lab_3

March 16, 2022

```
[ ]: from cmdstanpy import CmdStanModel
     import pandas as pd
     import arviz as az
     import numpy as np
     import matplotlib.pyplot as plt
     import scipy.stats as stats
```

0.1 Excercise 1 - Divergence problem

0.1.1 Variant A

```
[ ]: model = CmdStanModel(stan_file='stan1.stan')
```

```
INFO:cmdstanpy:compiling stan file C:\Users\norbe\Desktop\DataAnalytics\Lab
3\stan1.stan to exe file C:\Users\norbe\Desktop\DataAnalytics\Lab 3\stan1.exe
INFO:cmdstanpy:compiled model executable:
C:\Users\norbe\Desktop\DataAnalytics\Lab 3\stan1.exe
```

```
[ ]: result = model.sample(data={'N':1, 'y':[1]},
                                seed = 9012022,
                                chains = 4)
```

```
INFO:cmdstanpy:CmdStan start processing
chain 1 |          | 00:00 Status


chain 1 |        | 00:00 Iteration:  900 / 2000 [ 45%]  (Warmup)


chain 1 |      | 00:00 Sampling completed
chain 2 |      | 00:00 Sampling completed
chain 3 |      | 00:00 Sampling completed
chain 4 |      | 00:00 Sampling completed



INFO:cmdstanpy:CmdStan done processing.
```

[ ]:

```
print(result.diagnose())
```

```
Processing csv files:
C:\Users\norbe\AppData\Local\Temp\tmpsvkedz7a\stan1-20220316110957_1.csv,
C:\Users\norbe\AppData\Local\Temp\tmpsvkedz7a\stan1-20220316110957_2.csv,
C:\Users\norbe\AppData\Local\Temp\tmpsvkedz7a\stan1-20220316110957_3.csv,
C:\Users\norbe\AppData\Local\Temp\tmpsvkedz7a\stan1-20220316110957_4.csv

Checking sampler transitions treedepth.
Treedepth satisfactory for all transitions.

Checking sampler transitions for divergences.
256 of 4000 (6.40%) transitions ended with a divergence.
These divergent transitions indicate that HMC is not fully able to explore the
posterior distribution.
Try increasing adapt delta closer to 1.
If this doesn't remove all divergences, try to reparameterize the model.

Checking E-BFMI - sampler transitions HMC potential energy.
E-BFMI satisfactory.

Effective sample size satisfactory.

Split R-hat values satisfactory all parameters.

Processing complete.
```

[ ]:

```
#Convertion to arviz InferenceData
arviz_result = az.from_cmdstanpy(
    posterior=result
)
#Calculating log sigma
post = arviz_result.posterior
post["log_sigma"] = np.log(post["sigma"])
arviz_result
```

[ ]:

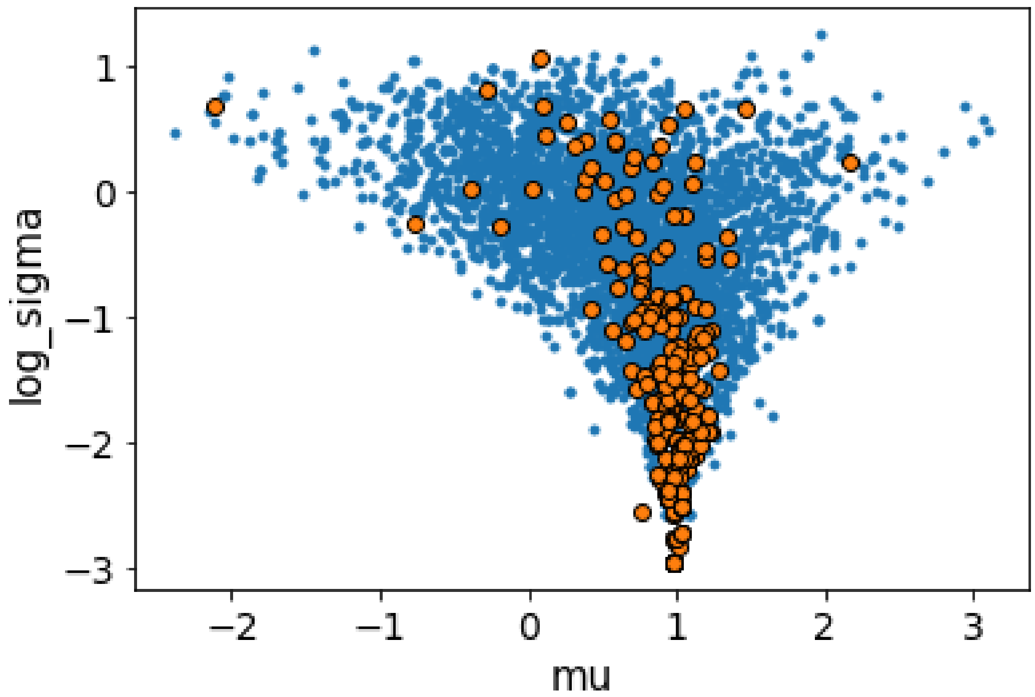
```
Inference data with groups:
        > posterior
        > sample_stats
```

[ ]:

```
#Arviz plot
az.plot_pair(arviz_result, var_names=['mu', 'log_sigma'], divergences=True)
```

[ ]:

```
<AxesSubplot:xlabel='mu', ylabel='log_sigma'>
```



0.1.2 Variant B

[ ]:
```
model = CmdStanModel(stan_file='stan1.stan')
```

```
INFO:cmdstanpy:found newer exe file, not recompiling
```

