“Theory API” discussion

Steven Gardiner

Fermilab Generator Tools Workshop

9 January 2020

Disclaimers

- Nothing in this talk is intended to mandate what generator groups should do
- I do not speak for GENIE (or any other generator) in this talk, but I hope to be fair to all points of view, including GENIE's
- My interpretation of my role: "Go think about it and talk to others, then give a report to (hopefully) spur useful community discussion"
- No "silver bullet" or "one size fits all" solution

What does “theory API” mean?

- **A possibly useful shorthand, but . . .**
  - Scope of the problem to be solved is not purely technical
  - Defining code interfaces is probably not the hardest technical challenge

- Some of the important issues in the discussion are more strategic / organizational / sociological in nature

- **Alternate framing of the topic:** “Are there ways we can make it easier / faster / more efficient / better incentivized to incorporate new theory models into our generators?”

- **Another formulation:** “For the models we plan to add, are there strategies we can use to get the most return on our investment of effort?”

  - “Recycling” code (where appropriate) for similar models
  - Interfacing to existing (and validated) theory code instead of trying to reinvent it
  - General techniques / combined efforts to solve common problems

Some healthy skepticism (1)

- Despite “pain points,” the status quo exists for good reasons

  - When their generator changes, experiments notice (and will complain) if there are problems

  - Enhancements need to be added carefully to avoid breaking existing code that experiments rely on

  - New models will need to be maintained and evolved to keep up with the rest of the code

  - Careful validations need to be done to ensure the model implementation is correct, that it works under all circumstances, etc.

- If we’re going to propose changes, we need to be clear about the relative costs and benefits to how we do things now

Some healthy skepticism (2)

- Various points of view exist on the usefulness of this exercise

  - What is the opportunity cost of the “theory API” effort? Are there other priorities that are more worth our time?

  - Are there ways of defining the problem that will focus any invested effort in a productive direction?

- Concluding that the problem is too ill-defined or hard is legitimate, but . . .

  - “Pain points” suggest that we shouldn’t dismiss the possibility out of hand

  - “Baby steps” in this direction (e.g., tables for Valencia MEC) have yielded something helpful that experiments are actively using

- **Limiting the scope seems useful**
  - Prioritize more limited strategies for specific goals that we care about

Some takeaways from previous ECT* discussion (June 2019)

- Clear preference for opt-in participation and “generator implements a specification” development model

  - As opposed to a shared community tool that “talks" to new theory models and to different generators

- Modern compilers make the technical parts of mixing languages (C, C++, and Fortran) manageable

- “Plugin" approach for model code distribution proposed

  - Fermilab prepares a website to host external theory contributions
  - Packaged with installation instructions for use with compatible generators
  - Each model labeled with a **version number** and a **paper to be cited**

- Summary slides available **here**

Implementing a new cross section calculation

- Requirements for, e.g., adding a new FSI model will be different

- The usual caveats against creating a “Franken-model” apply

- **What does the generator need to do for this new model?**
  - **Compute the differential cross section**
  - Integrate it to get the total cross section
  - **Sample kinematic variables**
  - If the model does not predict all outgoing 4-momenta, fill in the missing information somehow
  - Simulate FSIs
  - **Vary free parameters / reweight for cross section systematics studies**

- Essential theory input is the differential cross section $d^n\sigma/d\mathbf{X}$ or the ingredients needed to compute it

Possible strategies

- These differ in a number of ways, including the level of effort required by theory contributors

Implementation

- New code in the generator (the norm)
  - Often (at least in GENIE's case) via translation of original Fortran to C++
- "Lightweight generator"
- Tables for interpolation
- Direct interface to theory code
  - Possibly with a wrapper to interface with a different programming language (C++ to/from Fortran)

Physics model input

- $d^n\sigma/d\mathbf{X}$ itself
- The hard parts (e.g., the hadronic tensor $W^{\mu\nu}$)

“Lightweight generator” approach

- Dedicated generators are sometimes written to study a particular process
  - Distinct from effort to provide comprehensive solutions

- Potentially interesting physics can be done if we support the ability of our tools to interface with these codes

  - If well-designed, a flux driver that can “talk to” GENIE, GiBUU, NuWro, and NEUT should be able to work with smaller generators

- Already discussed earlier in the workshop
  - Worth pointing out as one route toward more model availability

- May be particularly useful for physics topics unlikely to be a priority for neutrino generator developers themselves
  - E.g., some BSM models

- Asks a lot in terms of theorist effort

Table-based example: hadronic tensor

- Use a very general form to provide differential prediction for lepton kinematics
  - Hadronic tensor pre-calculated and tabulated for efficient evaluation
  - Elements expressed as a function of

