main_junctions

September 2, 2021

1 Enrichment and Overlap of PGC2+CLOZUK

[1]:
```
import re
import os, errno
import functools
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
from scipy.stats import fisher_exact, binom_test

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

1.1 Config and Functions

[2]:
```
config = {
    'biomart_file': '../_h/biomart.csv',
    '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',
    'gwas_snp_file': '/ceph/projects/v4_phase3_paper/inputs/sz_gwas/pgc2_clozuk/
 ↪map_phase3/_m/libd_hg38_pgc2sz_snps.tsv'
}

config_feature = {
    'de_file': '../../differential_expression/_m/junctions/diffExpr_szVctl_full.
 ↪txt',
    'residual_expression_file': '../../differential_expression/_m/junctions/
 ↪residualized_expression.tsv',
    'fastqtl_output_file': '../../eqtl/caudate/summary_table/_m/
 ↪Brainseq_LIBD_caudate_4features.signifpairs.txt.gz',
}
```

```
feature = "junctions"
```

[3]:
```
@functools.lru_cache()
def feature_map(feature):
    return {"genes": "Gene", "transcripts": "Transcript",
            "exons": "Exon", "junctions": "Junction"}[feature]


@functools.lru_cache()
def get_de_df():
    """
    Load DE analysis
    """
    return pd.read_csv(config_feature['de_file'], sep='\t', index_col=0)


@functools.lru_cache()
def get_eqtl_df():
    eqtl_df = pd.read_csv(config_feature['fastqtl_output_file'], sep='\t')
    return eqtl_df[(eqtl_df["Type"] == feature_map(feature))]


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


@functools.lru_cache()
def get_integration_df():
    return get_gwas_snps().merge(get_eqtl_df(), left_on='our_snp_id',␣
 ↪right_on='variant_id',
                                 suffixes=['_PGC2', '_eQTL'])\
                        .merge(get_de_df(), left_on='gene_id',␣
 ↪right_index=True)


@functools.lru_cache()
def get_residual_expression_df():
    return pd.read_csv(config_feature['residual_expression_file'],
                       sep='\t', index_col=0).transpose()


@functools.lru_cache()
def get_pheno_df():
    return pd.read_csv(config['phenotype_file'], index_col=0)
```

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

[4]:
```
def agree_direction(row):
    return [-1, 1][row['pgc2_a1_same_as_our_counted']] * np.sign(row['OR'] - 1)␣
 ↪* np.sign(row['slope']) * np.sign(row['t'])


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


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))))


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

[5]:
```
@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_plink_tuple():
    '''
    Usage: (bim, fam, bed) = get_plink_tuple()
    '''
    return read_plink(config['plink_file_prefix'])


@functools.lru_cache()
def subset_bed():
    """
    This subsets the bed and bim file and returns the new subsetted
    data with shared brain_ids.

    This is to speed up accessing the bed file.
    """
    (bim, fam, bed) = get_plink_tuple()
    brain_ids = list(set(get_expression_and_pheno_df()['BrNum']).
 ↪intersection(set(fam['fid'])))
    fam_pos = list(fam[(fam["fid"].isin(brain_ids))].
 ↪drop_duplicates(subset="fid").loc[:, 'i'])
    unique_snps = get_eqtl_df().variant_id.unique()
    snp_info = bim[(bim["snp"].isin(unique_snps))].copy()
    snp_pos = list(snp_info.loc[:, "i"])
    new_bed = bed[snp_pos].compute()[:,fam_pos]
    new_bim = bim[(bim["i"].isin(snp_pos))].reset_index(drop=True)
    new_bim['ii'] = new_bim.index
    return new_bed, new_bim, brain_ids


@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
    '''
```

```
    bed, bim, brain_ids = subset_bed()
    snp_info = bim[bim['snp']==snp_id]
    snp_pos = snp_info.iloc[0]['ii']
    dfsnp = pd.DataFrame(bed[[snp_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'])
    # the 2 - in next line is to workaround a possible bug in pandas_plink? a1␣
↪and a0 inverted
    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
```

[6]:
```
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


@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
```

[7]:
```
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_on='BrNum', right_index=True)


def simple_snp_expression_pheno_plot_impl(snp_id, gene_id, snp_df_func,␣
 ↪pheno_var):
    df = get_snp_gene_pheno_df(snp_id, gene_id, snp_df_func)
    df['Dx'] = df.Dx.astype('category').cat.rename_categories({'Control':␣
 ↪'CTL', 'Schizo': 'SZ'})
    y0 = df[gene_id].quantile(.01) - 0.26
    y1 = df[gene_id].quantile(.99) + 0.26
    pjd = position_jitterdodge(jitter_width=0.27)
    p = ggplot(df, aes(x=snp_id, y=gene_id, fill=pheno_var)) \
    + geom_boxplot(alpha=0.4, outlier_alpha=0) \
```

```
    + geom_jitter(position=pjd, stroke=0, alpha=0.6) + ylim(y0, y1) \
    + labs(y='Residualized expression', fill='Diagnosis') \
    + theme_bw(base_size=20)\
    + theme(legend_title=element_text(face='bold'),
            panel_grid_major=element_blank(),
            panel_grid_minor=element_blank())
    return p


def simple_gwas_ordered_snp_expression_pheno_plot(snp_id, gene_id, pheno_var):
    return simple_snp_expression_pheno_plot_impl(snp_id, gene_id,␣
 ↪get_gwas_ordered_snp_df, pheno_var)
```

[8]:
```
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):
    r = get_eqtl_df()[(get_eqtl_df()['variant_id']==snp_id) &
                      (get_eqtl_df()['gene_id']==gene_id)]
    assert len(r)==1
    return 'eQTL nominal p-value: %.1e' % r.iloc[0]['pval_nominal']


def de_annotation(gene_id):
    de_df = get_de_df()
    de_df['Feature'] = de_df.index
    g = de_df[(de_df['Feature'] == gene_id)]
    return 'DE adj.P.Val: %.3f' % g.iloc[0]['adj.P.Val']


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


def gwas_annotated_eqtl_pheno_plot(snp_id, gene_id, pheno_var):
    p = simple_gwas_ordered_snp_expression_pheno_plot(snp_id, gene_id,␣
 ↪pheno_var)
    de_df = get_de_df()
    de_df['Feature'] = de_df.index
    de_df = de_df[(de_df['Feature'] == gene_id)]
    gene_symbol = de_df.iloc[0]['newGeneSymbol']

    title ="\n".join([gene_symbol,
                      gene_id,
                      gwas_annotation(snp_id),
                      risk_allele_annotation(snp_id),
```

```
                    eqtl_annotation(snp_id, gene_id),
                    de_annotation(gene_id)])

    p += ggtitle(title)
    return p
```

1.2 Junctions

[9]:
```
try:
    os.makedirs(feature)
except OSError as e:
    if e.errno != errno.EEXIST:
        raise
```

1.2.1 Enrichment

Integrate DEG with PGC2+CLOZUK SNPs

[10]:
```
dft = get_integration_df()
dft.shape
```

```
/home/jbenja13/.local/lib/python3.9/site-packages/numpy/lib/arraysetops.py:583:
FutureWarning: elementwise comparison failed; returning scalar instead, but in
the future will perform elementwise comparison
```

[10]:
```
(3190613, 57)
```

[11]:
```
agreement = {1: 'Yes', -1: 'No', 0: 0}
dft['agree_direction'] = dft.apply(agree_direction, axis=1)
dft.agree_direction = [agreement[item] for item in dft['agree_direction']]

table =  [[np.sum((dft['P']<5e-8) & ((dft['adj.P.Val']<.05))),
           np.sum((dft['P']<5e-8) & ((dft['adj.P.Val']>=.05)))],
          [np.sum((dft['P']>=5e-8) & ((dft['adj.P.Val']<.05))),
           np.sum((dft['P']>=5e-8) & ((dft['adj.P.Val']>=.05)))]]
print(table)
fisher_exact(table)
```

```
[[5196, 74475], [122145, 2988797]]
```

[11]:
```
(1.7071801613298905, 4.8917286718013116e-254)
```

[12]:
```
dft1 = dft[(dft['P']<5e-8) & ((dft['adj.P.Val']<.05))]
df = dft1.groupby('agree_direction').size().reset_index()
df
```

[12]:

	agree_direction	0
0	No	295
1	Yes	4901

[13]:
```
binom_test(df[0].iloc[1], df[0].sum())
```

[13]:
```
0.0
```

[14]:
```
dft2 = dft[(dft['P']<=5e-8) & (dft['adj.P.Val'] < 0.05)].copy()
dft2['risk_allele'] = dft2['our_snp_id'].apply(get_risk_allele)
```

[15]:
```
direction = {-1: 'Down', 1: 'Up'}
boolean_conv = {True: 1, False: -1}
dft2.pgc2_a1_same_as_our_counted = [boolean_conv[item] for item in␣
↪dft2['pgc2_a1_same_as_our_counted']]
dft2['eqtl_gwas_dir'] = [direction[item] for item in np.
↪sign(dft2['pgc2_a1_same_as_our_counted']) * np.sign(dft2['slope']) * np.
↪sign(dft2['OR'] - 1)]
dft2['de_dir'] = [direction[item] for item in np.sign(dft2['t'])]
dft2['eqtl_slope'] = np.sign(dft2['pgc2_a1_same_as_our_counted']) * np.
↪sign(dft2['OR'] - 1) * dft2['slope']
dft2 = dft2[['gene_id', 'newGeneSymbol', 'variant_id', 'A1', 'A2',␣
↪'risk_allele', 'OR',
            'P', 'pval_nominal', 'adj.P.Val', 'logFC', 't', 'eqtl_slope',
            'de_dir', 'eqtl_gwas_dir', 'agree_direction']].
↪rename(columns={"newGeneSymbol": "Symbol"})
dft2['Symbol'].fillna(dft2['gene_id'], inplace=True)
dft2.to_csv('%s/integration_by_symbol.txt' % feature, sep='\t', index=False)
```

[16]:
```
df2 = dft2.groupby(['Symbol']).first().reset_index().sort_values('P')
df2.groupby('agree_direction').size()
```

[16]:
```
agree_direction
No     10
Yes    11
dtype: int64
```

[17]:
```
df2.set_index('Symbol').rename(columns={'t': 'de_t', 'P': 'GWAS_P',␣
↪'pval_nominal': 'eQTL_pvalue',
                                        'adj.P.Val': 'de_FDR'})
```

[17]:

