

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 of the Data

Gene Symbol		Sample	Median Negative Control on Plate	Median Positive Control on Plate	PPT Sample as Percentage of Negative Control	PPT MAD Log Z-Score	Median PPT Log Mad Z-Score for all siRNAs having the same seed	Number of siRNAs having the same seed	Cell Count on Plate	Cell Count Negative Control on Plate	Cell Count Positive Control on Plate	Sample Cell Count Normalized to Negative Control	Median Negative Control on Plate	Median Positive Control on Plate	Sample Mitophagy, MAD Z-Score Normalized to Negative Control	HUGO Gene Entrez	Row Number	Column Number	Ambion siRNA ID	
4	UAMN1	9.81514	38.61376	96.73501	23.7885583	-2.41729712	-4.82722975	-2.09615684	21	1286	178	1030.5	1.08571777	8.942457	10.462281	22.32055	0.04662106	1	1	QD01771
6	MDI05	7.91536	33.346945	96.652828	23.7361874	-2.40057854	-4.77841378	-1.42101836	152	1203	1310.5	1050.5	2.46574877	16.791355	28.017045	49.016735	0.88787185	15	2	QD01996
7	PFN2	12.26818	31.366188	97.067871	25.4035876	-2.35831265	-5.53930527	-1.42101836	152	1177	1240	978.5	0.39946112	11.299915	29.03928	50.076025	-1.70001322	14	19	QD01288
8	KIAA1191	7.06286	43.564325	96.889563	25.6017581	-2.55312282	-5.51642745	-1.38836379	28	1060	1345.5	1106.5	1.02741248	11.132075	25.534625	27.7216	0.42808378	18	11	QD01435
