gse28619

Maria Teresa Rubio Martinez-Abarca and Guillermo Ayala

5/11/23

Table of contents

Introducción

The dataset has been downloaded from GEO

The summary given there is > Alcoholic hepatitis (AH) is the most severe form of alcoholic liver disease and occurs in patients with excessive alcohol intake It is characterized by marked hepatocellular damage, steatosis and pericellular fibrosis. Patients with severe AH have a poor short-term prognosis. Unfortunately, current therapies (i.e. corticosteroids and pentoxyphylline) are not effective in many patients and novel targeted therapies are urgently needed. The development of such therapies is hampered by a poor knowledge of the underlying molecular mechanisms. Based on studies from animal models, TNF alfa was proposed to play a pivotal role in the mechanisms of AH. Consequently, drugs interfering TNF alfa were tested in these patients. The results were disappointing due to an increased incidence of severe infections. Unluckily, there are not experimental models that mimic the main findings of AH in humans. To overcome this limitation, translational studies with human samples are required. We previously analyzed samples from patients with biopsy-proven AH. In these previous studies, we identified CXC chemokines as a potential therapeutic target for these patients. We expanded these previous observations by performing a high-throughout transcriptome analysis.

Downloading and preprocessing the dataset

The vignette `tamidata::gse28619` details how to download and preprocessing the dataset. The final result obtained using the code given there is an `ExpressionSet` used from now on.

Marginal differential expression

- Loading basic packages needed later.

```
library(Biobase)
```

```
Loading required package: BiocGenerics
```

```
Attaching package: 'BiocGenerics'
```

```
The following objects are masked from 'package:stats':

    IQR, mad, sd, var, xtabs
```

```
The following objects are masked from 'package:base':

    anyDuplicated, aperm, append, as.data.frame, basename, cbind,
    colnames, dirname, do.call, duplicated, eval, evalq, Filter, Find,
    get, grep, grepl, intersect, is.unsorted, lapply, Map, mapply,
    match, mget, order, paste, pmax, pmax.int, pmin, pmin.int,
    Position, rank, rbind, Reduce, rownames, sapply, setdiff, sort,
    table, tapply, union, unique, unsplit, which.max, which.min
```

```
Welcome to Bioconductor

    Vignettes contain introductory material; view with
    'browseVignettes()'. To cite Bioconductor, see
    'citation("Biobase")', and for packages 'citation("pkgname")'.
```

```
library(tami)
```

```
Attaching package: 'tami'
```

```
The following objects are masked from 'package:BiocGenerics':

    type, type<-
```

- Loading the dataset tamidata::gse28619.

```
data(gse28619,package="tamidata")
```

- The number of features (genes) and samples are

```
dim(gse28619)
```

```
Features  Samples
   54675       22
```

- Phenotipic variable?

```
data(gse28619,package="tamidata")
```

- The number of features (genes) and samples are

```
dim(gse28619)
```

```
Features  Samples
   54675       22
```

- Phenotipic variable?

```
pData(gse28619)[,"type"]
```

```
 [1] control   control   control   control   control   control   control
 [8] alcoholic alcoholic alcoholic alcoholic alcoholic alcoholic alcoholic
[15] alcoholic alcoholic alcoholic alcoholic alcoholic alcoholic alcoholic
[22] alcoholic
Levels: control alcoholic
```

- Gene identifiers (not shown)

```
fData(gse28619)
```

Marginal differential expression

The test used is `rowt` corresponding to the t-test assuming a common variance.

```
x1 = dema(x=gse28619, y = "type", test = rowt, correction = "BH",
fdr = 0.01,foutput = "gse28619")
```

```
Warning: replacing previous import 'utils::findMatches' by
'S4Vectors::findMatches' when loading 'AnnotationDbi'
```

Note that only those genes with an adjusted p-value lesser than `fdr` are returned. If we want the analysis for all genes then we have to set `fdr = 1`.

