

Building Functions

Eugen Buehler

November 14th, 2018

R Functions are objects

R is a functional programming language. This means that functions are “objects”, just like data frames, vectors, and other things that are assigned to variables and passed to other functions.

A rose by any other name

The name of a function is actually the name of a variable that contains the function, in the same way that the

```
log
```

```
## function (x, base = exp(1)) .Primitive("log")
```

This means that we can create a copy of a function by assigning its value to a new variable.

```
myLogFunction <- log  
myLogFunction
```

```
## function (x, base = exp(1)) .Primitive("log")
```

Functions are a kind of data and have a class

```
myNumber <- 7  
class(myNumber)
```

```
## [1] "numeric"
```

```
class(log)
```

```
## [1] "function"
```

Creating a function

We can create a new function using the word “function” followed by the functions arguments and one or more R statements.

```
myDumbFunction <- function() 42  
myDumbFunction()
```

```
## [1] 42
```

This is a function with **no** arguments. Usually functions have arguments, which we will see next. Here, `myDumbFunction` gives the same answer whenever it's called

Creating a multi-statement function

If there is more than one statement in a function, they should be enclosed in curly brackets:

```
doubleIt <- function(x) {  
  myResult <- x * 2  
  myResult # or, explicitly, return(myResult)  
}  
doubleIt(5)
```

```
## [1] 10
```

The last statement within the curly brackets will be the value returned by the function.

x is the function argument, in that it is a placeholder we can replace with an actual value when calling the function

Functions live in their own little world

Inside a function, variables that existed in your environment can be used and even changed. However, any changes made, including changing data stored in variables and creating new variables, happens solely within the function. Your environment stays the same.

```
exists("myResult")
```

```
## [1] FALSE
```

```
myResult <- 1000
```

```
doubleItOutput <- doubleIt(2)
```

```
myResult
```

```
## [1] 1000
```

Example Data Set

The data set used in today's lecture comes from an siRNA screen that we published a few years ago. The screen looked for genes that influence parkin translocation.

High-content genome-wide RNAi screens identify regulators of parkin upstream of mitophagy. Hasson SA, Kane LA, Yamano K, Huang CH, Sliter DA, Buehler E, Wang C, Heman-Ackah SM, Hessa T, Guha R, Martin SE, Youle RJ. Nature. 2013.

The data set will be available for download from the lectures portion of the class web page, also [here](#).