9	CIBOR1	8.87857	34.335125	96.121174	23.890885	-2.34679463	-4.46880947	-0.48514708	18	1277	1294.5	985.5	1.16824827	12.893082	22.099625	0.94152895	CIBOR1	12	12	QD01863
10	LOC50689	9.59596	36.724075	91.666073	26.1298885	-2.33993117	-4.56202707	-1.38836379	28	1093	1375.5	1061.5	0.56574784	14.181152	10.37487	18.885533	2.08854154	1	2	QD01761
11	UAMN1	7.04537	26.493405	95.491585	26.4482474	-2.3297673	-4.39629723	-1.42101836	152	1109	1209	1019.5	0.73286798	5.128205	8.7637465	20.23281	0.93484089	1	4	QD01367
12	FAM73A	10.74508	39.900205	96.750781	26.94172518	-2.31936042	-4.3660906	-0.762075318	28	854	1202	1059	0.21184759	23.633397	26.594221	44.646865	0.24361276	16	15	QD01088
13	C17orf77	11.32515	40.2143615	96.925455	28.181867	-2.28844919	-4.23547931	-1.07848186	75	1202	1355.5	1025	0.33326082	17.049181	25.9359045	41.648632	0.65791957	17	1	QD01253
14	SAMD1	12.39604	43.454325	96.02091	28.282032	-2.27916385	-4.19737272	-0.35800881	18	1136	1208	1005	0.46105972	9.154599	23.369625	37.7526	1.06135717	1	14	QD01816
15	LOC42441	11.426256	39.890205	96.750781	28.5726185	-2.27826069	-4.15771176	-0.10011668	8	1126	1242	1059	0.02140798	34.63358	26.594221	44.646865	1.84802138	1	4	QD01088
