Bayes' theorem application to physical model parameter extraction

Weiyao Ke

Modified from JetScape Winter School 2019 talk by Jake Coleman & Weiyao Ke

Dec 3, 2019

1. Bayes' theorem

2. Complex model: high dimensional input and output

3. Application: a toy model for jet-quenching

4. Summary and discussion

Bayes' parameter extraction

The problem:

1. Given a model $\mathcal{M}$: compute quantities $\mathbf{y}$ with input parameters $\mathbf{x}$.
2. Given a prior belief of $\mathbf{x}$' true value's distribution $P_0(\mathbf{x}_{\rm true})$
3. Given observations $\mathbf{y}_{\rm exp}$.

!! Ask for the updated probability distribution of $\mathbf{x}_{\rm true}$: $P(\mathbf{x}_{\rm true})$.

$$\text{Bayes' theorem:} \qquad \underbrace{P(\mathbf{x}_{\rm true}|\mathcal{M}, \mathbf{y}_{\rm exp})}_{\text{Posterior}} = \frac{\overbrace{P_L(\mathbf{y}_{\rm exp}|\mathcal{M}, \mathbf{x}_{\rm true})}^{\text{Likelihood}}\overbrace{P_0(\mathbf{x}_{\rm true})}^{\text{Prior}}}{\underbrace{\int P_L(\mathbf{x})P_0(\mathbf{x})d\mathbf{x}}_{\text{Normalization (evidence)}}}$$

Often the form P_L is unknown. Commonly assumed to take the form:

$$\ln P_L = C - \frac{1}{2}\Delta\mathbf{y}\Sigma^{-1}\Delta\mathbf{y}^T, \quad \Delta\mathbf{y} = \mathbf{y}_{\rm exp} - \mathbf{y}(\mathbf{x};\mathcal{M}), \Sigma: \text{Contains uncertainties}$$

A toy model: flip coin

The problem:

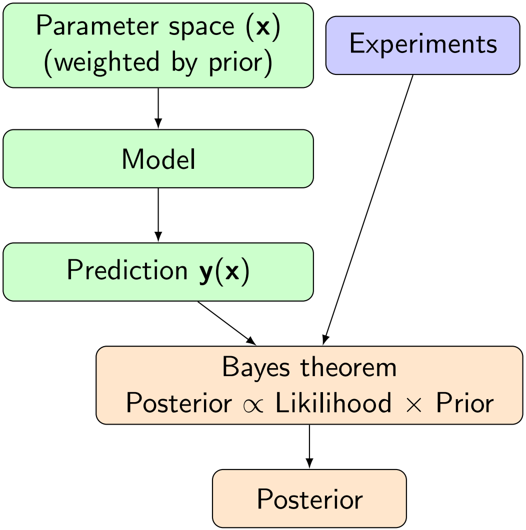
- A coin flip experiments gets $m = 7$ head out of $n = 10$ trials.
- What is the probability θ to get a head each time.

The model:

- Outcome follows a binomial distribution. We know the exact likelihood function $P_L(m \text{ out of } n|\theta) = C_n^m \theta^m (1-\theta)^{n-m}$.
- Try different prior: $P_0(\theta) \propto \theta^{\alpha-1}(1-\theta)^{\beta-1}$

The resulting inference on p:

- Posterior probability distribution $P(\theta) \propto \theta^{m+\alpha-1}(1-\theta)^{n-m+\beta-1}$
- Mean $\pm$ Std: $\theta = \frac{m+\alpha}{n+\alpha+\beta} \pm \sqrt{\frac{(m+\alpha)(n-m+\beta)}{(n+\alpha+\beta)^2(n+\alpha+\beta+1)}}$ ($P(\theta)$ has all the information).
- How do α, β change the inference?


Parameter space ($\mathbf{x}$)
(weighted by prior)
Experiments
Model
Prediction $\mathbf{y}(\mathbf{x})$
Bayes theorem
Posterior $\propto$ Likilihood $\times$ Prior
Posterior

For complex model, it is impossible to evaluate outcome at arbitrary input

Computationally intensive to evaluate the full model.