Preview the Data

Vendor Supplied		Percent Cells Plasmid Translocation Negative (PPT)				Cell Count (Viability)				Mitochondrial Signal				HUGO Gene Entrez		Row	Column	Ambion			
Sample	Plate	Median Negative Control on Plate	Median Positive Control on Plate	PPT Sample as Percentage of Negative	PPT MAD Z-Score	Median Log MAD Z-Score	Number of siRNAs having the same seed	Number of siRNAs having the same seed	Cell Count	Median Negative Control	Median Positive Control	Sample Cell Count	Median Negative Control	Median Positive Control	Sample MAD Z-Score	GeneID	DESCRIPTION	Number	Number	siRNA ID	
4	GENE001	8.913514	38.61376	96.755081	-2.41729712	-4.82222725	126	1286	1178	1030.5	1106.517777	8.942457	10.462261	22.38205	0.04662108	ILMAN1	QDA01771	1	4	48128	
5	ILMAN1	7.91236	33.346945	96.625828	-2.40278854	-4.77841378	152	1203	1310.5	1050.5	1240.9137	16.791355	28.081704	49.01673	0.88178185	PCOP3	QDA01996	1	21	12386	
6	PCOP3	7.91236	33.346945	96.625828	-2.40278854	-4.77841378	152	1203	1310.5	1050.5	1240.9137	16.791355	28.081704	49.01673	0.88178185	PCOP3	QDA01996	1	21	12386	
7	PFN2	7.96128	31.36618	97.067871	-2.40355876	-4.78108186	152	1172	1260.5	978.5	0.39941612	11.299915	29.03928	50.076025	-1.70001232	PFN2	QDA01288	14	19	10379	
8	KIAA1191	11.20288	43.7561875	96.4889565	-2.50137581	-5.05442945	128	1060	1340.5	1106.5	-0.27412438	11.32075	15.536245	27.72216	-0.42808038	KIAA1191	QDA01843	16	11	122909	
9	CIBOR5	8.87887	34.33545	96.121174	-2.45939098	-4.84680947	18	1272	1265.5	985	1.16624827	12.893862	12.099925	22.029965	0.94528995	CIBOR5	QDA01863	1	12	122581	
10	LOC50689	9.59596	36.724075	91.666073	-2.61298885	-4.45620077	28	1093	1307.5	1061.5	0.56747784	14.181152	10.374847	18.885315	2.09858154	LOC50689	QDA01761	2	1	123748	
11	ILMAN1	7.04537	26.494005	95.491585	-2.32679673	-4.39629773	12	1107	1209	1019.5	0.72890788	5.128205	8.734746	20.23281	-0.93404899	ILMAN1	QDA01867	1	4	48219	
12	FAM73A	10.77458	39.890226	96.760781	-2.61172128	-5.31804062	4	854	1022	1059	-0.13116479	23.653397	26.59421	44.646865	0.24364276	FAM73A	QDA02088	1	6	49518	
13	C17orf77	11.32515	40.214815	96.524555	-2.6181867	-5.24347931	75	1220	1315.5	1025	0.33320282	17.049181	23.9350045	41.648832	-0.67919657	C17orf77	QDA01213	10	1	144897	
14	SAMD1	12.39694	43.454125	96.062091	-2.79136180	-5.35020081	18	1136	1308	1005	0.46105972	9.154269	15.936625	32.7526	-1.06135717	SAMD1	QDA01816	1	14	124841	
15	LOC42441	11.46216	39.890226	96.760781	-2.61172128	-5.31804062	4	854	1022	1059	-0.13116479	23.653397	26.59421	44.646865	0.24364276	LOC42441	QDA01208	1	6	49518	
16	WRD24	11.19552	38.803265	96.993275	-2.85252262	-4.00417108	62	869	1314.5	1041.5	0.28469718	10.126582	27.585454	46.768615	-1.78558099	WRD24	QDA01274	1	18	13663	
17	C4orf7	9.948416	34.335125	96.121174	-2.45939098	-4.84680947	18	1306	1265.5	985	1.16624827	12.893862	12.099925	22.029965	0.94528995	C4orf7	QDA01863	6	6	49693	
18	LOC11855	10.96114	37.102918	96.613795	-2.52379411	-4.45620077	50	1231	1355	1037.5	0.47263007	27.819627	27.631765	42.340316	0.67513289	LOC11855	QDA01384	1	18	14517	
19	MGC16121	12.146893	41.133693	96.967804	-2.53730547	-5.25377054	62	1012	1213.5	1025.5	0.71828563	26.482214	25.362852	40.9147165	0.84743085	MGC16121	QDA01269	1	3	33958	
20	DFB129	12.52262	41.674112	96.520615	-2.60482993	-5.04430101	32	1091	1230	1080	0.65846216	8.890925	10.76316	22.96937	0.05415426	DFB129	QDA01337	17	1	142567	
21	ATP13	11.61611	37.937375	96.455925	-2.22617923	-4.39880862	31	1154	1246	1019	0.35238956	26.981189	29.042115	46.512616	0.27407092	ATP13	QDA01240	1	15	14816	
22	CHP	10.23808	34.179075	97.58857	-2.25677481	-4.38691417	107	1217	1386	804	0.87093933	16.93442	13.866883	23.879101	1.37446036	CHP	QDA01599	16	1	122326	
23	FLAD125	12.38245	40.214815	96.524555	-2.6181867	-5.24347931	75	1207	1315.5	1025	0.23619941	24.109362	23.9350045	41.648832	-0.39946978	FLAD125	QDA01253	16	1	145036	
24	VPK1C	4.535559	14.603112	87.703015	-3.09251211	-2.18427246	3	1224	1371	1088	-0.007055229	21.977123	21.4113465	37.0343345	0.77891145	VPK1C	QDA01214	12	1	129543	
25	AMOTL2	10.32615	33.346945	96.652828	-2.40355876	-4.78108186	152	1223	1310.5	1050.5	0.39061348	12.755519	28.081704	49.01673	-1.44708305	AMOTL2	QDA01996	7	17	128109	
26	TMC1	12.21492	39.331558	97.046655	-2.61649716	-5.12059607	15	1198	1275.5	1057	0.58183994	8.597663	13.744405	31.707255	-0.78138707	TMC1	QDA01810	1	12	142168	
27	SOD4B	12.44284	33.603315	97.079395	-2.51168819	-5.36119778	22	750	972.5	823	1.20215703	1.6	15.161845	24.521595	-2.88626506	SOD4B	QDA01991	14	14	133477	
28	LOC28523	10.76721	34.335125	96.121174	-2.45939098	-4.84680947	18	1152	1265.5	985	0.22157684	7.211121	12.059825	22.029965	0.88878205	LOC28523	QDA01996	7	17	128109	
29	FAM74A	10.75794	34.18713	94.94933	-2.13174636	-4.20941356	3	1221	1311	1005	0.00346737	13.79868	15.983825	31.66997	0.34400032	FAM74A	QDA01246	15	1	146022	
30	NCL	14.55729	32.07602	95.879448	-2.18832958	-4.81918084	1	1170	1255	1016	0.38117033	6.666667	10.4767765	21.464887	-0.73962486	NCL	QDA01863	16	4	449924	
31	RASL11	10.23117	31.579678	97.040433	-2.33984202	-4.38691417	107	1067	1203	818	0.1764255	2.811621	10.433005	21.094802	-1.2647193	RASL11	QDA01240	5	13	151009	
32	LOC70187	10.32188	39.8915025	96.71463	-2.6695407	-4.44669732	29	1117	1238	1067	0.08616468	24.082363	28.46302	46.045658	0.07532204	LOC70187	QDA01327	9	1	146037	
33	PRK8	12.43239	37.937375	96.455925	-2.22617923	-4.39880862	31	1154	1246	1019	0.35238956	26.981189	29.042115	46.512616	0.27407092	PRK8	QDA01240	1	15	14816	
34	SEMA6D	9.54974	36.130773	81.4146085	-2.37897595	-4.94841356	43	1281	1373.5	1178	0.613711143	8.430913	14.54753	25.32889	0.60233568	SEMA6D	QDA02046	13	15	138853	
35	FAM70B	10.29808	31.20667	95.909923	-2.30098662	-4.65827968	37	1299	1328	1013.5	0.71088861	4.937723	14.924265	21.11532	-1.74546812	FAM70B	QDA01856	2	18	123655	
36	SPY4S	12.02088	34.692378	97.121471	-2.47425254	-5.12349954	40	1022	1259	1015	0.60149451	11.813643	15.780395	22.121678	0.75303712	SPY4S	QDA01758	14	2	123816	
37	LOC70176	13.01421	38.8915025	96.71463	-2.6695407	-4.44669732	29	1117	1238	1067	0.08616468	24.082363	28.46302	46.045658	0.07532204	LOC70176	QDA01327	7	17	126621	
38	LOC729747	10.34078	38.8915025	96.455925	-2.22617923	-4.39880862	31	1154	1246	1019	0.35238956	26.981189	29.042115	46.512616	0.27407092	LOC729747	QDA01743	1	1	146011	
39	ATL1C	11.44312	40.16214	96.4889565	-2.45939098	-4.84680947	18	1089	1235.5	1071	0.70700914	21.303949	25.790515	42.022265	0.9494372	ATL1C	QDA01252	1	22	15848	
40	LCR	5.467801	16.30773	81.4146085	-2.33984202	-4.38691417	107	1540	1387.5	1178	0.2758456	9.300699	14.54753	25.32889	0.72759303	LCR	QDA02046	4	20	139044	
41	IL33	11.78189	41.133693	96.967804	-2.53730547	-5.25377054	62	1131	1233.5	1025.5	0.23619941	24.109362	23.9350045	41.648832	-0.39946978	IL33	QDA01269	9	2	146251	
42	AGR2	13.86963	41.055187	96.555149	-2.14602159	-5.68990722	18	1097	1312	1062.5	0.48737956	11.668186	16.447305	30.078175	-0.40427157	AGR2	QDA01856	16	6	49294	
43	TAC1	10.55555	31.20667	95.909923	-2.30098662	-4.65827968	37	6	1031	1238	1013.5	0.51557684	5.237633	14.924265	21.11532	-1.44268732	TAC1	QDA01856	1	12	139091
44	LOC72954	9.90909	29.06698	97.012075	-2.46628686	-4.23824287	28	851	1028.5	802	0.9090909	11.05889	19.45772	34.010118	-1.006308	LOC72954	QDA01262	1	15	152953	
45	LOC72954	9.90909	29.06698	97.012075	-2.46628686	-4.23824287	28	851	1028.5	802	0.9090909	11.05889	19.45772	34.010118	-1.006308	LOC72954	QDA01262	1	15	152953	
46	INP151	13.75789	40.214815	96.524555	-2.6181867	-5.24347931	75	1063	1315.5	1025.5	0.39097631	18.720022	23.9350045	41.648832	-0.40751865	INP151	QDA01213	6	15	14493	
47	LX1L	11.93909	39.728825	96.494648	-2.33984202	-4.38691417	107	975	1253	1029	1.31408827	14.461538	23.895603	41.75009	0.86056038	LX1L	QDA01259	15	13	144250	
48	C16orf5	11.196078	41.252145	96.413765	-2.13579121	-4.63095452	4	1223	1316	1089	0.352272194	14.472609	27.268265	25.963445	0.03807762	C16orf5	QDA01377	1	12	123748	
49	FLJ36032	14.25012	41.674112	96.520615	-2.60482993	-5.04430101	32	1091	1230	1080	-0.007199016	12.576316	22.96937	0.05415426	FLJ36032	QDA01377	1	6	494880		
50	KCN16	13.01786	38.172195	97.036425	-2.44353771	-5.13429742	37	84	1028.5	835.5	0.24279418	13.501144	25.744465	37.937898	-0.88828119	KCN16	QDA02242	12	1	17774	
51	LOC11855	10.96114	37.102918	96.613795	-2.52379411	-4.45620077	50	1231	1355	1037.5	0.47263007	27.819627	27.631765	42.340316	0.67513289	LOC11855	QDA01384	1	18	14517	
52	DZNF8	13.23741	38.63046	95.903535	-2.26642792	-4.65706183	36	1320	1313	961	2.27893585	15.515512	8.471308	19.361636	2.06133666	DZNF8	QDA01716	1	8	432958	
53	PGB3	10.11292	35.24025	96.21645	-2.3668037	-4.6448956	28	1011	1173.5	951	0.20938567	6.03363	9.811335	20.854155	-0.82821776	PGB3	QDA018				