16	WDR24	11.19552	38.802365	96.938275	28.8728575	-2.2694753	-4.16401364	-0.00417018	62	869	1345.5	1041.5	0.28469718	10.126582	27.585454	1.78558099	WDR24	1	18	QD01274
17	C14orf7	9.84816	34.335125	96.121174	28.77445692	-2.26784123	-4.15158176	-1.60215688	136	1296.5	985	1.43425507	10.414376	12.059825	22.029955	0.14847871	1	6	QD01363	
18	LOC71355	10.84614	37.102835	96.6317825	29.3279411	-2.26130825	-4.25445434	-0.41561139	50	1231	1355	1037.5	0.14736309	27.518622	27.637325	42.840236	0.67531389	1	13	QD01375
19	MGC16121	12.14683	41.1336935	96.96704	29.5327547	-2.253770524	-4.095566791	-0.00417018	62	1012	1333.5	1025.5	0.71828563	16.934422	25.36282	40.947165	0.84745085	1	1	QD01269
20	DEFB129	12.22562	41.674112	96.526455	30.04829953	-2.24664658	-4.04432101	-0.61994401	32	1091	1200	1080	0.65846236	8.890925	10.76316	22.06597	0.05415426	16	5	QD01377
21	ATXN1L	11.61615	37.9373795	96.845925	30.0519233	-2.22617923	-3.98808823	-1.46110951	31	1154	1346	1019	0.35238956	26.918189	29.042115	46.5116295	0.27740702	16	18	QD01486
22	CHP	12.39808	33.417905	97.58857	30.638986	-2.2567747	-3.98994167	-0.71271186	30	854	1061	809.5	0.87093993	16.934422	13.86683	23.87091	1.37446036	1	1	QD01259
23	FLJ0125	12.38245	40.2143615	96.925455	30.79110181	-2.21823206	-3.97234541	-1.97118214	78	1207	1315.5	1025.5	0.23617941	24.109362	25.9359045	41.648632	0.39946878	1	16	QD01253