- For example, 2+1D event-by-event viscous-hydro based HIC simulation $\sim$ 5-10 events/h. To compute observables with one set of input take $\mathcal{O}10^4$ minimum biased events $\rightarrow$ 1-2kh
- Compute at finite number of input points and interpolate the function $\mathbf{y}(\mathbf{x}; \mathcal{M})$

Two problems of interpolation:

1. How to efficiently interpolate high-dimensional input $\mathcal{O}(10)$?
2. How to efficiently interpolate a function with vector output.
   - For example $\mathbf{y} = (\langle E_T \rangle, dN_{\rm ch}/d\eta \langle p_T \rangle, v_2, v_3, v_4) \times [8 \text{ centrality}] \times [2 \text{ collision systems}]$ gives an 80-dimensional output at each input.

Latin hyper-cube sampling: optimized samples of high dimensional parameter space

Assume the model is well-behaved in the parameter space.

- Grid samples require too many samples $N \propto n^d$
- Random samples result in over populated regions and large gap.

Latin hyper-cube sampling (LHS): random sampling with

- Marginalized distribution is uniform.
- Maximize the minimum distance between two samples.
- Usually $N \propto d$

Gaussian emulator: fast interpolator

Given y values at finite number of parameter sets $\mathbf{x}_i$. Gaussian process is:

- A non-parametric interpolation.
- y^* at a new point $\mathbf{x}^*$ are inferred from its correlation with all other points (x_i).
- It also estimates the interpolation uncertainty!

How does it work:

- Assume $y(\mathbf{x}^*)$, $y(\mathbf{x}_i)$ form a multi-variated Gaussian distribution (centered):

$$\begin{bmatrix} y_* \\ y_i \end{bmatrix} \sim \mathcal{N}\left(\mu = 0, \mathrm{cov} = \begin{bmatrix} k(x^*, x^*), k(x^*, x_i) \\ k(x_i, x^*), k(x_i, x_j) \end{bmatrix}\right), \quad k(\cdot,\cdot) \text{ is a kernel function}$$

- $P(y^*)$ is simply the conditional probability $P(y^*) = \left.\frac{P(y, y_i)}{\int P(y, y_i) dy}\right|_{y_i = y(\mathbf{x_i})}$,

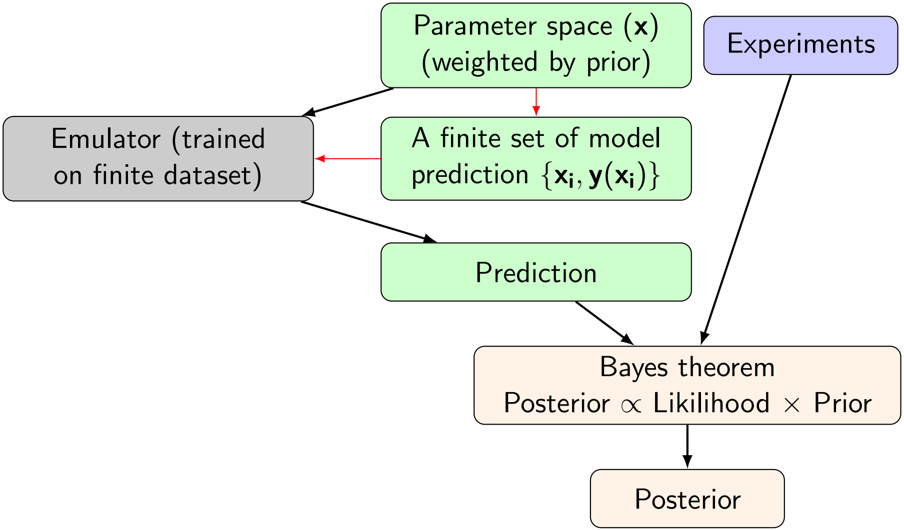
$$y^* \sim \mathcal{N}\left(\mu = k(x^*, x_i)k^{-1}(x_i, x_j)y_j, \mathrm{cov} = k(x^*, x^*) - k(x^*, x_i)k^{-1}(x_i, x_j)k(x_j, x^*)\right)$$