A `data.frame` with the results can be obtained.

```
x1_df = tidy(x1)
```

These results can be examined in a html file.

```
browseURL(glimpse(x1))
```

```
Warning: replacing previous import 'utils::findMatches' by
'S4Vectors::findMatches' when loading 'AnnotationForge'
```

```
Registered S3 method overwritten by 'GGally':
  method from
  +.gg   ggplot2
```

How many genes have an adjusted p-value lesser than `fdr = 0.01`?

```
nrow(x1_df)
```

```
[1] 9198
```

Which is the highest adjusted p-value for the returned genes?

```
max(x1_df[,"adjp"])
```

```
[1] 0.009994994
```

We want to know all the information about the gene with `ENTREZID` identifier `6898`. First we need to know the row within the `data.frame` `x1_df`.

```
grep("^6898",x1_df[,"ENTREZID"])
```

```
[1]  174 4243
```

The expression "^6898" looks for `ENTREZID` beginning with this code. Let us see these rows

```
x1_df[grep("^6898",x1_df[,"ENTREZID"]),]
```

```
            PROBEID ENTREZID         ENSEMBL         GO EVIDENCE ONTOLOGY
22260  1555189_a_at     6898 ENSG00000198650 GO:0004838      IDA       MF
402759    214413_at     6898 ENSG00000198650 GO:0004838      IDA       MF
       statistic         rawp         adjp         qval
22260   5.146400 4.913519e-05 0.0006416569 0.0003251238
402759  3.742766 1.282815e-03 0.0081103027 0.0041094428
```

Why we have two rows? Note that the `PROBEID`'s are different for each row. They are different probe sets but they corresponds to the same gene.

Note that the `ENSEMBL` and `GO` identifiers are `ENSG00000198650` and `GO:0004838` respectively.

Gene set collection

```
load("hsa_go.rda")
hsa_go
```

- How many gene sets?

```
length(hsa_go)
```

```
[1] 22934
```

The first gene set is

```
hsa_go[1]
```

```
$`GO:0000002`
 [1] "5428"   "6742"   "11232"  "55186"  "56652"  "84275"  "92667"  "201973"
 [9] "1763"   "7157"   "9093"   "10891"  "80119"  "83667"  "201163" "142"
[17] "1890"   "2021"   "3980"   "4358"   "4976"   "6240"   "7156"   "10000"
[25] "50484"  "64863"  "219736" "4205"   "9361"   "291"
```

We have the name and the (ENTREZID) identifiers of the genes belonging to this gene set. The elements of the set can be accessed with

```
hsa_go[[1]]
```

```
 [1] "5428"   "6742"   "11232"  "55186"  "56652"  "84275"  "92667"  "201973"
 [9] "1763"   "7157"   "9093"   "10891"  "80119"  "83667"  "201163" "142"
[17] "1890"   "2021"   "3980"   "4358"   "4976"   "6240"   "7156"   "10000"
[25] "50484"  "64863"  "219736" "4205"   "9361"   "291"
```

We can know the number of elements in this groups with

```
length(hsa_go[[1]])
```

```
[1] 30
```

It is not necessary to evaluate the `length` (cardinal) of each group. It can be known the cardinality of each set simultaneously using `lapply`.

```
ngs = lapply(hsa_go,length)
```

What kind of object has been returned?

```
class(ngs)
```

```
[1] "list"
```

The (previously calculated) length of the first gene set is

```
ngs[1]
```

```
$`GO:0000002`
[1] 30
```

or

```
ngs[[1]]
```

```
[1] 30
```

The lengths of the first gene sets can be obtained with

```
ngs[1:10]
```

```
$`GO:0000002`
[1] 30

$`GO:0000003`
[1] 1506

$`GO:0000009`
[1] 2

$`GO:0000010`
[1] 2

$`GO:0000012`
[1] 12

$`GO:0000014`
[1] 10

$`GO:0000015`
[1] 4

$`GO:0000016`
[1] 1
```

```
$`GO:0000017`
[1] 2

$`GO:0000018`
[1] 134
```

The largest gene set is

```
which.max(ngs)
```

```
GO:0005575
      2643
```

How many gene sets have exactly 10 genes?

```
table(ngs == 10)
```

```
FALSE  TRUE
22506   428
```

Other possibility is

```
sum(ngs == 10)
```

```
[1] 428
```

Note that the lengths calculated `ngs` is a `list`. Sometimes it is useful to have a vector.

```
ngs0 = unlist(ngs)
class(ngs0)
```

```
[1] "integer"
```