24	VPS13C	4.535559	14.602162	97.703815	30.9521131	-2.21842746	-3.959511391	-1.47071506	17	1224	1371	1088	0.007055429	21.977123	24.113465	37.6343345	0.77891145	16	12	QD01140
25	AMOTL2	10.32165	33.346945	96.652828	30.95220613	-2.21740958	-3.959390665	-1.42101836	152	1232	1303.5	1050.5	0.390613948	12.755519	28.0817045	49.016735	-1.44708005	1	7	QD01996
26	TMC1	12.21442	39.331158	97.046655	31.06497146	-2.21513704	-3.94719319	-1.70579607	15	1198	1257.5	1057	0.58183904	8.597663	13.744405	31.707255	0.78138707	1	1	QD01810
27	SOD4	10.44864	33.603115	97.397395	31.1168715	-2.21356819	-3.94172778	-0.73376651	22	750	972.5	823	1.20215703	1.6	15.16345	24.5215945	-2.88626506	16	14	QD01591
28	LOC28423	10.76721	34.335125	96.121174	31.5774517	-2.20766512	-3.918557383	-1.82819517	37	1151	1296.5	985	0.221578634	7.211212	12.059825	22.029955	-0.88878205	1	6	QD01964
29	FAM74	10.75914	34.17813	94.94933	31.5174636	-2.2041356	-3.90358313	-1.87178796	16	1212	1311	1005	0.30466717	17.19868	5.983825	31.66597	0.34400326	14	15	QD01246
30	NCL_1445727	10.32389	31.07602	95.879448	32.18321598	-2.18654036	-3.84119084	-1.76028254	16	1170	1305	1016	0.38117033	6.466647	10.476745	21.444887	0.73962146	1	1	QD01854