Import the data

```
library(readxl)
ambion <- read_excel("lecture_functions_data/nature.parkin.gw.xlsx", skip = 3)
str(ambion)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    65196 obs. of  25 variables:
## $ Vendor Supplied Gene Symbol                : chr
## $ Sample                                      : num
## $ Median Negative Control on Plate           : num
## $ Median Positive Control on Plate           : num
## $ PPT Sample as Percentage of Negative Control : num
## $ PPT MAD Z-Score                            : num
## $ PPT MAD Log MAD Z-Score                   : num
## $ Median PPT Log Mad Z-Score for all siRNAs having the same seed sequence: num
## $ Number of siRNAs having the same seed sequence : num
## $ Cell Count, Sample                         : num
## $ Median Negative Control Cell Count on Plate : num
## $ Median Positive Control Cell Count on Plate : num
## $ Sample Cell Count, MAD Z-Score Normalized to Negative Contol : num
## $ Sample__1                                  : num
## $ Median Negative Control Mitophagy on Plate : num
```

Check for missing data

```
options(dplyr.width = Inf) # show all cols
ambion %>% summarize_all(function(x) sum(is.na(x)))

## # A tibble: 1 x 25
##   `Vendor Supplied Gene Symbol` Sample `Median Negative Control on Plate`
##           <int> <int>                               <int>
## 1           441   441                               441
##   `Median Positive Control on Plate`
##           <int>
## 1           441
##   `PPT Sample as Percentage of Negative Control` `PPT MAD Z-Score`
##           <int>                               <int>
## 1           441                               441
##   `PPT MAD Log MAD Z-Score`
##           <int>
## 1           441
##   `Median PPT Log Mad Z-Score for all siRNAs having the same seed sequence`
##                                           <int>
## 1                                           441
##   `Number of siRNAs having the same seed sequence` `Cell Count, Sample` 11/32
```

Investigate Missing Data

```
#ambion[is.na(ambion[,1]),][1,]
ambion %>% filter(is.na(.[,1])) %>% slice(1)

## # A tibble: 1 x 25
##   `Vendor Supplied Gene Symbol` Sample `Median Negative Control on Plate`
##   <chr>                        <dbl>                                <dbl>
## 1 <NA>                          NA                                    NA
##   `Median Positive Control on Plate`
##   <dbl>
## 1 NA
##   `PPT Sample as Percentage of Negative Control` `PPT MAD Z-Score`
##   <dbl>                                <dbl>
## 1 NA                                    NA
##   `PPT MAD Log MAD Z-Score`
##   <dbl>
## 1 NA
##   `Median PPT Log Mad Z-Score for all siRNAs having the same seed sequence`
##   <dbl>
## 1 NA
##   `Number of siRNAs having the same seed sequence` `Cell Count, Sample` 12/32
```

Eliminate Missing Data

```
# ambion <- ambion[! is.na(ambion[,2]), ]
ambion <- ambion %>% filter(!is.na(Sample))
#apply(ambion, 2, function(x) sum(is.na(x)))
ambion %>% summarize_all(function(x) sum(is.na(x)))
```

```
## # A tibble: 1 x 25
```

```
##   `Vendor Supplied Gene Symbol` Sample `Median Negative Control on Plate`
##           <int> <int>                                <int>
## 1             0     0                                    0
```

```
##   `Median Positive Control on Plate`
##           <int>
## 1             0
```

```
##   `PPT Sample as Percentage of Negative Control` `PPT MAD Z-Score`
##           <int>                                <int>
## 1             0                                    0
```

```
##   `PPT MAD Log MAD Z-Score`
##           <int>
## 1             0
```

```
##   `Median PPT Log Mad Z-Score for all siRNAs having the same seed sequence`
##                                                                 <int>
```

