Minimizing emulator uncertainty

Oleh Savchuk

October 31, 2024

Please visit: https://bandframework.github.io

Dense Nuclear Matter Equation of State from Theory and Experiments

IRL NPA (In2p3/FRIB), Oct 28th - Nov 1st 2024, FRIB, East Lansing, MI, USA

Bayesian statistics

- Bayes Theorem for posterior:

$$\mathrm{P}(\theta \mid X) = \frac{P(X|\theta) \;\; P(\theta)}{P(X)}$$

- Parameters θ, observations, obsevation given parameters, prior, probability of observation,

$$P(X|\theta) = \exp\left\{ -\frac{1}{2} \sum \frac{(X_i - X_i^{theory}(\theta))^2}{\sigma_i^2} - \sum \ln \sqrt{2\pi\sigma_i^2} \right\}$$

- Might require 10^5 evaluations at different parameters θ

Bayesian statistics

- Bayes Theorem for posterior:

$$\mathrm{P}(\theta \mid X) = \frac{P(X|\theta)\ P(\theta)}{P(X)}$$

- Parameters θ, observations, obsevation given parameters, prior, probability of observation,

$$P(X|\theta) = \exp\left\{-\frac{1}{2}\sum \frac{(X_i - X_i^{theory}(\theta))^2}{\sigma_i^2} - \sum \ln\sqrt{2\pi\sigma_i^2}\right\}$$

- Might require 10^5 evaluations at different parameters θ

Bayesian statistics

- Bayes Theorem for posterior:

$$\mathrm{P}(\theta \mid X) = \frac{P(X|\theta)\; P(\theta)}{P(X)}$$

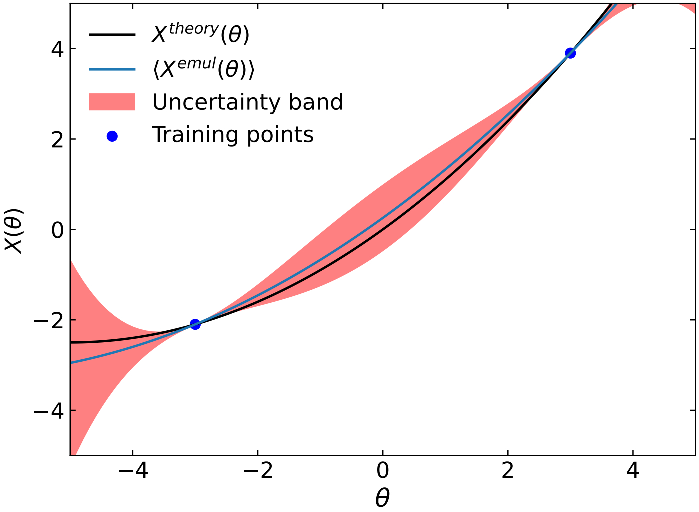
- Parameters θ, observations, obsevation given parameters, prior, probability of observation,

$$P(X|\theta) = \exp\left\{ -\frac{1}{2} \sum \frac{(X_i - X_i^{theory}(\theta))^2}{\sigma_i^2} - \sum \ln \sqrt{2\pi\sigma_i^2} \right\}$$

- Might require 10^5 evaluations at different parameters θ

Emulators

- Purpose: replace full-model (expensive) with emulators (cheap) trained on $\sim 10^2$ full-model runs.



Smooth emulator

- Based on Taylor series:

$$X^{emulator}(\theta) = \sum_{\{n\}} A_{\{n\}} \boxed{\sqrt{\frac{(n_1+\cdots+n_N)!}{n_1!\cdots n_N!}} \prod_i \left(\frac{\theta_i}{\Lambda}\right)^{n_i}} = \sum_i A_i \boxed{T_i(\theta)},$$

we call $(n_1 + \cdots + n_N) = rank$.

Smooth emulator

- Based on Taylor series:

$$X^{emulator}(\theta) = \sum_{\{n\}} A_{\{n\}} \sqrt{\frac{(n_1+\cdots+n_N)!}{n_1!\cdots n_N!}} \prod_i \left(\frac{\theta_i}{\Lambda}\right)^{n_i} = \sum_i A_i \, T_i(\theta),$$

  we call $(n_1 + \cdots + n_N) = rank$.