31	RASL1	10.23117	31.579678	97.040435	32.3984204	-2.17960151	-3.78968416	-0.48481356	43	1281	1373.5	1178	0.63711114	8.430913	14.54753	25.32889	0.25325658	1	13	QD01246
32	LOC70187	10.32368	39.8915025	96.71463	32.695407	-2.17426473	-3.79774619	-0.46697523	29	1117	1238	1067	0.08616468	24.082363	28.46302	46.04568	0.07532204	1	13	QD01267
33	PRK5	12.43529	37.9373795	96.845925	32.78615406	-2.17233843	-3.79174041	-0.60532382	34	1305	1346	1019	0.74757371	32.413792	29.042115	46.5116295	1.14259842	1	1	QD01587
34	SEMA6D	5.34974	16.330773	81.414805	32.75897595	-2.17960151	-3.78968416	-0.48481356	43	1281	1373.5	1178	0.63711114	8.430913	14.54753	25.32889	0.25325658	1	13	QD01246
35	FAM70B	10.29588	31.06667	95.909923	33.0009862	-2.16582798	-3.797988278	-1.23264529	34	1291	1328	1013.5	0.77388861	4.377323	14.492465	21.11532	1.74548612	1	18	QD01856
36	LOC728516	12.39008	37.94737	96.172192	33.7502524	-2.16514893	-3.752482408	-1.27594317	40	1202	1259	1015	0.60194531	11.813643	15.780395	22.121678	0.75330912	1	14	QD01758
37	LOC70376	13.31421	39.8915025	96.71463	33.390604	-2.15650145	-3.73529043	-0.686488	17	1168	1287	1067	0.07822386	17.025089	28.46302	46.04568	0.88795212	1	7	QD01267
38	LOC729747	10.43478	38.987595	96.808715	33.4554733	-2.15431199	-3.72766608	-0.47700527	20	1116	1261	1022.5	0.31475863	11.053985	10.666215	24.43183	0.82569498	1	1	QD01743
