main

September 1, 2021

1 Plotting eQTLs, increase font sizes

1.0.1 Kynon Jade Benjamin and Apuã Paquola

[1]:
```
import re
import functools
import subprocess
import numpy as np
import pandas as pd
from plotnine import *
from pandas_plink import read_plink
from warnings import filterwarnings
from matplotlib.cbook import mplDeprecation

filterwarnings("ignore",category=mplDeprecation)
filterwarnings('ignore', category=UserWarning, module='plotnine.*')
filterwarnings('ignore', category=DeprecationWarning, module='plotnine.*')
```

1.1 Configuration

[2]:
```
tissue = "caudate"; feature = "genes"
config = {
    'biomart_file': '../_h/biomart.csv',
    'residual_expression_file': "../../../../prep_eqtl_analysis/%s/%s/
 ↪covariates/" % (tissue, feature)+\
    "residualized_expression/_m/%s_residualized_expression.csv" % feature,
    'phenotype_file': '/ceph/projects/v4_phase3_paper/inputs/phenotypes/_m/
 ↪merged_phenotypes.csv',
    'plink_file_prefix': '/ceph/projects/v4_phase3_paper/inputs/genotypes/_m/
 ↪LIBD_Brain_TopMed',
    'eqtl_output_file': '../../../summary_table/_m/
 ↪Brainseq_sex_interacting_4features_3regions.eFeatures.txt.gz',
    'gwas_snp_file': '/ceph/projects/v4_phase3_paper/inputs/sz_gwas/pgc2_clozuk/
 ↪map_phase3/_m/libd_hg38_pgc2sz_snps_p5e_minus8.tsv'
}
```

1.2 Functions

1.2.1 Expression functions

[3]:
```
@functools.lru_cache()
def tissue_map(tissue):
    return {"caudate": "Caudate", "dlpfc": "DLPFC",
            "hippocampus": "Hippocampus"}[tissue]


@functools.lru_cache()
def feature_map(feature):
    return {"genes": "Gene", "transcripts": "Transcript",
            "exons": "Exon", "junctions": "Junction"}[feature]


@functools.lru_cache()
def get_biomart_df():
    biomart = pd.read_csv(config['biomart_file'], index_col=0)
    biomart['description'] = biomart['description'].str.replace('\[Source.
 ↪*$','', regex=True)
    return biomart


@functools.lru_cache()
def get_residual_expression_df():
    return pd.read_csv(config['residual_expression_file'], index_col=0).
 ↪transpose()


@functools.lru_cache()
def get_pheno_df():
    return pd.read_csv(config['phenotype_file']).set_index("BrNum").loc[:,␣
 ↪["RNum", "Sex", "Dx"]]


@functools.lru_cache()
def get_expression_and_pheno_df():
    return pd.merge(get_pheno_df(), get_residual_expression_df(),
                    left_index=True, right_index=True)


@functools.lru_cache()
def get_gene_id_df():
    return pd.DataFrame({'gene_id': get_residual_expression_df().columns,
                         'ensembl_gene_id': get_residual_expression_df().
 ↪columns.str.replace('\..+$','', regex=True)})
```

```
@functools.lru_cache()
def gene_info_from_symbol(gene_symbol):
    return␣
 ↪get_biomart_df()[get_biomart_df()['external_gene_name']==gene_symbol]\
        .merge(get_gene_id_df(), on='ensembl_gene_id', how='left')


@functools.lru_cache()
def gene_id_from_symbol(gene_symbol):
    df = gene_info_from_symbol(gene_symbol)
    assert df.shape[0] == 1
    return df[['gene_id']].iloc[0].values[0]
```