- with A_i being distributed according to:

$$P(\vec{A}) \sim \delta(A_i T_i(\theta_j) - X^{theory}(\theta_j)) \exp\left\{-\frac{A_i^2}{2\sigma_A^2},\right\}$$

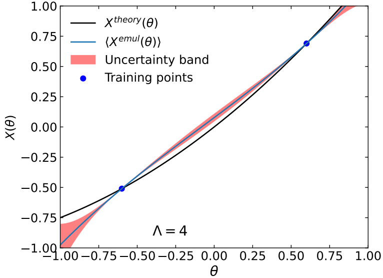
  and θ_j being training points.

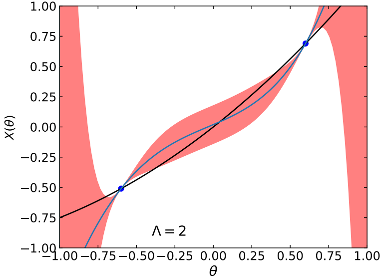
Λ smoothness parameter

$$X^{emulator}(\theta) = \sum_i A_i T_i(\theta) = \sum_{\{n\}} A_{\{n\}} \sqrt{\frac{(n_1 + \cdots + n_N)!}{n_1! \cdots n_N!}} \prod_i \left(\frac{\theta_i}{\Lambda} \right)^{n_i}$$

Λ smoothness parameter

$$X^{emulator}(\theta) = \sum_i A_i T_i(\theta) = \sum_{\{n\}} A_{\{n\}} \sqrt{\frac{(n_1 + \cdots + n_N)!}{n_1! \cdots n_N!}} \prod_i \left(\frac{\theta_i}{\Lambda} \right)^{n_i}$$





Including uncertainty from theory

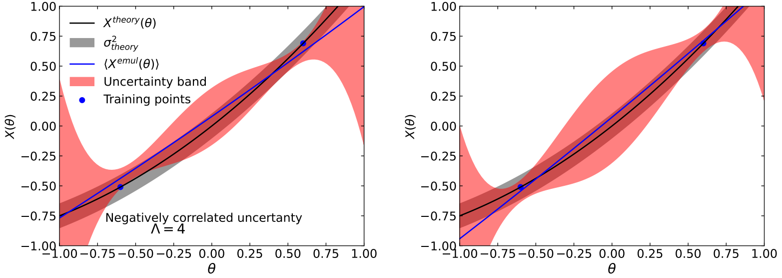
- We can incorporate theoretical model's noise into emulator:

$$\delta(X_j^{th} - X_j^{emu}) \rightarrow \frac{|\Delta|}{(\sqrt{2\pi})^{N_{train}}} \exp\left\{-\frac{1}{2}(X_j^{th} - X_j^{emu})\Delta_{jk}^{-1}(X_k^{th} - X_k^{emu})\right\}$$

Including uncertainty from theory

- We can incorporate theoretical model's noise into emulator:

$$\delta(X_j^{th} - X_j^{emu}) \to \frac{|\Delta|}{(\sqrt{2\pi})^{N_{train}}} \exp\left\{-\frac{1}{2}(X_j^{th} - X_j^{emu})\Delta_{jk}^{-1}(X_k^{th} - X_k^{emu})\right\}$$



Parameters

- σ_A^2, Λ can be obtained my maximizing probability:

$$P(X^{th}|\sigma_A,\Lambda) = \int dA_i P(A) = \frac{\sigma_A^{N_{train}} \exp\left\{-\frac{1}{2\sigma_A^2} X_j^{th}(O_{jk} + \frac{\Delta_{jk}}{\sigma_A^2}) X_k^{th}\right\}}{\sqrt{\det(O_{jk} + \frac{\Delta_{jk}}{\sigma_A^2})}},$$

with $O_{lm} = T_i(\theta_l) T_i(\theta_m)$.