- Question: what happens when $x^* \to x_i$

Gaussian emulator: kernel function and training

One do have freedom in chosen the kernel function and its "hyperparameters", a common choice is a Gaussian-like correlation function:

- $k(\mathbf{x}_i, \mathbf{x}_j) = \sigma^2 \exp(-\frac{1}{2}|(\mathbf{x}_i - \mathbf{x}_j)/\mathbf{L}|^2)$: two point correlation $\langle y(\mathbf{x_i}) y(\mathbf{x_j}) \rangle$
- $\sigma, \mathbf{L}$ called hyperparameters: They are, in principle, unknown. But one can use a set of "optimized values" by balancing the quality of the fit and the complexity of the emulator $\rightarrow$ training process
- Question: what happens if L is too small / too large.


Parameter space ($\mathbf{x}$) (weighted by prior)
Experiments
Emulator (trained on finite dataset)
A finite set of model prediction $\{\mathbf{x_i}, \mathbf{y}(\mathbf{x_i})\}$
Prediction
Bayes theorem
Posterior $\propto$ Likilihood $\times$ Prior
Posterior

Principal component analysis: data reduction for vector function

Suppose one has computed a list of observables at n design point,

$$\mathbf{x_1} \rightarrow \quad \mathbf{y}_1 = [y_1(\mathbf{x_1}), \cdots, y_m(\mathbf{x_1})]$$
$$\cdots$$
$$\mathbf{x_n} \rightarrow \quad \mathbf{y}_n = [y_1(\mathbf{x_1}), \cdots, y_m(\mathbf{x_n})]$$

Inefficient to construct m emulator for each quantity. Data has intrinsic correlation.

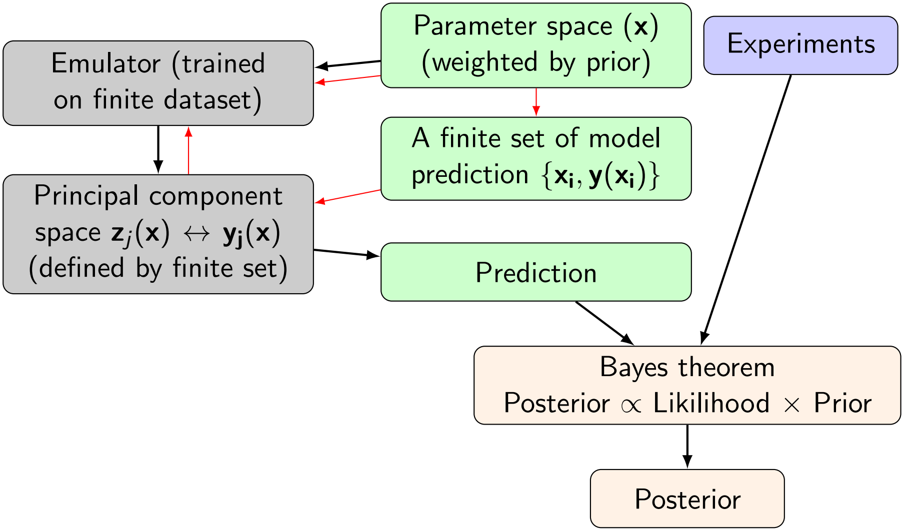
- For example, if y_i and y_j has a strong linear correlation, then there is effectively only one degree of freedom $(z_1(\mathbf{x}) = ay_1(\mathbf{x}) + by_2(\mathbf{x}), z_2 = by_1 - ay_2 \ll \mathcal{O}(z_1))$ instead of two.
- $(y_1(\mathbf{x}), y_2(\mathbf{x})) \xrightarrow{\text{Change basis}} (z_1(\mathbf{x}), z_2(\mathbf{x})) \xrightarrow{\text{Dim reduction}} (z_1(\mathbf{x}))$
- Principal component analysis (PCA): the statistical tool to generalize this simple example.

