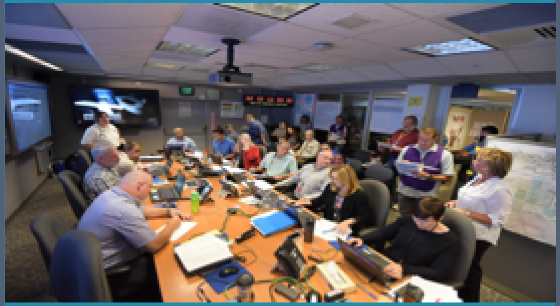
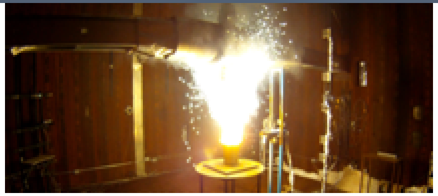
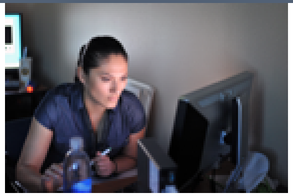
SAND2022-9266C

Neural-Network Based Collision Operators for the Boltzmann Equation







PRESENTED BY

Eric C. Cyr, Nathan V. Roberts, Stephen D. Bond (SNL)

Seam Miller (AMD)

Boltzmann Equation

$$\partial_t f + \mathbf{v} \cdot \nabla f + \mathbf{a} \cdot \nabla_{\mathbf{v}} f = C[f]$$

Particle Density Velocity coordinate Forc
e Collisions

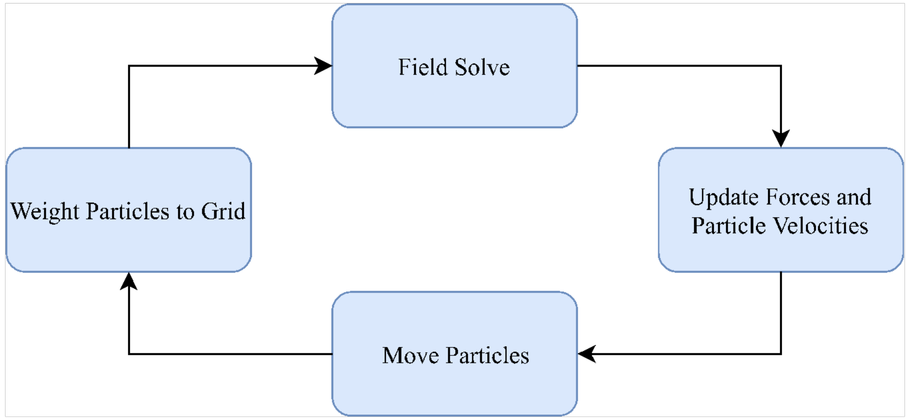
- Defines statistical evolution of system
- Models rarefied gas dynamics:
  - Plasmas (Vlasov)
  - Upper atmosphere fluid dynamics
- Physically modeled in 6D
  - 3D coordinate space (**x**)
  - 3D velocity space (**v**)

Apollo command module

Two Class of Numerical Methods

1. Particle-in-cell (PIC) methods

- Uses coordinate space mesh (3D)
- Particle evolution samples ‘f’
- Collison operator handled using Direct Sim. Monte Carlo (DSMC)
- Fast, with sampling errors



2. “Continuum” simulation

- Uses FD/FEM/FV 6D coord/velocity space mesh
- Collision operator can be expensive
- No sampling, 6D mesh-based methods

Collision Operator

$$C[f] = \iiint d\mathbf{v}_1 \iint dS\, B\left(\|\mathbf{v}-\mathbf{v}_1\|, \theta\right)\left(f\left(\mathbf{x},\mathbf{v}',t\right) f\left(\mathbf{x},\mathbf{v}_1',t\right) - f\left(\mathbf{x},\mathbf{v},t\right) f\left(\mathbf{x},\mathbf{v}_1,t\right)\right)$$

Collision Operator: Convolutional integral over velocity space

- Each point in velocity space “talks” to every other point

In 3D: N^3 points means naïve algorithm scales as N^6

- For hard-sphere collisions FFT methods have been developed scaling like[1]

$N^3 \log(N)$

- Recent work for general methods has developed methods that scale as[2,3]