[ ]:
```
result = model.sample(data={'N':5, 'y':[1.05, 0.87, -0.49, -0.22, 0.18]},
                      seed = 9012022,
                      chains = 4)
```

```
INFO:cmdstanpy:CmdStan start processing
chain 1 |          | 00:00 Status


chain 1 |      | 00:00 Iteration: 1300 / 2000 [ 65%]  (Sampling)


chain 1 |     | 00:00 Sampling completed
chain 2 |     | 00:00 Sampling completed
chain 3 |     | 00:00 Sampling completed
chain 4 |     | 00:00 Sampling completed
```

```
INFO:cmdstanpy:CmdStan done processing.
```

[ ]:
```
print(result.diagnose())
```

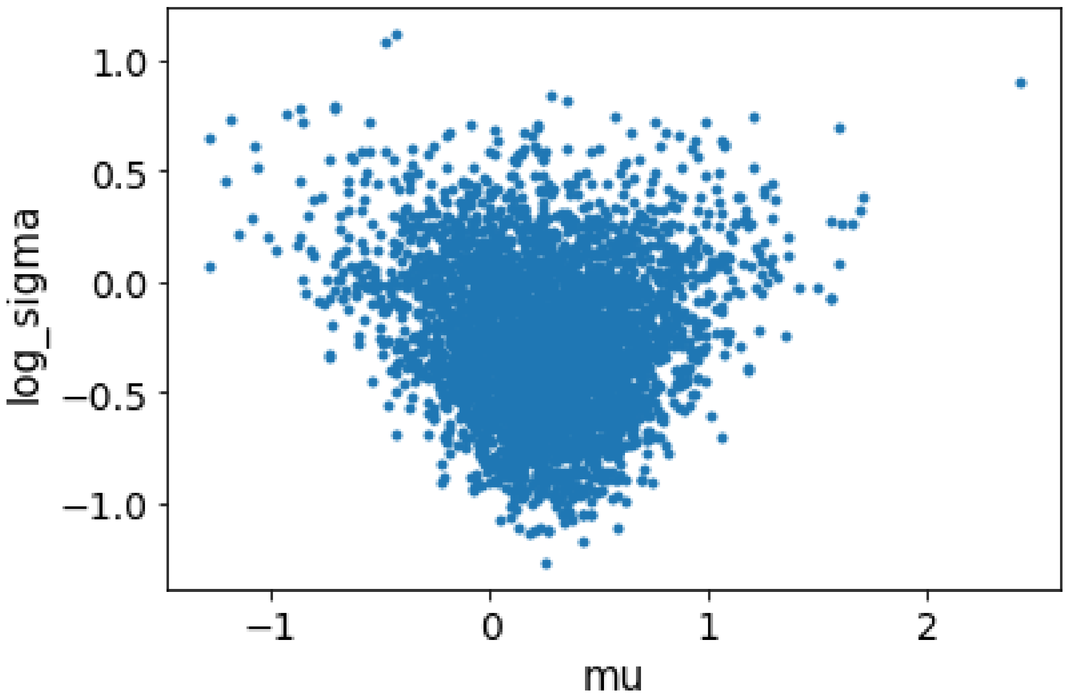
```
Processing csv files:
C:\Users\norbe\AppData\Local\Temp\tmpsvkedz7a\stan1-20220316111207_1.csv,
C:\Users\norbe\AppData\Local\Temp\tmpsvkedz7a\stan1-20220316111207_2.csv,
C:\Users\norbe\AppData\Local\Temp\tmpsvkedz7a\stan1-20220316111207_3.csv,
C:\Users\norbe\AppData\Local\Temp\tmpsvkedz7a\stan1-20220316111207_4.csv

Checking sampler transitions treedepth.
Treedepth satisfactory for all transitions.

Checking sampler transitions for divergences.
No divergent transitions found.

Checking E-BFMI - sampler transitions HMC potential energy.
E-BFMI satisfactory.

Effective sample size satisfactory.

Split R-hat values satisfactory all parameters.

Processing complete, no problems detected.
```

[ ]:
```
#Convertion to arviz InferenceData
arviz_result = az.from_cmdstanpy(
    posterior=result
)
#Calculating log sigma
post = arviz_result.posterior
post["log_sigma"] = np.log(post["sigma"])
arviz_result
```

[ ]:
```
Inference data with groups:
        > posterior
        > sample_stats
```

[ ]:
```
#Arviz plot
az.plot_pair(arviz_result, var_names=['mu', 'log_sigma'], divergences=True)
```

[ ]:
```
<AxesSubplot:xlabel='mu', ylabel='log_sigma'>
```



0.2 Excercise 2 - Binominial example

[ ]:

```
 #Data read fromm coin.csv
df = pd.read_csv("https://raw.githubusercontent.com/KAIR-ISZ/public_lectures/
 ↪master/Data%20Analytics%202022/Lab%203%20-%20Probability%20distribution/coin.
 ↪csv")
df.head()
y = df["Toss_Result"]
N = len(y)
print(y)
print(N)
```

```
0      0
1      0
2      1
3      0
4      0
      ..
413    0
414    1
415    1
416    0
417    1
Name: Toss_Result, Length: 418, dtype: int64
```

```
418
```