Symbol	gene_id	variant_id	A1	A2	\
HLA-C	chr6:31270086-31270209(-)	chr6:30770669:G:A	G	A	
C4B	chr6:32024325-32024469(+)	chr6:31530467:G:GC	G	GC	
ZSCAN26	chr6:28271935-28272021(+)	chr6:27772521:C:A	C	A	
ZFYVE21	chr14:103729183-103729792(+)	chr14:103633871:T:A	T	A	
CKB	chr14:103520312-103520468(-)	chr14:103710761:G:T	G	T	
BAG6	chr6:31643978-31644081(-)	chr6:31610995:T:C	T	C	
NGEF	chr2:232888108-232891357(-)	chr2:232926898:C:G	C	G	
PLCH2	chr1:2497011-2497501(+)	chr1:2440958:A:G	A	G	

HLA-DPB1	chr6:33080535-33080671(+)	chr6:33020111:A:G	A	G
DRD2	chr11:113412884-113415420(-)	chr11:113447023:G:C	G	C
ME1	chr6:83315414-83346172(-)	chr6:83463743:T:C	T	C
PACSIN3	chr11:47180691-47182402(-)	chr11:46729945:T:TAGG	T	TAGG
IP6K3	chr6:33728301-33735277(-)	chr6:33741539:G:A	G	A
PCCB	chr3:136262066-136283836(+)	chr3:136088767:G:T	G	T
NDFIP2	chr13:79551123-79552515(+)	chr13:79285321:A:C	A	C
NDRG4	chr16:58510684-58511421(+)	chr16:58505281:G:A	G	A
RERE	chr1:8359987-8360111(-)	chr1:8361143:T:C	T	C
MPP6	chr7:24623785-24641715(+)	chr7:24737470:CA:C	CA	C
REEP2	chr5:138445599-138445682(+)	chr5:138362610:C:T	C	T
C4A	chr6:31995795-31995954(+)	chr6:31495800:T:C	T	C
VRK2	chr2:58048968-58084088(+)	chr2:57867738:A:C	A	C

Symbol	risk_allele	OR	GWAS_P	eQTL_pvalue	de_FDR	\
HLA-C	G	1.218800	6.250000e-30	1.394910e-08	0.037406	
C4B	G	1.177150	1.610000e-26	9.700900e-05	0.046496	
ZSCAN26	C	1.104300	7.130000e-18	1.385660e-04	0.018932	
ZFYVE21	A	0.928260	1.540000e-13	4.963540e-05	0.002033	
CKB	T	0.928840	3.280000e-13	8.314590e-05	0.002976	
BAG6	C	0.927470	6.240000e-13	4.199930e-06	0.009575	
NGEF	G	0.934740	1.270000e-12	1.620400e-05	0.009518	
PLCH2	G	0.928730	4.630000e-11	2.336380e-05	0.006178	
HLA-DPB1	A	1.160000	2.230000e-10	6.643900e-06	0.040099	
DRD2	G	1.066000	4.300000e-10	1.110170e-05	0.004208	
ME1	C	0.940970	3.300000e-09	7.144370e-05	0.006069	
PACSIN3	TAGG	0.925705	3.875000e-09	2.207740e-04	0.022512	
IP6K3	G	1.076700	4.780000e-09	3.625090e-06	0.004709	
PCCB	T	0.944930	5.050000e-09	2.123870e-07	0.020973	
NDFIP2	A	1.057600	1.150000e-08	5.330420e-05	0.028837	
NDRG4	A	0.942820	1.240000e-08	6.879910e-13	0.027942	
RERE	T	1.059600	1.320000e-08	1.276590e-05	0.020434	
MPP6	C	0.931555	1.631000e-08	1.519980e-05	0.014432	
REEP2	C	1.056000	1.860000e-08	9.262890e-06	0.000011	
C4A	T	1.060300	1.890000e-08	2.176790e-07	0.031770	
VRK2	C	0.946730	2.360000e-08	4.121830e-06	0.042204	

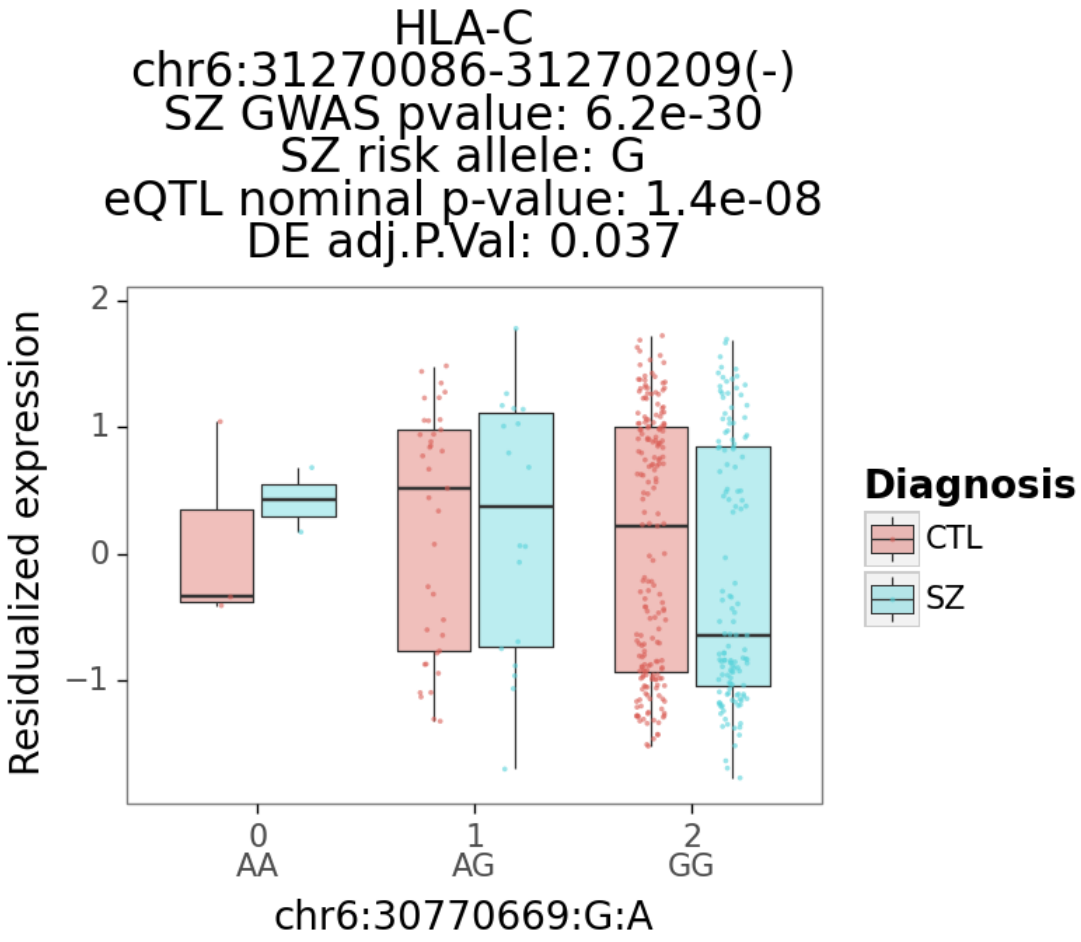
Symbol	logFC	de_t	eqtl_slope	de_dir	eqtl_gwas_dir	agree_direction
HLA-C	-1.434498	-3.191459	-0.621059	Down	Down	Yes
C4B	0.273063	3.088621	0.195040	Up	Up	Yes
ZSCAN26	-0.261770	-3.493012	-0.258408	Down	Down	Yes
ZFYVE21	0.137121	4.331524	0.245245	Up	Up	Yes
CKB	-0.094478	-4.201159	-0.136348	Down	Down	Yes
BAG6	-0.111484	-3.769427	0.281576	Down	Up	No
NGEF	0.115109	3.772815	-0.117766	Up	Down	No

PLCH2	-0.254239	-3.941689	-0.228091	Down	Down	Yes
HLA-DPB1	0.452891	3.159284	0.507916	Up	Up	Yes
DRD2	0.302443	4.079084	-0.260710	Up	Down	No
ME1	0.114156	3.949190	-0.189418	Up	Down	No
PACSIN3	-0.186281	-3.419199	0.208793	Down	Up	No
IP6K3	-0.370340	-4.042487	0.250279	Down	Up	No
PCCB	-0.105287	-3.448890	-0.253684	Down	Down	Yes
NDFIP2	0.081578	3.311700	-0.166771	Up	Down	No
NDRG4	0.171047	3.324829	-0.336307	Up	Down	No
RERE	-0.207443	-3.460156	0.187197	Down	Up	No
MPP6	0.079785	3.603734	0.154819	Up	Up	Yes
REEP2	0.141226	5.772797	0.154309	Up	Up	Yes
C4A	0.349910	3.265208	0.238766	Up	Up	Yes
VRK2	-0.148130	-3.135273	0.179123	Down	Up	No

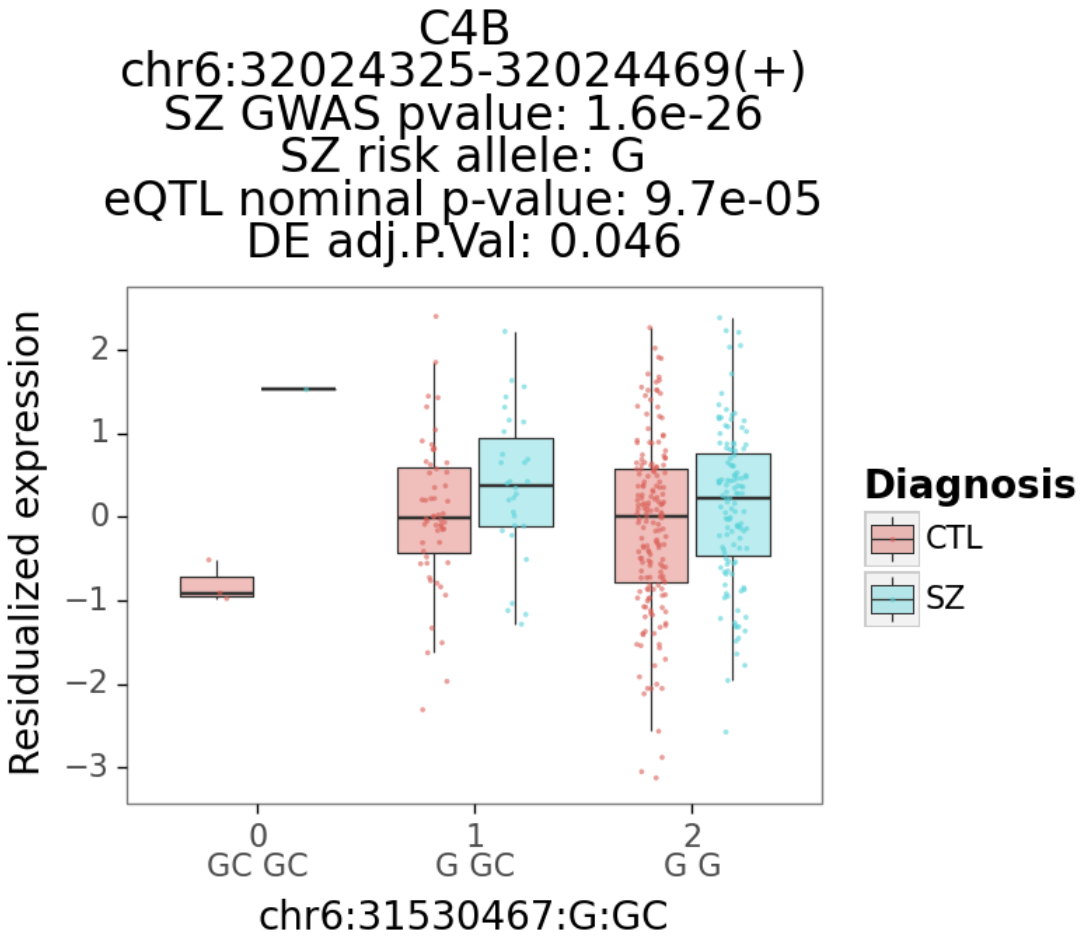
1.2.2 Plot with PGC2 risk allele

[18]:
```
for xx in range(df2.shape[0]):
    gg = gwas_annotated_eqtl_pheno_plot(df2.iloc[xx, :].variant_id,
                                        df2.iloc[xx, :].gene_id, 'Dx')
    print(gg)
    label = '%s/eqtl_gwas_%s' % (feature, df2.iloc[xx, :].Symbol)
    save_plot(gg, label)
```