$N^4 \log(N)$

1. C. Mouhot, L. Pareschi, Fast algorithms for computing the Boltzmann collision operator, Math. of Comp. (2006)
2. I. M. Gamba, J. R. Haack, C. D. Hauck, J. Hu, A fast spectral method for the Boltzmann collision operator with general collision kernels, SISC (2017)
3. S. Jaiswal, A. A. Alexeenko, J. Hu, A discontinuous Galerkin fast spectral method for the full Boltzmann equation with general collision kernels, JCP (2019

Collision Operator: Our Machine Learning Approach

$$C[f] \approx \frac{1}{\tau}(f_M - f) + C_{\text{ML}}[f]$$

Learnable Parameter (τ) · Maxwellian (f_M) · Neural network (C_{ML})

"Learn" perturbation to BGK operator

- BGK is damping around Maxwellian
- Will learn BGK parameter τ
- Perturbation will be a neural network
- Will "teach" the model using cross-sectional data from DSMC

Neural Networks

A neural network is a parameterized model:

Neural Network → $\mathcal{N}\mathcal{N}(x;\Theta) \to y$ ← Output

Input (x), Parameters (Θ)

It is composed of multiple layers:

$$u_1 = A_0 x + b_0,$$

$$u_{i+1} = f(u_i; \{A_i, b_i\}) \quad i = 1 \ldots L-1,$$

$$y = A_L u_L;$$

$$\Theta = \{A_i, b_i\}_{i=0}^{L-1} \cup \{A_L\}$$

Feature Vectors (u_1, u_{i+1}, u_L)

Determining the Parameters

Neural network should map data according to the sampled **training set**:

$$\{(x_i, y_i)\}$$

Input → x_i **Output** → y_i

Find Θ minimizing the **loss** in the model over the **training set**:

Parameters → Θ

$$\min_{\Theta} \sum_{n=1}^{N} \text{Loss}\left(\mathcal{NN}(x_n; \Theta), y_n\right)$$

Loss function is model/data difference:

- $\text{Loss}(y^{model}, y^{data}) = \|y^{model} - y^{data}\|^2$
- $\text{Loss}(\vec{y}^{\,model}, \vec{y}^{\,data}) = \sum_{c=1}^{N_c} y_c^{data} \log\left(y_c^{model}\right)$

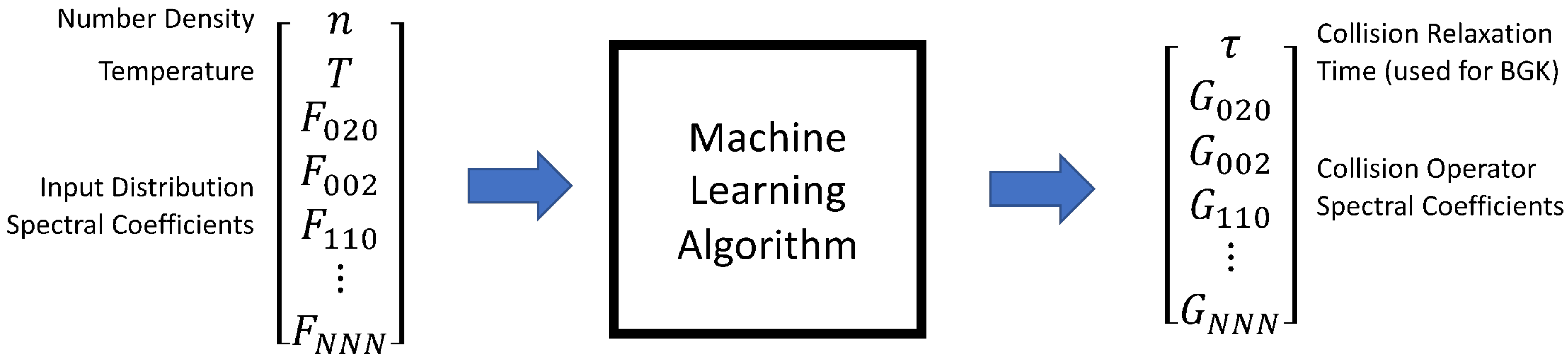
Feature Selection

Neural network inputs and outputs written in spectral form over velocity space

Input Distribution

$$f = f_0 \exp(-\boldsymbol{w}^2) \sum_{ijk} F_{ijk} H_i(w_x) H_j(w_y) H_k(w_z)$$

