Python/MATLAB for Physics
USPAS February 2022
Homework 3 Assignment

February 11, 2022

Due: 10am cst, Tues Feb. 15th

Problem 3.1: Exponential Decay

For a sufficiently large ensemble of particles undergoing exponential decay, the population of particles over time follows

$$N(t) = N_0 \exp(-\lambda t)$$

for characteristic decay constant of λ. Although the average particle decay time is λ^{-1}, individual particles in the ensemble may decay earlier or later than that time. The rate of particle decay is given by

$$\frac{d}{dt}N(t) = -\lambda N(t)$$

The number of particles decaying at a particular point of time is proportional to the number of particles surviving at that point in time. Each individual's particle probability of decaying over a particular interval of time (if it hasn't already decayed) is proportional to the interval of time (for a sufficiently short period of time):

$$\begin{aligned} N(t+\Delta t) - N(t) &\approx -\lambda N(t)\Delta t \\ \frac{N(t+\Delta t) - N(t)}{N(t)} &\approx -\lambda \Delta t \\ P_{decay}(\Delta t) &\approx -\lambda \Delta t \end{aligned}$$

Program

Write a Python program to simulate 10,000 exponetical decays, in timesteps of 0.01, each with a characteristic decay constant of 1. Make a 1D histogram plot from 0 to 5 with bin widths of 0.1 showing the distribution of actual decay times.

Fit this histogram with an exponential curve and using an error of each bin of $1/\sqrt(n)$ (where n is the number of entries in the bin). Output the decay constant obtained by the expontial fit. Plot the exponential curve fit over the 1D histogram plot.

Problem 3.2: Accelerator Optics Matching

The accelerator transfer matrix M is used to map the incoming phase-space coordinate x_0, x_0' to the outgoing phase-space coordinates x_1, x_1' (i.e. after passing through the accelerator elements). This matrix M can also be used to derive another matrix M_β which described how the incoming TWISS parameters $\beta_{x0}, \alpha_{x0}, \gamma_{x0}$ map to the outcoming TWISS parameters $\beta_{x1}, \alpha_{x1}, \gamma_{x1}$.

If M represents a stable one-turn map or a superperiodic cell, then the incoming TWISS parameters must match the outgoing TWISS parameters. Only the true TWISS parameters will satisfy these conditions, and therefore the TWISS parameters can be derived from the properties of the transfer matrix. In an unstable periodic system or a transfer line, the incoming TWISS parameters are instead determined by initial conditions. When injecting from a transfer line into a ring, the TWISS parameters at the end of the transfer line ideally should match the TWISS parameters in the ring.

Prep

Move the file `beta_match.py` from the `homework` folder of the `USPAS_Python_2022` public Github repository to your local submodule (to the same directory where you will run your Python code).

The file `beta_match.py` is a small python script that takes four input floats representing the incoming TWISS values ($\beta_{x0}, \alpha_{x0}, \beta_{y0}, \alpha_{y0}$) and calculates the outgoing TWISS values. For example, a valid command to run the script could be

```
python beta_match.py 5.0 0.0 5.0 0.0
```

The script `beta_match.py` creates the file `betas_out` and populates it with the four outgoing TWISS values. The values are on a single line, separated by space, and given in the same order as the inputs ($\beta_{x1}, \alpha_{x1}, \beta_{y1}, \alpha_{y1}$).

Program

Write a Python program to identify which four TWISS input values will result in the same four output values, thus indicating matching conditions.

Write a objective function which takes four floats as arguments, runs `beta_match.py` with those four arguments, reads the output file `betas_out`, calculates the difference between the input arguments and the output values, and returns the sum of the square of the differences.

Then use `minimize` from `scipy.optimize` to iterate through the objective function many times until the optimization converges on matching values. You should either impose bounds that β_{x1} and β_{y1} are only positive, or take the absolute value of your inputs within the objective function. Your program should output the matching values.