1.2.2 Genotype and eQTL functions

[4]:
```
def letter_snp(number, a0, a1):
    '''
    Example:
    letter_snp(0, 'A', 'G') is 'AA'
    letter_snp(1, 'A', 'G') is 'AG'
    letter_snp(2, 'A', 'G') is 'GG'
    '''
    if np.isnan(number):
        return np.nan
    if len(a0)==1 and len(a1)==1:
        sep = ''
    else:
        sep = ' '
    return sep.join(sorted([a0]*int(number) + [a1]*(2-int(number))))


@functools.lru_cache()
def get_plink_tuple():
    '''
    Usage: (bim, fam, bed) = get_plink_tuple()
    '''
    return read_plink(config['plink_file_prefix'])


@functools.lru_cache()
def get_eFeature_df():
    eqtl_df = pd.read_csv(config["eqtl_output_file"], sep='\t')
    return eqtl_df[(eqtl_df["Type"] == feature_map(feature)) &
                   (eqtl_df["Tissue"] == tissue_map(tissue))]
```

```
@functools.lru_cache()
def get_gwas_snps():
    return pd.read_csv(config['gwas_snp_file'], sep='\t', index_col=0)


@functools.lru_cache()
def get_risk_allele(snp_id):
    gwas_snp = get_gwas_snp(snp_id)
    if gwas_snp['OR'].iloc[0] > 1:
        ra = gwas_snp['A1'].iloc[0]
    else:
        ra = gwas_snp['A2'].iloc[0]
    return ra


@functools.lru_cache()
def get_snp_df(snp_id):
    '''
    Returns a dataframe containing the genotype on snp snp_id.
    The allele count is the same as in the plink files.

    Example:
    get_snp_df('rs653953').head(5)

            rs653953_num rs653953_letter rs653953
    Br5168             0              GG    0\nGG
    Br2582             1              AG    1\nAG
    Br2378             1              AG    1\nAG
    Br5155             2              AA    2\nAA
    Br5182             2              AA    2\nAA
    '''
    (bim, fam, bed) = get_plink_tuple()
    brain_ids = list(set(get_expression_and_pheno_df().index).
↪intersection(set(fam['fid'])))
    snp_info = bim[bim['snp']==snp_id]
    snp_pos = snp_info.iloc[0]['i']
    fam_pos = list(fam.drop_duplicates(subset="fid").set_index('fid').
↪loc[brain_ids]['i'])
    dfsnp = (pd.DataFrame(bed[[snp_pos]].compute()[:,fam_pos],
                          columns=brain_ids, index=[snp_id + '_num'])
             .transpose().dropna())
    my_letter_snp = functools.partial(letter_snp, a0=snp_info.iloc[0]['a0'],␣
↪a1=snp_info.iloc[0]['a1'])
    dfsnp[[snp_id + '_num']] = 2 - dfsnp[[snp_id + '_num']].astype('int')
    dfsnp[snp_id + '_letter'] = dfsnp[snp_id + '_num'].apply(my_letter_snp)
    dfsnp[snp_id] = (dfsnp[snp_id + '_num'].astype('str') + '\n' +
                     dfsnp[snp_id + '_letter'].astype('str')).astype('category')
```

```
    return dfsnp


@functools.lru_cache()
def get_gwas_ordered_snp_df(snp_id):
    '''
    Returns a dataframe containing the genotype on snp snp_id.
    The allele count is the number of risk alleles according to GWAS.

    Example:
    get_gwas_ordered_snp_df('rs653953').head(5)

            rs653953_num rs653953_letter rs653953
    Br5168             2              GG    2\nGG
    Br2582             1              AG    1\nAG
    Br2378             1              AG    1\nAG
    Br5155             0              AA    0\nAA
    Br5182             0              AA    0\nAA
    '''
    pgc = get_gwas_snps()
    dfsnp = get_snp_df(snp_id).copy()
    gwas_snp = get_gwas_snp(snp_id)
    if gwas_snp['pgc2_a1_same_as_our_counted'].iloc[0]:
        if gwas_snp['OR'].iloc[0] > 1:
            pass
        else:
            dfsnp[[snp_id + '_num']] = 2 - dfsnp[[snp_id + '_num']]
    else:
        if gwas_snp['OR'].iloc[0] > 1:
            dfsnp[[snp_id + '_num']] = 2 - dfsnp[[snp_id + '_num']]
        else:
            pass
    dfsnp[snp_id] = (dfsnp[snp_id + '_num'].astype('str') + '\n' +
                     dfsnp[snp_id + '_letter'].astype('str')).astype('category')
    return dfsnp
```

1.2.3 Plotting functions

[5]:
```
def get_snp_gene_pheno_df(snp_id, gene_id, snp_df_func):
    pheno_columns = list(get_pheno_df().columns)
    expr_df = get_expression_and_pheno_df()[pheno_columns + [gene_id]]
    snp_df =  snp_df_func(snp_id)
    return expr_df.merge(snp_df, left_index=True, right_index=True)


def simple_snp_expression_plot_impl(snp_id, gene_id, snp_df_func):
    df = get_snp_gene_pheno_df(snp_id, gene_id, snp_df_func)
```

```
    y0 = df[gene_id].quantile(.01) - 0.26
    y1 = df[gene_id].quantile(.99) + 0.26
    p = ggplot(df, aes(x=snp_id, y=gene_id, fill='Sex')) \
    + geom_boxplot(alpha=0.4, outlier_alpha=0) \
    + geom_jitter(position=position_jitterdodge(jitter_width=0.25),
                  stroke=0, alpha=0.6) \
    + ylim(y0, y1) \
    + theme_bw(base_size=15) \
    + theme(panel_grid=element_blank(),
            axis_title=element_text(face="bold"))
    return p


def simple_snp_expression_plot(snp_id, gene_id):
    return simple_snp_expression_plot_impl(snp_id, gene_id, get_snp_df)


def simple_gwas_ordered_snp_expression_plot(snp_id, gene_id):
    return simple_snp_expression_plot_impl(snp_id, gene_id,␣
 ↪get_gwas_ordered_snp_df)


def get_gene_symbol(gene_id, biomart=get_biomart_df()):
    ensge = re.sub('\..+$','', gene_id)
    ggg = biomart[biomart['ensembl_gene_id']==ensge]
    if ggg.shape[0]==0:
        return '', ''
    gs = ggg['external_gene_name'].values[0]
    de = ggg['description'].values[0]
    if type(de)!=str:
        de = ''
    de = re.sub('\[Source:.*$','',de)
    return gs, de


def get_gwas_snp(snp_id):
    gwas = get_gwas_snps()
    r = gwas[gwas['our_snp_id']==snp_id]
    assert len(r) == 1
    return r


def gwas_annotation(snp_id):
    return 'SZ GWAS pvalue: %.1e' % get_gwas_snp(snp_id).iloc[0]['P']


def eqtl_annotation(snp_id, gene_id):
```

```
    eqtl_df = get_eFeature_df()
    r = eqtl_df[(eqtl_df['variant_id']==snp_id) & (eqtl_df['gene_id']==gene_id)]
    assert len(r)==1
    return 'eQTL adjusted p-value: %.1e' % r.iloc[0]['BF']


def risk_allele_annotation(snp_id):
    return 'SZ risk allele: %s' % get_risk_allele(snp_id)


def annotated_eqtl_plot(snp_id, gene_id):
    p = simple_snp_expression_plot(snp_id, gene_id)
    gene_symbol, gene_description = get_gene_symbol(gene_id)
    title ="\n".join([gene_symbol,
                      eqtl_annotation(snp_id, gene_id)
                      ])
    p += ggtitle(title) + ylab('Residualized Expression')
    return p


def gwas_annotated_eqtl_plot(snp_id, gene_id):
    p = simple_gwas_ordered_snp_expression_plot(snp_id, gene_id)
    gene_symbol, gene_description = get_gene_symbol(gene_id)
    title ="\n".join([gene_symbol,
                      eqtl_annotation(snp_id, gene_id),
                      gwas_annotation(snp_id),
                      risk_allele_annotation(snp_id)
                      ])
    p += ggtitle(title) + ylab('Residualized Expression')
    return p


def save_plot(p, fn):
    for ext in ['png', 'pdf', 'svg']:
        p.save(fn + '.' + ext)
```

