale"]]
```

4| If desired, exchange *UniProtKB* node names in graphs for alternative names using the function *translateID* from the package *RpsiXML*. Each database repository will support a limited number of gene/protein names, so it is necessary to find a common naming scheme across databases. For example, translate the nodes from each *psimi25Graph* from UniProtKB AC to ordered locus names (as given by the Comprehensive Yeast Genome Database) and then add the data of Krogan *et al.*[11] manually. Replace the experiment IDs from each *psimi25Graph* from the pubmed ID to the first author and year by obtaining abstract information from NCBI. Manually add the name and date from the data obtained from primary source Krogan *et al*[11] and rename the list of large-scale AP-MS data sets.

```
> apmsLSLocusNamesNodes <- lapply(apmsLargeScale, function(x)
translateID(x,"cygd"))
> apmsLSLocusNamesNodes[[length(apmsLSLocusNamesNodes) + 1]] <-
krogan2006Graph
> apmsAbstract <- getAbstractByPMID(names(apmsLargeScale))
> fauthor <- sapply(apmsAbstract, function(x) {authors(x)[1]})
> datePub <- sapply(apmsAbstract, pubDate)
> expNames <- paste(fauthor, datePub, sep = ":")
> expNames[length(expNames) + 1] <- "NJ Krogan:Mar 2006"
> names(apmsLSLocusNamesNodes) <- expNames
```

© 2009 Nature Publishing Group http://www.nature.com/natureprotocols

TABLE 2 | A general overview of summary statistics for the large-scale AP-MS data sets.

Data sets	VB	VP	VBP	VBP/VB	VP/VB	TI	TI/VB
AC Gavin:Jan 2002	588	1,436	554	0.94	2.44	3,947	6.71
Y Ho:Jan 2002	493	1,315	232	0.47	2.67	3,686	7.48
MD Ohi:Apr 2002	7	44	7	1.00	6.29	167	23.86
P Grandi:Jul 2002	13	66	12	0.92	5.08	405	31.15
J Graumann:Mar 2004	22	376	5	0.23	17.09	563	25.59
TR Hazbun:Dec 2003	33	415	11	0.33	12.58	640	19.39
KK Baetz:Feb 2004	8	26	8	1.00	3.25	132	16.50
R Zhao:Mar 2005	53	74	8	0.15	1.40	133	2.51
AC Gavin:Mar 2006	1,752	1,790	991	0.57	1.02	19,105	10.90
NJ Krogan:Mar 2006	2,264	5,323	2,226	0.98	2.35	63,360	27.99

VB, number of viable baits; VP, number of viable prey; VBP, number of viable baits that are also viable prey; TI, total number of detected interactions. The data sets are named as described in Step 4 of PROCEDURE with the first author's last name and publication year. The statistics in this table can differentiate between experiments for which estimation might not yield the same features. For example, the last column details the number of interactions per VB; the experiments of MD Ohi:Apr 2002, P Grandi:Jul 2002 and J Graumann:Mar 2004 have remarkably high numbers of average interactions per VB, and so we anticipate markedly different feature estimates than that of AC Gavin:Mar 2006.

© 2009 Nature Publishing Group http://www.nature.com/natureprotocols

Exploratory analyses: descriptive analyses

5| Calculate VB, VP, VBP and TI measures for experiments of the same type using the function *createSummaryTables* from the package *ppiStats*. **Table 2** contains output for AP-MS data generated by the code given below:

```
> aTable <-
createSummaryTables(apmsLSLocusNamesNodes[sapply(apmsLSLocusNamesNodes,
isDirected)])
```

6| Generate bar charts summarizing the VBP, VBO and VPO sets relative to the entire yeast genome for each of the experimental data sets using the function *viabilityCharts* from the package *ppiStats*. **Figure 3** contains the bar charts generated by the code below:

```
>viabilityCharts(apmsLSLocusNamesNodes[sapply(apmsLSLocusNamesNodes,
isDirected)])
```

Exploratory analyses: sampling assessment

7| Examine prey partners for specific baits of interest to note sampling variations among experiments using the function *findAdjacent* from the package *ppiStats*. Specify a bait of interest, find its prey partners in the large-scale AP-MS experiments and examine pairwise intersections of detected prey for different experiments. Here, we give one specific example of the differences in prey detected for the same bait YDL047W in the set of large-scale AP-MS experiments:

```
> myBait <- "YDL047W"
> baitSubgraphNodes <- lapply(apmsLSLocusNamesNodes, FUN = findAdjacent,
bait = myBait)
> baitSubgraphNodes[sapply(baitSubgraphNodes, function(x) {!is.null(x)})]
$`AC Gavin:Jan 2002`
[1] "YDL047W" "YNR016C" "YMR024W" "YER155C" "YIL142W" "YLR310C" "YFR019W"
[8] "YGL197W" "YFR040W" "YJL098W" "YKR028W" "YKL195W" "YNL101W" "YNL187W"
[15] "YOR267C"
$`Y Ho:Jan 2002`
[1] "YDL047W" "YMR205C" "YDL029W" "YJL098W"
"YFR040W" "YMR246W" "YJL026W"
[8] "YBR039W" "YKR028W" "YNR016C" "YML048W"
"YJR072C" "YOR317W" "YMR145C"
[15] "YGR282C" "YDR188W" "YJR105W" "YMR028W"
"YHR096C" "YMR196W" "YDL171C"
```

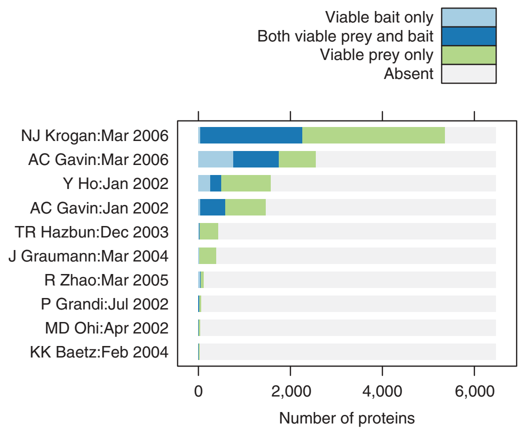


Figure 3 | Viable protein levels encapsulated in bar charts for the AP-MS experiments. The relative proportion of VBP against both VBO and VPO allows not only more statistical power with which to determine systematic versus stochastic errors but also more quantitative estimation of features such as protein complexes. In addition, viable protein levels allow an easy comparative analysis. For instance, with the exception of NJ Krogan:Mar 2006, the remaining large-scale interaction experiments cover at most half of the known yeast protein encoding genes.

© 2009 Nature Publishing Group http://www.nature.com/natureprotocols

```
[22] "YKL029C" "YDR465C" "YOR259C"
$‘AC Gavin:Mar 2006‘
[1] "YDL047W" "YBR118W" "YAL005C" "YBR127C" "YPL106C" "YER155C" "YJL098W"
[8] "YFR040W" "YGL197W" "YNR016C" "YOR267C" "YLL024C" "YDL229W" "YOL127W"
[15] "YGR085C" "YMR024W" "YKR028W" "YNL187W" "YNL101W" "YIL142W" "YLR310C"
[22] "YFR019W" "YKL195W"
$‘NJ Krogan:Mar 2006‘
[1] "YDL047W" "YFR040W" "YJL098W" "YLR097C" "YKR028W"
> intersect(baitSubgraphNodes[["AC Gavin:Jan 2002"]], baitSubgraphNodes[[
"AC Gavin:Mar 2006"]])
[1] "YDL047W" "YNR016C" "YMR024W" "YER155C" "YIL142W" "YLR310C" "YFR019W"
[8] "YGL197W" "YFR040W" "YJL098W" "YKR028W" "YKL195W" "YNL101W" "YNL187W"
[15] "YOR267C"
> intersect(baitSubgraphNodes[["AC Gavin:Jan 2002"]], baitSubgraphNodes[[
"NJ Krogan:Mar 2006"]])
[1] "YDL047W" "YFR040W" "YJL098W" "YKR028W"
> intersect(baitSubgraphNodes[["AC Gavin:Mar 2006"]], baitSubgraphNodes[[
"NJ Krogan:Mar 2006"]])
[1] "YDL047W" "YJL098W" "YFR040W" "YKR028W"
```

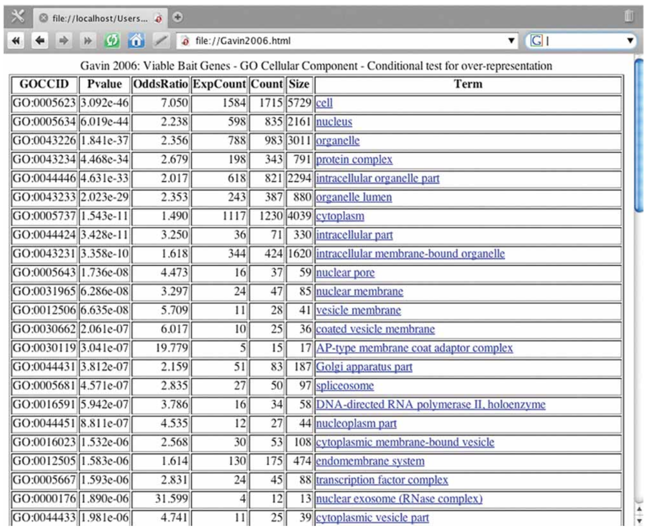
▲ CRITICAL STEP The intersection function only finds the common interactions that were reported but cannot determine meaningful statistics based upon the repetition of interactions. As the experiments might vary in both proteins sampled and experimental conditions, it is not recommended to make qualitative conclusions from this function. Venn diagram comparisons may also be misleading comparative tests.

▲ CRITICAL STEP The pairwise intersection of the prey found by YDL047W allows the inference of similar experimental conditions. The output above shows not only a large overlap between the two Gavin data sets, but also identical overlap when each Gavin data set is intersected with the data of Krogan *et al.*[11].

8| Assess sampling with regard to protein function as described by GO for individual experiments[40]. Use the function *ppiBuildParams4GO* from *ppiStats* to gather the appropriate data sets, ontology and other relevant parameters and then *ppiHGTest4GO* to execute the tests. **Figure 4** contains a screenshot of the html page generated to summarize the results from *ppiHGTest4GO*. For this and all remaining analyses, we will focus on the single high-throughput data set of Gavin *et al.*[10] (Gavin06):