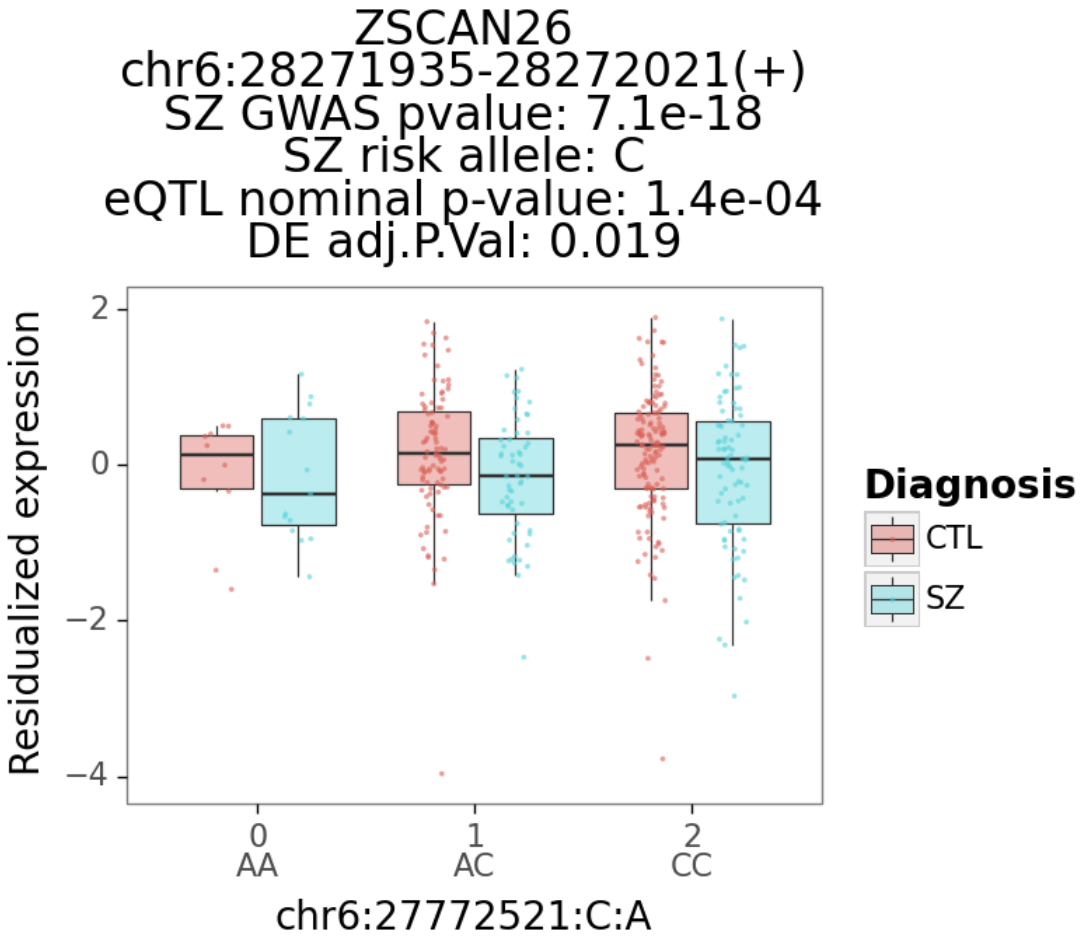
```
Mapping files: 100%|      | 3/3 [00:28<00:00,  9.42s/it]
```