We can evaluate a table of absolute frequencies with

```
table(ngs0)
```

ngs0

1	2	3	4	5	6	7	8	9	10	11	12	13
4466	2747	1783	1292	989	789	665	590	476	428	400	368	287
14	15	16	17	18	19	20	21	22	23	24	25	26
256	268	213	216	206	177	147	155	145	125	124	123	121
27	28	29	30	31	32	33	34	35	36	37	38	39
119	108	89	106	90	92	68	82	66	73	78	66	63
40	41	42	43	44	45	46	47	48	49	50	51	52
64	52	52	61	61	50	52	55	58	45	42	48	46
53	54	55	56	57	58	59	60	61	62	63	64	65
37	45	36	40	46	33	26	30	34	30	29	35	28
66	67	68	69	70	71	72	73	74	75	76	77	78
21	29	27	21	29	20	33	29	25	24	23	21	25
79	80	81	82	83	84	85	86	87	88	89	90	91
18	32	14	21	21	10	18	24	23	20	16	17	21
92	93	94	95	96	97	98	99	100	101	102	103	104
26	21	26	15	17	23	14	21	13	10	21	18	21
105	106	107	108	109	110	111	112	113	114	115	116	117
14	9	19	11	24	12	10	14	8	13	15	15	13
118	119	120	121	122	123	124	125	126	127	128	129	130
14	10	12	11	16	17	11	10	17	14	6	17	13
131	132	133	134	135	136	137	138	139	140	141	142	143
19	9	8	9	7	10	11	10	11	17	12	12	13
144	145	146	147	148	149	150	151	152	153	154	155	156
6	6	14	11	10	5	9	7	13	10	8	13	15
157	158	159	160	161	162	163	164	165	166	167	168	169
11	3	6	9	10	5	8	7	7	7	6	6	5
170	171	172	173	174	175	176	177	178	179	180	181	182
8	5	8	7	2	8	7	13	5	7	7	5	8
183	184	185	186	187	188	189	190	191	192	193	194	195
6	9	13	5	12	4	6	8	5	6	5	7	7
196	197	198	199	200	201	202	203	204	205	206	207	208
5	13	4	2	4	5	7	4	3	3	3	7	6
209	210	211	212	213	214	215	216	217	218	219	220	221
5	4	4	7	5	8	5	2	4	2	3	4	3
222	223	224	225	226	227	228	229	230	231	232	233	234
3	2	4	7	4	5	3	3	6	5	3	6	8
235	236	237	238	239	240	241	242	243	244	245	246	247
5	3	3	3	4	4	5	3	6	3	5	2	4
248	249	250	251	252	253	254	255	256	257	258	259	260
4	6	9	3	4	3	3	6	2	2	2	5	1
261	262	263	264	265	266	267	268	269	270	271	272	273
2	3	3	1	3	3	1	4	3	3	4	2	4

274	275	276	277	278	279	280	281	282	283	284	285	286
2	3	4	2	3	1	3	2	1	3	2	2	2
288	289	290	292	293	294	295	296	297	298	299	300	301
1	3	3	1	2	2	2	3	3	2	2	5	5
302	303	304	305	306	307	308	309	310	311	312	313	314
3	6	5	2	3	1	6	2	1	2	4	3	2
315	317	319	320	321	322	323	324	325	327	328	329	330
3	5	7	6	1	3	3	4	4	9	5	3	2
331	332	333	334	336	337	338	339	341	342	343	344	345
2	3	4	1	1	2	2	5	3	2	1	1	4
346	347	348	349	351	352	353	354	355	357	359	360	361
2	2	3	5	2	2	3	2	1	1	2	1	3
362	363	364	365	366	367	368	369	370	371	372	373	374
1	1	2	2	3	2	1	2	1	2	1	3	4
375	376	377	378	379	381	382	383	384	385	387	388	389
4	2	2	2	1	3	3	2	1	2	1	1	1
390	392	393	394	395	396	397	398	400	401	402	404	408
4	1	1	2	1	2	1	2	1	2	1	3	1
409	410	413	414	415	417	418	419	420	421	422	423	425
1	2	2	1	6	1	2	1	1	3	4	3	2
426	427	428	429	430	431	432	433	434	435	438	439	440
3	3	2	1	1	1	4	2	2	2	2	3	2
441	442	444	445	446	447	450	452	453	455	457	460	461
2	1	1	2	2	2	3	3	2	1	2	1	3
463	464	465	466	467	468	472	473	474	475	477	478	479
2	1	1	2	3	2	1	1	1	3	1	3	3
482	483	484	486	487	488	489	490	491	492	495	496	497
1	3	1	1	2	4	2	1	1	2	5	1	4
500	503	504	505	506	507	508	510	513	514	515	516	517
2	4	1	2	1	1	3	2	2	1	1	1	1
518	519	524	530	533	534	535	536	537	542	543	546	547
2	1	1	1	1	1	2	2	2	1	1	2	1
549	551	553	554	555	556	557	560	561	562	565	566	567
1	2	1	2	2	1	1	1	1	1	5	1	1
568	569	571	572	573	575	576	577	580	581	583	587	588
1	2	2	1	1	1	1	3	1	1	2	1	1
589	591	593	594	595	596	598	599	601	602	605	607	608
1	1	1	3	3	2	1	2	1	3	2	1	2
609	610	611	612	613	614	616	617	619	621	622	625	626
3	1	3	1	1	1	2	1	1	3	1	2	1
627	630	631	632	633	634	636	637	638	639	642	645	649
1	1	1	1	2	1	2	1	1	1	1	2	1
652	653	655	658	660	666	667	668	669	676	680	681	682