[ ]:
```
model = CmdStanModel(stan_file='stan2.stan')
result = model.sample(data={"N" : N, "y" : y},
                            seed = 9012022,
                            chains = 4)
```

```
INFO:cmdstanpy:compiling stan file C:\Users\norbe\Desktop\DataAnalytics\Lab
3\stan2.stan to exe file C:\Users\norbe\Desktop\DataAnalytics\Lab 3\stan2.exe
INFO:cmdstanpy:compiled model executable:
C:\Users\norbe\Desktop\DataAnalytics\Lab 3\stan2.exe
WARNING:cmdstanpy:Stan compiler has produced 1 warnings:
WARNING:cmdstanpy:
--- Translating Stan model to C++ code ---
bin/stanc.exe  --o=C:/Users/norbe/Desktop/DATAAN~1/LAB3~1/stan2.hpp
C:/Users/norbe/Desktop/DATAAN~1/LAB3~1/stan2.stan
Warning in 'C:/Users/norbe/Desktop/DATAAN~1/LAB3~1/stan2.stan', line 3, column
3: Declaration
    of arrays by placing brackets after a variable name is deprecated and
    will be removed in Stan 2.32.0. Instead use the array keyword before the
    type. This can be changed automatically using the auto-format flag to
    stanc

--- Compiling, linking C++ code ---
g++ -std=c++1y -m64 -D_REENTRANT -Wall -Wno-unused-function -Wno-uninitialized
-Wno-unused-but-set-variable -Wno-unused-variable -Wno-sign-compare -Wno-unused-
local-typedefs -Wno-int-in-bool-context -Wno-attributes -Wno-ignored-attributes
-I stan/lib/stan_math/lib/tbb_2020.3/include    -O3 -I src -I stan/src -I
lib/rapidjson_1.1.0/ -I lib/CLI11-1.9.1/ -I stan/lib/stan_math/ -I
stan/lib/stan_math/lib/eigen_3.3.9 -I stan/lib/stan_math/lib/boost_1.75.0 -I
stan/lib/stan_math/lib/sundials_6.0.0/include -I
stan/lib/stan_math/lib/sundials_6.0.0/src/sundials  -D_USE_MATH_DEFINES
-DBOOST_DISABLE_ASSERTS          -c  -x c++ -o
C:/Users/norbe/Desktop/DATAAN~1/LAB3~1/stan2.o
C:/Users/norbe/Desktop/DATAAN~1/LAB3~1/stan2.hpp
g++ -std=c++1y -m64 -D_REENTRANT -Wall -Wno-unused-function -Wno-uninitialized
-Wno-unused-but-set-variable -Wno-unused-variable -Wno-sign-compare -Wno-unused-
local-typedefs -Wno-int-in-bool-context -Wno-attributes -Wno-ignored-attributes
-I stan/lib/stan_math/lib/tbb_2020.3/include    -O3 -I src -I stan/src -I
lib/rapidjson_1.1.0/ -I lib/CLI11-1.9.1/ -I stan/lib/stan_math/ -I
stan/lib/stan_math/lib/eigen_3.3.9 -I stan/lib/stan_math/lib/boost_1.75.0 -I
stan/lib/stan_math/lib/sundials_6.0.0/include -I
stan/lib/stan_math/lib/sundials_6.0.0/src/sundials  -D_USE_MATH_DEFINES
-DBOOST_DISABLE_ASSERTS                -Wl,-
L,"C:/Users/norbe/.conda/envs/myenv/Library/bin/cmdstan/stan/lib/stan_math/lib/t
bb" -Wl,-
rpath,"C:/Users/norbe/.conda/envs/myenv/Library/bin/cmdstan/stan/lib/stan_math/l
ib/tbb"      C:/Users/norbe/Desktop/DATAAN~1/LAB3~1/stan2.o src/cmdstan/main.o
```

```
-static-libgcc -static-libstdc++      -Wl,-
L,"C:/Users/norbe/.conda/envs/myenv/Library/bin/cmdstan/stan/lib/stan_math/lib/t
bb" -Wl,-
rpath,"C:/Users/norbe/.conda/envs/myenv/Library/bin/cmdstan/stan/lib/stan_math/l
ib/tbb"   stan/lib/stan_math/lib/sundials_6.0.0/lib/libsundials_nvecserial.a
stan/lib/stan_math/lib/sundials_6.0.0/lib/libsundials_cvodes.a
stan/lib/stan_math/lib/sundials_6.0.0/lib/libsundials_idas.a
stan/lib/stan_math/lib/sundials_6.0.0/lib/libsundials_kinsol.a
stan/lib/stan_math/lib/tbb/tbb.dll -o
C:/Users/norbe/Desktop/DATAAN~1/LAB3~1/stan2.exe
rm -f C:/Users/norbe/Desktop/DATAAN~1/LAB3~1/stan2.o

INFO:cmdstanpy:CmdStan start processing
chain 1 |          | 00:00 Status


chain 1 |       | 00:00 Iteration: 1100 / 2000 [ 55%]  (Sampling)


chain 1 |      | 00:00 Sampling completed
chain 2 |      | 00:00 Sampling completed
chain 3 |      | 00:00 Sampling completed
chain 4 |      | 00:00 Sampling completed





INFO:cmdstanpy:CmdStan done processing.
```

[ ]:
```
result.summary()
```

[ ]:
```
         Mean     MCSE  StdDev      5%     50%    95%   N_Eff  N_Eff/s  R_hat
name
lp__  -290.00  0.01800   0.720 -290.00 -290.00 -290.0  1600.0   7200.0    1.0
theta    0.46  0.00066   0.025    0.42    0.46    0.5  1400.0   6300.0    1.0
```

[ ]:
```
#Convertion to arviz InferenceData
arviz_result = az.from_cmdstanpy(
    posterior=result
)
arviz_result
```

