UNIVERSITY OF OXFORD

Building high fidelity emulators with **D**eep **E**mulator **N**etwork **SE**arch

Muhammad F. Kasim

Oxford Physics

29 April 2020

Acknowledgements

D. Watson-Parris
L. Deaconu
S. Oliver
S. Khatiwala
P. Hatfield

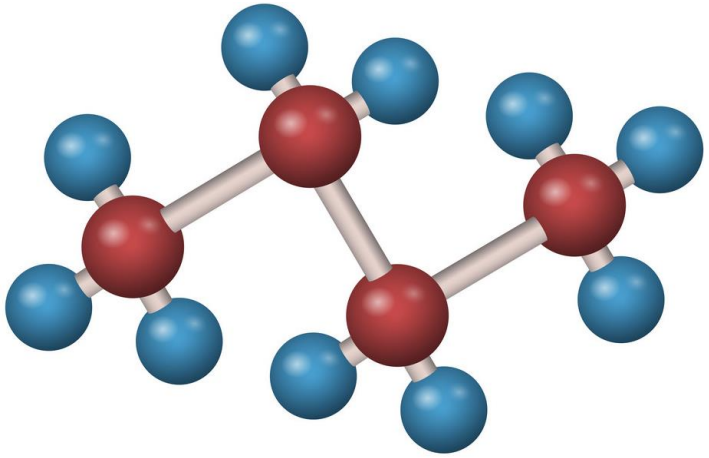
G. Gregori
M. Jarvis
J. Topp-Mugglestone
S. M. Vinko

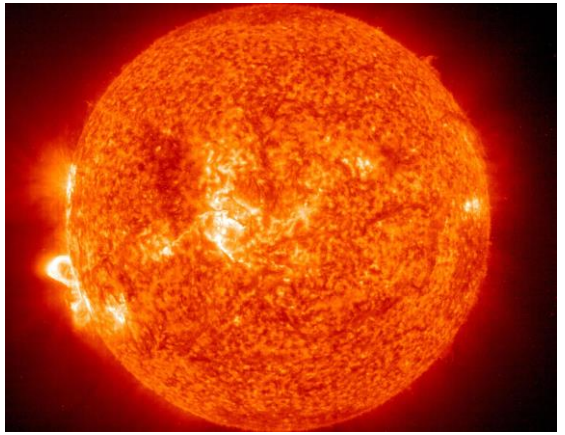
D. H. Froula

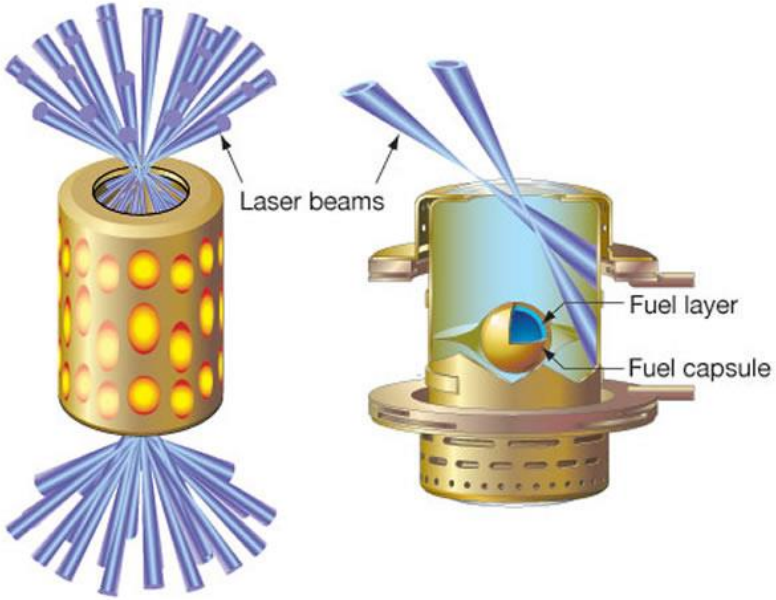
E. Viezzer

J. Korenaga

Computer simulations in research



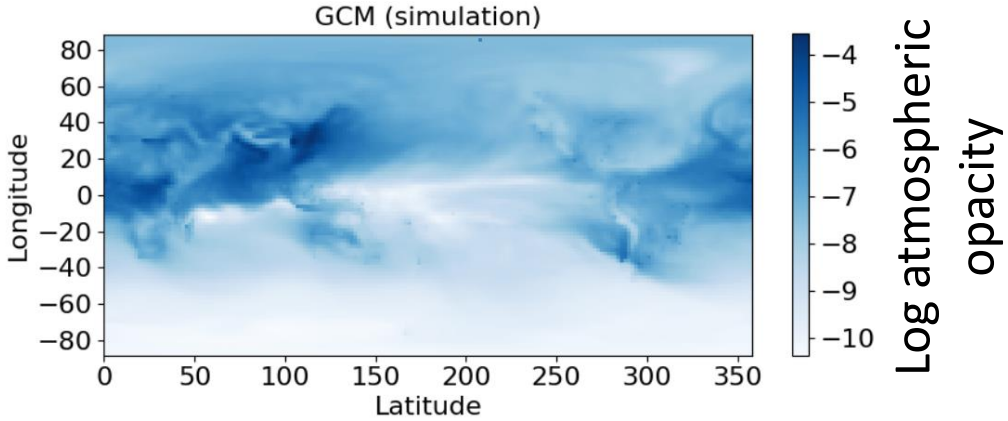




Example: Parameterized Global Climate Modelling

Human-activity parameters

Simulation



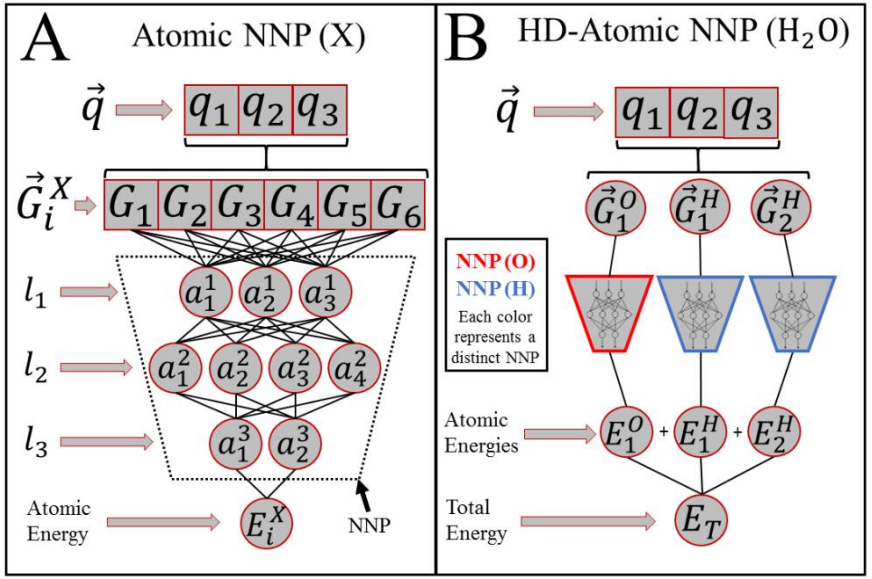
1 simulation: **1000 core-hours**

Hundreds-thousands of simulation to find a fit: **almost impossible**

Need an emulator to speed it up!

