extTADA for developmental disorder (DD)

This example describes steps to obtain results for DD de novo data.

Some main steps:

1. Estimate genetic parameters using MCCMC.
2. Use these parameters to calculate FDRs for each gene.

Load source files of extTADA

In [1]:

```
fileR <- dir("../script", ".R$")
for (ii in fileR)
    source(paste0("../script/", ii))
```

```
Warning message:
: package ‘rstan’ was built under R version 3.2.5Loading required package: ggplot2
Warning message:
: package ‘ggplot2’ was built under R version 3.2.4Loading required package: StanHeaders
Warning message:
: package ‘StanHeaders’ was built under R version 3.2.5rstan (Version 2.12.1, packaged: 2016-09-11
13:07:50 UTC, GitRev: 85f7a56811da)
For execution on a local, multicore CPU with excess RAM we recommend calling
rstan_options(auto_write = TRUE)
options(mc.cores = parallel::detectCores())
locfit 1.5-9.1   2013-03-22
```

In [2]:

```
data <- read.table("../data/data_mut_DD.csv", header = TRUE, as.is = TRUE)
head(data)
```

Out[2]:

	Gene	mut_lof	mut_missense	mut_damaging	dn_damaging_DD	dn_lof_DD	dn_missense_DD	dn_silent_DD
1	A1BG	9.996657e-07	2.300224e-05	4.54062e-07	0	0	0	0
2	A1BG-AS1	1.420491e-07	5.566513e-08	1.04146e-10	0	0	0	0
3	A1CF	2.144318e-06	1.68827e-05	3.35199e-06	0	0	0	0
4	A2M	3.981797e-06	4.039597e-05	7.82154e-06	0	0	0	0
5	A2M-AS1	9.14698e-08	5.566513e-08	1.04146e-10	0	0	0	0
6	A2ML1	3.779929e-06	4.007178e-05	2.85364e-06	0	0	1	0

In [ ]:

In [3]:

```
allDNData <- data[, paste0("dn_", c("damaging", "lof"), "_DD")]
allMutData <- data[,paste0("mut_", c("damaging", "lof"))]
head(data.frame(allMutData, allDNData))
```

Out[3]:

	mut_damaging	mut_lof	dn_damaging_DD	dn_lof_DD
1	4.54062e-07	9.996657e-07	0	0
2	1.04146e-10	1.420491e-07	0	0
3	3.35199e-06	2.144318e-06	0	0
4	7.82154e-06	3.981797e-06	0	0
5	1.04146e-10	9.14698e-08	0	0
6	2.85364e-06	3.779929e-06	0	0

Use the function *extTADA* to sample values of parameters

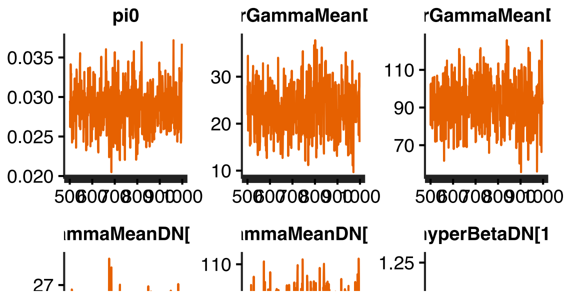
In [4]:

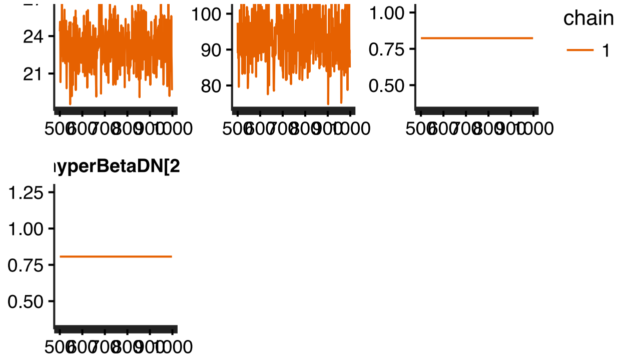
```
mcmcDD <- extTADA(modelName = DNextTADA, #extTADA for only de novo data
                 dataDN = allDNData, mutRate = allMutData,
                 Ndn = rep(4293, 2),
                 nIteration = 1000,
                 nIteration2 = 2000)
```

```
There are 19358 in this analysis

Sampling with nter = 1000 and nThin = 1


The model DNextTADA is used


clang: warning: optimization flag '-ffat-lto-objects' is not supported
clang: warning: argument unused during compilation: '-ffat-lto-objects'

SAMPLING FOR MODEL '493590f0b50b55273a713acc3c2b8395' NOW (CHAIN 1).

Chain 1, Iteration:   1 / 1000 [  0%]  (Warmup)
Chain 1, Iteration: 100 / 1000 [ 10%]  (Warmup)
Chain 1, Iteration: 200 / 1000 [ 20%]  (Warmup)
Chain 1, Iteration: 300 / 1000 [ 30%]  (Warmup)
Chain 1, Iteration: 400 / 1000 [ 40%]  (Warmup)
Chain 1, Iteration: 500 / 1000 [ 50%]  (Warmup)
Chain 1, Iteration: 501 / 1000 [ 50%]  (Sampling)
Chain 1, Iteration: 600 / 1000 [ 60%]  (Sampling)
Chain 1, Iteration: 700 / 1000 [ 70%]  (Sampling)
Chain 1, Iteration: 800 / 1000 [ 80%]  (Sampling)
Chain 1, Iteration: 900 / 1000 [ 90%]  (Sampling)
Chain 1, Iteration: 1000 / 1000 [100%]  (Sampling)
 Elapsed Time: 43.3239 seconds (Warm-up)
               32.5705 seconds (Sampling)
               75.8945 seconds (Total)


Sampling with nter = 2000 and nThin = 2


The model DNextTADA is used


clang: warning: optimization flag '-ffat-lto-objects' is not supported
clang: warning: argument unused during compilation: '-ffat-lto-objects'

SAMPLING FOR MODEL '493590f0b50b55273a713acc3c2b8395' NOW (CHAIN 1).

Chain 1, Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 1, Iteration:  200 / 2000 [ 10%]  (Warmup)
Chain 1, Iteration:  400 / 2000 [ 20%]  (Warmup)
Chain 1, Iteration:  600 / 2000 [ 30%]  (Warmup)
Chain 1, Iteration:  800 / 2000 [ 40%]  (Warmup)
Chain 1, Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 1, Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 1, Iteration: 1200 / 2000 [ 60%]  (Sampling)
Chain 1, Iteration: 1400 / 2000 [ 70%]  (Sampling)
Chain 1, Iteration: 1600 / 2000 [ 80%]  (Sampling)
Chain 1, Iteration: 1800 / 2000 [ 90%]  (Sampling)
Chain 1, Iteration: 2000 / 2000 [100%]  (Sampling)
 Elapsed Time: 75.5381 seconds (Warm-up)
               68.6742 seconds (Sampling)
               144.212 seconds (Total)
```

Take a quick look at the traces of parameters

```
In [5]: options(repr.plot.width=5, repr.plot.height=5)
        stan_trace(mcmcDD)
```





In [6]:
```
mcmcDD
```

Out[6]:
```
Inference for Stan model: 493590f0b50b55273a713acc3c2b8395.
1 chains, each with iter=2000; warmup=1000; thin=2;
post-warmup draws per chain=500, total post-warmup draws=500.

                       mean se_mean    sd     2.5%      25%      50%      75%
pi0                    0.03    0.00  0.00     0.02     0.03     0.03     0.03
hyperGammaMeanDN[1]   23.21    0.27  5.02    13.55    19.81    23.38    26.53
hyperGammaMeanDN[2]   92.04    0.65 12.34    69.32    83.19    92.65   100.08
gammaMeanDN[1]        23.12    0.08  1.60    20.03    22.06    23.05    24.09
gammaMeanDN[2]        93.57    0.37  6.61    81.51    88.92    93.21    97.91
hyperBetaDN[1]         0.82    0.00  0.00     0.82     0.82     0.82     0.82
hyperBetaDN[2]         0.81    0.00  0.00     0.81     0.81     0.81     0.81
lp__               -6792.93    0.07  1.49 -6796.77 -6793.69 -6792.59 -6791.82
                      97.5% n_eff Rhat
pi0                    0.03   311 1.00
hyperGammaMeanDN[1]   32.59   338 1.00
hyperGammaMeanDN[2]  114.98   361 1.00
gammaMeanDN[1]        26.13   409 1.00
gammaMeanDN[2]       105.68   317 1.00
hyperBetaDN[1]         0.82     1 1.00
hyperBetaDN[2]         0.81     1 1.00
lp__               -6791.03   423 1.01

Samples were drawn using NUTS(diag_e) at Wed Dec 14 16:00:11 2016.
For each parameter, n_eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor on split chains (at
convergence, Rhat=1).
```