1	1	1	1	3	1	1	2	2	1	1	1	1
683	686	689	690	691	692	693	695	698	701	702	703	706
1	1	2	1	2	1	1	1	1	1	1	2	1
707	708	717	719	720	721	725	726	727	729	731	732	734
1	1	2	1	3	1	2	1	1	1	1	2	1
739	741	746	747	748	751	755	759	764	765	767	769	770
1	2	1	1	2	1	1	1	1	2	2	1	2
773	777	779	782	783	785	787	788	789	790	791	796	803
1	1	1	2	1	1	1	1	1	1	1	1	1
807	808	812	815	818	820	823	826	828	831	833	834	835
1	1	1	1	1	2	1	1	2	1	1	2	1
836	840	843	845	847	850	852	855	860	863	864	871	875
1	1	1	1	1	1	1	1	2	2	2	1	1
878	880	888	896	897	900	902	905	911	913	916	919	921
2	1	1	1	1	1	1	2	1	2	1	1	1
926	929	930	932	933	934	936	938	940	942	948	950	954
1	2	1	2	3	1	1	1	1	1	1	1	1
956	957	960	961	962	963	964	967	969	976	979	980	986
1	1	1	2	1	1	1	1	1	1	1	1	1
994	1001	1002	1011	1012	1017	1018	1019	1024	1025	1026	1029	1033
1	1	1	1	1	1	1	1	1	1	1	2	1
1036	1040	1043	1044	1045	1046	1051	1063	1066	1069	1074	1078	1080
1	1	1	1	1	1	1	1	2	1	1	1	2
1093	1094	1096	1099	1104	1107	1112	1122	1127	1137	1140	1143	1144
2	1	1	1	1	1	1	1	1	1	1	1	1
1149	1154	1165	1170	1175	1176	1188	1192	1193	1204	1208	1209	1212
1	2	2	1	1	1	1	1	1	1	1	1	1
1220	1221	1223	1224	1227	1240	1245	1246	1248	1250	1251	1256	1260
1	1	1	1	1	1	1	1	1	1	1	1	2
1277	1278	1284	1285	1288	1291	1292	1294	1306	1308	1312	1325	1327
1	1	1	1	1	1	1	1	1	1	1	1	1
1343	1351	1352	1357	1360	1370	1371	1384	1387	1389	1398	1407	1419
1	1	1	1	1	2	1	1	1	1	1	1	1
1429	1430	1443	1446	1455	1459	1461	1467	1468	1473	1477	1478	1488
1	1	1	1	1	1	1	1	1	1	1	1	2
1493	1498	1505	1506	1508	1509	1511	1512	1515	1518	1522	1524	1537
1	1	1	1	1	1	2	1	1	3	1	1	1
1546	1547	1548	1552	1558	1572	1573	1575	1579	1580	1588	1590	1611
1	1	2	1	1	1	1	1	1	2	1	1	1
1624	1626	1637	1640	1667	1669	1678	1685	1687	1689	1690	1706	1710
1	1	1	1	2	1	2	1	2	2	1	1	1
1713	1719	1741	1767	1769	1771	1772	1776	1778	1779	1790	1807	1813
1	1	1	1	1	1	1	1	1	1	1	1	1