39	LOC728511	11.44312	40.16214	96.689905	33.47220031	-2.15388601	-3.72451011	-0.70677254	62	1089	1325.5	1072	0.70700914	21.303949	25.790515	42.022265	0.04945972	1	22	QD01545
40	LCOR	5.467801	16.330773	81.414805	33.4815106	-2.15379504	-3.72530737	-1.70596007	15	1430	1375.5	1178	0.22866756	9.300699	14.54753	25.32889	0.71923803	1	4	QD01246
41	IL33	13.78189	41.1336935	96.96704	33.5048627	-2.15360422	-3.72331757	-1.89533346	21	1313	1233.5	1025.5	0.230969879	21.750664	25.36282	40.947165	0.119945521	1	9	QD01269
42	AGR2	13.86963	41.055187	96.555149	33.78346078	-2.14602159	-3.68989072	-1.97118214	38	1097	1312	1062.5	0.48717956	11.668186	16.747305	20.578175	-0.404427157	1	6	QD01940
43	STC2	10.55555	31.06667	95.909923	33.82467919	-2.14595847	-3.69531221	-1.21462451	6	1031	1238	1013.5	0.11557684	5.237633	14.492465	21.11532	-1.44126872	1	12	QD01856
44	TCF11	9.80099	29.26668	97.201075	34.0664886	-2.13896212	-3.67461174	-1.25342587	28	851	1028.5	802	0.64581251	11.054829	19.45772	34.010118	-1.006308	1	8	QD01262
45	LOC729054	11.846154	40.2143615	96.925455	34.07200031	-2.13885996	-3.67423153	-1.70579607	15	1136	1249.5	1020.5	0.51404137	9.950571	11.610969	25.99139	0.60126346	1	20	QD01752