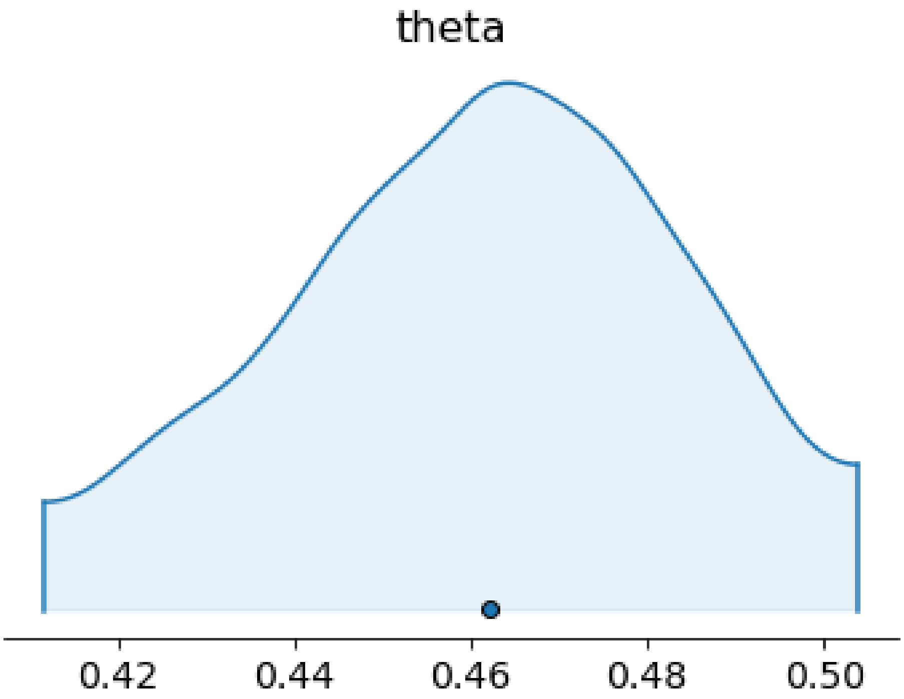
[ ]:
```
Inference data with groups:
        > posterior
        > sample_stats
```

[ ]:

```
#Plot
az.plot_density(arviz_result,shade=0.1)
```

[ ]:

```
array([[<AxesSubplot:title={'center':'theta'}>]], dtype=object)
```



0.3 Excercise 3 - Predictive checks

[ ]:

```
#Data read from normal.csv
df = pd.read_csv("https://raw.githubusercontent.com/KAIR-ISZ/public_lectures/
 ↪master/Data%20Analytics%202022/Lab%203%20-%20Probability%20distribution/
 ↪normal.csv", index_col=0)
df.head()
y = df["value"]
N = len(y)
```

0.3.1 Posterior predictive

[ ]:

```
model = CmdStanModel(stan_file='stan3.stan')
```

```
INFO:cmdstanpy:compiling stan file C:\Users\norbe\Desktop\DataAnalytics\Lab
3\stan3.stan to exe file C:\Users\norbe\Desktop\DataAnalytics\Lab 3\stan3.exe
INFO:cmdstanpy:compiled model executable:
C:\Users\norbe\Desktop\DataAnalytics\Lab 3\stan3.exe
```

[ ]:
```
result = model.sample(data={'N':N, 'y':y},
                              seed = 9012022,
                              chains = 4)
```

```
INFO:cmdstanpy:CmdStan start processing
chain 1 |          | 00:00 Status







chain 1 |     | 00:00 Sampling completed
chain 2 |     | 00:00 Sampling completed
chain 3 |     | 00:00 Sampling completed
chain 4 |     | 00:00 Sampling completed




INFO:cmdstanpy:CmdStan done processing.
```

[ ]:
```
result.summary()
```

[ ]:
```
                Mean    MCSE  StdDev     5%      50%    95%   N_Eff  N_Eff/s  \
name
lp__        -53.0000  0.0250   1.100 -55.00 -53.0000 -52.00  1700.0   1100.0
mu           -0.0350  0.0020   0.110  -0.21  -0.0340   0.14  2800.0   1700.0
sigma         1.0000  0.0013   0.073   0.91   1.0000   1.20  3100.0   1900.0
y_rep[1]     -0.0290  0.0160   1.000  -1.70  -0.0360   1.70  4259.0   2634.0
y_rep[2]     -0.0530  0.0160   1.000  -1.70  -0.0710   1.70  4170.0   2579.0
…                …      …       …      …      …       …       …        …
y_rep[96]    -0.0330  0.0170   1.000  -1.70  -0.0340   1.60  3868.0   2392.0
y_rep[97]    -0.0520  0.0170   1.000  -1.80  -0.0610   1.70  3928.0   2429.0
y_rep[98]    -0.0080  0.0170   1.000  -1.70  -0.0180   1.70  3928.0   2429.0
y_rep[99]    -0.0067  0.0180   1.000  -1.70   0.0091   1.70  3215.0   1988.0
y_rep[100]  -0.0520  0.0160   1.000  -1.70  -0.0400   1.60  3999.0   2473.0

            R_hat
name
lp__          1.0
mu            1.0
sigma         1.0
y_rep[1]      1.0
y_rep[2]      1.0
…            …
y_rep[96]     1.0
y_rep[97]     1.0
```

```
y_rep[98]     1.0
y_rep[99]     1.0
y_rep[100]    1.0

[103 rows x 9 columns]
```

[ ]:

```
#Convertion to arviz InferenceData
arviz_result = az.from_cmdstanpy(
    posterior=result,
    posterior_predictive=["y_rep"],
    observed_data={"y": y},
)
arviz_result
```

[ ]:

```
Inference data with groups:
        > posterior
        > posterior_predictive
        > sample_stats
        > observed_data
```