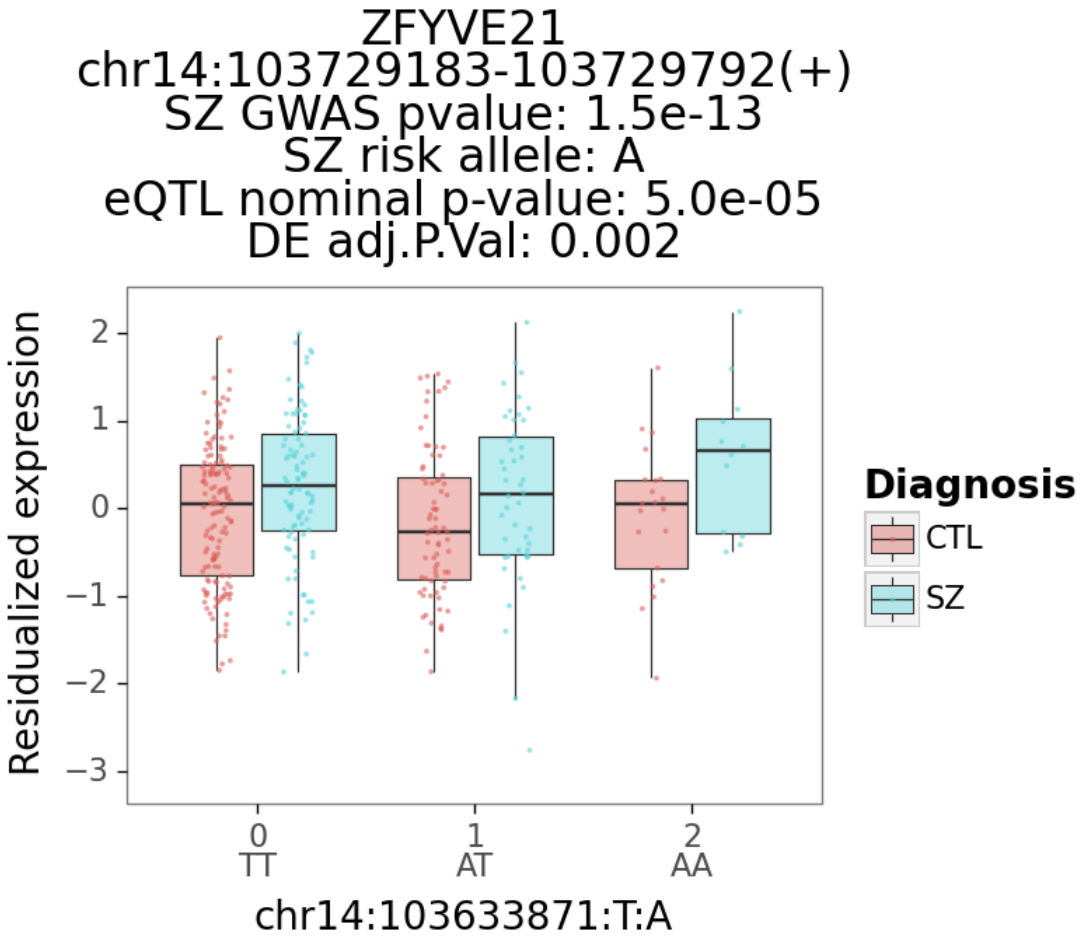

```
<ggplot: (8752693421011)>
```



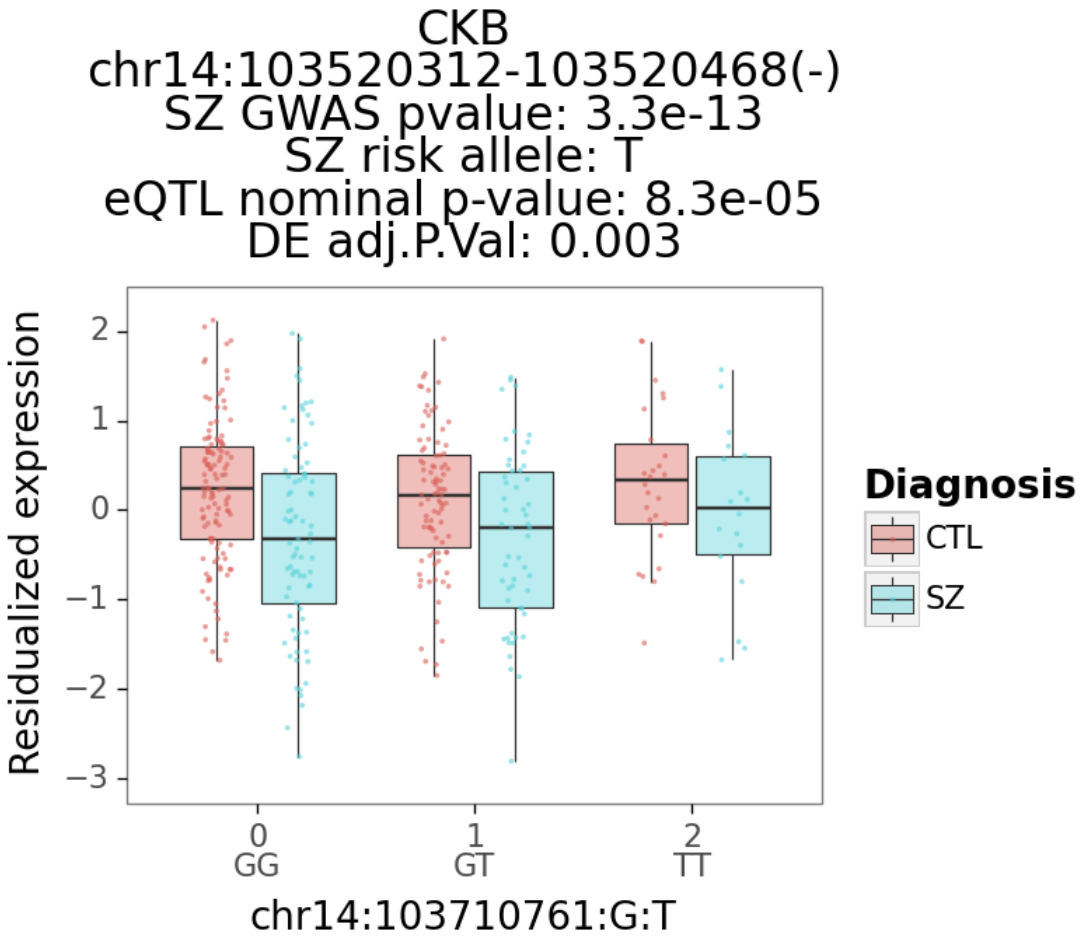

```
<ggplot: (8752694461880)>
```



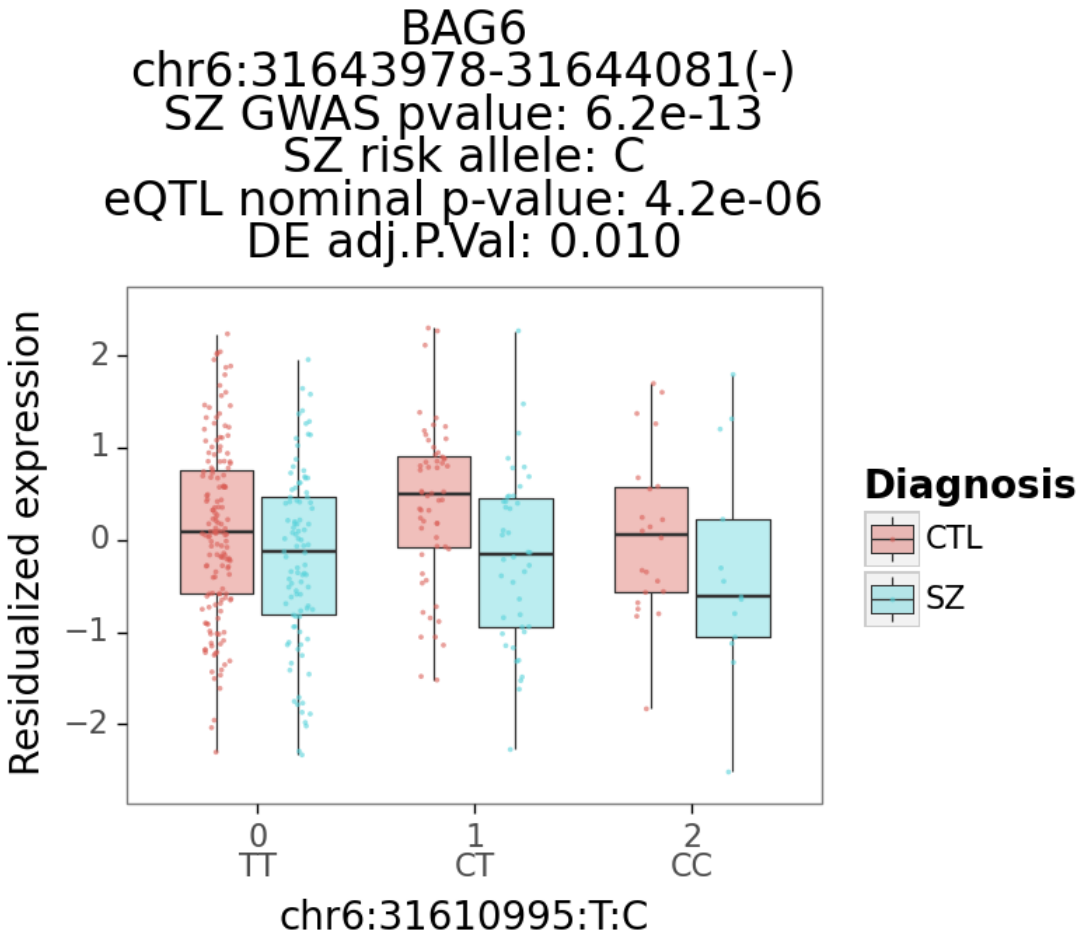

```
<ggplot: (8752694038673)>
```



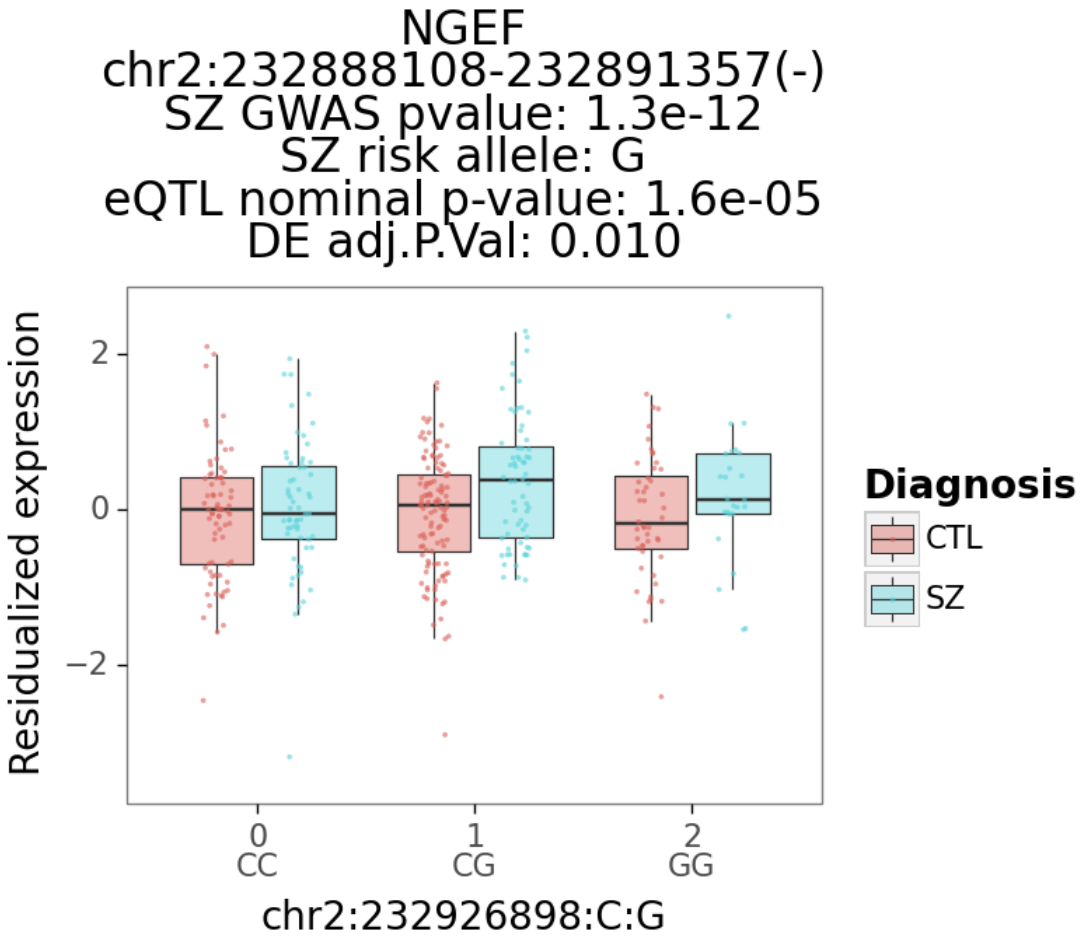

```
<ggplot: (8752699150942)>
