Task 1: simulating from Gaussian Process model

To make any simulations, first you need a set of observations:

In [11]:
```
from cmdstanpy import CmdStanModel

import arviz as az
import numpy as np
import scipy.stats as stats

import pandas as pd

import matplotlib.pyplot as plt
import matplotlib as mpl#Observed data
x_obs = [-10, -8, -6, -4, -2, 0, 2, 4, 6, 8, 10]
y_obs = [0.328572824089476, 4.20607004111644, 1.35507551134795,
0.161608755204364, -5.42320349780782, -3.05851276224202, -0.0764172642034
-4.55218472276499, -0.902226297922731, -5.8609833528976, -1.0585409091047


x_predict = np.linspace(-11,11,551,endpoint=True)
observed_idx = [26, 76, 126, 176, 226, 276, 326, 376, 426, 476, 526]

alpha = 3
rho = 5.5
sigma = 2

data = dict(N_predict = len(x_predict),
                x_predict = x_predict,
                N_obs = len(y_obs) ,y_obs = y_obs, observed_idx= observe
```

In [12]:
```
model = CmdStanModel(stan_file='gaussian_process.stan')
result = model.sample(data=data, chains = 4)
```

```
INFO:cmdstanpy:found newer exe file, not recompiling
INFO:cmdstanpy:CmdStan start processing
chain 1 |          | 00:00 Status
chain 2 |          | 00:00 Status
chain 3 |          | 00:00 Status
chain 4 |          | 00:00 Status

INFO:cmdstanpy:CmdStan done processing.
```

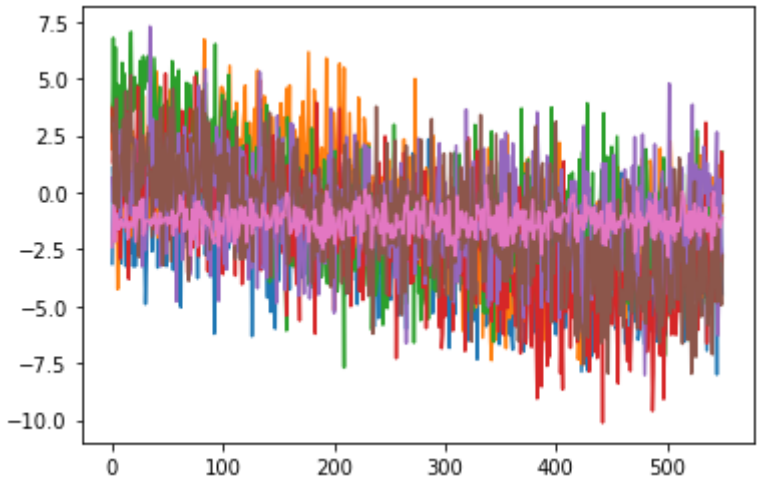
Your task now is to:

plot few (about 5-10) samples generated from model plot mean value from model with its confidence interval (standar deviation, using errorbar function) Also on each plot place points with obsserved data. Repeat the steps for two other parameters sets of your choice

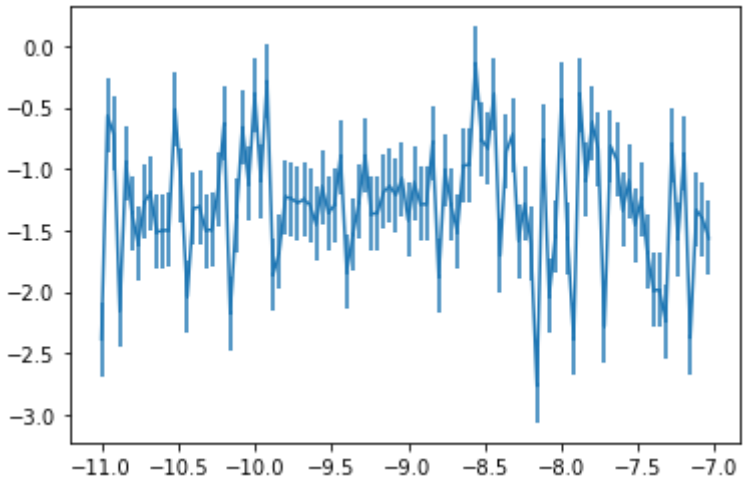
In [16]:
```
y_predict = result.stan_variable(var='y_predict')
```

In [61]:
```
plt.plot(y_predict[0])
plt.plot(y_predict[1])
plt.plot(y_predict[2])
plt.plot(y_predict[3])
plt.plot(y_predict[4])
plt.plot(y_predict[5])
mean_value = []
for i in y_predict:
    mean_value.append(np.mean(i))
plt.plot(mean_value[0:551]) #pink line
```

Out[61]:
```
[<matplotlib.lines.Line2D at 0x7f03ccf38970>]
```



In [43]:
```
ax = plt.gca()
ax.errorbar(x_predict[0:100], mean_value[0:100],0.3, 0.01)
plt.draw()
```