Building fast emulator with machine learning

- Computing molecular energy with neural network



Smith *et al.*, https://arxiv.org/abs/1610.08935

- Generating power spectra from cosmological parameters (cosmic emu)

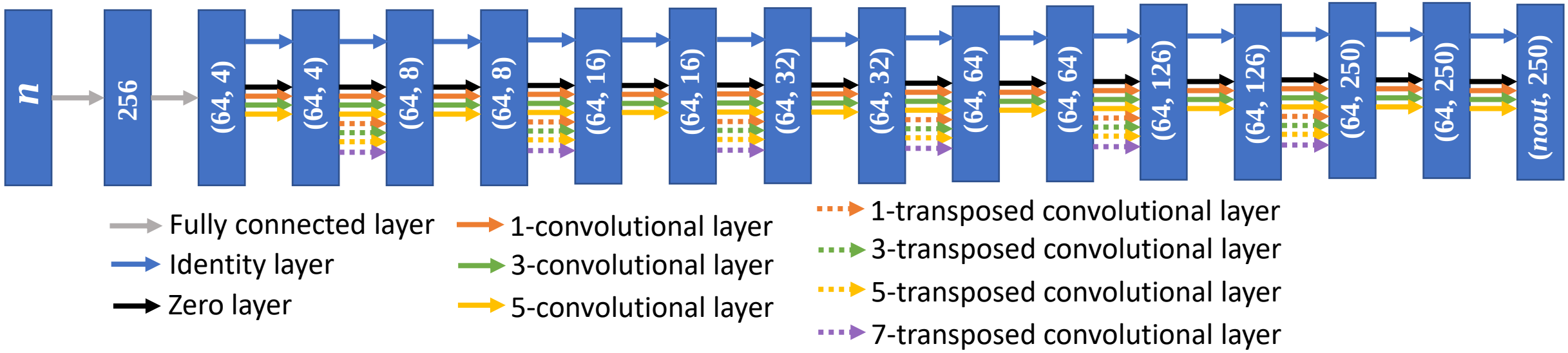
Kwan *et al.*, https://arxiv.org/abs/1311.6444

Challenges in building emulators

- Accuracy must be high!
- Slow simulations: can only generate ~1,000 – 10,000 datasets
- Most of the previous work only predict scalar outputs
- If only a few dataset is available, must avoid overfitting

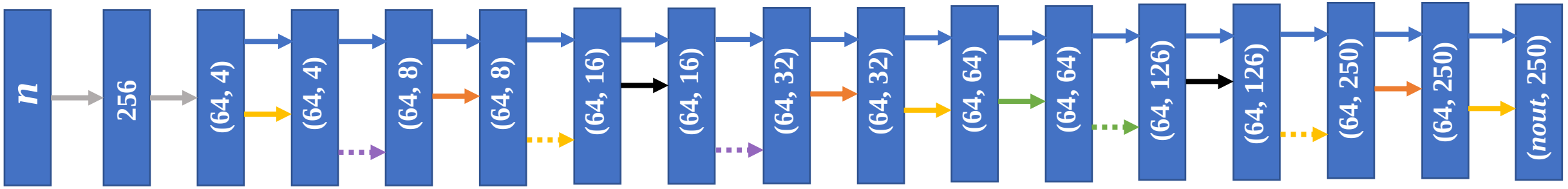
Idea

- Use **deep neural architecture search** to simultaneously train & find the right architecture of deep neural network