```
> Gavin2006Graph <- apmsLSLocusNamesNodes[["AC Gavin:Mar 2006"]]
> Gavin2006Graph <- removeSelfLoops(Gavin2006Graph)
> viableProteins <- idViableProteins(Gavin2006Graph)
> VB <- viableProteins[["VB"]]
> VBP <- viableProteins[["VBP"]]
> VPO <- setdiff(viableProteins[["VP"]], VBP)
> yg <- names(as.list(org.Sc.sgdALIAS))
> param <- ppiBuildParams4GO(geneSet = VB, universe = yg, direction ="over",
annot = "YEAST", ontology = "CC", cond =
TRUE, pThresh = 0.001)
> hg <- ppiHGTest4GO(parameter = param,
filename = "Gavin2006",
append = FALSE, label = "Gavin 2006", typeGe-
neSet = "Viable Baits",
cs = 50)
```

▲ CRITICAL STEP These analyses assume that there are no edges pointing from a node into itself. The function *removeSelfLoops* can be used to remove any such edges before analysis.



Gavin 2006: Viable Bait Genes - GO Cellular Component - Conditional test for over-representation

GOCCID	Pvalue	OddsRatio	ExpCount	Count	Size	Term
GO:0005623	3.092e-46	7.050	1584	1715	5729	cell
GO:0005634	6.019e-44	2.238	598	835	2161	nucleus
GO:0043226	1.841e-37	2.356	788	983	3011	organelle
GO:0043234	4.468e-34	2.679	198	343	791	protein complex
GO:0044446	4.631e-33	2.017	618	821	2294	intracellular organelle part
GO:0043233	2.023e-29	2.353	243	387	880	organelle lumen
GO:0005737	1.543e-11	1.490	1117	1230	4039	cytoplasm
GO:0044424	3.428e-11	3.250	36	71	330	intracellular part
GO:0043231	3.358e-10	1.618	344	424	1620	intracellular membrane-bound organelle
GO:0005643	1.736e-08	4.473	16	37	59	nuclear pore
GO:0031965	6.286e-08	3.297	24	47	85	nuclear membrane
GO:0012506	6.635e-08	5.709	11	28	41	vesicle membrane
GO:0030662	2.061e-07	6.017	10	25	36	coated vesicle membrane
GO:0030119	3.041e-07	19.779	5	15	17	AP-type membrane coat adaptor complex
GO:0044431	3.812e-07	2.159	51	83	187	Golgi apparatus part
GO:0005681	4.571e-07	2.835	27	50	97	spliceosome
GO:0016591	5.942e-07	3.786	16	34	58	DNA-directed RNA polymerase II, holoenzyme
GO:0044451	8.811e-07	4.535	12	27	44	nucleoplasm part
GO:0016023	1.532e-06	2.568	30	53	108	cytoplasmic membrane-bound vesicle
GO:0012505	1.583e-06	1.614	130	175	474	endomembrane system
GO:0005667	1.593e-06	2.831	24	45	88	transcription factor complex
GO:0000176	1.890e-06	31.599	4	12	13	nuclear exosome (RNase complex)
GO:0044433	1.981e-06	4.741	11	25	39	cytoplasmic vesicle part

Figure 4 | A screenshot of the html page that provides a user-friendly interface for the output of the hypergeometric test for functional category overrepresentation. Overrepresentation (or enrichment) shows that the experimental assay has a preference for a group of genes/proteins. Using information about enriched categories, such as GO terms, allows us to use previous information in the estimation of new features. Conversely, categories that show underrepresentation by genes/proteins also give previous information for which we would not anticipate to estimate from the data.

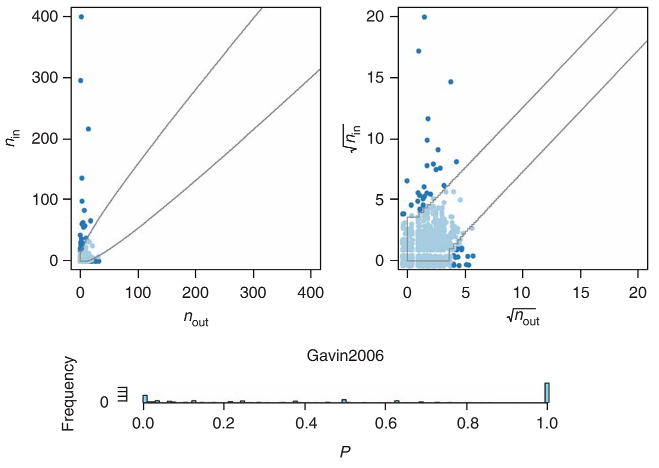


Figure 5 | Scatter plots corresponding to the in- and out-degree for each VBP of Gavin06. Each point in the left plot details the in-degree on the x axis and the out-degree on the y axis, whereas in the right plot, each degree is scaled by its square root. The points within the diagonal bands represent those proteins for which there is not enough evidence to reject the null hypothesis that the protein might be affected by some systematic bias, whereas the other points are rejected at the 0.01 P-value. The bottom plot describes the distribution of the P-values for each data point. Although the majority of proteins are not rejected, a substantial number of proteins are rejected. Those proteins that are rejected should not be analyzed using the standard statistical methods to model stochastic variability nor should they be used to estimate the underlying features within the data.

▲ CRITICAL STEP In this example, we specified the use of the Cellular Component (CC) ontology by setting *ontology="CC."* The biological process (BP) or molecular function (MF) ontologies from GO may also be used by specifying the parameter ontology in *ppiBuildParams4GO* accordingly. In addition, other categories may be tested, e.g., Kyoto Encyclopedia of Genes and Genomes or pFAM (cf. ?*ppiBuildParams4GO* in an R session).
▲ CRITICAL STEP The hypergeometric tests require specification of the *geneSet* of interest as well as the *universe* against which enrichment will be evaluated. In our example, we specify all known *Saccharomyces cerevisiae* encoding genes to comprise the universe. Other specifications could drastically change the results[40].
▲ CRITICAL STEP In addition to the output of *ppiHGTest4GO* in the R session (stored as *hg*), an html page containing results from the GO analysis will be created and stored in the working directory unless otherwise specified. The *filename* argument in *ppiHGTest4GO* is used to specify the file name, e.g., in this case, 'Gavin2006.html'.

Quality assessment: systematic error identification

9| Create scatter charts of in-degree (defined as the number of edges directed toward a node) versus out-degree (defined as the number of edges directed away from a node) for VBP nodes to visualize the subset of nodes prone to systematic error using the function *inOutScatterCharts* from the package *ppiStats*. Nodes with large disparities in in- versus out-degree are most likely subjected to some type of bias, as they do not exhibit variation consistent with technology-related stochastic measurement error rates. **Figure 5** contains the scatter charts generated by the code below:

```
> gl <- list(Gavin2006 = Gavin2006Graph)
> inOutScatterCharts(gl)
```

10| Identify specific VBP nodes prone to systematic error using the function *idSystematic* from the package *ppiStats*. The output of the *idSystematic* function is a character vector of those proteins/genes for which in-degree and out-degree is disproportionate based upon a Binomial test with $P = 1/2$ (see ref. 16). These proteins are removed from the graph for further analyses:

```
> systematicBaits <- idSystematic(Gavin2006Graph, VB, bpGraph = TRUE)
> unbiasedBaits <- setdiff(VB, systematicBaits)
> Gavin2006sub <- subGraph(c(unbiasedBaits, VPO), Gavin2006Graph)
> Gavin2006submat <- as(Gavin2006sub, "matrix")
> Gavin2006m <- Gavin2006submat[unbiasedBaits, c(unbiasedBaits,
VPO)]
```

Quality assessment: stochastic error estimation

11| Estimate stochastic error probabilities on the subgraph induced by the set of VBP nodes not prone to systematic error using the function *estimateCCMErrorRates* from the package *ppiStats*. This function estimates stochastic P_{TP} and P_{FP} probabilities by first calculating the method of moments solution manifold proposed by Chiang *et al.*[16] and then selecting a single pair of point estimates according to a GS. In this example, the object `ScISIC` is a set of manually curated protein complexes culled from MIPS, GO and *IntAct* that is used to create a set of GS positive complex comembership interactions[17]. The data set `ScISIC` is a part of the R package *ScISI*, which is installed as a dependency of *ppiStats*. **Figure 6** shows where the GS point estimates lie on the two-dimensional manifold of possibilities:

```
> data(ScISIC)
> errorRates <- estimateCCMErrorRates(m = Gavin2006m, GS = ScISIC)
> pTPestimate <- errorRates$globalpTP
> pFPestimate <- errorRates$globalpFP
```

© 2009 Nature Publishing Group http://www.nature.com/natureprotocols

Feature estimation: protein complex membership estimation

12| Estimate protein complex membership for AP-MS data using the function *findComplexes* from the package *apComplex*. The penalized likelihood approach used by the *apComplex* package seeks to locate tightly connected local features within the global set of detected interactions, as depicted in **Figure 7** (see ref. 19):