Weighting Function; Spectral Coefficients; Scaled Random Velocity Coordinate; Normalization Parameter; Hermite Polynomials



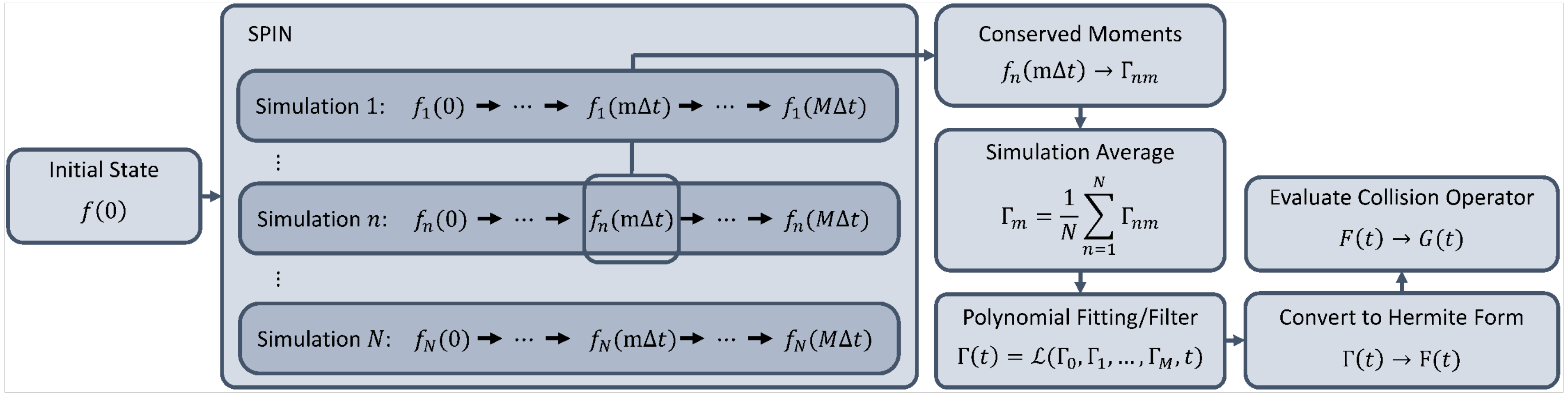
Spectral representation is designed and normalized to enforce the conservation of mass, momentum, and energy.

Training and testing data is generated from "converged" DSMC data.

Training Data

Using a DSMC code:

1. Generate initial particle distribution
2. Run DSMC simulation with desired collisional kernel
3. Extract data pairs input (density distribution) to output (collisional response)



Comments:

- Leverages existing cross-section tables
- Expensive offline phase for generating data
- Accuracy of trained model will be limited to range of training data
  (neural networks are not magic, extrapolation is still ~~hard~~ ill-posed)

Training Loss

$$\sqrt{\frac{1}{N_{\text{samples}}} \sum_{\text{samples}} (\tau - \bar{\tau})^2} \quad + \quad \frac{\sum_{\text{samples}} \sum_{ijk} |g_{ijk} - \bar{g}_{ijk}| \, \Omega_{ijk}}{\sum_{\text{samples}} \sum_{ijk} \left| \bar{g}_{ijk} + \frac{\delta_{ijk}}{\gamma_{ijk}} \right| \Omega_{ijk}}$$