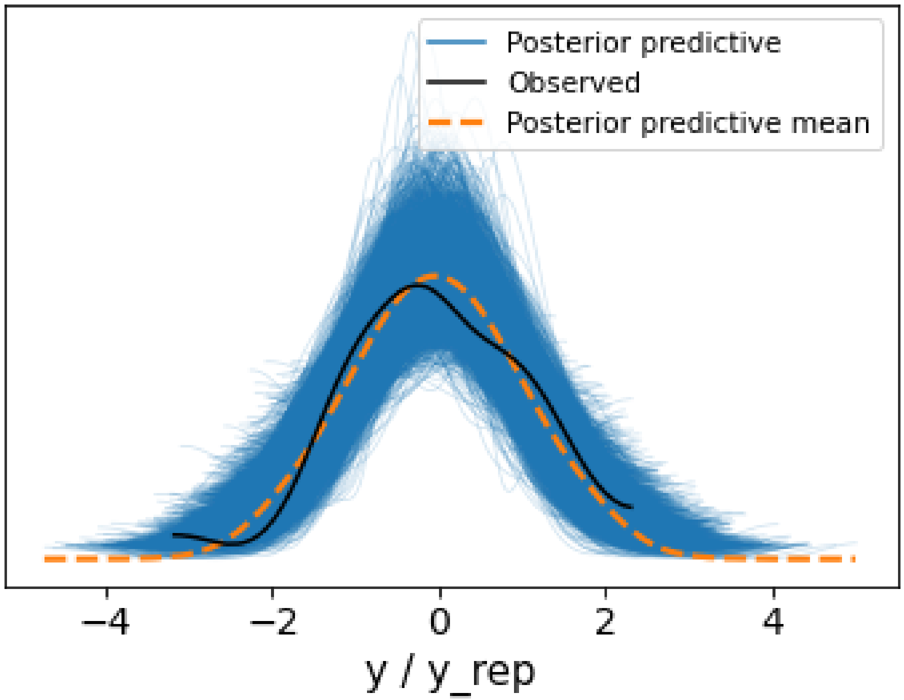
[ ]:

```
#Arviz plot
az.plot_ppc(arviz_result,data_pairs={"y": "y_rep"})
```

[ ]:

```
<AxesSubplot:xlabel='y / y_rep'>
```

```
C:\Users\norbe\.conda\envs\myenv\lib\site-
packages\IPython\core\pylabtools.py:151: UserWarning: Creating legend with
loc="best" can be slow with large amounts of data.
  fig.canvas.print_figure(bytes_io, **kw)
```



0.3.2 Prior predictive

[ ]:
```
model = CmdStanModel(stan_file='stan4.stan')
```

```
INFO:cmdstanpy:compiling stan file C:\Users\norbe\Desktop\DataAnalytics\Lab
3\stan4.stan to exe file C:\Users\norbe\Desktop\DataAnalytics\Lab 3\stan4.exe
INFO:cmdstanpy:compiled model executable:
C:\Users\norbe\Desktop\DataAnalytics\Lab 3\stan4.exe
WARNING:cmdstanpy:Stan compiler has produced 1 warnings:
WARNING:cmdstanpy:
--- Translating Stan model to C++ code ---
bin/stanc.exe  --o=C:/Users/norbe/Desktop/DATAAN~1/LAB3~1/stan4.hpp
C:/Users/norbe/Desktop/DATAAN~1/LAB3~1/stan4.stan
Warning in 'C:/Users/norbe/Desktop/DATAAN~1/LAB3~1/stan4.stan', line 7, column
26: Use
    of the `abs` function with real-valued arguments is deprecated; use
    function `fabs` instead.

--- Compiling, linking C++ code ---
g++ -std=c++1y -m64 -D_REENTRANT -Wall -Wno-unused-function -Wno-uninitialized
-Wno-unused-but-set-variable -Wno-unused-variable -Wno-sign-compare -Wno-unused-
local-typedefs -Wno-int-in-bool-context -Wno-attributes -Wno-ignored-attributes
-I stan/lib/stan_math/lib/tbb_2020.3/include    -O3 -I src -I stan/src -I
lib/rapidjson_1.1.0/ -I lib/CLI11-1.9.1/ -I stan/lib/stan_math/ -I
```

```
stan/lib/stan_math/lib/eigen_3.3.9 -I stan/lib/stan_math/lib/boost_1.75.0 -I
stan/lib/stan_math/lib/sundials_6.0.0/include -I
stan/lib/stan_math/lib/sundials_6.0.0/src/sundials  -D_USE_MATH_DEFINES
-DBOOST_DISABLE_ASSERTS          -c  -x c++ -o
C:/Users/norbe/Desktop/DATAAN~1/LAB3~1/stan4.o
C:/Users/norbe/Desktop/DATAAN~1/LAB3~1/stan4.hpp
g++ -std=c++1y -m64 -D_REENTRANT -Wall -Wno-unused-function -Wno-uninitialized
-Wno-unused-but-set-variable -Wno-unused-variable -Wno-sign-compare -Wno-unused-
local-typedefs -Wno-int-in-bool-context -Wno-attributes -Wno-ignored-attributes
-I stan/lib/stan_math/lib/tbb_2020.3/include    -O3 -I src -I stan/src -I
lib/rapidjson_1.1.0/ -I lib/CLI11-1.9.1/ -I stan/lib/stan_math/ -I
stan/lib/stan_math/lib/eigen_3.3.9 -I stan/lib/stan_math/lib/boost_1.75.0 -I
stan/lib/stan_math/lib/sundials_6.0.0/include -I
stan/lib/stan_math/lib/sundials_6.0.0/src/sundials  -D_USE_MATH_DEFINES
-DBOOST_DISABLE_ASSERTS                -Wl,-
L,"C:/Users/norbe/.conda/envs/myenv/Library/bin/cmdstan/stan/lib/stan_math/lib/t
bb" -Wl,-
rpath,"C:/Users/norbe/.conda/envs/myenv/Library/bin/cmdstan/stan/lib/stan_math/l
ib/tbb"      C:/Users/norbe/Desktop/DATAAN~1/LAB3~1/stan4.o src/cmdstan/main.o
-static-libgcc -static-libstdc++      -Wl,-
L,"C:/Users/norbe/.conda/envs/myenv/Library/bin/cmdstan/stan/lib/stan_math/lib/t
bb" -Wl,-
rpath,"C:/Users/norbe/.conda/envs/myenv/Library/bin/cmdstan/stan/lib/stan_math/l
ib/tbb"   stan/lib/stan_math/lib/sundials_6.0.0/lib/libsundials_nvecserial.a
stan/lib/stan_math/lib/sundials_6.0.0/lib/libsundials_cvodes.a
stan/lib/stan_math/lib/sundials_6.0.0/lib/libsundials_idas.a
stan/lib/stan_math/lib/sundials_6.0.0/lib/libsundials_kinsol.a
stan/lib/stan_math/lib/tbb/tbb.dll -o
C:/Users/norbe/Desktop/DATAAN~1/LAB3~1/stan4.exe
rm -f C:/Users/norbe/Desktop/DATAAN~1/LAB3~1/stan4.o
```