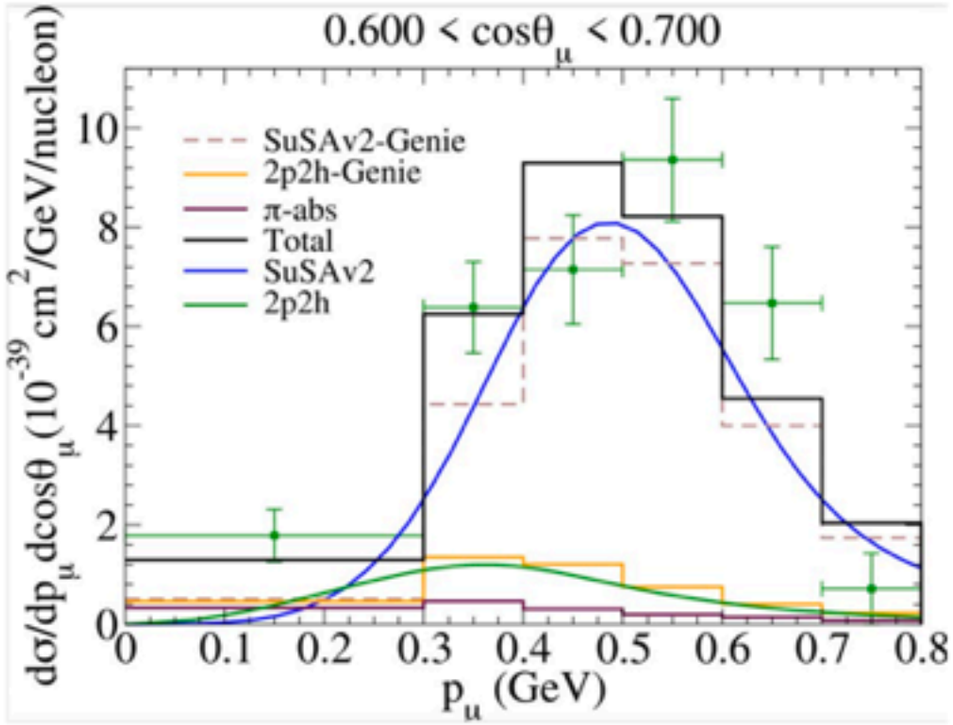
$$\omega = E_\ell - E_{\ell'}$$

$$q = |\mathbf{p}_\ell - \mathbf{p}_{\ell'}|$$

  - 5 elements at each (ω, q) grid point
  - Other variables integrated out
    → inclusive prediction only

$$\frac{d^2\sigma}{dE'_\ell\, d\cos(\theta'_\ell)} = \frac{|\boldsymbol{k}'|}{|\boldsymbol{k}|}\frac{G_F^2}{2\pi} L_{\mu\nu} W^{\mu\nu}$$

SuSAv2 prediction compared to T2K data



SuSAv2 implementation note

- Pros and cons discussed in detail at **ECT* workshop** in June 2019 (slides by S. Dolan)
- Valencia MEC available using this method in GENIE, NEUT, NuWro
- SuSAv2 also expected for GENIE v3.2 (other generators?)

Table-based approaches: hadronic tensors

- More complicated tables or sub-tables would be needed for more exclusive predictions

  - E.g., STA calculation (S. Pastore) can provide (ω, q)-dependent distributions of nucleon pair momenta ("response densities")

- Generator cannot vary the underlying parameters used to create the table
  - Additional tables (or a transformation rule) needed to apply variations

- Number and size of tables can increase to the point of being impractical
  - Different nuclei (not a problem for SuSAv2)
  - Different processes (or contributions to the same process)
  - More kinematic variables

- That being said, it's clear from real-world examples that theorists can provide this sort of input relatively painlessly

Directly interfacing with theorists' code

- **Benefits**
  - Some (not all) of the validation work has already been done for us
  - Inter-generator cooperation easier (common interface to standalone code)
  - More flexible (Ulrich: and faster!) than tables