Simplify the Data

Often it will be helpful to create a new data frame with only the data we wish to analyze.

```
#ambion.simple <- ambion[,c(19,25,1,21,7,13,17)]  
ambion.simple <- select(ambion, 19,25,1,21,7,13,17)  
ambion.simple[1,]
```

```
## # A tibble: 1 x 7  
##   `Entrez GeneID` `Ambion siRNA ID` `Vendor Supplied Gene Symbol`  
##           <dbl> <chr>           <chr>  
## 1           3998 s8218           LMAN1  
##   DESCRIPTION           `PPT MAD Log MAD Z-Score`  
##   <chr>                   <dbl>  
## 1 lectin, mannose-binding, 1           -4.82  
##   `Sample Cell Count, MAD Z-Score Normalized to Negative Control`  
##                                     <dbl>  
## 1                                     1.11  
##   `Sample Mitophagy, MAD Z-Score Normalized to Negative Control`  
##                                     <dbl>  
## 1                                     0.0466
```

Simplify our Column Names

```
library(knitr, quietly = TRUE)
colnames(ambion.simple) <- c("GeneID", "siRNA", "Symbol", "Description",
                             "PPT", "Cells", "Mitophagy")
kable(head(ambion.simple, n=4), format = "markdown")
```

GeneID	siRNA	Symbol	Description	PPT	Cells	Mitophagy
3998	s8218	LMAN1	lectin, mannose-binding, 1	-4.822230	1.1085178	0.0466291
51586	s28366	MED15	mediator complex subunit 15	-4.739415	0.2405872	-0.8887362
5217	s10379	PFN2	profilin 2	-4.539309	0.3994161	-1.7000123
57179	s226909	KIAA1191	KIAA1191	-4.516443	-1.0274124	-0.4280631

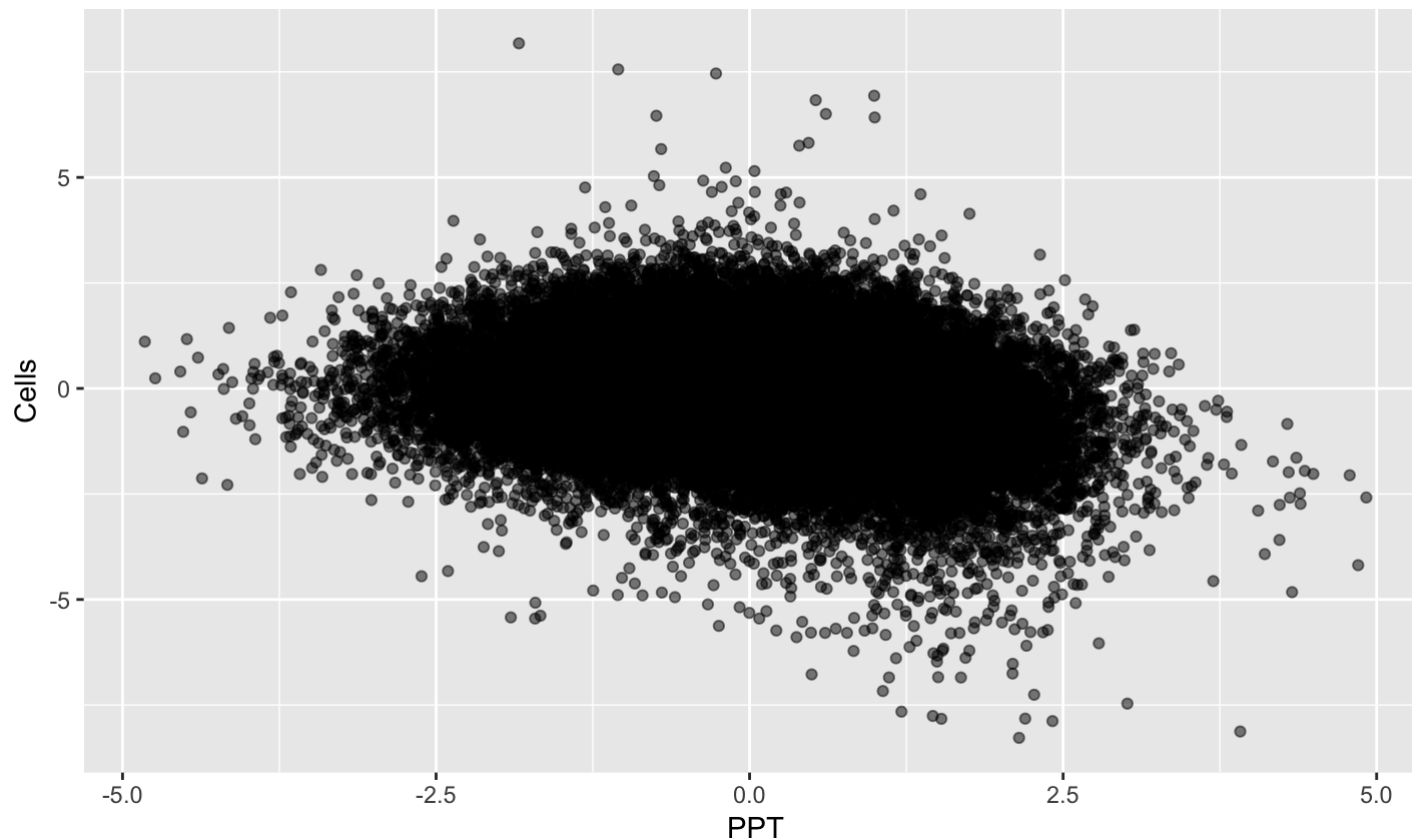
Simplify our Column Names

```
library(knitr, quietly = TRUE)
ambion.simple <- ambion.simple %>% set_names(c("GeneID", "siRNA", "Symbol", "Descriptio
      "PPT", "Cells", "Mitophagy"))
head(ambion.simple, n = 4) %>% kable(format='markdown')
```