```
> Gavin2006complexEstimates <-
findComplexes(Gavin2006m, sensitivity =
pTPestimate, specificity = 1 - pFPestimate)
```

▲ **CRITICAL STEP** The R function *findComplexes* requires specification of P_{TP} and P_{FP}. In this example, we use the GS estimates generated in the Quality Assessment exercises.

▲ **CRITICAL STEP** The R function *findComplexes* mines for maximal cliques of tested interactions within bait–prey graphs. The maximal clique problem is known to be NP-complete, so there is no known efficient deterministic algorithm for this task. For larger data sets, it is highly recommended that the servers have sufficient processing capabilities and random access memory.

▲ **CRITICAL STEP** The feature estimation analyses described here (Steps 12–15) are specific to AP-MS and Y2H data. They make use of the descriptive and quality assessment techniques and results discussed previously. Other feature estimation approaches are possible and relevant to other bait–prey data types.

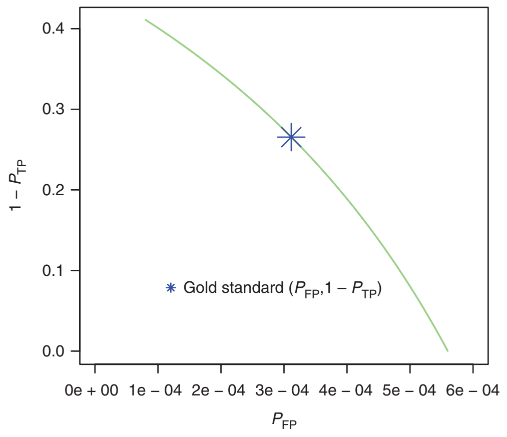


Figure 6 | Estimate of P_{TP} and P_{FP} for Gavin06 AP-MS data using the GS ScISIC reference data set and a manifold of possible solutions. In particular, the ScISIC protein complexes were used as a benchmark set of positive comembership interactions so that the $P_{FN} = 1$ P_{TP} rate could be estimated. With the P_{FN} estimate in place, an estimate for P_{FP} (blue star) is recovered from the solution manifold. Approximating P_{FN} and P_{FP} is essential for the estimation of features within the data.

13| Sort the protein complex estimates into different types of estimates using the function *sortComplexes* from the package *apComplex*. The *apComplex* algorithm yields three different types of complex estimates based on the number of bait proteins present in each complex estimate. MBME complexes contain at least two bait proteins and at least two edges and are most likely the most reliable complex estimates. Single-bait-multihit (prey) complexes contain only one bait protein, with the rest of them being prey-only proteins. Unreciprocated bait–bait complexes contain only two proteins, both baits and one unreciprocated edge connecting them:

```
> Gavin2006sorted <- sortComplexes(Gavin2006complexEstimates,Gavin2006m)
```

14| Plot complex estimates for visual inspection using the function *plotComplex* from the package *apComplex*. **Figure 8** depicts two examples of complex estimates from the Gavin06 analyses. In this example, *geneName* was set to *TRUE* so that nodes were labeled with gene names rather than locus identifiers:

```
> par(mfrow = c(1, 2))
> for (i in c(7, 31)) {
plotComplex(complexMembers = Gavin2006sorted$MBME[[i]],
g = Gavin2006sub, VBs = unbiasedBaits,
VPs = viablePrey, geneName = TRUE)
}
```

▲ **CRITICAL STEP** Descriptions of the bait/prey sampling scheme and functional representation of viable proteins as outlined in the exploratory analyses section will aid with understanding of the resultant protein complex estimates.

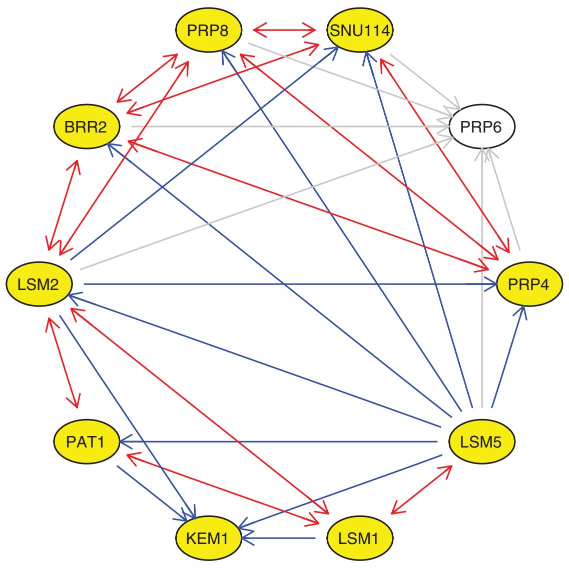


Figure 7 | This graph renders all the proteins found to interact with the preselected bait protein LSM5 (YER146W) within the Gavin06 data set. A simple visual inspection of this subgraph shows 42 observed interactions between these 10 proteins, which makes this a highly connected subgraph. Of the ten proteins that make up this subgraph, nine of the proteins (including LSM5) participated in the experiment as both viable baits and viable prey. Discounting self-interactions, there were 72 bilaterally tested relationships among the VBPs. Of these 72 tested interactions, Gavin and colleagues found 12 reciprocated and 12 unreciprocated interactions. In general, the feature of interest is to estimate whether or not these 10 proteins constitute a single functional unit or the amalgamation of several smaller units. Information, such as reciprocity, error rates and bias can all contribute to meaningful estimation.

© 2009 Nature Publishing Group http://www.nature.com/natureprotocols

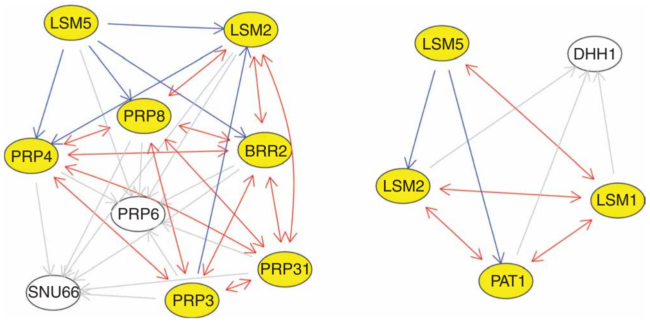


Figure 8 | Two examples of complex estimates using the *apComplex* algorithm. This particular example shows the assignment of Lsm5 to two of its characterized roles. The complex on the left depicts Lsm5 participating in a complex with other members of the U4/U6 × U5 trisnRNP complex[41]. The complex on the right depicts its activity with the Lsm1p-7p–Pat1p complex[42]. Estimating modular protein complexes through clustering mechanisms (as in **Fig. 7**) could not allow for Lsm2 and Lsm5 to participate in two unique complexes.

Feature estimation: physical interaction summarization

15| Summarize the overlay of Y2H physical interaction data on protein complex estimates derived from AP-MS data. In this example, we use the function *plotComplex* from *apComplex* to illustrate the set of detected physical interactions from the Ito *et al.*[3] and Uetz *et al.*[1] Y2H data sets for one of the protein complex estimates from the Gavin06 AP-MS data (**Fig. 9**):