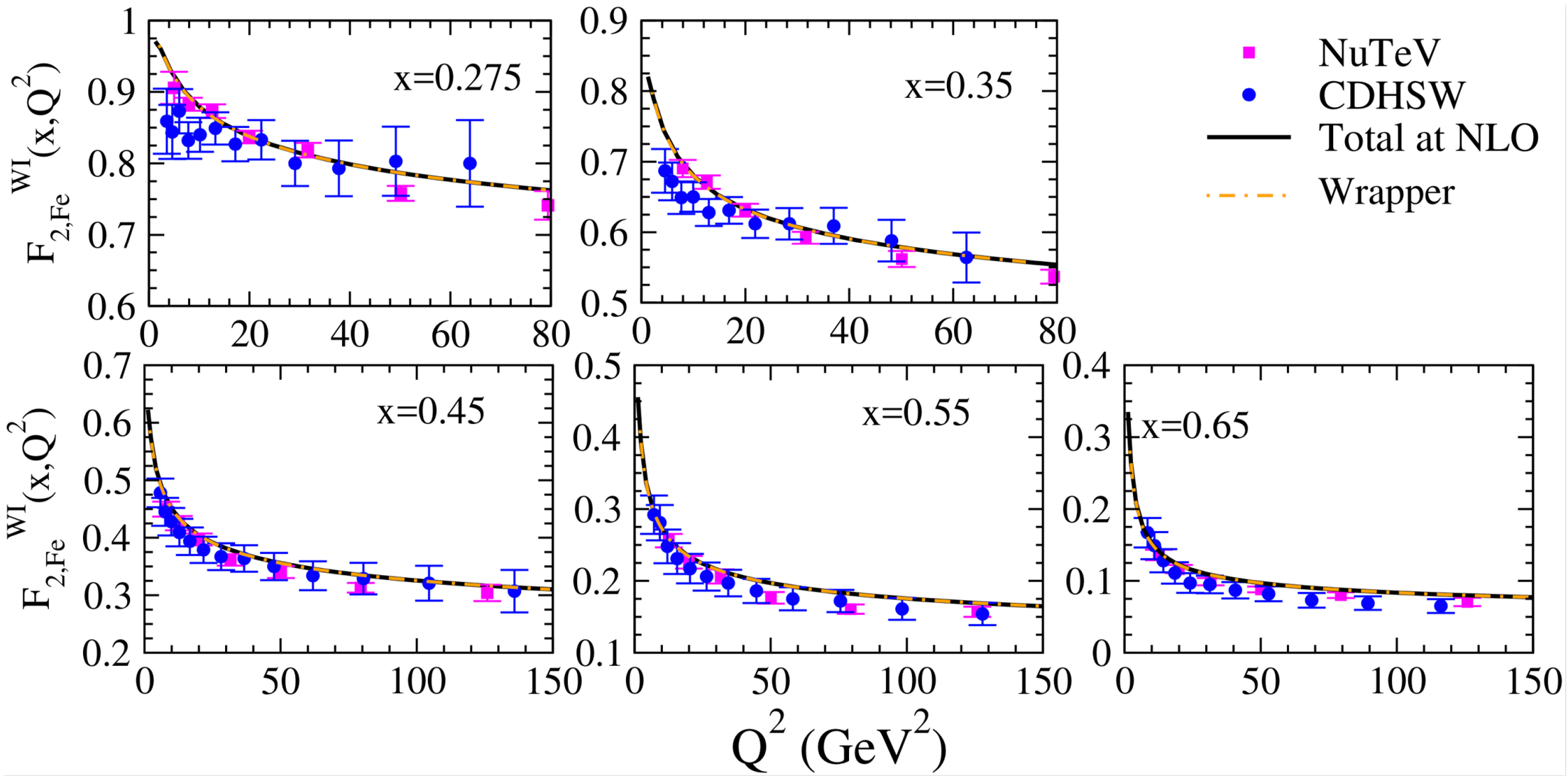
- **Drawbacks**
  - Build / installation process can be more cumbersome
  - Generators do not own the code
  - Standalone theory codes may not conform well to generator requirements:
    - Coding standards / conventions
    - Performance
    - Configuration (e.g., hard-coded parameters)

Fortran wrapper example

- $F_2(x, Q^2)$ structure function from H. Haider

  - Plot shows Fortran original calculation (black) and GENIE wrapper (dashed yellow)

  - Original calculation from **H. Haider et al., PRC84, 054610 (2011)**



Fermilab

Sampling

- Even with a good method for calculating $d\sigma/d\mathbf{X}$, other important work remains to be done

- Choosing event kinematics can be a tricky problem

- Event weights are one way to take care of this

  - Throw $\mathbf{X}$ uniformly over the phase space, weight by $d\sigma/d\mathbf{X}$

  - Hard to use in some cases (e.g., coupling with beam simulation)

  - Potentially lots of time spent in the tails of the distribution

- Generating unweighted events has its own set of challenges
  - Finding the maximum of $d\sigma/d\mathbf{X}$ (if using rejection sampling)
  - Multivariate distributions