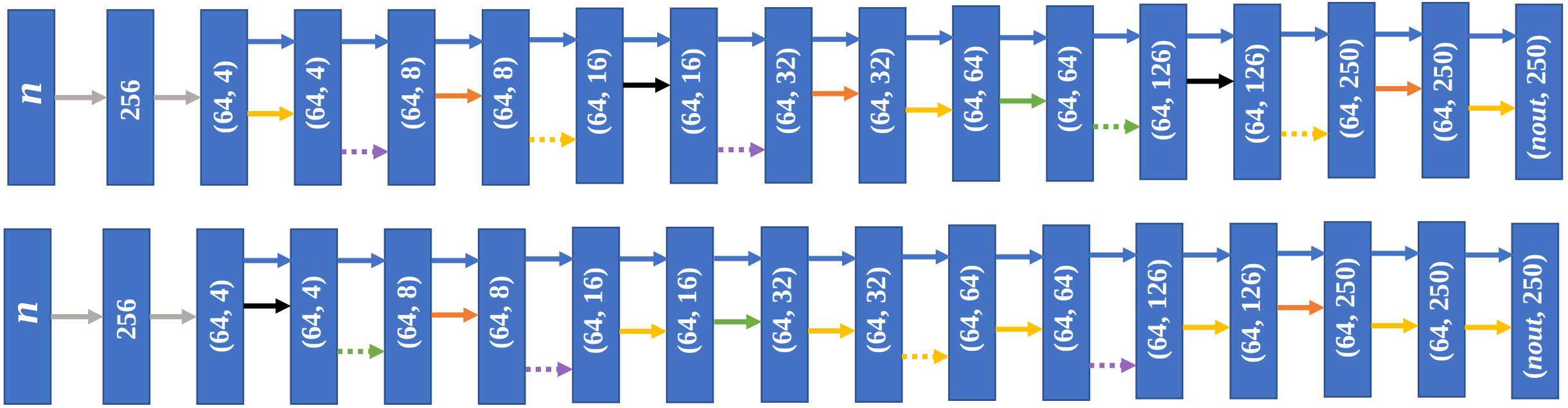
Two update steps learning

- Step 1: update the weights with supervised learning



Two update steps learning

- Step 2: update the probability with reinforcement learning



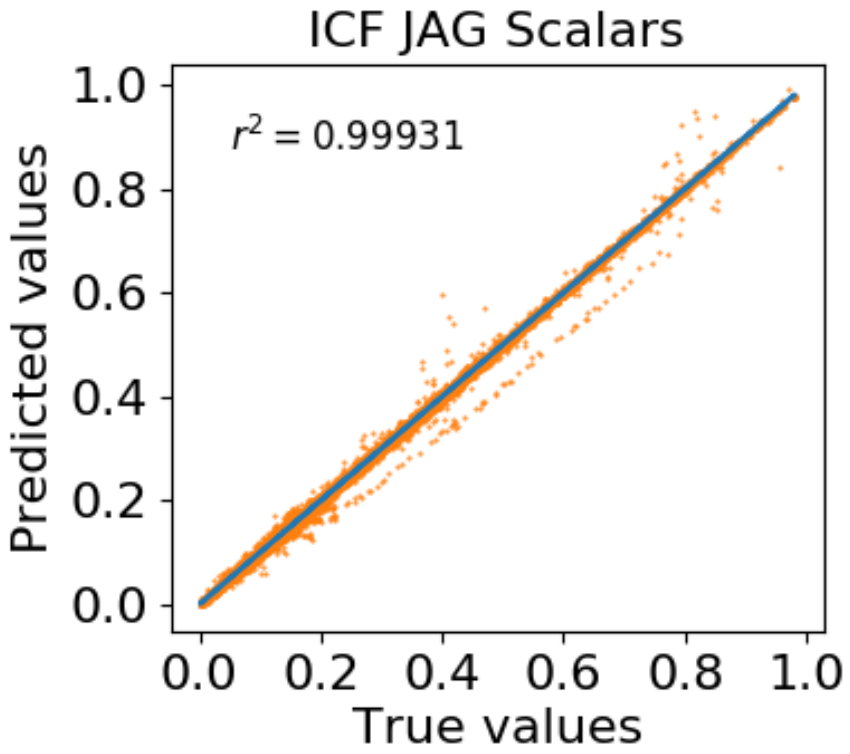
Test cases

No.	Test case	# Inputs	# Outputs	Output type	# Dataset	Est. running time
1	XRTS	3	1	1D (250 points)	14,000	15 seconds
2	OTS	5	1	1D (250 points)	14,000	15 seconds
3	XES	10	1	1D (250 points)	14,000	20 minutes
4	ELMs	14	10	1D (250 points)	14,000	15 minutes
5	Halo	5	1	1D (250 points)	14,000	5 seconds
6	ICF JAG	5	4	2D (64×64)	10,000	30 seconds
7	ICF JAG Scalars	5	15	0D (scalar)	10,000	30 seconds
8	SeisTomo	13	1	1D (250 points)	6,100	2 hours
9	MOPS	6	45	2D (128×64)	410	144 CPU-hours
10	GCM	3	12	2D (192×96)	39	1150 CPU-hours

50% for training
21% for validation
28% for test

Emulator results

Inertial Confinement Fusion model (ICF JAG): 5 input parameters & 15 scalar outputs