Relaxation Time Loss

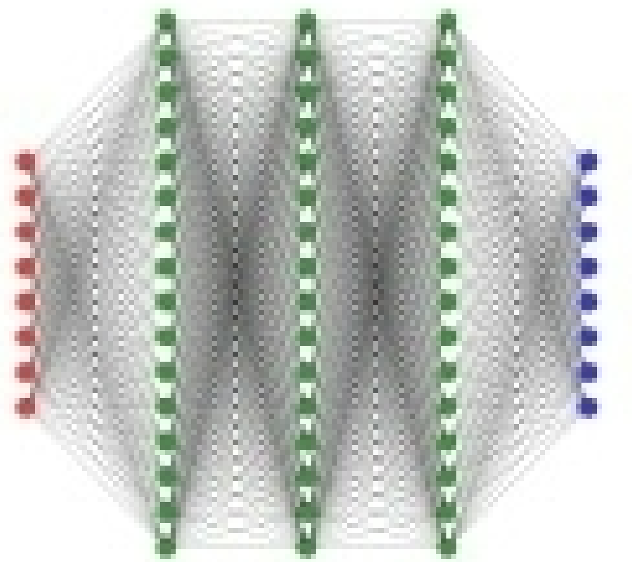
Weighted Error for Hermite Coefficients

$$\Omega_{ijk} = \iiint_{-\infty}^{\infty} \left| \gamma_{ijk} H_i(w_x) H_j(w_y) H_k(w_z) \exp(-\mathbf{w}^2) \right| \, d^3 w$$

Network Architectures

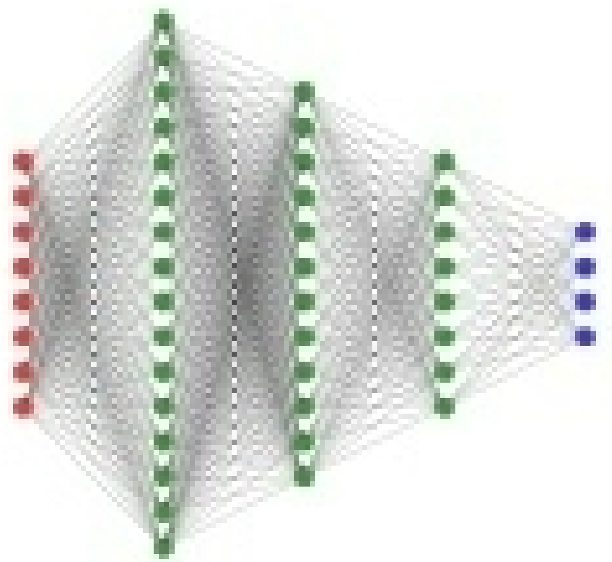
Vanilla neural networks with sigmoid activations, two flavors:

$$\mathcal{N}_{3,2}(8,8)$$



Input/Output Spectral space is the same

$$\mathcal{N}_{3,2}(8,4)$$



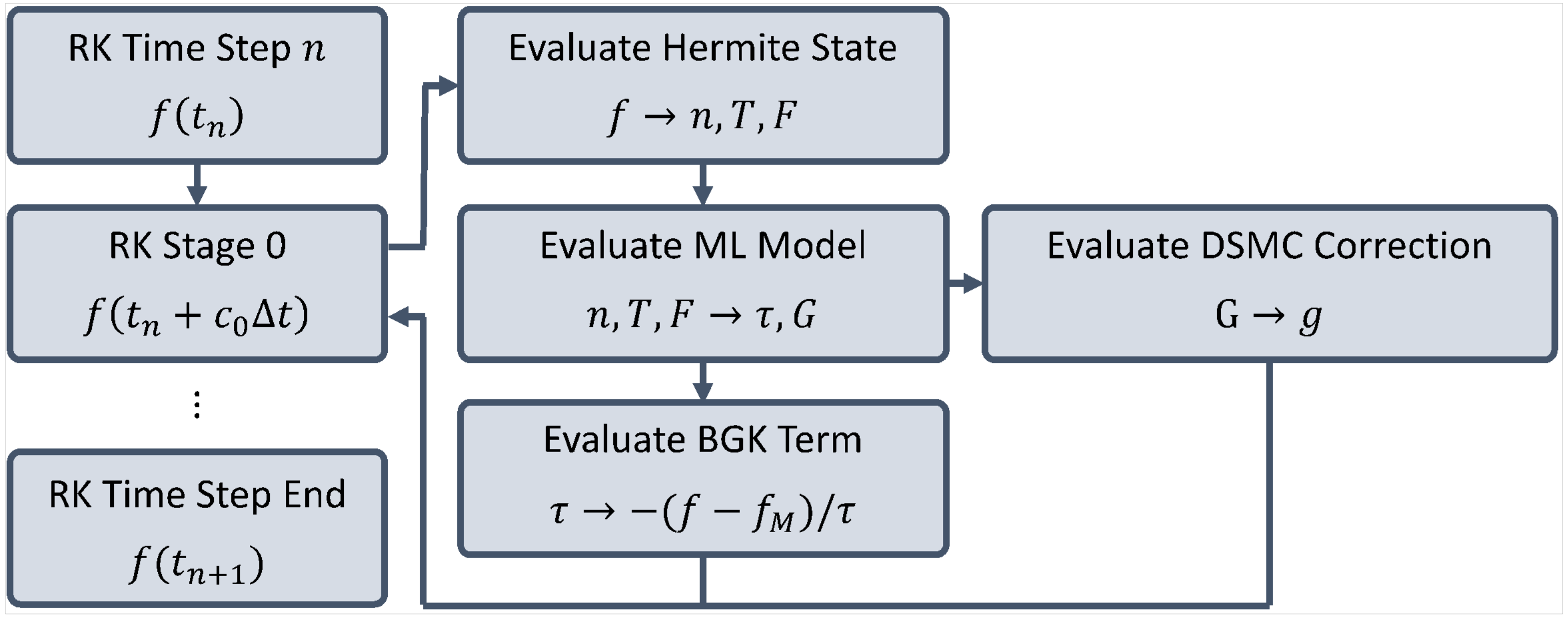
"Truncated": Output spectral space is smaller