[ ]:
```python
result = model.sample(data={'N':100},
                            chains = 1,
                            fixed_param=True,
                            seed = 523833
                            )
```

```
INFO:cmdstanpy:CmdStan start processing
chain 1 |      | 00:00 Sampling completed
```

```
INFO:cmdstanpy:CmdStan done processing.
```

[ ]:
```python
result.summary()
```

[ ]:

```
                  Mean   MCSE  StdDev     5%     50%  95%   N_Eff  N_Eff/s  \
name
lp__            0.0000    NaN     0.0  0.000  0.0000  0.0     NaN      NaN
mu             -0.0220  0.031     1.0 -1.700 -0.0150  1.6  1100.0   4600.0
sigma           0.7900  0.019     0.6  0.041  0.6700  1.9  1000.0   4400.0
y_prior[1]     -0.0540  0.042     1.4 -2.300 -0.0740  2.3  1126.0   4877.0
y_prior[2]      0.0360  0.045     1.4 -2.100 -0.0077  2.4  1003.0   4343.0
…                  …      …       …      …  …      …      …
y_prior[96]    -0.0490  0.042     1.3 -2.200 -0.0060  2.0  1045.0   4523.0
y_prior[97]     0.0027  0.050     1.4 -2.100  0.0300  2.2   770.0   3333.0
y_prior[98]    -0.0190  0.041     1.4 -2.100 -0.0500  2.3  1178.0   5099.0
y_prior[99]    -0.0450  0.044     1.4 -2.400  0.0110  2.4  1060.0   4589.0
y_prior[100]   -0.0370  0.043     1.4 -2.200 -0.0320  2.3  1051.0   4551.0

              R_hat
name
lp__            NaN
mu              1.0
sigma           1.0
y_prior[1]      1.0
y_prior[2]      1.0
…             …
y_prior[96]     1.0
y_prior[97]     1.0
y_prior[98]     1.0
y_prior[99]     1.0
y_prior[100]    1.0

[103 rows x 9 columns]
```

[ ]:

```
#Convertion to arviz InferenceData
arviz_result = az.from_cmdstanpy(
    prior=result,
    prior_predictive=["y_prior"],
    observed_data={"y": y}
)
arviz_result
```

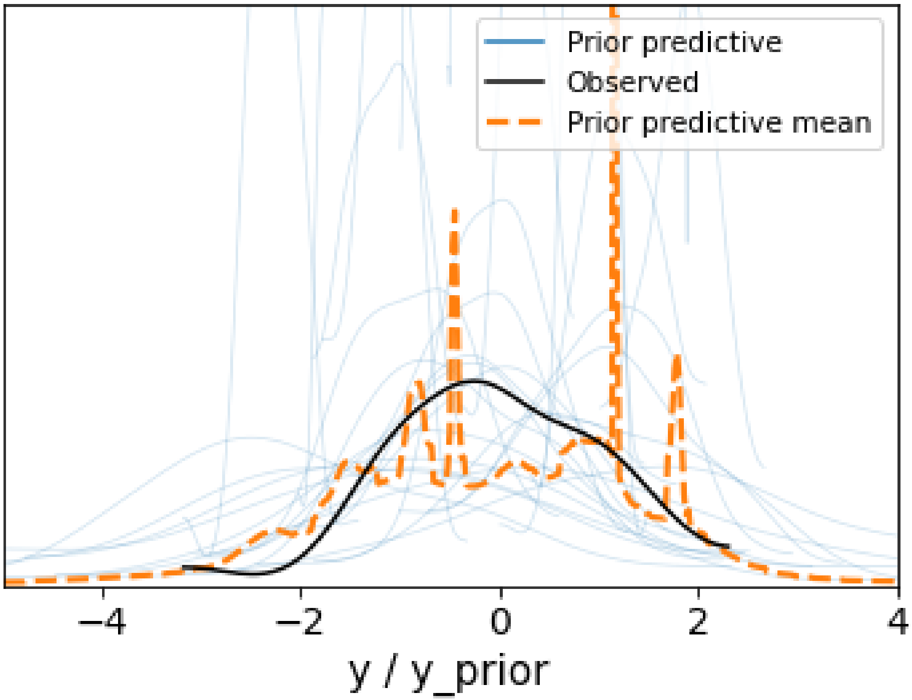
[ ]:

```
Inference data with groups:
        > prior
        > prior_predictive
        > sample_stats_prior
        > observed_data
```