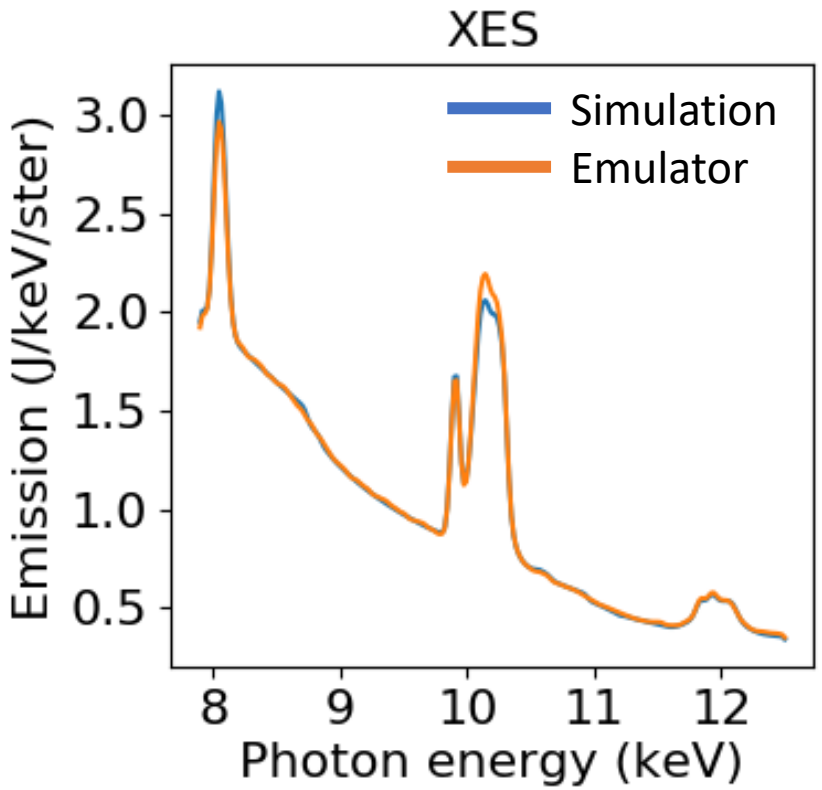


Simulation time:
- 30 seconds

Emulator time:
- 2.31 ms / 1024

Emulator results

X-ray Emission Spectroscopy (XES): 10 parameters & 1 one-dimensional output

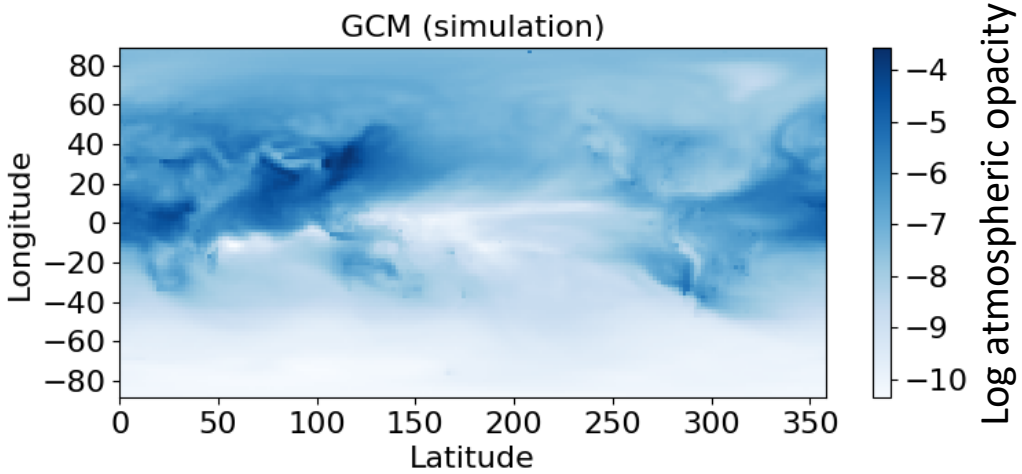


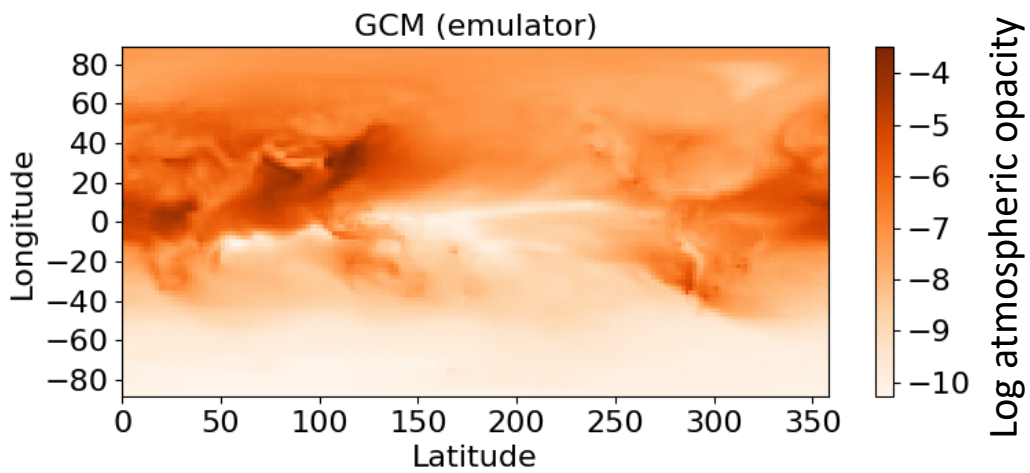
Simulation time:
- 20 minutes

Emulator time:
- 5.73 ms / 256

Emulator results

Parameterized Global Climate Model (GCM): 3 input parameters & 12 two-dimensional outputs

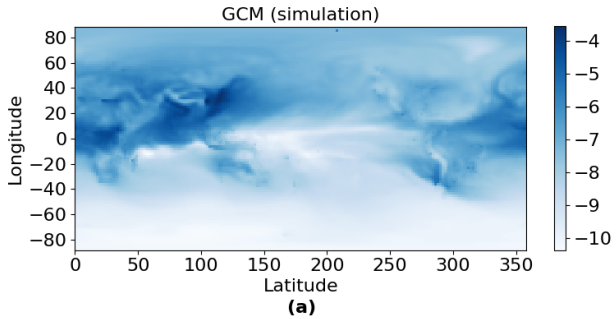


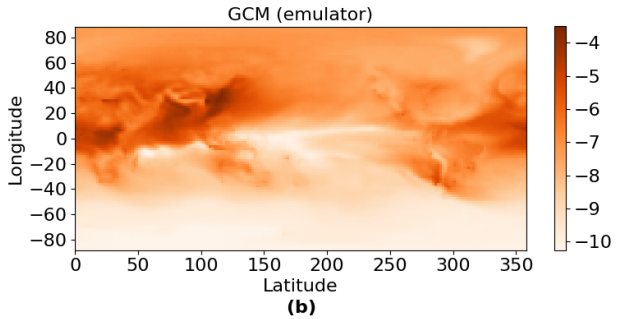


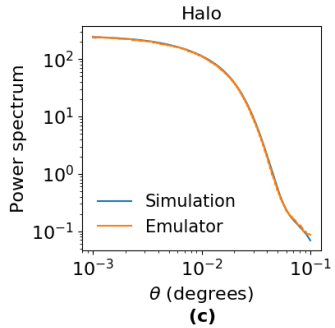
Simulation time:
1150 CPU-hours

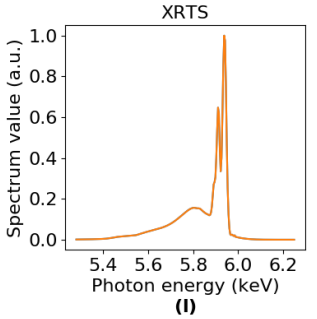
Emulator time:
114 GPU-milliseconds / 64