```
> y2hLSLocusNamesNodes <- lapply(y2hLargeScale,function(x)
     translateID(x,''cygd''))
> y2hLargeScaleIU <- y2hLSLocusNamesNodes[c(1, 3)]
> y2hviableProteins <- lapply(y2hLargeScaleIU,
               FUN = idViableProteins)
> par(mfrow = c(1, 2))
> allComplexMembers <- Gavin2006sorted$MBME[["MBME31"]]
> for (i in 1:2) {

 thisy2hGraph <- y2hLargeScaleIU[[i]]
 subComplexMembers <-
     intersect(allComplexMembers,nodes(y2hLargeScaleIU[[i]]))

 plotComplex(complexMembers = subComplexMembers,
     g = y2hLargeScaleIU[[i]],
     VBs = y2hviableProteins[[i]]$VB,
     VPs = y2hviableProteins[[i]]$VP,
     geneName = TRUE)
}
```

● **TIMING**

Depending on computational resources and the size of the data sets under study, some of the functions may take many minutes, possibly approaching hours, to run. The time needed to execute all the analytic commands within this protocol can be computed by calling the *proc.time* function twice, immediately before and after the first and last commands. If we compute the difference as `time.elapsed`, we can see that the duration in seconds is

```
> time.elapsed
     user system elapsed
1500.136 31.703 1536.745
```

The user and system times are determined by the operating system in use. The elapsed time documents the total time used for all processes. The *proc.time* function is inherently dependent on the computational power and memory available. We used a 64-bit Opteron-based Linux system with dual 2.4 GHz AMD Opteron CPUs with 8 GB of physical random access memory. The system runs a 64-bit SuSE Linux and provides a 64-bit R environment.

The function *findComplexes* from the package *apComplex* may require computational intensity for larger data sets. All other functions in this protocol should yield results very efficiently.

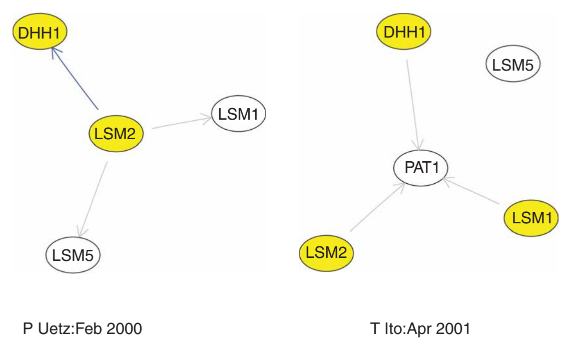


Figure 9 | An *apComplex* estimate with overlaid Y2H data from Ito *et al.*[3] and Uetz *et al.*[1] experiments. The AP-MS data for this estimate is depicted in the right panel of **Figure 8**. A node for PAT1 is not included in the P Uetz:Feb 2000 overlay graph (on the left), as it was found neither as viable bait nor as viable prey in this experiment. One possible explanation for the discrepancy between the physical layout using the Uetz data versus the Ito data (on the right) is that the interaction from LSM2 to both DHH1 and LSM1 is mediated through PAT1 in the Uetz data, but as PAT1 was neither a viable bait nor a viable prey, these interactions could not be observed.

© 2009 Nature Publishing Group http://www.nature.com/natureprotocols

? TROUBLESHOOTING

All functions in R packages are documented with 'help pages' describing function parameters and the details of values that the functions will yield. To access a help page for a specific function, e.g., *myFunction*, type `?myFunction` at the R prompt. For a more extended search of functions relevant to a specific topic, e.g., *myTopic*, type `help.search("myTopic")` at the R prompt. All Bioconductor packages also come with vignettes showing the use of the package. Questions not addressed in either the man pages or the vignette can be posted to the Bioconductor mailing list (http://www.bioconductor.org/docs/mailList.html).

ANTICIPATED RESULTS

Each section of the protocol is designed to systematically accomplish specific results in statistical analyses of bait–prey interaction graph data (**Fig. 2**). The preliminary setup exercises (Steps 1–4) should result in the necessary software functionality for completing statistical analyses and a tractable, analyzable data structure to facilitate the analyses. The exploratory analysis exercises (Steps 5–8) should yield descriptive statistics for the sampled bait and prey populations (**Fig. 3**), as well as an understanding of the functional representation of proteins in a data set (**Fig. 4**). The quality-assessment exercises (Steps 9–11) should assist in the diagnosis of systematically biased baits (see **Fig. 5**), as well as estimates of stochastic measurement error probabilities (**Fig. 6**). The feature estimation exercises (Steps 12–15) should yield estimates of biologically relevant characteristics, specific to the type of data being analyzed (**Figs. 7–9**).

The R packages (and therefore the R functions) provide an accessible framework in the estimation of biological features from molecular interaction data sets. Both computational and bench biologists can use the capabilities from these software tools not only to infer error and summary statistics, but also protein complex composition and integrity, which are essential components to a systems level study of biology. All reported results were generated using the following software package versions:

```
> sessionInfo()
R version 2.7.0 (2008-04-22)
x86_64-unknown-linux-gnu
locale:
LC_CTYPE=en_US.UTF-8;LC_NUMERIC=C;LC_TIME=en_US.UTF-8;LC_COLLATE=en_US.UTF-
8;LC_MONETARY=C;LC_MESSAGES=

attached base packages:
[1] splines grid tools stats graphics grDevices utils
[8] datasets methods base

other attached packages:
[1] ppiProtocolData_1.1.0 ppiStats_1.7.4 RColorBrewer_1.0-2
[4] lattice_0.17-6 ScISI_1.13.2 RpsiXML_1.0.0
[7] hypergraph_1.12.0 XML_1.95-2 ppiData_0.1.10
[10] GOstats_2.6.0 Category_2.6.0 genefilter_1.20.0
[13] survival_2.34-1 annotate_1.18.0 xtable_1.5-2
[16] GO.db_2.2.0 apComplex_2.7.3 org.Sc.sgd.db_2.2.0
[19] AnnotationDbi_1.2.0 RSQLite_0.6-8 DBI_0.2-4
[22] Biobase_2.0.1 Rgraphviz_1.18.0 RBGL_1.16.0
[25] graph_1.19.1 weaver_1.6.0 codetools_0.2-1
[28] digest_0.3.1