Use the function *estimatePars* of extTADA to obtain modes, and credible intervals (ID) of parameters

In [7]:
```
pars0 <- estimatePars(pars = c('pi0',
                               'hyperGammaMeanDN[1]', 'hyperGammaMeanDN[2]',
                               'hyperBetaDN[1]', 'hyperBetaDN[2]'),
                     mcmcResult = mcmcDD)
```

```
====
Only pi and hyper parameters are estimated in this step
extTADA does not calculate HPDs for hyper betas, just their medians
===
```

In [8]:
```
pars0
```

Out[8]:

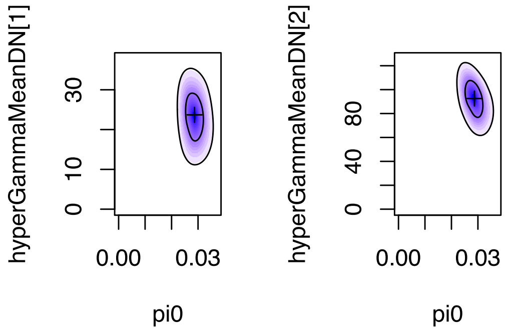
	Mode	lCI	uCI
pi0	0.02870958	0.02327248	0.03431055
hyperGammaMeanDN[1]	23.60775	13.52292	32.87455
hyperGammaMeanDN[2]	93.46348	68.64038	114.97203
hyperBetaDN[1]	0.8232544	0.8232544	0.8232544
hyperBetaDN[2]	0.8066756	0.8066756	0.8066756

Use the function *plotParHeatmap* of extTADA to draw heatmaps of pairs of pars

In [9]:
```
options(repr.plot.width=4, repr.plot.height=3)
```

```
In [9]: options(repr.plot.width=4, repr.plot.height=3)
        par(mfrow = c(1, 2))
        plotParHeatmap(pars = c("pi0", "hyperGammaMeanDN[1]"), mcmcResult = mcmcDD)
        plotParHeatmap(pars = c("pi0", "hyperGammaMeanDN[2]"), mcmcResult = mcmcDD)
```

```
Warning message:
In plot.xy(xy, type, ...): font width unknown for character 0x1Warning message:
In plot.xy(xy, type, ...): font metrics unknown for character 0x1Warning message:
In plot.xy(xy, type, ...): font width unknown for character 0x1Warning message:
In plot.xy(xy, type, ...): font metrics unknown for character 0x1
```



Use function *calculateFDR* of extTADA to obtain FDRs of genes

```
In [13]: ##Get gene list
         geneName <- data[, 1]
         ntrioDD = 4293
         ##Set parameters: use pars0 above
         parsFDR <- list(gammaMeanDN = pars0[, 1][2:3],
                         betaDN = pars0[, 1][4:5],
                     pi0 = pars0[, 1][1],
                     nfamily = rep(ntrioDD, 2))

         dataFDR <- calculateFDR(pars = parsFDR,
                                 dnData = allDNData, mutData = allMutData,
                                 geneName = geneName)
```

```
No parameters for case-control data;  therefore, these categories are not calculated in this step.
```

```
In [14]: head(dataFDR)
```

Out[14]:

	geneName	dn_damaging_DD	dn_lof_DD	mut_damaging	mut_lof	BF	qvalue
681	ANKRD11	0	32	1.12154e-05	4.918288e-06	2.153447e+62	0
1001	ARID1B	0	30	9.81608e-06	4.623958e-06	2.411403e+58	0
10146	MLL	1	26	1.19316e-05	9.084711e-06	1.877296e+49	0
347	ADNP	1	19	2.26581e-06	2.100249e-06	4.275545e+38	0
4832	DYRK1A	4	14	4.53477e-06	2.145461e-06	2.406051e+32	0
9906	MED13L	5	13	1.5154e-05	4.661034e-06	5.968457e+29	0

```
In [15]: dim(dataFDR[dataFDR$qvalue < 0.1, ])
         dim(dataFDR[dataFDR$qvalue < 0.05, ])
```

Out[15]:
1. 199
2. 7

Out[15]:
1. 162
2. 7