Principal component analysis: data reduction for vector function

- PCA: systematically construct the new basis by diagonalize the empirical covariance matrix

$$
\begin{aligned}
C_{ij} &= \frac{1}{n}\sum_{k=1}^{n} \underbrace{\frac{y_i(\mathbf{x_k}) - \bar{y}_i}{\sigma_i}}_{\text{Standardlized data } Y_{ik}} \frac{y_j(\mathbf{x_k}) - \bar{y}_j}{\sigma_j} = \frac{1}{n}\mathbf{Y}\mathbf{Y}^{\mathbf{T}} \\
\mathbf{C} &= \mathbf{V}\mathrm{Diag}\{\lambda_1, \lambda_2, \cdots\}\mathbf{V}^T, \quad \lambda_1 > \lambda_2 > \cdots
\end{aligned}
$$

- The new basis becomes $z_i(\mathbf{x}) = V_{ij}^T y_j(\mathbf{x})$, $i = 1, 2, \cdots m$, with variance λ_i.
- Not all of them are important, one usually truncates at $N \ll m$ and construct emulators for the first N most important principal component.
- Given $z_{i\leq N}$, y_i can be reconstructed $y_i = V_{ij\leq N} z_{j\leq N}$.


Parameter space ($\mathbf{x}$) (weighted by prior)
Experiments
Emulator (trained on finite dataset)
A finite set of model prediction $\{\mathbf{x_i}, \mathbf{y}(\mathbf{x_i})\}$
Principal component space $\mathbf{z}_j(\mathbf{x}) \leftrightarrow \mathbf{y_j}(\mathbf{x})$ (defined by finite set)
Prediction
Bayes theorem
Posterior $\propto$ Likilihood $\times$ Prior
Posterior

A toy-model example

A toy-model of jet-quenching

- Initial spectrum

$$\frac{dN_0}{dp_T} \propto \frac{p_T}{\left(3^2 + p_T^2\right)^3}$$

- Energy loss ΔE follows a Γ-distribution,

$$\begin{aligned} P(\Delta E) &\propto \Delta E^{\mu^2/\sigma^2 - 1} e^{-\mu \Delta E/\sigma^2} \\ \mu &= A\sqrt{p_T}, \sigma = B\mu \end{aligned}$$

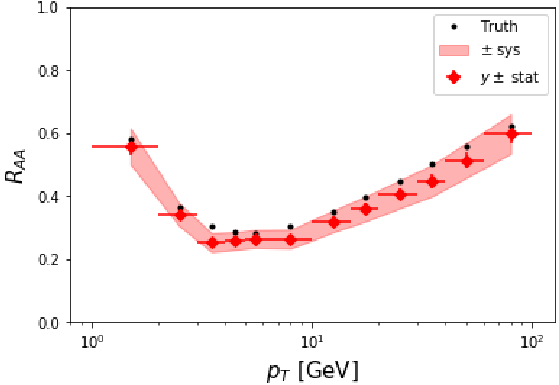
- The quenched spectrum

$$\frac{dN_1}{dp_T}(p_T) = \int \frac{dN_0}{dp_T}(p_T + x) P(x) dx$$

A toy-model example

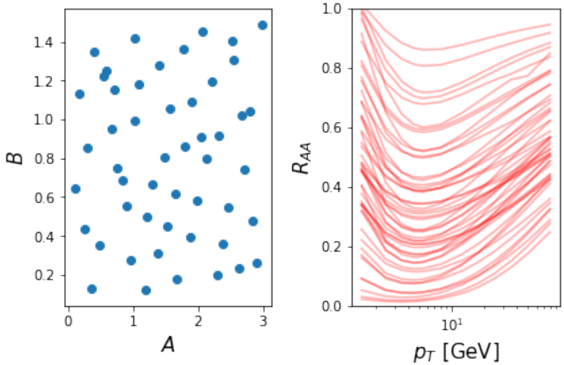
- Assume the model is perfect and the truths are: $A = 1.0, B = 0.5$.
- A measurement with finite statistics (5%) and systematic bias (10%).