1.3 Plot eQTLs

1.3.1 DRD2

[6]:
```
get_eFeature_df()[(get_eFeature_df()["gene_id"] == gene_id_from_symbol('DRD2'))]
```

[6]:
```
Empty DataFrame
Columns: [variant_id, gene_id, gencodeID, slope, statistic, pval_nominal, BF,
eigenMT_BH, TESTS, Type, Tissue]
Index: []
```

1.3.2 Top 5 eQTLs

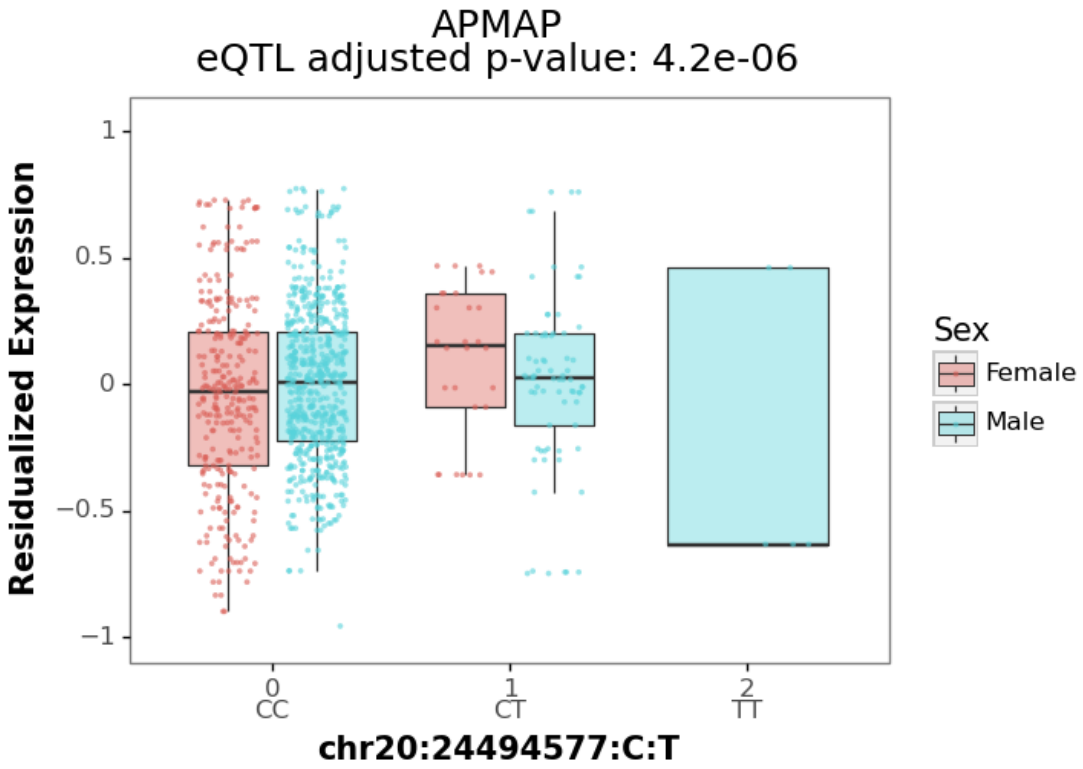
[7]:
```
eqtl_df = get_eFeature_df()
eqtl_df.head()
```

[7]:

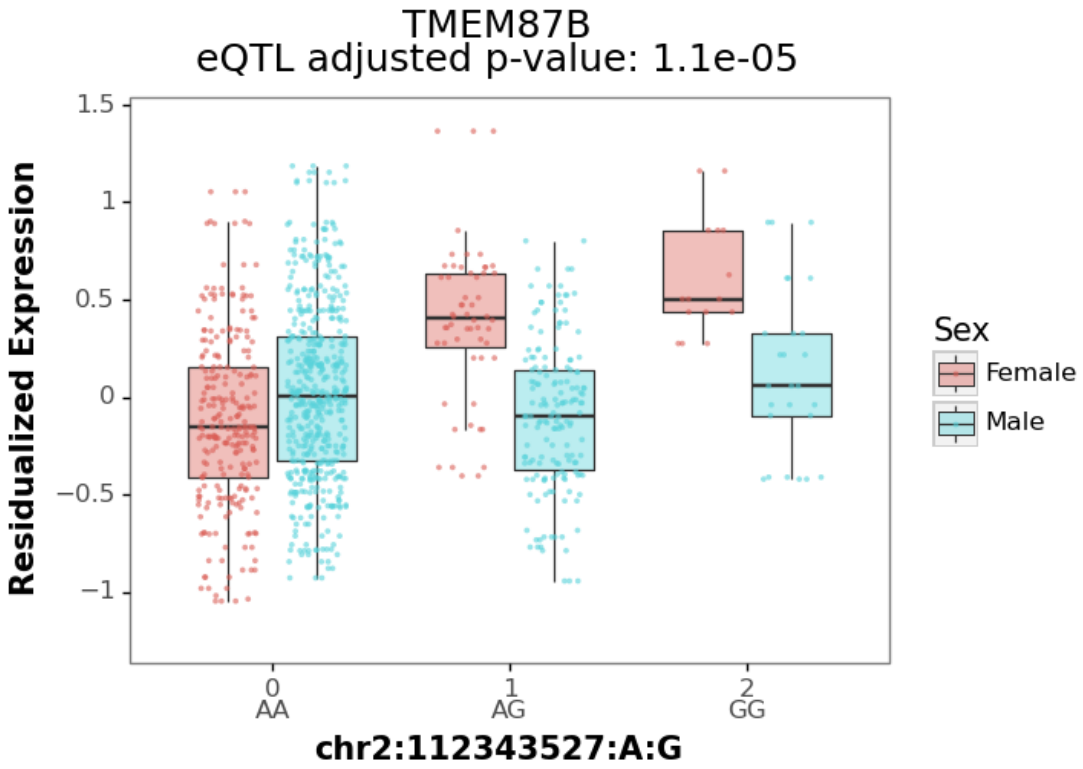
	variant_id	gene_id	gencodeID	slope	statistic	pval_nominal	BF	eigenMT_BH	TESTS	Type	Tissue
7125	chr7:42936690:A:C	ENSG00000002746.14	ENSG00000002746.14	0.436502	7.820971	0.000023	0.010399	0.445692	445	Gene	Caudate
7126	chr17:48075934:C:T	ENSG00000002919.14	ENSG00000002919.14	-0.536903	-7.959488	0.000127	0.037424	0.549367	294	Gene	Caudate
7127	chr7:7934340:A:C	ENSG00000003147.17	ENSG00000003147.17	0.213896	11.357107	0.000022	0.022195	0.503766	1002	Gene	Caudate
7128	chr16:89536451:CGGTGAGGCG:C	ENSG00000003249.13	ENSG00000003249.13	-0.524657	-9.921271	0.000028	0.014577	0.470811	529	Gene	Caudate
7129	chr8:17754947:C:T	ENSG00000003989.16	ENSG00000003989.16	-0.497351	-11.806077	0.000007	0.007416	0.417015	1040	Gene	Caudate

[8]:
```
top_5 = eqtl_df.sort_values('pval_nominal').reset_index(drop=True).head(5)
for x in top_5.itertuples():
    filename = "top_%d_eqtl_%s" % (x.Index, tissue)
    p = annotated_eqtl_plot(x.variant_id, x.gene_id)
    print(filename, x.Index, x.variant_id, x.gene_id)
    print(p)
    save_plot(p, filename)
```

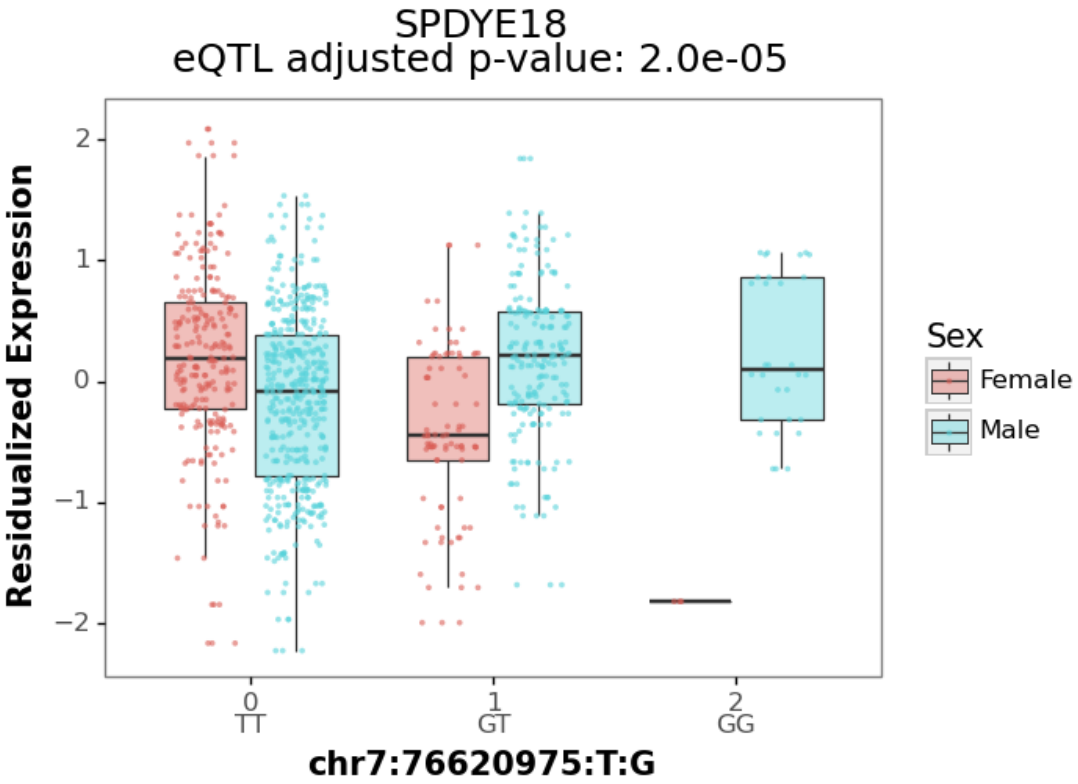
```
Mapping files: 100%|      | 3/3 [00:26<00:00,  8.79s/it]

top_0_eqtl_caudate 0 chr20:24494577:C:T ENSG00000101474.11
```