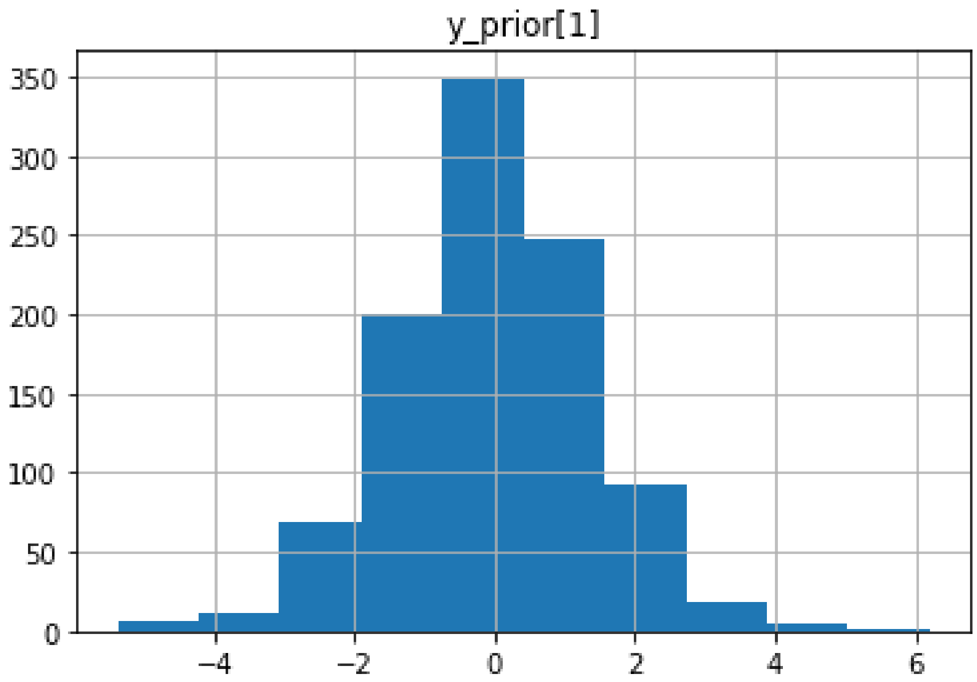
[ ]:

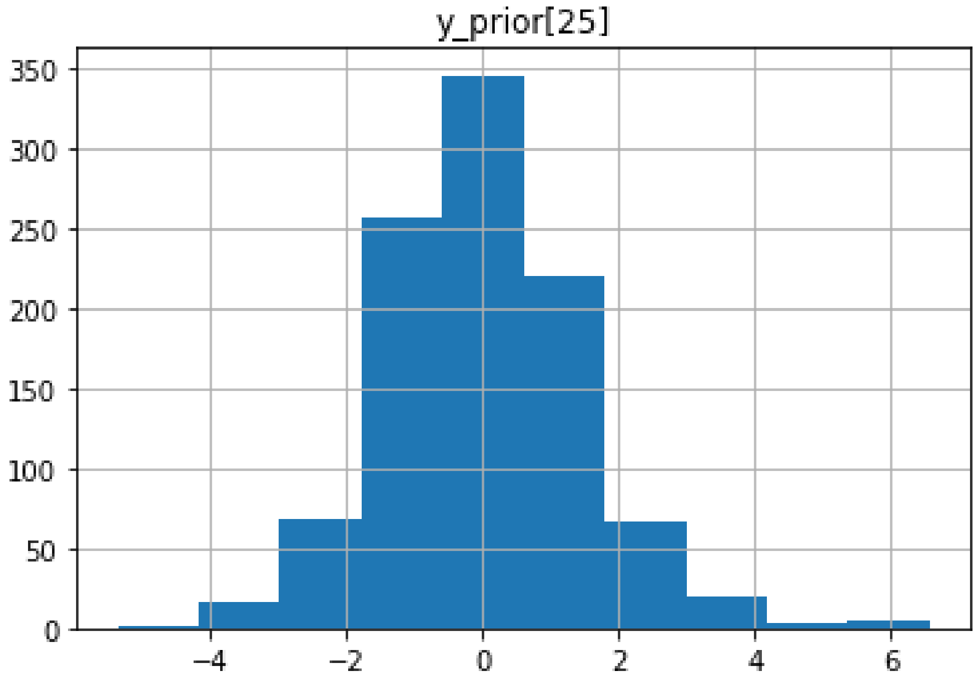
```
az.plot_ppc(arviz_result, group="prior", data_pairs={"y": "y_prior"},␣
 ↪num_pp_samples=30)
plt.axis([-5, 4, 0, 1])
```