```



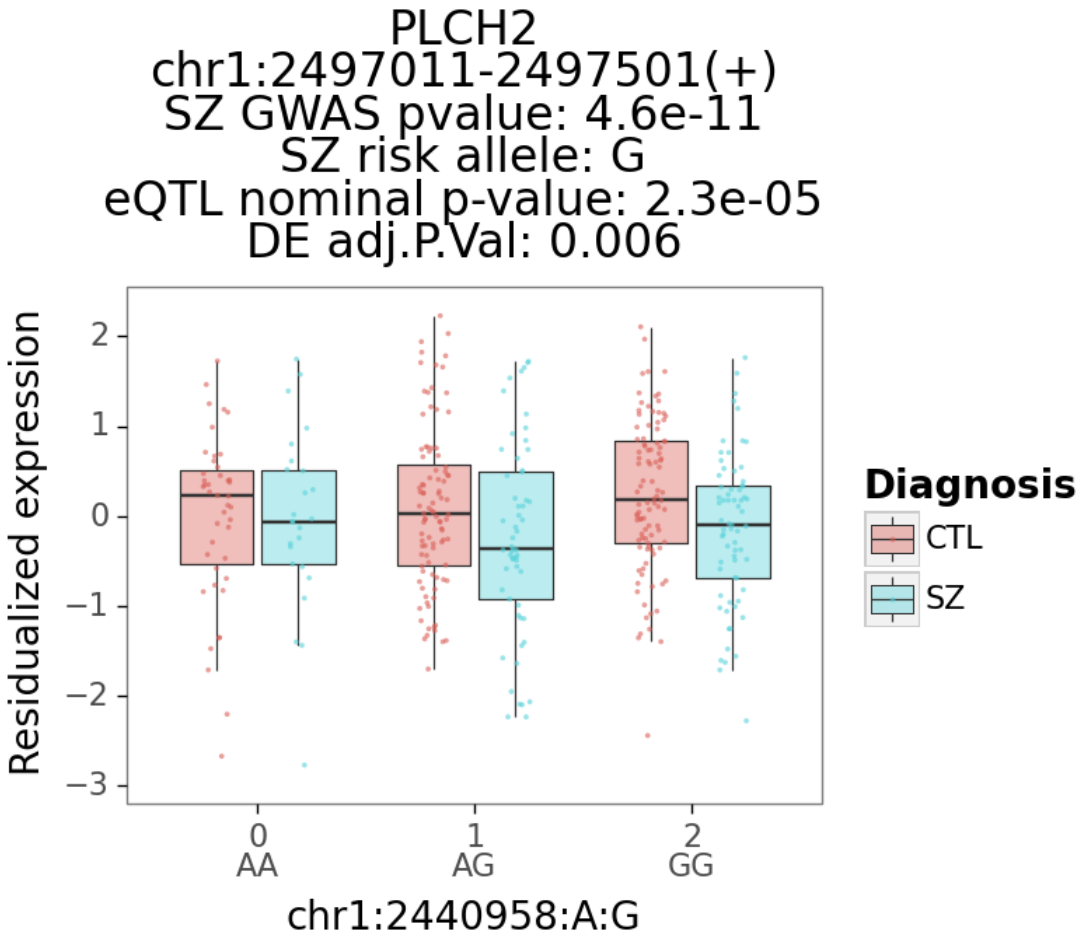

```
<ggplot: (8752693338700)>
```



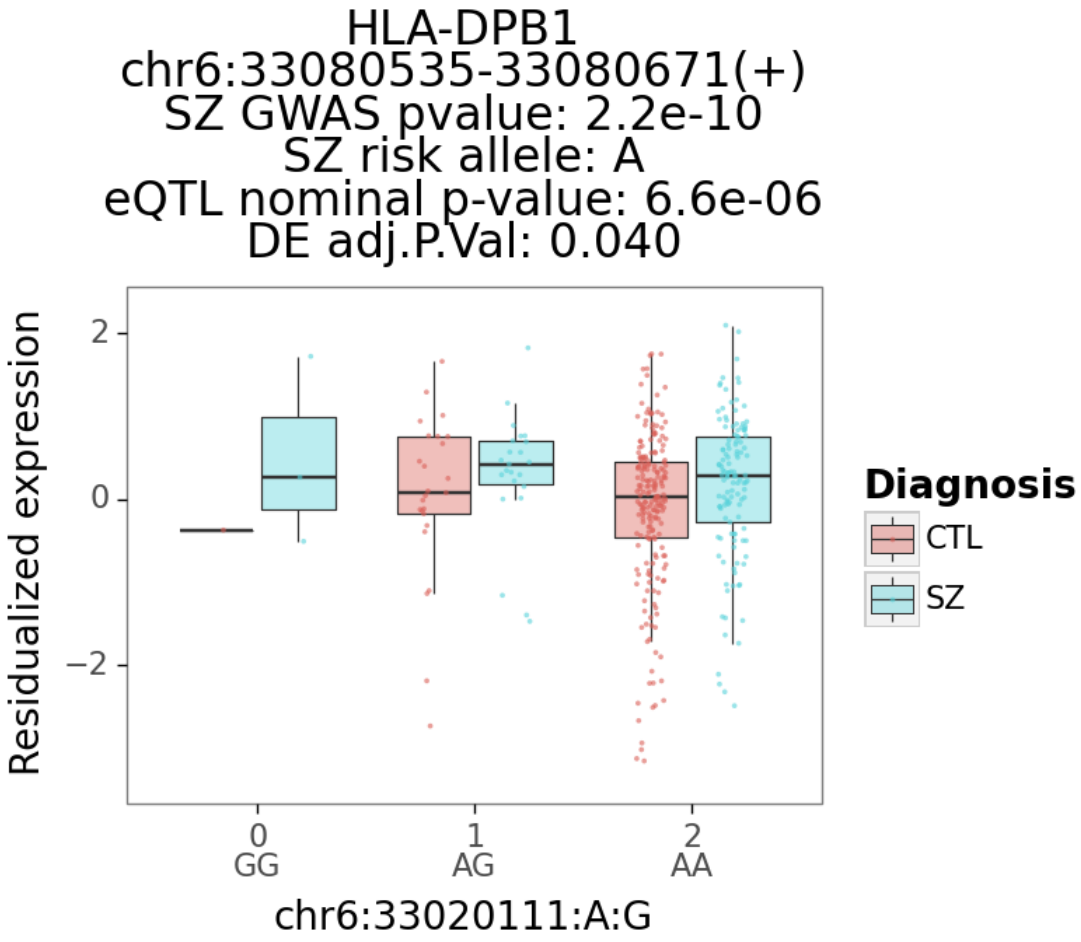

```
<ggplot: (8752700304969)>
```



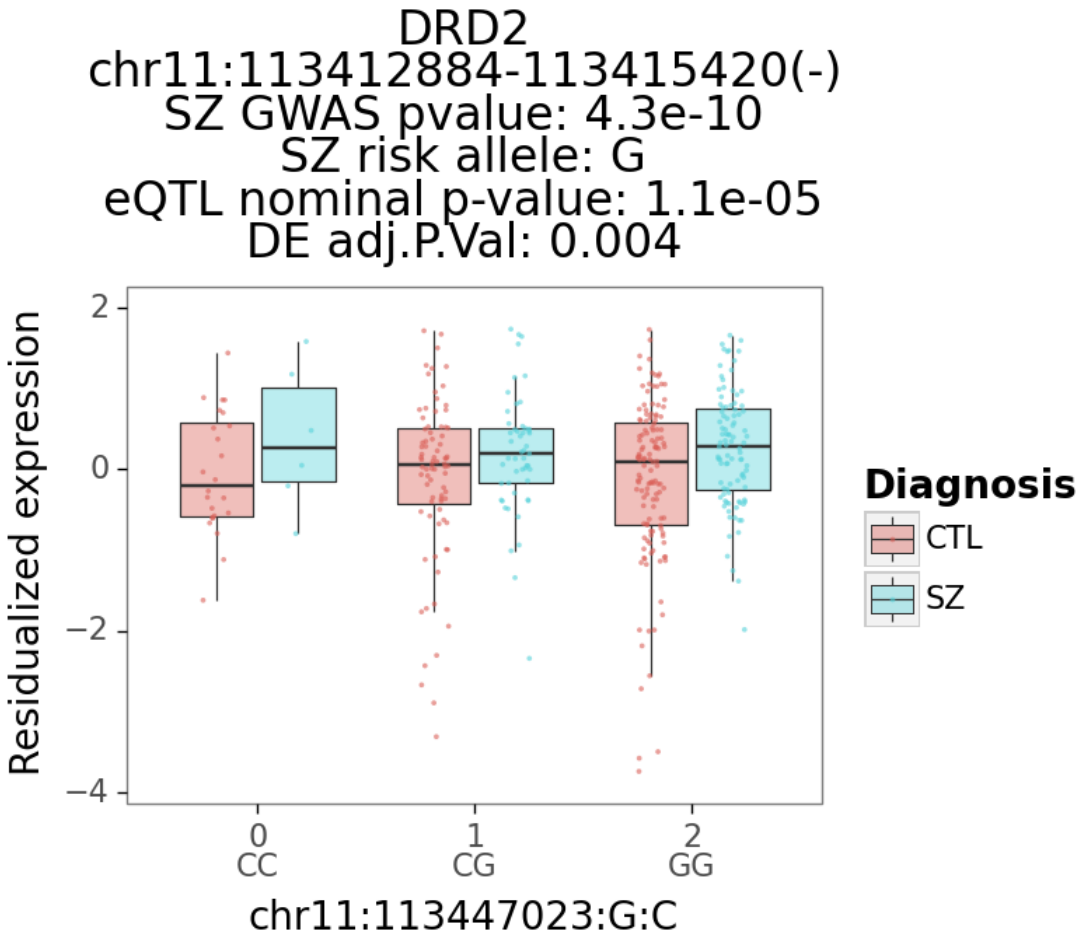

```
<ggplot: (8752691733613)>
```



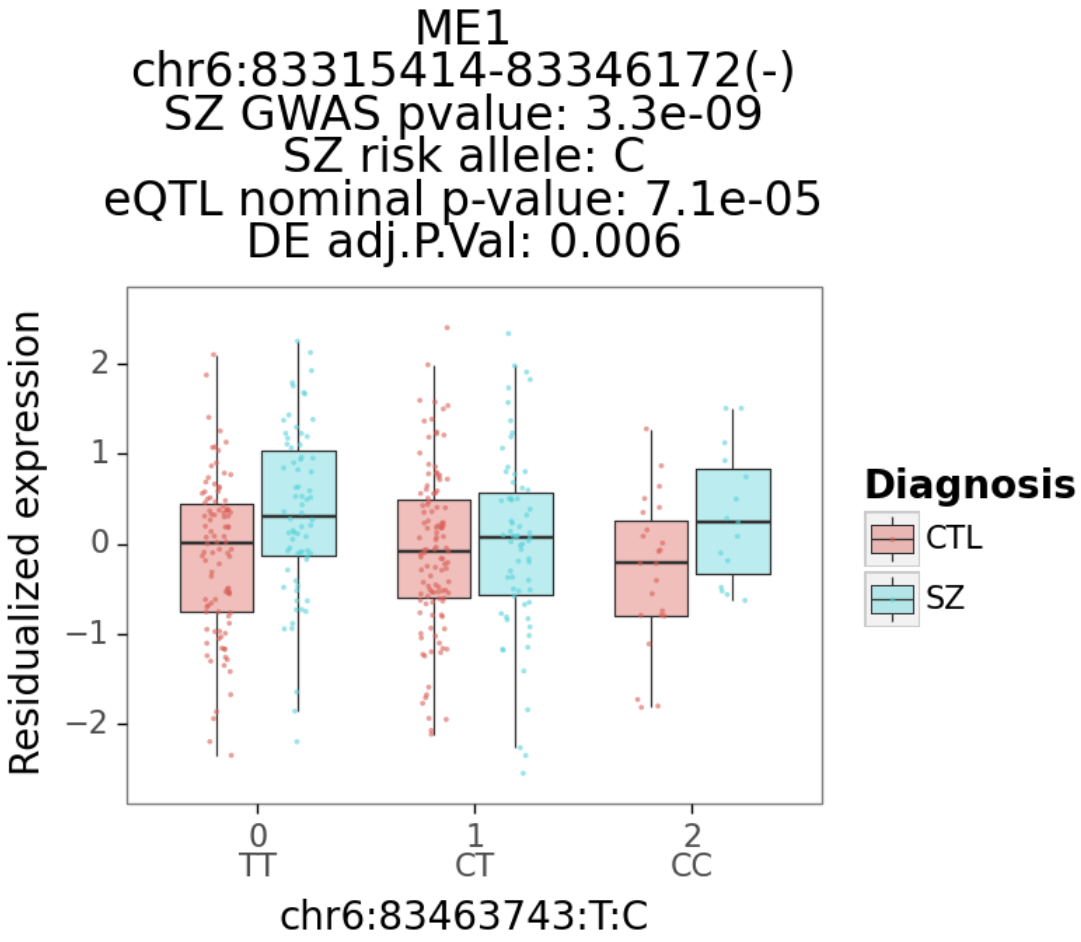

```