loaded via a namespace (and not attached):
[1] cluster_1.11.10 Rintact_1.3.3
```

Note: Supplementary information is available via the HTML version of this article.

ACKNOWLEDGMENTS We thank the reviewers for their insightful and useful critique of this manuscript. We also thank Wolfgang Huber, Robert Gentleman, Simon Anders and especially Sandra Orchard for their comments and suggestions as this manuscript evolved. T.C. has been supported by the Human Frontiers Science Program Grant (HFSP/LIP0442/2005) as well as the Ferris Fund from King's College, Cambridge University.

Published online at http://www.natureprotocols.com/
Reprints and permissions information is available online at http://npg.nature.com/reprintsandpermissions/

1. Uetz, P. *et al.* A comprehensive analysis of protein-protein interactions in *Saccharomyces cerevisiae*. *Nature* **403**, 623–627 (2000).
2. Newman, J.R., Wolf, E. & Kim, P.S. A computationally directed screen identifying interacting coiled coils from *Saccharomyces cerevisiae*. *Proc. Natl. Acad. Sci. USA* **97**, 13203–13208 (2000).
3. Ito, T. *et al.* A comprehensive two-hybrid analysis to explore the yeast protein interactome. *Proc. Natl. Acad. Sci. USA* **98**, 4569–4574 (2001).
4. Tong, A.H.-Y. *et al.* A combined experimental and computational strategy to define protein interaction networks for peptide recognition modules. *Science* **295**, 321–324 (2002).
5. Risseeuw, E.P. *et al.* Protein interaction analysis of SCF ubiquitin E3 ligase subunits from Arabidopsis. *Plant J.* **34**, 753–767 (2003).
6. Hazbun, T. *et al.* Assigning function to yeast proteins by integration of technologies. *Mol. Cell* **6**, 1353–1365 (2003).
7. Millson, S.H. *et al.* A two-hybrid screen of the yeast proteome for Hsp90 interactors uncovers a novel Hsp90 chaperone requirement in the activity of a

© 2009 Nature Publishing Group http://www.nature.com/natureprotocols

stress-activated mitogen-activated protein kinase Slt2p (Mpk1p). *Eukaryot. Cell* **4**, 849–860 (2005).
8. Ho, Y. *et al.* Systematic identification of protein complexes in *Saccharomyces cerevisiae* by mass spectrometry. *Nature* **415**, 180–183 (2002).
9. Gavin, A.-C. *et al.* Functional organization of the yeast proteome by systematic analysis of protein complexes. *Nature* **415**, 141–147 (2002).
10. Gavin, A.-C. *et al.* Proteome survey reveals modularity of the yeast cell machinery. *Nature* **440**, 631–636 (2006).
11. Krogan, N.J. *et al.* Global landscape of protein complexes in the yeast *Saccharomyces cerevisiae*. *Nature* **440**, 637–643 (2006).
12. Ohi, M.D. *et al.* Proteomics analysis reveals stable multiprotein complexes in both fission and budding yeasts containing Myb-related Cdc5p/Cef1p, novel pre-mRNA splicing factors, and snRNAs. *Mol. Cell. Biol.* **7**, 2011–2024 (2002).
13. Grandi, P. *et al.* 90S pre-ribosomes include the 35S pre-rRNA, the U3 snoRNP, and 40S subunit processing factors but predominantly lack 60S synthesis factors. *Mol. Cell* **10**, 105–115 (2002).
14. Zhao, R. *et al.* Navigating the chaperone network: an integrative map of physical and genetic interactions mediated by the Hsp90 chaperone. *Cell* **120**, 715–727 (2005).
15. Graumann, J. *et al.* Applicability of tandem affinity purification mudpit to pathway proteomics in yeast. *Mol. Cell. Proteomics* **3**, 226–237 (2004).
16. Chiang, T., Scholtens, D., Sarkar, D., Gentleman, R. & Huber, W. Coverage and error models of protein–protein interaction data by directed graph analysis. *Genome Biol.* **8**, R186 (2007).
17. Scholtens, D., Chiang, T., Huber, W. & Gentleman, R. Estimating node degree in bait–prey graphs. *Bioinformatics* **24**, 218–224 (2008).
18. Zhang, B., Park, B.H., Karpinets, T. & Samatova, N.F. From pull-down data to protein interaction networks and complexes with biological relevance. *Bioinformatics* **24**, 979–986 (2008).
19. Scholtens, D., Vidal, M. & Gentleman, R. Local dynamic modeling of global interactome networks. *Bioinformatics* **21**, 3548–3557 (2005).
20. R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria (2008).
21. Gentleman, R. *et al.* Bioconductor: open software development for computational biology and bioinformatics. *Genome Biol.* **5**, R80 (2004).
22. Shannon, P. *et al.* Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Res.* **13**, 2498–2504 (2003).
23. Cline, M. *et al.* Integration of biological networks and gene expression data using Cytoscape. *Nat. Protoc.* **2**, 2366–2382 (2007).
24. Milenkovic, T., Lai, J. & Przulj, N. GraphCrunch: a tool for large network analyses. *BMC Bioinformatics* **9**, 70 (2008).
25. Royer, L., Reimann, M., Andreopoulous, B. & Schroeder, M. Unraveling protein networks with power graph analysis. *PLoS Comput. Biol.* **4**, e1000108 (2008).
26. Tuttle, W.T. *Graph Theory*. Cambridge Mathematical Library, New York, (2001).
27. Stanley, R.P. *Enumerative Combinatorics I*. Cambridge University Press, Cambridge, (1997).
28. Carey, V.J., Gentry, J., Whalen, E. & Gentleman, R. Network structures and algorithms in bioconductor. *Bioinformatics* **21**, 135–136 (2005).
29. Huber, W., Carey, V.J., Long, L., Falcon, S. & Gentleman, R. Graphs in molecular biology. *BMC Bioinformatics* **8** (Suppl. 6): S8 (2007).
30. Chiang, T., *et al.* Rintact: enabling computational analysis of molecular interaction data from the IntAct repository. *Bioinformatics* **24**, 1100–1101 (2008).
31. The Gene Ontology Consortium. Gene ontology: tool for the unification of biology. *Nat. Genet.* **25**, 25–29 (2000).
32. Kanehisa, M. & Goto, S. KEGG: Kyoto Encyclopedia of Genes and Genomes. *Nucleic Acids Res.* **28**, 27–30 (2000).
33. Kerrien, S. *et al.* Broadening the horizon—level 2.5 of the HUPO-PSI format for molecular interactions. *BMC Biol.* **5**, 44 (2007).
34. Stark, C. *et al.* BioGRID: a general repository for interaction datasets. *Nucleic Acids Res.* **34**, D535–D539 (2006).
35. Salwinski, L. *et al.* The database of interacting proteins. *Nucleic Acids Res.* **32**, D449–D451 (2004).
36. Mishra, G. *et al.* Human protein reference database—2006 update. *Nucleic Acids Res.* **34**, D411–D414 (2006).
37. Kerrien, S. *et al.* IntAct—open source resource for molecular interaction data. *Nucleic Acids Res.* **35**, D561–D565 (2007).
38. Chatr-aryamontri, A. *et al.* MINT: the Molecular INTeraction database. *Nucleic Acids Res.* **35**, D572–D574 (2007).
39. Orchard, S. *et al.* The minimum information required for reporting a molecular interaction experiment (MIMIx). *Nat. Biotechnol.* **25**, 894–898 (2007).
40. Falcon, S. & Gentleman, R. Using GOstats to test gene lists for GO term association. *Bioinformatics* **2**, 257–258 (2006).
41. Stevens, S. & Abelson, J. Purification of the yeast U4/U6.U5 small nuclear ribonucleoprotein particle and identification of its proteins. *Proc. Natl. Acad. Sci. USA* **96**, 7226–7231 (1999).
42. Chowdhury, A., Mukhopadhyay, J. & Tharun, S. The decapping activator Lsm1p-7p-Pat1p complex has the intrinsic ability to distinguish between oligoadenylated and polyadenylated RNAs. *RNA* **13**, 998–1016 (2007).

© 2009 Nature Publishing Group http://www.nature.com/natureprotocols