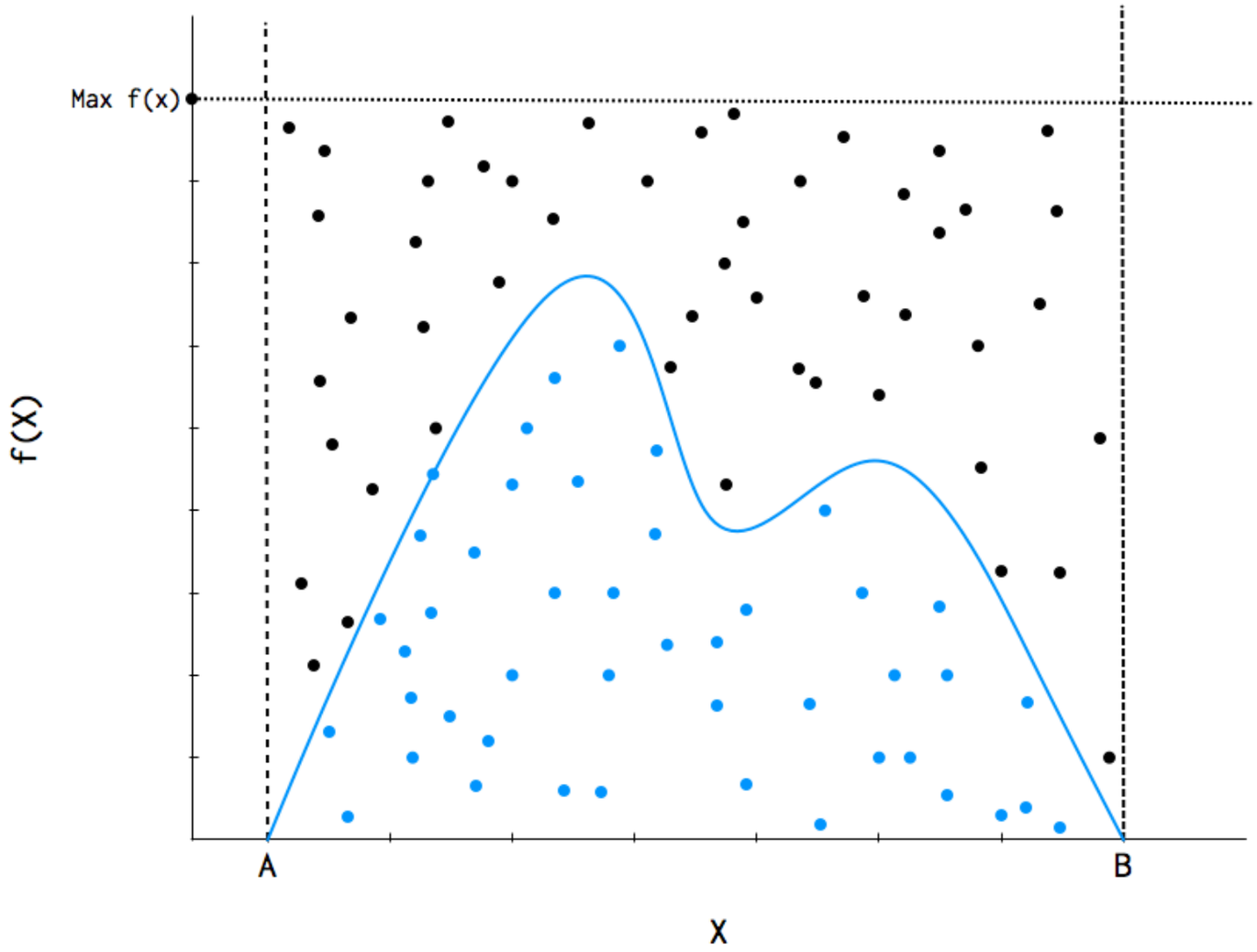
Rejection sampling

- Common in event generators

- Relies on having a good estimate of the distribution global maximum
  - Underestimate = bias
  - Overestimate = inefficiency

- **Curse of dimensionality**
  - Hypercube embedding the distribution grows quickly
  - “The darts miss more often”

- **Mitigation strategies**
  - Dependent distributions
  - Inverse transform sampling
  - Something else?