46	INR151	13.75783	40.2143615	96.925455	34.15641201	-2.13655124	-3.66854145	-0.46697523	29	1063	1315.5	1025	0.39890761	18.720022	25.9359045	41.648632	0.04755185	1	6	QD01253
47	UKL1	13.757949	39.728825	96.494448	34.189998	-2.13592827	-3.66442013	-1.45604417	65	975	1253	1029	0.13479018	12.445138	23.895603	41.70009	0.88823181	1	15	QD01259
48	C16orf53	12.19678	41.255145	96.411655	34.1863882	-2.13579121	-3.64695452	-1.47071506	4	1223	1316	1089	0.352272194	14.472609	27.068455	29.563445	0.00890762	1	1	QD01377
49	FLJ36032	12.452012	41.674112	96.526455	34.1963183	-2.13554007	-3.63909273	-0.69739234	49	1177	1208	1080	0.021799016	7.221175	10.76316	22.06597	-0.55899124	1	1	QD01377
50	KCN16	13.07186	38.712395	97.036425	34.2433771	-2.13427942	-3.63878406	-1.91091623	44	874	1328.5	835.5	0.429745018	13.501144	27.574465	37.397896	0.88823181	1	12	QD01242
51	LOC1	12.27212	41.674112	96.526455	34.2497084	-2.134256617	-3.63878406	-1.91091623	44	1211	1373.5	1064	0.38085329	6.438939	16.762116	22.365937	0.72745059	1	9	QD01377
52	ZN787	12.23741	38.987595	96.808715	34.26642792	-2.13373605	-3.65706183	-0.46969666	6	1320	1173	967	2.26793658	15.155152	8.471358	29.063166	0.2781366	1	1</	

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` 11/31
```

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` 12/31
```

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 Contol`