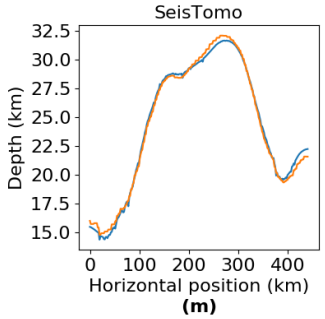
Emulator results

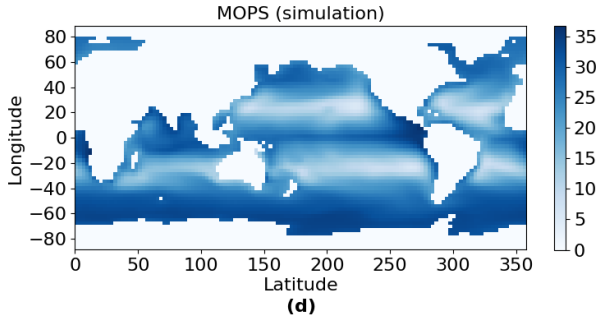


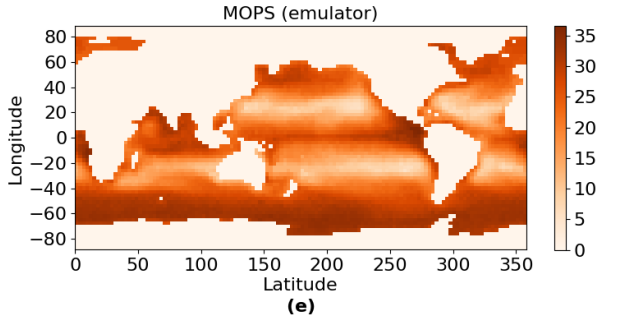


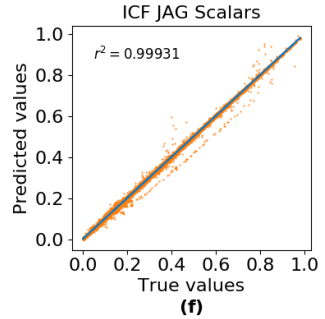






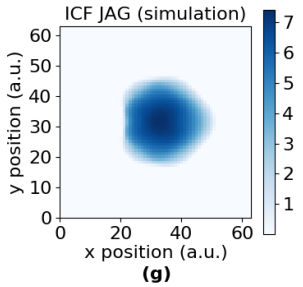


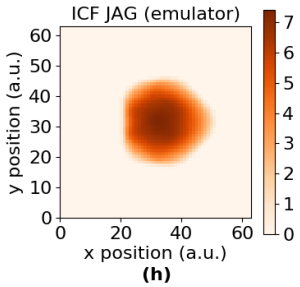


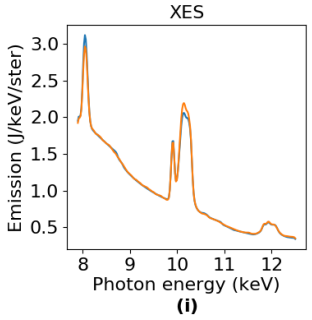