<ggplot: (8752691724837)>
```



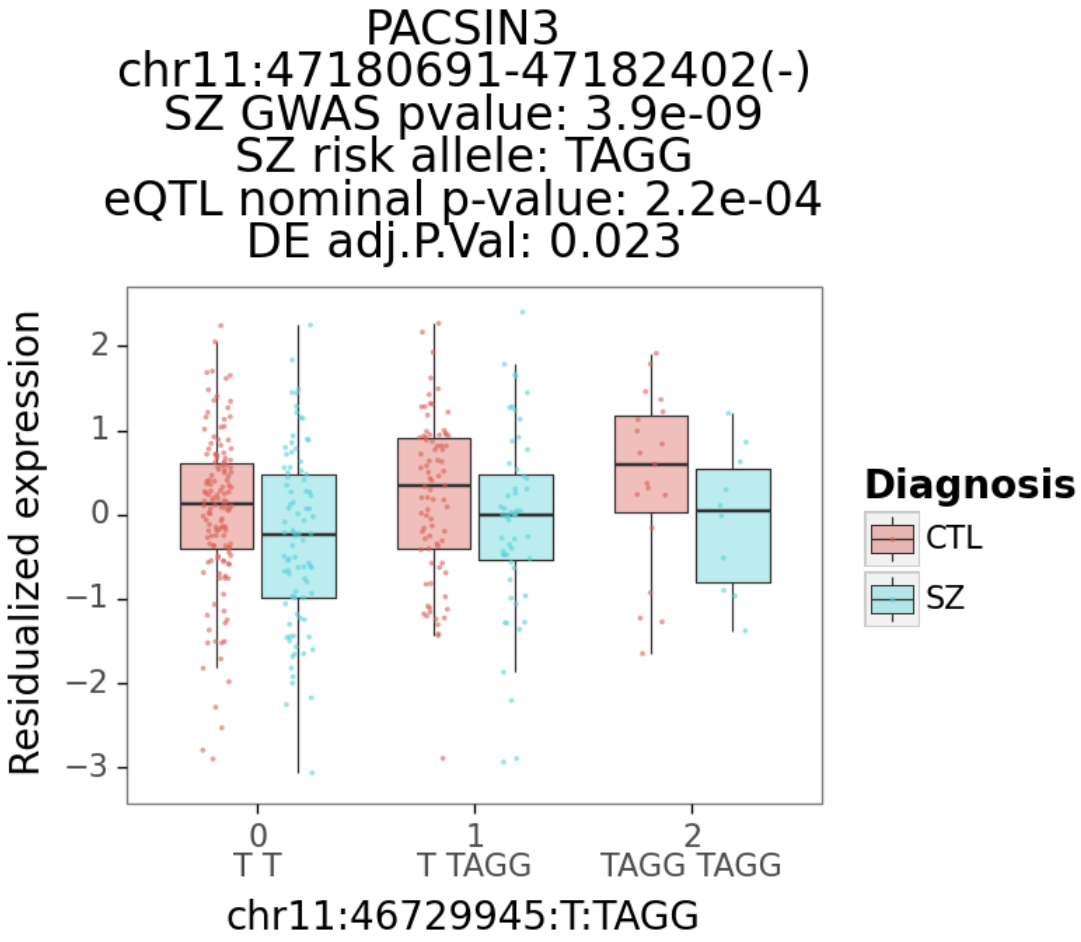

```
<ggplot: (8752693111953)>
```



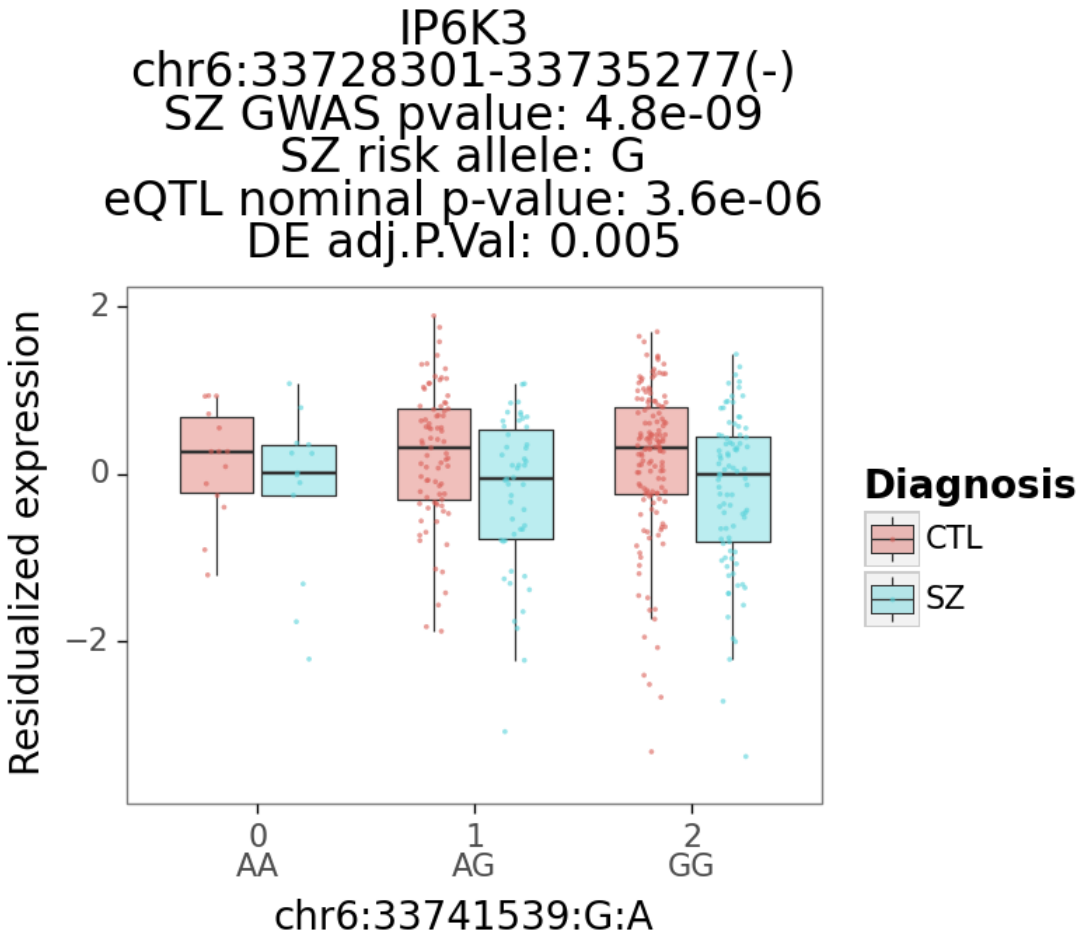

```
<ggplot: (8752699508809)>
```



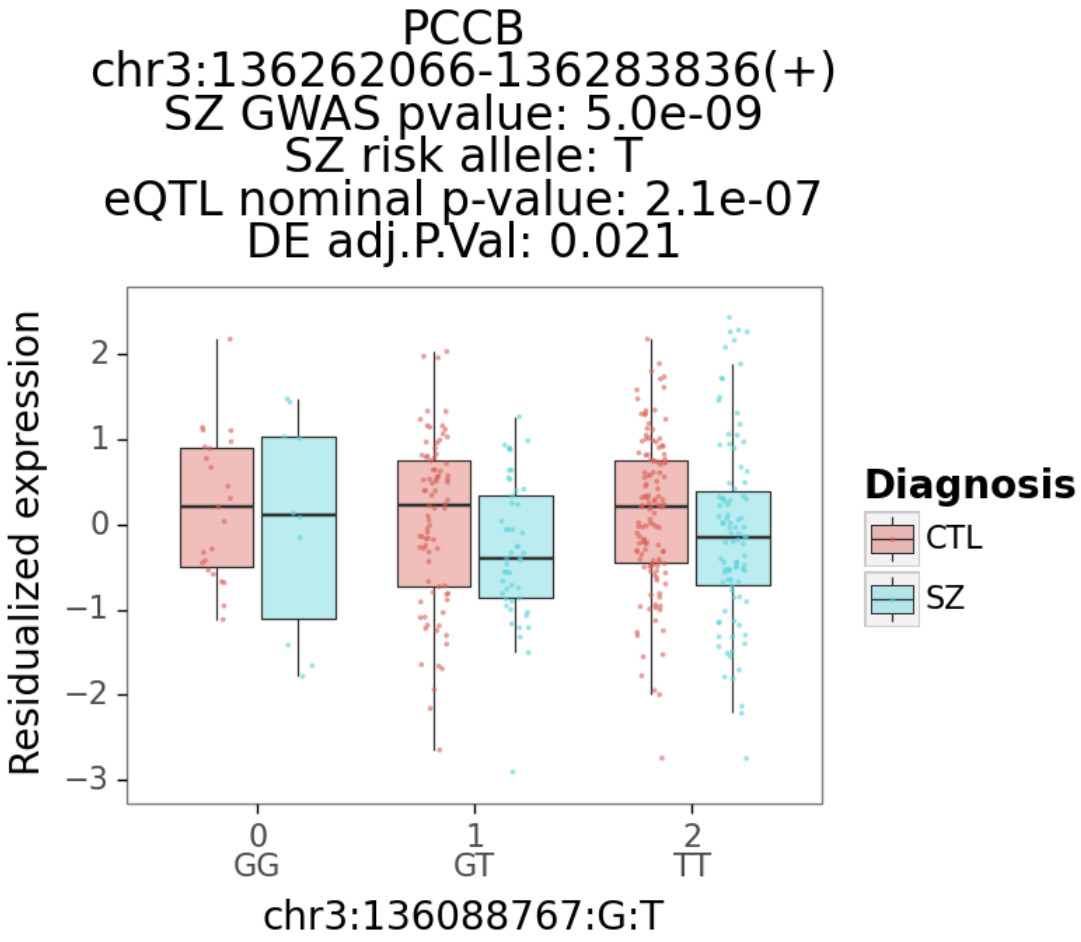

```
<ggplot: (8752700657565)>
```



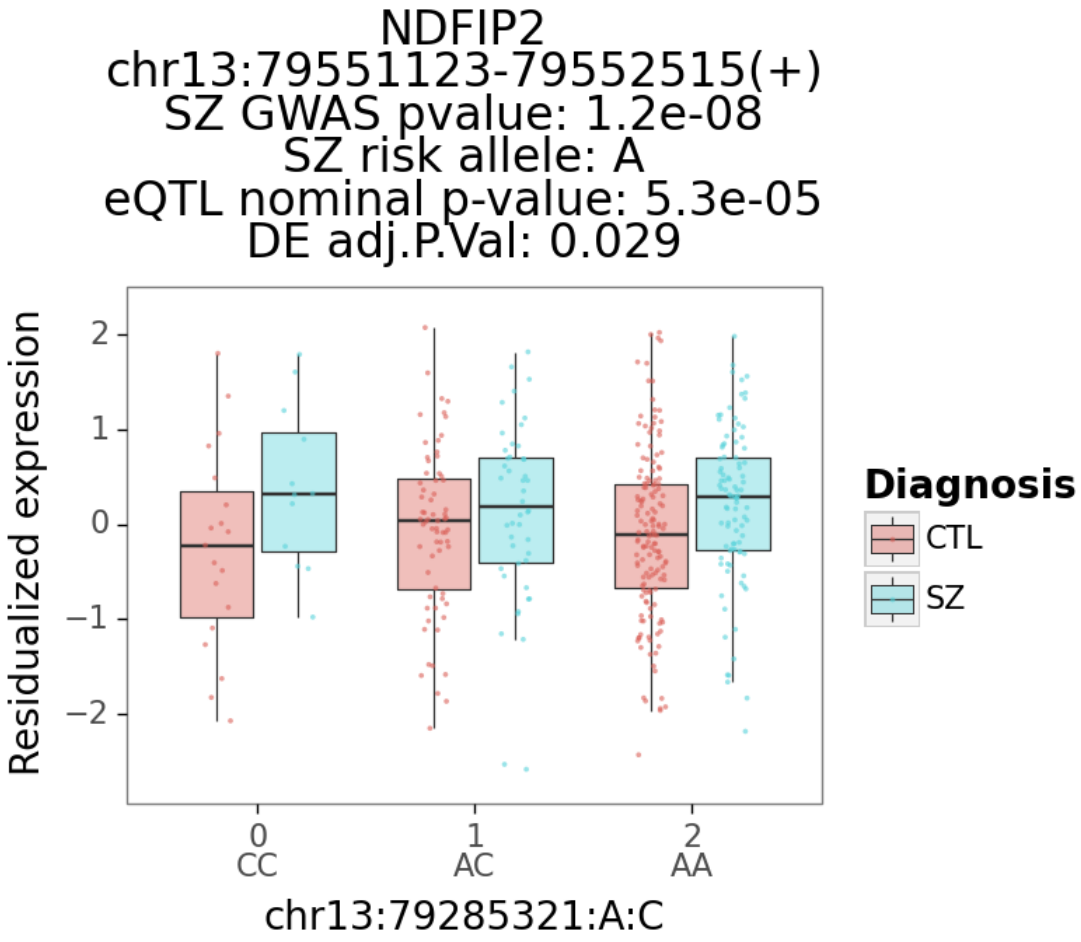