Width Mult.   In size

$$\mathcal{N}_{l,w}(i,o)$$

Hidden layers   Out size

Applying the network in a Boltzmann code



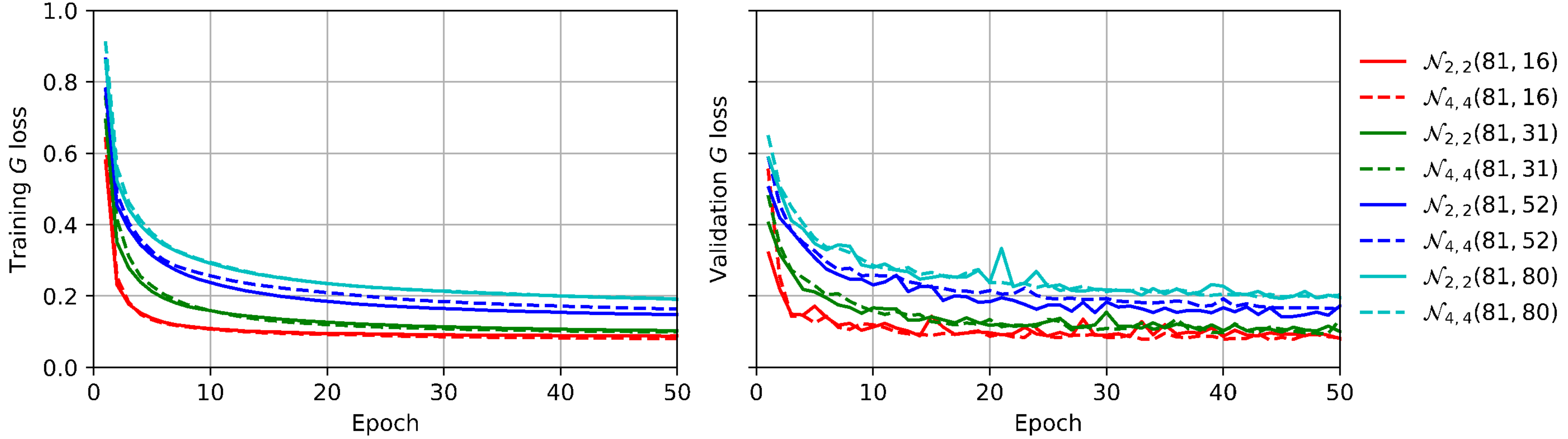
Run time (3D):