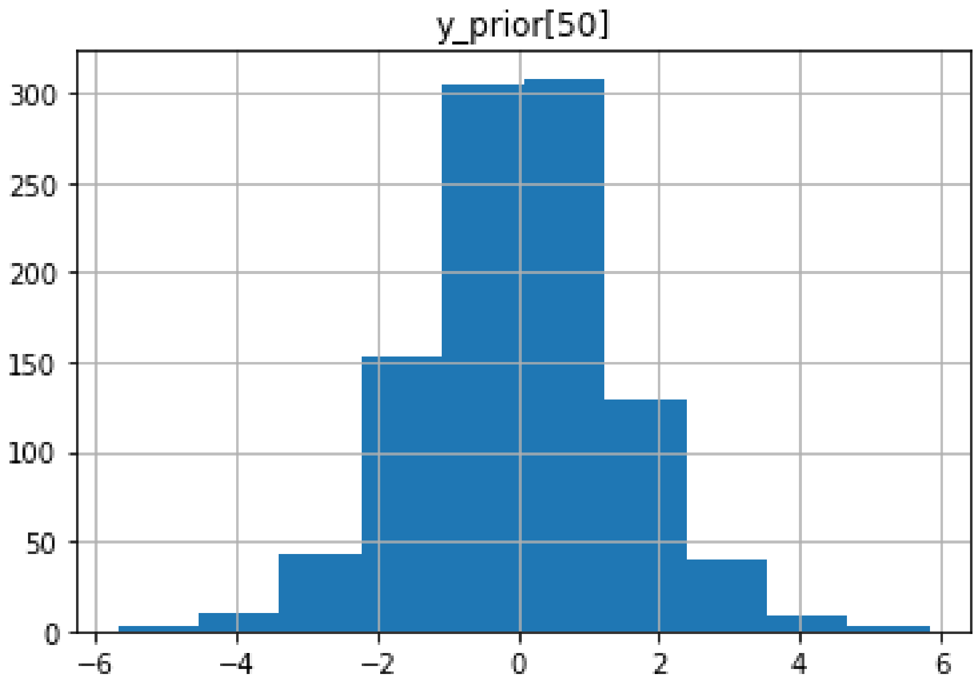
```
[ ]: (-5.0, 4.0, 0.0, 1.0)
```

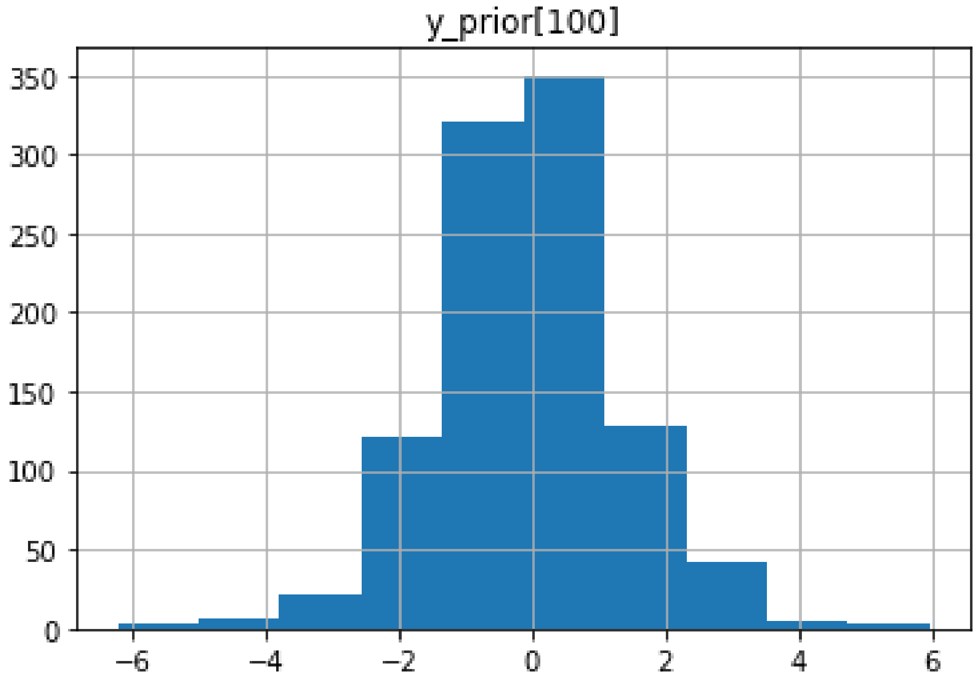


```
[ ]: #Draw y_prior histogram
     y_prior = result.draws_pd().drop(columns=["lp__", "accept_stat__", "mu",␣
      ↪"sigma"])
     y_prior["y_prior[1]"].hist()
     plt.title("y_prior[1]")
     plt.show()
     y_prior["y_prior[25]"].hist()
     plt.title("y_prior[25]")
     plt.show()
     y_prior["y_prior[50]"].hist()
     plt.title("y_prior[50]")
     plt.show()
     y_prior["y_prior[100]"].hist()
     plt.title("y_prior[100]")
     plt.show()
```

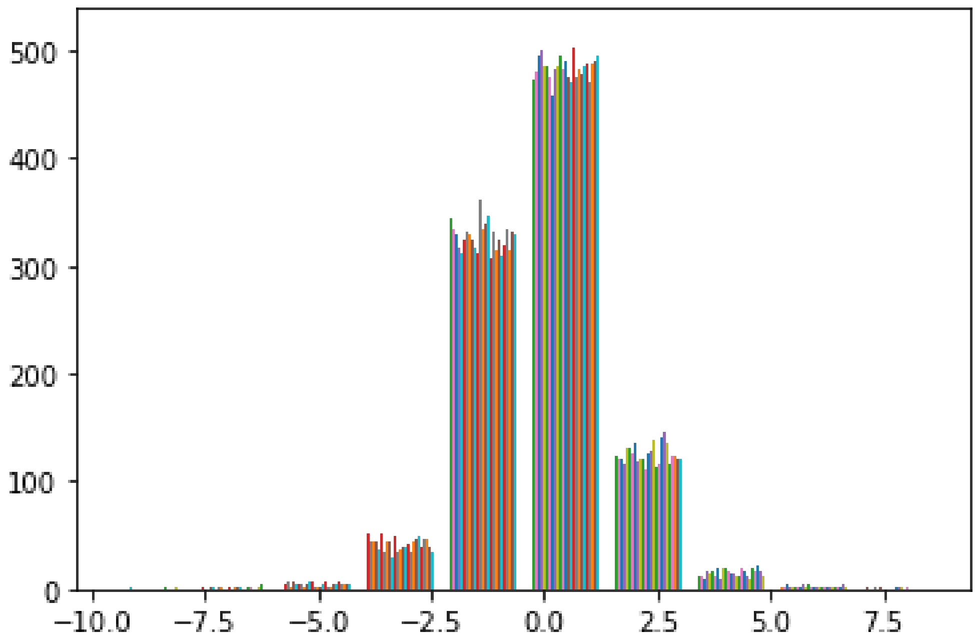

y_prior[1]
350
300
250
200
150
100
50
0
−4
−2
0
2
4
6


y_prior[25]
350
300
250
200
150
100
50
0
−4
−2
0
2
4
6





[ ]:
```
plt.hist(y_prior)
plt.show()
```


500
400
300
200
100
0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5