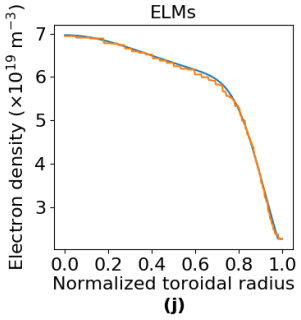
Simulations

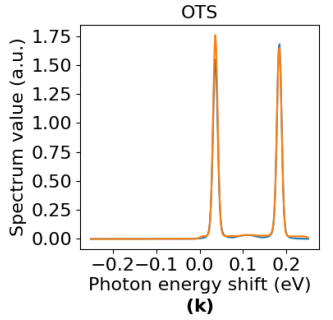
Emulators





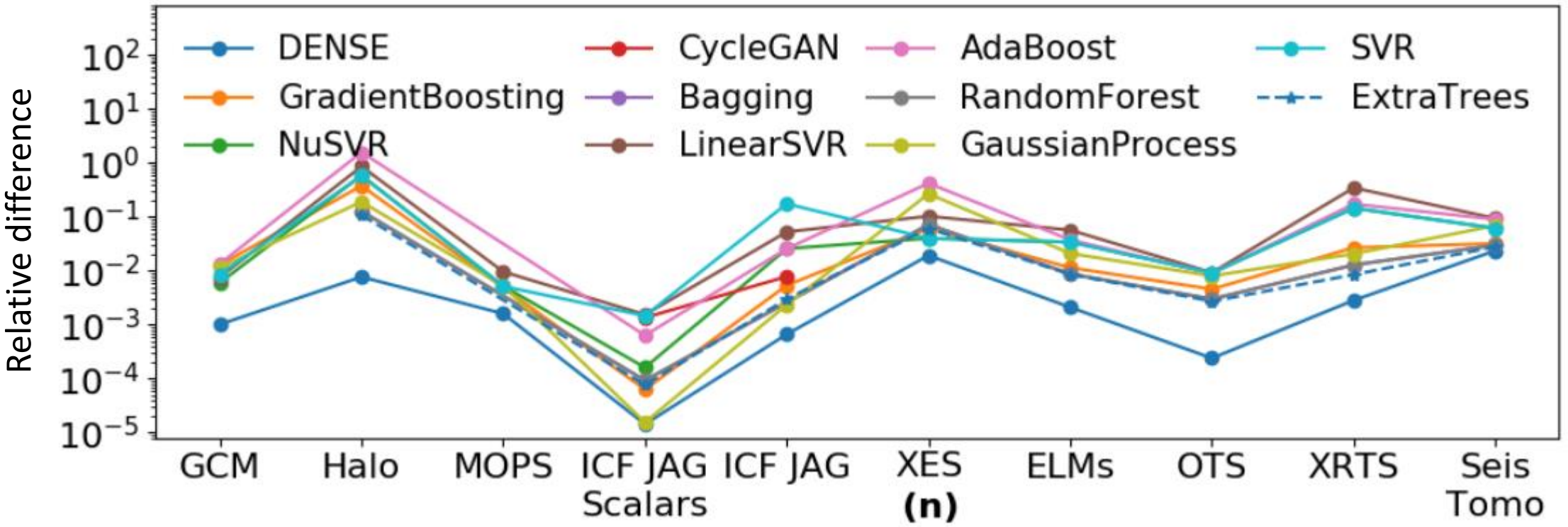




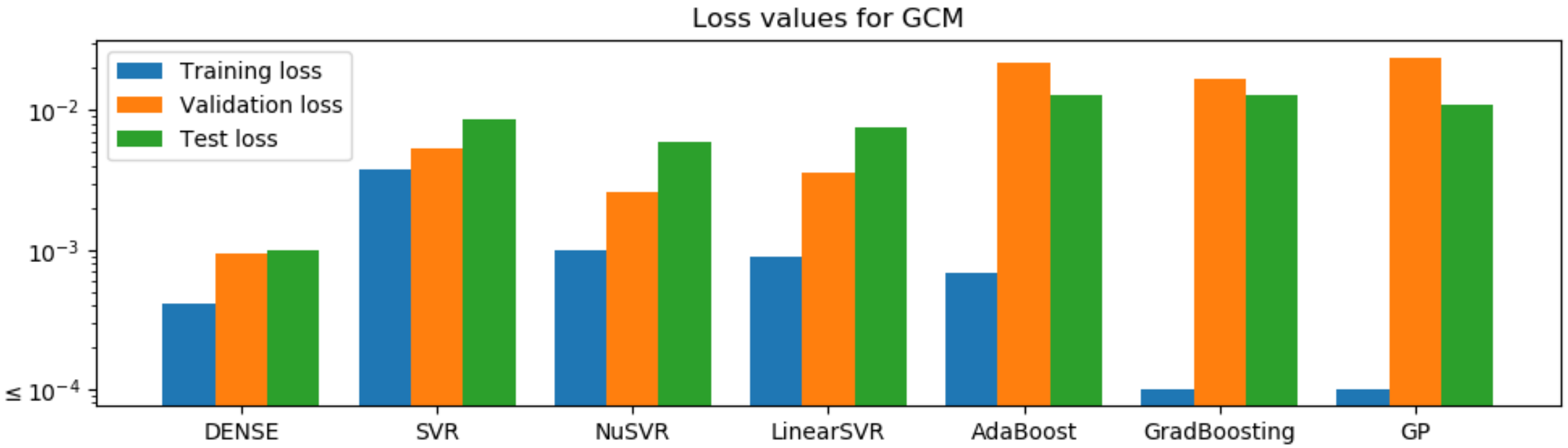


Quality comparison with other ML models

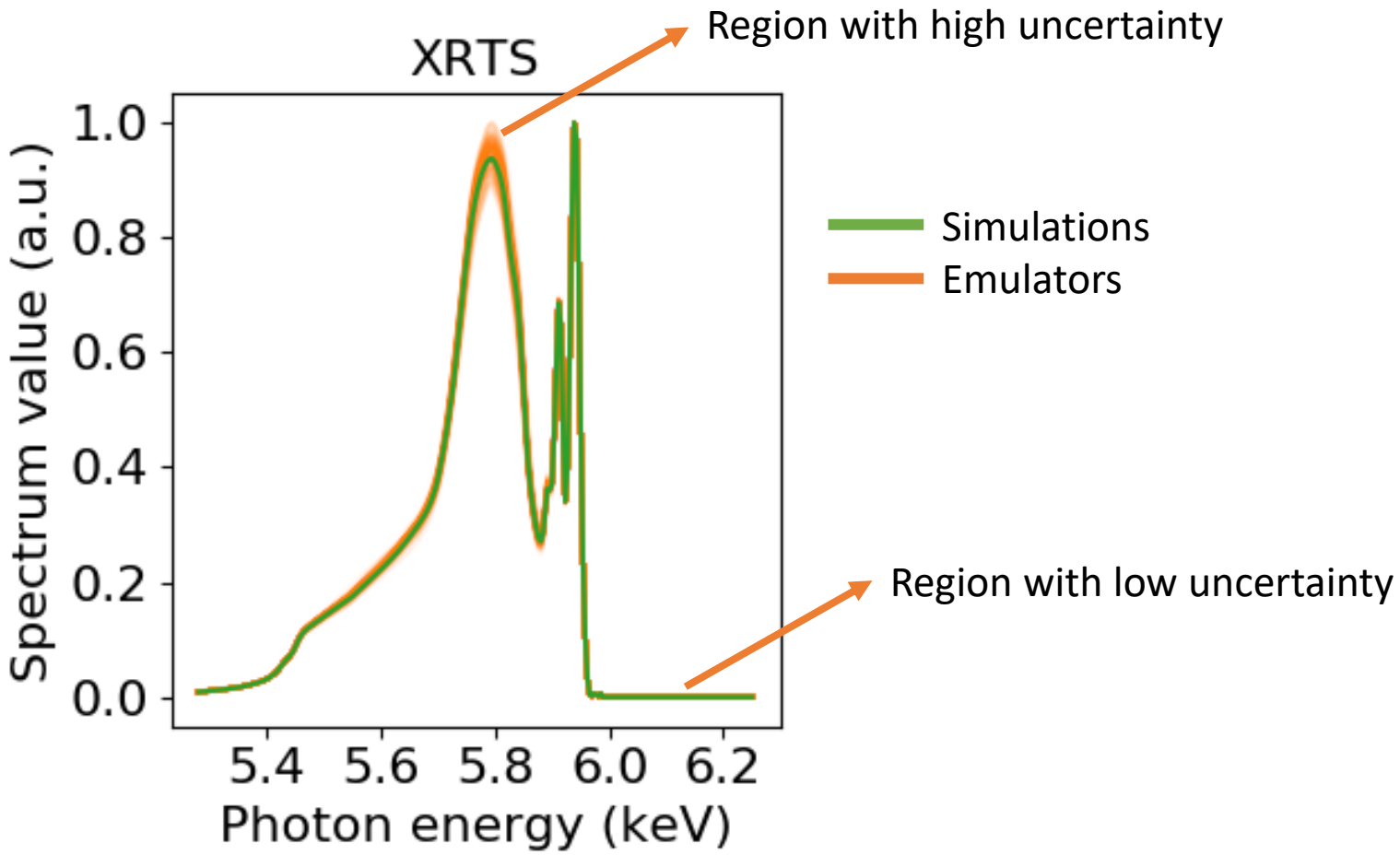
Difference between simulator outputs and model outputs (lower is better)