$$N_{\text{coeff}}^2 + (2N_{\text{coeff}} + 1)N^3$$

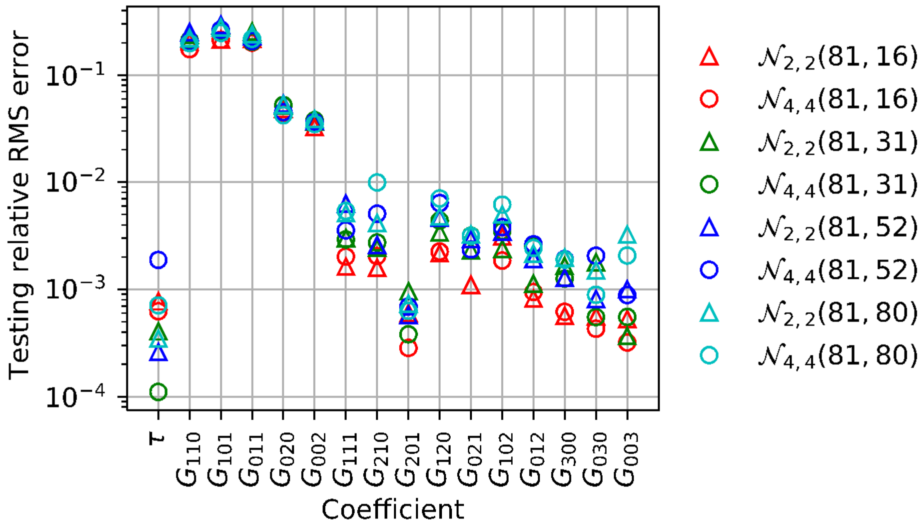
Neural Network Evaluation — **Hermite Coeff Calculation** — **BGK Evaluation**