```
<ggplot: (8752698978735)>
```



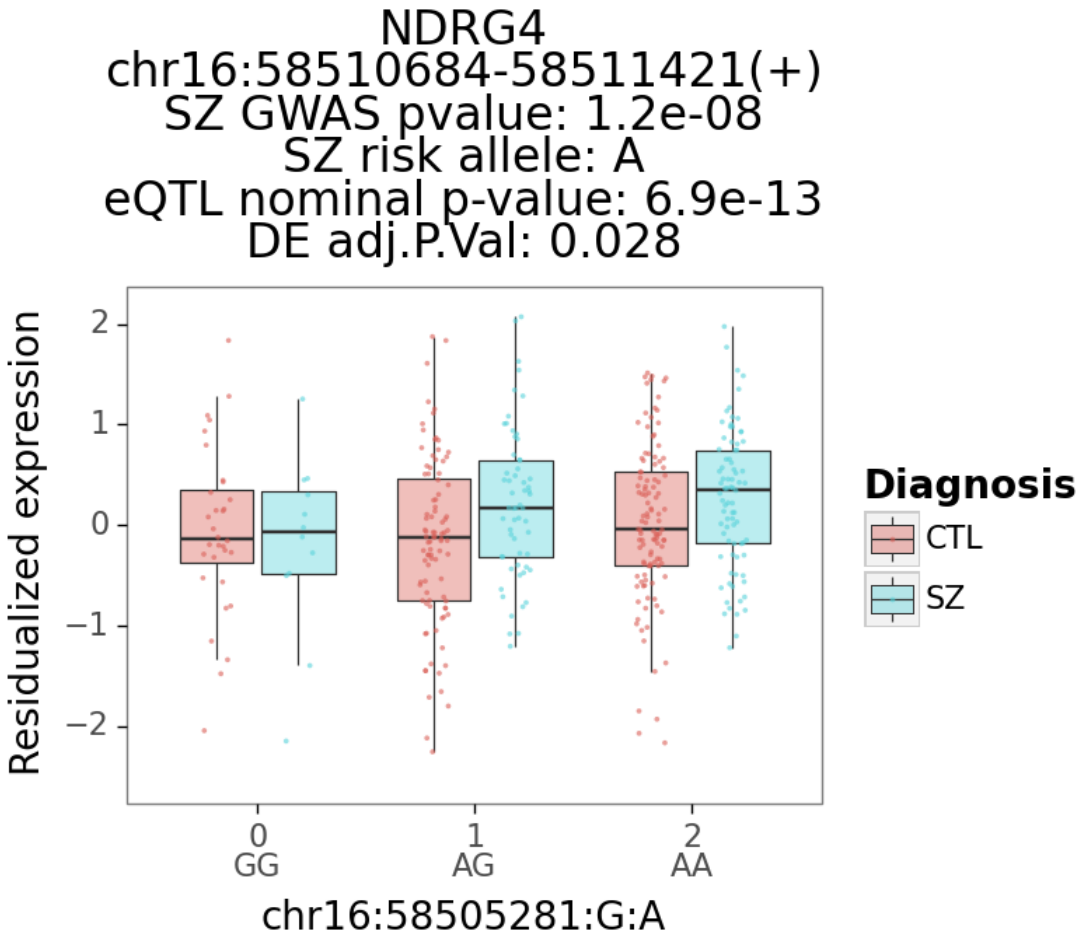

```
<ggplot: (8752691017488)>
```



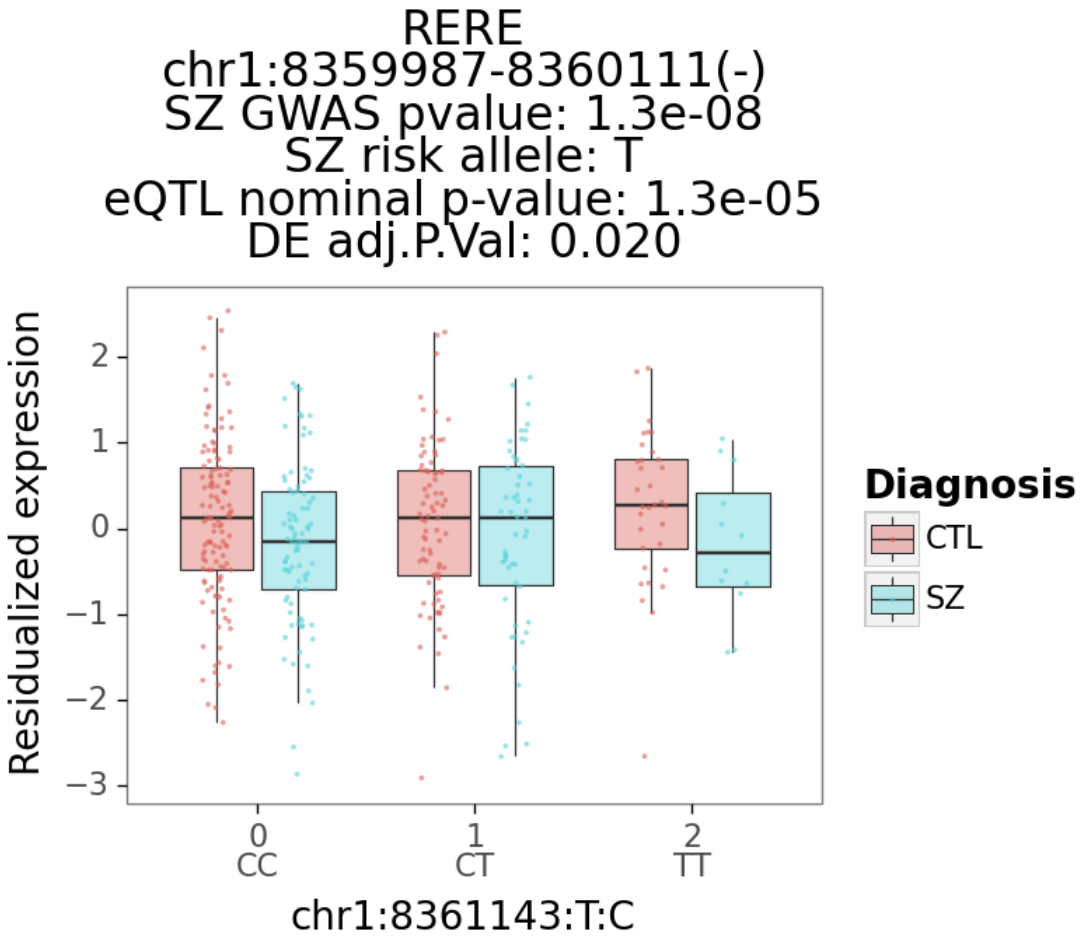

```
<ggplot: (8752694116145)>
```



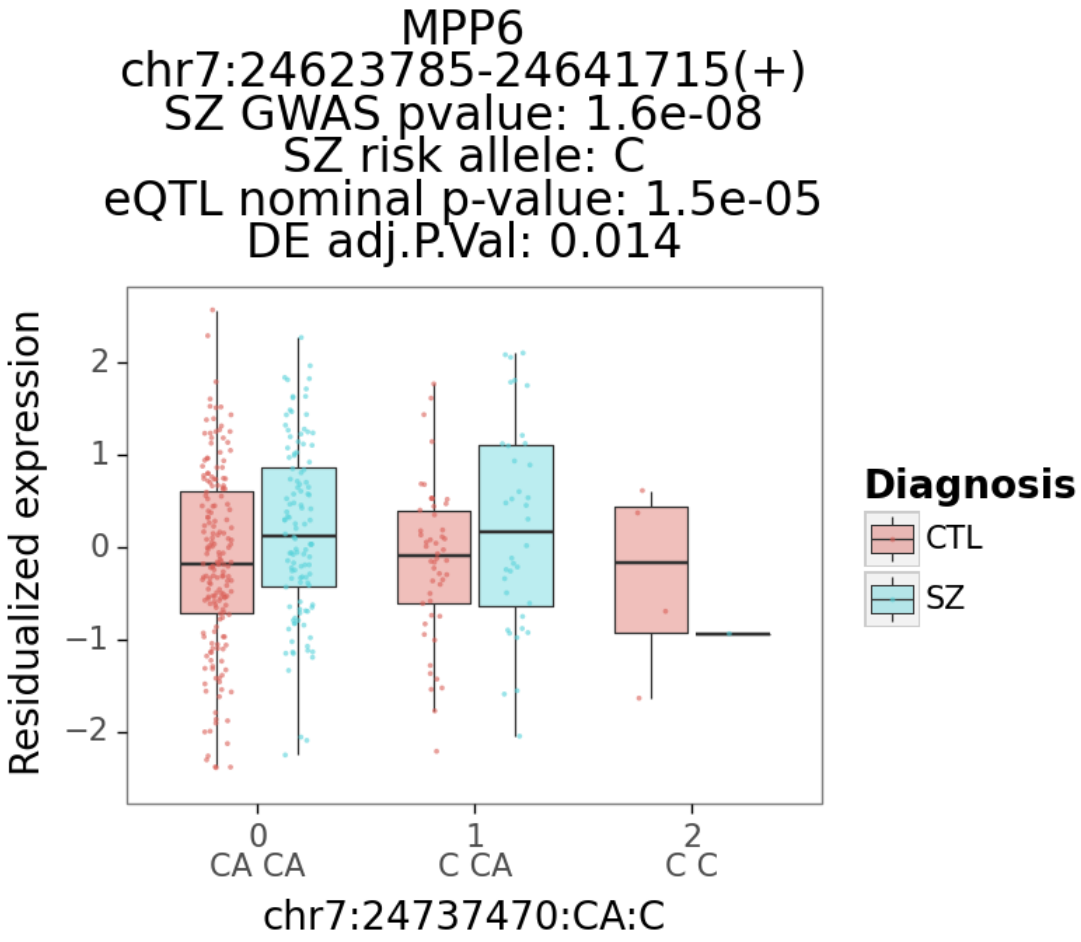

```
<ggplot: (8752700112821)>
```



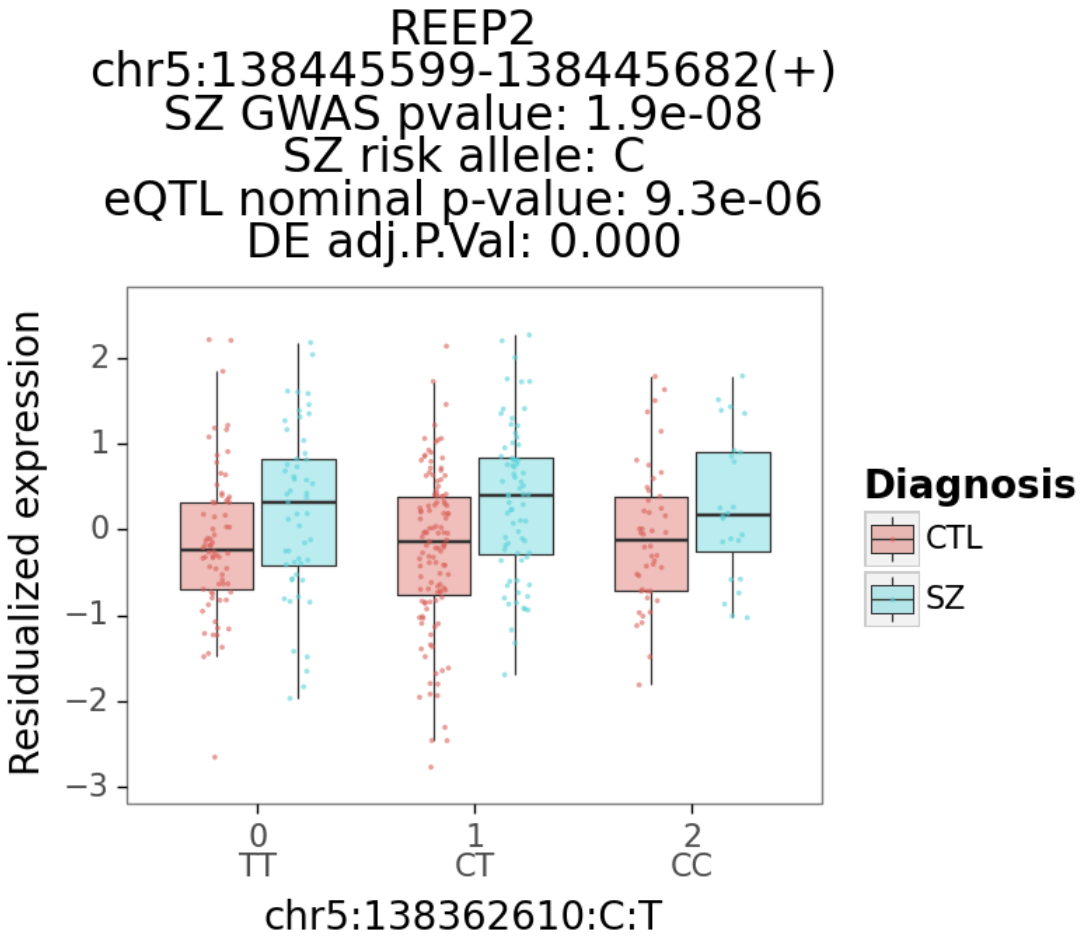