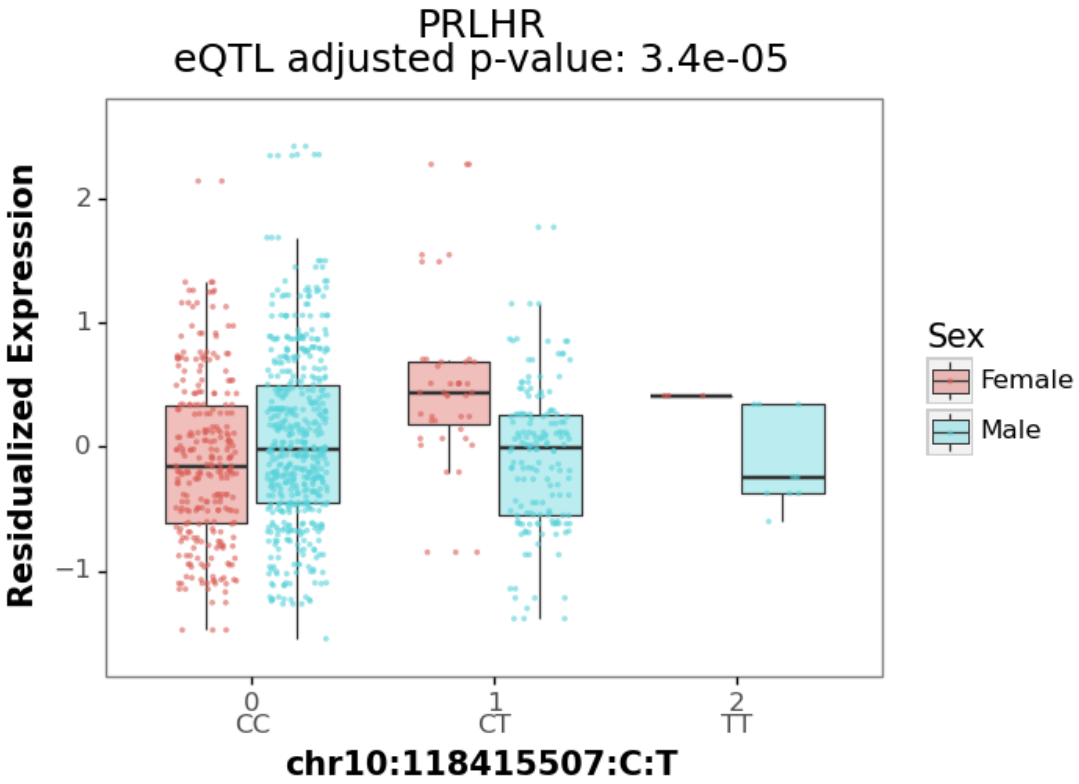

```
<ggplot: (8728063515606)>
top_1_eqtl_caudate 1 chr2:112343527:A:G ENSG00000153214.9
```



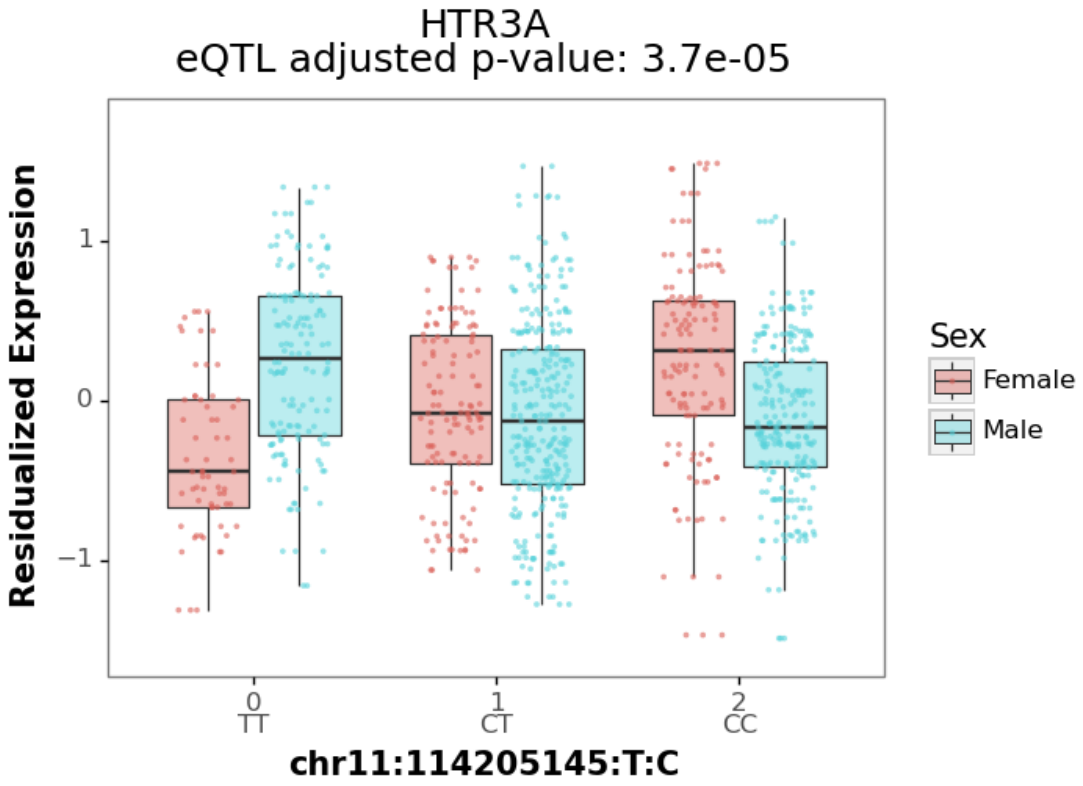

```
<ggplot: (8728063342174)>
top_2_eqtl_caudate 2 chr7:76620975:T:G ENSG00000205482.9
```