Training and Testing

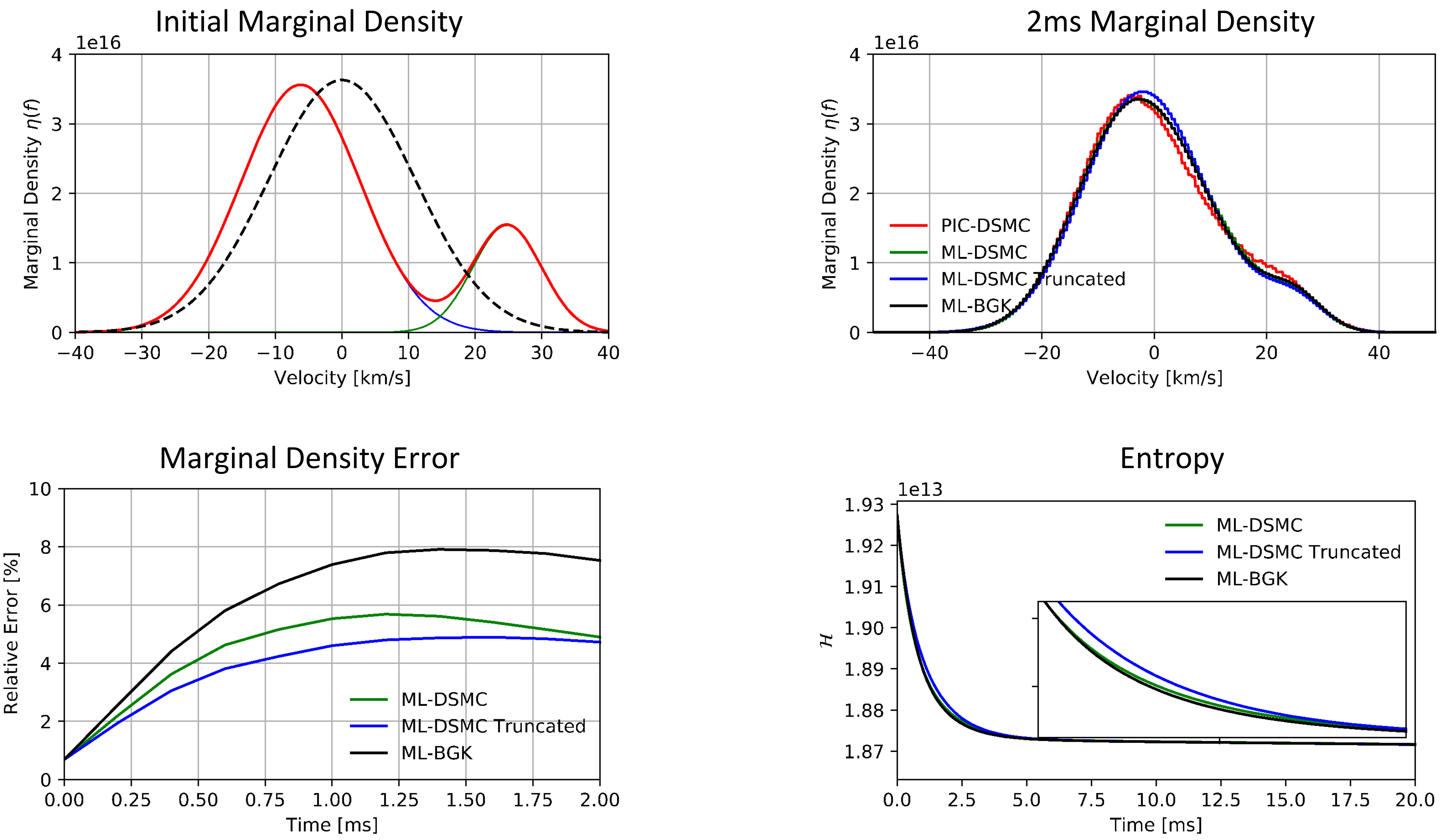
Using an Argon Maxwell-molecule



Neural network trains well, no sign of overfitting, error is reasonable on test set



Bump on Tail



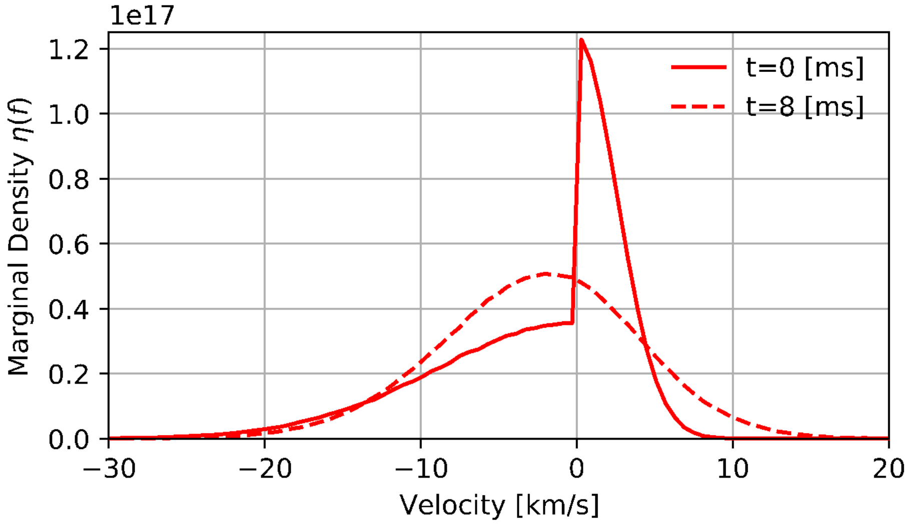
Initial Marginal Density
1e16
Marginal Density $\eta(f)$
Velocity [km/s]
2ms Marginal Density
PIC-DSMC
ML-DSMC
ML-DSMC Truncated
ML-BGK
Marginal Density Error
Relative Error [%]
Time [ms]
Entropy
1e13
$\mathcal{H}$

Double Half Normal

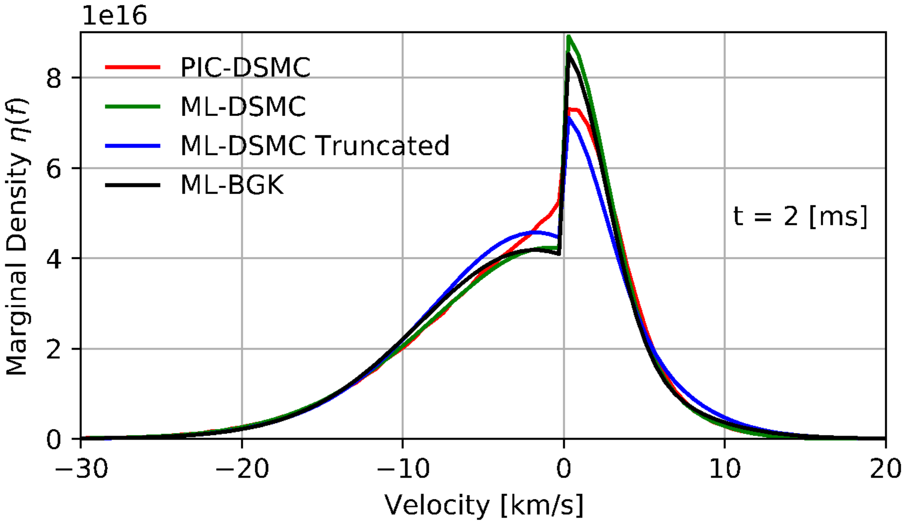
To test behavior in "novel" regimes (an extrapolation test):