```
<ggplot: (8752699380023)>
```



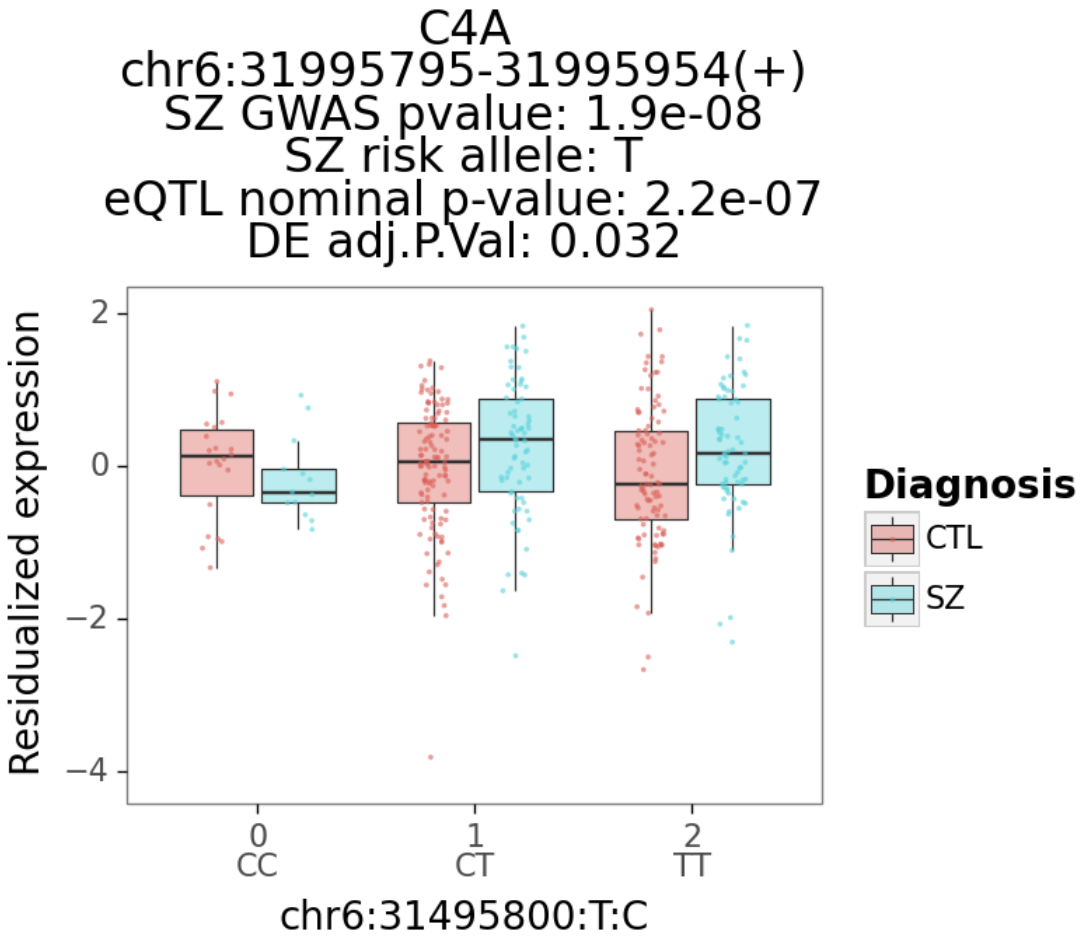

```
<ggplot: (8752654763460)>
```



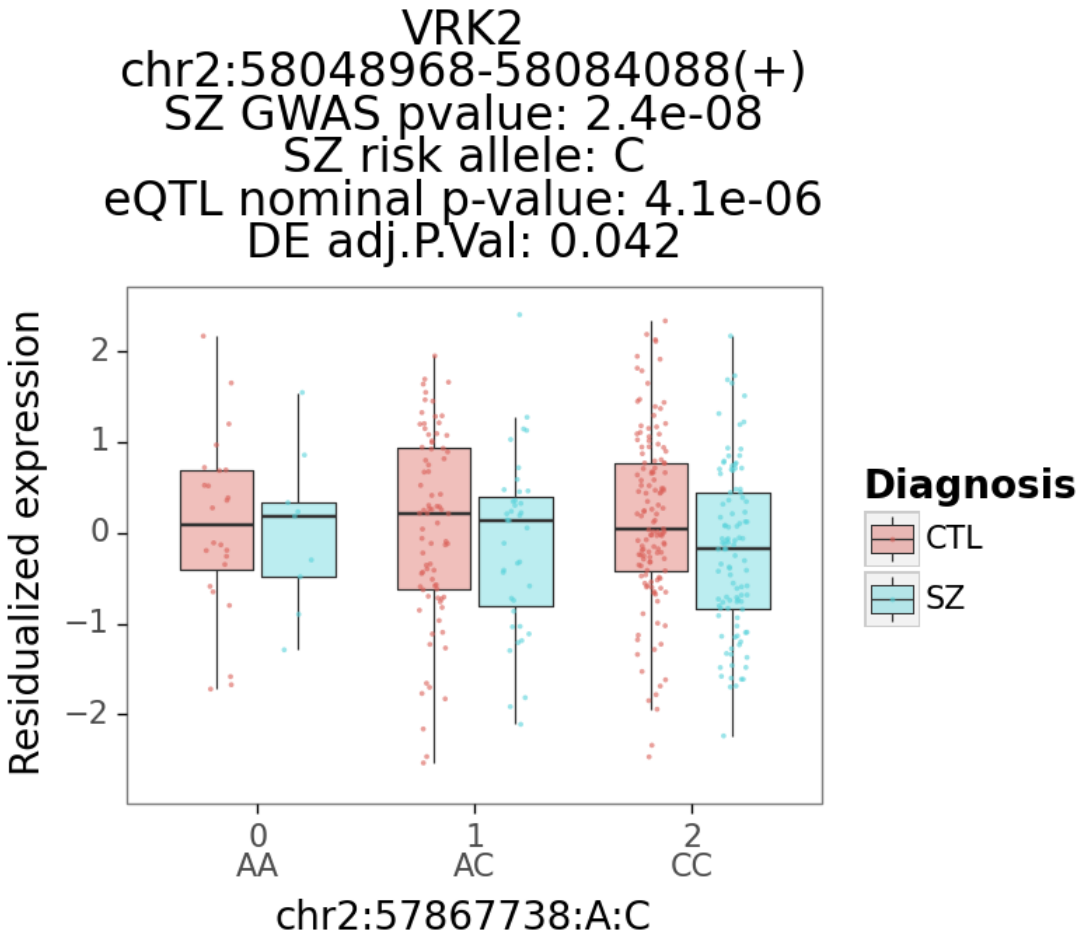
<ggplot: (8752699459801)>



<ggplot: (8752698812636)>




```
<ggplot: (8752658908564)>
```




```
<ggplot: (8752655412017)>
```