<ggplot: (8728063254492)>
top_3_eqtl_caudate 3 chr10:118415507:C:T ENSG00000119973.5



<ggplot: (8728062620182)>
top_4_eqtl_caudate 4 chr11:114205145:T:C ENSG00000166736.11




```
<ggplot: (8728062803366)>
```

1.3.3 Top 5 X-linked genes

[9]:
```
top_5_x = eqtl_df[eqtl_df['variant_id'].str.contains("chrX")].
 ↪sort_values("pval_nominal").reset_index(drop=True).head(5)
for x  in top_5_x.itertuples():
    filename = "top_%d_eqtl_xlinked_%s" % (x.Index, tissue)
    p = annotated_eqtl_plot(x.variant_id, x.gene_id)
    print(filename, x.Index, x.variant_id, x.gene_id)
    print(p)
    save_plot(p, filename)
```

1.3.4 Top 5 eQTL with GWAS significant index SNP

[10]:
```
gwas_eqtl_df = eqtl_df.merge(get_gwas_snps(), left_on = 'variant_id',
                             right_on = 'our_snp_id', suffixes=['','_gwas'])
print(gwas_eqtl_df.shape)
gwas_eqtl_df.head()
```

```
(3, 33)
```

[10]:
```
           variant_id             gene_id           gencodeID     slope  \
0  chr22:42119633:G:C   ENSG00000183172.8   ENSG00000183172.8  0.215764
1   chr6:32140074:G:A  ENSG00000204371.11  ENSG00000204371.11 -0.509024
2   chr6:31867387:G:C   ENSG00000244731.7   ENSG00000244731.7 -0.632753

   statistic  pval_nominal        BF  eigenMT_BH  TESTS  Type  … A2      OR  \
0  10.770408  5.788120e-05  0.024426    0.516399    422  Gene  …  C  1.0629
1  -8.596720  5.852160e-05  0.029436    0.536823    503  Gene  …  A  1.1331
2 -13.250431  1.564220e-07  0.000079    0.123254    502  Gene  …  C  1.1744

         SE             P  hg19chrc hg38chrc   hg38pos  \
0  0.010193  2.130000e-09     chr22    chr22  42119633
1  0.014200  1.370000e-18      chr6     chr6  32140074
2  0.014203  1.050000e-29      chr6     chr6  31867387

  pgc2_a1_same_as_our_counted       rsid  is_index_snp
0                       False  rs8143153         False
1                       False  rs3134962         False
2                       False   rs693906         False

[3 rows x 33 columns]
```

[11]:
```
top_gwas_eqtl_df = gwas_eqtl_df[(gwas_eqtl_df['is_index_snp'])].
↪sort_values(['BF', 'P'])
print(top_gwas_eqtl_df.shape)
top_gwas_eqtl_df.head()
```

```
(0, 33)
```

[11]:
```
Empty DataFrame
Columns: [variant_id, gene_id, gencodeID, slope, statistic, pval_nominal, BF,
eigenMT_BH, TESTS, Type, Tissue, chrN, our_snp_id, cm, pos, our_counted,
our_alt, chrom, SNP, Freq.A1, CHR, BP, A1, A2, OR, SE, P, hg19chrc, hg38chrc,
hg38pos, pgc2_a1_same_as_our_counted, rsid, is_index_snp]
Index: []

[0 rows x 33 columns]
```

[12]:
```
top_gwas_eqtl_df = gwas_eqtl_df.sort_values(['BF', 'P']).reset_index(drop=True)
print(top_gwas_eqtl_df.shape)
top_gwas_eqtl_df.head(10)
```