GeneID	siRNA	Symbol	Description	PPT	Cells	Mitophagy
3998	s8218	LMAN1	lectin, mannose-binding, 1	-4.822230	1.1085178	0.0466291
51586	s28366	MED15	mediator complex subunit 15	-4.739415	0.2405872	-0.8887362
5217	s10379	PFN2	profilin 2	-4.539309	0.3994161	-1.7000123
57179	s226909	KIAA1191	KIAA1191	-4.516443	-1.0274124	-0.4280631

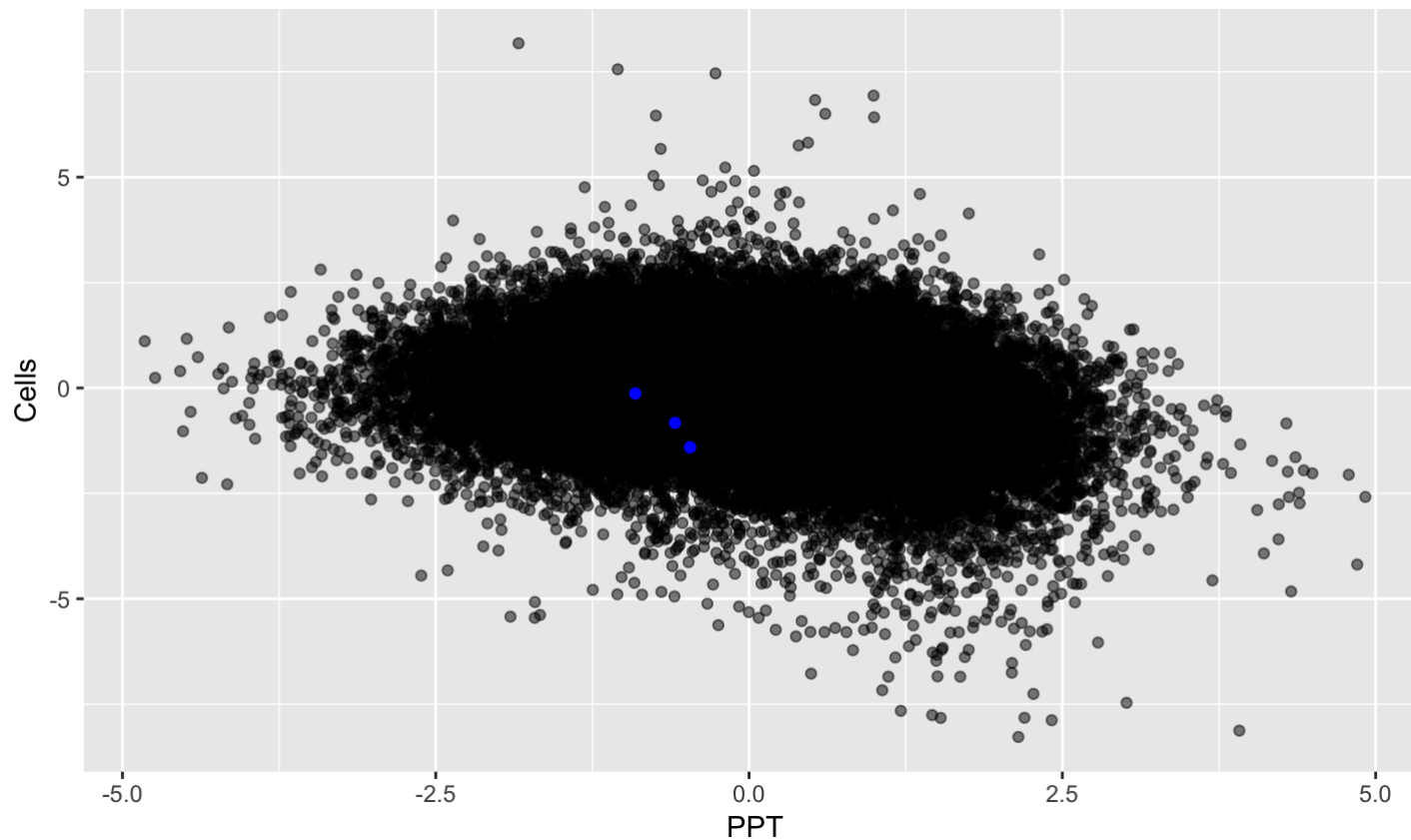
Evaluate how our variables interact

```
library(ggplot2, quietly = TRUE)  
ggplot(ambion.simple, aes(x=PPT, y=Cells)) + geom_point(alpha=0.5)
```