Parameters

- σ_A^2, Λ can be obtained my maximizing probability:

$$P(X^{th}|\sigma_A, \Lambda) = \int dA_i P(A) = \frac{\sigma_A^{N_{train}} \exp\left\{-\frac{1}{2\sigma_A^2} X_j^{th}(O_{jk} + \frac{\Delta_{jk}}{\sigma_A^2}) X_k^{th}\right\}}{\sqrt{\det(O_{jk} + \frac{\Delta_{jk}}{\sigma_A^2})}},$$

  with $O_{lm} = T_i(\theta_l) T_i(\theta_m)$.

- Emulator predictions:

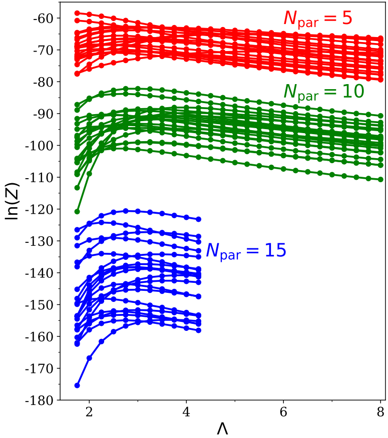
$$\langle X^{em}(\theta)\rangle = \sum_i \langle A_i\rangle T_i(\theta)$$

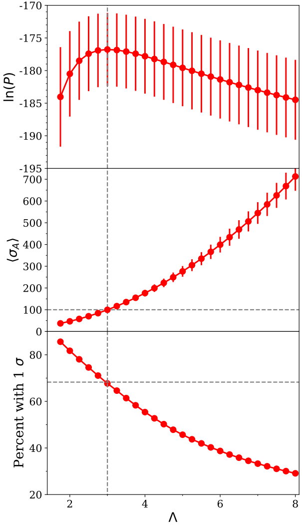
$$\langle \Delta X^{em}(\theta_1) \Delta X^{em}(\theta_2)\rangle = \sum_{ij} \langle \Delta A_i \Delta A_j\rangle T_i(\theta_1) T_j(\theta_2)$$

  can be solved for analytically.

Emulator tests

- Generating random models from σ_A^2 and Λ and using them in log-likelyhood estimation leads to correct parameter reconstruction.





Emulator Uncertainty

- At a point θ:

$$\sigma^2(\theta) = \langle \Delta X^{em}(\theta)\Delta X^{em}(\theta)\rangle = \sigma_A^2(T_i(\theta)T_i(\theta) - (O_{lm} + \sigma_A^{-2}\Delta_{lm})^{-1}S_l(\theta)S_m(\theta)),$$

where $S_l(\theta) = T_i(\theta_l)T_i(\theta)$.

- $\sigma^2(\theta_p) = 0$, at the training point θ_p.

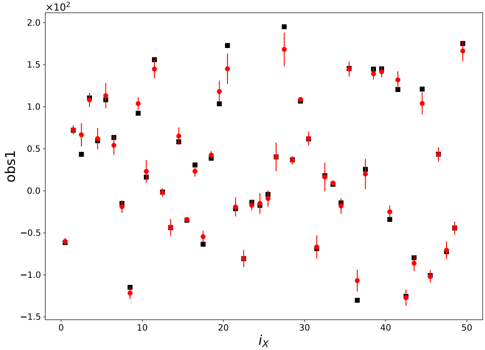
Emulator Uncertainty

- At a point θ:

$$\sigma^2(\theta) = \langle \Delta X^{em}(\theta) \Delta X^{em}(\theta) \rangle = \sigma_A^2 (T_i(\theta) T_i(\theta) - (O_{lm} + \sigma_A^{-2} \Delta_{lm})^{-1} S_l(\theta) S_m(\theta)),$$

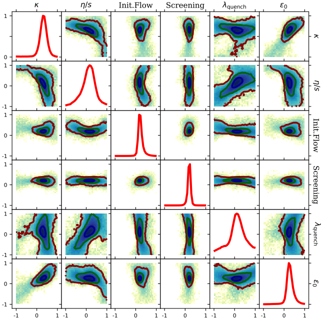
where $S_l(\theta) = T_i(\theta_l) T_i(\theta)$.