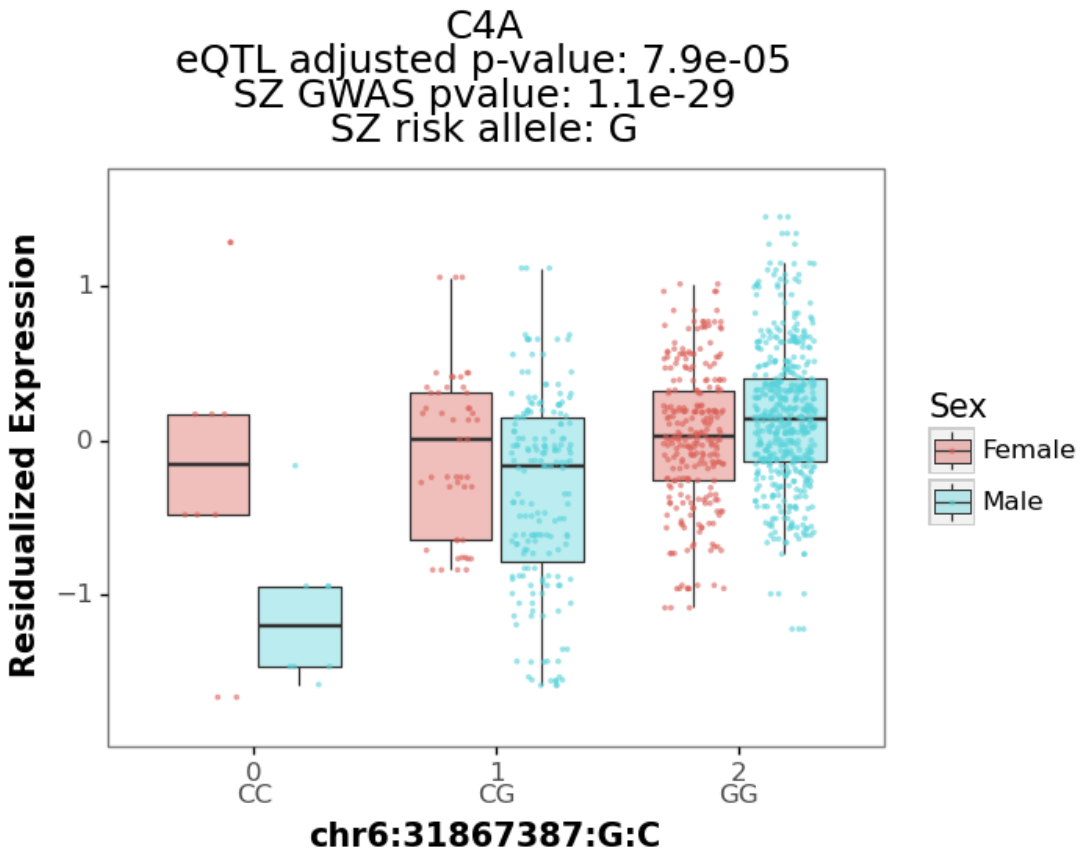
```
(3, 33)
```

[12]:
```
           variant_id             gene_id           gencodeID     slope  \
0   chr6:31867387:G:C   ENSG00000244731.7   ENSG00000244731.7 -0.632753
1  chr22:42119633:G:C   ENSG00000183172.8   ENSG00000183172.8  0.215764
2   chr6:32140074:G:A  ENSG00000204371.11  ENSG00000204371.11 -0.509024
```

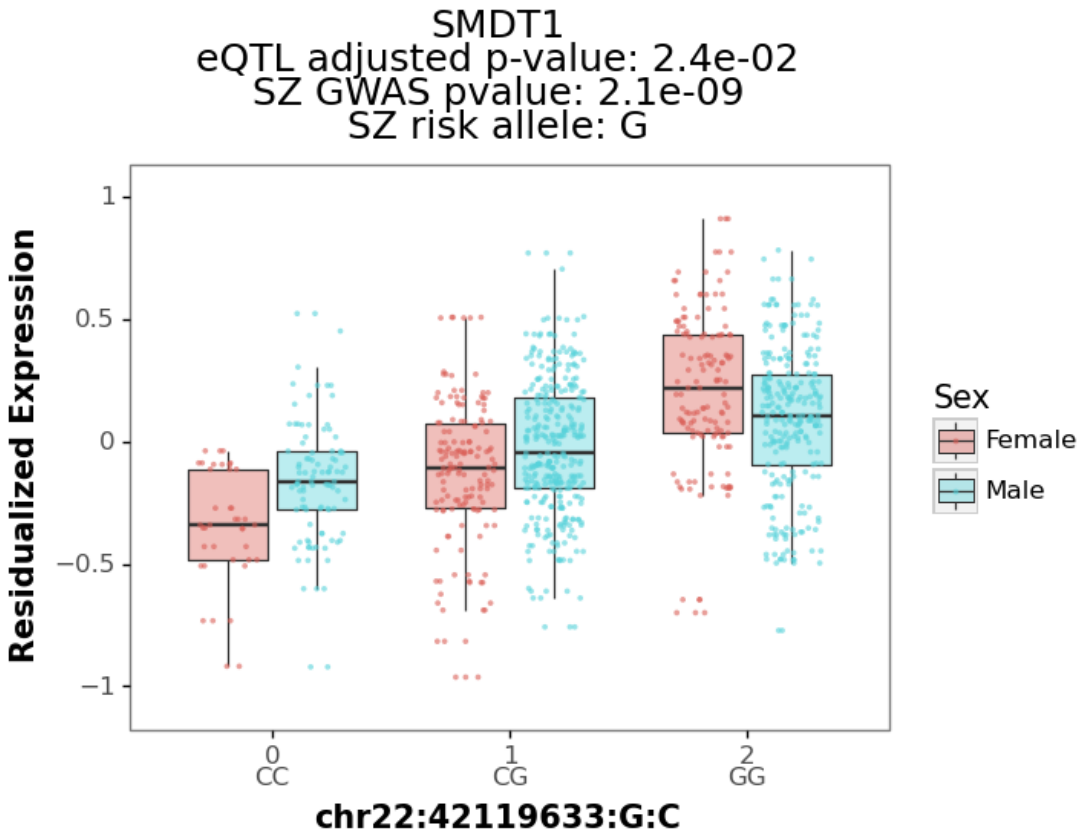
```
   statistic  pval_nominal        BF  eigenMT_BH  TESTS  Type  … A2      OR  \
0 -13.250431  1.564220e-07  0.000079    0.123254    502  Gene  …  C  1.1744
1  10.770408  5.788120e-05  0.024426    0.516399    422  Gene  …  C  1.0629
2  -8.596720  5.852160e-05  0.029436    0.536823    503  Gene  …  A  1.1331

         SE             P  hg19chrc hg38chrc   hg38pos  \
0  0.014203  1.050000e-29      chr6     chr6  31867387
1  0.010193  2.130000e-09     chr22    chr22  42119633
2  0.014200  1.370000e-18      chr6     chr6  32140074

  pgc2_a1_same_as_our_counted       rsid  is_index_snp
0                       False   rs693906         False
1                       False  rs8143153         False
2                       False  rs3134962         False

[3 rows x 33 columns]
```