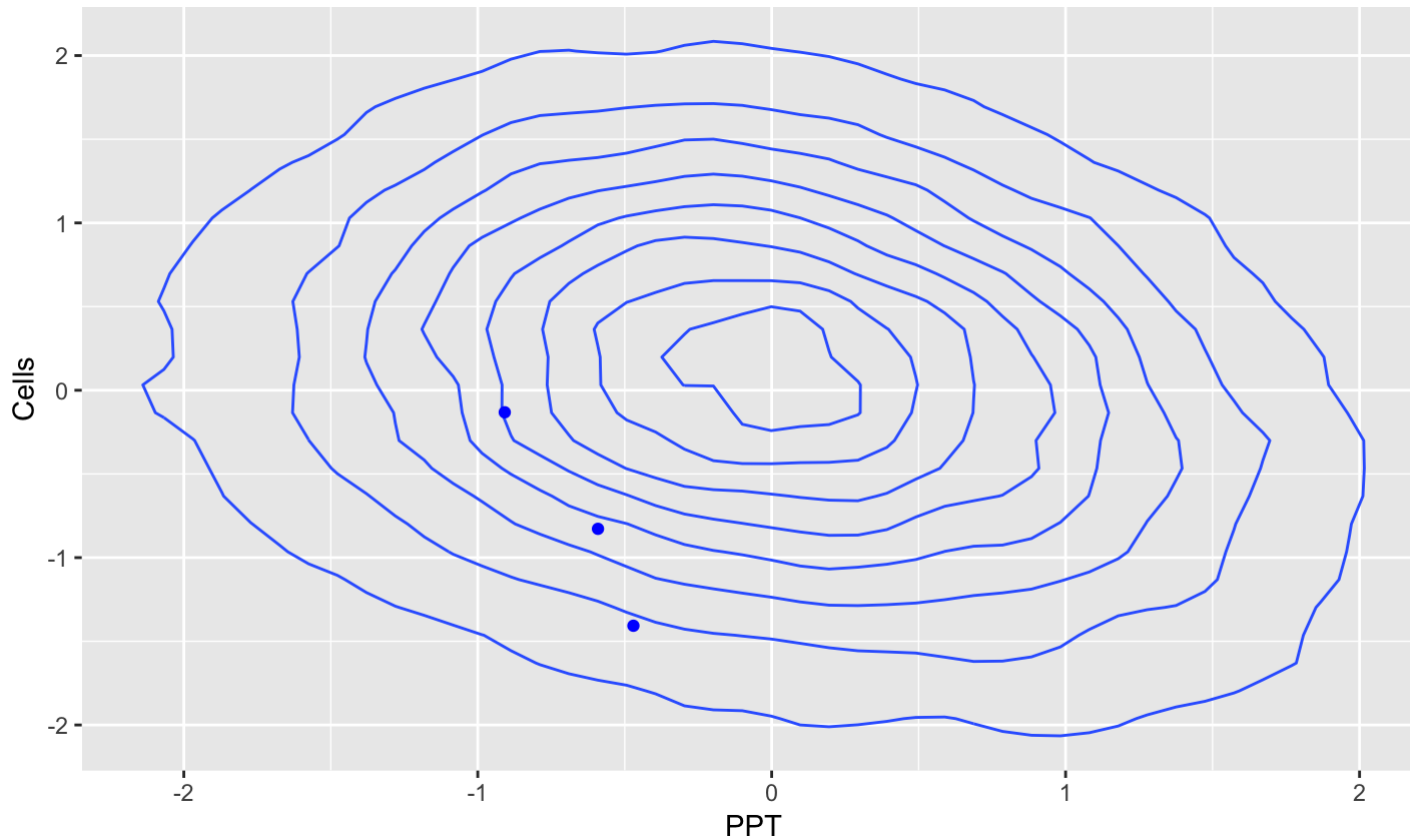
Evaluate how our variables interact

```
ggplot(ambion.simple, aes(x=PPT, y=Cells)) + geom_point(alpha=0.5) +  
  geom_point(data = ambion.simple %>% filter(Symbol=='PARK2'),  
            #ambion.simple[ambion.simple$Symbol == "PARK2", ],  
            color="blue")
```



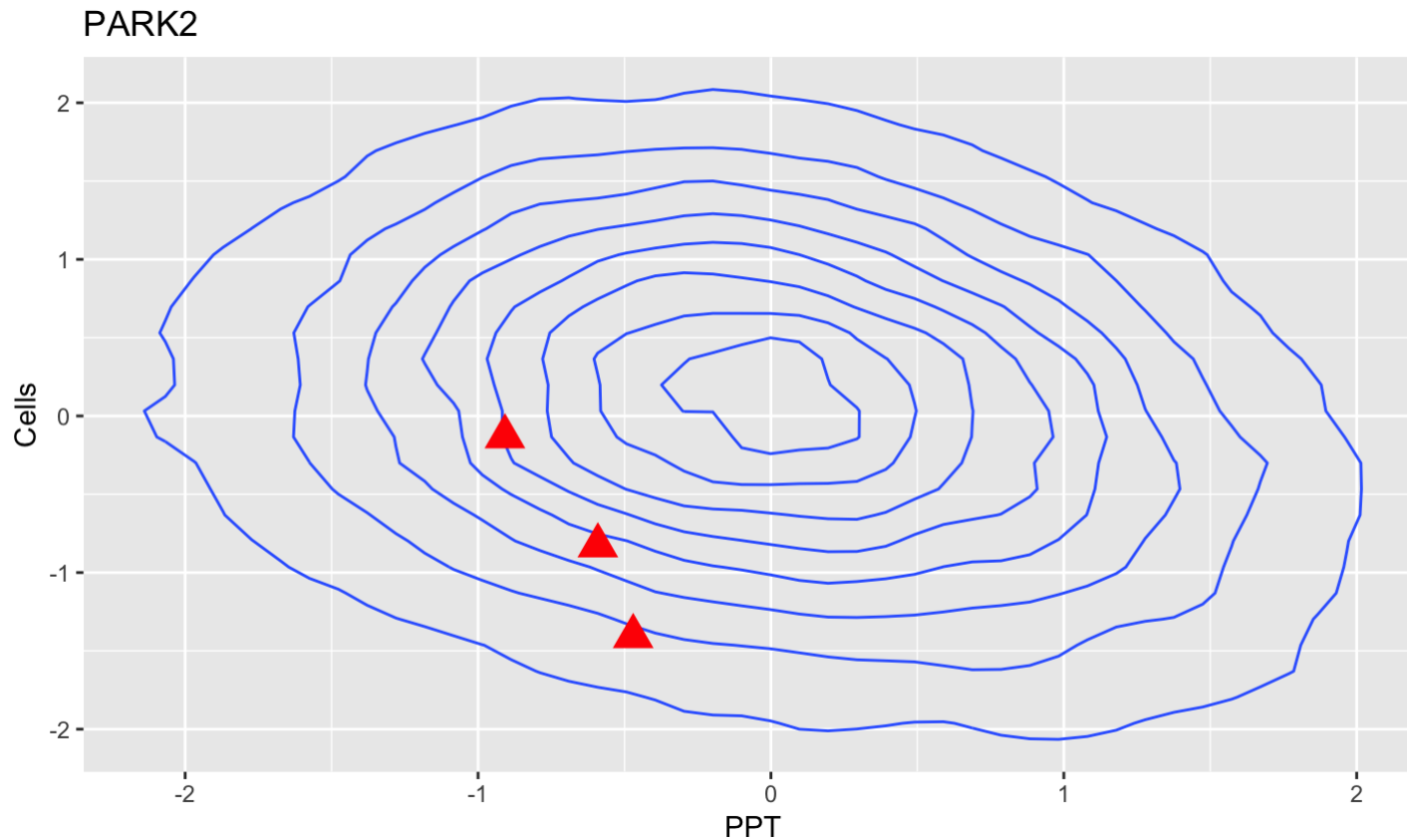
Refine the plot

```
ggplot(ambion.simple, aes(x=PPT, y=Cells)) + geom_density2d() +  
  geom_point(data = ambion.simple %>% filter(Symbol=='PARK2'),  
            color="blue")
```