- $\sigma^2(\theta_p) = 0$, at the training point θ_p.
- Smooth emulator is good for "smooth" functions. Prediction of uncertainty improved compared to gaussian process.



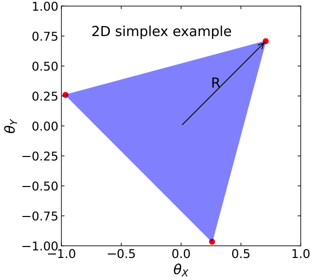
Current applications

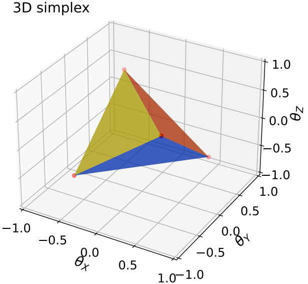
- For Charge Balance Functions: **OS, Scott Pratt (MSU/FRIB) and Andrew Gordeev (Duke)**
- For Jet Physics: **Christal Martin (Tennessee)**
- For BES dynamics: **Syed Afrid Jahan and Chun Shen (Wayne State)**
- Smooth emulator is written in $C++$ (fast) and soon available as Python package (easy to use)
- Available at github.com/bandframework



Simplex algorithm for choosing training points

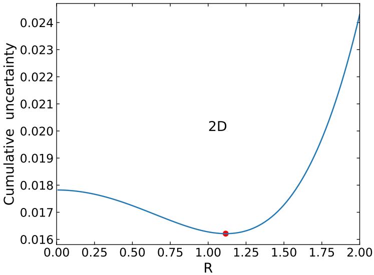
- Goal:
  1. Find optimum locations of training points.
  2. Should span large portion of space.
- Linear fit requires $N_{train} = N + 1$.
- Quadratic fit additional $N(N+1)/2$ training points.
- Simplex: $N + 1$ equally spaced from each other points:
  - 1D. Two points;
  - 2D. Equilateral triangle;
  - 3D. Tetrahedron;

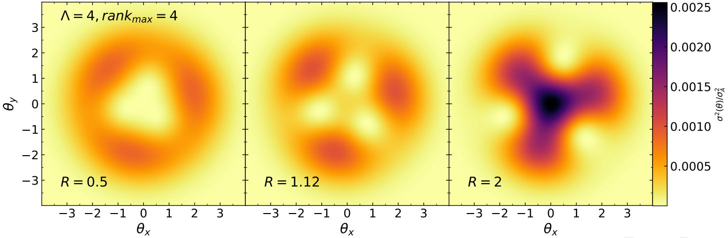




Optimizing size of simplex

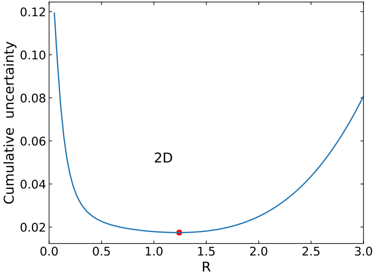
$$CU = \int d\theta \frac{\sigma^2(\theta)}{(2\pi)^{N/2}\sigma_A^2} \exp\left\{-\frac{\theta^2}{2}\right\}$$