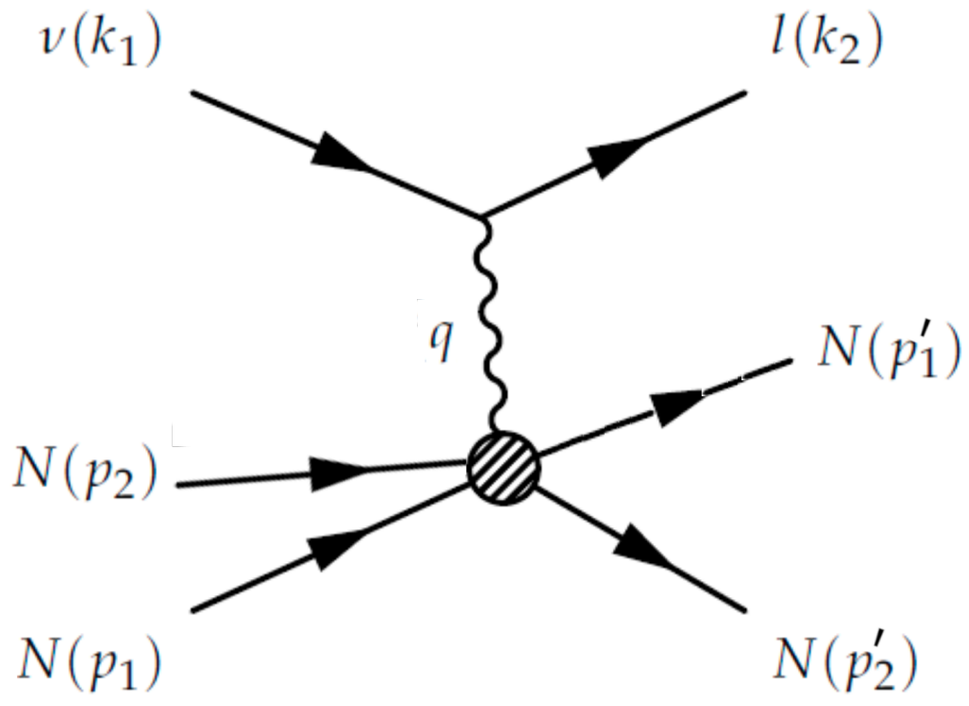


Do we need high-dimensional phase spaces?

- As we strive to incorporate more complete treatments of the physics, this problem seems unavoidable

- An example: exclusive MEC cross sections

- Not all variables may be hard to sample, but a significant fraction likely will be



How many free variables for a full description of MEC at the vertex?

- 6 external lines × 4 momentum components
- Known projectile 4-momentum: **-4**
- On-shell* final particles: **-3**
- 4-momentum conservation: **-4**

Phase space could be as large as 13D!

GENIE v3 example: CCQE

- 6D differential cross section $d\sigma/d^3\mathbf{p}\,dE\,d\Omega^0_\ell$
- **Splits high-dimensional sampling problem into smaller sub-problems**
- Not all variables need to be thrown at once because the differential cross section contains $P(\mathbf{p}, E)$
- Throw $\mathbf{p}, E$ from the nuclear model
  - In practice, $\Omega_\mathbf{p}$ is isotropic and (for a Fermi gas) E is fixed, so only $|\mathbf{p}|$ takes significant work
- Accept/reject lepton angles at fixed $\mathbf{p}, E$ from $\dfrac{1}{P(\mathbf{p}, E)}\dfrac{d\sigma}{d\mathbf{p}\,dE\,d\Omega^0_\ell}$
- If rejected, resample $\mathbf{p}, E$ before trying again (to preserve correlations)

Other solutions

- Interesting **slides** from A. Nikolakopoulos (U. Ghent) at October 2019 **NuSTEC workshop**

  - Throw Q^2 and W, then get pion angles via analytic inverse transform sampling

- Markov Chain Monte Carlo techniques?

  - Handling higher dimensions is easier, but there are drawbacks (burn-in time, autocorrelations)

- **A few questions**
  - Do we have existing strategies in one or more neutrino generators that could be generalized to address high-D sampling?
  - What have collider generator groups tried?
  - What techniques from other fields could be repurposed?

- Perhaps a useful project for inter-generator collaboration

Parameter variations / reweighting

- Experiments need to be able to vary generator models in order to obtain a good assessment of cross section systematic uncertainties

- Ideally this is done by varying free parameters in a theory model with well motivated uncertainties
  - Comparisons between competing models can also be helpful

- In practice, we often have to resort to more ad hoc approaches
  - Multiple experiments have developed tools to reshape the Valencia MEC model
  - Table-based, so no access to underlying model parameters

- Guidance from theorists on appropriate model variations is useful: we could benefit from more of it

- A “theory API” design should consider this need from the beginning
  - **Code:** bidirectional passing of parameters (floating-point, integer, …?)
  - **Tables:** scaling rules or alternate versions to apply variations (tricky)