Task 2 Optimize hyperparameters

In [45]:
```
data2 = dict(N_obs = len(y_obs), x_obs = x_obs, y_obs=y_obs)

model2=CmdStanModel(stan_file='gaussian_process_optimize.stan')
result2=model2.optimize(data=data2, seed=5838298)
```

```
INFO:cmdstanpy:compiling stan file /home/kasia/Documents/DataAnalytics/La
b6/gaussian_process_optimize.stan to exe file /home/kasia/Documents/DataA
nalytics/Lab6/gaussian_process_optimize
INFO:cmdstanpy:compiled model executable: /home/kasia/Documents/DataAnaly
tics/Lab6/gaussian_process_optimize
WARNING:cmdstanpy:Stan compiler has produced 2 warnings:
WARNING:cmdstanpy:
--- Translating Stan model to C++ code ---
bin/stanc  --o=/home/kasia/Documents/DataAnalytics/Lab6/gaussian_process_
optimize.hpp /home/kasia/Documents/DataAnalytics/Lab6/gaussian_process_op
timize.stan
Warning in '/home/kasia/Documents/DataAnalytics/Lab6/gaussian_process_opt
imize.stan', line 3, column 1: 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
Warning in '/home/kasia/Documents/DataAnalytics/Lab6/gaussian_process_opt
imize.stan', line 16, column 28: cov_exp_quad
    is deprecated and will be removed in Stan 2.32.0. Use gp_exp_quad_cov
    instead. This can be automatically changed using the canonicalize fla
g
    for stanc

--- Compiling, linking C++ code ---
g++ -std=c++1y -pthread -D_REENTRANT -Wno-sign-compare -Wno-ignored-attri
butes      -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_mat
h/ -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    -DBOOST_DISABLE_ASSERTS
-c -Wno-ignored-attributes   -x c++ -o /home/kasia/Documents/DataAnalytic
s/Lab6/gaussian_process_optimize.o /home/kasia/Documents/DataAnalytics/La
b6/gaussian_process_optimize.hpp
g++ -std=c++1y -pthread -D_REENTRANT -Wno-sign-compare -Wno-ignored-attri
butes      -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_mat
h/ -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    -DBOOST_DISABLE_ASSERTS
-Wl,-L,"/home/kasia/.cmdstan/cmdstan-2.29.1/stan/lib/stan_math/lib/tbb" -
Wl,-rpath,"/home/kasia/.cmdstan/cmdstan-2.29.1/stan/lib/stan_math/lib/tb
b"      /home/kasia/Documents/DataAnalytics/Lab6/gaussian_process_optimiz
e.o src/cmdstan/main.o        -Wl,-L,"/home/kasia/.cmdstan/cmdstan-2.29.1
/stan/lib/stan_math/lib/tbb" -Wl,-rpath,"/home/kasia/.cmdstan/cmdstan-2.2
9.1/stan/lib/stan_math/lib/tbb"   stan/lib/stan_math/lib/sundials_6.0.0/l
ib/libsundials_nvecserial.a stan/lib/stan_math/lib/sundials_6.0.0/lib/lib
sundials_cvodes.a stan/lib/stan_math/lib/sundials_6.0.0/lib/libsundials_i
das.a stan/lib/stan_math/lib/sundials_6.0.0/lib/libsundials_kinsol.a  sta
n/lib/stan_math/lib/tbb/libtbb.so.2 -o /home/kasia/Documents/DataAnalytic
s/Lab6/gaussian_process_optimize
rm -f /home/kasia/Documents/DataAnalytics/Lab6/gaussian_process_optimize.
o

INFO:cmdstanpy:Chain [1] start processing
INFO:cmdstanpy:Chain [1] done processing
```

In [51]:
```
rho2 = result2.stan_variable(var='rho')
alpha2 = result2.stan_variable(var='alpha')
sigma2 = result2.stan_variable(var='sigma')
print(rho2)
print(alpha2)
print(sigma2)
```

```
1.13658
3.1811
0.196539
```

In [52]:
```
optimized_data = dict(N_predict = len(x_predict),
                  x_predict = x_predict,
                  N_obs = len(y_obs) ,y_obs = y_obs, observed_idx= observe

result3 = model.sample(data=data, chains = 4)
```

```
INFO:cmdstanpy:CmdStan start processing
chain 1 |          | 00:00 Status
chain 2 |          | 00:00 Status
chain 3 |          | 00:00 Status
chain 4 |          | 00:00 Status

INFO:cmdstanpy:CmdStan done processing.
```

In [66]:
```
y_predict2 = result.stan_variable(var='y_predict')
plt.plot(y_predict2[0])
plt.plot(y_predict2[1])
plt.plot(y_predict2[2])
plt.plot(y_predict2[3])
plt.plot(y_predict2[4])
plt.plot(y_predict2[5])
mean_value2 = []
for i in y_predict2:
    mean_value2.append(np.mean(i))
plt.plot(mean_value2[0:551]) #pink line
#plt.errorbar(x_predict[0:551], mean_value[0:551],0.3, 0.01)
```

Out[66]:
```
[<matplotlib.lines.Line2D at 0x7f03cc9b2af0>]
```