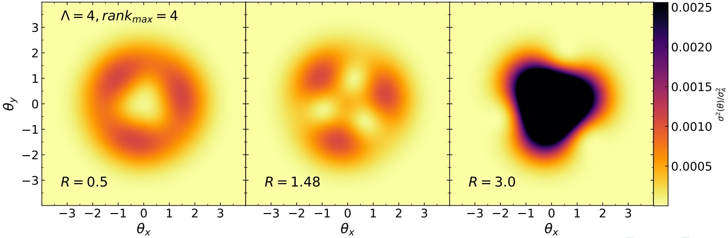




Optimizing with Uncertainty

$$\Delta_{lm} = 0.001\sigma_A^2\delta_{lm}$$

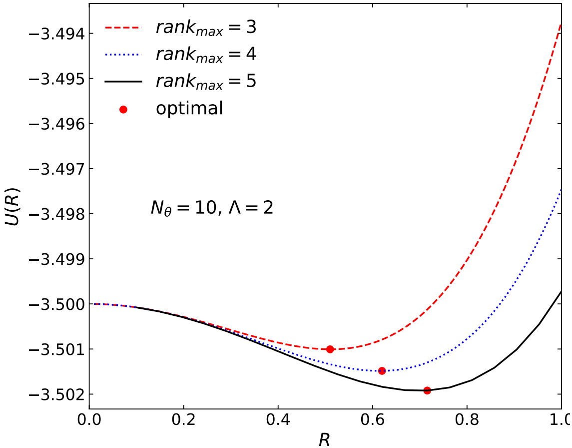




Higher dimensions

- R-dependent part of uncertainty:

$$U(R) = -O_{lm}^{-1} \int \frac{d\theta}{(\sqrt{2\pi})^N} exp^{\frac{-\theta^2}{2}} S_l(\theta) S_m(\theta)$$



Summary

1. Smooth emulator is easy and fast to train.
2. Smooth Emulator good choice for Smooth model, i.e. does not require high orders in Taylor expansion.
3. Does good job at predicting uncertainty.
4. Can be incorporated into Python, available to import as python module (contact me to try).
5. Simplex is a good starting point for training points.

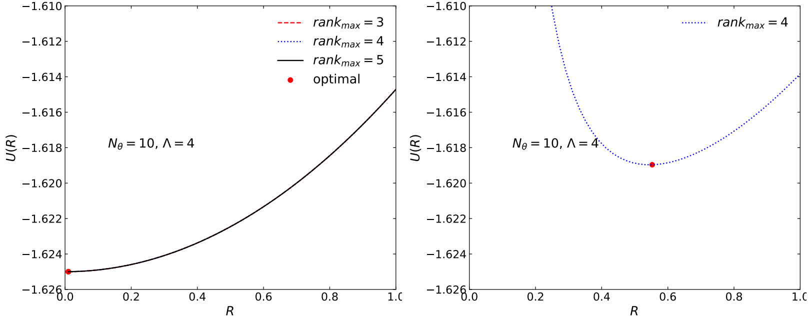
Summary

1. Smooth emulator is easy and fast to train.
2. Smooth Emulator good choice for Smooth model, i.e. does not require high orders in Taylor expansion.
3. Does good job at predicting uncertainty.
4. Can be incorporated into Python, available to import as python module (contact me to try).
5. Simplex is a good starting point for training points.

Thank you for attention!

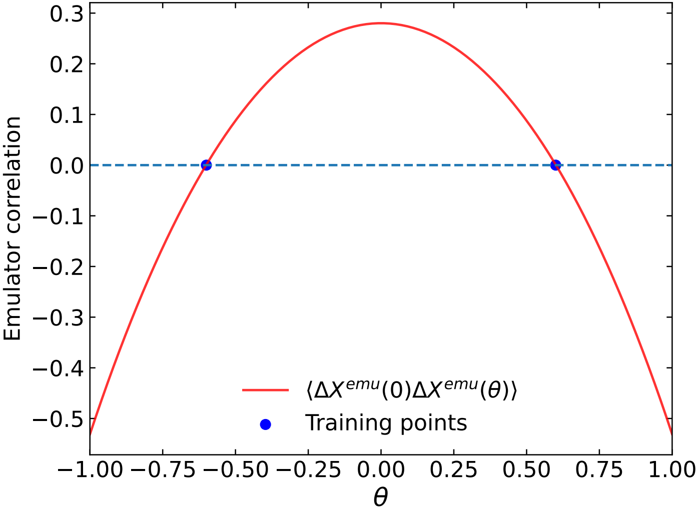
Very smooth functions

- For a smoother function smaller radii is preferred.
- Increase in number of parameters also leads to smaller radii.
- Maximal rank on the other hand leads to increase in radii.
- Uncertainty in theoretical model leads to large values of radii and makes minimum more pronounced.



Correlation function

- Correlation functions have a different structure compared with Gaussian Process emulators.



Log plots of uncertainty

- Training points are easier to spot.