$$y_{\rm exp} \approx y_{\rm true} \pm \sigma_{\rm stat} \pm \sigma_{\rm sys}$$



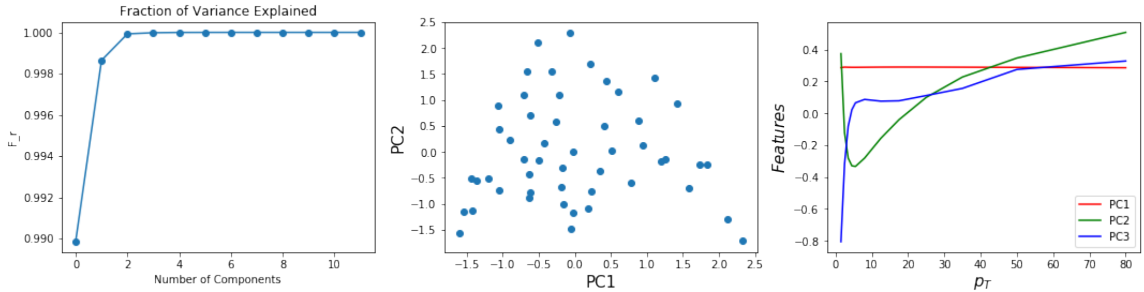
Toy model: make design

An 50-point design with $A \in [0.05, 3]$ and $B \in [0.05, 1.5]$.
Model calculations of R_{AA} widely spread between 0 and 1.



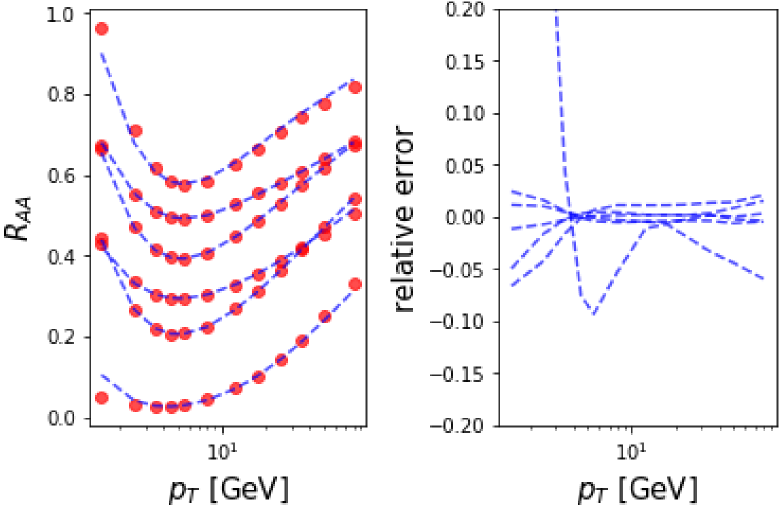
Toy model: PCA

The first two PCs account for more than 99.8% of the data variance.
PC1 is an overall shift, PC2 and PC3 capture the shapes.



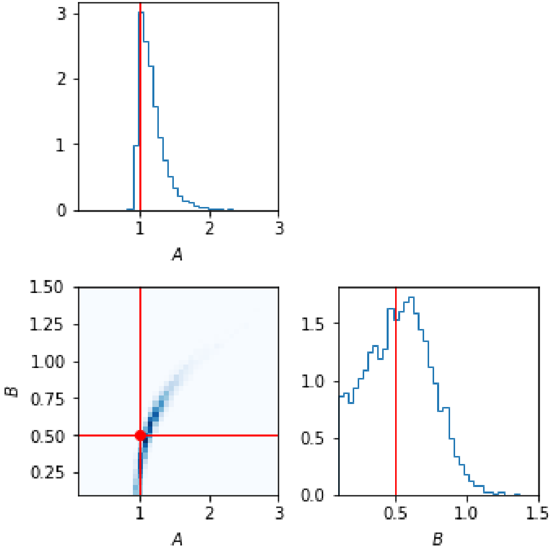
Toy model: Validation

Compare GPs' predictions to model calculations at new parameter sets. Relative error on the right.