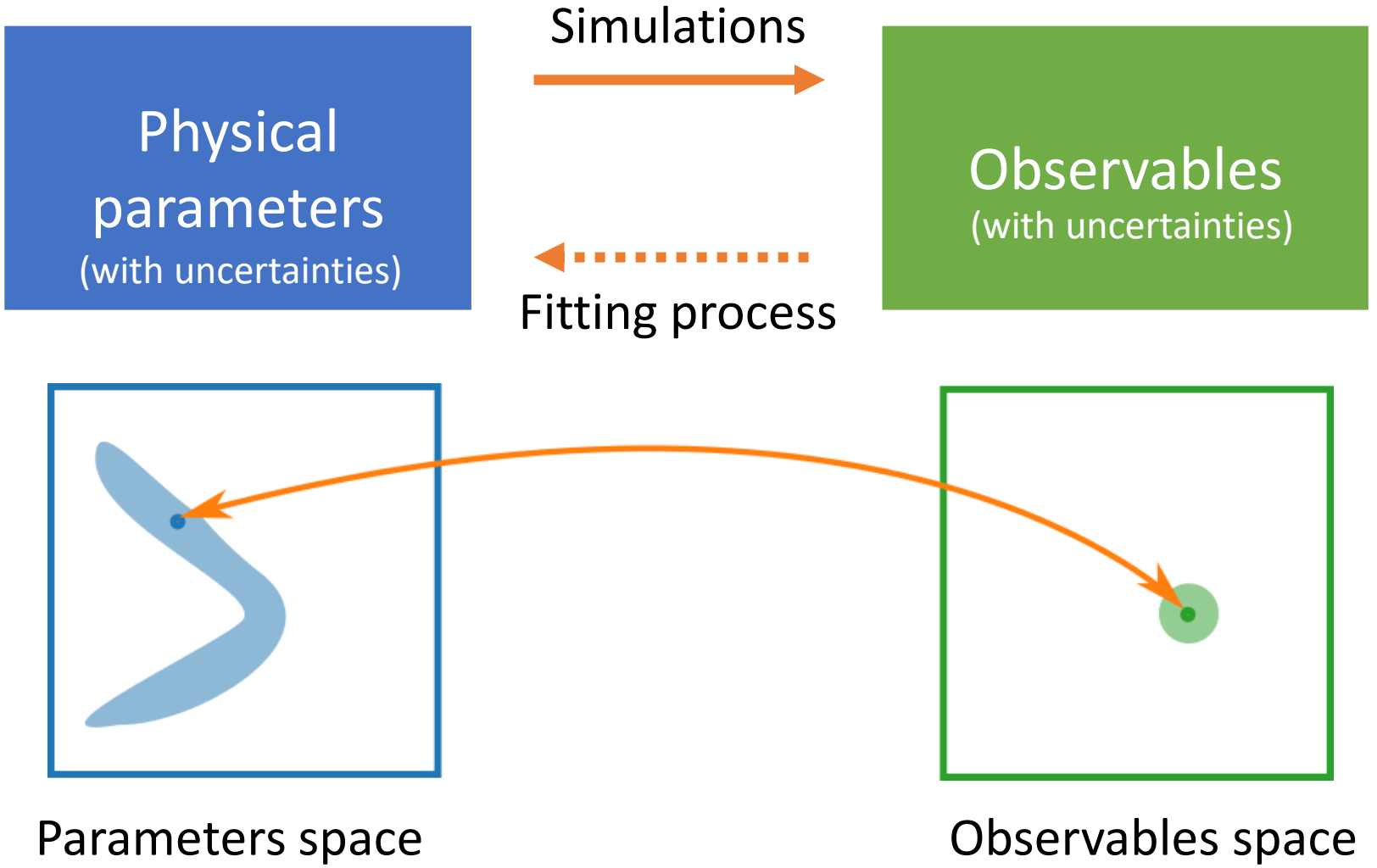
Minimal overfit on GCM



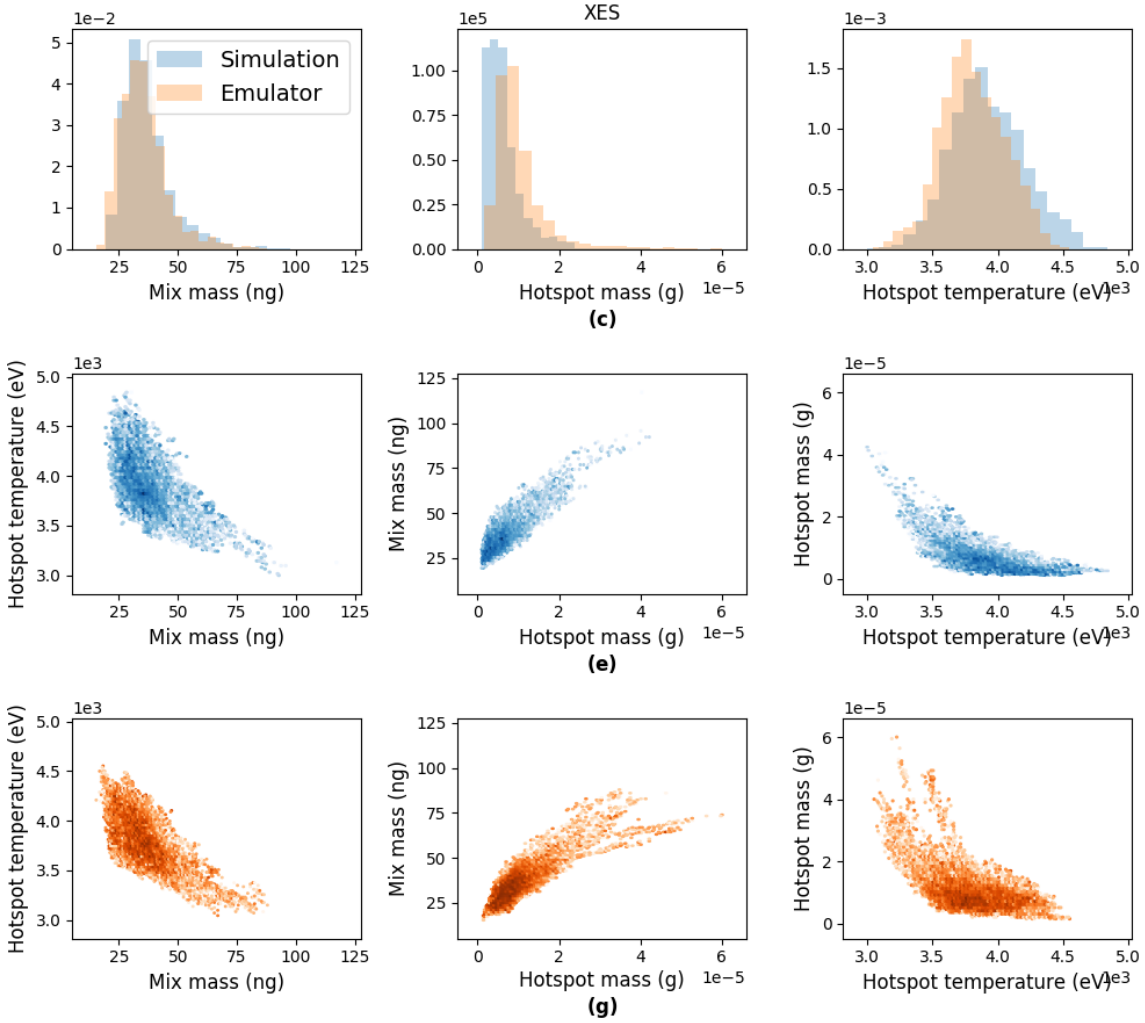
DENSE provides uncertainty in its prediction