Further refine the plot

```
ggplot(ambion.simple, aes(x=PPT, y=Cells)) + geom_density2d() +  
  geom_point(data = ambion.simple %>% filter(Symbol=="PARK2"),  
            color="red", shape=17, size =5) +  
  ggtitle("PARK2")
```



Adding gene description

```
description <- ambion.simple$Description[ambion.simple$Symbol == "PARK2"][1]  
description
```

```
## [1] "Parkinson disease (autosomal recessive, juvenile) 2, parkin"
```

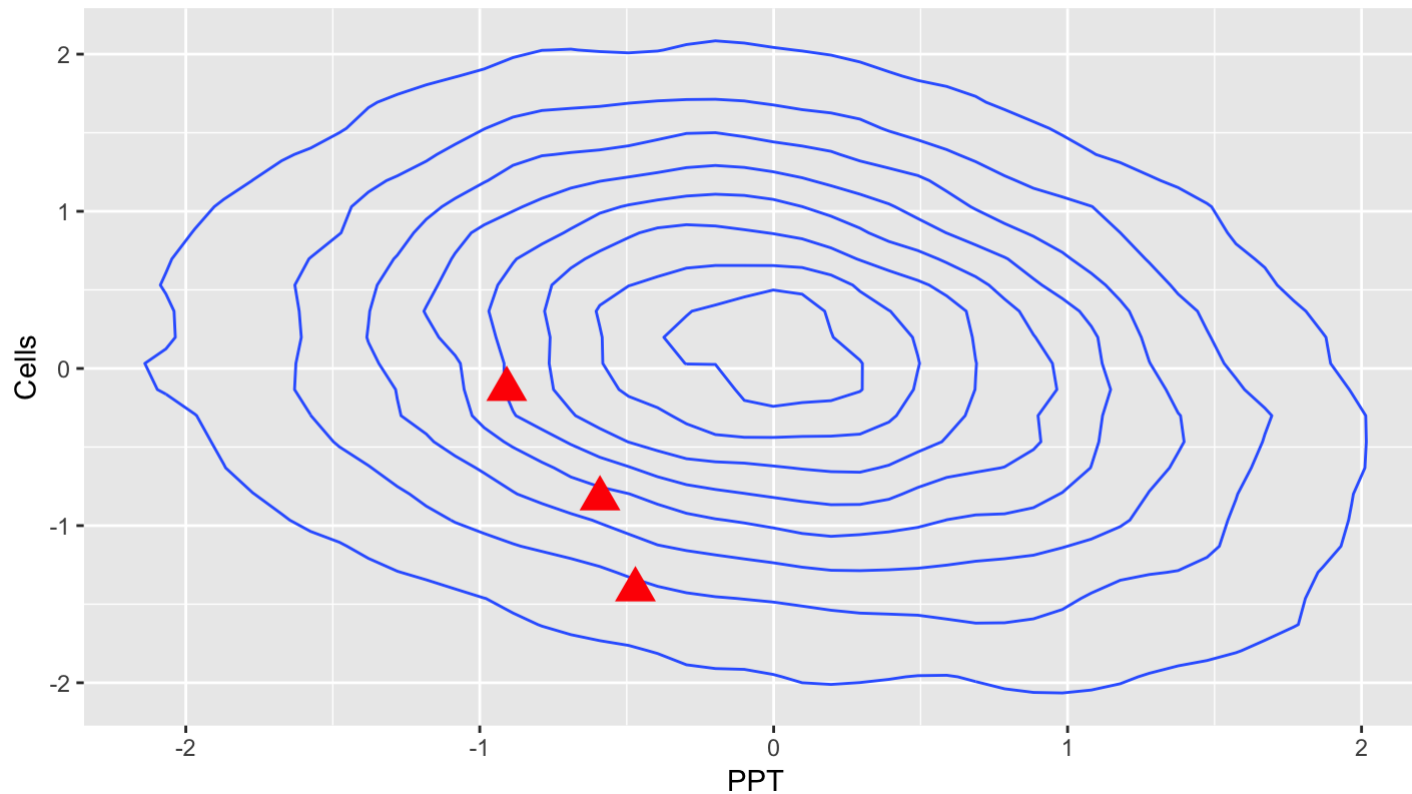
```
myTitle <- paste("PARK2",description,sep=": ")  
myTitle
```

```
## [1] "PARK2: Parkinson disease (autosomal recessive, juvenile) 2, parkin"
```

Final version of plot

```
ggplot(ambion.simple, aes(x=PPT, y=Cells)) + geom_density2d() +  
  geom_point(data = ambion.simple %>% filter(Symbol == "PARK2"),  
            color="red", shape=17, size =5) +  
  ggtitle(myTitle)
```

PARK2: Parkinson disease (autosomal recessive, juvenile) 2, parkin



Making the refined plot into a function

Now that we have our custom plot looking right, we would like to be able to do the same for other genes but without so much typing. First, make a new R Script in RStudio:

Constructing a new function from your history

Frequently, making a function will simply be a function of selecting the right parts of your history and hitting the “to source” button.