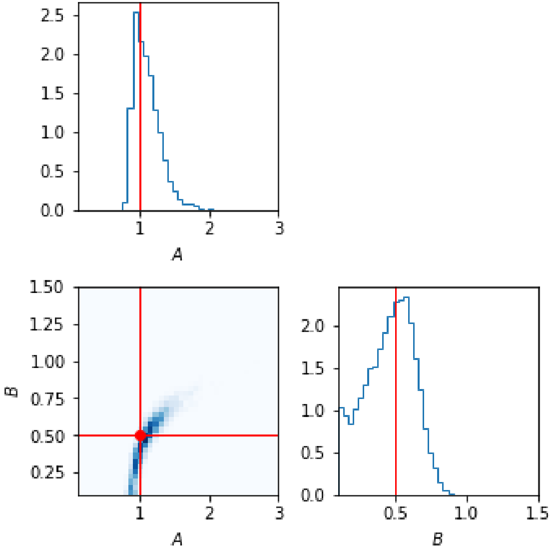


Toy model: Posterior for parameters

Using uncorrelated sys error



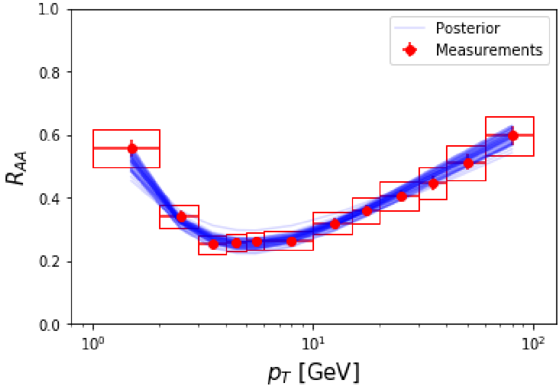
Using correlated sys error



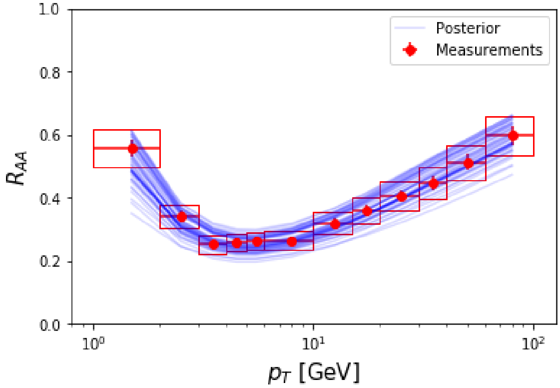
* Mistreating correlated uncertainty may bias the credible region.

Toy model: Posterior predictions

Using uncorrelated sys error



Using correlated sys error



* Mistreating correlated uncertainty may lead to overconfident prediction uncertainties.

Summary and discussion

- Pick design points (use LHS) $\mathbf{x}_i$, $i = 1, \cdots n$, compute the full model at these finite number of design points $y_j(\mathbf{x}_i)$.
- Construct principal components $z_j(\mathbf{x})$ and keep the first N components.
- Train N Gaussian process emulator on the $j \leq N$ PC to learn the mapping so that $\mathrm{GP}_j(\mathbf{x}) \sim \mathbf{x} \rightarrow \mathbf{z_j}(\mathbf{x})$
- For any $\mathbf{x}$, the prediction can be obtained by $y_j(\mathbf{x}) = V_{jk\leq N}\mathrm{GP}_{k\leq N}(\mathbf{x})$
- Construct the likelihood function, and Posterior $=$ Likelihood $\times$ Prior.
- Use MCMC to marginalize the high-dimensional posterior distribution.

For discussion:

- Emulator is not perfect. Make the function to be learned as simple as possible! (Input parameter space transformation).
- How to properly construct the co-variance matrix if the likelihood function is not known exactly?
- Model uncertainty.