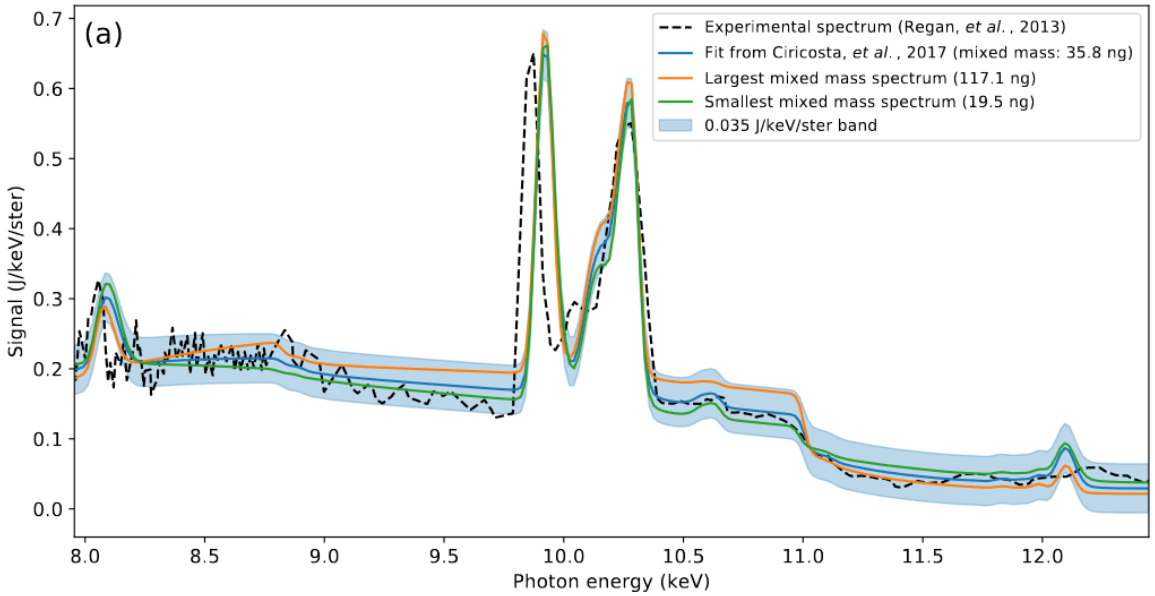


Uncertainty quantification



Usage in uncertainty quantification





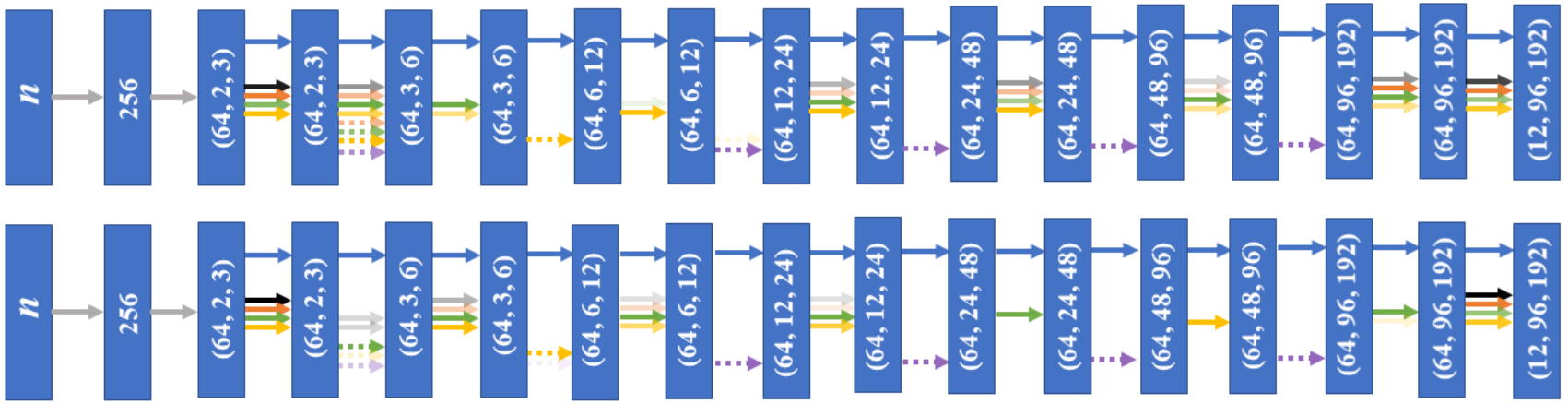
Conclusion

- DENSE works well on 10 simulation test cases in learning the input-to-output mapping.
- The accuracy achieved by DENSE is significantly higher than other non-deep learning technique in all but one case.
- Uncertainty quantification can be done with DENSE in a few seconds, as opposed to days-months with expensive simulations.
- DENSE could learn well even with limited training data.

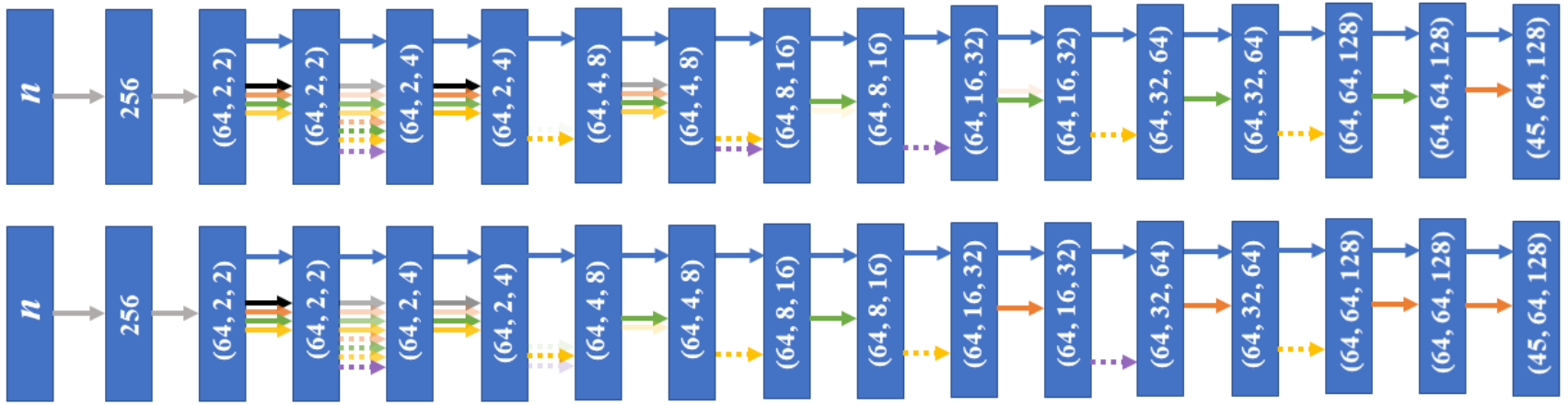
Limitations

- Does not seem to learn well in regions of high variability
- DENSE currently works for simulations with $<\sim 15-20$ scalar inputs

Appendix



(a) Architectures for GCM



(b) Architectures for MOPS