1814	1815	1823	1824	1830	1850	1854	1876	1893	1903	1912	1928	1952
1	1	1	1	2	1	1	1	1	1	1	1	1
1975	1977	1979	1989	1995	2009	2014	2015	2031	2048	2053	2060	2063
1	1	1	1	1	1	1	1	1	1	1	1	2
2068	2134	2139	2140	2142	2143	2150	2164	2182	2211	2215	2242	2243
1	1	1	1	1	1	1	1	1	1	1	1	1
2245	2260	2290	2308	2309	2322	2343	2344	2356	2377	2379	2406	2414
1	1	1	1	2	1	1	1	1	1	2	1	1
2474	2476	2483	2484	2485	2487	2504	2539	2544	2605	2610	2646	2663
1	1	1	1	1	1	1	1	1	1	1	1	1
2702	2721	2736	2756	2782	2813	2815	2837	2851	2875	2961	3023	3087
1	1	1	2	1	1	1	1	1	1	1	1	1
3099	3101	3107	3157	3246	3248	3271	3273	3276	3323	3360	3362	3381
1	1	1	1	1	1	1	1	1	1	1	1	1
3397	3510	3518	3538	3547	3557	3562	3592	3593	3687	3822	3825	3837
1	1	1	1	1	1	1	1	1	1	1	1	1
3877	3897	3908	3911	3988	3999	4024	4060	4065	4097	4126	4155	4193
1	1	1	1	1	1	1	1	1	1	1	1	1
4209	4260	4316	4344	4346	4498	4517	4650	4676	4701	4707	4721	4862
1	1	1	1	1	1	1	1	1	1	1	1	1
4939	4944	5222	5306	5411	5430	5446	5516	5532	5589	5611	5668	5715
1	1	1	1	1	1	1	1	1	1	1	1	1
5724	5725	5731	5744	5790	5902	5917	5962	6024	6026	6066	6075	6133
1	1	1	1	1	1	1	1	1	1	3	1	1
6147	6185	6205	6217	6352	6374	6426	6460	6476	6517	6550	6566	6582
1	1	1	1	1	1	1	1	1	1	1	1	1
6628	6894	7070	7531	7803	7919	9141	9509	9930	10046	10156	10563	11047
1	1	1	1	1	1	1	1	1	1	1	1	1
11721	12037	12044	12146	12378	12496	13520	13602	13998	14483	15196	16581	16932
1	1	1	1	1	1	1	1	1	1	1	1	1
18369	18614	19103	19518									
1	1	1	1									

How many gene sets have more than 5 genes?

```
sum(ngs0 > 5)
```

```
[1] 11657
```

Or greater or equal to 5?

```
sum(ngs0 >= 5)
```

```
[1] 12646
```

How many groups have a number of genes greater than 7 and less than 90?

```
sum(ngs0 > 7 & ngs0 < 90)
```

```
[1] 7604
```

How many groups have less than 34 or more than 67 genes?

```
sum(ngs0 < 34 | ngs0 > 67)
```

```
[1] 21321
```

Over representation analysis

We have loaded the gene set collection `hsa_go` previously downloaded.

```
x1_ora = overRepresentation(x1,minsize=5,maxsize = 100,
                            correction = "BH",
                            GeneSetList = hsa_go,
                            foutput="x1_ora")
```

A `data.frame` with the results.

```
x1_ora_df = tidy(x1_ora)
```

A html report

```
glimpse(x1_ora)
```

```
[1] "./reports/x1_ora.html"
```

that can be opened with

```
browseURL(glimpse(x1_ora))
```

Gene set analysis

```
x1_self_mean = GeneSetTest(x = gse28619 ,y="type",
         test = rowt,association="pvalue",correction="BH",
         GeneNullDistr = "randomization",minsize = 5,
         GeneSetNullDistr = "self-contained",
         alternative="less",nmax = 1000,
         id = "ENTREZID",gsc=hsa_go,descriptive=mean,
         foutput = "x1_self_mean")
```

```
x1_self_mean_df = tidy(x1_self_mean)
glimpse(x1_self_mean)
```

```
[1] "./reports/x1_self_mean.html"
```

```
browseURL(glimpse(x1_self_mean))
```