##                                           <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

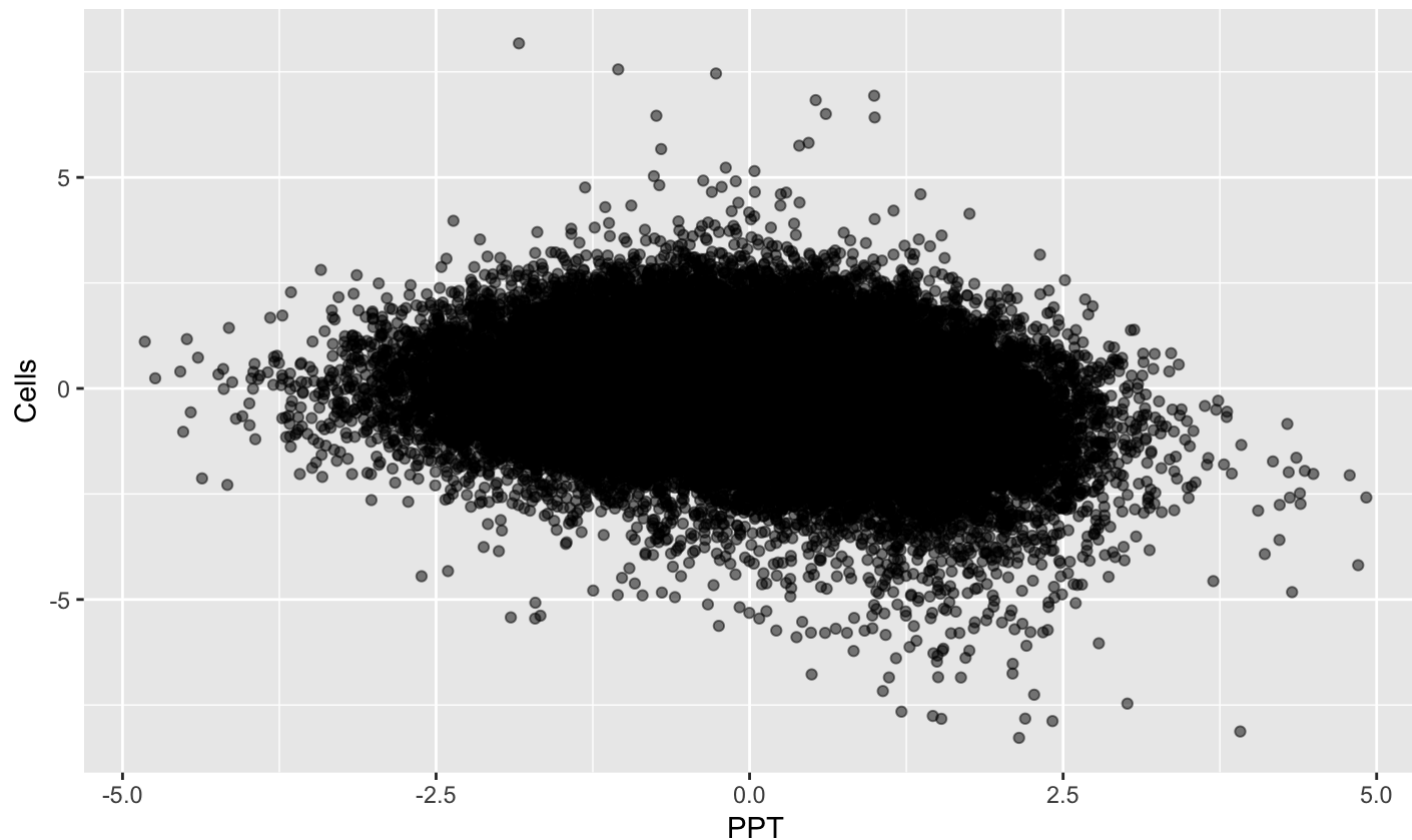
Simplify our Column Names

```
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

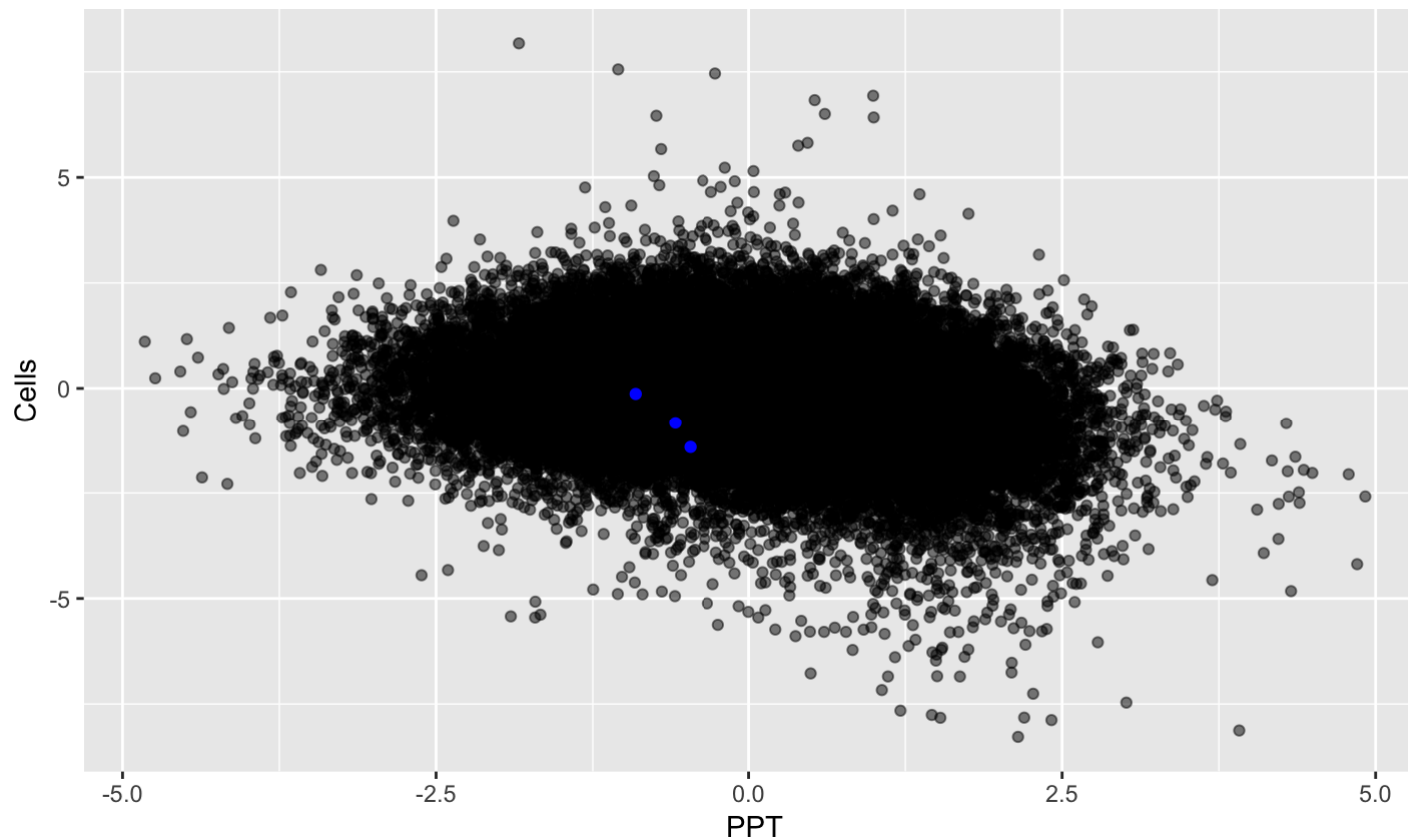
Evaluate how our variables interact

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



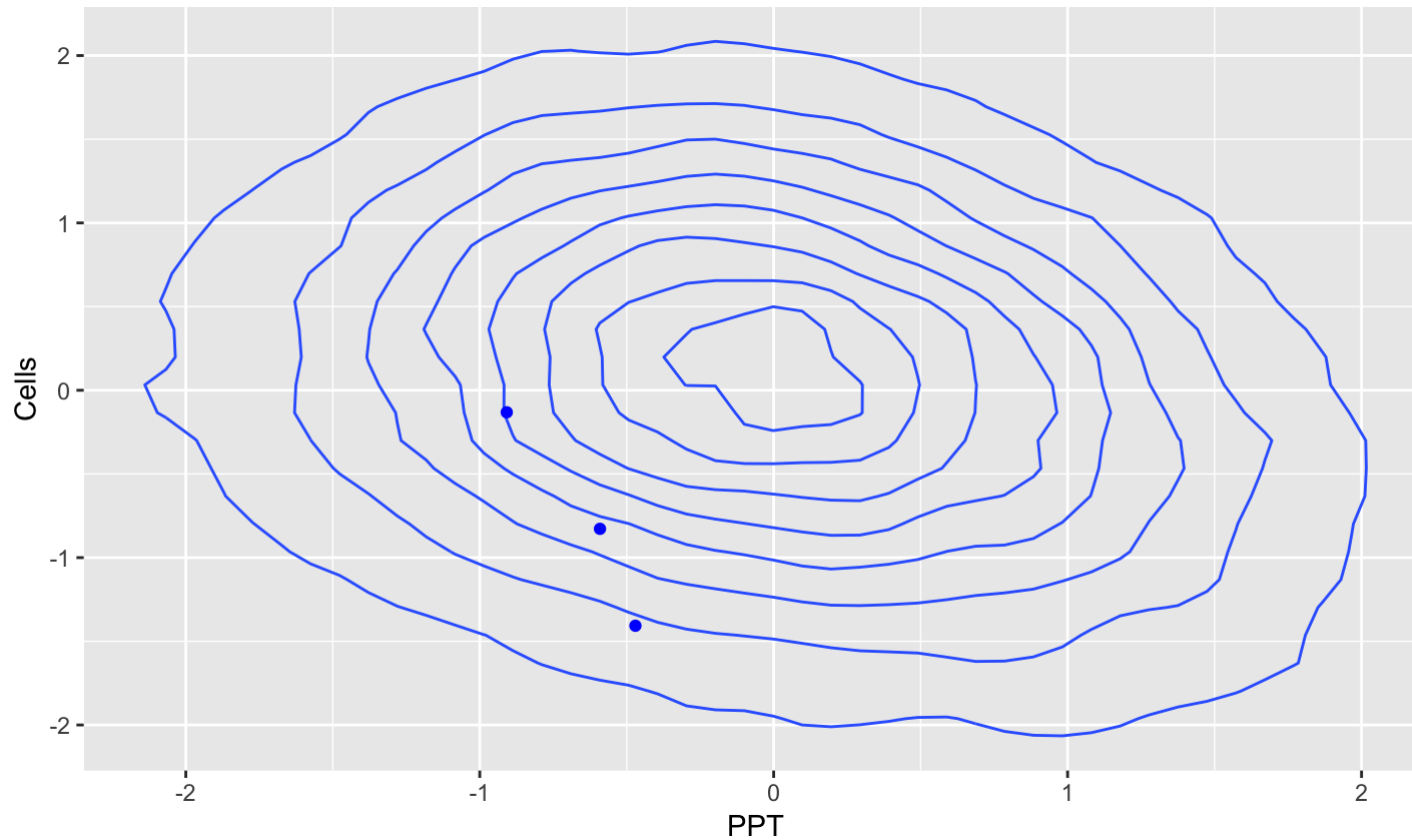
Evaluate how our variables interact

```
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")
```



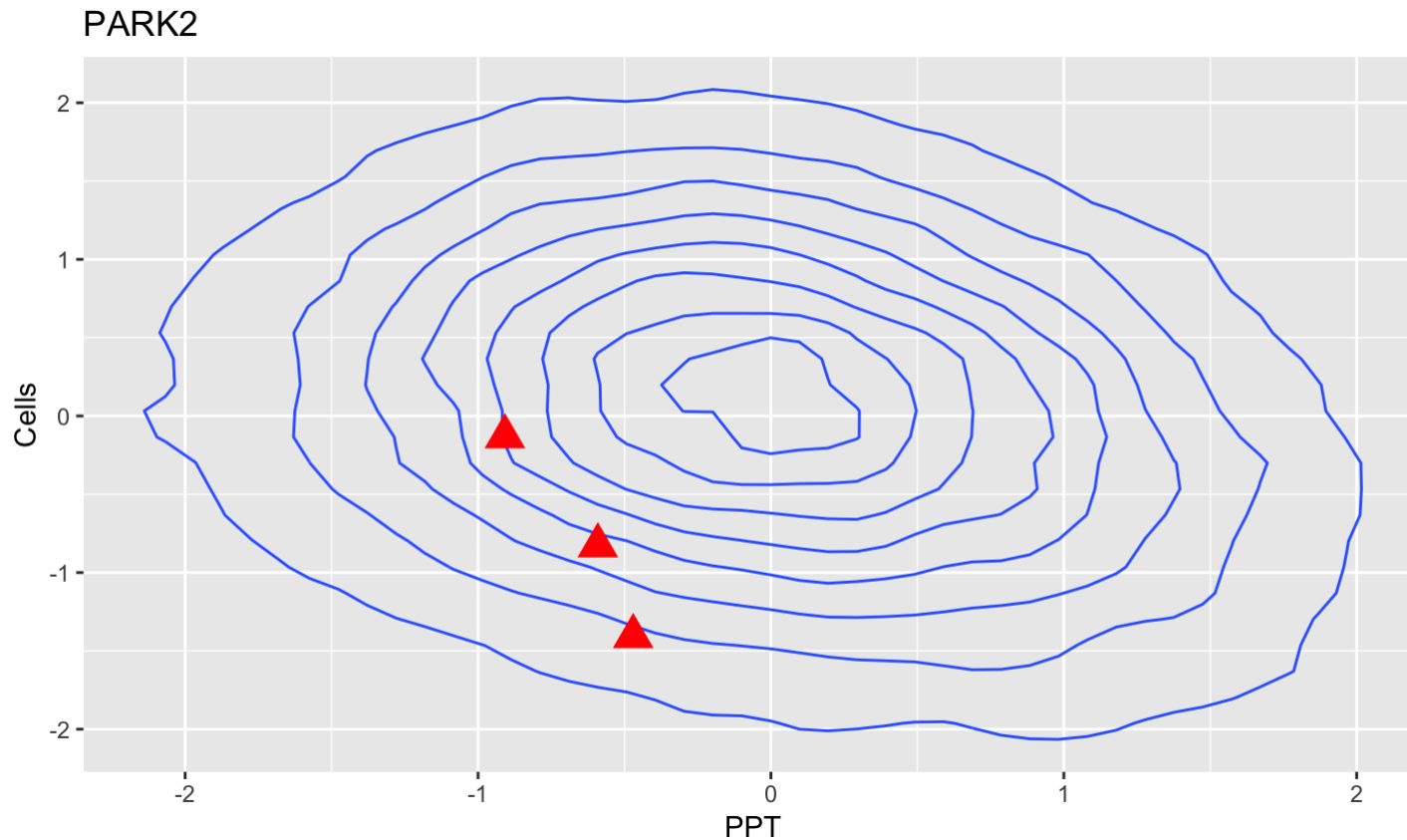
Refine the plot

```
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"
```

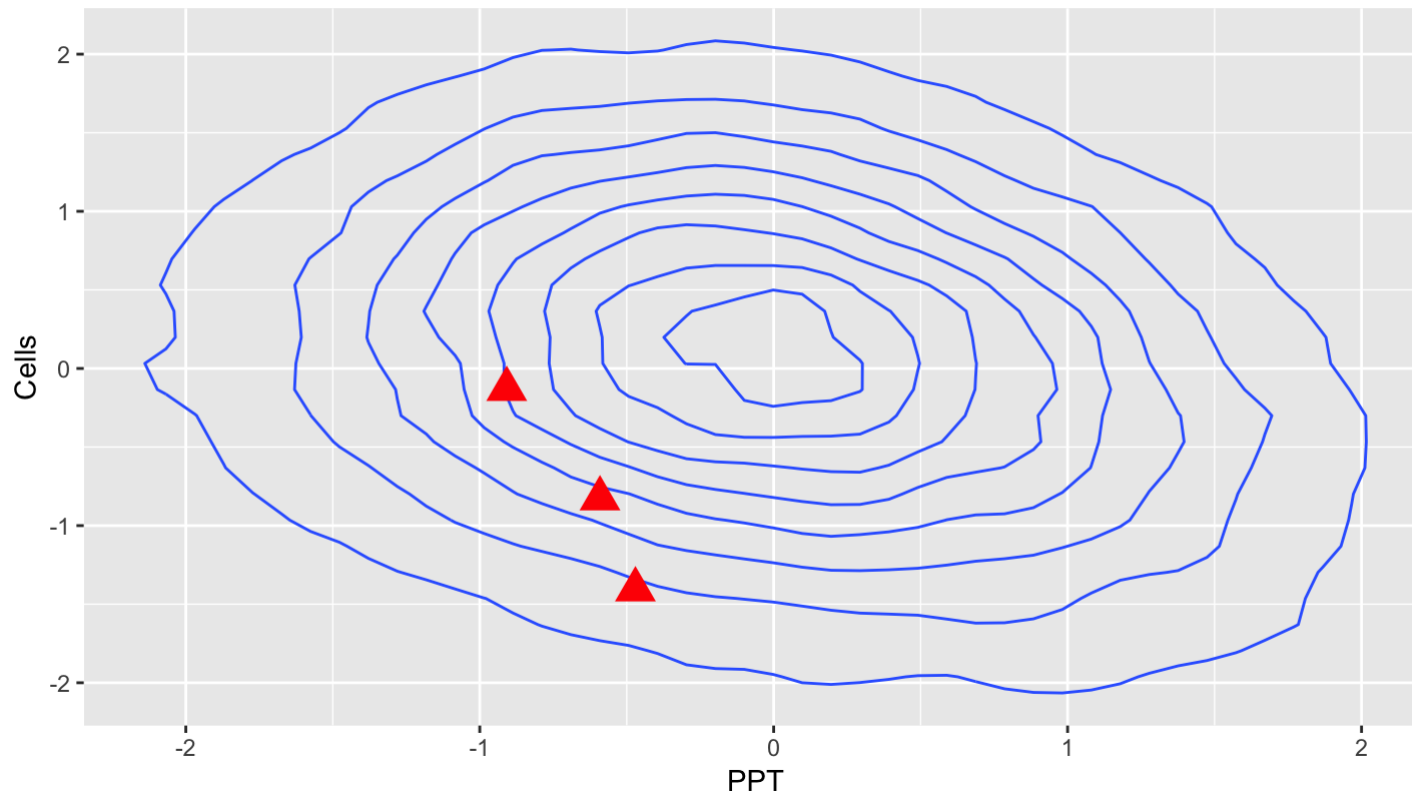
```
myTitle <- paste("PARK2",description,sep=": ")  
myTitle
```

```
## [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

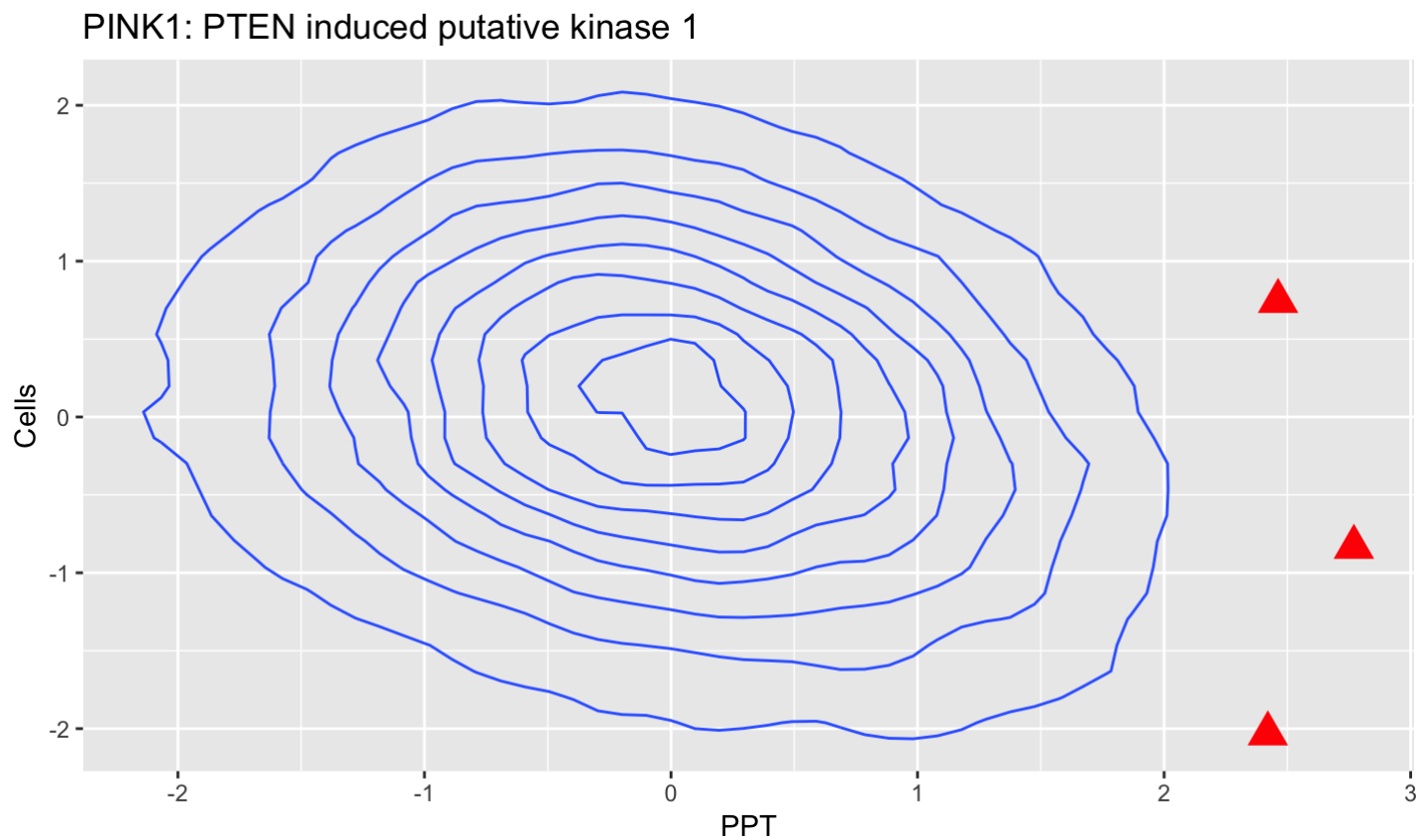
```
graphGene <- function(gene) {  
  description <- ambion.simple$Description[ambion.simple$Symbol == "PARK2"][1]  
  myTitle <- paste("PARK2",description,sep=": ")  
  ggplot(ambion.simple, aes(x=PPT, y=Cells)) + geom_density2d() +  
    geom_point(data = ambion.simple %>% filter(Symbol=="PARK2"),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

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"))
```