- Is machine learned model accurate? No ☹
- Is machine learned model stable? Yes ☺

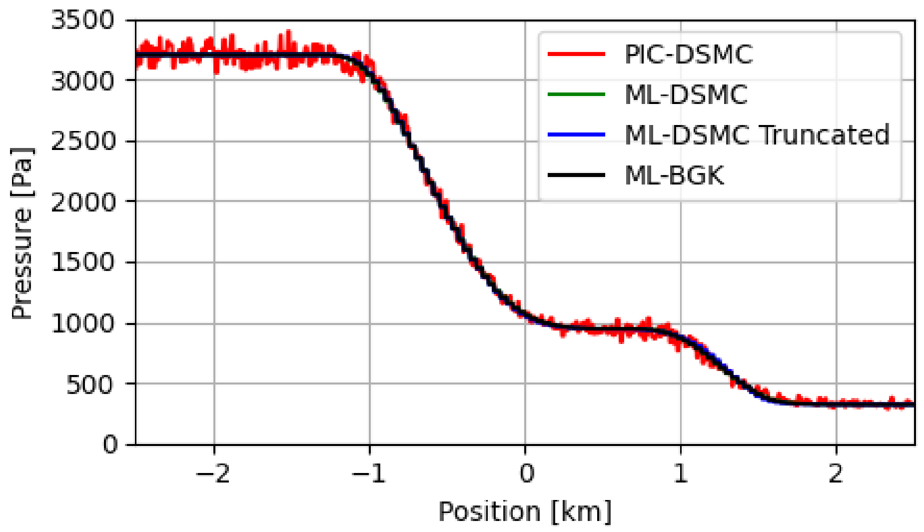
Marginal Density

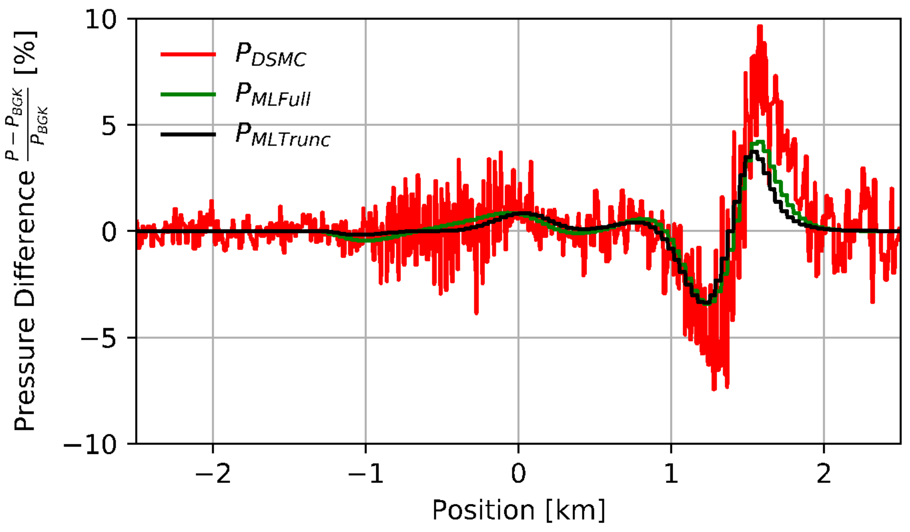


2ms Marginal Density



Collisional Shock-Tube





Moderately collisional regime shows formation of shock front

- Good agreement in comparison to PIC (within noise)
- Difference to BGK, shows correction can be significant

Final Thoughts

Presented a machine learned model for evaluating collisions

- Useful in the context of a continuum Boltzmann solver
  - Compares well to PIC simulation (shock tube)
  - Stable in extrapolation regime (double half normal)
  - Transient accuracy (bump on tail)
- Theoretical performance in line with other spectral approaches
- Model generated directly from a DSMC code

Presentation based on the Paper:

S. T. Miller, N. V. Roberts, E. C. Cyr, Neural-Network Based Collision Operators for the Boltzmann Equation, under review at JCP, 2021.

U.S. DEPARTMENT OF ENERGY | Office of Science

Thanks to the DOE Office of Science ASCR Early Career Research Program for supporting this work!