[13]:
```
top_5_gwas = top_gwas_eqtl_df.head(5)
for x  in top_5_gwas.itertuples():
    filename = "top_%d_eqtl_in_gwas_significant_snps_%s" % (x.Index, tissue)
    p = gwas_annotated_eqtl_plot(x.variant_id, x.gene_id)
    print(filename, x.Index, x.variant_id, x.gene_id)
    print(p)
    save_plot(p, filename)
```

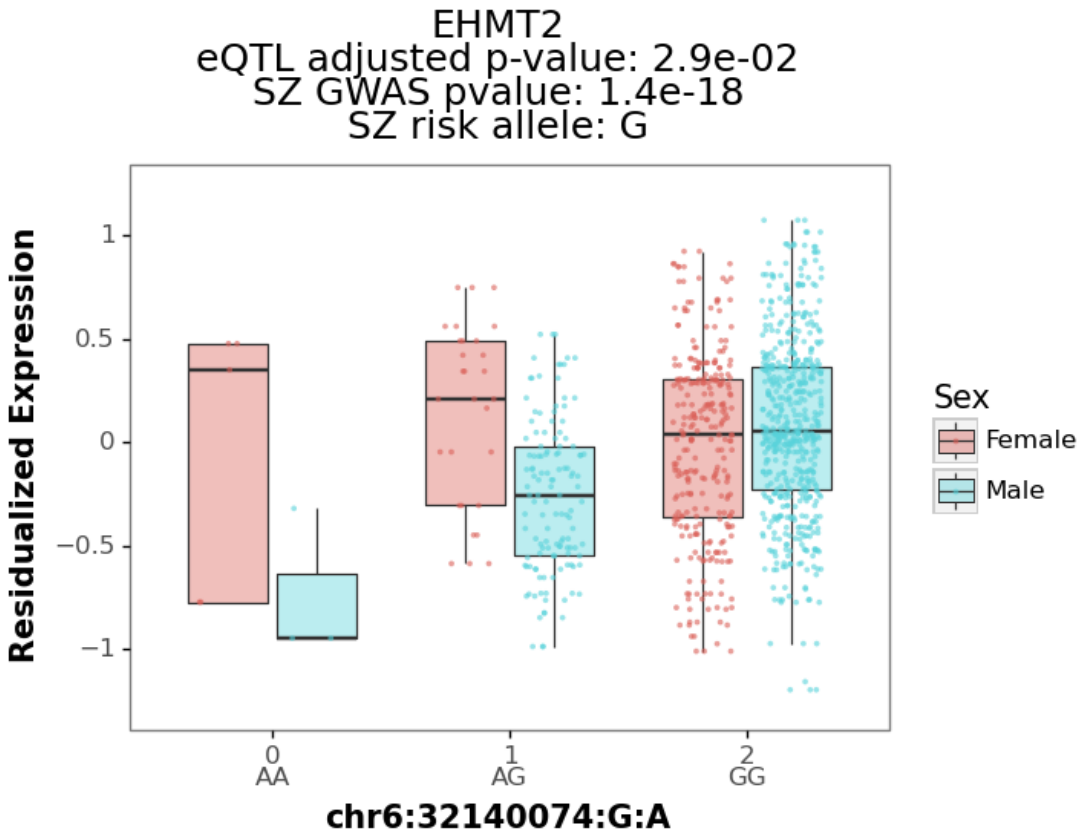
```
top_0_eqtl_in_gwas_significant_snps_caudate 0 chr6:31867387:G:C
ENSG00000244731.7
```




```
<ggplot: (8728062453709)>
top_1_eqtl_in_gwas_significant_snps_caudate 1 chr22:42119633:G:C
ENSG00000183172.8
```




```
<ggplot: (8728184152691)>
top_2_eqtl_in_gwas_significant_snps_caudate 2 chr6:32140074:G:A
ENSG00000204371.11
```




```
<ggplot: (8728062942880)>
```

```
[ ]:
```