Function with PARK2 hard coded

```
graphGene <- function(gene) {  
  description <- ambion.simple$Description[ambion.simple$Symbol == "PARK2"][1]  
  myTitle <- paste("PARK2",description,sep=": ")  
  ggplot(ambion.simple, aes(x=PPT, y=Cells)) + geom_density2d() +  
    geom_point(data = ambion.simple %>% filter(Symbol=="PARK2"),  
              color="red", shape=17, size =5) +  
    ggtitle(myTitle)  
}
```

Function made generic

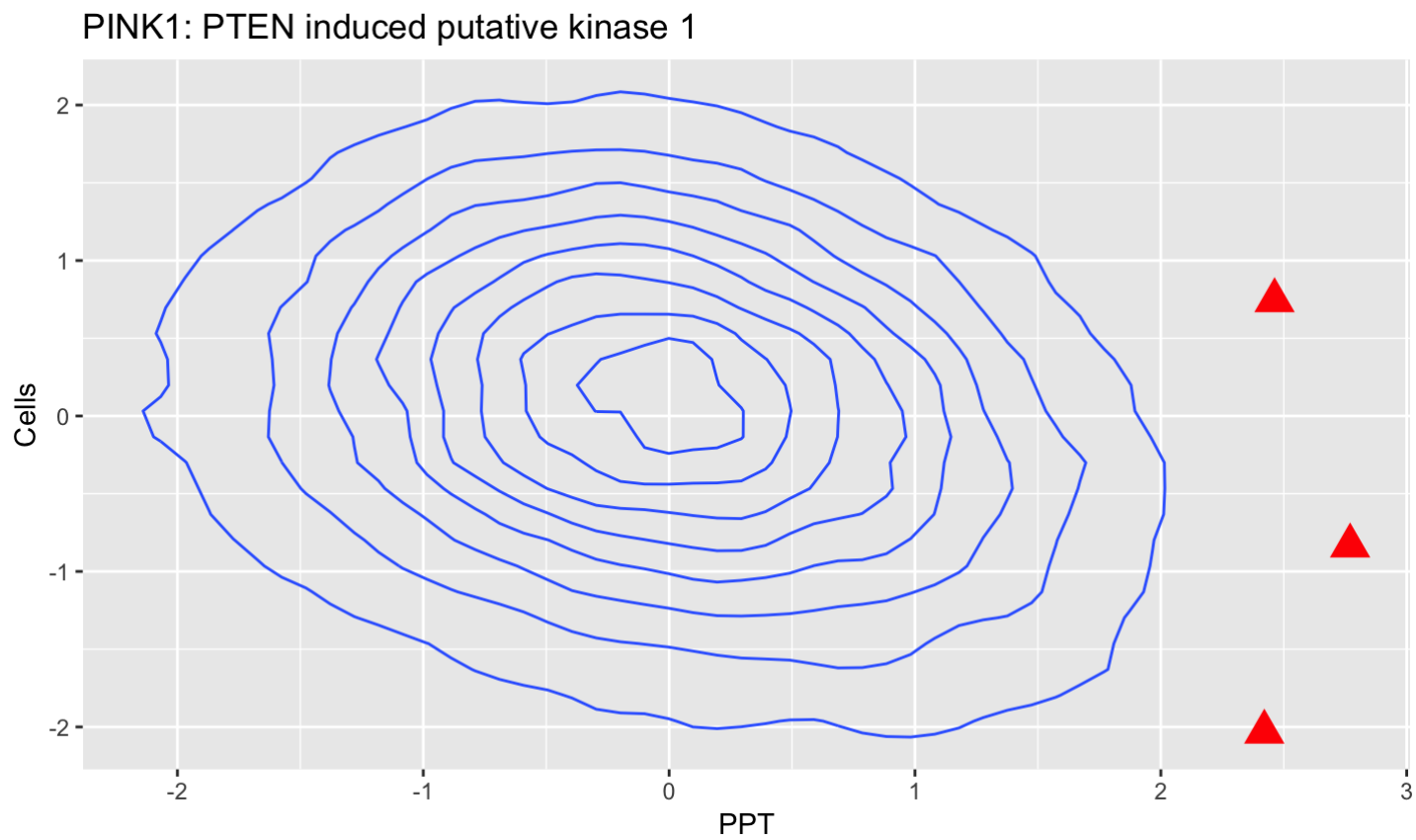
```
graphGene <- function(gene) {  
  description <- ambion.simple$Description[ambion.simple$Symbol == gene][1]  
  myTitle <- paste(gene,description,sep=": ")  
  ggplot(ambion.simple, aes(x=PPT, y=Cells)) + geom_density2d() +  
    geom_point(data = ambion.simple %>% filter(Symbol == gene),  
              color="red", shape=17, size =5) +  
    ggtitle(myTitle)  
}
```

Function made generic, with checks

```
graphGene <- function(gene) {  
  if(!is.character(gene)) stop("Need to provide a string")  
  if(!(gene %in% ambion.simple$Symbol)) stop("Need to provide a valid gene")  
  description <- ambion.simple$Description[ambion.simple$Symbol == gene][1]  
  myTitle <- paste(gene,description,sep=": ")  
  ggplot(ambion.simple, aes(x=PPT, y=Cells)) + geom_density2d() +  
    geom_point(data = ambion.simple %>% filter(Symbol == gene),  
              color="red", shape=17, size =5) +  
    ggtitle(myTitle)  
}
```

Our function in action

```
graphGene( "PINK1" )
```



Default values for function arguments

```
pdfGene <- function(gene, file=paste(gene, ".pdf", sep="")) {  
  pdf(file, width=5, height=5)  
  graphGene(gene)  
  dev.off()  
}
```

Passing on extra arguments to our function

We can use the ellipse notation (...) to indicate that extra arguments to our function should be passed on to a function that is inside our function (in this case pdf).

```
pdfGene <- function(gene, file=paste(gene, ".pdf", sep=""), ...) {  
  pdf(file, ...)  
  graphGene(gene)  
  dev.off()  
}  
pdfGene("PINK1", width=10, height=10)
```

Control of Flow: If/Else

We can decide whether something happens in our function using “if” and “if/else”.

```
sillyFunction <- function(x) {  
  if (x < 5) {  
    returnValue <- x  
  }  
  else {  
    returnValue <- x / 2  
  }  
  return(returnValue)  
}  
sillyFunction(12)
```

```
## [1] 6
```

Control of Flow: For

```
pdfGenes <- function(genes, ...) {  
  for (gene in genes) { # This will work through gene by gene  
    pdfGene(gene, file = paste0(gene, '.pdf'), ...)  
  }  
}  
pdfGenes(c("PLK1", "PINK1", "BRCA1"))
```